

# character studio<sup>®</sup>

VERSION THREE

*[www.discreet.com](http://www.discreet.com)*

3

REFERENCE AND TUTORIALS

# Copyright © 2001 Autodesk, Inc.

All Rights Reserved

This publication, or parts thereof, may not be reproduced in any form, by any method, for any purpose.

**AUTODESK, INC. MAKES NO WARRANTY, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, REGARDING THESE MATERIALS AND MAKES SUCH MATERIALS AVAILABLE SOLELY ON AN "AS-IS" BASIS.**

**IN NO EVENT SHALL AUTODESK, INC. BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH OR ARISING OUT OF PURCHASE OR USE OF THESE MATERIALS. THE SOLE AND EXCLUSIVE LIABILITY TO AUTODESK, INC., REGARDLESS OF THE FORM OF ACTION, SHALL NOT EXCEED THE PURCHASE PRICE OF THE MATERIALS DESCRIBED HEREIN.**

Autodesk, Inc. reserves the right to revise and improve its products as it sees fit. This publication describes the state of this product at the time of its publication, and may not reflect the product at all times in the future.

## Autodesk Trademarks

The following are trademarks of Autodesk, Inc., in the USA and/or other countries: 3D on the PC, ACAD, Advanced User Interface, AEC Office, AME Link, Animation Partner, Animation Player, Animation Pro Player, A Studio in Every Computer, ATLAST, Auto-Architect, AutoCAD Architectural Desktop, AutoCAD Architectural Desktop Learning Assistance, AutoCAD Learning Assistance, AutoCAD LT Learning Assistance, AutoCAD Simulator, AutoCAD SQL Extension, AutoCAD SQL Interface, Autodesk Animator Clips, Autodesk Animator Theatre, Autodesk Device Interface, Autodesk Inventor, Autodesk PhotoEDIT, Autodesk Point A (logo), Autodesk Software Developer's Kit, Autodesk View DwgX, AutoFlix, AutoPAD, AutoSnap, AutoTrack, Built with ObjectARX (logo), ClearScale, Colour Warper, Combustion, Concept Studio, Content Explorer, cornerStone Toolkit, Dancing Baby (image), Design 2000 (logo), DesignCenter, Design Doctor, Designer's Toolkit, DesignProf, DesignServer, Design Your World, Design Your World (logo), Discreet, DWG Linking, DWG Unplugged, DXF, Extending the Design Team, FLI, FLIC, GDX Driver, Generic 3D, Heads-up Design, Home Series, iDesign, i-drop, Kinetix (logo), Lightscape, ObjectDBX, onscreen onair online, Ooga-Chaka, Photo Landscape, Photoscape, Plugs and Sockets, PolarSnap, Pro Landscape, QuickCAD, Real-Time Roto, Render Queue, SchoolBox, Simply Smarter Diagramming, SketchTools, Suddenly Everything Clicks, Supportdesk, The Dancing Baby, Transform Ideas Into Reality, Visual LISP, Visual Syllabus, VIZable, Volo, Where Design Connects, and Whereware.

## Third-Party Trademarks

All other brand names, product names, or trademarks belong to their respective holders.

## Third-Party Software Program Credits

Character Studio™ software is produced exclusively for discreet, a division of Autodesk, Inc., by Unreal Pictures, Inc. Copyright 1997, 1998, 2000 Unreal Pictures, Inc.

InstallShield™ Copyrighted © 1998, 2000 InstallShield Software Corporation. All rights reserved.

Portions Copyrighted © 1998, 2000 Microsoft Corporation. All rights reserved.

## GOVERNMENT USE

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in FAR 12.212 (Commercial Computer Software-Restricted Rights) and DFAR 227.7202 (Rights in Technical Data and Computer Software), as applicable.

Autodesk, Inc. reserves the right to revise and improve its products as it sees fit. This publication describes the state of this product at the time of its publication, and may not reflect the product at all times in the future.



---

# Contents

<b>Welcome to character studio 3 . . . . .</b>	<b>.xix</b>
What's New in character studio 3 . . . . .	.xix
Nonlinear Animation . . . . .	xx
Track Operations . . . . .	xx
Behavioral Crowd Animation . . . . .	xx
Inverse Kinematic Pivots . . . . .	xxii
Keyframe Animation Workflow . . . . .	xxiii
Footstep Animation Improvements . . . . .	xxiii
Figure Mode Improvements . . . . .	xxiv
Motion Capture Editing . . . . .	xxiv
Skin Deformation . . . . .	xxiv
Programmatic Access . . . . .	xxiv
Product Information . . . . .	xxv
System Requirements . . . . .	xxv
Optional . . . . .	xxvi
Installing character studio 3 . . . . .	xxvi
Authorizing character studio . . . . .	xxvi
<b>Chapter 1 Using character studio . . . . .</b>	<b>1</b>
Introduction to character studio . . . . .	1
What You Should Know to Use character studio . . . . .	2
Biped Features and Benefits . . . . .	2
Rapid Creation of Biped Skeletons . . . . .	2
Freeform Animation . . . . .	3
Footstep-Driven Animation . . . . .	3
Advanced Inverse Kinematics . . . . .	4
Motion Retargeting Between Characters . . . . .	4
Motion Capture Operations . . . . .	4
Understanding Biped . . . . .	5
The Biped . . . . .	5
Biped Center of Mass . . . . .	5
Keyframing the Biped . . . . .	6
Footstep Method . . . . .	6

Freeform Method . . . . .	7
Inverse Kinematics . . . . .	7
Key Interpolation . . . . .	8
Productivity . . . . .	8
Physique Features . . . . .	9
Skin Attachment with Envelopes . . . . .	9
Weighted Blending . . . . .	9
Skin Sliding . . . . .	10
Muscle Bulging . . . . .	10
Tendon Effects . . . . .	10
Support for Patches NURBS and Polygons . . . . .	10
Using Splines with Physique . . . . .	10
Using Non-Hierarchical Bone Structures with Physique . . . . .	11
Performance . . . . .	11
Understanding Physique . . . . .	11
Biped and Physique . . . . .	12
Envelopes and Weighted Vertices . . . . .	12
Deformable and Rigid Envelopes . . . . .	12
The Number of Links that can affect a Vertex . . . . .	13
Physique Workflow . . . . .	13
Overview of Crowd Features . . . . .	13
Basic Setup with Crowd and Delegate Helper Objects . . . . .	14
Scatter Objects . . . . .	14
Crowd Behaviors . . . . .	14
Cognitive Controllers . . . . .	15
Global Clip Controllers . . . . .	15
Motion Synthesis . . . . .	15
Understanding character studio Workflow . . . . .	15
Create Skin Geometry . . . . .	16
Create a Biped Skeleton . . . . .	16
Attach the Skin . . . . .	16
Adjust Skin Behavior . . . . .	16
Animate the Biped Skeleton . . . . .	17
Use Freeform Techniques . . . . .	17
Use a Footstep-Driven Technique . . . . .	17
Convert Between Animation Types . . . . .	17
Use Layers to Apply Global Changes . . . . .	18
Use In Place Mode to Control the View . . . . .	18
Import Motion Capture Files . . . . .	18
Use Track View for Keyframe Editing . . . . .	18
Use Motion Flow to Combine Animations . . . . .	19

Refining Your Character . . . . .	19
Use Crowds to Animate Groups of Characters . . . . .	19
Starting Biped . . . . .	20
Creating a Biped . . . . .	20
Changing Biped Parameters . . . . .	20
Creating a Biped Using AutoGrid . . . . .	21
Naming the Biped . . . . .	22
Understanding Biped Anatomy . . . . .	22
Using Animation Tracks . . . . .	22
Changing the Biped Hierarchy . . . . .	23
Repositioning Biped Body Parts . . . . .	23
Adding extra Biped Body Parts . . . . .	23
Changing Initial Biped Anatomy . . . . .	24
Deleting a Biped . . . . .	24
Loading and Viewing Sample Biped Animations . . . . .	24
Previewing Biped Motion . . . . .	25
In Place Mode . . . . .	26
In Place Mode Options . . . . .	26
Biped Display Options . . . . .	27
Trajectory Display . . . . .	27
Display Preferences . . . . .	28
Retargeting Freeform Motion . . . . .	29
IK Constraints and Pivots for Freeform and Footstep Animations . . . . .	29
Join to Previous IK Key . . . . .	29
About Freeform Animations . . . . .	30
Freeform Walking Animation Using IK Constraints . . . . .	30
Biped Tracks . . . . .	30
Selecting Tracks with the Select by Name Tool . . . . .	31
Selecting Tracks from the Track View . . . . .	32
Expanding Biped Animation Tracks . . . . .	32
Transforming Links . . . . .	32
Moving Links . . . . .	33
Rotating Links . . . . .	33
Special Rotation: Elbows and Knees . . . . .	34
Balance: Spine . . . . .	35
Independent Orientation: Arms, Head, Feet . . . . .	35
Ground Plane Collision Detection: Pelvis, Feet . . . . .	35
Rotating Multiple Links . . . . .	36
Selecting and Rotating Multiple Links . . . . .	36
Using Bend Links Mode . . . . .	36
Setting Biped Keys . . . . .	37

Adjusting Multiple Keys . . . . .	38
Apply Increment. . . . .	38
Select Multiple Keys . . . . .	39
Scale Tail Keys . . . . .	39
Using In Place Mode to Adjust Keyframes . . . . .	39
Copying and Pasting Postures. . . . .	40
Copying the Entire Biped Posture . . . . .	41
Copy Tracks. . . . .	41
Globally Position the Biped with Freeform Animation . . . . .	41
Changing the Biped Structure . . . . .	43
Posing a Biped . . . . .	45
How Biped Uses Figure Mode. . . . .	45
Restructure Biped to Match File . . . . .	45
Talent Figure Mode and Adjust Talent Pose . . . . .	45
Saving and Loading Biped Figure Files . . . . .	46
Scaling Links. . . . .	46
Rubber-Banding Arms and Legs . . . . .	47
Rubber-Banding the Center of Mass . . . . .	48
Balance Factor . . . . .	50
Linking Objects to the Biped. . . . .	50
About Footstep Animation . . . . .	51
IK Constraints and Footstep Animation. . . . .	53
Composing Footsteps. . . . .	53
Workflow . . . . .	53
Planning for Footsteps . . . . .	55
Footstep Timing: Gait Parameters. . . . .	56
Walking . . . . .	56
Running . . . . .	56
Jumping . . . . .	56
Creating Multiple Footsteps . . . . .	57
Creating Individual Footsteps . . . . .	57
Editing Footsteps in Space . . . . .	58
Activating and Deactivating Footsteps . . . . .	59
Editing Footsteps in Time: Track View. . . . .	60
Restrictions. . . . .	61
Sliding Footsteps . . . . .	62
Motion Capture Import . . . . .	62
Adjusting Vertical Dynamics . . . . .	63
Dynamics of Walking . . . . .	63
Dynamics of Ballistic Gaits: Running, Hopping, and Jumping . . . . .	64
Airborne Vertical Dynamics. . . . .	64

Springing and Landing Dynamics . . . . .	65
Ballistic Tension . . . . .	65
Adjusting Biped Keys in Track View . . . . .	65
Locked Keys . . . . .	65
Restrictions . . . . .	65
Adapting Biped Keys to Footstep Edits . . . . .	66
Adapting Keys to Footstep Space Edits . . . . .	66
Adapt Locks . . . . .	67
Adapting Keys to Footstep Time Edits: Track View . . . . .	67
Motion Interdependencies Between Body Parts . . . . .	68
Freeform Editing Between Footsteps . . . . .	68
Splicing Biped Motion . . . . .	68
Retargeting Biped Motion . . . . .	69
Scale Stride Mode . . . . .	69
Loading and Saving Biped Step Files . . . . .	70
Troubleshooting . . . . .	70
Converting Between Footstep and Freeform Animations . . . . .	70
Key Info Rollout . . . . .	72
Vertical and Horizontal Center of Mass Tracks . . . . .	72
Using IK Keyframe Parameters . . . . .	72
IK Blend . . . . .	73
Body/Object Option . . . . .	73
Join to Previous IK Key . . . . .	74
Freeform Walking Animation Using IK Constraints . . . . .	74
Walking Keys . . . . .	74
Pivots (IK Extensions) . . . . .	77
Biped IK . . . . .	77
Animatable IK Attachments to 3D Studio MAX Objects . . . . .	79
Using Layers . . . . .	81
Mirroring Motion . . . . .	81
Bending the Center of Mass Track and Trajectory Key Editing . . . . .	82
Filtering Motion Capture and Marker Data . . . . .	83
Transitions . . . . .	86
Random Motion . . . . .	86
Unified Motion . . . . .	86
Files and Directories . . . . .	86
Envelopes . . . . .	88
Deformable and Rigid Envelopes . . . . .	88
Blending Between Links . . . . .	89
Inner and Outer Bounds . . . . .	89
Bulges and Tendons . . . . .	90

Skeletons to Use with Physique . . . . .	91
Creating a Skin . . . . .	92
Creating a Skin for Biped . . . . .	93
Adjusting Biped Arms Legs and Torso . . . . .	93
Using the Box Generator Utility for Non-Biped Skeletons . . . . .	94
Spline-Based Physique Deformation . . . . .	95
Floating Bones . . . . .	95
Applying Physique . . . . .	95
Initializing Physique . . . . .	96
Previewing Motion . . . . .	96
Adjusting Default Envelope Shape . . . . .	97
Fine-Tuning Envelopes . . . . .	97
Using Cross Sections . . . . .	98
Using Control Points . . . . .	98
Rotating and Scaling . . . . .	98
Interactive Redraw . . . . .	99
Changing Display Options . . . . .	99
Working with Deformable Envelopes . . . . .	99
Adjusting Link Parameters . . . . .	100
Creating Bulges . . . . .	101
Setting Bulge Angles . . . . .	102
Shaping the Bulge . . . . .	102
Fine-Tuning in the Bulge Editor . . . . .	103
Adding More Poses to Your Character . . . . .	103
Creating Tendons . . . . .	103
Working with Rigid Envelopes . . . . .	103
Working with Both Deformable and Rigid Envelopes . . . . .	104
Using Vertex Sub-Objects to Override Envelopes and Blending . . . . .	104
Shaded Display of Vertex Weight Values . . . . .	105
Reinitializing Physique Settings . . . . .	105
Reinitializing Old Files . . . . .	105
Initialization Area . . . . .	106
Working with an Initial Pose . . . . .	106
Improving Interactive Performance . . . . .	106
Level of Detail Controls . . . . .	106
Turning Blending On and Off . . . . .	107
Using the Optimize Modifier with Physique . . . . .	107
Combining Physique with Other Modifiers and Stack Update Options . . . . .	107
Stack Update Options . . . . .	108
Using Physique with Changing Geometry . . . . .	108
Saving and Loading Physique Data . . . . .	108

Crowd Object. . . . .	110
The Setup Rollout. . . . .	110
Crowd Delegates . . . . .	111
Creating a Crowd System . . . . .	111
Crowd Behaviors Overview . . . . .	112
Understanding Behaviors . . . . .	113
Using Behaviors . . . . .	114
Behavior Tips . . . . .	114
Obstacle-Avoidance Behavior . . . . .	115
Attaching an Object Instance to a Crowd System . . . . .	116
Using Bipeds with Crowd Delegates . . . . .	116
The Motion Flow Network Defines Possible Scripts . . . . .	117
Bipeds Use Delegate-Directed Behavioral Goals . . . . .	118
Biped Crowd Avoidance, Priority, and Backtracking . . . . .	118
Clip Controllers . . . . .	119
Two Approaches to Animation . . . . .	119
Cyclic In-Place Animation . . . . .	119
Animation with Lateral Motion . . . . .	119
Global Motion Clip. . . . .	120
MasterMotionClip . . . . .	120
Synthesis Dialog . . . . .	120
<b>Chapter 2 Procedures. . . . .</b>	<b>123</b>
Using Biped and Physique . . . . .	123
Using the Crowd System . . . . .	127
Biped FAQs and Procedures . . . . .	128
Keyboard Shortcuts . . . . .	130
Facial Animation with character studio . . . . .	131
How the Dummies Should Be Linked . . . . .	131
<b>Chapter 3 User Interface . . . . .</b>	<b>133</b>
Biped User Interface . . . . .	133
Center of Mass . . . . .	134
Body Vertical Track: Dynamics and Ballistic Tension . . . . .	135
Body Vertical Track: Balance Factor. . . . .	135
Linking to the Center of Mass Object . . . . .	135
Center of Mass Shadow . . . . .	136
character studio and 3DS MAX Bones. . . . .	136
Bones that compress . . . . .	137
IK Solution . . . . .	137
Merging and Cloning a Biped Character. . . . .	139
Clones . . . . .	140

Preset Set Keys & Convert Bips	141
Authorization	146
Internet Registration	146
Phone or Email Registration	146
General Rollout	147
Figure Mode	156
Notes on Fitting the Biped to a Mesh in Figure Mode	157
Convert to Freeform or Footsteps Dialogs	158
When to Use Convert	159
Track Selection Rollout	160
Key Info Rollout	161
Activating Parameters	162
Tension Continuity and Bias (TCB)	162
Biped Dynamics Parameters	165
IK Key Info Rollout	167
Inverse Kinematics	167
Keyframing Rollout	173
Display Rollout	176
Display Preferences Dialog	177
Layers Rollout	178
Layers versus Set Multiple Keys	179
Animation Properties Rollout	181
Biped Dynamics and Spline Dynamics	181
Separate Tracks	181
Structure Rollout	184
Motion Capture Buffer	187
Marker Files	187
Motion Capture Conversion Parameters Dialog	192
Motion Capture Buffer	192
Motion Capture Batch File Conversion Dialog	197
Marker Display Dialog	197
Character Studio Marker Files	198
Prop Bone	198
BVH File Specification	199
Overview of BVH-supported Hierarchy and Naming	199
Character Studio Marker Name File (MNM)	200
Establishing a Correct Neutral Pose	201
General BVH File Structure	201
Sample Biped-Supported BVH File	204
BVH Guidelines for Using the Biped MoCap Converter	207
CSM File Specification	213



Overview of CSM Workflow and General Use . . . . .	213
Summary of New Biped Variables in 3dsmax	
\plucfg\biped.ini for CSM Processing (as of R2.2) . . . . .	216
CSM.MAX for Placement and Export of CSM Markers . . . . .	216
Character Studio Supported Marker Names and Attachments . . . . .	217
Overall CSM File Structure . . . . .	219
Handling Missing “dropout” marker data . . . . .	221
Additional Syntax Rules . . . . .	222
Character Studio Marker Name File (MNM) . . . . .	222
Sample CSM and MNM . . . . .	223
Random Motion and Crowds . . . . .	227
Workflow: Getting Started with Clips and Transitions in Motion Flow mode . . . . .	228
Motion Flow Rollout . . . . .	229
Motion Flow Graph Dialog . . . . .	230
Random Scripts for One or More Biped . . . . .	231
Motion Flow Script Rollout . . . . .	234
Scripts . . . . .	234
Transitions Between Clips . . . . .	235
Random Motion . . . . .	235
Position the Entire Animation . . . . .	235
Transition Editor . . . . .	237
Transitions . . . . .	237
Automatic Transitions . . . . .	238
Length (Transition Duration) . . . . .	238
Editing Transitions Manually (Ghosts) . . . . .	238
Other Transition Editor Features . . . . .	238
Save Segment Dialog . . . . .	241
Create Random Motion Dialog . . . . .	241
Shared Motion Flow Dialog . . . . .	243
Transition Optimization Dialog . . . . .	245
Clip Properties Dialog . . . . .	246
Footstep Mode . . . . .	247
Footstep Creation Rollout . . . . .	248
Create Multiple Footsteps Dialog: Walk . . . . .	250
Create Multiple Footsteps Dialog: Run . . . . .	253
Create Multiple Footsteps Dialog: Jump . . . . .	255
Footstep Operations Rollout . . . . .	258
Biped and Track View . . . . .	262
Footsteps in Track View . . . . .	263
Separate Tracks . . . . .	263
Editing Biped Keys in Track View . . . . .	263

Set Multiple Keys . . . . .	264
How Dynamics and Footsteps Relate . . . . .	264
Freeform Animation . . . . .	265
Footstep Mode Dialog . . . . .	267
Set Multiple Keys Dialog . . . . .	269
Synthesis Dialog . . . . .	270
ClipState Dialog . . . . .	275
Automatic State Creation . . . . .	275
MotionClip Parameters Dialog . . . . .	283
Track View Pick Dialog . . . . .	283
Motion Library . . . . .	284
Footsteps . . . . .	284
Freeform . . . . .	288
Motion Capture . . . . .	288
Motion Files . . . . .	289
Sample Animations . . . . .	290
Directory Structure . . . . .	290
Sample Files . . . . .	290
Ballistic Tension . . . . .	291
Gravitational Acceleration . . . . .	291
Stride Length Sample Animation . . . . .	291
Time to Next Footstep . . . . .	292
Center of Mass Position . . . . .	292
Dynamics Blend . . . . .	292
Physique Load Specification Dialog . . . . .	293
Box Generator Utility . . . . .	294
Floating Bones Rollout . . . . .	295
Physique Rollout . . . . .	296
Physique Level of Detail Rollout . . . . .	298
Envelope Sub-Object . . . . .	300
Patches . . . . .	300
Workflow . . . . .	300
Partial Blending and Weight Assignments . . . . .	306
In cases where no envelopes use Partial Blending (the default) . . . . .	306
In cases where all envelopes use Partial Blending . . . . .	306
In cases where some envelopes use Partial Blending and some do not . . . . .	306
Link Sub-Object . . . . .	307
The Physique Deformation Spline . . . . .	307
Bulge Sub-Object . . . . .	313
Workflow to Create a Biceps Bulge . . . . .	313
Bulge Editor . . . . .	314

Bulge Editor . . . . .	318
Tendons Sub-Object . . . . .	326
Workflow . . . . .	326
Vertex Sub-Object . . . . .	331
Physique Initialization Dialog . . . . .	335
Creating Physique Links and Envelopes for the First Time . . . . .	335
Link Settings, Joint Intersections, and Cross Section Rollouts. . . . .	336
Reinitialize Physique . . . . .	338
Scaling a Character. . . . .	339
Physique and Free Form Deformations (FFDs). . . . .	340
Geometry Parameters Rollout . . . . .	342
Motion Parameters Rollout . . . . .	342
Crowd Behaviors . . . . .	346
Behavior Rollout . . . . .	351
Solve Rollout . . . . .	352
Priority Rollout . . . . .	354
Using Priorities . . . . .	355
Smoothing Rollout. . . . .	357
Collisions Rollout . . . . .	358
Geometry Rollout . . . . .	359
Global Clip Controllers Rollout . . . . .	359
Scatter Objects Dialog . . . . .	362
Random Placement Difficulty Dialog . . . . .	370
Object/Delegate Associations Dialog . . . . .	370
Edit Multiple Delegates Dialog. . . . .	372
Associate Biped With Delegates Dialog. . . . .	374
Behavior Assignments and Teams Dialog . . . . .	375
Select Behavior Type Dialog. . . . .	380
Select Delegates Dialog . . . . .	381
State Dialog. . . . .	385
State Transition Dialog. . . . .	386
The Transition Script. . . . .	386
Behavior Rollout . . . . .	392
Avoid Behavior . . . . .	392
Orientation Behavior . . . . .	396
Path Follow Behavior . . . . .	398
Repel Behavior. . . . .	400
Scripted Behavior. . . . .	402
Seek Behavior . . . . .	403
Space Warp Behavior. . . . .	404
Speed Vary Behavior . . . . .	405

Surface Arrive Behavior . . . . .	406
Surface Follow Behavior. . . . .	409
Wall Repel Behavior. . . . .	411
Wall Seek Behavior. . . . .	414
Wander Behavior . . . . .	416
Create Method Rollout . . . . .	419
Lattice Parameters Rollout . . . . .	419
Obstacle Parameters Rollout . . . . .	420
<b>Tutorial Introduction. . . . .</b>	<b>423</b>
Welcome . . . . .	423
How to Use These Tutorials. . . . .	423
Online and Printed Tutorials . . . . .	423
Opening the Online Tutorials . . . . .	424
Finding the Tutorial Files. . . . .	424
Tutorials and Procedures . . . . .	424
MAXScript Tutorial. . . . .	424
What You Will Learn in These Tutorials . . . . .	424
The Cast of Characters . . . . .	426
<b>Tutorial 1 Biped and Physique . . . . .</b>	<b>429</b>
Lessons. . . . .	429
Lesson 1: Creating a Biped . . . . .	429
Lesson 2: Modifying the Biped Structure in Figure Mode . . . . .	433
Lesson 3: Biped with Physique . . . . .	438
Lesson 4: Merging and Cloning Characters. . . . .	442
<b>Tutorial 2 Freeform Biped Animation . . . . .</b>	<b>447</b>
Lessons. . . . .	447
Lesson 1: Creating a Simple Freeform Animation . . . . .	447
Lesson 2: Animating a Freeform Walk Cycle . . . . .	457
<b>Tutorial 3 Footstep Animation . . . . .</b>	<b>469</b>
Animating a Biped with Footsteps. . . . .	469
Lessons. . . . .	469
Lesson 1: Creating an Expressive Walk . . . . .	470
Lesson 2: Modifying Footsteps . . . . .	477
Adding a jump . . . . .	479
Lesson 3: Gymnastic Motion Flips with Ballistic Tension . . . . .	481
Adjusting the Body Motions . . . . .	485
Adding the Twist . . . . .	491
Lesson 4: Animating a Pratfall . . . . .	494
Lesson 5: Changing Footsteps using IK Keys . . . . .	500

<b>Tutorial 4 Animating Bipeds Interacting with Objects</b>	<b>503</b>
Making Bipeds Interact with Objects.	503
Lessons.	503
Lesson 1: Dribbling a Basketball	504
Lesson 2: Climbing a Ladder Using IK Blend	507
Lesson 3: Picking Up and Carrying Using Link Controller	516
Lesson 4: Creating the Illusion of Weight	519
Lifting Heavy Objects	519
Pushing Heavy Objects	521
Lesson 5: Using In Place Mode	522
<b>Tutorial 5 Motion Flow Editing</b>	<b>525</b>
Lessons.	525
Lesson 1: Creating Clips in Motion Flow Mode	526
Setup	526
Lesson 2: Creating and Using Motion Flow Scripts	526
Setup	526
Lesson 3: Looping Animation in Motion Flow Mode	529
Lesson 4: Using a Shared Motion Flow	530
Lesson 5: Using the Create Random Motion Feature	532
<b>Tutorial 6 Working with Motion Capture Data</b>	<b>535</b>
Setup	536
Lessons.	536
Lesson 1: Importing Motion Capture Data	536
Setup	536
Lesson 2: Comparing Trajectories	538
Setup	538
Using Calibration Controls	538
Using Marker Files	538
Lesson 3: Using High-Frequency Data and Looping	539
Lesson 4: Editing with Layers	540
<b>Tutorial 7 Skinning and Linking with Physique</b>	<b>543</b>
Skinning with Physique.	543
Lessons.	544
Lesson 1: Aligning a Biped to the Mesh Model	544
Setup	544
Lesson 2: Applying and Adjusting the Physique Modifier	548
Setup	548
Lesson 3: Adjusting Envelopes and Weighted Vertices.	549
Setup	549
Lesson 4: Fixing Possible Problem Areas: Shoulders and Pelvis.	551

Setup . . . . .	551
Lesson 5: Animating Muscles with the Bulge Editor. . . . .	552
Lesson 6: Scaling Characters with Physique . . . . .	553
Lesson 7: Linking to the Biped . . . . .	554
<b>Tutorial 8 Working with Crowd Animation . . . . .</b>	<b>555</b>
Lessons. . . . .	555
Lesson 1: Getting Started with Behaviors . . . . .	556
Lesson 2: Using Multiple Delegates and Behaviors . . . . .	559
Lesson 3: Applying Avoidance and Animating Behavior Assignments . . . . .	562
Lesson 4: Applying Logic to Crowd Behavior . . . . .	564
Lesson 5: Using Crowd with Animated Non-Biped Objects . . . . .	567
Lesson 6: Creating a Crowd of Swimming Bipeds . . . . .	573
Lesson 7: Advanced Crowd/Bipeds . . . . .	577
<b>Tutorial 9 Animating a Multilegged Creature . . . . .</b>	<b>585</b>
Lessons. . . . .	586
Lesson 1: Animating a Quadruped with Freeform Animation . . . . .	586
Lesson 2: Adding Extra Limbs . . . . .	598
<b>Biped MAXScript Extensions. . . . .</b>	<b>601</b>
Biped Load and Save Methods . . . . .	601
Biped Creation . . . . .	603
Biped Display Preferences Access . . . . .	605
Biped Sample Scripts . . . . .	605
biped_object : GeometryClass . . . . .	606
Biped Node Hierarchy . . . . .	606
Biped Layers . . . . .	609
Layer Related Methods . . . . .	609
Biped Vertical_Horizontal_Turn(Body):Matrix3 Controller. . . . .	610
Adapt Locks Group. . . . .	612
Separate Tracks Group . . . . .	613
FootSteps : Matrix3 Controller . . . . .	619
Biped Slave Controller . . . . .	620
Biped Footprints . . . . .	621
Biped Class : MultFprintParams. . . . .	623
FootSteps : Matrix3 Controller . . . . .	626
BipedFSKey : MAXObject . . . . .	627
MoFlow : MaxWrapper. . . . .	628
MoFlowScript : MaxWrapper . . . . .	630
MoFlowTranInfo : MaxWrapper . . . . .	633
MoFlowTransition : MaxWrapper . . . . .	635
Accessing a Biped controller key by index. . . . .	636

BipedKey : MAXObject . . . . .	637
BipedFSKey : MAXObject . . . . .	640
<b>Crowd MAXScript Extensions . . . . .</b>	<b>649</b>
Crowd : helper . . . . .	649
Delegate : Helper . . . . .	651
CrowdScatter: . . . . .	657
CrowdAssignment : MAXObject . . . . .	663
CrowdTeam : ReferenceTarget . . . . .	664
CrowdState:ReferenceTarget . . . . .	665
CrowdTransition : MAXObject . . . . .	666
Crowds - Methods . . . . .	667
CogControl : MAXObject . . . . .	670
Avoid_Behavior : MAXObject . . . . .	671
Orientation_Behavior : MAXObject . . . . .	673
Path_Follow_Behavior : MAXObject . . . . .	675
Repel_Behavior : MAXObject . . . . .	677
Scripted_Behavior : MAXObject . . . . .	678
Seek_Behavior : MAXObject . . . . .	686
Space_Warp_Behavior: MAXObject . . . . .	687
Speed_Vary_Behavior : MAXObject . . . . .	688
Surface_Arrive_Behavior : MAXObject . . . . .	689
Surface_Follow_Behavior : MAXObject . . . . .	693
Wall_Repel_Behavior : MAXObject . . . . .	694
Wall_Seek_Behavior : MAXObject . . . . .	696
Wander_Behavior : MAXObject . . . . .	697
Vector_Field: SpacewarpObject . . . . .	699
<b>CS3Tools.cui Tutorial . . . . .</b>	<b>705</b>
Overview . . . . .	705
Launching the CS3CustomKeys User Interface . . . . .	705
File Details . . . . .	707
Design Details. . . . .	707
The Change_Key_FN Code. . . . .	711
CS3Save_Presets Code . . . . .	714
Set_Key_FN Code . . . . .	715
Naming the Buttons . . . . .	719
Extending the Number of Presets . . . . .	719
<b>Glossary . . . . .</b>	<b>721</b>





---

# Welcome to character studio 3

Welcome to **character studio 3** for 3D Studio MAX R3.1.

For a quick overview of the changes and additions in this version, see *What's New in character studio 3* (see page xix).

## See also

*Introduction to character studio* (see page 1)

*Introduction to Biped* (see page 5)

*Introduction to Physique* (see page 11)

*Overview of Crowd Features* (see page 13)

*Tutorials* (see page 423)

Note: A number of character studio 3-specific additions have been made to the MAXScript language. The MAXScript Reference is updated when you install the character studio software, and is still available online from Help menu > MAXScript Reference. MAXScript information specific to this product is also available as a separate chapter in the printed Reference.

---

## What's New in character studio 3

**character studio 3** for 3D Studio MAX R3.1 combines evolutionary and revolutionary enhancements to the 3D character animation process. These improvements offer dramatic new capabilities for:

- *Nonlinear animation* (see page xx) (Biped Motion Flow)
- *Track operations* (see page xx) (Biped)
- *Behavioral crowd animation* (see page 559) (Crowd)
- *Inverse kinematic pivots* (see page xxii) (Biped)

## Welcome to character studio 3

- *Keyframe animation workflow* (see page xxiii) (Biped)
- *Footstep animation improvements* (see page xxiii)
- *Figure mode improvements* (see page xxiv)
- *Motion capture editing* (see page xxiv)
- *Skin deformation* (see page xxiv) (Physique)
- *Programmatic access* (see page xxiv) (Biped and Crowds)

## Nonlinear Animation

Unique enhancements to the Motion Flow Editor accelerate layout, tuning, and sharing of large-scale nonlinear animation systems between Biped characters.

- **Create Clips from Files.** Lets you load multiple files into the motion flow editor in a single step. See *Motion Flow Graph* (see page 230).
- **Synthesize Motion Flow Graph.** This will create optimized transition between all the clips in the motion flow graph. This allows you to quickly create scripts or random motion. With a single click, hundreds of stock animations on disk can be assembled automatically into a nonlinear Motion Flow graph representing every possible nonlinear transition.
- **Optimize Selected Transitions.** Selection of blend points for each transition in the graph means that best-case blending of large nonlinear Motion Flow graphs can be delivered in hours instead of weeks.
- **Create Random Motion.** Probability-based traversal of a nonlinear motion flow graphs means that a character's animation can cycle autonomously, within its prescribed Motion Flow graph, based on per-transition probabilities.

- **Shared Motion Flow.** Sharing of Motion Flow graphs between Biped means that 100's of characters in a scene can motivate from a central nonlinear graph of possible motions and probabilities, each yielding a unique, but predictable behavior. Built-in Biped motion mapping means that sharing of Motion Flow graphs between characters of different size and bone structure is trivial.
- **Create Unified Motion.** Any path through a nonlinear Motion Flow graph (called a Motion Flow Script) can now be unified with the click of a button to yield a high-quality animation segment that incorporates IK constraints and pivot points through each blend period.

## Track Operations

A simple, yet powerful extension to Biped's existing compliment of track operations opens new doors for transferring entire tracks of motion between Biped body parts, between separate Biped, and even between stacked layers of Biped motion.

- **Copy/Paste Tracks.** Allows the entire motion from any Biped track to be copied into a buffer and easily reflected or pasted onto any other biped part. Tracks can also be copied and pasted between different biped characters -- with motion mapping adaptation automatically applied. Biped motion stored within Biped layers can now be separated and recombined onto other Biped -- or reordered within the same Biped.

## Behavioral Crowd Animation

**character studio 3** provides tools and controls for intelligent animation of large systems of arbitrary characters via the Crowd system.

New Crowd and Delegate helper objects let you design crowd animation which can be based on behaviors. Use animated 3DS MAX objects to create flocks and herds, or use bipeds to make groups of humans with motion-flow script driven animation. When using bipeds and delegates together, you can take the speed from the biped motion, and the turning and position from the motion of the delegate. When using standard animated 3DS MAX objects, an extension to Block Controllers called Clip Controllers are used to attach non-bipedal animated 3DS MAX objects to the crowd of delegates.

- **Delegate Helper Objects.** Crowd delegates are lightweight placeholders, or dummy objects, that may be assigned any number of available stock behaviors like seek, repel, or follow surface. The delegate object acts as a behavioral driver for its assigned character (a biped or 3D Studio MAX object).
- **Delegate Attributes.** Independent of the behaviors, delegates have their own weighted attributes for speed, turning and banking. Attribute values can also be selectively randomized across hundreds of delegates in a single selection, for efficient control of large populations. These attributes may be modified or animated individually, separately from their behaviors, to give each a unique personality.
- **Teams.** Delegates may be organized into teams, to help simplify behavior assignment for large systems. See *Delegate Helpers* (see page 342).
- **Crowd Helper Objects.** The crowd object acts as a handle for managing the entire crowd system and its assigned delegates; it is the primary source for configuring and solving crowd simulations. Allows you to scatter delegates in various positions based

on a random seed. Crowd objects provide the interface to create and assign behaviors, edit multiple delegates, link objects to delegates, access the cognitive controller editor, and more. Solving a crowd simulation generates key-framed motion for every delegate that has been assigned to the system. Use Step Solve with the spacebar to solve a frame at a time. See *Crowd Helper Objects* (see page 346).

- **Behaviors.** Delegate helper objects can be animated via behaviors. Delegates can avoid each other, or a delegate can seek another delegate or an object. You can fine-tune and name stock behaviors, or develop completely new ones using MAXScript. You can even animate the intensity of a particular behavior for a delegate or team over time, providing intuitive mixing control for delegates with multiple behaviors. Behaviors include: *Avoid* (see page 392), *Orientation* (see page 396), *Path Follow* (see page 398), *Repel* (see page 400), *Scripted* (see page 402), *Seek* (see page 403), *Space Warp* (see page 404), *Speed Vary* (see page 405), *Surface Arrive* (see page 406), *Surface Follow* (see page 409), *Wall Repel* (see page 411), *Wall Seek* (see page 414), and *Wander* (see page 416).

Use the space warps that affect particles and dynamics. Use MAXScript to write other behaviors. Behavior parameters appear in a behavior rollout in the crowd object. See *Crowd Behaviors* (see page 112).

- **Cognitive Controllers.** Create decision trees, using the Cognitive Controller Editor to create states to work with the Behaviors. Different behaviors will be applied depending on the state. See *Cognitive Controllers* (see page 382).

- **Vector Fields.** Create Vector Field space warps to use with crowds and behaviors. Computed from arbitrary 3D objects, vector fields generate structured forces that can repel or contain. See *Vector Field Space Warps* (see page 417).

## How Crowd Animation Works

When a crowd system is solved, characters in the scene that have been attached to crowd delegates move along the delegates' paths. A crowd delegate can be attached either to a Biped or to an arbitrary 3D Studio MAX object that represents a character.

Crowd uses two different methods for attaching delegates to Biped and to arbitrary 3D Studio MAX objects and for managing their respective motions. This separation allows the Crowd system to work well for any "non-terrestrial" 3D Studio MAX character, but to also take full advantage of Biped's nonlinear motion flow features and motion mapping capabilities for "terrestrial" bipedal characters. This latter connection brings the full power of Biped animation tools to bear on the behavioral animation of large systems of bipedal characters.

For delegate-driven bipeds, the Motion Flow graph provides critical information to the crowd system about the character's library of possible moves. With this advance knowledge, the crowd system solves the delegate's path and speed accordingly, while selecting and bending "best fit" motion clips from the Biped's Motion Flow graph -- driving the character over terrain and through obstacles -- synchronizing its motion with the ground at all times. After solution, each Biped possesses a lightweight Motion script which may be viewed and edited for maximum control.

The use of shared Motion Flow graphs between Biped driven by a crowd system allows hundreds of Biped to potentially share the identical motion flow graph and assets -- and yet move through a scene with completely unique results.

Global motion clips and master motion clips can be used to apply best fit motion clips from an existing animation sequence to non-Biped objects linked to delegates, based on the trajectory of the delegates. This is ideally suited for realistically animating such phenomena as flocks of birds and schools of fish.

## See also

*Using Crowd Animation* (see page 109)

*Crowd Animation User Interface* (see page 341)

## Inverse Kinematic Pivots

- **Animated Pivot Points for hands and feet.** The Biped inverse kinematic system now incorporates a fast, intuitive method for animating between user-specified pivot points on the hands and feet. Pivot points establish a center for rotation during a period of IK contact with a surface. Since they can be specified on a per-key-frame basis, complex interactions between feet, hands, and surfaces can be rapidly set up and animated, with high-integrity interpolation between keyframes with different pivot points.
- **Select Pivot.** Lets you select and keyframe pivot points in the viewport. Now you can animate the foot pivoting on the heel at one frame, and pivoting on the ball of the foot at another frame, then the toe at another. Includes a special feature, if the pivot is on the ball of the foot, to allow the toes to stay locked in world space so they don't dip

below the floor plane. Similarly, you can select pivots for the hands and fingers. Selecting the pivot instantly sets the IK Blend information at that keyframe. If you set the pivot at the toe, you can raise the knee and the heel will raise up, leaving the toes on the ground.

- **Quadruped animation.** Unification of pivot point animation for both hands and feet opens the door to more sophisticated quadruped use of the Biped armature. Because the hands and feet now offer identical capabilities and IK interpolation, quadruped animation using Biped is more fluid and controllable -- and transferable. Even walking fingers may be animated with surprising ease, since pivot points can be transferred to the ends of different fingers on a pose-by-pose basis.
- **Ankle tension spinner.** This determines joint precedence between the ankle and the knee. With Ankle Tension set to 1, the ankles remain stiff, like walking in snowboarding boots.
- **Set Planted, Sliding or Free Keys.** New set key buttons in the IK Key Info rollout let you do in one step what used to take three. Setting a Planted Key will change IK Blend to 1, will turn on Join to Prev IK Key, and will select Object space. Setting a sliding key sets IK Blend to 1, and turns on Object space, but leaves join to Prev IK Key off. Set a Free Key and IK Blend is set to 0, and Body space is turned on.
- **IK Key Info rollout.** A new rollout for faster workflow combines the existing IK Blend and Body/Object space controls with new icons for creating planted, sliding or free keys.

## Keyframe Animation Workflow

- **Custom Key Presets.** New tear-off custom key sets provide traditional animators with the most efficient means available for setting up and tuning straight-ahead character animation -- with all the power of blended inverse and forward kinematic solutions. With MacroKeys, traditional animators can configure their favorite key types, combining up to 13 different "set key" attributes, and store them by name, for rapid access via the new **character studio** tool panel.
- **Copy and Paste Tracks.** Provides simple and rapid transfer of Biped track motion between body parts, between Bipedes, and even between Biped layers.
- **Trajectories.** You can now see and edit Biped trajectories directly in the viewport. Use the Biped Display rollout to view the trajectories, edit the keys on the trajectory by turning on Sub-object and using the Transform gizmo.

## Footstep Animation Improvements

- **Footstep Placement with AutoGrid.** Footsteps can be placed on an arbitrary terrain by turning on AutoGrid. Footsteps will automatically snap to the center of each face.
- **Changing Footstep Duration.** Inserting, deleting or changing the IK space or blending will now change footstep duration in footstep animations. Previously, this was only possible through Track View manipulation of the footstep track. Footsteps can also be created and deleted by changing IK blend key parameters.
- **Better conversion between freeform and footstep animation.** This new IK constraint

system provides for two-way conversion between freeform and footstep animations that are more consistent and accurate. You can now convert back and forth between the two modes without loss of motion.

## Figure Mode Improvements

- **Most Recent .fig file.** You can now define the figure file you will use before you create a biped. You can load the FIG file directly from the Biped Creation rollout, or choose to use the most recent FIG file.

## Motion Capture Editing

- **File import.** Now you can load BVH and CSM file types directly.
- **Better motion capture file conversion and editing.** The new IK constraint system allows for more precise and accurate motion capture file conversion. Motion capture import with no key reduction using footstep extraction provides a precise means of dramatically editing raw motion files.
- **Prop Bone.** Import for CSM file now supports additional bones for extra elements. For swords, canes, hats and any else held in the hands. See *Prop Bone* (see page 198).

## Skin Deformation

Physique offers dramatic improvements in performance and deformation functionality, especially for highly detailed polygon and Bezier patch characters.

- **Performance.** Every major aspect of Physique's performance has been optimized and multithreaded to provide new levels of interactivity. Load times, stack updates, and general interactivity in the viewport have been tripled, or greater, even on single

processor systems. Multi-threading means that users with dual-processor systems running Windows NT or Windows 2000 can approach another doubling of speed over equivalent single-processor systems.

Additionally, new user-configurable Stack Update settings allow Physique to respond intelligently to animated changes in the 3D Studio MAX modifier stack. When enabled, these new settings ensure proper behavior when Physique is applied on top of animated modifiers like Morph or Mesh Smooth. When disabled, they allow Physique to focus on interactive performance, when minimal changes are occurring beneath it in the modifier stack.

- **Deformation Tools.** Physique now supports the use of non-linked bones and splines for animating and deforming non-hierarchical portions of a character. For example, facial animation, muscle-effects, and "breathing bones" can now be implemented by simply adding a bone or spline to Physique's list of bones for a given character. All of Physique's familiar deformation envelopes and vertex weighting tools may be used to control regional influence and deformation effects along each bone or spline. Animating the bone or the individual points along the spline yield a corresponding change in the mesh.

## Programmatic Access

- **Biped.** This version of Biped has been exposed to MAXScript to allow comprehensive "set" and "get" functions for creation, editing, and export of all aspects of Biped character data. New areas of programmatic access include biped creation, loading and saving, setting and getting keys and all key attributes,

footprints, Motion Capture import, and Motion Flow I/O.

- **Crowds.** The Crowd system is exposed to MAXScript, primarily for setting and getting of delegate attributes, for creation of cognitive controller and clip controller logic, and for creation of entirely new behaviors. These capabilities are detailed in a new update to the MAXScript online help system.

Note: A number of **character studio 3**-specific additions have been made to the MAXScript language. The MAXScript Reference is updated when you install the **character studio** software, and is still available online from Help menu > MAXScript Reference. MAXScript information specific to this product are also available as a separate chapter in the printed Reference.

## How to Contact discreet

We're interested to hear your views about the software. We'd like to hear ways you think we can improve our program, features you're interested in, as well as your views on the documentation set. Please let us know what you like, what you don't like, what you'd like to see added, or what we should do differently.

If you have information Also, please contact us using the feedback form on our company home page: <http://www2.discreet.com/company> and click "submit comments and suggestions".

## Product Information

For information about **character studio** products, go to <http://www2.discreet.com/games> and click "character studio".

# Installing character studio

## System Requirements

The following minimal combination of hardware and software is required to operate **character studio**:

- **Software:** 3D Studio MAX R3.1
- **Computer:** Pentium®II CPU, running at 200 megahertz or better. (The software is thoroughly multi-threaded to take full advantage of multiple processor systems; dual processor system recommended).
- **Operating system:** Microsoft®Windows NT 4.0 with Service Pack 3 or greater (SP4 is required for Year 2000 compliance), Workstation or Server version, or Windows 98.

Note: Windows 95 is not supported.

- **RAM:** Minimum 128 megabytes (MB). Recommended is 256 MB. Depending on scene complexity, more RAM may be desirable.
- **Free hard disk space:** Typically 350 MB of free hard disk space for installation. This varies according to the custom components you choose to install.

Swap File: The Windows swap file size should be a minimum of 300 MB.

Recommended swap file size is three times the amount of physical RAM on your computer. Depending on scene complexity, more Windows swap space may be desirable.

- **Display:** Graphics card supporting a minimum resolution of 1024 x 768 x High Color. Recommended resolution is 1280 x 1024 x True Color. (256 color mode is not supported.)

- OpenGL and Direct3D hardware acceleration supported, 24-bit 3D graphics accelerator preferred.
- **Pointing device:** Microsoft-compatible pointing device.  
**Important:** Make sure you are using the most current driver for your pointing device. The best method to obtain the most current driver for any peripheral you are using is to go to the manufacturer's web site and download from there.
- **CD-ROM:** Required for loading software, and needed for access to tutorial and sample files.

### Optional

- **Sound card and speakers:** Required for hearing soundtracks.
- **Network (Optional):** A TCP/IP-configured network for use in network rendering. Network rendering is only available on Windows NT systems.
- 3D hardware graphics acceleration, video input and output devices.
- **Other:** To view the online help systems, you need Internet Explorer® 5.0 or higher already present. Internet Explorer 5.0 is provided on the 3D Studio MAX R3.1 CD. You also need NT Service Pack 4 or later. NT Service Pack 4 is supplied on the program CD.

---

## Installing character studio 3

To install **character studio 3**, insert the CD in the CD-ROM drive, and the install program should begin automatically.

If the automatic install does not take place, use Windows Explorer or My Computer to navigate to the CD-ROM drive and double-click *setup.exe*. This will start the installation program.

---

## Authorizing character studio

You must authorize **character studio 3** before you can use it to create character animations. The first time you create a biped, a crowd, or apply Physique to a skin, the software displays an authorization dialog.

Instructions for authorizing your copy of the software are on the registration card included in the product package.

Note: Your copy of **character studio** is tied to the 3D Studio MAX lock you use during the authorization process. Be sure to authorize **character studio** on the 3D Studio MAX installation where you plan to use the product. If you change or replace locks, you will need a new authorization code.

### See also

*Authorization (see page 146)*



# Using character studio

---

## Introduction to character studio

**character studio** provides professional tools for building and animating 3D characters. It is an environment in which animators can quickly and easily build skeletons and then animate them, thus creating motion sequences. The animated skeletons are used to drive the movement of 3D Studio MAX geometry, thus creating virtual characters. Crowds of these characters can be generated using **character studio**, and animated using a system of delegates and procedural behaviors.

**character studio** consists of three plug-ins for 3D Studio MAX: Biped, Physique, and Crowd.

- **Biped** (*see page 723*) builds and animates skeletal armatures. You can combine different animations into sequential motion scripts, or layer them together. You can also use Biped to edit motion capture files.
- **Physique** (*see page 731*) uses the biped armatures to animate the actual 3D characters. You can also use Physique with other 3DS MAX hierarchies beside the biped skeleton.
- **Crowd** (*see page 724*) animates groups of 3D objects and characters using a system of delegates and behaviors.

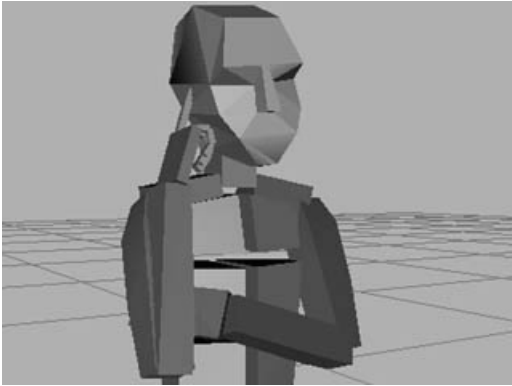
If you are new to 3D Studio MAX, read *What You Should Know to Use character studio* (*see page 2*) and *Understanding character studio Workflow* (*see page 15*).

If you are already a 3D Studio MAX user, proceed to *Understanding Biped* (*see page 5*) to continue.

To find out about Physique, read *Understanding Physique* (see page 11).

To learn about the Crowd system, read *Overview of Crowd Features* (see page 13).

## What You Should Know to Use character studio



**character studio** provides you with a broad range of tools that help you create and animate virtually any type of two-legged character. In addition, the software uses many built-in 3D Studio MAX features to provide a foundation for character skin modeling, as well as general manipulation tools for object transformations and keyframe editing.

To expedite your introduction to the product, you should be familiar with the following 3D Studio MAX concepts:

- Creation, transformation, and modification of objects
- Selection of objects through clicking or dragging in viewports, and by using the Select by Name dialog

- Navigation in viewports and changing your viewport configuration
- Use of Track View and the trackbar to view and edit animation tracks and keys

Refer to 3D Studio MAX printed and online documentation for information on these areas. If you don't know how to use 3DS MAX, do some of the introductory tutorials found online in Help > Learning 3D Studio MAX.

## Biped Features and Benefits

Biped combines the fast accuracy of footstep sequence editing, motion capture import, and *motion blending* (see page 728) with the expressive control of traditional *keyframe animation* (see page 728) and innovative *inverse kinematics* (see page 727) (IK). The following is a brief summary of significant Biped features and benefits.

### Rapid Creation of Biped Skeletons

**Instant Animatable Armatures.** Create entire biped character skeletons instantly and then alter their structure using Biped's parametric figure controls. Add tails and ponytails for animating ears, jaws, or horns. Add modifiers to biped objects to create precise bone shapes. Use Autogrid to build biped skeletons on top of other objects.

**Figure Files.** Save custom character figures separately and re-use them in different scenes or even with different Biped motion sets to quickly generate completely new character animations. Mix and match the action and the geometry, regardless of scale and proportion.

## Freeform Animation

**Traditional Keyframe Animation.** Take total control of your character with powerful freeform keyframe animations tools. Use *freeform animation* (see page 726) to make your characters fly through the air, or swim in the sea, to stand up, sit down, or talk on the phone. *Animate traditional walk cycles* (see page 457) by blending between forward and inverse kinematics. Animate between different pivot points on the hands and feet.

**Conversion from Footsteps.** Automatically convert your *footstep animation* (see page 725) sketch into a freeform representation, or create your animation by hand without footsteps by keyframing character poses and setting intuitive IK constraints for your character's hands and feet. Convert back and forth between freeform and footstep animation modes maintaining motion integrity.

**Convert from motion capture.** Bring in animations captured from actors or from other animation programs. Reduce keyframes for easier editing.

**In Place viewing.** Use *In Place mode* (see page 727) to automatically view and modify your character's motion in the center of your current view.

**Animation Layers.** Make subtle or dramatic changes in character's motion using layers of animation on top of your initial sketch, then collapse your layers into a single set of keyframes.

**Expandable Animation Tracks.** Get the level of keyframe control you want by expanding or collapsing animation tracks for Biped arms, legs, spine, neck, tail, and ponytails. Expand tracks to set keyframes independently for each joint, or collapse them to reduce the number of changes you need to make.

**Fine-tuning for Keyframes.** Adjust the subtlest aspects of your character's motion by fine-tuning each individual keyframe to adjust its tension, continuity, bias, ease-in, and ease-out characteristics.

## Footstep-Driven Animation

**Rapid footstep creation.** Easily create footstep-driven animations like walking, running, jumping, and dancing by placing footprints interactively in the 3D scene. Edit footstep timing and position at any time to see an instant update of your character's posture and motion.

**Automatic balancing.** Help your characters correctly move, rotate, and dynamically balance about their center of mass. Biped's footstep mode ensures that characters walk with correct relationships between the ankle, leg, and pelvis, and bank into turns as a function of speed and path curvature. Use Ankle tension to adjust the joint precedence between the knee and the heel.

**Footstep Splicing.** Easily splice footstep motions together by copying and inserting sequences of footsteps and their associated upper body movements. Generate seamless motion cycles by copying and then reinserting selected sequences of character footsteps.

**Uneven terrain.** Create footsteps with varying heights and orientations to animate walking up stairs, jumping off a two-story building, or running up a hill. Create uneven terrain either when you initially generate footstep, or when you adapt them later. Use Autogrid to place footsteps on uneven terrain.

**Accurate foot/ground collision response.** The Biped character's specific foot structure accurately collides with and rolls over each footstep location in a natural way, pivoting between the heel, sole, and potential contact points of the toes (or claws).

## Advanced Inverse Kinematics

**Natural IK for 3D characters.** Easy to use and 100% precise IK that is specialized to work for character limbs, rather than arbitrary robotic linkages. You can quickly specify IK paths to work in either the coordinate system of the character's body or the coordinate space of some other animated object in the scene or the world coordinate system.

**Animatable IK attachments to MAX objects.** Dynamically attach and detach the hands and feet of a biped character to a 3D Studio MAX object during an animation, as in catching and throwing a basketball, rowing a boat, or dancing with another biped character.

**Animated IK Blending.** Selectively blend between IK goals and joint rotations using IK Blending keys for the hands and feet of a Biped. IK Blending allows you to easily craft sophisticated motions like punching or walking that represent a change in the character's motivation or reaction to a fixed surface.

## Motion Retargeting Between Characters

Apply footsteps or freeform animation from one Biped character to any other Biped character regardless of height, proportion, or even structural differences. Craft great motion sequences once and use them over and over with all your characters. With dimensional scaling, Biped adapts the placement of footsteps and joint rotations to account for differences in leg and pelvis dimensions, and adapts gravity to the relative stance of the Biped character.

## Motion Capture Operations

**Motion Capture File Import.** Import motion capture data from either *positional/optical markers* (see page 728) in **character studio's** *.csm*

format or from rotational hierarchies in BioVision's *.bvh* format. Choose the format that matches your raw motion capture data.

**Motion Capture Filtering.** Preserve only the most significant attributes of the motion capture animation by automatically reducing motion capture keyframe data. Ensure that your character's feet stay locked on the ground at the right times without compromising the integrity of the original captured animation by extracting footsteps during import. Create traditional keyframe animations that can be freely altered without the use of footstep constraints by importing your motion capture data as freeform animation.

**Motion Capture Editing.** Progressively refine your motion capture import by projecting it onto an existing set of footsteps and freeform periods. Keep a record of your latest import of motion capture data in its raw unfiltered form as a visual reference of your true captured data. You can paste specific poses and keyframes from the motion capture buffer back onto your Biped figure at any frame. Batch conversion allows you to convert hundreds of motion capture files into the Biped format with one step.

**Motion Flow Blending and Scripting.** Combine separate Biped animations into continuous sequences using *motion flow* (see page 729) and *motion blending* (see page 728) to develop scripts that seamlessly incorporate many Biped animation files. Loop any motion using the *motion flow editor* (see page 729). Create a library of motions by saving scripts which you can then re-map to any Biped character. Use *Motion Flow Scripts* (see page 729) to create the animations for crowds of characters. Create motion flow networks that automatically look at a group of motions and creates transitions between them. Generate shared motion flow networks to be shared between bipeds.

## Understanding Biped

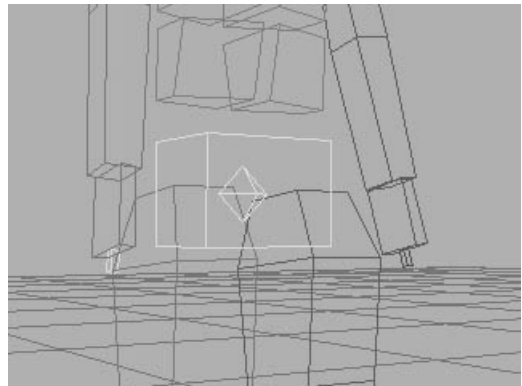
*Biped* (see page 723) is a 3D Studio MAX system plug-in that you access from the Create panel. Once you create a biped, you animate it using the Biped controls on the 3DS MAX Motion panel. Biped provides the tools you need to design and animate the motion of characters.

### The Biped

The biped structure has a number of properties designed to help the animator.

- Joints on the biped are hinged to follow human anatomy. By default, the biped resembles a human skeleton and has a very stable Inverse Kinematics hierarchy. This means that when you move a hand or foot, for example, the corresponding elbow or knee will orient itself accordingly, and produce a natural human posture.
- When you rotate the biped spine, the arms maintain their relative angle to the ground, rather than behaving as if they were fused to the shoulders. For example, if the biped is in a standing position, arms hanging at its sides, when you rotate the spine forward, the biped's fingers will touch the ground rather than point behind it. This is a more natural position for the hands, and this speeds the process of keyframing the biped. This feature also applies to the biped head. When you rotate the spine forward, the head maintains a forward-looking orientation.
- Using the Move transformation tool on the 3DS MAX toolbar allows you to position the biped arms, legs, and center of mass.
- Using the Rotate tool on the 3DS MAX toolbar allows you to rotate any biped object.
- Using the *Bend Links* (see page 722) tool will rotate all of the spine objects simultaneously. This helps you to create posture keys for the biped. Biped playback is fast even on slower systems, allowing you to judge motion accurately.
- If you want to keyframe your character with the mesh visible, use the 3DS MAX Freeze command to keep the mesh visible but unselectable. As you keyframe the underlying biped, the mesh follows the biped to animate the character, allowing you to see exactly how the mesh will behave as the biped limbs are positioned.

### Biped Center of Mass



Moving or rotating the Biped center of mass (COM), an object shaped like a diamond in the pelvis area, positions or orients the entire biped. The center of mass uses three tracks to store motion keys: *Body Vertical*, *Body Horizontal*, and *Body Rotation*.

Body Vertical and Body Horizontal keys have additional parameters for gravity and balance, referred to as *Biped\_Dynamics*.

Body Vertical keys have Dynamics Blend and Ballistic Tension, respective parameters that

control gravity and the amount of knee bend when a biped lands after an *airborne period* (see page 721). These parameters only apply for lift-off and landing keys on either side of an airborne period. In an animation containing footsteps, an airborne period is an interval of time when neither of the biped feet is on a footstep. In a footstep animation using Biped Dynamics, you won't have to create Body Vertical keys for the apex of a jump or for the lowest part of the landing period; the Biped Dynamics will automatically calculate a trajectory for the center of mass. Gravity plays no role when a character is walking. In an animation containing footsteps, walking is defined as the biped having at least one foot on a footstep at all times.

Body Horizontal keys have a parameter called *Balance Factor* (see page 722). Balance Factor, whose parameters are stored in the center of mass horizontal track, helps the animator by automatically shifting all of the biped limbs to naturally position the character's center of mass, over the feet when the spine is rotated. You can actually see the biped pelvis moving away from the center of mass object if the spine is rotated. This relieves the animator of having to set extra keys to simulate the character's balance. Balance Factor positions the biped for you.

## Keyframing the Biped

There are two primary methods used in creating biped animation: *Footsteps Method* (see page 725) and *Freeform Method*. Each method has advantages, and you can convert from one method to the other, or you can use a combination of both techniques in a single animation.

## Footstep Method

In the viewports, footsteps represent support periods in space for the biped feet. Moving or rotating footsteps in space is done in the viewports. In Track View, each footstep appears as a block that represents a support period in time for each of the biped's feet. Moving footsteps in time is done in Track View. The footstep position and orientation in the viewport controls where the biped will step.

There are three ways to create footsteps for the biped. The first way is to place footsteps individually, or one at a time. The second way is to invoke Biped's multiple footstep creation tools to create a walk, run, or jump animation. The third way is to extract footsteps from raw motion capture data.

A key advantage of the footstep method is the natural adaptation of the biped that occurs when the footsteps are edited in time and space. Editing footsteps in the viewports allows you to reposition all of the footsteps to move the entire animation. In Footstep mode, stride, length, width, and direction can be changed quickly for an entire animation and the biped automatically adapts. Using the Footsteps Show/Hide button on the Display rollout, all footsteps can become visible. Move the footsteps in the viewports to position them for proper ground collision with the terrain object. For example, if the biped toes are rotated for the Lift key at the last frame of a footstep (to create more toe curl as the character walks) the leg automatically repositions itself to maintain foot contact with the ground (footstep).

These adaptations speed up the process of creating and editing animation for the biped. If necessary, the animator can prevent biped adaptation by using the *Adapt Locks parameters* (see page 721) on the Animation Properties rollout.

## Foot States

Within a footstep animation, there can be four *foot states* (see page 725): *move*, *touch*, *plant*, and *lift*. These correspond with the state of the biped feet in relationship to the footsteps. Use the foot state displays in the General rollout to determine the state of the biped feet when you are editing the biped foot or leg keys. Foot states are used on the Set Multiple Keys dialog to select multiple keys in Track View.

## Freeform Method

A freeform animation contains no footsteps; instead it relies on the transforms of the biped body objects and center of mass. Use Freeform for motions like swimming or falling where footsteps are not necessary. If you are a familiar with creating all of your keyframes manually to animate a character, you may want to use the freeform method exclusively.

To start a freeform animation, turn on the 3DS MAX Animate button and start positioning the biped. Or you can leave the Animate button off and use the red set key buttons to create keyframes.

You can also create freeform animation by importing motion capture data and choosing freeform rather than footstep.

**Tip:** Take advantage of both methods by combining footsteps and freeform animation. You can create a freeform period for any airborne period between footsteps. A freeform period replaces the ballistic motion calculated using the

GravAccel value with a user defined spline motion.

If you are using footstep extraction with motion capture data, you often need a freeform interval to accommodate falling or tumbling motion in the data. The Fit to Existing option on the Motion Capture Conversion Parameters dialog allows for a combination of both methods. Extracting footsteps from motion capture files eliminates sliding feet, a common problem with motion capture data.

Note that while you can add a freeform period to a footstep animation, you *cannot* add a footstep period into a freeform animation. If you want to add a footstep animation to an existing freeform, you can use the motion flow editor to create a script that sequences the footstep with the freeform.

## Inverse Kinematics

Footstep and Freeform animations use the same IK constraints and extensions. This means that in a footstep animation, you can now edit keys to change footstep duration. By definition, a footstep is the start and end of a sequence of IK constraints in World Space with an IK Blend value greater than 0. Deleting and inserting keys or changing IK space or IK blending alters *footstep duration* (see page 500).

In cases involving edits that alter the length of ballistic intervals (when a biped is airborne), the software ensures that there is a vertical key occurring at the lift-off and touchdown frames. This calculates the correct ballistic motion, so vertical keys are automatically inserted if not present.

There are three types of IK keys you can create: *planted*, *sliding* and *free* keys.

- **Planted keys** have an IK Blend of 1. They are joined to the Previous IK Key and are in Object space, rather than Body space. Planted keys lock the foot or hand to the ground, or to any object.
- **Sliding keys** have moving IK constraints. Sliding footsteps are created if there is a *moved* IK constraint occurring in the footstep interval. In a footstep animation this means that the foot can be placed anywhere, even though there is a footstep icon. Footstep icons can be thought of as gizmos rather than the absolute location of a foot. Sliding keys have an IK blend of 1, and are in object space, but are not joined to the previous IK key.
- **Free keys** have an IK blend of 0, and are in body space. They are not joined to the previous IK key. Free keys have no IK constraint.

IK constraints are implemented with a pivot-based system. This allows you to pivot a hand or foot around a selected pivot. For example, in a walking motion you can select a pivot on the heel of a foot and rotate the foot around it. You can then shift the pivot to the ball of the foot.

## Key Interpolation

The way that *Biped Dynamics* (see page 723) affect the biped center of mass is discussed earlier in this topic. *Spline Dynamics* (see page 732) is an alternative method of keyframe interpolation. If you are starting a new animation, use the *Animation Properties* rollout (see page 181) to select the method you want to use. Spline Dynamics will be most familiar to new users. Selecting this method and creating new footsteps generates Body Vertical keys using Spline Dynamics for

interpolation; Balance Factor and Gravity are turned off. You can convert from one method to the other by selecting one or all of the Vertical Center of mass keys in Track View and setting the Dynamics Blend parameter to 1 for Biped Dynamics, or 0 for Spline Dynamics.

## Productivity

*Motion Capture Import* (see page 729), *Motion Flow Mode* (see page 729), and Footstep creation tools are designed to increase your output.

- **Motion Capture** (see page 186). Typically, you will not use a complete motion capture file in its entirety unless you specifically capture sequences you need for your production. Once you are familiar with the motions in the files that you have, you can join together the motions that you need to create a specific animation using Motion Flow Mode.
- **Motion Flow Mode** (see page 229). Cutting animation together is a fast way to create the animation you want. For example, you can splice just the stretch motion in one file into the walk motion of another file. If two characters are interacting, load both characters into your scene, then perform Motion Flow editing on one character while the other character's motion is visible in the viewports.

To save your script motion into one long *.bip* file, use Create Unified Motion in the motion flow script rollout. When you leave Motion Flow mode you will see the keys for the whole animation, now as one single sequence. Or use Save Segment to generate a single long *.bip* file and save it in a single step.

- **Footstep Creation.** To generate a motion, select Footstep mode and use controls on



the *Footstep Creation* rollout (see page 248) to quickly generate a walk, run, or jump pattern. These motions typically need editing to produce more than a generic motion. However, they will provide a good starting point when creating a new animation. For example, to create your own running motion, generate a default run in Footstep mode. Then use Set Multiple Keys, on the Keyframing rollout, in conjunction with Track View to create or manipulate keys that will adjust the motion to your liking. Track View in conjunction with Set Multiple Keys is used when many keys need to be adjusted simultaneously.

- **Layers.** You can use the layers rollout to make adjustments to your animation. Create a new layer and make any freeform changes you like to your existing biped animation. This is non-destructive, you can always turn off the layer to return to the earlier animation. This is very useful for experimentation when you're not sure what direction you want to take. Layer animation can be collapsed into a single layer when you want to commit to it. Layers let you build infinite variety from one set of motion capture; you can add secondary motion without losing your base animation.

### See also

*Create Panel* (see page 143)

*Motion Panel* (see page 146)

*Track View* (see page 262)

## Physique Features



Physique enables you to depict the intricate changes in musculature that occur as a character moves, by providing you intuitive and comprehensive control over muscle movement and skin blending over joints. The following is a brief summary of significant Physique features and benefits.

### Skin Attachment with Envelopes

Attach skin quickly and intuitively with 3D deformation envelopes that show exactly how each character's bones will affect the surrounding skin. Stretch, shift, and shape each envelope independently to enclose the unique shape of your character's skin. Save time when modifying or replacing your character's skin by reusing envelope settings with modified skin geometry.

### Weighted Blending

Ensure that skin moves evenly and naturally around joints. Automatic weighted blending works with any number of overlapping deformation envelopes. Fine-tune skin behavior by adjusting the amount of weighting applied by each envelope. For video game characters,

you can select basic blending to preview real-time game characters.

## **Skin Sliding**

Pinch and stretch your character's skin interactively to create creases or remove bunching effects around joints. Tune sliding separately for each joint on both the inner and outer sides of each joint.

## **Muscle Bulging**

Add animated bulges along arbitrary bones like the arms and legs to simulate subtle changes in shape with character movement. Physique generates muscle bulges based on the behavior of the skeleton's joint angles and their relationship to one another. Define a muscle's profile at any given point using Physique's Bulge Editor to give your model as much detail as you wish for any position. Define overall muscle movement by establishing the relationship between various muscle cross sections and the joint bend angle. Control the influence, weight, and power of the animated bulge as joints bend.



## **Tendon Effects**

Control tendons for realistic effects by linking muscle movement to arbitrary bones and deforming the muscles across multiple links. Physique selectively assigns portions of skin to move with individual bones. Pull skin to follow limb motion using angular pull to duplicate the effects of movement, such as the chest rising as the arms are lifted upward. Pinch skin to follow limb bending, such as the pectoral muscles protruding forward as the shoulders are shrugged. Stretch skin as other limbs move, using radial stretch to duplicate effects such as tendons appearing when muscular characters lift weights.

## **Support for Patches NURBS and Polygons**

Choose the geometry type that suits your character best. Physique operates seamlessly on all 3D Studio MAX surface types, including NURBS, surface patches, splines, polygons, and *freeform deformations* (FFDs) (see page 725). If you change your mind after you've begun deformation envelopes make it easy to change your character's skin shape or structure without sacrificing your initial setup time.

## **Using Splines with Physique**

You can use splines to deform and animate 3D Studio MAX geometry just the same as you use a biped armature. Splines can act just like bones for easy animations of tails or creating facial expressions.

## Using Non-Hierarchical Bone Structures with Physique

Additional bones, splines, and hierarchies can be added to the Physique modifier without having to recreate the assignments or reinitialize in Physique. Bones and additional hierarchies do not need to be added to the biped structure for physique to use them.

## Performance

Physique performance is greatly improved in version 3. Some operations are six times faster than previous releases.

## See also

*Physique Rollout (see page 296)*

*Physique Level of Detail Rollout (see page 298)*

*Envelope Sub-Object (see page 300)*

*Link Sub-Object (see page 307)*

*Bulge Sub-Object (see page 313)*

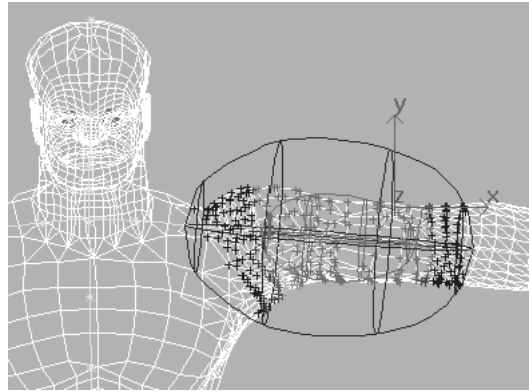
*Tendons Sub-Object (see page 326)*

*Vertex Sub-Object (see page 331)*

*General rollout (see page 147)*

*Figure Mode (see page 156)*

## Understanding Physique



Physique is a modifier that when applied to a mesh, allows the movements of an underlying skeleton to seamlessly move the mesh like bones and muscle under a human skin. Physique will work on any point-based objects including geometric primitives, editable meshes, patch-based objects, NURBS, and even FFD space warps. For NURBS and FFDs, physique deforms the control points, which in turn deform the model. It will attach to any skeleton structure including a biped, 3DS MAX bones, splines, or any 3DS MAX hierarchy. When you apply Physique to the skin object(s) and attach it to the skeleton, Physique determines how each component of the skeleton influences each vertex of the skin based on settings you specify.

Physique affects a mesh after the Attach to Node command on the Physique rollout is used and a root node is selected in the viewports. During the attach process, Physique works its way through all of the children in a hierarchy starting at the object you select and creates its own links with associated envelopes for each link it finds. The links created by Physique are referred to within this documentation as the *Physique deformation spline*. Vertices that fall

within envelopes are influenced to follow the links and animate the mesh. Splines and 3DS MAX bones can also be added using the Add command in the Physique Bones rollout.

## Biped and Physique

When the Physique Attach to Node command is turned on and the biped pelvis is selected in the viewports, Physique traces its way from the pelvis down the legs to the toes. From the pelvis it also traces its way up through the spine and branches at the collar to the arms, hands, and fingers and up the neck to the head. A link and associated envelopes are created for each link found. If any other objects, including 3DS MAX bones, are linked to the biped, Physique treats them similarly; it creates a new link and envelopes. Keep this in mind when you use Physique to attach a mesh to the biped; if your character has additional limbs, link 3DS MAX bones to the biped for the extra arms before using the Attach to Node command in Physique to create links and envelopes. When Physique is applied (Attach to Node is used), it creates links and envelopes for all the links in the biped as well as for the linked bones. Objects that should not deform like skin but need to be linked to the biped, like a sword, should be linked after the Physique Attach to Node command is used to link a mesh to the biped. This way a link and envelopes will not be created for the sword.

## Envelopes and Weighted Vertices

Envelopes are Physique's primary tool for controlling skin deformation; tendons and bulge angles are used to fine tune mesh deformation after envelopes are adjusted. All envelopes have an inner and outer bound (boundary). Any vertices falling within the inner bound of a single link receive a full weight of 1.0 from that link. Any vertices falling outside the

outer bound receive no weight from that link. Vertices falling between the inner and outer bounds receive some weight between 0 and 1. Vertices move together with the link that influences them. Where multiple envelopes encompass a vertex, that vertex receives weight from each envelope and follows each link to an average position based on these weights. This weighting from multiple links is considered *blending*.

It is possible that weights assigned to some vertices don't reach a total weight of 1.0 or greater. Rather than leaving these vertices behind, Physique by default normalizes them to a value of 1.0.

Adjusting falloff, overlap, scale and other envelope parameters changes vertex weight distribution across links. This, in turn, changes the way skin behaves as the biped moves. Much of the work in correcting the way skin deforms on a character is done by adjusting envelopes.

## Deformable and Rigid Envelopes

There are two Envelope types per link, *deformable* (see page 724) and *rigid* (see page 725). Deformable envelopes follow the Physique Deformation Spline that runs through the joints in the hierarchy and can be deformed using bulge angles, tendons, and link parameters. Rigid Envelopes determine vertex-link assignment based upon the linear 3DS MAX link and move in an immobile relationship to the link. Vertices in a rigid envelope, however, are deformed (blended) in the overlap area of other envelopes.

Typically you use deformable envelopes; however, game developers with game engine restrictions may want to exclusively use rigid envelopes. Both rigid and deformable envelopes can be turned on for the same link. For example,

by scaling both envelopes, you could deform the shoulder with a rigid envelope and the armpit with a deformable envelope.

## The Number of Links that can affect a Vertex

Any number of overlapping envelopes (*N Links* (see page 729)) can influence vertices. Normally, N Links are preferred. For special purposes such as games requirements, you can limit the number of links (envelopes) that can affect a vertex. The No Blending parameter is similar to the method used in version 1 of the software; a vertex is assigned to only one link.

## Physique Workflow

Before Physique is applied, align the biped to the mesh in *Figure mode* (see page 725). Use a pose with the arms outstretched so the hands are away from the torso. Save a figure file, so it's easy to return to this pose whenever you need. Select the mesh and choose Physique in the Modify panel. Turn on Attach to Node and select the root node in the hierarchy (biped Pelvis or root node in a bones hierarchy, not the COM). In the Physique Initialization dialog, click Initialize to create default envelopes based on the links in the hierarchy. The remainder of the work is adjusting envelopes and optionally adding bulge angles and tendons.

Envelope size, overlap, and other parameters are adjusted with the character in an animated position (turn off Figure mode); by scrubbing the time slider back and forth, problem areas can be spotted and the envelopes affecting the problem areas adjusted. In Place mode is useful to keep the character in place during envelope adjustment.

Link parameters, Bulge angles and tendons are the finishing touches. Skin sliding, the amount

of twist, and crease blending as a character moves are controlled using link parameters. Bulge angles are used to bulge areas like the biceps, legs and chest relative to the angle created by a link and its child in the hierarchy. Tendons can span multiple links in the hierarchy to stretch and pull a character's skin.

## See also

*Physique Rollout* (see page 296)

*Physique Level of Detail Rollout* (see page 298)

*Envelope Sub-Object* (see page 300)

*Link Sub-Object* (see page 307)

*Bulge Sub-Object* (see page 313)

*Tendons Sub-Object* (see page 326)

*Vertex Sub-Object* (see page 331)

*General rollout* (see page 147)

*Figure Mode* (see page 156)



## Overview of Crowd Features

Crowd animation, one of the most important new features in **character studio 3**, lets you simulate the behavior of groups of people, animals, and other animated entities parametrically, using several different types of 3DS MAX objects. The crowd system lends itself to animation of a wide range of group behaviors, from a flock of birds flying around a church steeple to a group of sports fans exiting an arena. By using procedural animation you can create autonomous characters who animate themselves based on chosen conditions. Specific behaviors include *Seek* (see page 732), *Avoid* (see page 722), *Path Follow* (see page 730), *Surface Follow* (see page 735), *Repel* (see page 731), and additional behaviors can be added using MAXScript.

## Basic Setup with Crowd and Delegate Helper Objects

To get started with crowd animation, you add a *Crowd helper object* (see page 346) and one or more *Delegate helper objects* (see page 342) into your scene. The Crowd object directly controls delegates, which are stand-ins for the members. Crowd lets you collect the delegates into teams, link them with bipeds or other animated objects, such as animals or vehicles, and apply behaviors such as Seek and Avoid. It will distribute the member's local animations to match the behaviors. The Delegate object's controls include speed, acceleration and deceleration, and turning-angle ranges.

## Scatter Objects

The Crowd object's Scatter Objects functionality lets you clone objects and hierarchies and distribute them at arbitrary positions throughout volumes, on a grid, or over surfaces. You can orient clones toward a specific object or along a line between two objects with optional random variation. You can also specify scaling on any or all axes, again with optional randomization. The All Ops feature lets you apply randomization repeatedly to any or all of Scatter Objects functions until you get a configuration you like.

## Crowd Behaviors

You can apply any number of behaviors to delegates, weighting each for priority. You can animate weights as well.

Following is a list of available behaviors:

- **Avoid Behavior** (see page 392). Prevent delegates from colliding.
- **Orientation Behavior** (see page 396). Applies a fixed orientation or orientation range to delegates, so they face a specific direction instead of toward the destination.
- **Path Follow Behavior** (see page 398). Restricts motion to a spline or NURBS curve; options include back-and-forth “patrol” movement.
- **Repel Behavior** (see page 400). Forces delegates to move away from a target.
- **Scripted Behavior** (see page 402). Uses MAXScript to specify behavior.
- **Seek Behavior** (see page 403). Delegates move toward a target or targets.
- **Space Warp Behavior** (see page 404). Use any dynamics-oriented space warp to control movement, including wind and gravity. Included is *Vector Field* (see page 738), a crowd-specific space warp that lets delegates avoid irregularly-shaped objects while following the objects' contours.
- **Speed Vary Behavior** (see page 405). Lets delegates change speed for more realistic movement.

- **Surface Arrive Behavior** (see page 406). Delegates move toward a surface using an intermediate goal (offset), and then land on the surface using specified final-approach parameters.
- **Surface Follow Behavior** (see page 409). Delegates move along a surface, which can be animated. Also, you can specify whether the delegates are to move straight ahead or skirt hills and depressions.
- **Wall Repel Behavior** (see page 411). Uses a grid to repel delegates; ideal for keeping objects inside an enclosed, straight-sided room.
- **WallSeek Behavior** (see page 414). Uses a grid to attract delegates. You can use this as a doorway for crowd-controlled bipeds to walk through.
- **Wander Behavior** (see page 416). Induces a realistic semi-random movement for characters such as shoppers at a mall.

## Cognitive Controllers

*Cognitive controllers* (see page 723) let you assign a number of behaviors to a delegate or delegate team, and switch between them based on transition conditions you define. For example, a flock of birds could be flying toward a distant goal at one moment, and then fleeing from a perceived threat at the next. Transition conditions use MAXScript; we provide a number of examples for you to learn from and adapt to your own needs.

## Global Clip Controllers

*Global Clip Controllers* (see page 726) are useful when controlling non-biped animated objects with a Crowd object. They let you specify that an object is to blend automatically between existing animation clips based on criteria such as

speed, acceleration, pitch, pitch rate, and heading rate. For example, you could set a bird to flap its wings at a certain rate as it flies straight ahead, faster as it flies upward, and not at all as it glides downward. The Global Clip controller senses its pitch and automatically blends between animation clips depending on the angle. As with Cognitive controllers, you can also specify transitions with MAXScript snippets.

## Motion Synthesis

You can incorporate bipeds into a crowd simulation by combining delegate behavior with biped animation. In *motion synthesis*, any number of bipeds share a shared motion flow graph; each biped takes a unique path through the graph based on the actions of its associated delegate. Find more information and further references at *Using Bipeds with Crowd Delegates* (see page 116).

---

## Understanding character studio Workflow

Biped, Physique, and the Crowd system work together within 3D Studio MAX to provide a complete set of character animation tools. Although these plug-ins can be used in a variety of ways, it is helpful to approach **character studio** with a basic understanding of how a typical character animation is created.

The following sections provide a brief summary of the basic workflow and related benefits to creating a character with Biped and Physique. You may not use all the following steps, but you're likely to do them in the following order.

## Create Skin Geometry

Create a basic skin shape for your character using any of 3D Studio MAX modeling tools and surface types. Be sure to place your character's skin in a neutral pose with arms outstretched and legs spaced slightly apart. You may also want to add sufficient detail to your skin's mesh or control points around joints to facilitate deformation during movement.

Note: Since Physique deformations are based on a volume, you can refine your geometry at a later time with minimal impact on your skin behavior. This means you can create your animation before you've built your model if you wish.

## Create a Biped Skeleton

Biped automates the creation of bipedal character skeletons. It also lets you introduce significant changes to the skeleton's structure and sizing at any point during your animation without adversely affecting your character's motion. This means you can animate your character without knowing if it is short or tall, skinny or fat. It also means that if the director changes the character's proportions, the animation will still work.

## Attach the Skin

- Position the Biped character within its modeled skin. Use Biped's special *Figure mode* (see page 725) tools to scale bone lengths and to orient the Biped skeleton exactly within the skin's volume. Scale bone thickness as desired to achieve a good initial fit. Then save a figure file, so it's easy to return to this pose.
- Use Physique to attach the skin geometry automatically to the biped or a 3DS MAX Bones hierarchy. The attachment is typically

made to the root node in the hierarchy: the pelvis object on the biped or the root node on a Bones hierarchy, not the center of mass. The attached skin is deformed as the biped or Bones hierarchy moves.

- The links in the Bones hierarchy are used to create a system of 3D envelopes that enclose nearby vertices. Envelopes typically overlap at the parent and child ends of links. Vertices within overlapping envelopes are blended to create smooth skin deformation over joints as the character moves.

## Adjust Skin Behavior

Adjust Physique parameters and introduce skin behavior effects to achieve the desired characterization.

- Change default envelope shapes by adding cross sections and control points to isolate a more specific volume of vertices for each bone. Or use exclusion lists or per-vertex weighting to apply fine-tuning control over individual vertices.
- Introduce bulge angles to change muscle shape based on the angle of a particular joint. Create tendons to simulate the motion of tendons under the skin, based on link movements.
- Adjust link parameters to change skin twisting, sliding, and creasing as the biped moves. Sliding allows the skin to compress at the biceps and forearm as the elbow is bent. Twisting controls the amount of skin twisting across a joint intersection.
- Create extra links using 3DS MAX bones and dummy objects for added control. Links can be created for the abdominal area to control compression, for example, or to create a link to animate the chest rising and falling as the character breathes. If a



character has extra appendages, 3DS MAX Bones can be added to animate them. One common usage is to add a bone to animate the jaw.

## Animate the Biped Skeleton

Once the skin is attached to a Biped structure, you can freely animate the Biped character and see skin behavior update automatically, based on a current pose. Since Physique skin deformation can slow visual playback of your Biped animation, you can temporarily hide the skin object or reduce its resolution in the modifier stack to improve performance.

You can also choose to develop Biped animations in a separate scene entirely, and apply them to your final skinned character when you are satisfied with your final motion.

A *Biped* (see page 723) character is essentially an integrated hierarchy of bones that you can position freely using keyframes, IK goals, and footsteps. You can position a Biped character using all the rotation and transformation tools found on the 3DS MAX toolbar.

Many of the 3DS MAX coordinate systems can be used to position the biped. Local coordinates are useful to move a limb along its axis (local X is always the axis along the Biped limb); world coordinates are handy when there is any confusion regarding which way is up. You can use world coordinates as a home base. In 3DS MAX the world Z axis is always up.

Note: Rotations always occur about the local axis.

## Use Freeform Techniques

Biped provides a variety of methods for creating character motion easily. You can start with a purely traditional approach by manually creating keyframes in freeform mode for

different poses, letting the computer interpolate between joint positions and IK goals.

## Use a Footstep-Driven Technique

You can also choose a partially assisted approach by using footsteps and Biped dynamics to help you create a default walk, run, or jump cycle, and then adjust the biped keyframes and footsteps individually.

When using footsteps, Biped dynamics helps you by simulating gravity and balance.

- Gravity can help in a jumping motion to accelerate a character during the falling period and to bend the legs naturally on landing.
- Balance adjusts a character's position when the spine is rotated and keyframed to retain a character's balance.
- Dynamics can be turned off on a per-key basis or for the entire animation. The animator can override center of mass keyframes created using Dynamics calculation at any time. Simply set the dynamic properties of these keys or chose *Spline Dynamics* (see page 732) for keys generated by newly created footsteps.

## Convert Between Animation Types

Once you are satisfied with a particular footstep animation and its corresponding dynamic behavior, you can convert it automatically to a freeform animation consisting of a simple combination of keyframes and IK goals. This intelligent conversion gives you control of animation behavior at every frame, for every joint of the character.

## Use Layers to Apply Global Changes

*Biped Animation Layers* (see page 721) offer you a powerful tool for introducing global changes to an existing character animation. For example, by adding a layer on top of an existing run motion, and creating a single keyframe with the biped's spine rotated forward, an upright running motion can be turned into a crouched run. Layered changes can be stacked up, allowing you to refine your motion composition and eventually collapse your layers into a standard non-layered keyframe animation.

## Use In Place Mode to Control the View

*In Place Mode* (see page 727) is a general tool that you can use to keep your Biped character in view during animation playback. It offers a convenient way of adjusting and adding keyframes to a character without constantly changing your view to follow the character's motion.

## Import Motion Capture Files

Motion capture files can be imported from the .BVH or .CSM formats, edited, and saved as .BIP files. These files can be imported with or without footsteps and dynamics and can be combined in Motion Flow mode with other animations.

You can use the supplied motion capture samples as is or adjust them to suit your needs using Biped's collection of animation tools. The ability to import motion capture marker files directly into **character studio** using the .CSM file eliminates much of the cost of post-processing optical motion capture data. You can import motion capture files with an additional prop bone, to define the motion of an object such as a sword or club.

Motion Capture Files can be imported with key reduction which makes for more manageable tracks for subsequent editing.

## Use Track View for Keyframe Editing

Track View allows you to edit keys and footsteps relative to the animation time line. Periodic motions can be adjusted. If a character nods its head from left to right, you can select all the appropriate head keys and adjust them at the same time using controls in the Biped Multiple Keys dialog.

Footstep editing in Track View allows you to move footsteps in time. If a character needs to jump higher between footsteps, move the landing footsteps further down in time, dynamics automatically compensate by making the character jump higher to keep the character airborne for a longer time period.

You can specify a freeform period in a footstep animation, using Track View. This allows you to take advantage of footsteps and dynamics for part of an animation and switch to manual keyframing during the freeform period. This approach can be particularly useful in animations where there is a mix of animation with the feet are on the ground and then off. Examples of this type of animation include running and diving, or walking and then sitting down.

Keyframe adjustment tools allow you to find the next or previous key for the selected biped body part, use the Time spinner to slide a key back and forth in time, change Tension Continuity and Bias for a key, and to display trajectories.

You can place arms and legs into the *coordinate space* (see page 724) of another object or the world to simulate interaction with fixed or moving objects. In Freeform mode, for example, putting the character's legs into *world space* (see

page 739) prevents them from sliding or moving when the animator is keyframing the character's center of mass. Putting a character's hand in the coordinate space of a ball allows the hand to move wherever the ball moves.

Many tools in 3DS MAX can be leveraged with **character studio**. For example, the 3DS MAX Link tool can be used to attach objects to the biped. If a character is mechanical and needs no skin deformation, its body parts can be attached to the biped using the Link tool.

If a character is to pick up and carry an object and then put it down, the Link Controller can be used to animate the duration of the attachment. 3DS MAX bones can be used to animate character sub-assemblies, like pistons, and to create extra links for Physique.

## Use Motion Flow to Combine Animations

After you have created and modified various animation sequences, and stored them in Biped motion files (.BIP format), you can use Motion Flow mode. This feature allows you to combine various Biped motion files into longer animations that can be quickly previewed and edited. *Motion Flow mode* (see page 729) automatically places the animations end-to-end, allowing you to mix and match both freeform and footstep-driven motion files. Transitions between successive motions are automatically created for you, to provide a first-pass blending between overlapping frames of animation.

The Motion Flow transitions use *velocity interpolation* (see page 738) to create seamless transitions between clips. You can use the transition editor to modify a variety of blending parameters, including transition start frame, length, and angle between clips.

## Refining Your Character

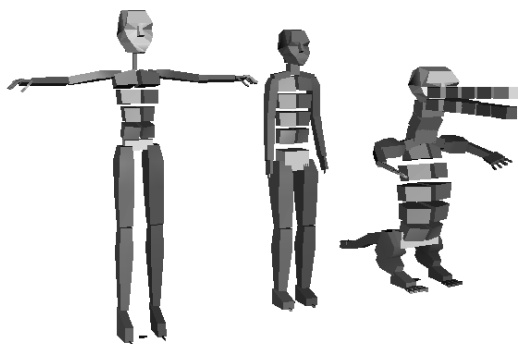
Great character animation is a result of many refinements that tune the overall personality of your character. You will find the need to refine progressively both the skinning behavior and the animation timing of your **character studio** character. Biped and Physique make this iteration process straightforward by fully utilizing 3DS MAX modifier stack and undo methods.

In addition, Biped's ability to map motions between characters makes it very easy to interchange great animations with existing characters, and tune their behaviors to achieve true integration of motion with character motivation and personality.

## Use Crowds to Animate Groups of Characters

Once you've created animation sequences for characters or other models (such as a bird flapping its wings), you can replicate the models or characters and apply the motions to these groups using the Crowd system. You can also combine them with a wide range of supplied behaviors to create lifelike activities in crowds, such as people streaming through a doorway, street traffic, or birds and fish flocking and avoiding obstacles. You can use Motion Flow mode to create motion clip networks so that characters perform animation sequences appropriate to their current movement and transition smoothly between sequences. And you can use Crowd's cognitive controllers to transition between behaviors based on a variety of criteria. For more on Crowds, see *Overview of Crowd Features* (see page 13).

## Creating a Biped Character



A biped model is a two-legged figure: human or animal or imaginary. Each biped is an armature designed for animation, created as a linked hierarchy. The biped skeleton has special properties that make it instantly ready to animate. Like humans, bipeds are especially designed to walk upright, although you can use bipeds to create multi-legged creatures. The joints of the biped skeleton are limited to match those of the human body. The biped skeleton is also specially designed to animate with **character studio** footsteps, which help solve the common animation problem of locking the feet to the ground.

The parent object of the biped hierarchy is the biped's center of mass object, which is named *Bip01* by default.

### Starting Biped

Biped

After you have installed **character studio**, a button for creating bipeds appears in the Create panel under Systems. The first time you click this button, a dialog asking for authorization appears. See *Authorization* (see page 146), if you need information on authorizing Biped.

### Creating a Biped

Once you have completed the authorization, you are free to create bipeds by clicking and dragging in the active viewport. You interactively define the height of the biped as you move your mouse. For a step-by-step procedure to create a biped, see *To Create a Biped* (see page 143)

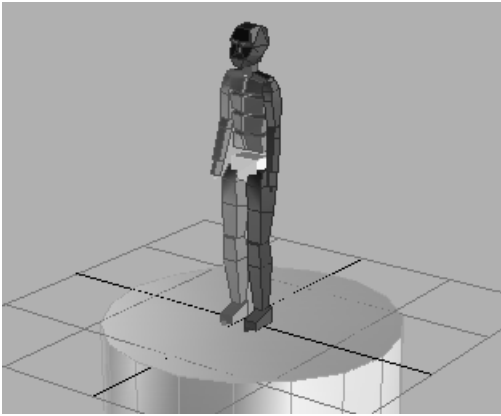
During biped creation, you can change any of the default settings that are used to define its basic structure. The default settings—Arms, Neck Links, Spine Links, and so on are for a human figure.

If you turn on Most Recent .fig File, the biped you make will use the parameters stored in the last .FIG File you've loaded.

### Changing Biped Parameters

Like other 3D Studio MAX objects, you can change biped parameters in the Create panel at creation time. However, to modify or animate a biped you access parameters in the Motion panel. For more information on changing biped parameters, see the *Structure rollout* (see page 184).

## Creating a Biped Using AutoGrid




The Autogrid feature lets you create objects on top of other objects, rather than on the home grid. Autogrid automatically creates a grid for construction. When you turn Autogrid on, a tripod cursor is displayed in the viewport. As you move your mouse over the geometry in the scene, the tripod cursor shifts to match the face position. A grid is created at that spot and is used to create the biped.

By turning on AutoGrid under Systems the Biped creation will respect the Autogrid, letting you build bipeds with their feet on top of geometry. You can also use Autogrid to place footsteps on uneven terrain.

### Procedures

#### To create a biped on a surface

1.  On the Systems panel, click Biped.
2. Turn on AutoGrid.


3. Move your cursor over any geometry in the viewport.


A Transform gizmo moves with your cursor to indicate the location of the AutoGrid.

4. Drag out a biped.

The biped feet will be in contact with the geometry.

#### To create footsteps using AutoGrid

1.  On the Systems panel, click Biped.
2. Turn on AutoGrid.

3.  On the Motion Panel, turn on Footstep Mode.

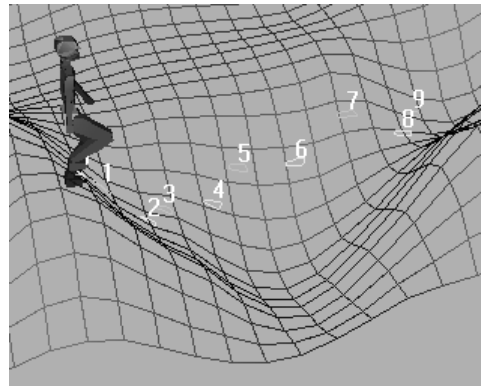
4. In the Footstep Creation rollout, choose Create Footsteps ( at current frame).

5. Move the cursor over the geometry in the viewport.

A Transform gizmo moves with your cursor to indicate the location of the AutoGrid.

6. Click to place each footstep individually.

The footsteps will be placed in contact with the geometry.



## Naming the Biped

If your scene is going to contain more than one biped, it's a good idea to give the biped a unique name. By default, the first biped in a scene is called Bip01. Succeeding bipeds have the same name except that the two-digit number is replaced by another number in sequence: 02, 03, 04, and so on.

Since the parent of the biped hierarchy is the biped's center of mass object, that is also the object selected when you choose Bip01. You can link the biped center of mass to other objects or dummies for additional animation control. For more information on the biped's hierarchy, see *Understanding Biped Anatomy* (see page 22).

## Procedure

### To change the biped name

- On the Create Biped rollout, enter the new name in the Root Name field.

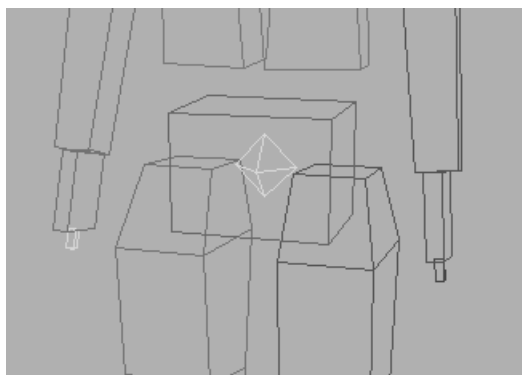
The center of mass is renamed and the entire biped hierarchy inherits the new name.

If you've exited the Create Biped rollout, you can also rename the biped using the Root Name field on the Structure rollout on the Motion panel.

**Warning:** Do not use the usual Name and Color rollout. This changes the name of only the biped's root object (its center of mass) without updating names in the hierarchy.

## Understanding Biped Anatomy

The geometry of a biped is a linked hierarchy of objects that by default resemble those of a human. The parent or root object of the biped is its *center of mass* (see page 723) (COM). This object is displayed as an octahedron near the center of the biped's pelvis. Moving the COM positions the entire biped.




### Center of Mass Object


You can select the center of mass by choosing *Bip01* from the select by name dialog. You can also select the center of mass by choosing either the Body Vertical, Body Horizontal or Body Rotation buttons in the Track Selection Rollout. These will select the center of mass as well as the particular track you wish to change.

## Using Animation Tracks


The center of mass uses three animation tracks. Two of these tracks, Body Vertical and Body Horizontal, contain Biped Dynamics options.

-  The Body Vertical track uses the Dynamics Blend parameter to control gravity in a footstep animation. A Dynamics Blend value of 1 uses the value of *GravAccel*

(see page 726) (global gravity value) to calculate an airborne trajectory for the biped. No keyframes are required to position the biped in the air, a trajectory is calculated automatically. A value of 0 uses Spline Dynamics for the vertical position of the biped; you must create keyframes to position the biped vertically.

-  The Body Vertical track also has a *Ballistic Tension* (see page 722) parameter that controls how much the biped knees bend when the biped lands from an airborne period. This means that keys do not need to be created at the lowest position of the biped after landing, a trajectory is calculated automatically.

Note: Keys can be created manually to override the calculated trajectory during the landing period. However, vertical keys must be set to Dynamics Blend=0.0 in order to fully override the trajectory during the airborne period.

-  The Body Horizontal track has a *Balance Factor* (see page 722) parameter that automatically orients the biped to maintain balance. This saves the animator from having to reposition the pelvis when the biped leans forward, backwards, or sideways.

Note: *Biped Dynamics* (see page 723) parameters can be animated from no effect to full effect at each keyframe.

- If you are starting a new footstep animation, but would prefer to use *Spline Dynamics* (see page 732) rather than the default Biped Dynamics, then turn on Spline Dynamics in the Animation Properties rollout before creating any footsteps.

- If you are starting a freeform animation (no footsteps), only Balance Factor is available. Freeform animation uses Spline Dynamics for Center of Mass positions.

## Changing the Biped Hierarchy

The biped Hierarchy is a little different than your standard 3DS MAX hierarchy in that you can't delete any of the components of the skeleton. If you try to delete any of the biped skeleton, you delete the entire hierarchy. If you want to create a partial biped, for example a biped with no head, simply hide the objects you don't want to see.

## Repositioning Biped Body Parts

Certain Biped body parts can be repositioned in figure mode to suit different characters. You can move entire arm assemblies by selecting the clavicles and moving them up or down. You can also reposition the fingers, tail and ponytails as you like.

## Adding extra Biped Body Parts

To add extra legs, arms or other body parts you need to create 3DS MAX geometry for those parts, then link them to the biped hierarchy. You can use Snapshot to duplicate biped body parts to create these as well. In either case you will need to animate them with standard MAX rotations, biped IK will not be available on these extra parts.

---

## Changing Initial Biped Anatomy

Use the body parameters to change initial biped anatomy. The Body parameters are in the Create Biped rollout that appears in the Create panel under Systems when you create a biped.

Note: You can only change body parameters in the Create panel immediately after creating a biped. Once you leave the Create panel, these settings are still available, but from the Structure rollout in the Motion panel. The body parameters in the Structure rollout are enabled when the biped is selected and Figure mode is active.

### See also

*Create Panel (see page 143)*

*Structure Rollout (see page 184)*

---

## Deleting a Biped

If the character you've created is not what you need, and you want to start over, you can delete it from the 3D Studio MAX scene.

### To delete a biped

- Use any selection tool to select the entire biped or any part of a biped, and press DEL.

---

## Loading and Viewing Sample Biped Animations

**character studio** ships with many sample animations, or motion files, which you can load and view at any time. All you need is a biped in the scene, onto which the motion will be mapped. These sample motion files are stored in the program's internal .BIP format, and installed into directories in the `\cstudio\motions` path.

For a listing of the samples that ship with **character studio**, see *Samples (see page 290)*.

Motion files save all information about biped motion: footsteps, keyframe settings, the scale of the biped, and the active gravity (GravAccel) value. IK Blend values for the keys and Object Space settings are also saved. (The Object Space object itself is not saved with the .BIP file. If you are using Object space objects, save a .MAX file.)

### See also

*General Rollout (see page 147)*

### To load biped motion

1. Select the biped you want to animate, and make sure you are not in Figure mode.  
Note: When *Figure mode (see page 725)* is active the Load File option loads figure files. Anything done in Figure mode changes the basic shape and structure of the biped. When Figure mode is turned off, the Load File tool loads .BIP files, which animate the figure
2. Click Load File.
3. In the file dialog, choose the .BIP motion file to load, and then click OK.
4. The biped repositions itself in the scene, as it assumes the initial position of the animation file. You may need to use Zoom



Extents to see the biped after it is repositioned.

The Biped keyboard shortcut ALT + R sets the animation range to that of the currently selected biped. Since the length of the animation can change after loading a .BIP file, this key can be useful. Make sure that the Plug-in Keyboard Shortcut Toggle is active.

### To save biped motion

1. Select the biped whose motion you want to save, and make sure you are not in Figure mode.
2. Click Save File.
3. In the file dialog, enter a name for the motion file, and then click OK.



**Tip:** Any .Bip files that you plan to use in Motion Flow mode must be saved to a directory that Motion Flow Editor files (.mfe) can recognize. A location of the referenced .BIP files is saved in the .mfe file. If the .BIP file cannot be found, the program looks to the motion flow directory specified in `\plugcfg\biped.ini`. By default, this setting is:

MoFlowDir=<maxdir>\cstudio\scripts

If a referenced .BIP file cannot be found in its current location, you need to move it to the specified Motion Flow directory. You can change the location of this directory at any time by editing your *biped.ini* file with a text editor. The new directory will be used the next time you restart 3D Studio MAX. You can also add multiple search paths to your *biped.ini* file by repeating the *MoFloDir=* statement multiple times. The program will search the directories in the order they appear and will use the first instance of the file that it finds.

## Previewing Biped Motion

There are two types of animation playback available within **character studio**.

-  You can use the play animation button in the VCR playback controls to play biped animation, (just the same as any other 3DS MAX animation).  
Biped has a specialized stick figure playback mode that hides everything in the scene and just plays the biped motion.
-  Use the *Biped Playback button* (see page 723) in the Motion panel for this type of playback.

### To preview biped motion using Biped

1. Activate the viewport with the view you want to see.
2. Hide or show the bipeds you want to appear in the playback.
3. On the Motion panel, on the General rollout click Biped playback.

### To preview biped motion using 3DS MAX, do one of the following

- Drag the time slider.
- Click Play Animation.

Depending on your system, biped animation might not play back in real time using 3DS MAX viewport playback.

Biped always previews the existing animation range. Biped Playback responds to the parameters in the 3DS MAX Time Configuration dialog. If Real Time playback is selected in this dialog, Biped plays back at the current frame rate, sometimes skipping frames if necessary. If Real Time is turned off, Biped plays back as fast as it can, depending on the capacity of the graphics card installed on your system.

**Warning:** You might miss critical frames of your animation if Real Time is selected in the Time Configuration dialog.

Biped playback previews the motion of all existing, visible bipeds. If you hide a part or all of biped, the hidden biped or biped part does not appear in the Biped preview. Objects in the animation that are not bipeds will not appear in the Biped preview. If the 3DS MAX home grid is visible, a grid appears at Z=0 in the Biped preview.

While Biped controls are visible in the Motion panel, you can also press V to start or stop Biped Playback. Make sure that the Plug-in Keyboard Shortcut Toggle is active.

Note: Hardware acceleration has no effect on Biped playback. If you are using a hardware-accelerated display card, you may find 3D Studio MAX playback to be faster under certain circumstances.

---

## In Place Mode



*In Place mode (see page 727)*, allows you to display biped motion as if it were occurring on a treadmill. Regardless of the distance the biped covers under control of the current motion file, the biped stays within the active viewport when you've turned on In Place mode.

In Place playback prevents lateral (XY) movement of the biped center of mass during animation playback; vertical motion along the Z-axis is preserved. Biped limbs, footsteps and center of mass keys can be adjusted in this mode. When the center of mass is moved on the XY axes in this mode, the footsteps move.

Use this feature to view biped playback without requiring a follow camera. In this viewing mode, visible footsteps appear to slide under the biped.

For export to games, this feature is valuable since many game engines intelligently move the character's center of mass laterally according to game play. In Place mode makes it easy to view, tune, and export animation in a manner that is complimentary to game engine playback.

Note: Trajectories do not display using In Place mode.

## In Place Mode Options

In Place Mode is a three-button flyout, with In Place Mode (locking movement on both the X and Y axes) the default selection, with these two further options:

- **In Place X Mode.** Locks center of mass X-axis motion. Use this for game export where the character stays in place but the swinging motion of the hips and upper body along the Y-axis is preserved.
- **In Place Y Mode.** Locks center of mass Y-axis motion. Use this for game export where the character stays in place but the swinging motion of the hips and upper body along the X-axis is preserved.

## See also

*General rollout (see page 147)*

## Biped Display Options

For greater speed in displaying bipeds, or to make your viewports less cluttered while you edit your scene, Biped lets you turn off the display of some biped elements. These display controls are found in the Display rollout in the Motion Panel, rather than in the Display tab of the command panel.



These controls allow you to quickly turn on and off the biped bones, objects and footsteps, as well as footstep numbers and trajectories. There is also a Display Preferences dialog accessed from here that lets you control which bipeds are visible during Biped Playback.

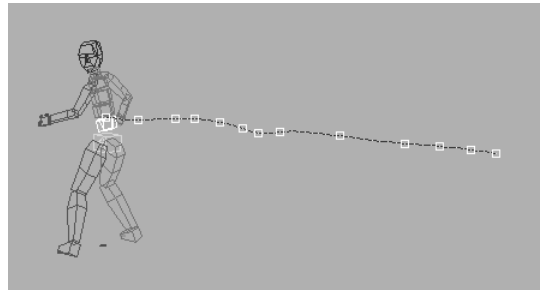
### See also

*Display rollout (see page 176)*

### To change biped display

1. Go to the Motion panel.
2. Select a button to correspond to the display choices you wish.

## Trajectory Display



If you have a motion file loaded, you can not only view the motion using Biped Playback, but you can also see the path, or trajectory, the biped (or selected biped links) follows throughout the motion. You turn on trajectory display by clicking Trajectories in any of these environments:

- Display rollout (Motion panel)
- Key Info rollout (Motion panel)
- Key Info dialog (Track View)

Do not use the 3D Studio MAX Trajectories button to display Biped trajectories.

Trajectories provide useful visual feedback as you edit keys, showing the effects on the motion path for parameters you're adjusting. You also will use trajectory display to compare filtered and unfiltered motion capture data.


Changes to Tension, Continuity, Bias, Dynamics Blend, Ballistic Tension and the overall gravity setting GravAccel are reflected in the trajectories.

Note: Trajectories do not display using In Place mode.

The Display Preferences dialog lets you specify the trajectory information for the selected biped link. You can choose between Bone base and Bone Tip, show entire trajectory or a moving range of frames. See *Display rollout* (see page 176)

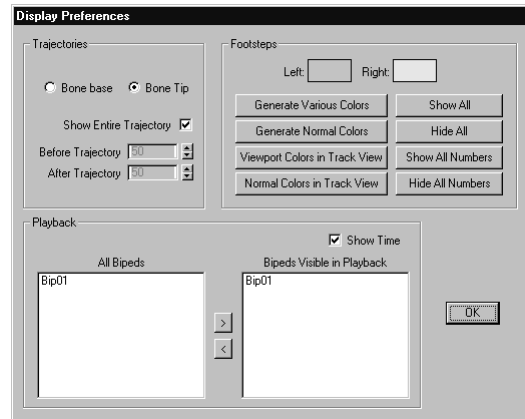
## Procedure

### To edit Biped Trajectory keys in the viewport

1. Select the Body Horizontal or Body Vertical Track for an animated biped.
2.  On the Biped Display rollout, turn on Display Trajectories.
3. Turn on Sub-Object selection level.
4. In the 3d Studio MAX Main toolbar, choose Select and Move.
5. Use the Transform gizmo to move the keys on the trajectory.

## Display Preferences

When you click Display Preferences in the Display rollout on the Modify panel, the *Display Preferences dialog* (see page 177) appears.



Use controls on this dialog to change footstep, trajectory and playback display.

In the Footsteps group you can define colors for the left and right footsteps, and generate various colors or standard colors in the viewport or Track View. You can also Show or Hide All Footsteps or Footstep Numbers.

In the Trajectories group you can choose between the Bone Base or Bone tip for trajectory display. You can show the entire trajectory, or define a range of frames for partial trajectory display.

In the Playback group you can define which bipeds will appear in Biped Playback

See also

*Display rollout* (see page 176)

## Creating Freeform Animations

**character studio** gives you the option to animate character poses with and without the aid of footsteps.

In Freeform mode (without footsteps), you can pose every joint of your character exactly as you like using traditional keyframe methods. You can even blend dynamically between forward kinematics and inverse kinematics to introduce higher-level control.

### Retargeting Freeform Motion

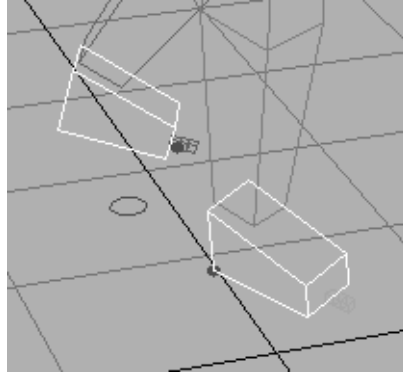
One of the more powerful features of Biped is the ability to retarget or map the motion of any biped onto any other biped. If you map the motion of a biped without a tail onto a biped with a tail, default tail motion for the biped with a tail will be computed. Default motion will also be computed when mapping the motion of a biped with fewer links in the legs, spine, or neck onto a biped with more links in the legs, spine, or neck. There are a few ways of performing motion mapping. You can:

- Go into Figure mode and change the structure of your biped. When you exit Figure mode, the new structure of the biped will adapt to the existing animation.
- Save a *.bip* or *.stp* file from one biped, and load it onto another biped of a totally different structure and size.

### IK Constraints and Pivots for Freeform and Footstep Animations

Freeform and Footstep animation use identical IK constraints and a system of pivots on the hands and feet. In both methods of animating the biped you set IK constraints and select pivots

for the hands and feet using tools in the IK Key Info rollouts. IK constraints allow you to place a hand or foot into world space, biped space, or in the coordinate space of another object in the scene.



**Pivots are used in Freeform and Footstep animations**

### Join to Previous IK Key

You can set an IK parameter that links a hand or foot to the previous IK key. This allows you to rotate a hand or foot in the coordinate space of the previous IK key, which is useful when your keyframing a walking motion for example. There are three set key buttons in the IK Key Info rollout to simplify setting IK constraints.

### See also

*About Footstep Animation (see page 51)*

*Pivots (see page 77)*

*IK Key Info rollout (see page 167)*

---

## About Freeform Animations

While Biped calculates vertical dynamics and gravity based on its footstep-driven technology, you don't always want your character strictly under these controls. You might want the character to fly, swim, or to do something improbable in a physical world.

For these situations, Biped supports a comprehensive set of freeform animation controls that allow you to take total creative control over your character's movement and timing.

You can create a freeform animation for a new biped or you can convert an existing footstep animation into a freeform animation and back again.

### To create a purely freeform animation in Biped

1. Create a biped.
2. Begin creating keys.

### To convert a footstep animation to a freeform animation, or vice versa

1. Select the biped whose footstep animation you want to convert to freeform or vice versa.
2. On the Motion panel General rollout, click Convert.  
A Convert To dialog displays. Convert to Freeform or Convert to Footsteps, depending on the animation method of the currently loaded motion.
3. Change options as necessary on the Convert To dialog, and choose OK.

You also have the option to insert a freeform period in a footstep sequence.

---

## Freeform Walking Animation Using IK Constraints

Walking motions in both Freeform and Footstep animations should follow the rules of certain IK constraints. In Freeform and Footstep animation, a footstep interval is the start and end of a sequence of IK constraints in World Space with IK Blend greater than 0. A biped foot in the Move state should have body space turned on with an IK Blend of 0. By using these IK constraints you can convert between the two animation methods seamlessly.

Rather than having to set these various IK parameters manually, you can use the preset set key controls in the IK Key Info rollout. The preset set key buttons apply the most common IK constraints.

When creating a Freeform walking animation follow these IK constraints by moving the biped feet and then using the appropriate set key buttons.

### See also

*IK Key Info rollout (see page 167)*

---

## Biped Tracks

To animate your character with freeform methods, you need to know how to select the body part you want to animate, as well as the type of movement you want to affect for that part of the body.

3D Studio MAX and Biped provide a number of different methods for selecting and moving these animation tracks. Several involve using 3DS MAX's Track View dialog, a powerful environment for viewing and managing the geometry and motion data in your scene.

Motion data is viewable in the Track View for each biped body part or on the Track Bar. Once you've selected the biped object, using one of the methods described below, you can see its associated motion data, displayed in the Transform branch for that object or the Track Bar.

## See also

*Track View (see page 262)*

### To access Track View

- Click Track View on the 3DS MAX toolbar.
- Right-click over a selected biped object and select Track View Selected in the pop up menu.

### To select from the screen

1. Place the cursor over the biped link whose tracks you want to examine. Use zoom controls as necessary.
2. Pause over a part of the biped, and a tool tip appears, informing you which object is at the cursor position.
3. Click to select the biped link.

**Tip:** Another feedback device is available once you've made your selection. The selected object's name displays in the topmost field in the Modify, Hierarchy, Motion, and Display panels.

4. On the Track View, click Zoom Selected Object (at the bottom left of the dialog).
5. The Track View hierarchy window repositions to show the selected object at the top of the window display. Nested below the object's name are the animation, or transform, tracks for that object, if any exist for the current motion.

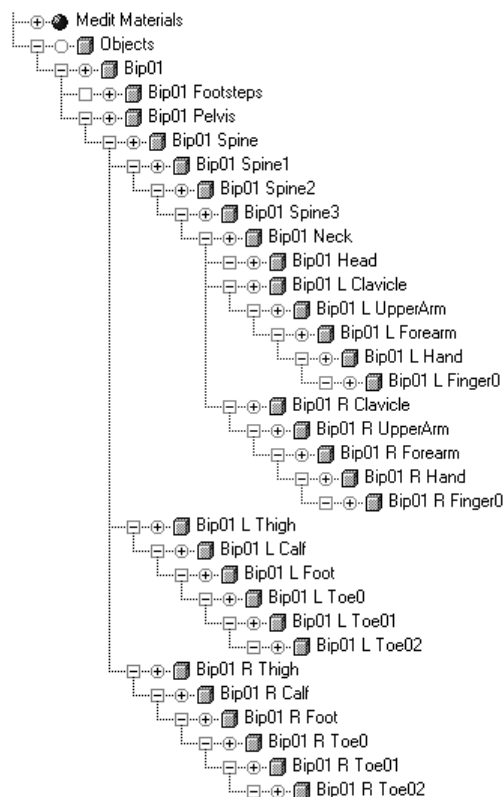
**Note:** By default, all keys for arm objects are in the Clavicle track, and all keys for leg objects are in the Thigh track. If you select a biped foot and click Zoom Selected Object, you may need to scroll up the Track View hierarchy to view the Thigh track transform keys. These keys also store foot transforms.

**Tip:** Use controls in the Separate Tracks area of the *Animation Properties* rollout (see page 181) if you prefer transform tracks for individual objects.

## Selecting Tracks with the Select by Name Tool

You can navigate to and select tracks for a given object using the text-entry field next to the Zoom Selected Object in Track View. Enter the object name in the text field, using wildcards as necessary, and press ENTER. For more information, see the 3DS MAX documentation.

## Selecting Tracks from the Track View



You can navigate to a given object simply by traversing the Biped hierarchy in the Track View hierarchy window.

Note that child objects in the hierarchy are nested below parent objects. You may need to open several parent objects in order to get to the nested object you want. Biped's tracks are grouped as follows:

- Center of Mass
- Footsteps
- Pelvis (branch1) Spine, Neck (branch), Head, Ponytail.
- Pelvis (branch2) Thigh, Calf, Foot, Toes.

- Pelvis (branch3) Tail.
- Neck (branch 1), Head
- Neck (branch 2), Clavicle, UpperArm, Forearm, Hand, Fingers.

Each of these tracks can be seen in Track View.

Note that the hands and fingers are grouped with the arms, and the fingers and toes are grouped with the legs.

The body's translation keys are decomposed into separate vertical and horizontal tracks.

---

## Expanding Biped Animation Tracks

Biped lets you expand and collapse certain animation tracks to give you more control over your character's movement as you set keyframes. For example, tracks for body parts in the arms can be animated using five separate tracks for maximum control.

You can also collapse these tracks for simplicity, and use a single key to pose the entire arm. Tracks can be expanded or collapsed in this way for the arms, legs, ponytails, neck, tail, and spine.

See also

*Animation Properties rollout (see page 181)*

---

## Transforming Links

Biped restricts the axis and reference coordinate system of various transformations. It also restricts when certain transformations can be performed. When the Motion panel is active for a biped figure, Biped sets or disables the appropriate axis constraints and coordinate system to show the current constraints.



For example, you can only scale biped parts while Figure mode is active. The constraint overrides are most elaborate for rotations. See *Rotating Links* (see page 33).

When Biped controls are not visible in the Motion panel, or another panel is active, transform constraints still apply to biped parts, but the appropriate icons in the toolbar will not be disabled.

**Tip:** Always have Biped controls visible in the Motion panel when you transform biped links.

---

## Moving Links

Use the standard 3D Studio MAX Move transform to move biped links.

As shown in the table below, there are three types of movements you can apply to a biped link:

- **General Move.** When you select and move the center of mass object, it and all of its children move; that is, the entire biped moves. When you move the center of mass and one or more legs are planted on the ground with IK constraints (IK Blend=1), the Biped tries to maintain the legs' planted position while the body moves.
- **Inverse Kinematics.** Applies to all parts of the arms, legs, fingers, and toes, except the clavicles. If you attempt to move a hand or foot beyond the biped's kinematic limit, the arm or leg straightens out and moves as far as possible in the direction you dragged.
- **Moving a link relative to the rest of the body.** Since this really alters the structure of the figure, you position a biped part relative to the rest of the body only when you are in Figure mode. The tail, the spine along with the upper body, the clavicles along with the arms, the fingers, ponytails, and the toes

can all be positioned relative to the rest of the body.

The pelvis, head, neck, non-base spine links, and non-base tail links cannot be moved, although they move when their parent moves.

When you select multiple biped parts and move them, the move applies only to selected biped parts that have no selected ancestors. For example, when the entire biped is selected, only its center of mass object is moved. When all of the finger joints are selected, the bases of all the fingers are moved.

**Tip:** You can select the children of a hierarchy by double-clicking the parent.

Biped provides controls to help you give both arms or both legs the same pose. See *Copying and Pasting Postures* (see page 40).

---

## Rotating Links

Use the standard 3D Studio MAX Rotate transform to adjust a biped's posture by rotating its links.

Biped joints limit how the links can rotate, in order to make biped movement appear natural. For example, the pelvis behaves like a hinge. It can rotate in its local Y axis, but can't twist or roll. On the other hand, ball-and-socket joints, such as hips or ankles, can rotate in all three axes, X, Y, and Z. The pelvis is said to have a single degree of freedom, while the ball-and-socket joints have three degrees of freedom.

The table below shows the freedom of each joint. Biped allows a bit greater freedom than most human bodies are capable of. All rotations are performed in the local coordinate system.

**Tip:** First select the link to rotate, then click Rotate. While the Motion panel is active, Biped sets the transform managers to reflect the degrees of freedom of that link. For example, if

you select the pelvis and then click Rotate, Biped changes the coordinate system to Local and turns on the Y axis constraint. (For joints that have more than one degree of freedom, you might later need to change the axis constraint setting.)

Some biped parts have special-case conditions that govern how you can transform them, as described in the sections that follow.

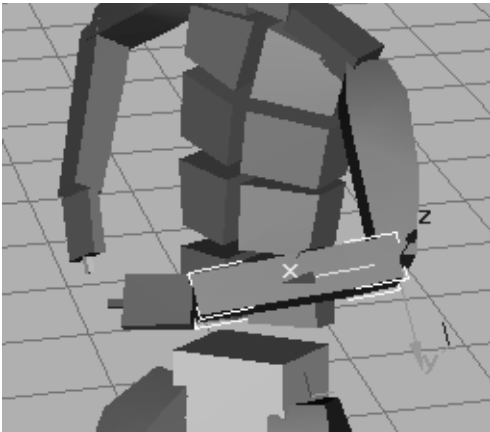
Biped Link	Link Name	Free Axes	Comments
Center of mass	Bip01 (default)	X, Y, Z	Rotates entire biped
Pelvis	Pelvis	Y	If feet are planted, adjusts legs to keep feet and toes above ground
Head	Head	X, Y, Z	
Neck	Neck	X, Y, Z	Neck orientation does not affect head orientation
Spine	Spine, Spine1-4	X, Y, Z	Spine rotation adjusts overall balance
Tail	Tail, Tail1-4	X, Y, Z	
Clavicles	R Arm, L Arm	Y, Z	Shoulder orientation does not affect clavicle orientation
Shoulders (upper arm)	R Upper-Arm, L UpperArm	X, Y, Z	Rotating pivots from shoulder to wrist
Elbows (lower arm)	R Forearm, L Forearm	X, Z	Hinge plus special rotation
Hips (upper leg)	R Thigh, L Thigh	X, Y, Z	Rotating pivots from hip to ankle
Knees (lower leg)	R Calf, L Calf	X, Z	Hinge plus special rotation
Hands	R Hand, L Hand	X, Y, Z	

Biped Link	Link Name	Free Axes	Comments
Feet	R Foot, L Foot	X, Y, Z	If feet are planted, adjusts legs to keep feet and toes above ground
Fingers	Finger0, 01, 02 Finger1, 11, 12, etc.	X, Y, Z	Finger bases have three free axes; other finger joints have Z only
Toes	Toe0, 01, 02 Toe1, 11, 12, etc.	X, Y, Z	Toe bases have three free axes; other toe joints have Z only

## Special Rotation: Elbows and Knees

Elbows and Knees perform a special rotation when they are rotated about their X axis. They don't actually rotate around their X axis; this does not make sense because they have one degree of freedom. Instead, the upper and lower arm/leg are rotated together about an invisible axis defined by the line stretching from the shoulder/hip to the wrist/ankle. This special rotation can be very useful for positioning the arms and legs.

The special rotation can also be useful for creating characters with reverse knee bends. When the knees are rotated backward, at more than a 90-degree rotation from the front-facing human knee posture, Biped assumes the character has backward knees or bird legs, and uses this as a reference position for all *.bip* motions.



Rotating the forearm along the X axis rotates the arm elements about an invisible axis between the shoulder and wrist.

## Balance: Spine

Biped uses only the spine, in conjunction with the center of mass, to maintain the biped's balance. Because of this, rotating all of the spine or any one of its links causes the horizontal position of the body to change relative to its center of mass. These adjustments are performed in the center of mass's local reference coordinate system, ensuring that the figure will rotate naturally about its center of mass; for example, during flips in the air.

On the Motion panel Key Info rollout, you can turn this behavior off by setting Balance Factor to 0.0 for corresponding horizontal center of mass keys.

## Independent Orientation: Arms, Head, Feet

Changing the orientation of a clavicle (the root Arm object) changes the position, but not the orientation, of its corresponding upper arm. In effect, the clavicles are a support from which the arms are suspended. Likewise, the orientation of the neck changes the position, but not the orientation, of the head. Although linked to the neck, the head typically rotates independent of the neck, and interpolation of these independently set orientations produces more natural looking motion. Similar to the head and arms, changing the orientation of the upper or lower leg changes the position, but not the orientation, of the corresponding foot. In this way, the foot orientation remains relative to the ground plane.

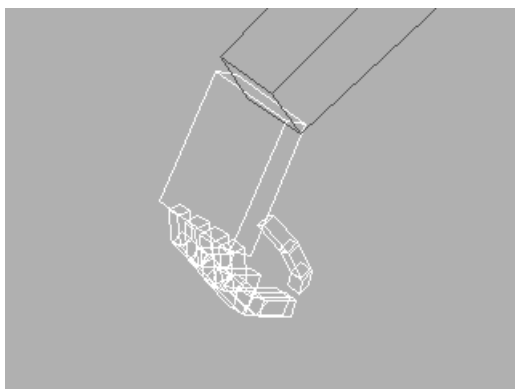
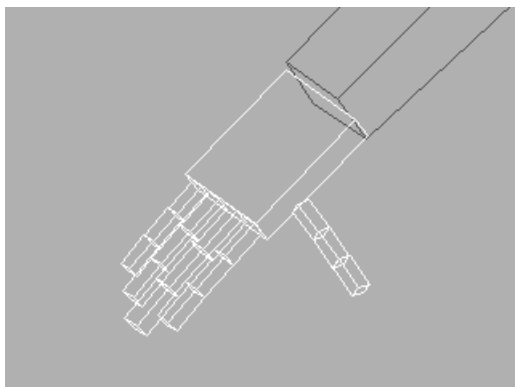
## Ground Plane Collision Detection: Pelvis, Feet

When one or two feet are planted, and the pelvis or a planted foot is rotated, Biped detects collisions of the foot and its toes with the ground plane defined by the planted foot's footstep. As the pelvis or foot is rotated, Biped uses inverse kinematics to alter the rotations of the leg joints so that the foot and its toes do not go below the plane defined by the footstep.

Note: A biped foot on a Sliding Footstep is a special case. The biped foot can be moved to any position and orientation for the duration of the footstep.

Note: Changing rotation values for the legs and toes also maintains collision detection, but the position adjustment is not made until you set a key.

## Rotating Multiple Links



**Selecting and rotating a hand and all its fingers causes the fingers to curl**

You rotate multiple links to produce curling effects such as fingers curling around a glass or a tail curling up and down. You can choose from two methods to rotate multiple links:

- Select and rotate multiple links manually to use the 3DS MAX technique of applying an equal rotation to each selected link.
- Enable Bend Links mode and then select and rotate any spine, neck, or tail link to use the Biped technique of naturally bending the entire spine, neck or tail.

## Selecting and Rotating Multiple Links

When you select and rotate multiple links, the rotation is individually applied to each selected link. This is a convenient way to get fingers to curl, for example, or to keyframe a biped's arms, legs, multiply-jointed neck, or tail.

### To select and rotate multiple links

1. Select and rotate any number of links.
2. Typically you select an object and all of its children. For example, selecting the hand and all of its fingers.

## Using Bend Links Mode

You can use Bend Links mode to rotate multiple links for the biped's spine, neck, or tail. Bend Links transfers the rotation of one link to the other links in a natural way. When applied to the spine, it is particularly useful for positioning the biped's hips.

### To rotate all links in the spine, neck, or tail

1. On the General rollout, click to turn on Bend Links Mode.
2. Select and rotate a single spine, neck, or tail link. The other links in the spine, neck, or tail rotate to match the single link's rotation.

Use Bend Links mode either to pose the biped while in Figure mode, or to animate the spine, neck, or tail while in Keyframe mode.

## Setting Biped Keys

You animate biped's body parts as you do other 3D Studio MAX objects: by setting keys for postures at keyframes.

Biped provides two different ways to set keys at the current frame:

- Pose the biped body part and then click one of the set key buttons in the IK Key Info rollout. Or you can click on Set Key in the Key Info and Keyframing rollouts of the Motion panel.
- Turn on the Animate button and then pose the biped part.



If you are animating a walk cycle or intricate hand animation, then you should make use of the three different types of set key buttons in the IK Key Info rollout. Each set key button applies different IK constraints depending on whether a foot or finger is in a planted state, a move state, or a sliding state. See *IK Key Info rollout* (see page 167) for more information on the three set key buttons.

If a Bones system using the IK controller or a particle emitter is linked to the biped, the Animate button must be on in order to position the biped's limbs. This is also true for biped objects if you select 3DS MAX Trajectories on the Motion panel, or if you choose Views > Show Ghosting.



The Set Key option in Biped has the advantage that you can easily experiment with different poses for your character without unintentionally setting keys as a side-effect. The animate mode approach is especially useful when you make adjustments to keyframes that have already been set.

**Tip:** Use Set Key to insert a key on a frame where a key doesn't exist. You'll often want to refine the motion using controls in the Key Info rollout, without selecting and moving an object in the viewport.

Keyboard shortcut: Pressing 0 (zero) is equivalent to clicking Set Key. Make sure that the Plug-in Keyboard Shortcut Toggle is active.



You can move back and forth between Biped's keys by clicking the Previous Key and Next Key buttons on the Key Info rollout. As with other 3DS MAX objects, you can also move back and forth between keys by turning on the 3DS MAX Key Mode Toggle, selecting an element associated with a given track, and then using the 3DS MAX buttons Next Frame and Previous Frame. For example, you can view keys, move between them, and set keys for a right arm track if any of the right arm's objects (clavicle, upper arm, lower arm, hand, fingers) are selected.



You can delete a key by clicking Delete Key. The biped part that has the key must be selected, and the current frame must correspond with the keyframe you want to delete. Keys that are locked (appearing in red in Track View) cannot be deleted.

**Tip:** When you've selected the biped part you want to transform, click Lock Selection Set on the 3DS MAX prompt line. Now you can transform the part without accidentally selecting a different part of the biped. The default keyboard shortcut for Lock Selection Set is the SPACEBAR.

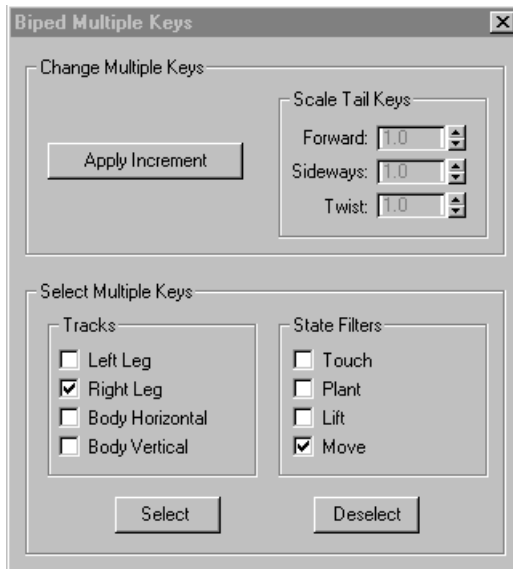
## Adjusting Multiple Keys

Biped offers special operations on the Motion panel that can quickly change the values of several keys. You access these features by clicking Set Multiple Keys on the Keyframing rollout, to display the Biped Multiple Keys dialog. All operations in this dialog apply to the currently selected biped.

### See also

*Keyframing rollout (see page 173)*

## Apply Increment



Apply Increment is a powerful tool. It allows you to increment a set of keys in a biped track. For instance, if you have an animation of a running biped and you want the biped's legs to kick higher, you can select all of the biped's moving leg keys and increment them slightly so that all the kicks are higher.

The apply increment operation depends on two things:

- The translation or rotation most recently performed on the biped
- Which keys are selected in Track View from the tracks on which the most recent translation or rotation was performed

When you click Apply Increment, the same transformation (only move, rotate, but not scale) that was most recently performed on the biped is now applied to the biped's selected keys in Track View. It is applied only to the selected keys that lie in the tracks on which the most recent translation or rotation was performed. An increment made to a leg or arm can also be applied to the opposite leg or arm.

Although this might be difficult to understand at first, it is actually quite easy to use. Typically, you apply an increment to only one track. For example, follow these steps, making sure that you're not in sub-object mode:

1. Start with a simple biped animation, either by loading in a .max file or by creating multiple footsteps and activating them.
2. At any frame in the animation, select one or more spine links, and bend the biped over.
3. On the Track Bar select all of the keys in the spine track.
4. Click Set Multiple Keys in the Keyframing rollout and the dialog appears.
5. In the dialog click Apply Increment.
6. Play back the animation. The biped is now bending down during the entire animation.

**Tip:** Make sure the Animate button is turned off and that you don't set a key while performing the bend. If you have that new key selected when using Apply Increment, the bend value is applied to the new value, and the bend value is doubled on that

keyframe. You also do not want to do this on an established keyframe.

## Select Multiple Keys

The controls in the lower area of the Biped Multiple Keys dialog provide a way to select and deselect certain keys easily in Track View. They are very helpful when you want to apply an increment to all keys of a particular type (touch, plant, lift, move) in a particular track.

At least one Track and one State Filter must be selected for the Select and Deselect buttons to be enabled. When you click Select/Deselect, all keys in the checked tracks that are in one of the checked filter states will be selected/deselected.

For example, if you want to increment all the moving leg keys, follow the steps in the example above, selecting a leg instead of the spine and raising it the way you want it to be incremented. Then, instead of step 4, select Left Leg and Right Leg from the Tracks section of the dialog, and select Move from the State Filters section. The Select button becomes enabled. Click it to select all of the moving leg keys. Continue with steps 6 and 7.

## Scale Tail Keys

The controls in the upper right area of the Biped Multiple Keys dialog provide a way to scale the tail animation. Each spinner controls a different part of the tail animation, the tails forward, sideways, and twisting motion. These spinners act only upon the selected tail keys.

Note: The spinners are active once you've selected Destination keys in Track View.

Immediately after you use the spinner to set a new value, Biped resets the spinner to its default value of 1.0. To put it another way, scaling tail keys is always based on the tail keys' current

scale; the scale parameter is not an absolute value.

If you spin up, the tail's current animation in the chosen direction is exaggerated, and the tail whips around in extreme positions. If you spin down, the tail's current animation is toned down and the tail does not move as much. If you spin to 0.0, the animation in the chosen direction disappears. If you spin negative, the animation is reversed; instead of going from right to left or up to down, it goes from left to right or down to up.

---

## Using In Place Mode to Adjust Keyframes

In Place Mode is a good way to adjust keys on a biped that already has animation applied to it. Rather than scrolling the view at different frames to keep a running biped visible, turn on In Place Mode. Now when you scrub the time slider, or use the Next Key and Previous Key buttons, the biped remains visible. A key that needs adjustment can be quickly spotted and corrected.

Note: Judging lateral center of mass motion using In Place mode is difficult. In Place mode limits center of mass motion on the XY axes; all sense of body momentum on these axes is suspended during playback. You may want to exit In Place Playback to gain a sense of lateral momentum when setting or adjusting center of mass horizontal keys (Body Horizontal track).


Note: In Place Mode is a three-button flyout.

### See also

*General rollout (see page 147)*

### To use In Place Mode to adjust keyframes


1. Select a biped that has animation.

2.  On the Motion panel, on the General rollout, click In Place Mode to turn it on.
3. Turn on the Animate button.
4. Find a frame where the biped needs adjustment and modify or add keyframes.

## Copying and Pasting Postures

Biped's *Keyframing* rollout (see page 173) in the Motion panel provides controls to help you copy and paste biped postures and parts of biped postures as well as copying and pasting an entire track.

### To copy a posture

1. Select the set of biped parts that defines the part of the biped's posture you want to copy.
2.  Click Copy Posture in the Keyframing rollout.
3. The posture of the set of selected objects in the biped is copied to an internal buffer.

### To paste the posture

Note: Turning on the Animate button before pasting posture automatically sets a key for all objects whose posture has changed.

-  Click Paste Posture in the Keyframing rollout.

The copied posture is pasted onto the selected biped. It doesn't matter which part of the biped is selected, all biped parts whose posture was saved by the most recent Copy are pasted.

The Paste command is useful for copying a posture in one frame of an animation to another frame of the animation. Copy the

posture, move to another frame, and then paste the posture.

**Warning:** If you are in Keyframe mode and you paste a posture while the Animate button is off, no key is set. If you want the pasted posture to set a key, you must set one for all selected objects using the Set Key button.

Paste is also useful for copying a pose from one biped to another. Copy the posture, select the other biped, and then paste the posture.

### To paste a limb posture on the opposite limb

1. Select all of the limb, arm or leg, whose posture you want to copy.

2.  Click Copy Posture.

3.  Click Paste Posture Opposite.

The opposite limb now has the posture of the limb you copied.

The Paste Posture Opposite command is particularly useful for copying the posture of an arm or leg or a part of an arm or leg onto the opposite arm or leg of the same biped.

Both Paste Posture and Paste Posture Opposite work differently in and out of Figure mode. Out of Figure mode, only the orientation of the copied links is pasted. In Figure mode, both the orientation and the scale of the copied links are pasted. Also, when you paste the finger base, toe base, spine base, tail base, or clavicles in Figure mode, the position of that link relative to the biped's body is pasted.

Keyboard shortcuts: The following are Biped keyboard shortcuts to the copy and paste



posture commands. Make sure that the Plug-in Keyboard Shortcut Toggle is active.



Shortcut	Meaning
ALT + C	Copy Posture
ALT + V	Paste Posture
ALT + B	Paste Posture Opposite

## Copying the Entire Biped Posture

In the Keyframing rollout in **character studio** is a flyout containing the Copy Posture and Copy Pose buttons.

Use Copy Pose to copy the entire biped pose. Use Copy Posture to copy the posture of one or multiple selected biped limb(s). You can then paste this pose or posture at a different frame or onto a different biped.

### To copy the entire pose of a biped:

1. Select any part of the biped whose pose you want to copy.
2.  Click Copy Pose, the bottom control on the Copy flyout.
3. Advance to a new frame, or select another biped.
4. Turn on the Animate button.
5.  Click Paste Posture.

## Copy Tracks

Biped's Keyframing rollout in the Motion panel provides controls to help you copy and paste biped tracks. Copy Tracks is one of the options in the three button fly-out (Copy Posture, Copy Pose, Copy Track) on the Keyframing rollout. Use Copy Tracks if you want to copy and paste an entire animated track from one part of the

biped to another part of the biped or onto a different biped.

See the *Keyframing rollout* (see page 173) topic for procedures and reference information on Copy Tracks.

## Globally Position the Biped with Freeform Animation

A biped with footstep animation can be repositioned by moving the footsteps. To reposition a biped animated in freeform follow these steps:

### For a biped whose limbs are attached to another "Object Space" object

1. Use Link on the 3DS MAX toolbar to link the Object Space object and the biped's COM (Center of Mass "Bip01") to a dummy object at the same frame.
2. Animate the dummy object to globally reposition the biped.

### For a biped whose limbs are in world space

World space has IK set to 1.0 in Object Space with no Object Space object specified.

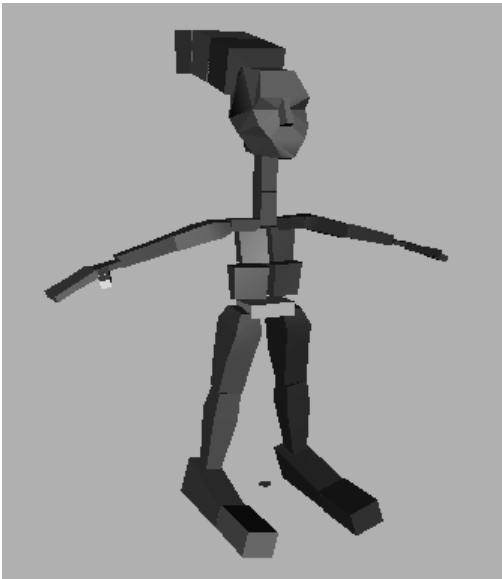
1. Use Link on the 3DS MAX toolbar to link the limbs and the biped's COM (Center of Mass) to a dummy object.
2. Reset IK Blend and Object Space for each IK key.

### To reposition a biped with freeform animation using layers

1. On the Layers rollout click Create Layer.
2. Turn on the Animate button.
3. Reposition the biped center of mass.

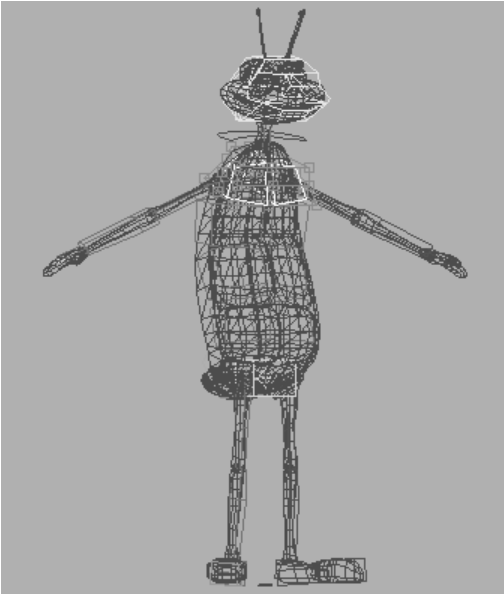
## Customizing Biped Characters in Figure Mode

After creating the default biped, you may need to change the proportions of the skeleton. You use Figure mode to allow changes to occur to the biped structure. With the biped selected, choosing Figure mode returns the biped to its original creation location and orientation. With the biped in Figure mode, you can use the 3DS MAX transform tools to change the proportions and positions of body parts. For instance, you might apply a non-uniform scale to shorten the legs for a cartoon character.



**Biped proportioned to fit inside of Dr. X character geometry**

You might rotate the spine objects to create the figure for a hunchback or a dinosaur. Use the move tool to change the position of the thumb or the arms. You can even apply modifiers to the biped skeleton pieces, such as using an FFD on the biped head to adjust its shape.



**FFD Modifier used to shape spine and head**

Once you have a default biped, you'll need to match its proportions to the character's geometry. It is quite typical to find the character's geometry with the arms outstretched. The common workflow is to freeze the character and then in Figure mode, reposition the biped, so the center of mass is at the base of the torso. The spinal objects, legs, and feet are scaled and rotated to fit within the confines of the mesh, then the arms and hands, neck and head. The tail and ponytail objects can be used for animating wings, fins, jaws, ears, horns, or hair.

Once your biped proportions are correct, you can save them in a .FIG file. Since biped saves the character in the .FIG file, and then animation in the .BIP file, you can change the character's proportions without affecting the animation.

### See also

*Changing the Initial Biped Structure (see page 43)*

*Posing a Biped (see page 45)*

*Saving and Loading Biped Figure Files (see page 46)*

*Scaling Links (see page 46)*

*Rubber Banding Arms and Legs (see page 47)*

*Rubber Banding the Center of Mass (see page 48)*

*Linking Objects to the Biped (see page 50)*

*Figure Mode (see page 156)*

*General rollout (see page 147)*

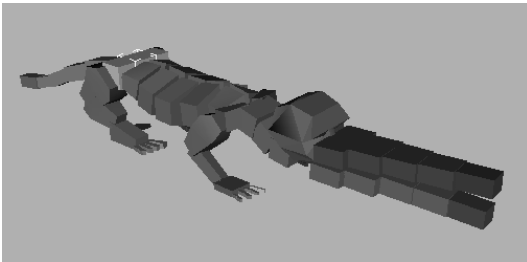
---

## Changing the Biped Structure

Bipeds don't have to appear human. You can change their elements and form to create other kinds of characters. Although you can change some initial aspects of the biped's structure in the Create panel, you use Figure mode to change all aspects of the biped's structure after its creation.



Bipeds don't have to be human



Or even walk on two legs

Any number of bipeds can be in Figure mode at the same time, though you can work only on one biped at a time. When you select a biped, the Figure mode button shows whether or not the biped is in Figure mode.

In Figure mode you can:

- Specify the number of links in each part of the biped.

- Define the position of the base of the fingers, toes, clavicles, spine, tail, and ponytails relative to the body.
- Define the position of the feet relative to the ankles.
- Define the default pose of the biped before animation is applied to it, for example define a hunchback.
- Scale the biped and its various parts.
- Simultaneously scale and position biped parts using *Rubber Band mode* (see page 731). See *Rubber-Banding Arms and Legs* (see page 47).
- Create natural links for Physique using Triangle Pelvis.

For more information on adjusting the biped's proportions see *Posing a Biped* (see page 45).

## Procedure

### To work in Figure mode

1. Select the biped you want to pose, and then go to the Motion panel.  
The Motion panel doesn't show Biped controls unless the biped is selected.
2. On the General rollout, click Figure mode.



The button turns blue to indicate a special edit mode.

The biped's pose and location change to the one last specified when the biped was in Figure mode. If the biped was just created and, therefore, never was in Figure mode before, then it changes to the pose and location it had when you created it. When you turn off Figure mode, the biped returns to its animated pose and location in the scene.

Note: Biped disables all keyframe editing tools when Figure mode is active. Adjustments made in Figure mode are not animatable.

## Posing a Biped

There are two main reasons for defining a biped's pose in Figure mode:

- Fitting the biped to a mesh in preparation for using Physique to attach the mesh to the biped. Leave Figure mode on during the attach procedure in Physique.
- To correct posture in a motion file.

A particular motion file may position a biped body part inappropriately; for example, the collar bones may be rotated down too far, affecting your mesh deformation. Simply rotating the biped collar bones up and exiting Figure mode corrects the collar bone position for the entire animation. The motion references the Figure mode position, if the biped is adjusted, and this adjustment is reflected in your animation when you exit Figure mode.

Note: Save a .FIG file for the biped pose you used when you applied Physique. You could use the .FIG file to reload this position, if you need to reapply the Physique modifier or reinitialize the Physique settings.

## How Biped Uses Figure Mode

When you animate the biped, the Biped plug-in maintains the at-rest pose you have created for these elements of the biped body:

- Spine
- Neck
- Clavicles
- Tail
- Ponytails
- Center of mass position, relative to the body

When Biped adapts the keyframed motions stored in .BIP files to different characters, the keyframes of the above elements are recreated as an offset from the at-rest posture associated with each character's figure. The at-rest posture associated with the arms and legs is always assumed to be a standing posture, with straight legs.

## Restructure Biped to Match File

When you load a .BIP file, there is an option to restructure the biped to match the file. If you turn this on, when you load the file, the biped's structure will change to match the figure of the biped in the .BIP file.

## Talent Figure Mode and Adjust Talent Pose

*Talent Figure Mode* (see page 737) and *Adjust Talent Pose* (see page 721) on the Motion Capture rollout are used in functions similar to that of Figure mode. They are used to size and position biped body parts to better fit raw motion capture data. After importing motion capture data, you may discover that certain biped limbs or the biped scale need a global adjustment in order to provide a closer match to the figure of the talent who performed the motion.

## See also

*Motion Capture rollout* (see page 186)

## Saving and Loading Biped Figure Files

You can save and load biped figure files. Figure files have a .FIG file name extension. They save all information about a biped's anatomy: links, link positions and Figure mode posture, and the scale of geometric elements. This information is also stored in .max files. Figure files do not save animation data, you use .BIP motion files for that.



If you are working with Physique, you should save one figure file as your reference pose. Whenever you want to work on fitting the biped into the mesh, you must load this figure file, if it isn't already loaded or saved in the .MAX file.

See also

*Figure Mode (see page 156)*



### Procedures

#### To save a biped figure

1. Select the biped to save.
2.  Activate Figure mode in the Motion panel.
3.  On the Motion panel > General rollout, click Save File.
4. In the file dialog, enter a name for the figure file, and then click OK.  
**Tip:** While you work on creating a variant biped or a biped pose, save your work frequently in a figure file.

#### To load a biped figure

1. Select the biped to replace with a saved figure.

2.  Activate Figure mode in the Motion panel.
3.  On the Motion panel, on the General rollout, click Load File.
4. In the file dialog, choose the figure file to load, and then click OK.

Note: **character studio** installs several sample .FIG files to the following directory in your 3D Studio MAX path:  
... \cstudio\characters\figures.

**Warning:** Loading a figure file replaces the selected biped's pose and base parameters. If you have created a new pose or a new biped structure, save it in a figure file before you load a different biped figure.

## Scaling Links

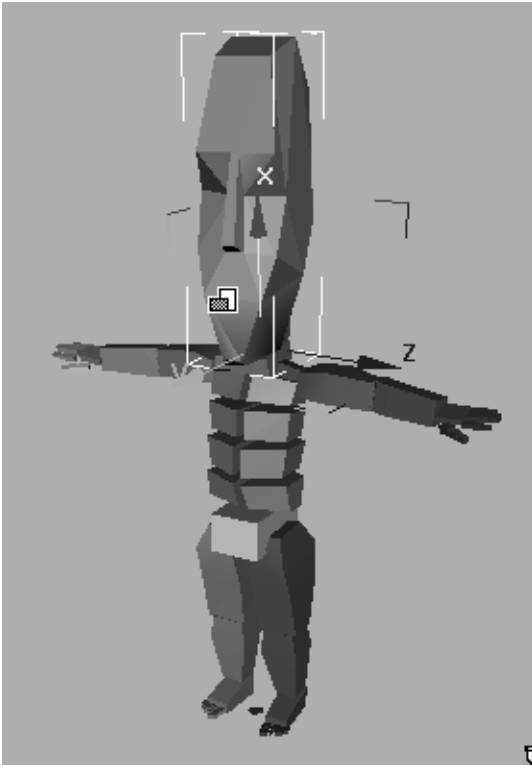
You use the standard 3D Studio MAX Scale transforms to adjust a biped's posture by scaling the size of its links. You must be in Figure mode to scale the biped links. If you try to scale a biped without going into Figure mode, nothing happens.

As with Rotate, when you Scale biped links, Biped constrains the transform to use the link's Local coordinate system. The position of other biped links can change so they remain attached to the resized link. If you shorten the thigh, the calf and ankle will maintain their size, but change their position.

To scale a link, select the non-uniform scale icon from the Scale flyout on the Main Toolbar. When you select a body part to scale, use the Transform gizmo to scale along one axis at a time.


**Tip:** When you use Non-Uniform Scale to scale an object in 3D Studio MAX, a warning appears

that the scale information is not contained in the Modifier stack. Turn on the checkbox in the warning dialog to disable it.




#### Use NU Scale and transform gizmo to scale links

If your character is symmetrical, select body parts in pairs and scale them at the same time. Select one body part and then choose

Symmetrical  in the Track Selection rollout. Now both are selected.

Or you can scale one and then use copy posture



, and paste posture opposite  from the Keyframing rollout. This is handy to insure symmetry in your character.

**Tip:** Use the PAGE UP and PAGE DOWN keys to move through your hierarchy as you work. Scale the thighs, then press PAGE DOWN to select the calves.

Scaling the biped limbs to fit snugly to the mesh will help when Physique is used to attach the mesh to the biped. The Bounding Box option uses biped limb dimensions to size the envelopes. This saves time when you adjust envelopes in Physique.

#### See also

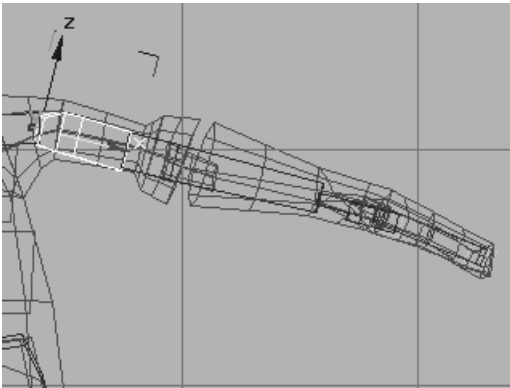
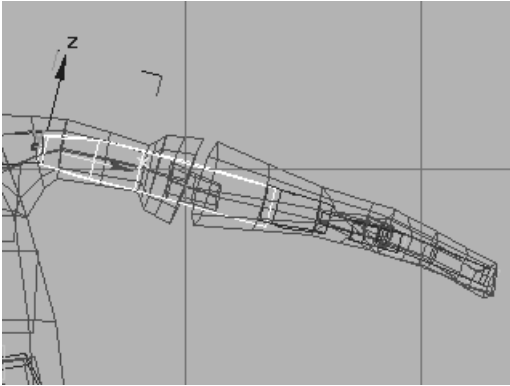
*Rubber Banding Arms and Legs (see page 47)*

*Rubber Banding the Center of Mass (see page 48)*

## Rubber-Banding Arms and Legs

Rubber Band mode provides a way to proportion the arm and leg links simultaneously, by moving the link with the Move transform, instead of using scale. Rubber Band mode scales both the link and its child in a single step.



This is particularly useful when fitting a biped to a skin prior to applying the Physique modifier. For example, rubber-banding the upper arm rescales the upper and lower arm objects and moves the elbow link without affecting the position of the shoulder or the wrist. If you've spent a lot of time getting the fingers in the right place, you can reposition the elbow by rubber-banding, without affecting the hands.



Using Rubber Band to resize an arm without changing the hand position

## Procedure

### To rubber band an arm or leg link

1.  Turn on Figure mode.
2. Select the arm or leg link you want to rubber band.
3. Turn on the Move transform.
4.  On the Motion panel > General rollout, turn on Rubber Band mode.  
This button is grayed out if you are not in Figure mode, or if a part of the biped other

than the upper or lower arm or leg, or the center of mass is selected.

5. Move the selected arm or leg link.

As you move the link, it stretches the link and its child.

---

## Rubber-Banding the Center of Mass

When Rubber Band mode is active, you can move the biped's center of mass in relation to the rest of the body, changing the biped's overall balance. Rubber banding the center of mass defines your character's balance point in at-rest pose and in any resulting motion.

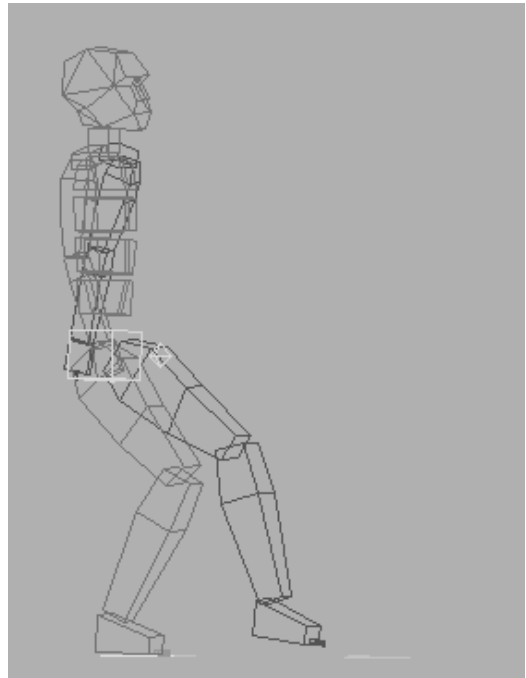
When a character is standing at rest, with feet side by side, the center of mass should be directly above the area where the feet touch the ground. The center of mass determines the center of your character's distribution of weight. For most characters, this is typically located near the pelvis.

Typical placements for the center of mass are:

- Characters that stand erect, such as humanoids, use the default location of the center of mass within the pelvis.
- Characters that naturally lean forward, such as some dinosaurs and birds, should have the center of mass moved slightly forward of the pelvis. This is also good for robots and droid soldiers.
- Characters that are holding a heavy weight out in front of them may need their center of mass moved slightly forward of the pelvis.
- Characters that are pushing or pulling objects may need their center of mass moved slightly behind the pelvis.



Note: If you place the center of mass too far in front of the pelvis, the character unnaturally compensates for balance. As each step is taken, the legs and body move in an awkward fashion, as if there was an invisible weight attached to the front of the character.



**Effects of Rubberbanding the Center Mass behind and before the biped**

For a tutorial that uses this technique, see *Creating the Illusion of Weight* (see page 519).

## Balance Factor

Use Rubber Band mode to position the biped's center of mass in relation to the rest of the body. Then set the Balance Factor in the Key Info rollout to fine tune the biped's balance.

Balance Factor can be animated. If a character is sitting down and stands up, Balance Factor could change from 0 to the default of 1 to account for the shift in weight.

- At a value of 0, the spine bends forward, pivoting at the center of mass. The character's weight is supported by the chair.
- At a value of 1 the pelvis shifts away from the center of mass as the spine bends forward to maintain balance. The character, no longer supported by the chair, must adjust body position to maintain balance.

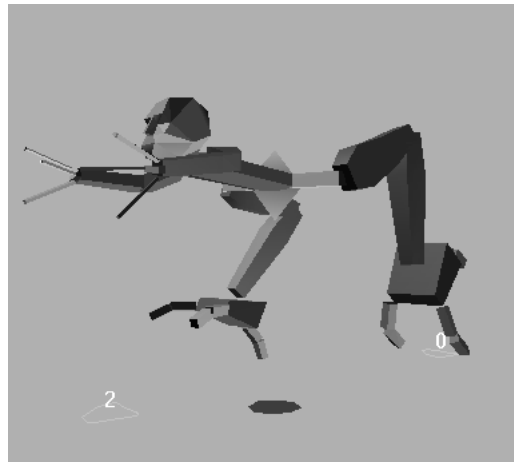
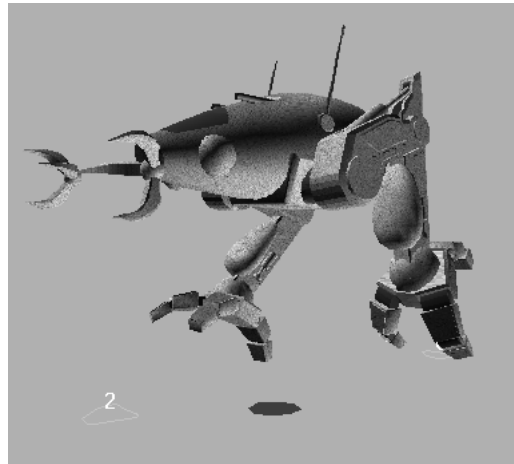
### See also

*Key Info rollout (see page 161)*

*General rollout (see page 147)*

## Linking Objects to the Biped

With the Physique plug-in, you can use Biped to animate a deformable skin, usually a mesh object. However, some animations don't require deformation. For example, a knight clad entirely in rigid metal armor doesn't need to deform as skin does. Also, figures seen from a distance don't require the same degree of realism as figures seen close by.



**Jointed character linked with biped skeleton**

Characters available commercially often come in two varieties: jointless and jointed. Jointless characters have a seamless mesh at limb joints. Jointless characters should be attached to the biped using Physique. A jointed character has separate objects with ball joint geometry for the limbs, and lends itself to the linking technique described in this topic.

Linking objects and other geometry to the biped can also be used in cases such as the following:

- Link a weapon or a flower to the biped hand.
- Link eyeballs or teeth to the biped head.
- Link extra 3DS MAX bones or splines to the biped to create extra envelopes when Physique is applied.
- Link 3DS MAX bones to the biped to automate mechanical assemblies when the biped is keyframed.
- Link particle emitters to the biped hands or feet to create smoke or dust.

Note: If you've linked particle emitters or 3DS MAX bones (with the IK controller) to the biped, the Animate button must be on when you reposition the biped.

### See also

*General rollout (see page 147)*

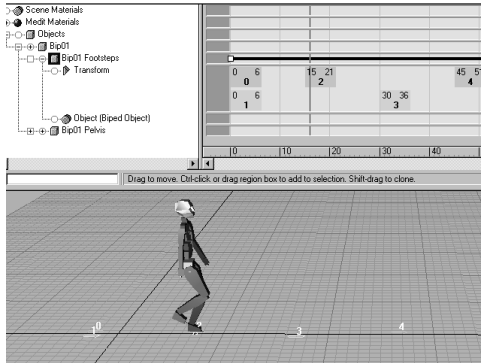
## Creating Footstep Animations

### About Footstep Animation

To animate a biped, follow these overall steps. However, the design process in Biped is not meant to be a linear progression. In practice you will frequently alternate between editing footsteps and keyframes

- Choose a gait pattern for determining the initial timing of a newly added footstep.  
Biped can create walking, running, or jumping footstep patterns. These gaits are foundation for a more complex biped animation, Character Studio provides you with the basic movement. The fine touches are left to you.
- Create footsteps.  
For extended movement that can be described by periodic gaits, such as walking through the scene, you can quickly create multiple footsteps. For more detailed footstep placement that is non-periodic, such as a dance, you can create footsteps individually. After the footsteps are initially created, you can adjust or change both their spatial location and their timing (in Track View). At this stage they are displayed in dark colors to indicate that no keyframes have been generated for them. They are called *inactive*.
- Activate the footsteps.  
When you activate the footsteps, Biped creates default keyframes for each of the tracks of the figure: the head, spine, pelvis, arms, legs, and, if appropriate, tail and ponytails. These keys form an initial sketch of your animation. The default keys, when

*interpolated* (see page 727), form the basic, minimal motion required to animate the figure according to the footstep pattern. Once the keys have been generated for them, they are active and their footstep color changes to a lighter shade of green (right) and blue (left).



Note: By default, *Biped Dynamics* (see page 723) is selected in the Animation Properties rollout; gravity (Dynamics Blend) and ballistic tension calculate the trajectory of the center of mass for all newly created keys in a footstep animation containing a running or jumping motion. If *Spline Dynamics* (see page 732) is selected in the Animation Properties rollout before footsteps are created and activated, then the center of mass uses Spline Dynamics. Using Spline Dynamics, you must set keys for the top of a jumping motion or the dip when the character lands; this trajectory is automatically calculated with Biped Dynamics.

- Edit the default keys to refine the movement.

Using the default keyframes as a starting point, you can interactively insert, replace, or delete keyframes in order to refine the motion of the biped and fill in the details of

movement that are unique to your animation.

- Edit the footstep pattern

At any point in the design process, you can choose to interactively edit your footstep's spatial pattern in the scene or the timing of footsteps in Track View. The keyframes adapt to each edit: changes to footstep location retain the details of all your keyframe positions. Keyframe timing remains synchronized with changes to footstep timing, except in cases where default leg keys must be regenerated to account for timing edits that alter the basic gait pattern, such as creating a hop in the middle of a walk.

Use Footstep mode to create and edit footsteps. Use Keyframe mode to create and edit your character's keys. You can always edit the timing of both footsteps and keyframes in Track View.

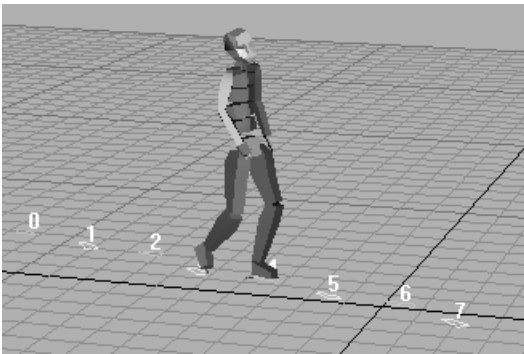
While the biped's feet are airborne, you can animate its legs as you do its upper body. Biped does not create keys based on physics while the biped is off the ground, so animating the legs might be necessary to make long leaps realistic. Alternatively, you might want to make the biped appear to be floating in midair, or underwater, or have it ride a bicycle. See *Freeform Editing Between Footsteps* (see page 68).

You can make the biped interact with other objects in the scene: throwing or kicking a ball, opening a door, and so on. You do this by attaching a biped limb to an object in the scene.

An animatable IK Blend parameter lets you store the anchored position and combine inverse with forward kinematics. Remove the anchors once keys have been set.

**Tip:** You can often get good results by loading an existing biped motion and then varying it. **character studio** installs a set of sample motion files in the `\cstudio\motions` directory.

Biped's patented footstep-driven keyframe animation feature allows animators to use footsteps to create broad, global brush strokes for character movement. Once footsteps are in place, key frames are generated automatically to produce an initial sketch of the 3D character's motion. Throughout edits and revisions, the original nuances of the character are preserved; Biped remembers everything about how a character moves, and it makes all of the appropriate adjustments if the footsteps are changed.



IK constraints are applied the same way in a Footstep and Freeform animation.

## IK Constraints and Footstep Animation

Rather than creating footsteps you can load a motion file that has footsteps in it. You can load a motion capture file and extract footsteps from a file that contains no footsteps. Once a Footstep animation is loaded you can edit footsteps using the same editing techniques as editing Freeform animation. You can apply IK constraints to the hands and feet and use pivots to rotate a hand or

foot. Footstep duration can be changed by applying IK constraints. You don't need to open track view to change footstep duration. IK constraints are applied using tools in the *IK Key Info rollout* (see page 167).

## Composing Footsteps

Footsteps are a central compositional tool in Biped. Footsteps are like *gizmos* (in 3DS MAX terms) for creating and manipulating keyframes. The language of footsteps allows you to more directly describe and initially compose the complex temporal and spatial relationships that are found in different forms of dance, acting, and gait-based locomotion.

Footsteps are biped sub-objects. In viewports, they look like the diagrams often used to explain ballroom dancing. In Track View, each footstep appears as a block that represents a support period for the corresponding foot and leg, left or right, of the biped. The footstep's position and orientation in the scene controls where the biped steps.

## Workflow

In general, compose your footstep pattern with these actions:

- Create the initial footstep pattern in time and space.  
You can place footsteps individually, one at a time, or by invoking Biped's Multiple Footstep creation tools. When you first create them, footsteps are inactive. They exist in the scene but don't yet control biped motion.
- Edit the Inactive Footstep Pattern.  
Footsteps can be edited in both time and space, at any time, either before or after you have activated them. You can move, rotate,

or scale footsteps to change their position in the scene; you can also bend the footstep path. You are free to edit their timing, duration and order in Track View.

- Activate the footsteps to create default keys as a motion sketch.

Activating footsteps causes the biped to move according to footstep placement and the other parameters you have set.

- Preview biped motion.

The default sketch allows you to evaluate the broad strokes of your character's timing and dynamics.

- Edit the keyframes to refine the motion or rearrange the basic footstep pattern.

You can opt to either incrementally edit keyframes and footsteps, or simply deactivate or delete footsteps in order to start from scratch.

In viewports and in Track View, left footsteps are blue and right footsteps are green. Inactive footsteps are dark, or saturated, blue or green, and active ones are a paler shade. You can change these color assignments using controls in the Display Preferences dialog, available on the Motion panel Display rollout. See the Online Reference for details.

Footstep creation controls appear in the Motion panel when you turn on Footstep mode.



**Footstep controls in Motion panel**

## Planning for Footsteps



If you are creating custom steps such as a dance, it is helpful to do some careful planning when laying out your footstep pattern. Things to keep in mind are:

- What is the sequence of the footsteps? In other words, what foot are you stepping with next? Press Q (on the keyboard) to toggle which footstep gets created next.
- Is there a spatial pattern to the footsteps in the scene? Arrange the footsteps in the viewports using Move on the 3DS MAX toolbar.
- How many footsteps are there going to be? It is often useful to find a rhythm in the footstep sequence. Does the footstep pace follow some kind of a beat? If it does, synchronize the footstep spacing in Track View to this beat. You may find it useful to use the Sound track in Track View to synchronize the footsteps to the beat.
- How long is the biped going to be standing on each foot as it goes through the motion? Adjust the lengths of the footsteps in Track View to change the time the feet are in contact with the footstep.
- What is the support relationship between the left and right footstep? At what frames, and how long, are the single support, double support, and the airborne periods? Does one foot cross in front of another foot? This is a very important step because changing the support period may change the *gait pattern* (see page 726), which would force Biped to resynchronize the lower body keys, and possibly introduce or remove dynamics periods. This type of a change may remove or reset user-defined keys surrounding the footstep boundaries and should be avoided during later stages of the animation process. On the other hand, changes to the footstep timing without introducing changes to *gait type* (see page 726) are much more controllable and can occur at any time during the animation process.

These steps can occur before or after activating the footsteps, but should certainly be considered before refining the motion. Since motion is adapted from its current position to new position, you may find that the default motion may differ as multiple adaptations occur. If you have made multiple timing changes after activating footsteps you may want to deactivate and reactivate the edited footstep.

### Steps to creating a quick dance motion

1. Turn on Footstep mode.
2. On the Footstep Creating rollout, turn on Run.
3. Turn on Create Footsteps (at current frame).
4. In the Top viewport click multiple times to create footsteps.  
Place the footsteps in a dance pattern.
5. On the Footstep Operations rollout click Create Keys for Inactive Footsteps.
6. Click play to view the animation.

## Footstep Timing: Gait Parameters

When you create new footsteps, the timing for the footsteps is determined by the gait you have chosen, walk, run, or jump, and the parameters for that gait. Gait parameters are in the Footstep Creation rollout in the Motion panel.

After creating the footsteps, you can change footstep timing in Track View. See *Editing Footsteps in Time: Track View* (see page 60).

Gait parameters are only one way to define the timing and nature of the biped's gait. For a more complete description of gaits and other Biped parameters that alter the nature of the biped's motion, see *Adjusting Vertical Dynamics* (see page 63).

### Walking



Turn on Walk before creating footsteps.

In a walk, at least one foot is always in contact with the ground. The periods where one or both feet are in contact with the ground are known as support periods. If both feet are on the ground, this is known as a double support period.

The gait parameters of a walk are:

**Walk Footstep.** The number of frames that each footstep remains on the ground. Default=18.

**Double Support.** The number of frames in a double support period, that is, when both feet are on the ground. Default=3.

Changing these values changes the biped's walk behavior.

### Running



Turn on Run before creating footsteps.

In running, there is a period between each support period in which the body is airborne. While it is airborne, it moves forward horizontally at a constant speed. In general, the longer the body is in the air, the higher it must fly after lifting off from the supporting foot.

The gait parameters of a run are:

**Run Footstep.** The number of frames that each footstep remains on the ground. Default=6.

**Airborne.** The number of frames that the biped is airborne, that is, when neither foot is on the ground. Default=9.

Changing these values changes the biped's run behavior.

### Jumping



Turn on Jump before creating footsteps.

Jumping is a special case of running. Both feet are in contact with the ground at the same time, or airborne at the same time. As with running, forward motion is horizontal and constant, but vertical motion depends on the length of the jump.

The gait parameters of a jump are:

**Feet Down.** The number of frames in which both feet are on the ground. Default=5.

**Airborne.** The number of frames that the biped is airborne, that is, when neither foot is on the ground. Default=15.

Changing these values changes the biped's jump behavior.






## Creating Multiple Footsteps

Multiple footstep creation is the easiest way to create a sustained walk or run, have the biped climb or descend a flight of stairs, hop repeatedly, speed up or slow down, and so on. Multiple footstep creation places the footsteps for you, generating perfectly timed and spaced footsteps.

### See also

*Create Multiple Footsteps (Walk) (see page 250)*

### To create multiple footsteps

1.   On the Motion panel General rollout, click Footstep Mode. Biped makes Footsteps the active sub-object selection level. You are now in Footstep mode, where you can create, activate, or edit footsteps.
2. In the Footstep Creation rollout, choose the gait you want to create: Walk, Run, or Jump.
3.  In the Footstep Creation rollout, click Create Multiple Footsteps. The Create Multiple Footsteps dialog appears.
4. Set the multiple footstep parameters, and then click OK.
5. Click on Create Keys For Inactive Footsteps. Now you can view your animation by clicking play.

## Creating Individual Footsteps

Creating footsteps individually is useful for complicated footwork, such as dancing.

Individual footstep creation allows you to place each new footstep carefully where you want it. The footsteps will locate and orient themselves on the active grid.

See *Creating Multiple Footsteps (see page 57)* for how to generate a simple walk, run, or jump.

There are two ways to create individual footsteps. You can create footsteps at the current frame. Then the first footstep created will begin at the current frame, and the following footsteps will extend in time from the first footstep. If you attempt to create a footstep at the same time as an existing footstep on the same side, an alert appears and you are not allowed to create the footstep.

You can also append footsteps onto the end of the existing footsteps. Then Biped computes the frame at which the first footstep should be created based on the chosen gait and the existing footsteps. This option is disabled if there are no existing footsteps.



If inactive footsteps exist, you can only create footsteps within the area of the inactive footsteps or directly before or after the inactive footsteps. This is because the complexity of Biped's algorithms make it necessary to disallow the existence of more than one sequential set of inactive footsteps simultaneously.

You choose the gait and how to create footsteps in the Footstep Creation rollout.



Footstep creation and gait buttons

### To begin footstep creation

1.   On the Motion panel General rollout, click Footstep Mode. Biped makes Footsteps the active sub-object selection level. You are now in Footstep mode, and can create, activate, or edit footsteps.
2. In the Footstep Creation rollout, choose the gait you want to create: Walk, Run, or Jump. See *Adjusting Vertical Dynamics* (see page 63) for more details about the different gaits and how Biped creates them.
3. In the Footstep Creation rollout, use the spinners to set the parameters for the chosen gait. See *Footstep Timing: Gait Parameters* (see page 56).

### To create footsteps beginning at the current frame

1. Click Create Footsteps (at current frame).
2. Click in a viewport to create a footstep. Continue clicking to create more footsteps.  
**Tip:** Use the top viewport to create the footsteps.
3. When you are done creating Footsteps, click on Create Keys For Inactive Footsteps. Now you can view your animation by clicking play.

### To append footsteps onto the existing footsteps

1. Click Create Footsteps (append).
2. Click in a viewport to create a footstep. Continue clicking to create more footsteps  
By default, footsteps alternate from foot to foot. The first click creates a right footstep, the next click creates a left footstep, and so on. Look at the prompt line and the cursor to see which type of footstep will be created next.

Press Q before you click to change the default alternation. For example, if you create a right footstep and then press Q, the next click in the viewport creates another right footstep. In viewports and in Track View, left footsteps are blue and right footsteps are green, unless otherwise specified in the Display Preferences dialog.

**Tip:** A Top viewport is usually best when you create footsteps individually.

Note: During footstep creation, 3D Studio MAX remains at the current frame; however, Biped creates the footsteps sequentially in time, which you can see if Track View is open. If the footsteps you create require more frames than are in the active time segment, Biped extends the active time segment, which can create new frames.

## Editing Footsteps in Space

This section applies to both inactive and active footsteps. It does not, however, discuss *adaptation* (see page 721), what happens to the biped's keyframes when you change the position and orientation of the biped's active footsteps. See *Adapting Biped Keys to Footstep Edits* (see page 66) for a discussion of adaptation.

There are several different ways to edit footsteps in space after you've created them:

- You can use standard 3D Studio MAX controls to select, move, rotate, or delete footsteps.
- You can use Footstep mode controls in the Motion panel to edit the footsteps.  
With Footsteps as the active sub-object selection level (this happens automatically when you turn on Footstep mode), you can select and move or rotate footsteps, either individually or multiply.

The Scale transform does not apply to footsteps, since the size of the footsteps is controlled by Biped. The footstep size always perfectly matches the size of the foot. It even changes the shape to match the proportion of the toes to the feet, and to match unevenly sized left and right feet.

Note: If you scale a foot in Figure mode, when you exit Figure mode, **character studio** will recalculate the footstep display size and the footstep dynamics, automatically.

Biped's Footstep Operations rollout provides ways to scale the stride length and width of a set of footsteps, or to bend the footsteps' path.


See also

*Footstep Operations rollout (see page 258)*

## Activating and Deactivating Footsteps

Once you have created footsteps, you need to activate them. Biped generates default keys to go with the footsteps.

### To activate footsteps

-  While still in Footstep mode, click Create Keys For Inactive Footsteps in the Footstep Operations rollout.

Biped generates the motion keys.

After you've activated the footsteps, you can preview the biped motion. You can fine-tune the motion by editing footsteps or keys.

You can deactivate activated footsteps. There are two main reasons for doing so:

- Editing footsteps more quickly  
As soon as you edit an activated footstep, Biped adapts motion keys. This can take time. You might find your work goes more quickly if you deactivate the footsteps, edit them, and then activate them again.
- Restoring the default motion for the footsteps.  
If you have added user keys to a footstep sequence but then are not satisfied with the result, you can restore the original, default motion keys by deactivating the footsteps and then activating them again.

Biped restricts many operations when inactive footsteps exist. This is because it is difficult to adapt keys near the borders of inactive footsteps. Also, it is unwise to adapt keys to footsteps that are extremely unlike the final footsteps you desire to create.


The following restrictions apply when inactive footsteps exist:

- You cannot set, delete, move, or edit keys in any way; tracks are disabled in Track View.
- You can create footsteps only when the new footsteps are among and adjacent to existing inactive footsteps.
- You can delete only inactive footsteps; at least one inactive footstep must remain.
- Also, you are not allowed to deactivate a set of non-sequential footsteps.

**To deactivate footsteps**

1. Select the footsteps to deactivate.

The footsteps you select must be in sequence. Biped doesn't allow you to deactivate a non-sequential set of footsteps.

2.  Click Deactivate Selected Footsteps.

The footsteps are deactivated. Viewports and Track View display them in saturated colors again.

Note: Deactivating footsteps does not delete the keys generated when you activated the footsteps. The biped still moves as it did before, until you click Create Keys once again. However, editing inactive footsteps doesn't affect the keys, and activating the footsteps again generates new default keys that replace the previous keys.

Note: Deactivating/reactivating footsteps destroys user-defined keys. Deactivate footsteps only if you want to delete user-defined keys and return keys to their default values.

**See also**

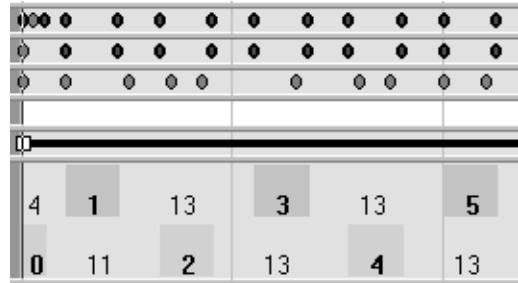
*Footstep Operations rollout (see page 258)*

**Editing Footsteps in Time: Track View**

This section applies to both inactive and active footsteps. It describes the mechanics of editing footsteps in Track View. It does not, however, discuss adaptation, what happens to the biped's keyframes when you change the timing of the biped's active footsteps. See *Adapting Biped Keys to Footstep Edits (see page 66)* for a discussion of adaptation.

Footsteps have their own special display in Track View. To see footstep display, expand the objects

branch of the biped and the parameters branch of Footsteps.

**Footsteps in Track View**

The left leg's footsteps are blue and the right leg's footsteps are green. In Track View Orange footsteps are *sliding footsteps (see page 732)*. Inactive footsteps are more saturated values of blue and green, while active footsteps are pale blue and green. The left edge of each footstep block indicates when the foot touches the ground and the right edge indicates the last frame during which the footstep is on the ground. The space between footsteps indicates the amount of time the foot is off the ground in between footsteps.

You can change the length or spacing of footsteps in Track View to change their timing.

**To select a footstep**

- Click near the center of the footstep's area. The footstep's border turns white to indicate it's selected.

**To select multiple footsteps, do one of the following**

- Click to select a footstep, and then use CTRL+click near the center of other footsteps to add them to the selection set.
- Drag in the footstep track.

Dragging creates a selection box that selects each footstep it touches.

### To move footsteps in the time sequence

- Drag selected footsteps forward or backward.

### To change the duration of a footstep

1. Click the left or right edge of a footstep.  
In addition to a white border, a small white dot appears to indicate the border is adjustable.
2. Drag to make the footstep longer or shorter.  
You can adjust a selection of multiple footsteps this way.  
You can also adjust any combination of left edge, right edge, and center selections.  
Dragging in Track View adjusts any combination of edge or center selection.
3. Right-click the footstep track to display the Footstep Mode dialog, described in *Footstep Mode Dialog* (see page 267).

## Restrictions

This section does not discuss restrictions relating to adapting keys to changes made in Track view to the footstep track. See *Adapting Biped Keys to Footstep Edits* (see page 66) for this information.

### The following restrictions apply to footstep timing:

- There must be at least one frame in-between any two sequential footsteps on the same side (left/right).
- Footsteps cannot appear in negative frames.
- Every footstep must be at least two frames long.

If you violate these restrictions when performing any operation in Track View, upon mouse up,

Biped restores the footstep track to the state it was in before the change was attempted.

Also, if you make a change that causes two footsteps on the same side (right/left) to overlap, or moves a footstep completely in front of or past another one on the same side, upon mouse up Biped restores the footstep track to the state it was in before the change was attempted.

Below is a list of operations for the footstep track in Track View that either have not been implemented, do not work as expected, or should not be used at all.

## Edit Keys

- Add Keys does nothing.  
See *Creating Individual Footsteps* (see page 57) and *Creating Multiple Footsteps* (see page 57) for how to add footsteps.
- Cloning footsteps in Track View does nothing.
- Align Keys does not work correctly.  
**Warning:** Do not use Align Keys for footsteps.

## Edit Time

- Reduce Keys does nothing.
- Reverse Time does nothing.
- Cut/Copy/Paste do nothing.  
See *Splicing Biped Motion* (see page 68) for how to copy and paste footsteps.
- Delete Time does nothing.  
It does not make sense to delete time in the footstep track. Delete selected footsteps instead, or use the Track View Edit Keys mode Delete Keys button.

There is no function curve for the footstep track, so function curve editing is not available.

If you want to scale a biped's animation, you can either adjust the footstep track's range bar, or use the Scale Time button in Edit Time mode. These two methods will scale both the time and duration of the footsteps. If you use the Scale Keys button in Edit Keys mode, the start time, but not the duration of the footsteps will scale. This is typically not desirable.

**Warning:** Do not use the Re-scale Time option in 3DS MAX's Time Configuration dialog. If you must use this, you'll have to jiggle the footstep track's range bar in Track View immediately afterwards to get the rest of the biped's tracks to adapt to the scaled footstep track. Even so, there might still be errors, so this is not recommended.

**Warning:** Since Biped does some internal quantization, it is strongly recommended to leave on the Snap Frames button in Track View when editing any biped track.

See also

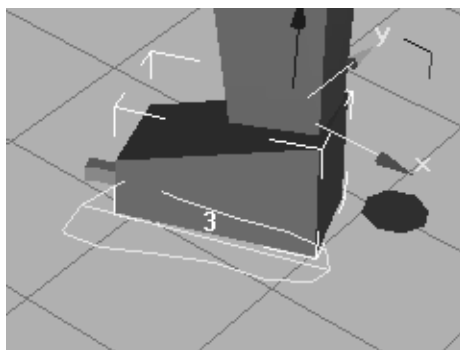
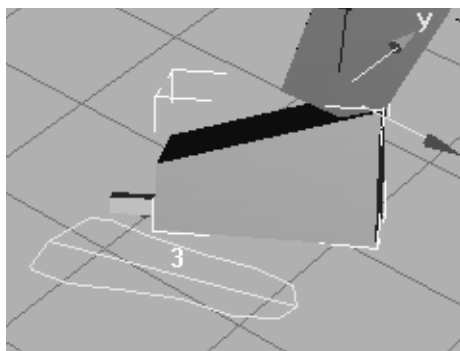
*Track View (see page 262)*

## Sliding Footsteps

Changing biped foot key parameters enables the biped feet to move or slide during a footstep period. This feature is also available for motion capture file import to allow the biped feet to slide or pivot. Sliding footsteps display with a line through the middle of the footstep in the viewports and as an orange color in Track View. Use this feature in a dance motion where the character pivots on the balls of its feet or if the character is standing or walking on rocky terrain and its toes must curl down for grip, or if a character's feet must skid.

A sliding key is created by using the Set Sliding Key button in the IK Key Info rollout. This turns

off Join to Previous IK Key allowing the footstep to be positioned anywhere.



On a sliding footstep the biped foot can be placed anywhere relative to the footstep gizmo.



## Motion Capture Import

Enable this feature automatically when importing motion capture data using the Sliding Tolerance and Sliding Angle controls in the *Motion Capture Conversion Parameters* dialog (see page 192).

## Procedure

**To enable foot motion on a footstep**

1. Select a biped foot.

2.  On the IK Key Info rollout, click the Next Key and Previous Key buttons to locate a biped foot key for the footstep you want to change.
3.  When you've located a footstep key, click Set Sliding Key in the IK Key Info rollout.
4. You are now able to rotate and position the biped foot during the footstep.  
The footstep has a line through the middle of it indicating a sliding footstep.

## Adjusting Vertical Dynamics

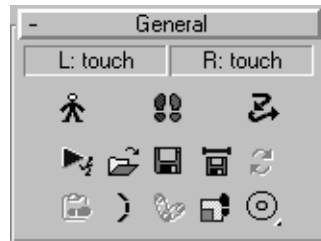
Because Biped's keyframe interpolation is based on the mechanics of two-legged characters, it varies according to whether the feet are on or off the ground. When you activate footsteps, the default keyframes for the leg tracks are given states that correspond to the footstep pattern from which they were created.

A leg's motion has four phases, beginning with the foot on the ground. Then the foot lifts, moves through the air, and returns to the ground again. Biped divides this motion into four phases as follows:

- **Touch** occurs at the leg keyframe where the leg's foot first touches the ground and always corresponds with the start frame of a footstep in Track View.
- **Plant** occurs after touching, and before lifting. It is always in between the start and end frames of a footstep in Track View.
- **Lift** occurs at the keyframe where the leg's foot lifts off the ground and always corresponds with the end frame of each footstep in Track View.
- **Move** occurs while the foot is in the air and is always in the intervals in between steps in Track View. In walking, while one foot moves the body is supported by the other leg. In running or jumping, while a foot moves there is a period where the body is not supported, and moves in midair.

Biped displays the state of each foot in the General rollout of the Motion panel.

Regardless of the footstep pattern, in order to properly interpolate your character's motion, Biped will always generate and maintain leg keyframes at each lift and touch transition in the steps.



Footstep states in General rollout

## Dynamics of Walking

For our purposes, a *walking gait* (see page 738) is defined to be any footstep pattern in which there is always at least one foot on the ground. Of course, this is a more general use of the term since it includes many irregular and non-periodic footstep patterns, and even standing in place. The main feature of walking gaits is that there is no dynamics of ballistic motion at work.

When Biped first activates footsteps to create default keyframes for walking footstep patterns, it attempts to create as few keys as possible to establish a fundamental framework for you to work from. In general it will create:

- Lift and touch leg keys at each support transition, several plant leg keys to both

flatten and pivot the foot, and two move leg keys, one in the middle of the swing of the leg, and one to straighten it out just before its heel strikes.

- Vertical keys at each support transition and one in the middle of each support period.
- Keys for other body parts according to the rhythm of walking.

## Dynamics of Ballistic Gaits: Running, Hopping, and Jumping

A *ballistic gait* is defined to be any footstep pattern in which there are periods with no feet on the ground, causing the biped to become airborne, or ballistic. For example, running, hopping and jumping are ballistic gaits. Of course, in practice, your footstep pattern will typically combine both walking and ballistic patterns.

When Biped first activates footsteps to create default keyframes for ballistic footstep patterns, it creates:

- Lift and touch leg keys at each support transition, several plant leg keys to both flatten and pivot the foot, and one move leg key in the middle of the swing of the leg.
- Vertical keys at each support transition.
- Horizontal keys at each support transition.
- Keys for other body parts according to the rhythm of running.

### Airborne Vertical Dynamics

Each airborne period in the motion always begins and ends with keys for the vertical and horizontal position of the biped's center of mass. These keys act to define the position of the biped at lift-off and touchdown during its flight.

When the biped is in the air, the vertical motion is governed by physically based dynamics. Its trajectory is a function of gravity (or gravitational acceleration), the height of the vertical key at lift-off and touchdown, and the amount of time in the air.

*GravAccel* (for Gravitational Acceleration) lets you scale the height of the airborne periods. The greater this value, the greater the height. If the biped appears to be going too high, reduce this value; if the biped goes too low, increase it. Each biped has its own Gravitational Acceleration value. Default=Based on the height of the biped.

If feet are the active 3D Studio MAX units and the biped is about 5 feet 10 inches tall, then Gravitational Acceleration equals about 32, for 32 feet per second per second. For other biped heights, Biped scales this value to naturalistically fit the scene; the Gravitational Acceleration value also changes to agree with other unit systems, such as metric.

The Gravitational Acceleration spinner is in the Animation Properties rollout of the Motion panel.

**Warning:** If there is not enough gravitational acceleration or time in the air to account for differences in vertical height (such as a biped falling from footsteps on a ledge in slow motion), the system simply puts the biped on the ground at the touchdown frame, causing a discontinuity. You can fix this by either raising GravAccel or increasing the ballistic interval in Track View. See *Editing Footsteps in Time: Track View* (see page 60).

At any point in the design process, you are free to alter the value of GravAccel. If you do so after there are active footsteps with keys, your keyframes will be adapted to match the new value of gravity.



## Springing and Landing Dynamics

Biped ensures that the upward vertical speed at lift-off and the downward speed at touchdown match the requirements of gravity, the difference in heights at lift-off and touchdown, and the time spent in the air.

- Springing dynamics occur between the vertical key preceding lift-off until the key at lift-off.
- Landing dynamics occur between the vertical key at touchdown and key that follows.

You can, therefore, adjust the timing of spring and landing dynamics by setting, or deleting, vertical keys that precede lift-off keys and follow touchdown keys. You can also vary the apparent depth of the spring and landing by setting the ballistic tension at the lift-off and touchdown vertical keys.


## Ballistic Tension

Increase to 1.0 to make the spring or landing shallow and stiff (with high tension in the legs). Decreasing the tension to 0.0 makes the legs to appear to be less stiff. The default is 0.5.

You can edit Ballistic Tension at Vertical track keyframes where the body touches down. On the Motion panel Track Selection rollout, Body Vertical must be selected.

If there are more than three vertical keys during a support period, you can also edit Ballistic Tension for the lift-off key; otherwise, Biped uses the same value for touchdown and lift-off since it assumes that there is only one vertical dip in the motion.

## To set Ballistic Tension at vertical track keyframes

1.  Select the biped's Body Vertical track.
2. Move to the Vertical track keyframe you want to adjust.
3. Adjust the Ballistic Tension value.  
Adjusting the ballistic tension changes the amount of crouch before a jump, and the amount of dampening that occurs after landing.

---

## Adjusting Biped Keys in Track View

In general, except for the footstep track, editing biped keys in Track View is the same as editing keys of any sort in 3D Studio MAX. There are a few differences. Some biped keys are locked, there are some restrictions to Track View operations, and function curve editing is not supported.

## Locked Keys

Locked keys for the vertical Center of Mass track are created at the lift and touch frames of a ballistic gait in a footstep animation. These are the only keys that are locked. These keys can be moved in time by changing footstep duration in Track View.

## Restrictions

You are not allowed to add keys beyond the last footstep in the animation or before the first footstep in the animation. You are also not allowed to move keys into the areas outside of the first and last footsteps.

**Tip:** If you want to animate a biped performing various movements with his upper body while standing, extend the length of the standing

footsteps to cover the length of the standing animation.

## Edit Keys

If Edit Keys is enabled on the Track View toolbar, the following functions are possible:

- **Clone Keys.** Use Shift+drag to clone and position selected key(s) in time.
- **Scale Keys.** Turn on Scale Keys on the Track View toolbar and scale selected keys. The scale center is the current frame.
- **Add Keys.** Turn on Add Keys on the Track View toolbar and add keys to a biped track.

## Edit Time

If Edit Time is enabled on the Track View toolbar, the following functions are possible:

- **Scale Time.** If your animation is purely freeform, you can scale time for all tracks of the biped. If it is a footstep-driven animation, you are restricted to scaling time for the footstep track. Turn on Scale Time on the Track View toolbar, select the Footstep transform track in the Track View hierarchy window then select and drag to scale footsteps time in Track View edit window.
- **Insert Time.** If your animation is purely freeform, you can insert time for all tracks of the biped. If it is a footstep-driven animation, you are restricted to scaling time for the footstep track. Turn on Scale Time on the Track View toolbar, select the Footstep transform track in the Track View hierarchy window then select and drag to scale footsteps time in the Track View edit window.
- **Copy and Paste Time.** For all biped tracks other than the Footstep track. You could for example copy time from the left arm track

and paste it to the right arm track. This will copy the left arm keys to the right arm. You can also paste time onto another biped in the scene.

---

## Adapting Biped Keys to Footstep Edits

One of the most powerful features in Biped is the ability to adapt keyframes to edits you might want to make in your footstep pattern. By analogy, the footsteps become a kind of *gizmo* for manipulating the keyframes of your character's animation. In most cases, edits you make to footsteps act upon your keys in an intuitive fashion. The purpose of this section is to clarify the basic principles that Biped follows during the adaptation process.

### Adapting Keys to Footstep Space Edits

To move or rotate active footsteps, follow the method described in *Editing Footsteps in Space* (see page 58). As soon as you release the mouse, however, Biped immediately adapts the keyframes that are influenced by the edit. The following keyframe tracks are influenced in the vicinity of the edited footsteps:

- Body's Horizontal Keys change to step or hop within range of new footstep locations
- Body's Vertical Keys change to match possible changes in stride length, since the body must be lower to step longer distances
- Body's Turning Keys change to bank into turns created by changes in path curvature or body speed
- Right or Left Leg Keys in Move State must be adapted to step between new locations
- In all cases, any edits or adjustments you have made to the keyframes themselves are preserved in the adaptation process. For

example, if you set the body to be kneeling low and banked to the right, altering a nearby footstep will maintain your height and turning bias as much as possible.

## Adapt Locks

To avoid automatic adaptation when making Footstep Space edits, you can use Adapt Locks to lock specified tracks so that Biped doesn't adapt them when you edit footsteps.

### To lock one or more tracks so Biped doesn't adapt them

1. On the Motion panel, on the *Animation Properties* rollout (see page 181), in the Adapt Locks area, select the option for the keys you want to lock.
2. Edit footsteps.

The tracks you selected as locked are unaffected by footstep editing.

## Adapting Keys to Footstep Time Edits: Track View

To edit active footsteps in time, you follow the same methods described in *Editing Footsteps in Time: Track View* (see page 60). As soon as you release the mouse, however, Biped immediately adapts the keyframes that are influenced by the edit.

A fundamental factor in how your keyframes are adapted is whether or not the sequence of leg support transitions have changed order. For example, such a leg support sequence might be:

1. Both legs on ground, then
2. Left leg on ground, then
3. No legs on ground, then
4. Right leg on ground, then
5. Both legs on ground

In this sequence, the biped is standing, then takes a hop from the left to right leg, and is then standing again. If you were to edit a footstep to, for example, close the airborne gap between footsteps to eliminate the hop, the leg support sequence would change since the no legs on ground phase of the sequence would be missing. In essence, changes to the leg support sequence mean that the gait has changed. In this case, from a hopping step to a walking step.

The general rule is:

- If the leg support sequence retains the same order, keyframes are simply synchronized to match the new footstep pattern relative to their place along the starting and ending frames of each footstep block that has been altered.
- If the leg support sequence does not match, default leg keyframes are created and locally inserted to fill in the parts that are different.

This is somewhat intuitive since there are no obvious rules for making walking keys adapt to running keys or vice versa. In effect, when you alter the leg support sequence, you are creating entirely new motions.

**Warning:** Editing a jumping sequence so that the feet hit or leave the ground slightly out of phase introduces a change in the leg support sequence, since a one leg on ground phase has been introduced. Hence, even in this case, the leg keys will be locally replaced with default keys.

---

## Motion Interdependencies Between Body Parts

The motion of a character is greater than the sum of its parts. Any time the position of the body's center of mass or pelvis changes, the position of the legs must change as well. When you set a key, Biped is able to isolate these interdependencies to ensure that the feet move from footprint to footprint in the same way.

Interdependencies between the upper body and legs are updated after a key has been set, sometimes causing the appearance of a pop in the leg position.

For example, if you rotate the spine forward, the horizontal position of the pelvis will move backward in order to balance. Because the pelvis is further back, when you set a key for the spine, interdependent leg keys adjust themselves so the feet travel the same relative distance between footprints.

When you set a leg key while the foot is on the ground, Biped recalculates the location of the foot, based on the natural roll of the foot from its point of initial contact with the ground. The pivot points are based on collisions between the footstep plane, the two corners of the heel, and the bones that connect the toes' links.

Biped's ability to preserve interdependent relationships allows you to easily experiment and improvise with motion since you are assured that edits to one body part will not corrupt the integrity of other parts.

As a general rule, changes to the body tracks, spine, and pelvis cause the legs to adjust. Changes to the legs do not affect vertical position.

---

## Freeform Editing Between Footsteps

For certain types of animation it is necessary to turn off Biped's vertical dynamics. For instance, if you want to make a biped fly or swim or row a boat, dynamics computations must be suspended so that you can control the vertical position of the biped.

You can create a freeform area in any part of your animation in which the biped is airborne. In a freeform area, vertical dynamics are suspended, and the vertical body path uses *spline dynamics* (see page 732).

### See also

*Footstep Mode Dialog* (see page 267)

---

## Splicing Biped Motion

You can copy the motion of a biped footstep sequence, and splice it either onto the end or into the middle of another footstep sequence. With this capability, you can build an extended, or cyclic, motion sequence out of shorter sequences. You can also copy footsteps from one biped and paste them onto another biped.

**Tip:** Use motion splicing when you want to edit segments of a single *.bip* file. Use scripting when you want to build longer animations by connecting finished *.bip* files. For information on scripting, see *Motion Flow Mode* (see page 227).

### See also

*Footstep Operations Rollout* (see page 258)

## Retargeting Biped Motion

One of the more powerful features of Biped is the ability to retarget or map the motion of any biped onto any other biped. If you map the motion of a biped without a tail onto a biped with a tail, default tail motion for the biped with a tail will be computed. Default motion will also be computed when mapping the motion of a biped with fewer links in the legs, spine, or neck onto a biped with more links in the legs, spine, or neck. There are a few ways of performing motion mapping. You can:

- Go into Figure mode and change the structure of your biped. When you exit Figure mode, the new structure of the biped will adapt to the existing animation.
- Save a *.bip* or *.stp* file from one biped, and load it onto another biped of a totally different structure and size.
- Copy footsteps from one biped, and paste them onto the footsteps of a differently structured and scaled biped.

### Scale Stride Mode

Scale Stride mode gives you control over whether or not certain aspects of this motion mapping occur.

If Scale Stride mode is active (the default):



- When you exit Figure mode after loading a *.fig* file or changing the biped's leg length, pelvis width, or height, the locations of the footsteps will be scaled to match the leg length and pelvis width, and gravity will be changed to match the new height of the biped.
- When you load a *.bip* or *.stp* file, the locations of the footsteps in the file will be scaled to match the leg length and pelvis width of the existing biped. Gravity will be

adjusted to be proportional to the gravity stored in the file. (A motion stored in a *.bip* or *.stp* file has a gravity value associated with it.)

- When you paste footsteps copied from one biped onto another biped, the locations of the footsteps in the buffer will be scaled to match the leg length and pelvis width of the existing biped. Gravity will be adjusted to be proportional to the gravity stored in the file. (A motion stored in a *.bip* or *.stp* file has a gravity value associated with it.)

If Scale Stride mode is not active, no computations will be performed in any of the above cases. Then you might see your biped moving over footsteps that are spaced inappropriately far apart or close together for the size of your biped. Typically, you should leave Scale Stride mode active, unless you want to maintain the spatial relationship between the biped and other objects in your scene.

### To turn off Scale Stride mode

1.  On the General rollout, click Scale Stride Mode to turn it off.  
The button changes to indicate stride scaling has been turned off.
2.  Use Figure mode to edit the figure.  
When you return to Keyframe mode or Footstep mode, the biped's stride length is unchanged, regardless of the biped figure's new proportions.

---

## Loading and Saving Biped Step Files

Step files save footsteps, but don't save body keyframes. They are an ASCII file format that enables developers to write programs that generate step files for biped motion. The online document *stp.rtf*, provided with **character studio** in the `\cstudio\docs` directory, describes the *.stp* format.

### To load footstep data

1. Select the biped to animate via saved footsteps, and make sure you are not in Figure mode.
2. Click Load File.
3. In the file dialog, choose Step Files (*.stp*) as the file type to load.
4. Choose the footstep file to load, and then click OK.

The footsteps will be loaded onto the biped and new default keys will be generated to match the footsteps.

### To save footstep data

1. Select the biped whose footsteps you want to save, and make sure you are not in Figure mode.
2. Click Save File.
3. In the file dialog, choose Step Files (*.stp*) as the file type to save.
4. Enter a name for the footstep file, and then click OK.

---

## Troubleshooting

As discussed in *Adjusting Biped Keys in Track View* (see page 65), certain keys in the leg and vertical tracks must exist for a Biped script to run. At the first frame of every footstep, there must be a leg

key in the touch state for the leg corresponding to that footstep (left/right). See *Adjusting Vertical Dynamics* (see page 63) for a description of leg states. At the last frame of every footstep, there must be a leg key in the lift state for the leg corresponding to that footstep (left/right).

For every footstep edge surrounding a ballistic (mid-air) period, there must be a horizontal and a vertical key. Any leg key that occurs during a footstep on its side should be in the plant state. Any leg key that does not occur during a footstep on its side should be in the move state.

If any of these rules are violated, the biped's animation is unstable. Adaptation might not work correctly. When you create animation in Biped, there should be no way to create a script that does not follow these rules. However, if somehow you get into a situation where these rules have been violated, there is a way to fix your animation. There are some commands to change the lock state of keys and the leg state of leg keys manually. These commands are only available in the Biped Motion panel.

**Warning:** You should only use these commands as a last resort.

CTRL+ALT+L toggles the locked state of the vertical track's key at the current frame. You can watch the key turn from red to gray and back in Track View as you lock and unlock it.

---

## Converting Between Footstep and Freeform Animations

You may be more used to working with freeform animations than ones using footsteps, or just the opposite. Or you want to reuse part of a colleague's animation in yours, but the formats differ. You may also have used Save Segment to store a part of a scripted animation in Motion Flow mode, but now need that sequence in

footsteps format. In all these cases, you need a way to get from freeform to footsteps, or footsteps to freeform.

Using the Convert tool on the General rollout, you can convert a footstep animation to freeform, or in some cases, a freeform animation to footsteps.

**Tip:** In order to convert a freeform animation to a footstep animation, the file must be properly set up by locking the feet to World space using IK Blend before converting to footsteps. When creating freeform animations, set your leg keys to IK Blend=1.0 during the periods you want the feet to be planted, and set the move keys to IK Blend=0.0. This will ensure that the feet are locked and rid of unnecessary foot sliding as the body moves. When converting, if the leg keys have been set up this way, Biped will extract footsteps during any duration where IK Blend=1.0.

Convert is also useful if you have already converted from footsteps to freeform in which case the feet are assigned an IK Blend value of 1.0 for the keys that represented footsteps.

Note: If your freeform motion hasn't been prepared using IK Blend for converting to footsteps, you can also use Load Motion Capture File, from the Motion Capture rollout, to convert between formats. This will perform a proximity-based conversion.

## See also

*General Rollout (see page 147)*

### To convert between footstep and freeform animations

1. On the Motion panel General rollout, click Convert.

Depending on which is the current animation method, either the Convert to

Freeform dialog or the Convert to Footsteps dialog appears.

2. Change settings in the dialog as necessary and click OK.

If you're converting to freeform, the footstep patterns are removed from the viewport, gravity and dynamics are removed, and footstep mode is disabled. You are now working with a purely freeform animation.

If you're converting to footsteps, and the operation is successful, footsteps appear in the viewport, dynamics will be turned on, and footstep mode is enabled.

3. Convert to Freeform Dialog Settings.
4. Generate a keyframe per frame.  
Creates keys at every frame.

Note: Converting footsteps to freeform creates foot IK Blend values of 1 for the biped feet for the original footstep keys; this simplifies keyframing by putting the feet into world coordinate space, which prevents them from sliding when the biped is moved. These foot IK Blend values are also used when Convert is used to convert back to footsteps.

1. Convert to Footsteps Dialog Settings
2. Flatten Footsteps to Z=0  
All extracted footsteps are placed on the Z=0 plane.
3. Generate a key per frame  
Creates a key at every frame.

Footsteps are extracted based on foot IK Blend values equal to 1.

Note: An IK Blend value of 1 for the feet puts them into world coordinate space and prevents them from slipping while setting biped keys in a freeform animation.

## Advanced Biped Features

You'll need to use the advanced features if you want more control over the biped. Set IK constraints for the feet in a freeform walking motion. Change the interpolation between keys, or affect how the biped reacts to other objects in your scene.

The advanced features are:

- Tension, Continuity, and Bias (TCB)
- Dynamics Blend
- Ballistic Tension
- Balance Factor
- Object Space Object
- IK Blend
- IK Constraints
- Pivots
- Preset Set Keys
- Animatable IK Attachments
- Layers
- Mirroring Motion
- Editing keys on the Center of Mass trajectory

## Key Info Rollout

Tools in the Key Info rollout allow you to do the following:

- Find the next or previous key for the selected biped body part
- Use the Time spinner to slide a key back and forth in time
- Change Tension, Continuity, and Bias for a key
- Display trajectories for the selected biped body part
- Change Balance Factor, Dynamics Blend and Ballistic Tension parameters

## Vertical and Horizontal Center of Mass Tracks

When the Vertical Center of Mass track is selected, you can change the vertical dynamics of the motion on a key-by-key basis. When the Horizontal Center of Mass track is selected, you can change the balance factor for shifts in weight distribution.

For details on parameter settings, see *Key Info Rollout* (see page 161).

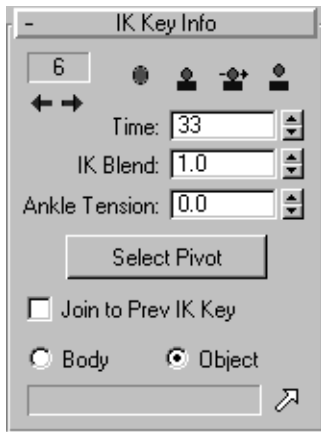
## Using IK Keyframe Parameters

Biped's inverse kinematics has three parameters that you can vary during the limb's motion by setting them at each key of the arm and leg keyframe tracks. As the limb moves through each key:

- IK Blend sets the motion interpolation to be a blend of forward and inverse kinematics. This allows you to blend swinging motions with hand/foot directed motions. Using an arm to move a hand is an example of forward kinematics. Using the hand to move the arm is an example of inverse kinematics. The default is 0.0, or full forward kinematics. An IK Blend = 1.0, is full inverse kinematics.
- Body/Object determines the reference coordinate space of the IK path. This allows you to move the IK path with your character's body or temporarily attach the hands/feet to follow other objects. The default is Body.
- Join to Previous IK Key determines if the key should be part of the previous key (same reference position as previous key).



## IK Blend



The IK Blend control is in the IK Key Info rollout; you set IK Blend while in Keyframe mode.

IK Blend can be set per key for any arm or leg track. The IK Blend parameter determines whether, at a particular key, an arm or leg is moving through it via inverse kinematics, forward kinematics, or a blending of the two kinematic solutions.

An IK Blend value of 0.0 means full forward kinematics. The arm (or leg) is moved by interpolating the rotations of the joints at the keys. The hand (or foot) tends to move along sweeping circular arcs in this case, and the motion appears to be motivated by the apparent swinging at the joints.

An IK Blend value of 1.0 means full inverse kinematics with the hand (or foot) being used as an end-effector. A spline path is computed through the keys of the hand, and the hand moves along that spline. Joint angles for the rest of the arm are computed to allow the hand to follow the spline. The motion, in this case, appears to be directed by the hand (or foot).

An IK Blend value between 0.0 and 1.0 means a combination of inverse and forward kinematics;

when IK Blend is closer to 0.0, forward kinematics are more heavily weighted in the solution, and when IK Blend is closer to 1.0, inverse kinematics are more heavily weighted.

It is best to use forward kinematics when you want the arms to swing, such as when a biped is walking. In the case of a boxer, however, since the hand should follow a directed path when punching, inverse kinematics should be used.

### To set the IK Blend value of a key

1. Select a single arm or leg track by selecting one or more parts of a biped's arm or leg. The IK Blend spinner and the other controls in the Kinematics area are enabled only when a single arm or leg track is selected.
2. Set a key if one doesn't already exist.
3. Set the desired value of IK Blend.

### Body/Object Option

By default, Biped calculates the kinematics solution using the coordinate system of the biped figure's center-of-mass, or the Body coordinate system. This means that the IK path of the hand (or foot) translates and rotates with your character as it moves. For example, the boxer's hand trajectory always moves relative to the weaving, bobbing, and turning of the boxer's body.

The Object option is used for animating dynamic links between the limbs and other objects in the scene.

The IK Blend control activates when a biped arm or leg (hand and foot) key is current.

- 0 with Body turned on is *forward kinematics* (see page 726), or normal biped space.
- 1 with Body turned on is *inverse kinematics* (see page 727), creates more straight-line motion between biped keys.

- 1 with Object turned on, but no Object Space Object specified, puts the limb fully into world space. Use this to control foot sliding in a freeform animation.
- 1 with Object turned on and an Object Space Object specified puts the biped limb into the coordinate space of the selected object; the biped limb follows the specified object.

**Body.** The biped limb is in biped coordinate space. When the center of mass is moved, the biped limb will follow.

**Object.** Object Space, the biped limb is either in World coordinate space or the coordinate space of the selected object. Coordinate space can be blended between keys.

### Join to Previous IK Key

This IK constraint is used to specify if a footstep is sliding or planted. If Join to Previous IK Key is on, then the biped foot maintains a reference position to the previous key, keeping the foot planted. If Join to Previous IK Key is off during a footstep, then the foot can be moved to a new position creating a sliding footstep.

Join to Previous IK Key also functions to lock the biped hands in space. Use Set Planted Key on the biped hands to lock their position in space.

**TIP** If you are having trouble with a foot or hand popping back to a previous 'keyed' position, check to make sure that the Join to Previous IK Key is unchecked, especially when using pivot points.

### See also

*IK Key Info rollout (see page 167)*

---

## Freeform Walking Animation Using IK Constraints

In a freeform walking animation you typically need two key types for the legs. If the foot is planted on the ground, the key must have IK Blend=1 in Object Space with Join to Previous Key turned on. If the foot is in the air between footsteps, then IK Blend=0 in Body Space with Join to Previous Key turned off. In the course of creating a walk or run cycle, you need to alternate these IK constraints for the feet. If the foot is sliding on the ground, then the IK constraints are IK Blend=1 in Object Space with Join to Previous Key turned off. All of these IK parameters can be found in the IK Key Info rollout.

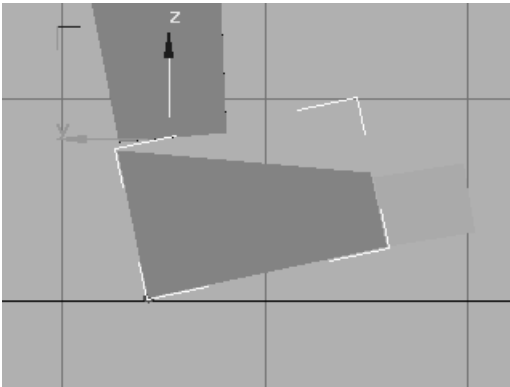
To speed up the process of applying these IK constraints, the IK Key Info rollout has three additional set key buttons: Set Planted Key, Set Sliding Key, and Set Free Key. By clicking one of these buttons, all the necessary IK constraints are applied automatically. For example, by clicking Set Planted Key, all of the IK constraints necessary are applied at once; IK Blend=1 in Object Space with Join to Previous Key turned on.

### See also

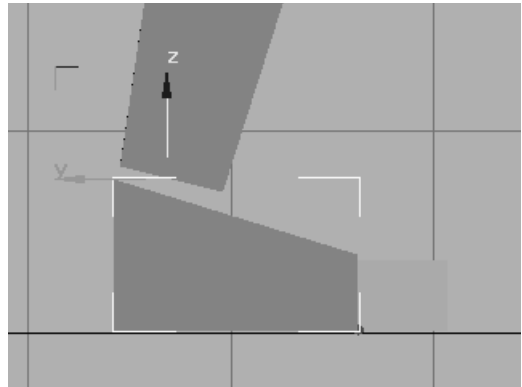
*IK Key Info rollout (see page 167)*

## Walking Keys

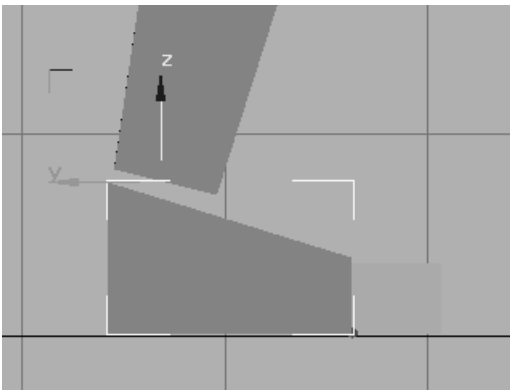
Lets examine the IK constraints for one footstep. The footstep rolls on the heel of the foot, then rotates down flat on the ground, then raises up on the ball of the foot, rotates at the end of the toes and finally lifts off of the ground.



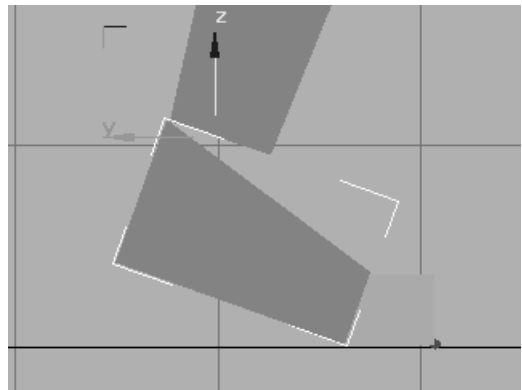
The foot is touching the ground at the heel. In the IK Key Info rollout, Set Planted Key is clicked to set IK Blend=1 in Object Space with Join to Previous Key turned on. A pivot is selected on the heel of the foot.



This key has the pivot on the ball of the foot as well. Set Planted Key is clicked. Two consecutive keys with the pivot at the ball of the foot are necessary to rotate the foot about the ball of the foot.



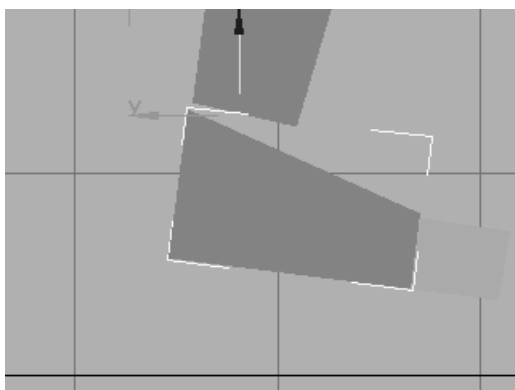
The next keyframe is also a planted key as the foot is flat on the ground. In the IK Key Info rollout, Set Planted Key is clicked. The pivot on the ball of the foot is selected.



After rotating the foot about the pivot at the ball of the foot another planted key is set with the pivot on the toe.



Here is another planted key with a pivot at the end of the toe; the foot rotates about the tip of the toe.



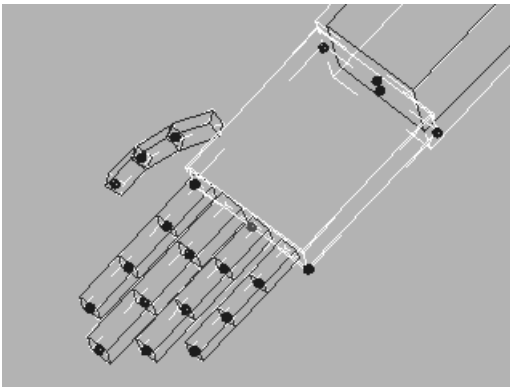
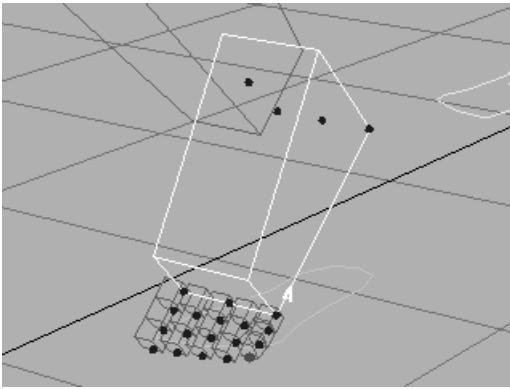
Here, Set Free Key is clicked to set a free key, the foot is off of the ground.

The cycle is repeated to create a walk or run cycle in a freeform animation. The pivot points are selected by turning on Select Pivot on the IK Key Info rollout, selecting a pivot in the viewports, turning off Select Pivot and then rotating the foot in the viewports.

Note that when Join to Previous is turned on, as it is with all planted footstep keys, the foot pivots on the previous keys pivot. If you find that the foot is not rotating around the visible pivot, remember that it is using the previous keys pivot. In such a case you must set two consecutive keys with the pivot at the same location, then the foot will rotate around the displayed pivot.

If you adhere to these rules in creating footsteps in a freeform animation then you can use Convert in the General rollout to easily change from a freeform animation to a footstep animation. Note that IK constraints in a footstep animation are applied exactly the same way as in a freeform animation. If you examine a footstep animation you will notice that the foot IK constraints follow the same rules as a freeform animation. Footsteps are now simply gizmos that define a foots coordinate system. A foot can slide and move relative to the footstep. Also, if you delete or add a key in a footstep animation the footstep duration is changed, before you could only change a footstep duration in Track View.

## Pivots (IK Extensions)



In both Freeform and Footstep animation, pivots allow you to rotate the biped's hands and feet around various points. The biped's hands and feet have the same number of pivots and pivot location is similar. By activating a pivot on the ball of a foot you can rotate the foot around the ball of the foot for example. Pivots are only active if the biped hand or foot is in world or object coordinate space. In a walking motion you can pivot on the heel first, then the ball of the foot and finally the toes. Pivots are essentially extensions of the IK chain.

By setting a planted key for the hand it is anchored in world space, you can move the biped or the collar bone and the hand remains planted. Pivots on the hands makes it easy to animate hands and fingers.

### Biped IK

To understand Biped's interactive IK limb manipulation, it is useful to distinguish two types of limb joints: primary and secondary.

The primary joints are the shoulder, elbow, hip, and knees. Animals use primary joints to coordinate the positioning of hands and feet since these joints have the most influence and flexibility in positioning tasks. Even the placement of fingers and toes over specific spots is the task of the primary joints.

The secondary joints are the wrists, ankles, toes, and fingers. These joints are typically used for grasping and support rather than positioning, so they most frequently exercise independent joint angle control suited to a particular task, such as rotating the foot to raise the heels during walking or the curling of the fingers around an object. Because these joints have little influence on end effector position, they are rarely engaged for positioning tasks. For example, if you want to place your finger on your nose, you will most naturally rotate only your shoulder and elbow, keeping your wrist stationary.

Biped mimics the IK behavior of "natural" systems. The primary joints are used for IK positioning while the secondary joints are independently and precisely controlled by the animator. The secondary joints will not rotate unless the animator explicitly selects and rotates them. Therefore, if an IK pivot is set, and the limb is interactively manipulated, (say, the finger is moved onto the nose), only the primary joints adjust to satisfy the IK constraint. The

secondary joints remain in place and will never “drift” in a IK solution, they simply obey the values set by the animator. Because there is no drift for the secondary joints, the animator does not have to tediously assign “tension” values to hold them in place.

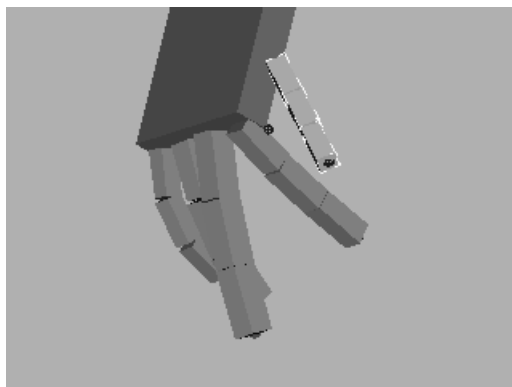
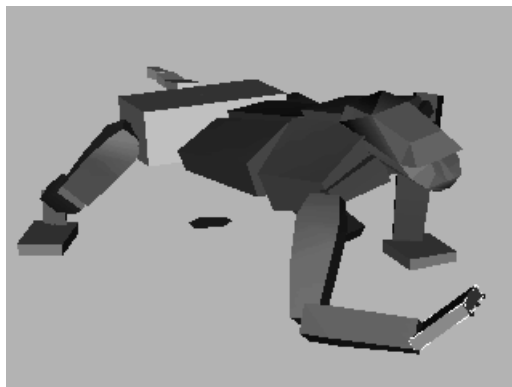
If an IK pivot point constraint has been set, a byproduct of Biped’s “natural IK” is that:

1. Interactively rotating a secondary joint will always adjust the primary joints to hold the IK pivot in place.
2. Biped’s “special rotations”, the forearm/calf X axis rotations, rotate the elbow about the axis from the shoulder to the wrist and rotate the knee about the axis from the hip to the ankle. These rotations give the animator a direct way to interactively rotate the primary joints while holding the IK pivot in place.
3. Interactively translating a limb part to move the IK pivot will only alter the primary joints.
4. Interactively rotating a primary joint will move the IK pivot with it.

After an IK pivot has been placed, applying 1) and 2) gives the animator exact control over all possible IK solutions. The IK pivots can be easily moved by applying 3) and 4). In general, these “rules” don’t need to be remembered since the system works in an intuitive way. The animator merely needs to position the pivots as desired, set the IK constraint, and then fine tune the posture of the limb, if desired, by rotating the secondary joints and using the special rotation to adjust the primary joints.

Note: During animation playback, the “Ankle Tension” parameter is used to set the relative importance of the interpolated ankle joint over the interpolated knee joint for intervals in

between keyframes. This is only relevant to Biped’s interpolation.



**Using IK constraints and pivots on the biped hands and feet allows you to animate quadrupeds.**

## See also

*IK Key Info Rollout (see page 167)*

## Procedure

### To use pivots

1. Select a biped hand or foot that’s in world or object space.
2. Turn on Select Pivots in the IK Key Info dialog.

3. Select a pivot in the viewports.
4. Turn off Select Pivots.
5. Rotate the hand or foot around the selected pivot.

Set two consecutive keys with the same pivot if you find that you are not rotating around the selected pivot.

## Animatable IK Attachments to 3D Studio MAX Objects

A biped can interact with other objects in the 3D Studio MAX scene. In 3DS MAX, links between objects are static, unless you're using the Link controller. For Biped, such attachments are "animatable" as well; during the course of an animation, the links between the hands, feet, and objects in the scene can change as your character interacts with them. This capability is useful for:

- Creating freeform motions (without footsteps and gravity) that require that feet or hands must be planted with IK, and then released. Examples are climbing a ladder, riding a bike, or rowing a boat.
- Motions that involve the temporary manipulation of objects, such as bouncing or kicking a ball, opening a door, or touching another biped.

**To create animatable IK attachments from a hand or foot to another object, in general terms you must do the following:**

1. Select the other object.
2. Define the *object space object* for the limb in the Kinematics area of the IK Key Info rollout. If this field is left blank, a default of World space rather than Object space is used.


3. Set keys for the limb to define when the hand or foot attaches, and when it detaches from the object.

To create a temporary attachment, you will typically set one key where you want to start the attachment and one key where you want to end the attachment. To do this you set each key to be in Object Space with an IK Blend of 1.0. The frames in between the two keys will then be interpolated to have IK Blend in Object Space.

The general rule is that both the preceding and following keys must be set to Object Space. This defines an Object Space interval, and the duration of a temporary attachment. If either key is in Body space, it is assumed that Body space was intended.

Note: If Object Space is selected with no Object Space Object specified, the IK constraints will be in World Space.

### To define the object space object

1. Select an arm or leg.
2.  In the Kinematics area of the IK Key Info rollout, click Select Object Space Object. In the viewports, the cursor changes to a plus sign. It turns to an X shape when it is over an object you cannot select as the object space object.
3. Click to select the object you will use for a potential attachment.

The object's name appears in the field at the bottom of the Kinematics area.

Now you are ready to set keys that will define when and where the linkages between the hand or foot and the selected object take place. Since the linkage, at this point, is only a "potential" one, you will need a device to actually make the hand or foot adhere to the object while you are setting keys.

You may do this by:

- Setting temporary IK constraints that hold the end of the limb in place by means of anchor buttons, or
- Snapping an IK constraint to a previously set key by means of the “Join to Prev Key” checkbox.

**To activate an anchor, do the following steps in either order:**

- Select the appropriate hand or foot and move or rotate it into the desired attached posture in relation to the “Object Space” object. If there is no object space object specified (in the Object Space field on the IK Key Info rollout), the hand or foot will then become anchored in World space.



- On the Keyframing rollout, click the button for the limb you want to anchor: Anchor Right Arm, Anchor Left Arm, Anchor Right Leg, or Anchor Left Leg.

Note: The arm or leg you select beforehand does not actually have to be the same as the arm or leg you are anchoring.



**To set keys that define animatable IK attachments**

1. At the frame for the start of the attachment, move your hand or foot to the desired location in relation to the Object (or World Space, if no object is specified).



2. On the Keyframing rollout, set an anchor for that limb so that the hand or foot remains attached in the same place.

This step is optional since you can also snap to the same pivot point by checking the Join to Prev Key option later on in step 6.

3.  On the IK Key Info rollout, click Set Key with IK Blend=1.0 and Object Space chosen or, as a shortcut, click the more convenient Set Planted Key button.
4. Click on Select Pivot Point and then select one of the end effector's red spheres displayed in the viewpoint
5. Move to the frame of detachment and reposition the hand or foot (if desired).
6.  Click Set Key with IK Blend=1.0 and Object Space chosen or, as a shortcut, if you wish to snap to the same pivot point as the previous key, click Set Planted Key button again. This will turn on the Join to Prev Key checkbox. If you wish to allow the pivot point to move, click Set Sliding Key. This will turn off the Join to Prev Key checkbox.
7. If you used anchors, turn off the anchors and move between the attachment and detachment keyframes, noting whether the hand or foot is attached for the desired interval.
8. If the hand or foot detaches during the interval you should delete the default keys for the attached limb in the interval between the start and end attachment keys.  
Note: The attached hand or foot may drift from its attach point as the object tries to maintain a smooth, continuous trajectory. To insure linear motion between keys, you should also set Continuity to 0 for the keys defining the attach interval. You can also adjust the tension to 50 to eliminate any undesired overshooting.



You can now continue to animate each limb's IK attached object conventionally -- moving or otherwise transforming it at different keyframes. The arm or leg moves to follow it using the kinematic solution specified by the IK Blend setting.

Note: The procedure above can be embellished by setting more than two consecutive keys with IK Blend=1.0 in order to move the hand or foot over the object in some specific way during the attachment. Also, subtle changes in the dynamics of the attachment are possible by manipulating the IK Blend values.

An eased detachment can be created, for example, by setting the key that follows the detach key to be in Object space with IK Blend=0.0. This will cause the blended IK values to remain in Object space while the IK Blend values ramp gradually down from 1.0 to 0.0.

Note: You can change the position of the elbow or knee without affecting the attach point of the hand or foot. While the anchors are turned on, select the forearm or a calf and rotate it around its X-axis. The hand and foot stays put while the limb is rotated.

Note: Saving a .bip file from a motion containing IK Blend keys will store the IK Blend information in the file. When the .bip file is loaded onto a new biped, the IK Blend settings and the body/object state are mapped onto whatever Object Space Object is already defined for the existing biped. If no Object Space Object is assigned, then the IK Blend setting will be stored in World space.

**Warning:** You must save the animation as a 3DS MAX file in order to save the animation keys for the Object Space Object.

See also

*IK Key Info rollout (see page 167)*

---

## Using Layers

Layers allow you to add successive layers of animation above the original biped animation. This is a powerful way of making global changes to your character animation. For example, simply add a layer, and rotate the spine forward at any frame, and a run cycle becomes a crouched run. The original biped motion is kept intact and can be viewed by switching back to the original layer. Layers can be viewed individually, or as a composite of all the animation in all the Layers. Layers behave like a freeform animation; the biped can adopt any position.

Layers allow you to easily adjust raw motion capture data containing keys at every frame. Simply add a layer, and keyframe the biped. You can also use layers to change the global position of the biped in a freeform or footsteps animation by adding a layer and moving the center of mass.

See also

*Layers Rollout (see page 178)*

---

## Mirroring Motion

Mirror, on the Keyframing rollout, mirrors the motion of the biped through both the X and Y axes of the World Coordinate System. The entire biped animation, including all footsteps and keys, is mirrored symmetrically through the center for the biped.

Use Mirror if you want to create the opposite of a motion you've already created. For instance, if a biped walks to the left and swings its right arm, mirroring the motion results in a motion where the biped walks to the right and swings its left arm.

Note: This feature mirrors only those biped tracks with at least one key present. The position of a keyless object is not mirrored.

See also

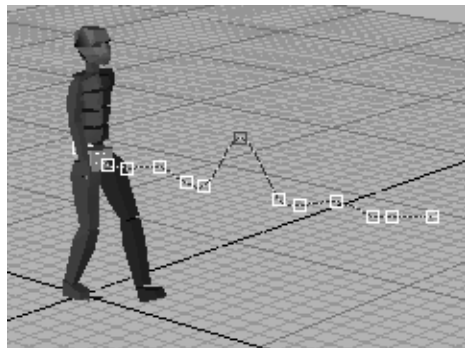
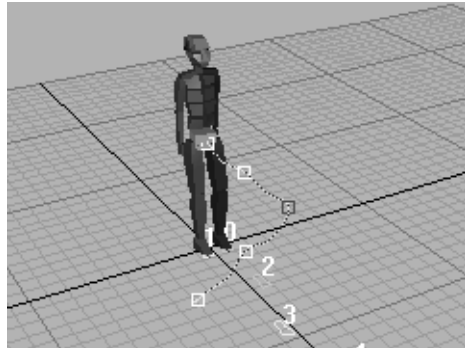
*Keyframing Rollout (see page 173)*

---

## **Bending the Center of Mass Track and Trajectory Key Editing**

By turning on biped trajectories, selecting the Center of Mass track, turning on Sub Object Trajectories you can select a key on the trajectory and either use Bend Horizontal in the Keyframing rollout to bend the trajectory about the selected key or simply move the key. You can move Center of Mass trajectory keys either horizontally or vertically. This allows you to view the entire trajectory of the Center of Mass and bend or edit the trajectory.

Note: You can select the Center of Mass horizontal or vertical tracks by clicking the appropriate button in the Track Selection rollout.



The first image illustrates bending of the trajectory, the second image illustrates editing keys on the trajectory.

See also

*Keyframing Rollout (see page 173)*

## Importing Motion Capture Data

In **character studio**, you can import both rotation and position type motion capture files.

- Biovision (*.bvh*) files contain limb and joint rotation data.
- **character studio** marker files (*.csm*) contain raw marker position data generated by a motion capture device: markers are attached to an actor during a motion capture performance.

Marker files typically require some calibration. If necessary the biped is sized to fit the markers first, then the biped limbs are oriented to align them to the markers. Marker files should be loaded with keys at every frame and no footstep extraction, this is required for calibration, and this also enables the calibration controls. Calibration controls are the second row of buttons on the Motion Capture rollout.

The *.csm* marker file format now supports a prop bone in either or both hands. See *Prop Bone* (see page 198).

Key reduction, track selection, footstep extraction, and clip looping are some of the options available using the Motion Capture import filter. Key reduction makes it easy to manipulate the biped and personalize the imported motion capture data. Extracting footsteps from motion capture data prevents inappropriate sliding feet, a common problem with motion capture data.

Typically you do not use an entire motion capture clip as is, unless you capture motions at a studio for your own production. You should become familiar with the body motion in the files you have, then use Motion Flow mode to cut portions of these files together to create

animation. For example, take a stretch motion in one clip, and combine it with the walking motion in another clip. The edited script can then be saved as a *.bip* file using the Save Segment command on the General rollout. Load this *.bip* for standard motion editing. This provides a good starting place for you to edit the result to your liking.

All motion capture controls are on the *Motion Capture* rollout (see page 186).

## Filtering Motion Capture and Marker Data

Motion capture and marker data typically have keys at every frame. Filtering motion capture data reduces keys, simplifying the job of altering or personalizing the motion data. Biped lets you filter the data of each track with its own filtering settings, so you have control over which nuances of motion you want to pick up without filling the rest of the tracks with unwanted keys. Filtering is done using the Motion Capture Conversion Parameters rollout.

Other filtering options include footstep filtering and extraction, looping the data, and importing a portion of the motion capture file.

**character studio 3** ships with a variety of raw (unfiltered) motion capture data files, in *.bip*, *.csm*, and *.bvh* formats. Some of the same data is available in filtered versions, either with footsteps or freeform. Try your own filtering adjustments on the raw versions of this data. Importing the raw data displays the original motion very accurately when you select Show Buffer on the Motion Capture rollout. Use the Motion Capture buffer as a guide when adjusting and refining the filtered data. Several tools are available in the Motion Capture rollout to aid you in this process.

Create your own library of imported and optimized motion capture data by saving *.bip* files for use with other characters, or as part of a longer script in Motion Flow mode. Use a biped that has no mesh attached with Physique.


**Tip:** Overall, you import the data, adjust it to your liking, and save it as a *.bip* file. You can also run standard *.bip* files through this filtering process to create loops or to extract footsteps from a freeform animation.

Note: Marker files, such as *.csm*, contain position data. Hierarchical motion capture files, such as *.bvh*, contain joint rotation data.



**Motion Capture rollout**

### To import a motion capture file

1. Select a biped in the viewports.
2.  On the Motion Capture rollout, click Load Motion Capture File.
3. Choose the file type: *.bvh*, *.bip*, or *.csm*. Search for files in the *cstudio\motions\mocap* directories.

**Tip:** *.csm* marker files, loaded for the first time, should be imported with no key reduction and no footstep extraction. This enables the calibration buttons. Marker files typically need some calibration.


4. Select a file and click Open.  
The Motion Capture Conversion Parameters dialog displays (see *Motion Capture Conversion Parameters Dialog* (see page 192)).
5. Select the filter options you want and click OK.

The biped adapts itself to the motion data. If Footstep Extraction is turned on, footsteps appear.

**Tip:** Use a biped that does not have a mesh attached with Physique. Import motion capture data with the idea of then saving a *.bip* file that can be used for any character. If skeletal scale information is loaded from a motion capture file, a mesh with a Physique modifier may deform unnaturally.

### To import a marker file

Typically, when a marker file is loaded for the first time, it requires scale and position calibration. A raw marker file must be loaded, with no key reduction or footstep extraction, to enable the calibration functions. After calibration is performed, use Convert from Buffer to extract footsteps and reduce keys.

1. Select a biped.
2.  On the Motion Capture rollout, click Load Marker Name File to load a marker name file (*.mmm*).

This step is not required if the marker names in the marker file adhere to the **character studio** marker naming convention.

3. On the Marker Name File dialog, click Load CSM Marker File, and choose the *.mmm* file from the file open dialog that appears.



4.  On the Motion Capture rollout, click Load Motion Capture File and choose a *.csm* marker file.

The Motion Capture Conversion Parameters dialog displays (see *Motion Capture Conversion Parameters Dialog* (see page 192)).

5. Adjust the filter parameters and click OK.

Note: Load raw marker data (No Key Reduction, Freeform) to enable the marker calibration buttons.

The biped adapts itself to the marker data.

6.  Select Show Markers and Show Recognized Markers on all objects in the dialog. Now click Talent Figure Mode and use Non-Uniform Scale or Rubber Band Mode (on the General rollout) to size the biped to the displayed markers.  
Note: This step is optional and should be used if you need to correct for slight differences in limb scale between the original talent who performed the motion and the scale of the biped after the data is imported. For instance, scale the length of the leg in Talent Figure mode to adjust the knee position if the leg is too short.
7.  Click Talent Figure Mode again to exit the mode.
8. Key adaptation takes place when you exit Talent Figure mode. Now biped limb positions relative to the markers can be adjusted.
9. Align the biped limbs to the markers if necessary, then click Adjust Talent Pose to compute the offset for the entire animation.
10. Use Save Talent Figure Structure and Save Talent Pose Adjustment as a *.fig* and *.cal* file.
11. Load these files in the Motion Capture Conversion Parameters dialog when similar marker files are imported in the future.



At this point, you can use Convert from Buffer to extract footsteps and reduce keyframes. Both scale and position adjustments will be incorporated. Save the motion as an optimized *.bip* file.

## Using Motion Flow Mode to Combine Animations

Motion Flow Mode is used to animate one or more bipeds using a network of clips. The network of clips are joined together by transitions. This network is used to animate one biped or a crowd of bipeds. To animate one biped a single script is created that uses a list of clips to animate the biped. To animate a crowd of bipeds you can either use the random method of clip selection or a delegate driven approach. The random method simply picks clips at random and creates random scripts for each biped. This approach works well if the bipeds are standing still and don't require collision detection, a crowd at a ball game for example. The delegate driven approach also uses a network of clips but instead of random selection it bases clip selection on a delegates speed and heading. The delegate driven approach uses many parameters to simulate moving crowds and incorporates collision detection, surface follow and other parameters. For a detailed breakdown of delegate driven crowd behavior see *Crowd Animation* (see page 109).

Motion Flow Mode provides an area to arrange clips into a network and tools to create and edit transitions. Clip and transition percentages are also set using controls in Motion Flow mode. Clip and transition percentages are used by the Create Random Motion command during motion synthesis. In a delegate driven crowd simulation, clips are arranged to follow a logical sequence. For example, the first clip could be a start walk clip, then a walk loop, then a branch to a turn right and turn left clip, then a slow to stop clip and so on. During synthesis this arrangement is used to pick clips. If the software senses a collision ahead, the slow to stop clip is selected, or a veer to avoid clip is chosen.

## Transitions

Transitions link motion files (clips) together to create longer character animation and crowd simulations. Transitions can be created manually with the transition editor or automatically by the software. Optimized transitions use minimum foot sliding rather than the default minimum motion loss. To create optimized transitions use the Optimize Selected Transition on the Motion Flow Graph toolbar or Optimize Transition in the Transition Editor. Optimized transitions take time to calculate but yield high quality transitions.

Crowd simulations can potentially use dozens of motion clips so automatic creation of optimized transitions can be a big time saver. Whether you plan on animating one or many bipeds using Motion Flow mode you'll need transitions between the clips in the Motion Flow Graph.

## Random Motion

The Create Random Motion command uses clip and transition percentages to create random scripts to animate one or more bipeds. First you put bipeds into a Shared Motion Flow, and then you select clips and create transitions between the clips. The software then randomly traverses the clips to create random scripts for each biped.

## Unified Motion

The Create Unified Motion command allows you to create one motion from a script. The entire unified motion is then available when you exit Motion Flow Mode.

## Files and Directories

The location of the referenced *.bip* files is saved in the *.mfe* file. If a *.bip* file cannot be found, the program looks to the motion flow directory specified in *\plugcfg\biped.ini*. By default, this directory is *MoFlowDir=<maxdir>\cstudio\scripts*

If a referenced *.bip* file cannot be found in its current location, you will need to move it to the specified Motion Flow directory. You can change the location of this directory at any time by editing your *biped.ini* file with a text editor. The new directory will be used the next time you restart 3D Studio MAX. You can also add multiple search paths to your *biped.ini* file by repeating the *MoFloDir=* line multiple times. The program will search the directories in the order they appear and will use the first instance of the file that it finds. When network rendering the filenames need to be UNC compatible.

## See also

*Motion Flow Mode (see page 227)*

*Motion Flow rollout (see page 229)*

*Motion Flow Graph Dialog (see page 230)*

*Motion Flow Script rollout (see page 234)*

*Transition Editor (see page 237)*

*Save Segment dialog (see page 241)*

*Create Random Motion dialog (see page 241)*

---

## Getting Started with Physique



Use the Physique modifier to attach a skin to a skeleton structure such as a biped. The skin is a 3D Studio MAX object: it can be any deformable, vertex-based object such as a mesh, a patch, or a shape. When you animate the skeleton with skin attached, Physique deforms the skin to match the skeleton's movement.

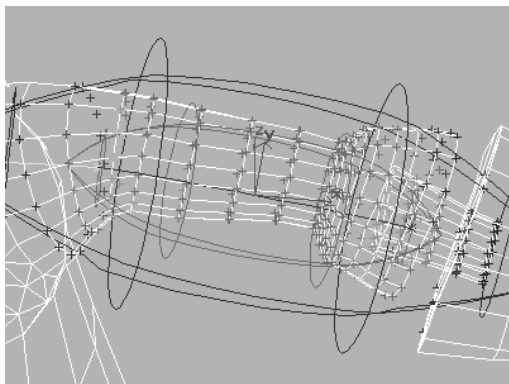
Animating the underlying skeleton enables you to animate a single contiguous model of a character that bends, creases, and bulges about an arbitrary number of joints within the attached skeleton.

With Physique, you can define how the skin behaves when it deforms. For example:

- You can make portions of the skin solid, excluding them from Physique's deformation, though solid portions still move along with the root node of the skeleton they are attached to. These solid portions are said to be root vertices.
- You can make portions of the skin deformable. They move with the deformation spline, the smooth curve running through the links of the skeleton they are attached to.
- You can make portions of the skin rigid, directly moving along with the skeleton they're attached to.
- You can add bulges to simulate bulging muscles. Bulges are controlled by editable cross sections of the skin, and by bulge angles that you set.
- You can add tendons to distribute the effect of one bone's motion to areas of the skin other than those around the bone itself.

Physique works with bipeds created and animated using the Biped plug-in, and with 3DS MAX hierarchies, including the Bones systems. Physique also works with bones that are not in a hierarchy and splines.

## Envelopes



**Envelopes**

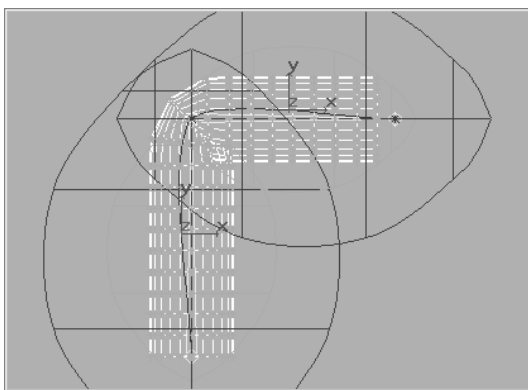
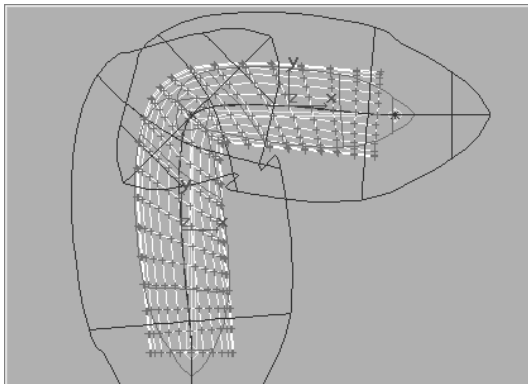
Envelopes are Physique's primary tool to control skin deformation. Envelopes define an area of influence about a single link in the hierarchy and can be set to overlap adjacent links. Vertices that fall in the overlap area of the envelopes are weighted to produce smooth blending at joint intersections. Each envelope comprises a pair of inner and outer bounds, each with four cross sections.

### Deformable and Rigid Envelopes

There are two envelope types: deformable and rigid.

- Deformable envelopes influence vertices they encompass to follow the deformation spline created through the hierarchy. Only the vertices encompassed by deformable envelopes can be affected by bulge angles or tendons.
- Vertices in a rigid envelope are linked to the node (the bone) and move in an immobile relationship to the link. Vertices in a rigid envelope, however, are deformed (blended) in the overlap area of other envelopes. There

is a twist parameter in Link Sub-Object that can be enabled in a rigid envelope. This allows the rigid envelope to twist along the length of the link.



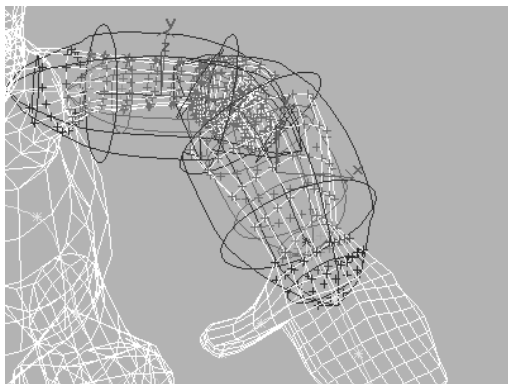
**Deformable envelopes (above); Rigid envelopes (below)**

Typically, you use deformable envelopes when you attach a mesh to the biped pelvis to produce a soft, flexible skin. Later you reassign certain links, such as the character's head, to the rigid envelope to minimize the deformation. For special cases, you can turn on both deformable and rigid envelopes for the same link. This advanced feature allows you to average the effect of the two types of skin deformation for



additional firmness in the skin. The forearms and legs are sometimes good candidates for this.

## Blending Between Links

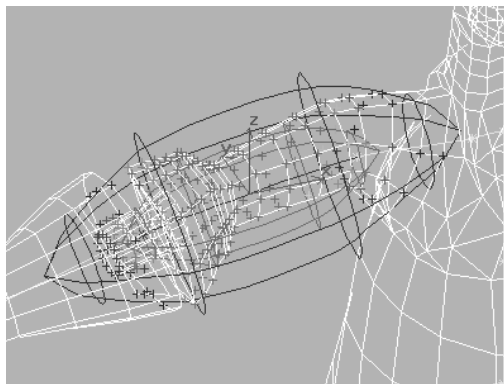


**Blending between envelopes controls deformation.**

Blending controls specify the influence of overlapping envelopes on vertices contained within the overlap area. By the actual shape of adjacent envelopes, you can control the degree of influence each has on blending at the overlap area.

You can further control the number of envelopes that participate in the blending effect, or whether no blending takes place at all. Where you specify no blending, a vertex in an overlap area is influenced by a single link only.

## Inner and Outer Bounds



### Inner and outer bounds

Where a vertex falls within the inner and outer bounds determines the percent of influence of the attached link(s). Vertices that fall within the inner bound have a weight of 1. Vertices that fall between the inner and outer bound have a weighted value that falls off to 0 at the outer bound.

Vertices are color coded in the viewports according to their weight; the color is based on parameters you specify.

### See also

*Envelopes (see page 300)*

## Bulges and Tendons

For some animations, simply attaching the skin and correcting its vertex assignments results in an animated skin you can use in final renderings. For other animations, you might need to give the skin more realistic movement, for example, muscles that bulge.

Physique lets you simulate an underlying musculature for the skin by adding tendons and bulges:

- Bulges change the skin's profile to simulate bulging muscles. You create the bulge by establishing bulge angles, relationships between cross-sectional slices of the skin and specific poses of the skeleton joint. Imagine a cross section to be a slice through the skin's mesh, perpendicular to the link. By making changes to cross sections, you in turn distort the shape of the mesh. Bulges in your character can be constructed by associating certain poses with related changes to the cross sections, in other words by defining bulge angles.

At any joint angle, you can define a bulge angle, and you may define as many bulge angles as needed. The bulge angle consists of the current orientation of the joint together with any defined cross sections. In addition, you can adjust the influence of a bulge angle. Physique considers all the bulge angles as the character moves. The resulting bulge is created by interpolating the effects of the various bulge angles having some influence at the current joint angle.

For example, to create a bulging biceps muscle, in Bulge sub-object level, on a selected link, insert a cross section near the center of the upper arm. Pose the arm into a

flexed position, with the angle between upper and lower arms at 90 degrees or less. Insert a bulge angle and adjust the cross section so that it distorts the mesh appropriately. In the viewports and in the Bulge Editor, you can edit the shape of the bulge to look like a flexed biceps muscle: higher and wider above the bone than below it. Now as the elbow bends from a straight orientation up and toward the shoulder, Physique bulges the biceps appropriately.

See *Creating Bulges* (see page 101) for more about how to create bulges.

- Tendons tie links together, extending the effect of moving one link to another link where the tendon is based. Their effect is similar to that of tendons in a body. For example, raising an arm usually stretches the skin along the same side of the body. To get this effect using Physique, you could base a tendon on a spine link, then attach it to the upper arm or collarbone, so when the arm lifts, the skin around the torso stretches.

Because tendons and bulges are optional, you can approach Physique animation in a couple of ways:

- Apply only as much detail as you need to get the effect you want for a particular scene or insert in your animation. This is probably the best approach when the Physique animation is meant to be used only once, or is not the main focus of the animation.
- Define a fully deformable character, with tendons and bulges for its entire range of motion. This is probably the best approach when you intend to reuse the Physique character in an ongoing series of

animations, for example, or in a video game that has a variety of character action.

## See also

*Bulge Sub-Object (see page 313)*

## Skeletons to Use with Physique

The skeleton to which you attach a skin using Physique can be a 3D Studio MAX hierarchy, bones in a hierarchy, bones not in a hierarchy and splines. Physique deforms the skin based on the relative position of the *bone* or links in the hierarchy. Specifically, it uses the length of each link and the angle between two connected links; it can also use the scale of a link.

The skeleton hierarchy can also be a 3DS MAX system object that defines a behavior as well as a hierarchy. Three kinds of objects that are especially useful with Physique:

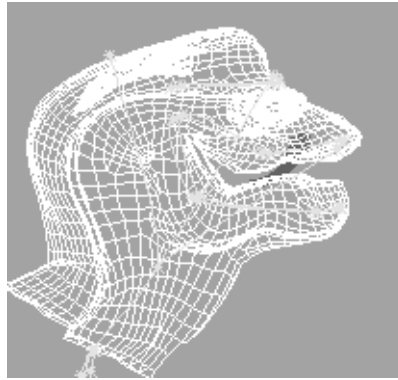
- Bipeds are provided by **character studio's** Biped plug-in.
- Bones are a standard Systems object provided with 3DS MAX.
- Splines can be used rather than a “bones” hierarchy.



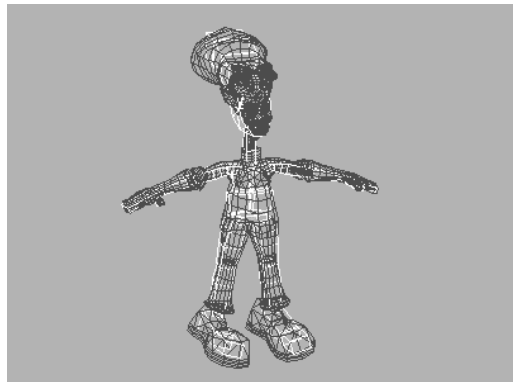
You create both bones and bipeds using the Systems object category in the Create panel.

Bones are useful for facial animation, a face with moving lips for example, or for non-bipedal characters. Bipeds are the system of choice for humanoid and other bipedal characters.

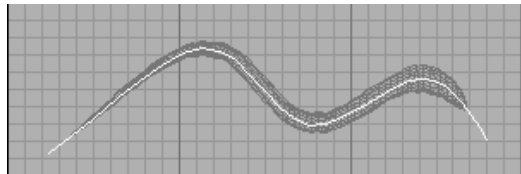
Usually you create the Physique skin before you create the skeleton, because you must adapt the skeleton's dimensions to the dimensions of the skin, in order to optimize vertex assignment to the links in the hierarchy.



This Physique hierarchy is created with dummy objects linked to each other



Physique mesh with a biped skeleton



Physique can use a spline to deform the mesh

## Creating a Skin

Physique is a modifier you apply to a skin, a deformable, point-based 3D Studio MAX object, where points are the vertices of a mesh or control points of other object types. A Physique skin can be:

- A parametric geometry primitive such as a cylinder.  
Geometric primitives are useful mainly for simple applications of Physique; for example, a cylinder with two bone links to depict an arm.
- A patch object.
- A spline or text shape.
- A NURBS object.
- A Free From Deformation (FFD) modifier.
- A mesh object you import from another application such as AutoCAD®
- An object with modifiers or a compound object.

Although you can apply Physique to an object with modifiers, this can affect performance. If you use other animated modifiers in combination with Physique, this reduction is unavoidable. If the other modifiers are not animated, you can collapse the stack to remove modifiers prior to Physique; this can significantly improve performance. An exception is the Optimize modifier, which can be useful for improving performance while you work with Physique.

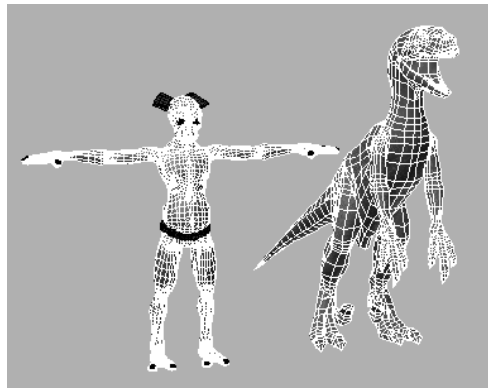
- An editable mesh object collapsed from either an object with modifiers or a compound object.

**Warning:** After you collapse a compound or modified object, you can no longer edit it parametrically. If you work extensively with complex meshes of this sort, it is probably

best to save two 3DS MAX *.max* files: one to contain the original, editable objects and modifiers, and the other to contain only the collapsed mesh.

Whichever way you create the skin, it must have a sufficient number of vertices so Physique can deform it smoothly. On the other hand, try to keep the skin as simple as feasible, because large numbers of faces and vertices use system resources. A highly complex mesh can slow the performance of Physique and 3DS MAX.

You can create a figure's skin out of several disconnected skins, for example, with separate objects for body and clothing. In this case, apply Physique to all the objects at once.



Meshes for different skeletons

## Creating a Skin for Bipeds

You create a biped skin the same way you create other meshes for use with Physique. When you create a skin to use with a biped figure, you should also position the arms and legs of the skin in a standard reference pose.



Mesh in reference pose for use with bipeds

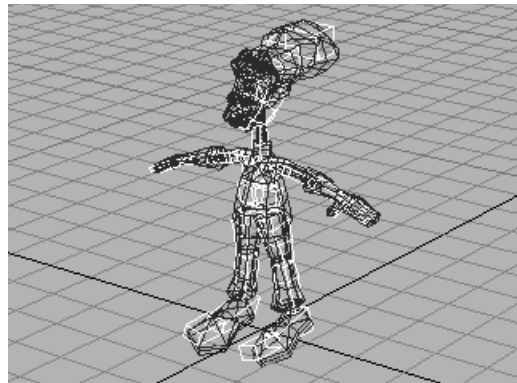
Use the following positions when you create the reference pose:

- Spread the legs somewhat apart, at a *parade rest* position.
- Spread the arms wide, level with shoulder height. The hands should be level with the arms, not dangling: palms facing down, fingers straight and slightly spread apart.
- Position the head so it will face in the correct direction when you load the biped's at-rest standing pose. If the skin and biped

are for a figure that stands erect, position the head normally. If the character stoops forward, for example, a chimpanzee, make the head face upward so that it will face forward after the spine is bent.

As a general rule, create a reference pose that has the limbs outstretched, but otherwise represents the character's natural at-rest posture.

## Adjusting Biped Arms Legs and Torso




Fit the biped to a mesh by first turning on Figure mode and then scaling the biped limbs to fit the mesh. Figure Mode is a reference position between the mesh and the biped.

**Tip:** Physique defaults to using the bounding box method to size envelopes about the biped links. Overall, you'll have best results using Physique by aiming for these results in the fitted biped:

- The biped joints match the skin's joint.
- The biped links are centered in the mesh.
- The bounding boxes closely fit the skin.

Before you adjust the biped's pose and proportions, make sure its center of mass is positioned at the crotch of the skin.

**Tip:**  Display the biped's skeleton before you proceed with skinning a character. Physique uses the pivot points for the various nodes that make up the skeleton hierarchy. The biped's renderable element objects give you a good 3D reference for positioning the figure inside of the mesh; however, it is the pivot points for these objects, displayed by the interconnected lines of the skeleton, that define the locations of joints for the Physique links.

### See also

*Figure Mode (see page 156)*

*General rollout (see page 147)*

---

## Using the Box Generator Utility for Non-Biped Skeletons

Before applying Physique to a mesh object about a non-Biped skeleton, such as a 3D Studio MAX bones system, use the Box Generator utility. Physique will use the boxes to determine the radial scale for its default envelopes.

Physique can create default envelopes based on either the link length or the size of the bounding box. While link length sometimes gives acceptable results, you'll generally need to revisit each envelope and adjust the radial scale to correctly encompass the mesh. Using bounding box to establish the default radial scale means that Physique envelopes are more likely to fit the skin without much additional work.

### To use the Box Generator utility

1. On the Utilities panel, click More, and select Box Generator.
2. Enter the name of the boxes in the Box Name field, or accept the default name.

3. Set the Width parameters of your boxes so they roughly fit your mesh. You have the option of either basing the box width on a percentage of the link length, or using an absolute value where all the boxes have the same width.
4. Click Pick Root Bone.
5. Pick the root bone of your hierarchy: either click it in the viewport, or click Select by Name and choose it on the Select Objects dialog.

The utility generates boxes about each link in the hierarchy.

6. Non-uniformly scale the box width and length to snugly fit your mesh.

Note: You may get a 3DS MAX Non-Uniform Scale warning at this point. Ignore the warning.

You can also adjust the object parameters in the Modify panel to fit each box in the mesh.

Now, when you attach the Physique mesh to the bones hierarchy, you can use the Object Bounding Box option for generating default cross sections, and your Physique mesh should deform well using default settings. See *Initializing Physique (see page 96)*.

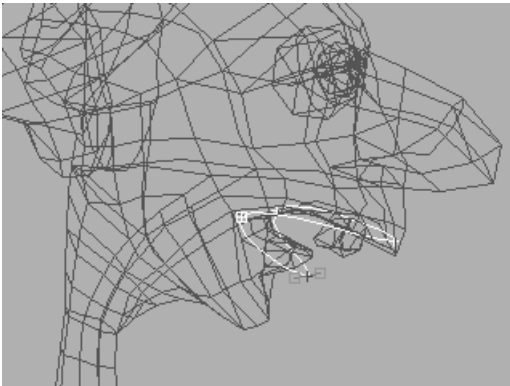
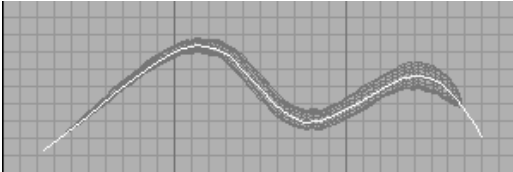
**Tip:** Once Physique has been initialized and you're satisfied with the vertex linking results, you may want to select these box objects (this is where the name you entered in step 2, above, can be handy) and hide them in the scene. That will reduce some of the screen clutter that can happen, especially with complicated meshes.

## 👁 Spline-Based Physique Deformation

In addition to the biped and 3DS MAX bones, Physique now supports spline and NURBS curves for mesh deformation. By animating vertices on a spline you can animate the mesh. Use this for facial animation or to deform any mesh.

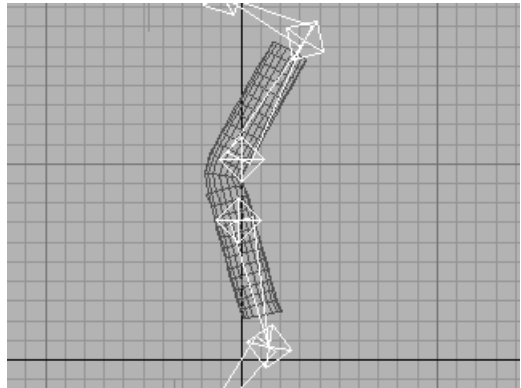
### To control a mesh with a spline

1. Place a spline inside a mesh.
2. Select the mesh and add the Physique modifier.
3. In the Physique Floating Bones rollout click Add and select the spline in the viewports.
4. Select the spline, turn on Vertex Sub-Object and move the spline vertices to animate the mesh.



## 👁 Floating Bones

Floating bones are bones that are not linked together and know nothing about each other. By adding floating bones to Physique you can deform the mesh by animating the bones. This is in contrast to using Attach to Node and clicking on the root of a hierarchy, like the biped pelvis. For Attach to Node to work all the bones should be linked together.



## Applying Physique

After you have created a skin and a skeleton, and fitted the skeleton to the skin, you apply the Physique modifier to the skin. The process entails these steps:

- Selecting the mesh
- Turning on Figure Mode (if a biped is used)
- Adding the Physique modifier to the stack
- Attaching the Physique skin to the skeleton
- Initializing Physique

**Note:** Before applying Physique to a skin with a 3D Studio Bones system underlying hierarchy, run the Box Generator utility. This simplifies default envelope sizing when you apply

Physique. See *Using the Box Generator Utility for Non-Biped Skeletons* (see page 94)

## See also

*Physique rollout* (see page 296)

## Initializing Physique

When you use Attach to Node to attach a Physique skin to a hierarchy, the Physique Initialization dialog appears.

Note: This dialog also appears when you want to reset Physique settings by clicking Reinitialize in the Physique rollout.

Physique initialization settings affect how envelopes are created and blending is handled. The Link Settings, Joint Intersections, and Cross Sections rollouts are used later to change default settings globally. For this reason, the Vertex-Link rollout is open when the dialog appears. This is where you determine the following default settings:

- Whether envelopes should be used to manage the vertex-to-link assignments
- Type of envelopes Physique uses to manage the vertex-to-link assignments: deformable or rigid
- Number of links considered for blending

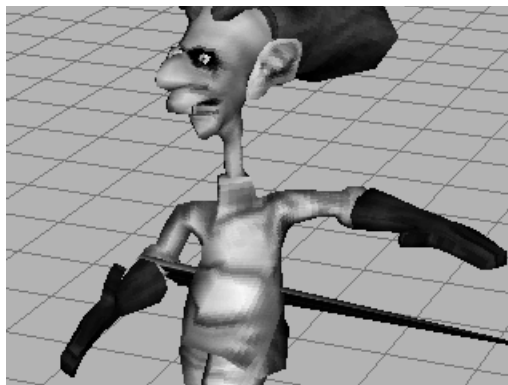
## See also

*Physique Initialization dialog* (see page 335)

## Previewing Motion

After you have attached the skin to the skeleton, Physique deforms the skin when the skeleton is animated. From this point on, you can preview animation to see how the Physique skin deforms and whether you can use it in a finished

animation or need to correct and refine it further.



### A stray vertex not encompassed by an envelope

Very likely, you'll need to adjust some of the default envelope settings to ensure all vertices are being properly handled. It's by moving the skinned character in the viewports that you'll see vertex assignment problems.

See the *General rollout* (see page 147) for procedures and interface.

**Tip:** Physique skins are usually too large for 3DS MAX to play them back in real time at 30 frames per second. However, speed improvements in **character studio 3** will allow fluid motions of many skinned meshes. When you attempt to play back at 30 fps, 3DS MAX may drop frames during playback, which makes it hard to see how Physique has animated the skin. Use the Time Configuration dialog to turn off real time playback. The animation plays back more slowly than usual, but it plays every frame.

**Tip:** You can also specify which elements of the characters not to display in the viewports, and thereby speed up redraw. See *Level of Detail Controls* (see page 298).



For a skin attached to a complicated skeleton such as a biped, you will almost always need to correct some vertex assignments before the skin animates correctly. Most often, you'll do so simply by adjusting the envelope's shape. See *Adjusting Default Envelope Shape* (see page 97).

## Adjusting Default Envelope Shape

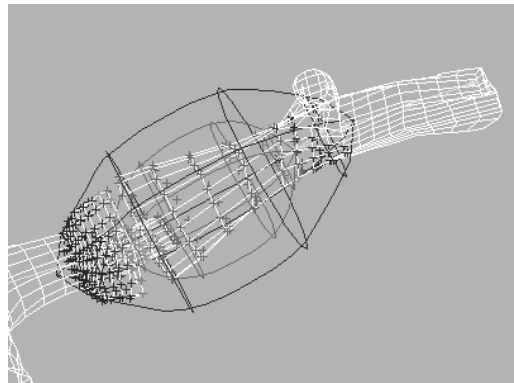
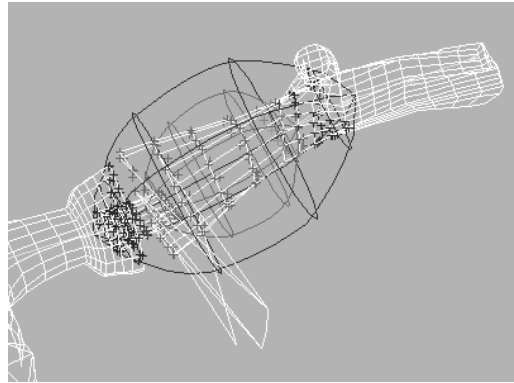
When you encounter and want to address vertex linking anomalies, the first step is to resize or reposition the envelope about the problem link. Most often, the culprit is an envelope having been created too small to surround, and thereby influence, certain vertices.

This can happen because your biped character misaligns slightly with the mesh, or the link lengths used to create envelopes with unbounded Bones systems links were insufficient to surround all vertices attached to the link.

Another potential source of problems is overlapping inner bounds. This can sometimes create too strong a deformation at the joint.

You'll note problems with vertices as you preview motion. It sometimes looks like vertices got left behind when the link they were supposed to be attached to moves in 3D space. In fact, that's just what is happening: they weren't assigned to any link at all, and remain where they were at the initial skeletal pose: the pose of the mesh and its skeleton at the time Physique was applied and initialized.

The other case would be vertices being moved too much, resulting in a dent in the mesh. This would be fixed by decreasing the overlap of the envelopes affecting a joint.



**Stray vertices (above) reassigned to correct link(s) by modifying envelope shape (below)**

See also

*Envelope Sub-Object* (see page 300)

## Fine-Tuning Envelopes

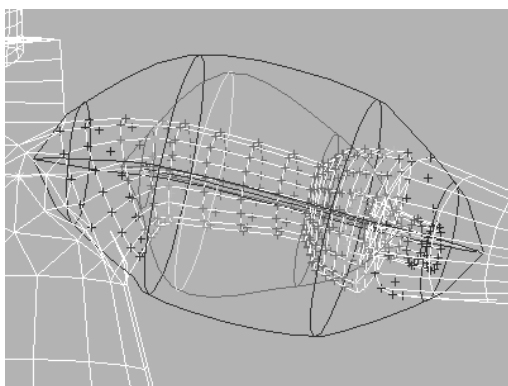
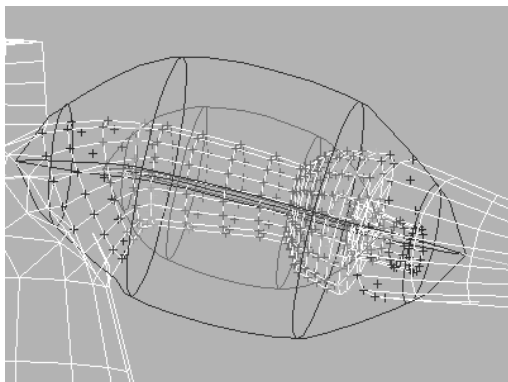
Once you have adjusted inner and outer bounds of envelopes at the Envelope Sub-Object level, you may find you need still finer control: choose either the Cross Section or Control Point selection level.

See also

*Envelope Sub-Object* (see page 300)

## Using Cross Sections

Both inner and outer envelope bounds are like *hulls* about *ribs*, the cross sections. For each envelope, there are by default four cross sections you can use to alter the envelope's shape.



Envelope cross sections can be scaled and moved

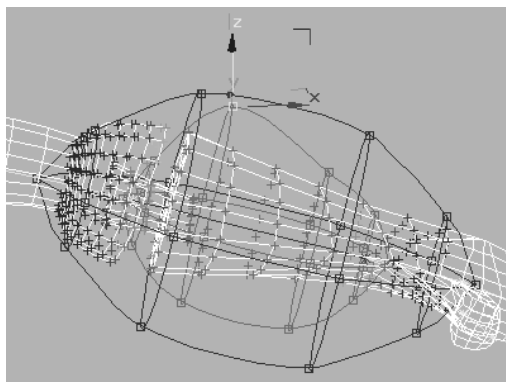
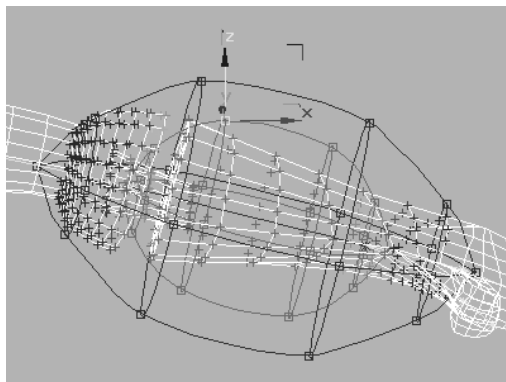
## Using Control Points

About each cross section are a default of four control points. You can use these to alter the shape of a single cross section. The control points define the shape (perimeter) of the section. Moving one point affects the shape between itself and its neighbors on each side.

Selecting a control point, limiting or extending the selection is the same as with cross sections, described above.

## Rotating and Scaling

Control points are internally positioned using cylindrical coordinates, that is a distance from the link at some angle around the link. To move a control point, you must scale it to the correct distance from the link and then rotate it into the orientation you desire. The move tool actually serves the dual function of rotating and scaling at the same time.



By moving a control point the envelope is scaled also

## Interactive Redraw

By default Interactive Redraw, an option in envelope and bulge sub-object, is turned off. After changing an envelope the mesh is refreshed on mouse up. Turn this option on for interactive viewing of how envelopes and bulges are affecting vertices. As you adjust the envelopes the vertices move to show the effects of the resized envelopes

## Changing Display Options

You can change the display settings for envelopes and their component parts. Click the associated color sample to open the Color Selector and change color setting for:

- Inner and outer bounds on both deformable and rigid envelope types.
- Differently weighted vertices, as they fall within the inner bound, or between it and the outer bound.
- Selected cross sections and control points.
- Select or clear the associated check box to determine whether any of the above envelope components are displayed.
- Number of sides for inner and outer bounds. The default is 4.

Select Initial Skeletal Pose if you want to work in the position of the mesh and its skeleton at the time Physique was applied. Returning to this pose can be useful if the display becomes confused with the effects of unassigned vertices when the character moves through an animation.

## Fine-Tuning the Skin with Physique

Fine tuning involves working with Bulge and Tendons sub-object types to add further expressiveness to your character's skin, and working with Vertex sub-object controls to assign individual vertices to links.

See also

*Link Sub-Object (see page 307)*

*Bulge Sub-Object (see page 313)*

*Tendon Sub-Object (see page 326)*

## Working with Deformable Envelopes

Once you've got the deformable envelopes working the way you want to control overall skin deformation, you may want to turn to adjusting finer aspects of skin control:

- Skin bending, twisting, sliding, and scaling about single links are controlled at the Link sub-object level. See *Adjusting Link Parameters (see page 100)*.
- Link sub-object controls primarily the deformation spline. You use Link sub-object options to affect the shape of the spline and the smoothness of the skin. See *Adjusting Link Parameters (see page 100)*.
- Crease behavior where links meet and bend is controlled at the Link sub-object level with Joint Intersection parameters.
- Muscle bulges as your character's limbs move through a range of motion are controlled at the Bulge sub-object level. See *Creating Bulges (see page 101)*.
- Skin stretching, generally between non-contiguous links, is controlled at the

Tendon sub-object level. Tendons provide a secondary movement (a pulling or stretching) based on links farther up or down the skeleton. See *Creating Tendons* (see page 103).

### See also

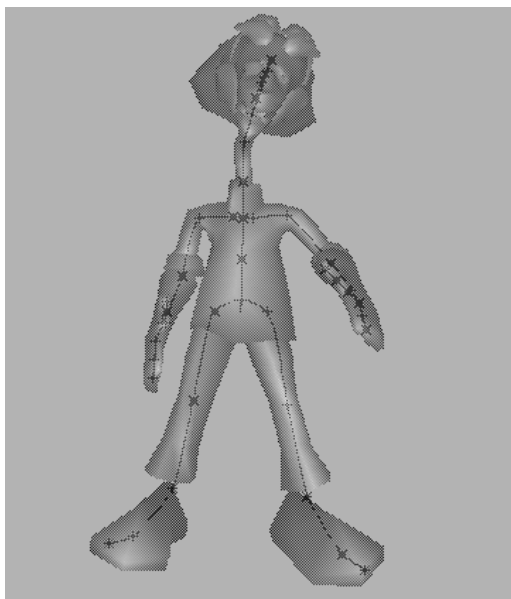
*Link Sub-Object* (see page 307)

*Bulge Sub-Object* (see page 313)

*Tendon Sub-Object* (see page 326)

---

## Adjusting Link Parameters



Physique creates a continuous spline through each of the joints in the hierarchy. The spline is represented by an orange line within the object. It maintains continuity through each joint: as a joint angle changes, the spline passing through the joint remains a smooth curve. Physique relies on the deformation spline to obtain smooth bends of the skin object. Physique offers

bias and tension controls to adjust the shape of the deformation spline.

You can use controls at the Link sub-object level to adjust the skin behavior about a given link. These link parameters define how the skin deformation behaves, relative to the motion of the underlying skeleton.

There are four kinds of link parameters:

- Bend parameters affect the curvature of the deformation spline through the joint at a given link. As a result the skin deformation can range from more angular at the joint to more like a stiff rubber tube.
- Twist parameters control how the skin deforms when a joint rotates along its length, as in turning a doorknob. Consider, for example, winding up the rubber band on a toy airplane propeller; Twist parameters determine how the twist is distributed along the link and across the joint.
- Sliding parameters control how the skin moves along the length of a link as a joint rotates. On the outside, the side where the angle is greater than 180 degrees, the skin moves toward the joint. On the inside, the side where the joint angle is less than 180 degrees, the skin moves away from the joint.

The sliding effect tightens the outside to keep detail at the joint and prevent the facets of the mesh from moving apart. It relaxes the inside to prevent the mesh from bunching up at the joint.

- Radial Scale settings affect the influence on the skin caused by scaling links of the skeleton. Certain settings, such as Breathe and Stretch, react to scaling of the skeleton,

but only with standard 3DS MAX bones or a 3DS MAX hierarchy.

Note: Bipeds and 3DS MAX IK controllers don't support scaling.

Link Scale provides a way to globally scale all the vertices influenced by the link. CS Amplitude scales the effect of bulge cross sections across the entire link. It can be used for animated scaling of the bones or for general mesh adjustments. The intent is not to edit the character itself, but rather the effect the bones have on it.

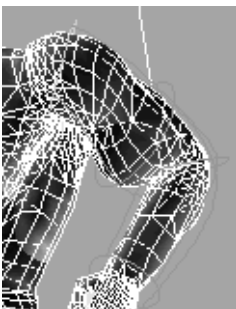
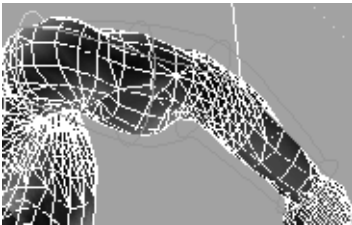
You set Link parameters at the sub-object level, in the Physique Link Settings rollout.

See also

*Link Sub-Object (see page 307)*

---

## Creating Bulges



**Bulge shape is interpolated as joint movement approaches a bulge angle.**

Bulges simulate bulging muscles. Physique creates bulges based on bulge angles and cross section shapes you specify, not on keyframe settings. You create a bulge by:

- Reposing the character to a position where the bulge will have its greatest effect. This can just be a matter of using the time slider to scrub to that place in a loaded motion file.
- Setting a bulge angle between two links, the currently selected link and its child link in the hierarchy. The bulge angle is the angle of the joint where the bulge has its full effect. When the joint has a different angle, Physique interpolates so the bulge can grow as the joint flexes toward that angle. See *Setting Bulge Angles (see page 102)* for more information.

Note: The resulting bulge for any given frame in an animation is determined by the interpolated effects of all bulge angles for the link, based on the relationship of each bulge angle to the current joint angle. Bulge angles are not directly associated with keyframe parameters, but are relative to the skeleton's behavior.

- Creating and shaping the cross sections associated with the bulge angle. A link's cross sections and its profile are spline controls of the shape of the skin. To create and shape cross sections, see *Shaping the Bulge (see page 102)*.

Each bulge angle affects both neighboring links. Therefore, each link contains a set of cross sections for each bulge at both its parent and child joint angles. For example, the forearm link can be deformed by bulge angles associated with both the elbow and wrist joints.

- Adjusting bulge parameters, including the joint intersection parameters. Bulge parameters control the smoothness and the strength of the bulge and are found at the Bulge sub-object level.

Joint intersection parameters control how the skin behaves when bulges would overlap each other if there were no collision detection for skin vertices. They are in a Physique rollout for Link sub-objects. See *Reinitializing Physique Settings* (see page 105).

### See also

*Bulge Sub-Object* (see page 313)

---

## Setting Bulge Angles

A bulge angle associates an angle value and a name. By default, each link has one bulge angle whose default name is the name of the link followed by “Bulge 0”. The default bulge angle’s initial angle value is the angle between the link and its child when you first attach Physique to the skeleton.

Bulges are effective because they grow and shrink as the joint moves. The initial bulge angle defines one shape that the skin can deform to; this would normally be like a default musculature. With no other bulge angles defined, the skin would always look like the first bulge angle, regardless of pose. Additional bulge angles provide the other shapes the mesh can deform to.

In the simple case of a flexing biceps muscle, one bulge angle defines the relaxed position and another defines the muscle in its flexed pose. Both are exposed so you can add definition to the default character. As in a biceps muscle, a heavily-muscled character might have some shape even in a relaxed pose. This lets you

change the baseline without disturbing the original mesh. You can actually use Physique bulge cross sections to model your character.

The cross section deformation for a given link is determined by interpolating between the contributions of all bulge angles that affect the link. This includes bulge angles for both the link’s parent joint angle and its own (child) joint angle. Each bulge angle’s contribution is determined by:

- Influence (how far away the bulge angle is from the current joint angle rotation)
- Power (an ease based on the influence of the angle)
- Weight (the relative strength of the bulge angle)

Keep in mind that bulge angles can be set for arbitrary rotations, and are not limited to single-axis hinge joints.

### See also

*Bulge Sub-Object* (see page 313)

---

## Shaping the Bulge

Once you’ve inserted the bulge angle, and recorded the angle setting, you then shape the bulge by inserting and adjusting cross sections. You can either move and scale the cross sections, or adjust each one’s shape using control points.

### See also

*Bulge Editor* (see page 318)

---

## Fine-Tuning in the Bulge Editor

The Bulge Editor duplicates many of the controls available at the Bulge sub-object level. It gives you a focused, two-dimensional view of the current bulge settings.

See also

*Bulge Editor (see page 318)*

---

## Adding More Poses to Your Character

For more complex body movements, you may need to add bulges for different positions. Overall, you might want to build a generic character that works in a number of situations, regardless of the type of motion. Some examples are:

- Moving the character into several extreme poses.
- Putting the arms high over the head and bending them into different positions.
- Making the character squat and lifting the legs into various positions.
- Adding bulge angles to the links at places where real muscles would bulge.

Once the character's skin reacts well to the full range of motions, you can then put it into your less demanding animation and it should work perfectly.

---

## Creating Tendons

Like tendons in an actual body, tendons in Physique link one bone to another. They spread the effect of moving one link to the skin around a different link. Tendons can improve the realism of skin movement when it is animated by Physique.

Basically, a tendon consists of base points that live on a cross section near the skin. A base point is attached to another link on the skeleton that pulls it. As the tendon base point is pulled, it deforms the skin around it. Several parameters control how much the point is pulled in each direction.

You insert the tendon cross section at the location where you want the skin to stretch. You then attach it to the link that influences the movement.

Each link can contain several tendon cross sections, and each control point may be attached to a separate link. In a practical application, however, such as in the area below the armpits, you might have two control points attached to each of the left and right clavicles.

See also

*Tendons Sub-Object (see page 326)*

---

## Working with Rigid Envelopes

Vertices influenced by a rigid envelope follow the link itself rather than the deformation spline, like deformable vertices do. Low vertex-count models used for games are typically connected to the skeletons with rigid envelopes to simplify export to game engines. In such a case, each vertex can be simply described as assigned to a specific link. It can be located by giving a length along the link, a distance from the center of the link, and an angle around the link. As the link is transformed, all the vertices assigned to the vertex are uniformly transformed by the same amount. This makes it simple for game engines to move the character's skin.

The vertices move equally with the controlling link, so the skin appears rigid. For instance, in

the case of a flexing arm, the skin between the wrist and elbow tightly follows the movement of the forearm, with no change in the spacing between vertices. The skin in between the elbow and shoulder tightly follows the upper arm. Vertices around the elbow influenced by both links, smoothly blend to an average position between the two changed links.

Rigid and deformable envelopes can be mixed on a given character. You might, for instance, use deformable envelopes for most of the character, but choose rigid for the head and for a suit of armor around his torso.

Note that, in Link sub-object, you can turn on twisting for rigid envelopes. As the wrist twists the vertices on the forearm will twist also.

### See also

*Envelope Sub-Object (see page 300)*

---

## Working with Both Deformable and Rigid Envelopes

You can assign both Deformable and Rigid to a given link. For example, you might assign both to the shins, and size the envelopes differently to gain the benefits of both.

First size the deformable envelope to affect the whole lower leg area and smoothly transition into the thigh and ankle areas. Next, you define a smaller rigid envelope to encompass the stiff unbendable section of the shin. Increase the weight of the rigid envelope to balance its effect against that of the deformable envelope.

Some users might use both envelopes as a way to have a bone affect two completely separate regions of a mesh. The envelopes are scaled so that the deformable influences certain vertices on one side of the link and the rigid affects the other side.

### See also

*Envelope Sub-Object (see page 300)*

---

## Using Vertex Sub-Objects to Override Envelopes and Blending

Use Vertex sub-object controls to:

- Override vertex-link assignments currently in effect.
- Change the blending type.
- Change the blending weight.
- Assign vertices to their links yourself, to bypass Physique's envelope method for vertex-link assignment.

You might want to reassign link assignments in effect about adjacent fingers, for example, even after working to reshape envelope size and position. You may find that you need greater control in areas where adjacent envelopes overlap: perhaps movement in the ring finger is causing unwanted movement in some vertices for the middle finger. You'd use Vertex sub-object controls to remove ring-finger vertices from the middle finger links, in effect reassigning them solely to the ring-finger links.

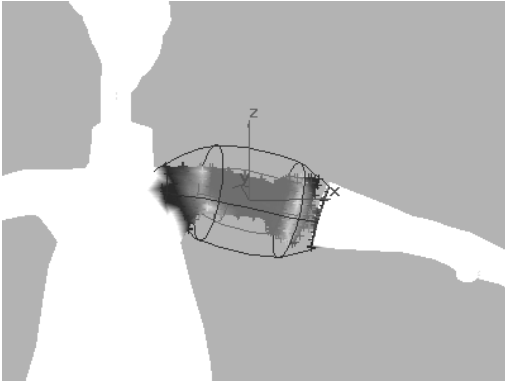
When you choose Vertex as the sub-object to edit in the Physique modifier, the Physique Selection Status and Vertex-Link Assignment rollouts appear.

### See also

*Vertex Sub-Object (see page 331)*



## Shaded Display of Vertex Weight Values



The Shaded option in Envelope Sub-Object allows you to display vertex weights in a shaded display. This feature can be helpful during envelope editing.

See also

*Envelope Sub-Object (see page 300)*

## Reinitializing Physique Settings

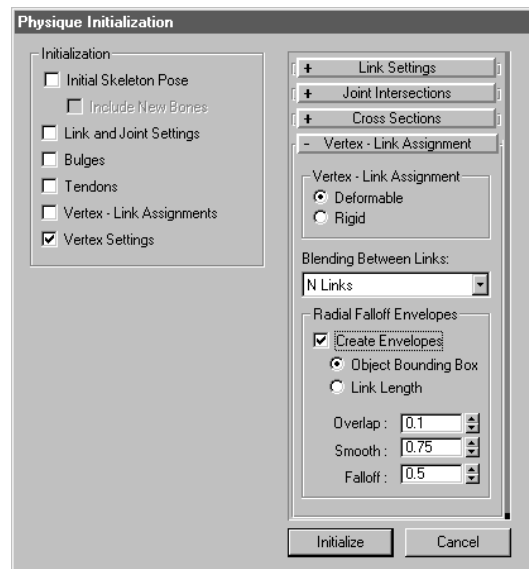
When you need to reset vertex, envelope and other skin parameters on a Physique mesh, click Reinitialize to display the Physique Initialization dialog. Using controls in this dialog, select the category to update, and apply the new global settings.

For example, if you've added a new bone to the hierarchy and want it included and influenced by the Physique modifier, use the reinitialization mechanism to effect its inclusion. Or maybe you've repositioned the biped structure relative to the mesh, you'd need to reinitialize Physique settings to recognize those changes.

## Reinitializing Old Files

Files from previous versions of the software go through a semi-automatic physique reinitialization process. When you load in an old file that uses physique you're prompted to put the biped or bones system into Figure mode or in the initial position and reinitialize. To reinitialize, select the mesh, open the Modify panel and click Reinitialize. A warning displays to put the skeleton into the initial position. Click OK to reinitialize the file.

Reinitializing with changed settings applies them as new defaults in the areas you choose. Reinitializing without changing settings on the Physique Initialization dialog is a method of *erasing* unsatisfactory changes you've made to the Physique settings, again in those areas you specify.



Reinitialization mode of Initialization dialog

## Initialization Area

The Initialization area in the Physique Initialization dialog is the area of focus when you reinitialize a Physique mesh. It's here you name the category of settings to reset. (When you applied Physique to the mesh initially, these settings appeared unavailable; in fact, all settings are set at Initialization.)

**Warning:** Only select the check boxes for those settings you need to reset. If you select everything, your model is returned to the state it was in immediately after applying Physique. That's a quick way to discard changes that aren't working; just make sure that's what you want to do. If you change settings in rollouts on this dialog and fail to select the corresponding check box, those rollout settings are ignored and initial defaults are reestablished.

See *Reinitialize Physique* (see page 338) for procedures and interface.

---

## Working with an Initial Pose

At times, you'll need to alter the fit of the hierarchy in the Physique mesh, or change its structure. To do so, you change the default, initial pose, which Physique uses as a reference for various operations, including reinitializing.

- For a biped, you change the structure in Figure mode, then reinitialize the Initial Skeleton Pose.
- For a Bones system with the IK Controller, there is a "Show Initial State" check box in the motion panel, IK Controller Parameters rollout.
- For any other 3DS MAX hierarchy, frame 0 is considered the initial pose. Be sure your animation does not affect the position of the skeleton on frame 0 in this case.

## To load a file created in different system units

Note: If the System Unit Scale field in Customize > Preferences, General tab, System Unit Scale has a value that is different from that of the file you are loading, 3DS MAX prompts you to rescale the scene. If you do so, any objects with a Physique modifier exhibit a double scaling. Do the following:

1. Select the object.
2. Reinitialize with both Initial Skeletal Pose and Vertex Settings (the last check box) selected.

## See also

*Figure Mode* (see page 156)

---

## Improving Interactive Performance

### Level of Detail Controls

Physique is now multithreaded and optimized for modifier stack changes below the Physique modifier. This makes for greatly improved performance.

The controls in the Physique Level of Detail rollout help you optimize performance while working with Physique, by letting you specify which skin deformations are refreshed automatically in viewports. The more complex the skin object, the more effective these controls can be at speeding up your work.

Note: The 3DS MAX renderer also heeds Level of Detail settings. Don't forget to reset these controls before the final render.

## See also

*Physique Level of Detail rollout* (see page 298)

## Turning Blending On and Off

When Initializing, Blending Between Links is set to N-Links. This means that every envelope must be considered when determining the influence on any vertex. You can reinitialize with No Blending (or 2,3, or 4 links) to reduce this calculation permanently, although at the cost of losing some or all blending at the joints.

Note: Setting to No Blending is only advised for those developing characters to be used by real-time game engines.

You can also set No Blending at the Vertex sub-object level. You can temporarily disable Link Blending on the Physique Level of Detail rollout, on the Modify panel.

## Using the Optimize Modifier with Physique

The standard 3DS MAX Optimize modifier allows you to reduce the level of mesh detail in order to increase display performance and speed up screen refreshes. Optimize can be helpful with Physique when you work with complicated skin objects such as detailed biped figures.

**Warning:** Any modifier in the stack below Physique causes Physique to reevaluate vertex assignments each time the stack is changed. While this technique will improve performance in one way, it will hurt performance if you need repeatedly to make changes down in the stack. There are stack update options that determine how the stack is updated. See the *Physique Level of Detail* (see page 298) rollout for a reference on the stack update options.

### To optimize skin objects

1. Select the skin.  
If the skin consists of multiple objects, select all of its component objects.

2. On the Modify panel, make sure that the skin object appears in the Modifier Stack dropdown.

If the skin consists of multiple objects, Multiple Selected appears at the top of the panel and no name is visible in the Modifier Stack dropdown.

3. Click More in the Modifiers rollout.

4. Choose Optimize from the list in the Modifiers dialog, and then click OK.

The Optimize modifier's Parameters rollout appears in the command panel.

5. In the Optimize area of the Parameters rollout, increase the value of Face Threshold.

As you increase Face Threshold, Optimize reduces the number of faces in the mesh.

The optimized mesh displays more quickly and speeds up your work with Physique. However, it doesn't show full detail, and sometimes an optimized skin does not animate correctly, because of the reduced number of vertices.

**Tip:** Always turn off Optimize before you render the animated skin.

---

## Combining Physique with Other Modifiers and Stack Update Options

Although you can apply Physique to an object with modifiers, this can affect performance. If you use other animated modifiers in combination with Physique, this reduction is unavoidable. If the other modifiers are not animated, you can collapse the stack to remove modifiers prior to Physique. This can significantly improve performance.

How Physique reevaluates vertices coming up the stack depends on the stack update options.

## Stack Update Options

In Physique there are three stack update options to determine how Physique handles animated vertices coming up the modifier stack.

### Add Change

Add Change adds in the changes from the stack and then applies physique deformation. No vertex remapping or reassigning is done. This is the default option and generally will give you the deformation that you want.

### Remap Local

Remap Local resets vertex position on the spline used for bending and the link position used to interpolate twist. Use this option when vertices are sliding along the length of the spline and you want them to bend and twist based on the spline position but don't want the weights to change.

### Reassign Global

Reassign Global does a complete vertex reassign for each frame. Use this option when vertices are moving to different envelopes and you want them reassigned to the new envelopes. This option should rarely be used. The only reason would be if vertices are sliding along a link and you want the twist to be interpolated based on the new position. This is how stack updates were handled in **character studio v2**.

### See also

*Physique Level of Detail Rollout (see page 298)*

---

## Using Physique with Changing Geometry

Physique can effectively handle dynamically changing geometry. That is, if the geometry coming through the modifier stack to Physique changes, Physique dynamically adjusts to accommodate the changes to the geometry, while maintaining its own parameter settings, including manual vertex assignments. This is why, when using Optimize, you can effectively change the resolution of the geometry deformed by Physique.

Only in situations where the changes dramatically increase the density of data is there ever a need to manually adjust vertex assignments. Physique remembers your manual vertex assignments, and uses these assignments to reassign vertices when the geometry changes.

### See also

*Physique Level of Detail rollout (see page 298)*

---

## Saving and Loading Physique Data

You can save Physique data to a Physique (.phy) file to save data common to all objects sharing a given Physique modifier.

Later, you can reload the data file, either to restore the data that belongs to a particular skin or portion of skin, or to transfer the Physique of one skin (or portion) to a different one.

Note: **characters studio 3**, no longer saves or loads the Vertex Assign (.vph) file format due to the dramatically improved vertex assignment method of envelopes and weights. Game developers needing access to this vertex data should use the Physique SDK included with the software.

See also

*Physique Rollout (see page 296)*

*Physique Load Specification Dialog (see page 293)*

---

## Crowd Animation



**character studio 3** has a new crowd animation system. A combination of new helper objects (delegates and crowds) form the basis of this system. The animation capabilities for crowds and delegates includes the use of behaviors and states. Behaviors include Avoid, Seek, Wander and Speed Vary. Behaviors can be scripted, and you can convert certain behaviors into scripts.

A crowd of delegates can drive the animation of multiple bipeds and other animated objects or characters within the scene. Individual delegates can make use of cognitive controllers to tell them how to behave in varying circumstances.

These topics provide further explanation of the principles of crowd animation, and contain links to additional topics with details on using the crowd system:

*Crowd Object (see page 110)*

*Crowd Delegates (see page 111)*

*Creating a Crowd System (see page 111)*

*Crowd Behaviors Overview (see page 112)*

*Understanding Behaviors (see page 113)*

*Obstacle-Avoidance Behavior (see page 115)*

*Attaching an Object Instance to a Crowd System (see page 116)*

*Using Bipeds with Crowd Delegates (see page 116)*

*Clip Controllers (see page 119)*

## Crowd Object


The Crowd helper object serves as the command center for setting up and solving crowd simulations in **character studio**. You use this object to clone delegates and distribute the clones, add behaviors, apply behaviors to delegates, link delegates to animated objects, and much more. Incidentally, you'll rarely, if ever, need more than one Crowd helper object per scene.

The Crowd object provides a number of rollouts, but the most important is the Setup rollout.






### The Setup Rollout

You use the Setup rollout controls to clone delegates and other objects, add and set up behaviors, assign objects and bipeds to delegates, assign behaviors to delegates, and create cognitive controllers.

The Setup rollout includes six buttons that open dialogs for various aspects of crowd control:

-  Use the *Scatter Objects dialog* (see page 362) for cloning delegates and other objects and distributing the clones over a surface or within a volume. Scatter Objects also lets you orient clones toward a specific object or along a line between two objects, with optional random variation. You can also specify scaling on any or all axes, again with optional randomization. The All Ops feature lets you apply randomization repeatedly to any or all of Scatter Objects

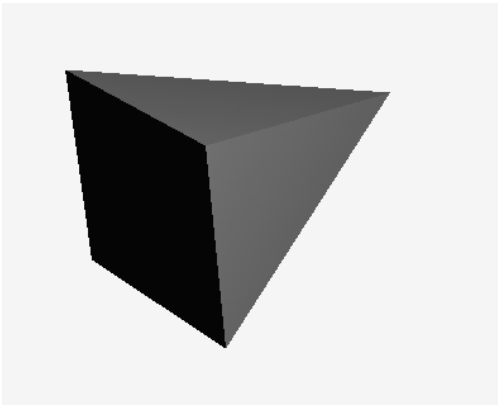
functions until you get a configuration you like.

-  Use the *Object/Delegate Associations dialog* (see page 370) to link any number of delegates with an equal number of non-biped objects. You can also use this dialog to align objects with delegates, optionally matching scaling factors as well. Lastly, you can specify that the linked objects use the delegates' controllers.
-  Use the *Associate Bipeds With Delegates dialog* (see page 374) to link any number of delegates with an equal number of biped objects. You can make the associations in listed order, or simply use proximity in the scene.
-  Use the *Edit Multiple Delegates dialog* (see page 372) to set parameters for groups of delegates, with optional randomization within a range you specify. This dialog also lets you adjust delegate settings to match an animated object.
-  Use the *Behavior Assignments and Teams dialog* (see page 375) to group delegates into teams, and assign *behaviors* (see page 361) and *cognitive controllers* (see page 382) to individual delegates and teams
-  Use *cognitive controllers* (see page 382) to sequence different behaviors using state diagrams, where conditionals written in MAXScript impose changes in behavior.

### See also

*Crowd Helper Object (see page 346)*

## Crowd Delegates



**character studio's** Crowd system uses Delegate helper objects as intermediaries between the crowd simulation and the influenced objects. A delegate is a non-rendering pyramid-shaped object whose apex points in the forward direction. A scene should contain as many delegates as objects to be animated by the crowd system. You can add them one by one, or use any of the standard 3DS MAX methods for cloning objects, including Shift-clone and the Array function. However, the Crowd object offers a convenient *Scatter Objects* (see page 362) function that lets you clone delegates and distribute the clones over a surface or within a volume, with options for orientation and scaling.

Once you've added enough clones for your crowd simulation, you can use additional Crowd facilities for grouping the delegates and applying behaviors. Because of its simple geometry and fast display, the delegate is ideal for producing an initial sketch of the crowd simulation. Once you're satisfied with the animation, you can use Crowd to link delegates to bipeds and other animated objects.

Delegate parameters define the nature of their motion. You access these parameters in the delegate's Motion Parameters rollout. They include speed, turning, and banking parameters. The speed parameters let you describe a delegate's average speed and maximum acceleration. You can also define how much a delegate slows down as it turns or goes upward, and how much it speeds up when it goes down. Turning parameters let you indicate how quickly a delegate can turn, and how much it can turn upward and downward. Banking parameters describe how much and how quickly a delegate banks and its banking limit. All of these parameters work together to describe different types of creatures. A small fish, for instance, can turn more quickly than a large bird. A fish's speed would not be effected when the fish traveled up or down, but a bird's speed might be altered by its upward or downward direction.

One particularly useful feature of delegates is their ability to display, via colored vectors, the strength and direction of the various forces acting upon them during solution of the crowd simulation. Each force can have a unique, identifying color. For example, the Seek behavior uses green by default, while the Wander behavior uses aqua. You can change these to any you like. If a simulation isn't proceeding as expected, you can debug it by observing the vectors during the solution. And if the solution occurs too quickly, you can use the Step Solve feature to solve the simulation one frame at a time.

## Creating a Crowd System

The following is a basic procedure for creating and using a crowd system:

1. Add a Crowd helper object.

2. Add a Delegate helper object.  
This is a prototype for your crowd; a representative member.
3. Set appropriate speed and turning limits for the delegate.
4. This is analogous to defining how a particular animal moves. Is it a bird, a fish, or a slug?
5. Use Crowd's *Scatter Objects* (see page 362) to create multiple clones of the delegate and distribute them within a volume or over a surface.
6. Use the Crowd object to add one or more behaviors.
7. Modify the behavior's default settings to be appropriate for the members of your crowd.
8. For situation-dependent (dynamic) simulations, create one or more *cognitive controllers* (see page 382).
9. Assign the behaviors and/or cognitive controllers to the delegates.
10. Test and refine the simulation.
11. This is an iterative process. You may have to modify the delegate and/or behavior parameters a number of times before you get the simulation you want.
12. Assign bipeds and/or other animated objects to the delegates.
13. Test and refine the simulation.
14. Add lights, cameras, other objects and effects as necessary and render the animation.

You can find further information and detailed procedures in *Crowd Helper Object* (see page 346). Also, be sure to try the *crowd tutorials* (see page 555).



## Crowd Behaviors Overview

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button  
Select a Crowd object > Modify panel > Setup rollout > New button

Behaviors let you assign procedural activity types to *delegates* (see page 342) and objects linked to delegates. You can associate any number of behaviors with each *Crowd object* (see page 346), and then link delegates and teams of delegates to each behavior. A specific behavior assigned to a Crowd object belongs only to that crowd; it cannot be assigned to any other crowds.

**Tip: character studio** gives generic names to behaviors. It's a good idea to rename them with more evocative descriptors; for example, "Seek Grail" or "Avoid Obfuscation".

Note: Certain behaviors, such as Seek and Avoid, let you specify "target" objects. You must specify these from the Modify panel; the button to do so is unavailable in the Create panel.

Following is a list of available behaviors:

*Avoid Behavior* (see page 392)

*Orientation Behavior* (see page 396)

*Path Follow Behavior* (see page 398)

*Repel Behavior* (see page 400)

*Scripted Behavior* (see page 402)

*Seek Behavior* (see page 403)

*Space Warp Behavior* (see page 404)

*Speed Vary Behavior* (see page 405)

*Surface Arrive Behavior* (see page 406)

*Surface Follow Behavior* (see page 409)

*Wall Repel Behavior* (see page 411)

*WallSeek Behavior* (see page 414)



*Wander Behavior* (see page 416)

The first time you add a behavior to the scene, a new rollout appears for the behavior below the Setup rollout. This rollout lets you change settings for the behavior. To display the rollout for a different behavior in the scene, choose it from the list below the Convert to MAXScript button. To see the controls available in the rollout for a behavior type, follow the link from its entry in the above list.

Note: In addition to the controls available in behavior rollouts, you can use the *Behavior Assignments and Teams dialog* (see page 375) to turn behaviors on and off (with the Active check box), and for all behaviors except Avoid, Orientation, and Surface Follow, you can set and animate Weight. The Active status is animatable for all behaviors.

---

## Understanding Behaviors

In the real world, different crowds exhibit diverse behaviors, and even members of the same crowd can conduct themselves in various and sundry ways. Included with **character studio's** Crowd system is an assortment of behaviors that let you simulate a range of crowd activities. And if you need custom behaviors, you can create your own.

Following is a list of available behaviors:

- **Avoid Behavior** (see page 392). Prevent delegates from colliding. Avoidance can use any combination of turning, braking/stopping, repelling, and vector field.
- **Orientation Behavior** (see page 396). Applies a fixed orientation or orientation range to delegates, so they face a specific direction instead of toward the destination. You can specify orientation in absolute terms, or relative to the direction the delegate currently faces.
- **Path Follow Behavior** (see page 398). Restricts motion to a spline or NURBS curve; options include back-and-forth patrol-type movement.
- **Repel Behavior** (see page 400). Forces delegates to move away from a target.
- **Scripted Behavior** (see page 402). Uses MAXScript to specify behavior.
- **Seek Behavior** (see page 403). Moves delegates toward a target or targets.
- **Space Warp Behavior** (see page 404). Uses any dynamics-oriented space warp to control movement, including wind and gravity. Vector Field, a crowd-specific space warp that lets delegates avoid irregularly shaped objects while following their contours, is included with **character studio**.
- **Speed Vary Behavior** (see page 405). Lets delegates change speed for more realistic movement.
- **Surface Arrive Behavior** (see page 406). Lets delegates move toward and land on a surface, with custom speed and acceleration parameters.
- **Surface Follow Behavior** (see page 409). Delegates move along a surface, which can be animated. Also, you can specify whether the delegates are to move straight ahead or skirt hills and depressions.
- **Wall Repel Behavior** (see page 411). Uses a grid to repel delegates; ideal for keeping objects inside an enclosed, straight-sided room.
- **WallSeek Behavior** (see page 414). Uses a grid to attract delegates. You can use this as a doorway for crowd-controlled bipeds to walk through.

- **Wander Behavior** (see page 416). Induces a realistic semi-random movement for characters such as shoppers at a mall.

## Using Behaviors

To use a behavior, you apply it to a delegate or a team of delegates using the *Behavior Assignments and Teams dialog* (see page 375). In this dialog, each assignment of a behavior to a delegate is given a weight. You can modify and/or animate these weights to influence the simulation.

Behavior assignment weights can profoundly effect a simulation. When applying two or more behaviors to the same delegate, the weights define the relationship between the behaviors, making one more or less powerful than the other. One way to visualize a behavior assignment weight is to examine the behavior's force vector during a crowd simulation. The vector's length indicates the behavior's weight upon the delegate.

Each behavior has its own parameters which appear in the Behavior rollout, available in the Crowd object's Modify panel. These parameters describe how the behavior works, and can sometimes contribute to the behavior's strength as well. For instance, Seek, Repel, Wall Seek, and Wall Repel, all have specific volumes of influence. Outside these volumes they have no effect and essentially have a weight of zero. This rollout lets you specify whether or not you wish to see behavior's force vector dynamically displayed during a Crowd simulation, and what color that vector should be.

When working with the Crowd system, it is critical to play with behavior assignment weights, as well as each behaviors' parameters. Typically, you run the simulation repeatedly, changing the weights and parameters to get the desired result.

A few behaviors cannot be weighted. These are Avoid, Surface Follow, and Orientation. Avoid and Surface Follow take over after all of the other behaviors have been applied to a delegate. They can take stringent measures to affect the delegate, possibly overpowering other behaviors in order to meet their constraints. Orientation simply sets the delegate's facing direction. It cannot be weighted and does not apply a force.

## Behavior Tips

A few helpful things to know about behaviors in **character studio**:

- You can create conditional behavioral systems with Crowd's Cognitive Controller feature. This uses 3DS MAX's MAXScript scripting language to determine when to effect a transition from one behavior to another; we've provided a number of sample scripts for you to learn from and adopt to your own simulations in *Cognitive Controller Editor* (see page 382) and *State Transition Dialog* (see page 386).
- The Behavior rollout appears immediately after the Crowd object > Setup rollout in the Modify panel. However, it doesn't show up until you've added at least one behavior to the crowd object.
- The Crowd panel displays only one Behavior rollout at a time. To access a different one, choose its name from the drop-down list at the bottom of the Crowd object's Setup rollout.
- As with most scene entities in 3DS MAX, it's a good idea to give behaviors custom names, such as "Seek Doorway" or "Follow Hilly Surface." You do this by clicking its name in the Setup rollout and entering a new one from the keyboard.

- The default behavior settings may not always give the ideal results. The optimal settings depend vary with the particulars of your simulation setup; in many cases, if not most, you'll need to experiment with the settings to get the results you want. In some cases, you may need to animate settings as well.

---

## Obstacle-Avoidance Behavior

An important part of crowd behavior is avoidance of obstacles. Think of an obstacle as anything that impedes a crowd member's progress. Examples of obstacles include walls, telephone poles, and fences, as well as other crowd members. Encountering such objects can cause avoidance behavior, which consists of any combination of slowing down, turning, and stopping.

There are several ways to implement avoidance in **character studio**, including:

- Use the *Avoid behavior* (see page 392) primarily to cause crowd members to avoid other crowd members. It works by creating a spherical *volume of avoidance* around the avoided object, so it doesn't accommodate irregular objects.  
The Avoid behavior is unlike any other behavior in Crowd. After all the other behaviors exert their forces on the delegates, Avoid takes over and has the power to turn, slow down, and even stop a delegate in order to make it avoid an obstacle.
- Use the *Wall Repel behavior* (see page 411) to cause crowd members to avoid broad, flat objects such as walls and fences. You can set a maximum distance for the repel effect, and describe the rate at which the force away from the wall increases as a delegate approaches the wall.  
Unlike the Avoid behavior, which can stop or slow down a delegate, Wall Repel simply exerts a force on the delegate to turn it away from the wall. It does not guarantee wall avoidance. You must work with its distance and falloff parameters, as well as its weight in the Assignments and Teams dialog, to control its strength.
- Use the *Repel behavior* (see page 400) to cause crowd members to turn away from an object. It works exactly like Wall Repel except that it uses a spherical volume rather than a plane. You can set a maximum distance for the repel effect, and describe the rate at which the repel force increases as the delegate approaches the obstacle.  
Repel exerts a force on the delegate to turn it away from the obstacle. It does not guarantee avoidance. You must work with its Distance and Falloff parameters, as well as its weight in the *Behavior Assignments and Teams dialog* (see page 375), to control its strength. Repel can be used instead of the Avoid behavior as a simple avoidance technique for non-terrestrial creatures such as fish or birds.
- Use a *vector field* (see page 417). A vector field is a special type of space warp that crowd members can use to move around irregular objects such as a curved, concave surface. You can use a vector field in conjunction with the Avoid behavior to make delegates slow down when they approach a complex object, and then go around it. This guarantees that the delegate will not pass through the obstacle's surface.  
You can also use a vector field in conjunction with the *Space Warp behavior*

(see page 404). This simply exerts a force on the delegate that mimics the contours of the object. It does not assure that the delegate will not pass through the surface of the obstacle. You can use a vector field with both the Space Warp and Avoid behaviors to combine their effects.

---

## Attaching an Object Instance to a Crowd System

You can use **character studio's** crowd system to populate a scene with copies of an animated object and link the copies to delegates. You can then use segments of the original object's animation to animate the copies in different ways depending on the delegates' activity.

Here's a general procedure:

1. Animate your master object with different motions for various activities it might undertake. For example, a bird might flap its wings rapidly to ascend, and then more slowly to maintain altitude. So you might animate a fast wing-flap cycle from frames 1 to 20, and then a slow wing-flap cycle from frames 31 to 70.
2. Make a copy of this object to use for cloning in step 4. Keep the original separate; this is a requirement of the Global Clip Controllers functionality.
3. Add a Crowd helper object and a Delegate helper object.
4. Use Crowd's *Scatter Objects* (see page 362) function to clone both the delegate and the animated object, and distribute them within a volume or over a surface. If you use the same Seed value, each object-delegate pair ends up in the same place.
5. Add behaviors and apply them to the delegates. Test and refine the simulation.
6. Use Crowd's *Object/Delegate Associations* (see page 370) function to link each object with its associated delegate.
7. Use the *Global Clip Controllers* (see page 359) tools to specify animation clips, assign states to each clip, and assign the clip controller to the object clones.
8. Solve the simulation, and use the Global/Master Clip controls to refine the animation clips and states as necessary.

### See also

*Clip Controllers* (see page 119)

*Synthesis Dialog* (see page 270)

*ClipState Dialog* (see page 275)

---

## Using Bipedals with Crowd Delegates

The biped crowd is a special class of Crowd necessitated by the complex nature of legged animal movement. Biped locomotion exhibits intricate dynamics and exacting IK foot constraints. As such, the smoothly curving trajectories computed from delegate motion parameters, while suitable for birds, fish, insects, and snakes, are not rich enough to animate the microstructure of bipedal motion. Therefore, several features in Crowd are focused on the special needs of bipeds.

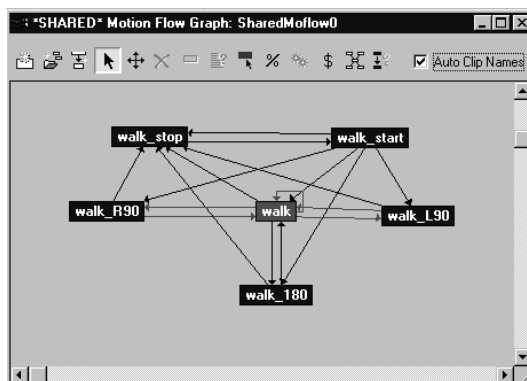
In order to generate the required level of nuance, animated motion clips form the basis for the repertoire of biped movements. With this approach, the animator can precisely control details in the motion either by using hand animation or employing motion capture to produce a set of clips that describe how a member in the crowd behaves.

For example, if you wanted to animate a crowd of marathon runners making their way through the streets of a city, you would need motion clips for various kinds of walking, running, jogging, resting, drinking water, cheering, etc. In effect, each of the motions you might expect to see in a marathon race could be represented as a clip. But motion must be more than a fragmented collection of clips. You must also consider how motions might be sequenced. Which motion transitions are possible from a given motion clip?

## The Motion Flow Network Defines Possible Scripts

Biped's *Motion Flow* (see page 227) functionality provides the mechanism for defining how separate motions fit together into a fluid animation. In effect, the motion flow network describes which motions can follow from other motions. Once the motion flow network is defined, a broad set of animated actions is possible by following different paths through the network. In Biped, a path through the network is called a motion flow script.

For example, shown below is a motion flow network used in the sample file *Walkers.max*. You can find this file in the *cstudio\tutorials\tutorial\_8* directory in your 3DS MAX path. This directory also contains the *.bip* files used in the motion flow network.



This is a fairly simple network of possible motions, because the characters can only start, stop, turn at 90 degree angles left and right (*walk\_L90* and *walk\_R90*), and do an about-face (*walk\_180*). However, for more natural crowd interaction, it's advisable to expand the motion flow network to include more finely tuned variation such as turning at 45 degree increments, moving in different directions while facing the same way, loitering motions, and moving at different speeds. The Biped Motion Library has a comprehensive list of clips for you to experiment with.

**Tip:** You can create motion clips that curve slightly to the left and right by applying Biped's footstep bending operation to straight-line motion clips. If the clips are motion captured, you should employ footstep extraction during import in preparation for the bending operation. Adding clips that turn slightly will let the biped crowd simulation make minute corrections in heading in order to achieve goal locations more precisely.

Motion flow networks may be shared by many bipeds using Shared Motion Flow Networks. Therefore, it's practical to make motion flow networks large without taxing your computer's memory.

## Bipeds Use Delegate-Directed Behavioral Goals

You can give a biped a behavioral goal by associating it with a delegate in the Crowd system, and then assigning behaviors to the delegate.

For example, in the *Walkers.max* sample, the behavioral goals of each of the biped's delegates are to:

- Move toward the sphere using the Seek behavior.
- Avoid hitting each other using the Avoid behavior.

During a biped crowd simulation, the software attempts to compute the best motion flow script for each biped member of the crowd that satisfies the behavioral goals of its associated delegate. In other words:

- The biped's crowd delegate defines the behavioral goals of the biped.
- The possible motions to reach those goals are defined by the biped's motion flow network.
- The Crowd plug-in's Solve operation computes the script, or path through the motion flow network, that best meets the goals of the biped's delegate.

So in the *Walkers.max* sample, the simulation will always choose the best available walking clip in the network that directs the biped's delegate toward the sphere. Each biped's script evolves as the crowd "Solve" computes. This is somewhat like a real-time "game engine" in that the crowd solver's choice of the next best clip for a given biped is restricted by that biped's currently active clip.

## Biped Crowd Avoidance, Priority, and Backtracking

Because bipeds in crowds are always following motion flow scripts, the avoidance behavior for bipeds works differently. Unlike ordinary delegates, biped delegates can move only along motion flow-scripted paths, so if a collision takes place, the software will *backtrack* (see page 354) to the previous clip in the current script and find another path. This may take some time to compute when complex crowd interactions are present since a single backtrack may not be enough. The computation will explore all paths from a given backtracking clip, and if that fails, it will backtrack to the previous clip, and so on, until a solution is found.

In the example, if the current script of a biped is:

```
walk_start
walk
walk_L90
```

and a collision is encountered during the *walk\_L90* clip, the biped will backtrack to the end of the *walk* clip and attempt to try a different clip in place of the failed left turn. If that fails, it will try the next best choice, and so on.

**Tip:** The inclusion of stopping and loitering motions in the motion flow network is sometimes helpful in preventing excessive backtracking since stopping is always an effective way to avoid collisions in a tight situation. In general, the more variation in speed and direction that is possible, the more quickly the backtracking feature will find a solution.

In order to make the backtracking computationally manageable, the biped crowd members are computed one at a time in order of *priority* (see page 354). Thus, the crowd interaction is accumulated with each successive biped added to the animation. In other words,

each waits its turn to compute its complete animation, which entails avoiding the bipeds that have been computed before it. It follows that bipeds with the lowest priorities generally encounter the most collisions, since they must steer around all the bipeds that have higher priorities.

### See also

*Crowd Helper Object (see page 346)*

*Creating a Crowd of Swimming Bipeds (see page 573)*

*Advanced Crowd/Bipeds (see page 577)*



## Clip Controllers

The Global Motion Clip and Master Motion Clip controllers are used to create animation for multiple objects. Groups of creatures such as birds, butterflies, schools of fish and bugs can be animated using these tools. You can create clip controllers either as block controllers in Track View or more directly in the Crowd helper controls on the Global Clip Controllers rollout. Clip controllers are generally used to animate non-biped creatures.

### Two Approaches to Animation

You can animate your creature either in place with looping animation but no transformational motion (such as a bird flapping its wings), or you can incorporate transformational motion into the animation as well (the bird moves upward while flapping its wings). In-place cyclic motion lends itself to flying or swimming motions like birds and fish, while adding lateral motion lends itself to crawling type animation where feet should be planted on the ground and not sliding. Depending on which you use, you toggle options on the Motion Clips tab of the

Synthesis dialog. In both cases, you use crowd delegates driven by behaviors to motivate the creatures, which are linked to those delegates.

### Cyclic In-Place Animation

First you create a creature with a few short loop cycles, like the beating of wings, gliding, turning left and turning right. This creature is assigned as the *Global Object* or the master object from which the motion clips will be derived. Then clones of the original creature are created. The clones are positioned and linked to delegates. States are created to select which clips will play based on a state.

For example, if a bird (delegate) is pitching up or accelerating, the fast-beating clip is used; if the bird (delegate) slows to a stop, the wings-at-rest clip is used, and so on. During synthesis, the software determines which state should be active depending on the speed and direction of the delegates. An active state determines which clip should be applied to the clones of the original object. Clips are blended together to create the animation. Available states are speed, acceleration, pitch, pitch velocity, heading velocity, or script (MAXScript).

The *Using Crowd with Animated Non-Biped Objects (see page 567)* tutorial covers this method.

### Animation with Lateral Motion

For multi-legged creatures that walk, you can animate lateral motion as well as the cyclic motion of the legs moving. This is done to ensure that the creatures' feet do not slide as it moves. The software then uses the lateral motion information to create a state that perfectly matches the actual motion. The software then strips the actual motion out. When a delegate approaches the speed and heading recorded in that state, the appropriate

motion clip is triggered. This technique minimizes sliding feet.

Use the **character studio** crowd tools to create the initial motion for the delegates. Use a seek or avoid behavior to steer birds, for example. Your object with the loop animation is then copied and the copies are linked to the delegates to create the complete animation. The delegate handles the path and the clip controllers handle the looped animation.

You can create Master Motion Clip and Global Motion Clip by going into Track View and assigning a controller to the available controller under Block Control. It is, however, simpler to use the user interface in the Crowd helper controls on the Global Clip Controller rollout to apply and use these clip controllers.

### See also

*Synthesis Dialog (see page 270)*

*Clip State Dialog (see page 275)*

## Global Motion Clip

Global Motion Clips store the clips to be shared among multiple MasterMotionClips.

GlobalMotionClips also contain the logic for performing motion synthesis on a collection of objects with trajectories and states associated with clips. Controls for motion synthesis are in the Synthesis dialog.

The way the motionclip keys are scaled and ordered depends upon user-defined states. Each state contains one or more motionclips that will be played when the state is active.

## MasterMotionClip

MasterMotionClips are controllers that contain *motionclips* or individual clips of animation. These motionclips are combined end to end to

create animation. Motion clips that overlap are blended to smooth the transitions between clips.

## Synthesis Dialog

All of the work involved in copying and synthesizing clips takes place using controls in the *Synthesis Dialog (see page 270)*.

### Steps to synthesize objects

1. First, a *global object* is selected that subsequent motionclips will be created from. Think of the global object as a template that is used to create the motionclip animations, but itself is never synthesized.  
A motionclip is a subset of an object's animation. For example, flapping and gliding motion in the objects animation yields two motionclips.
2. From this global object, animations are created and stored as motionclips.  
You can animate the character in place or with lateral movement, the software can handle both cases. A bird can have a flap animation, a dive animation, and a land animation, which when blended, scaled, and combined can create effective-looking animations for a large number of creatures.
3. Create states are created that specify when a particular motionclip is active. For example, a flap motionclip is active if the acceleration of the bird (delegate) is greater than zero, while a glide motionclip is active if the acceleration is less than zero.
4. Objects to be synthesized will probably be clones of the global object and need to contain the same node and controller hierarchy. Otherwise they can be different



in other ways; for example, they may be scaled, textured or skinned differently. For each object selected to be synthesized, an associated MasterMotionClip controller is created under the GlobalMotionClip in Track View. It is here that the synthesis algorithm places the keys.

5. You also need to specify where the blending between two motionclips occurs. You can do this automatically by letting the computer analyze the motionclips and find a transition point that minimizes error, or you can manually specify the frame number at which the blend should start.
6. Synthesis for the selected objects occurs based upon which states are active, the available motionclips under each state, and the specified transition points.

You can also hand modify the resulting animation by manipulating the keys inside Track View, or by collapsing the keys completely (Collapse Selected on the Synthesis tab) and then hand modifying the resulting animation in the scene.



---

## Chapter 2

# Procedures

Most of these procedures appear elsewhere in this Online Reference, embedded in their associated topics. Here they are ordered in a logical progression.

## Using Biped and Physique

### Creating or deleting a biped

*To create a biped (see page 143)*

*To delete a biped (see page 24)*

### Naming the biped and creating a named selection

*To name the biped (see page 184)*

*To create a named selection for the biped center of mass (see page 146)*

### Loading a biped motion file and seeing the biped in motion

*To load a biped motion file (see page 151)*

*To preview biped motion using 3DS MAX (see page 151)*

*To preview biped motion using Biped controls (see page 151)*

### Fitting the biped to a mesh, then attaching the mesh to the biped using Physique or linking (to see your own character in motion)

*To create and fit a biped to a mesh for use with Physique in Figure mode (see page 147)*

*To link a mechanical character to the biped (without Physique) (see page 148)*

*To fit biped legs to the skin (see page 148)*

*To fit the spine to the skin torso (see page 149)*

*To fit biped arms to the skin (see page 149)*

*To use Rubber Band to position the elbows and knees (see page 150)*

*To save a biped Figure file (see page 150)*

*To load a biped Figure file (see page 150)*

*To attach a mesh to a biped using Physique (see page 296)*

*To attach a mesh to a 3DS MAX bones hierarchy using Physique (see page 296)*

### **Adjusting skin behavior with Physique parameters** (once the mesh is attached to the biped with Physique)

*To adjust envelopes around the biped's pelvis with Physique (see page 301)*

### **Animating the biped in freeform: set all the biped center of mass and limb keys**

*To start a freeform animation (see page 128)*

*To prevent sliding feet in a freeform animation (see page 128)*

*To put the biped legs into world space (see page 168)*

*To reposition a biped with freeform animation (see page 128)*

*Using Custom Keys (see page 141)*

*To set a pivot point for a hand or foot rotation (see page 168)*

*To use pivots (see page 78)*

*To set keys that define animatable IK attachments (see page 170)*

### **Animating the biped using the footstep approach: get a head start by generating a walk, run or jump**

*To create multiple footsteps (see page 248)*

*To walk up stairs (see page 250)*

*To walk down stairs (see page 250)*

*To walk in place (see page 250)*

*To walk backwards (see page 250)*

*To create footsteps manually beginning at the current frame (see page 248)*

*To append footsteps onto the existing footsteps (see page 248)*

*To make the biped speed up as it walks (see page 250)*

*To activate footsteps (see page 258)*

*To deactivate footsteps (see page 258)*

*To enable foot motion on a footstep (see page 62)*

### **Editing footsteps in space and time**

*To move footsteps (see page 258)*

*To rotate footsteps (see page 258)*

*To bend the footsteps' path (see page 258)*

*To delete footsteps (see page 258)*

*To move footsteps in the time (see page 266)*

*To change the duration of a footstep (see page 266)*

*To select a footstep (see page 266)*

*To select multiple footsteps (see page 266)*

*To scale the stride length and/or width (see page 258)*

*To create a "high step" type walk from a default biped walk cycle using Set Multiple Keys (see page 173)*

*To make the biped stand still after a walk period (footstep animation) (see page 129)*

*To make the biped walk on a moving object in your scene (see page 129)*

*To display footsteps (see page 176)*

*To save footstep data (see page 151)*

*To load footstep data (see page 151)*

*To save biped motion file (.bip) (see page 151)*

*To edit the footstep buffer before you insert its contents by splicing (see page 260)*

## Splicing footsteps

*To edit the footstep buffer before you insert it by splicing (see page 151)*

*To splice biped motion (see page 259)*

## Separating tracks and editing biped keyframes (keys)

*To create separate tracks for biped arms (see page 181)*

*To use In Place mode to adjust keyframes (see page 150)*

*To locate vertical center of mass keys (see page 160)*

*To set a body vertical key (see page 160)*

*To set a body rotation key (see page 161)*

*To change TCB for a biped arm (see page 162)*

*To make a global posture change to the biped (see page 179)*

*To change vertical dynamics for multiple center of mass keys in Track View (see page 266)*

*To set a body horizontal key (see page 160)*

*To use Symmetrical Tracks (see page 160)*

*To use Opposite Tracks (see page 160)*

*To turn on Spline Dynamics for newly created keys (see page 181)*

*To copy keys between bipeds (see page 129)*

*To link the biped hand to an object using Object Space Object (see page 169)*

*To use Adapt Locks to prevent keyframe adaptation (see page 182)*

*To bend the horizontal center of mass track (see page 173)*

*To edit keys on the Center of Mass trajectory (see page 173)*

*To copy and paste an entire biped track (see page 173)*

*To switch the left and right arm tracks (see page 173)*

## Key editing in Track View

*To clone biped keys in Track View (see page 266)*

*To create a freeform period (see page 267)*

*To copy keys in the left arm track to the right arm track (see page 266)*

## Optimizing raw motion capture data before using Motion Flow Mode to cut motions together

*To import a motion capture file (see page 188)*

*To use Fit to Existing to import a motion capture file (see page 193)*

*To use Convert From Buffer (see page 187)*

*To compare raw and filtered trajectories (see page 187)*

*To use Show Buffer (see page 188)*

## Motion Flow mode

*To create clips in the Motion Flow Graph (see page 231)*

*To create a script (see page 235)*

*To create a manual transition between two clips (see page 238)*

*To load a MFE file (see page 229)*

*To share a motion flow among multiple bipeds (see page 244)*

*To create a random script for one biped. (see page 242)*

*To load scripts and create optimized transitions (see page 232)*

*To move footsteps to line up clips (see page 235)*

*To use the Optimize Transition Range dialog (see page 246)*

## Physique

*To add a bone after Attach to Node is used (see page 296)*

*To troubleshoot bulges and tendons (see page 298)*

*To select cross sections in the inner or outer bounds (see page 301)*

*To copy an envelope and its settings to the mirrored link (see page 301)*

*To copy all bulge angles for one link to its opposite (see page 315)*

*To use Select Nearest Bulge Angle (see page 315)*

*To remove a link's influence on vertices (see page 331)*

*To check vertex assignments (see page 331)*

*To remove deformable vertices from a link's influence (see page 331)*

*To assign deformable blended vertices manually (see page 331)*

*To override vertex assignments manually (see page 332)*

*To make vertices rigid (see page 332)*

*To have a spline influence a mesh (see page 295)*

**Finishing your character by bulging the mesh at appropriate biped limb angles in the Bulge Editor and at the Bulge sub-object level**

*To create a new bulge angle using the bulge editor (see page 319)*

*To create a new bulge angle on a selected link (see page 314)*

*To add a cross section (see page 320)*

*To change the shape of a cross section (see page 321)*

*To make a cross section the active cross section (see page 320)*

*To choose a specific bulge angle for editing (see page 319)*

*To change a bulge angle value (see page 319)*

*To select multiple cross sections (see page 320)*

*To move cross sections along the link (see page 320)*

*To copy and paste cross sections (see page 321)*

*To change the Profile view orientation (see page 321)*

*To delete a bulge angle (see page 320)*

*To delete a cross section (see page 320)*

## Using Physique Links and Tendons sub-objects

*To adjust joint intersection parameters (see page 308)*

*To create and attach a tendon (see page 327)*

*To attach a tendon to another link (see page 327)*

*To delete a tendon (see page 327)*

## Saving and reusing Physique information

*To save Physique data (see page 297)*

*To load Physique data (see page 297)*

## Using Figure mode

*To adjust the biped center of mass with Rubber Band (see page 150)*

*To scale a biped and a Physique mesh, using Height on the Structure rollout (see page 184)*

## Miscellaneous

*To create bounding boxes on a non-biped hierarchy (see page 294)*

*To disable the Non-Uniform Scale warning (see page 294)*

*To scale a biped and mesh attached with Physique (see page 150)*

*To play a sample animation (see page 290)*

*To apply Physique to an FFD to animate the entire mesh (see page 340)*

*To use an FFD to complement the effects of Physique on a portion of a character mesh (see page 340)*

*To use facial animation with character studio (see page 131)*

*To make compressible bones (see page 138)*

*To add a bone after Physique is applied using Reinitialize (see page 138)*

*To add a bone after Physique is applied using Add (Add Bone) (see page 139)*

*Using Custom Keys (see page 141)*

*To activate an anchor (see page 170)*

## Using the Crowd System

### Using a delegate helper object

*To add a delegate object (see page 342)*

*To link objects and delegates (see page 370)*

### Using the crowd helper object

*To add a Crowd helper object (see page 347)*

*To produce multiple clones of an object (see page 362)*

*To group delegates into a team (see page 376)*

*To create a new behavior assignment (see page 376)*

*To modify an existing behavior assignment or assignments (see page 377)*

*To use Crowd and Delegate helper objects together (see page 347)*

*To use bipeds in a crowd simulation (see page 347)*

## Using behaviors

*To use the Avoid behavior (see page 393)*

*To use the Orientation behavior (see page 396)*

*To use the Path Follow behavior (see page 398)*

*To use the Repel behavior (see page 401)*

*To use the Seek behavior (see page 403)*

*To use the Speed Vary behavior (see page 405)*

*To use the Surface Arrive behavior (see page 407)*

*To use the Surface Follow behavior (see page 410)*

*To use the Wall Repel behavior (see page 412)*

*To use the Wall Seek behavior (see page 414)*

*To use the Wall Seek behavior (see page 414)*

## Using cognitive controllers

*To set up and use a cognitive controller (see page 382)*

*To test a modifier parameter (see page 388)*

*To test a particle system parameter (see page 387)*

*To test an atmospheric property (see page 387)*

*To test an object position (see page 387)*

*To test another delegate's behavior (see page 388)*

*To test the distance between two objects (see page 387)*

## Using vector fields

*To add a Vector Field space warp (see page 417)*

*To use a Vector Field space warp (see page 418)*

*To use a Vector Field space warp with a particle system (see page 419)*

## Using clip controllers

*To use the Synthesis and ClipState dialogs (see page 275)*

See also

*Biped FAQs and Procedures (see page 128)*

*Keyboard Shortcuts (see page 130)*

## Biped FAQs and Procedures

This topic contains procedures that answer frequently asked questions about Biped animation.

See also

*Procedures (see page 123)*

### To start a freeform animation

1. Create a biped (see page 143).
2. Open the Motion panel to display Biped controls.
3. Turn on the Animate button and move any part of the biped.
4. Click Yes in the Freeform dialog.
5. Turn on the Animate button and keyframe the biped limbs at various frames.
6. Keyframe the biped center of mass position to move the entire biped.

Keyframe the biped limbs and the center of mass at different frames to create your character animation. Click play to view results. The next How To explains how sliding feet can be rectified in a freeform animation.

### To prevent sliding feet in a freeform animation

1. Move the time slider to frame 0.
2. Select a biped foot.
3. Turn on the Animate button.
4. On the *Key Info* rollout (see page 161), set IK Blend to 1 and turn on Object (Body is selected by default).

This puts the biped leg into world space. The foot stays planted if you move the center of

mass to reposition the biped. New keys retain IK Blend and Object settings. IK Blend can be varied to 0 for foot keys in the air; this changes the foot trajectory from one footstep to the next. Tension, Continuity and Bias parameters also change trajectories.

5. Repeat the process for the opposite foot. This makes it easier to re-position the biped without the feet sliding.

Note: You can also use Set Planted Key in the IK Key Info rollout. This automatically sets IK Blend to 1 in Object space with Join to Prev IK Key turned on.

### To reposition a biped with freeform animation

Unlike moving all the footsteps to reposition a character using footsteps, the center of mass is linked to a dummy and the dummy is moved to reposition a freeform animation.

1. On the Create panel, click Helpers, then select Dummy.
2. Create a dummy in the viewports. The dummy can be created anywhere.
3. Select the biped center of mass.
4. Click to turn on the Link tool on the 3DS MAX toolbar and drag from the center of mass object to the dummy object in the viewports.
5. Select and move the dummy object to reposition the biped and its freeform motion.

Note: The position spinners on the Motion Flow Script rollout can also be used to reposition a biped with freeform motion. After using the spinners to reposition the biped, use Save Segment to store a repositioned BIP file. Exit Motion Flow mode and load this BIP file (the one with the biped in a new position).



### To make the biped stand still after a walk period (footstep animation)

This assumes you are working with a default walk, generated with Create Multiple Footsteps.

1. Open Track View and stretch the tail edge of the last two footsteps to line up at the same frame in time to create a “stand still” period.
2. In Footstep Mode, move the last footstep in space and position it next to the second-to-last footstep. The character’s feet will be close together during the standing period.  
Note: The Bip01 transform track in Track View. The center of mass has a vertical, horizontal, and turning key at the end of the last footstep. The last frame of the footsteps should be lined up in time at this frame.
3. Select Body Vertical in the Track Selection rollout, then use Next Key in the Key Info rollout to set the current frame at end of the last footstep.
4. Click Copy Pose in the Keyframing rollout. This is a two-button flyout: be sure to click Copy Pose, not Copy Posture. The entire biped pose is stored.
5. Use Previous Key in the Key Info rollout to find the vertical key at the start of the standing period and set the current frame.
6. Click Paste Posture in the Keyframing rollout.  
The entire biped posture is pasted at this frame. Keys are inserted for the horizontal and turning tracks.
7. In Track View, and for any biped track, delete keys that exist during the standing period.
8. In the Key Info rollout you can decrease continuity for the vertical, horizontal and turning keys on either side of the standing

period to eliminate motion between the keys.

### To copy keys between bipeds

1. Scrub the time slider to find a posture you want to copy.
2. Select biped limbs and click Copy Posture on the Keyframing rollout.
3. Select another biped in the scene.
4. Click Paste Posture.

Posture from the first biped is pasted to the second biped. One biped can remain in the scene as the posture biped; if you constantly need to use certain postures, key them at different frames on one biped, then copy and paste them to the biped you are animating. Use Copy Pose to copy the entire biped including the center of mass Position. Use Copy Posture to copy selected limbs.

### To make the biped walk on a moving object in your scene

1. Select the biped center of mass.
2. Turn on the 3DS MAX link tool on the toolbar and drag from the center of mass to another object in the viewports.

If the object is moving, the biped moves with it; biped animation is now relative to this object. Use this to animate a biped on a ship or train for example.

## Keyboard Shortcuts

Customize menu > Preferences > Preference Settings dialog > Keyboard tab > Choose Plug-Ins. > Choose Biped.

The table in this topic shows the default keyboard shortcuts for **character studio 3**. New in this version is the ability to customize keyboard shortcuts. To do so, use the path specified above.



Use the Plug-in Keyboard Shortcut toggle by the 3D Studio MAX prompt line to enable **character studio** keyboard shortcuts.

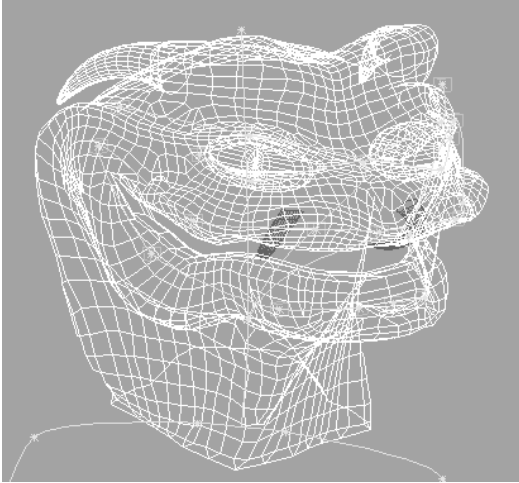
All **character studio** keyboard shortcuts activate when the Biped panel is active and the Plug-in Keyboard Shortcut button is active.

Shortcut	User Interface Function
TAB	Toggles forward between selection levels when in sub-object level for Envelope, Bulge, and Tendons: from Select Link to Select Cross Section to Select Control Point.*
SHIFT-TAB	Toggles backward between selection levels when in sub-object level for Envelope, Bulge, and Tendons: from Select Control Point to Select Cross Section to Select Link.*
PAGE UP PAGE DOWN	Takes you to the next or previous link, cross-section, or control point, depending upon the current selection.*
0	Sets a biped key.
Q	Alternates footstep creation from left-to-right and back. Watch the footstep type change in the status panel.*
S	Solves crowd simulation.*
T	Step-solves crowd simulation.*
V	Toggles Biped playback.
CTRL+E	Resets envelopes for selected links to their default values.

Shortcut	User Interface Function
ALT+A	Selects all left edges of the selected footsteps in Track View.
ALT+B	Pastes Posture Opposite for the currently selected biped
ALT+C	Copies the posture of the selected biped objects to the clipboard.
ALT+D	Selects all right edges of the selected footsteps in Track View.
ALT+R	Resets the total animation length to the length of the current biped footstep range.
ALT+S	Selects both edges of the selected footsteps in Track View.
ALT+V	Pastes the posture from the clipboard onto the currently selected biped.
CTRL+ALT+L	Toggles locked keys (red) on and off for the selected leg or vertical track key at the current frame. In Track View, you can watch the key turn from red to gray, and back again, as you lock and unlock it.
CTRL+ALT+S	Toggles foot states of the selected leg at the current frame. View the state change in the leg states displayed on the General rollout.
CTRL+ALT+F	Searches for any problem in the motion and prompts you whether or not to fix problems it encounters. Problems it looks for include overlapping keys, keys past the end of the footstep range, keys at negative frames, or illegal footstep timing. Click OK when prompted to fix these problems automatically.
CTRL+ALT+E	This toggle strips the scale from the biped. Developers should use this when exporting biped objects as regular 3D Studio MAX links through the 3DS MAX SDK. Animators should not use this shortcut.

\* Not available for customization in Preferences.

## Facial Animation with character studio



Note: *snake2.avi*, *snake2.max*, and *cs.wav* are the files used in this example of facial animation. You can locate them in the *cstudio\tutorial* directory.

As an alternative to morphing, Physique can be used effectively for detailed facial animation when a character's face is set up with extra links. Using 3DS MAX bones or placing dummy objects at appropriate locations on the mesh head and linking them to one another, you can define a skeletal structure for moving the facial features. You can also use splines linked to the biped head to deform the mesh. You can add a spline to a character that already has Physique applied by using Add in the Physique Bones rollout and clicking the spline in the viewports.

In the example used for this topic, a snakelike character is made to talk. Dummy objects (blue cubes in the image) were linked to each other, and the parent-most was linked to the Biped head. Speech and facial expression is created by keyframing the dummy objects.

Vertex-Link assignments are determined by Physique envelopes. A single large rigid envelope on the head link insures that the whole head follows the biped. The new facial links each have deformable envelopes that add influence to specific facial features.

The 3DS MAX hierarchy does not recognize Biped's Figure mode, so special consideration is needed to establish the initial skeletal pose. Leave Figure mode on to position and link the dummy objects at frame 0; frame 0 will act as the "at rest" position for the dummies. You should start keyframing the face at frame 1 or later.

In this example the dummy in the middle of the upper lip and the dummy between the eyebrows are linked to the biped head. All the dummy objects that are positioned around the character's lips are linked in a chain to the dummy in the middle of the upper lip. The dummy objects around the character's brow are linked in a chain to the dummy between the eyebrows.

### How the Dummies Should Be Linked

The dummies around the mouth are linked to create a deformation spline that follows the contours of the lips when Physique is applied. This allows smooth lip deformation when the mouth dummies are animated. The dummies on the lower eyelid create envelopes to control cheek movement, as when the character smiles, and also act as an anchor to the cheeks when the upper lip is animated. In this example, the eyebrow dummies and corresponding links control eye expression and blinking.

The snake's face is relatively simple. On the other hand, if your character has a large jaw, you may want to create links around the jawbone to separate lip motion from jaw motion. If your character has large eyes, you can add dummies

and links to control eyelids separately from eyebrow motion.

The envelopes on the links between the biped head and the dummy objects are turned off at the Envelope sub-object level. Envelopes around the lips and brow are adjusted, also at the Envelope sub-object level.

## Procedures

### Isolate vertices of the lips from influence by inappropriate links

When working with a complicated facial bone structure, envelopes for the lower lip are bound to affect vertices in the upper lip, and vice versa. In general, this can be corrected in at the Vertex sub-object level:

1. Select the head and on the Modify panel turn on Vertex sub-object.
2. Select Initial Skeletal Pose
3. Click Select and fence select vertices of the lower lip, and then click Remove from Link.
4. Select the links of the upper lip, and click Lock Assignments.
5. Fence select vertices of the upper lip, and click Remove from Link.
6. Select links of the lower lip, and click Lock Assignments.

### Keyframing the Dummies

By default, when a dummy is moved the child objects of that dummy will move also. Use the Don't Affect Children control in the Hierarchy panel, with Pivot active, if you don't want to move the child objects of the dummy you are positioning.

For example, if the dummy in the middle of the upper lip is moved, the entire mouth moves, because the child dummies inherit the move.

Turn on Don't Affect Children to move this dummy independently.

Bezier position interpolation, the default 3DS MAX position controller, will sometimes overshoot a dummy key. For example, after setting a lower lip key at frame 10, you may find that at frame 13 the lip is positioned too low. Use the Bezier spline controls in the Motion panel or in Track View to correct for overshoot. For example, choose linear interpolation for the "In" and "Out" for a key.

### Lip Sync

A sound track can be loaded in Track View by selecting Sound in the Track View hierarchy, selecting properties, and using the Sound Options dialog to load an audio file. Turn on Real Time in the 3DS MAX Time Configuration dialog to enable sound when you use the Play button. By scrubbing the time slider, you can locate a sound and keyframe the dummies to appropriate positions.

---

## Chapter 3

# User Interface

---

## Biped User Interface

The biped user interface is split up into “modes” of operation. There are three modes available: Figure mode, Footstep mode and Motion Flow mode. You activate these modes by selecting the appropriate button in the General rollout of Motion Panel, which are visible when a biped is selected.



Figure mode is used to change the biped skeletal structure and to align the biped to a mesh.



Footstep mode is used to create and edit footstep animation.



Motion Flow mode is used to create scripts that combine motion files into longer animation.

As with other parts of 3DS MAX, the rollouts change depending on the mode the software is in. When no modes are active, the following rollouts are displayed:

- General
- Track Selection
- Key Info
- IK Key Info
- Keyframing
- Display
- Layers
- Motion Capture
- Animation Properties
- Structure

Sometimes a rollout is displayed, but the controls are unavailable. For example, the Structure rollout controls are only available when Figure mode is active. Similarly in Footstep mode, Footstep Operations are unavailable until you have created some footsteps.

In Figure mode the rollouts displayed are: Track Selection, Keyframing, Display and Structure.

In Footstep Mode the rollouts displayed are: Track Selection, Footstep Creation, Footstep Operations, Display and Animation Properties.

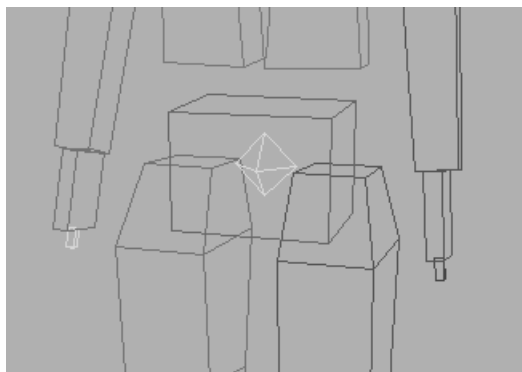
In Motion Flow mode the rollouts displayed are Motion Flow, Motion Flow Script, Track Selection, Display and Structure.

When no modes are active, you can edit tracks and keys, set IK constraints, work with layers and work with motion capture data. You can also create freeform animation without any of the modes active, simply by turning on the Animate button, and then moving or rotating any part of the biped.

---

## Center of Mass

The parent object of the biped is its center of mass (*Bip01*). This object appears as an octahedron near the center of the biped's pelvis.



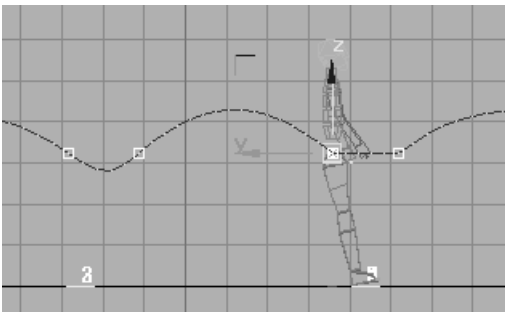
**The Center of Mass (COM) Object is the root.**

Moving the center of mass repositions the entire biped. The center of mass uses three animation tracks to animate the biped. Two of these tracks, Body Vertical and Body Horizontal, contain Biped Dynamics parameters.

## Body Vertical Track: Dynamics and Ballistic Tension

The Body Vertical track uses the Dynamics Blend parameter to control gravity in a footstep animation. A Dynamics Blend value of 1 uses the value of GravAccel (global gravity value) to calculate an airborne trajectory for the biped; no keyframes are required to position the biped in the air, a trajectory is calculated automatically. A value of 0 uses Spline Dynamics for the vertical position of the biped; you must create keyframes to position the biped vertically. A Dynamics Blend value between 0 and 1 will use a blend of Spline Dynamics and Biped Dynamics to calculate a trajectory for the center of mass.

The Body Vertical track also has a Ballistic Tension parameter that controls how much the biped knees will bend when the biped lands from an airborne period. This means that keys do not need to be created at the lowest position of the biped after landing, a trajectory is calculated automatically.



Ballistic tension set to 0 in the first dip and 1 at the second dip.

Note: Center of mass keys can be created manually to override the calculated trajectory.

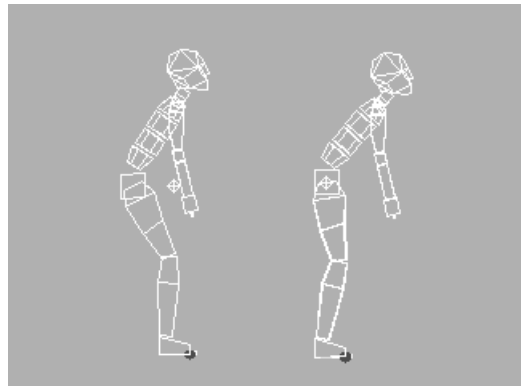
## Body Vertical Track: Balance Factor

The Body Horizontal track has a Balance Factor parameter that automatically orients the biped to maintain balance. This saves the animator from having to reposition the pelvis when the biped leans forward, backwards or sideways.

Note: Biped Dynamics parameters can be animated from no effect to full effect between keyframes.

If you are starting a new footstep animation, but would prefer to use Spline Dynamics rather than the default Biped Dynamics, then turn on Spline Dynamics on the Animation Properties rollout before creating any footsteps.

If you are starting a freeform animation (no footsteps), only Balance Factor is available. Freeform animation uses Spline Dynamics for center of mass positions.

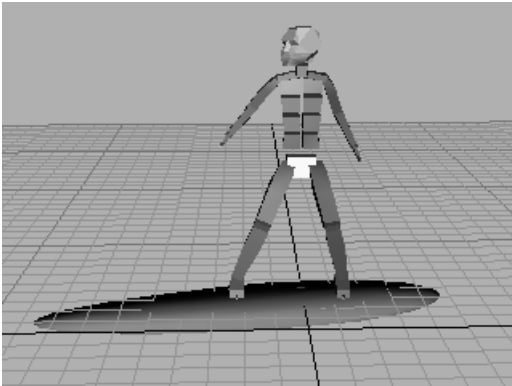


Balance Factor set to 2 on the left biped and 0 on the right biped.

## Linking to the Center of Mass Object

You can link the center of mass object to another object if you need to reposition an animation sequence. An example of this might be a surfer. You can create an animation of a biped running up and down the surfboard, hanging its toes off

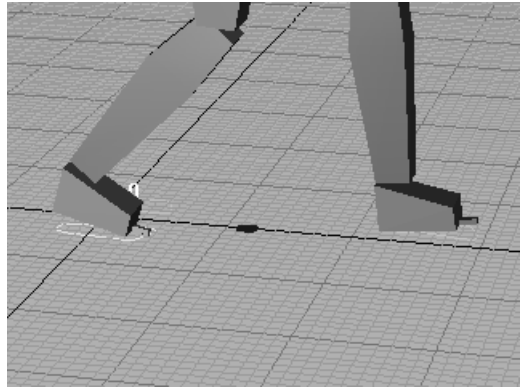
the end of the surfboard, etc., and then link the center of mass object to the board. When you animate the surfboard, the biped animation will travel with it.



The biped moves with the surfboard because the center of mass is linked to the surfboard.

### Center of Mass Shadow

The center of mass shadow, the circle between the biped's feet on the world plane, provides a sense of where the character's center of mass is positioned relative to the feet. Another use of the center of mass shadow is that objects can be linked to it. For example, a camera and camera target can be linked to the shadow object to follow the character.



The center of mass shadow between the bipeds feet.

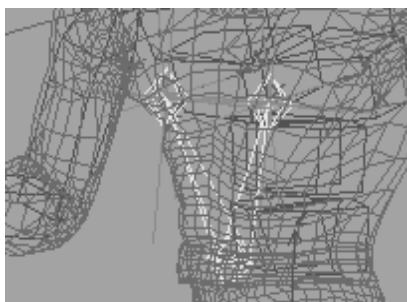
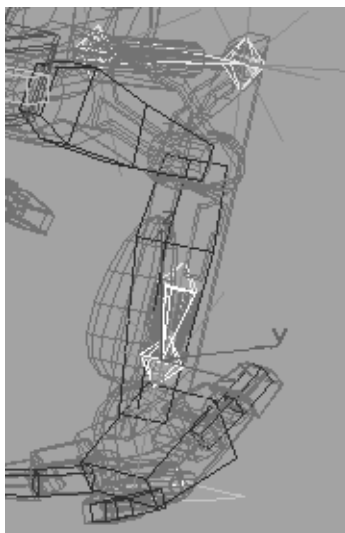
---

## character studio and 3DS MAX Bones

3DS MAX bones with or without the IK Controller can be used with the biped for various effects. They can be used with Physique to add extra links and envelopes for any character, or to animate assemblies on a robot or mechanical character. Bones can also be added to animate extra appendages, a hat, a jaw and so on.

Note: If 3DS MAX bones using the IK Controller are linked to the biped, the Animate button must be left on while the biped is positioned.





### Two examples of compressible bones

In the image on the left, 3DS MAX bones are used to animate the linked piston assemblies. The image on the right shows 3DS MAX bones used for added control when Physique is applied. In both cases the bones compress automatically as the biped is positioned.

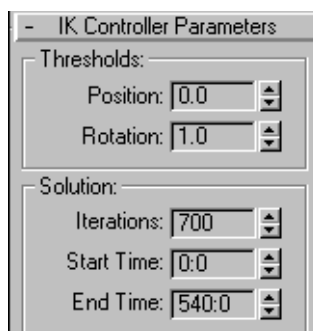
### Bones that compress

One way to use bones with the biped is to use the Link tool on the 3DS MAX toolbar to link the root of the bone to one part of the biped and have the bone End Effector follow another part

of the biped. As the biped moves these bones compress and expand.

In the image of the mechanical leg, piston objects are linked to the bones. As the bones compress and expand they also animate the linked piston assembly. As the bones compress and expand in the abdominal area in the image on the right, the envelopes created by Physique for the bones compress and expand to animate this area on the mesh.

### IK Solution

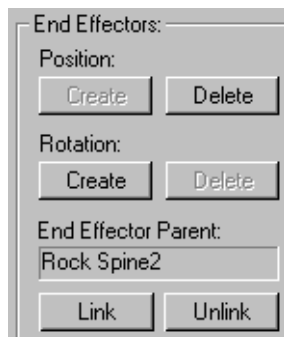


The bone end effector (blue cross in the viewports) will follow its parent, which can be another part of the biped. As the biped moves, the bone will compress and expand. Use Link on the Motion panel to specify an End Effector Parent. In most cases all you need is a Position End Effector.

In the IK Controller Parameters rollout, the accuracy of the IK solution and the frame duration of the solution are specified. For bones that compress, you generally want an accurate solution, this is to ensure that the bone follows the end effector perfectly, without any drift. In the image of the rollout Position has a value of 0, the least amount of allowable distance (Rotation is at its default value of 1, the Rotation End Effector is not used in this example).

Iterations is set high, this ensures that that the solution is accurate even if the character is scaled down a great deal. Start and End Time should span the animation.

There are many ways to use bones with the biped, you could automatically animate the bending of a hose that is attached to a character's shoulder and mouth when the biped head rotates for example. Bones attached to the biped create links and envelopes when Physique is applied; these extra Physique links offer localized skin control if necessary. See *mech.max* in `\cstudio\characters` for one example of how bones can be used.



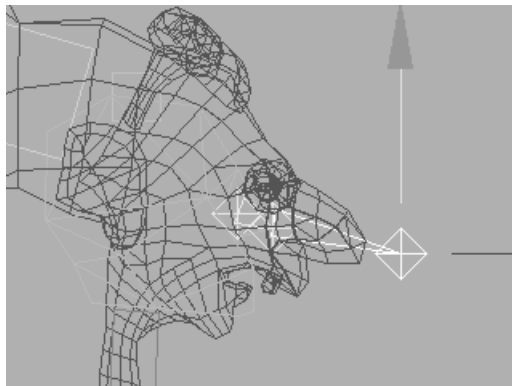
## Procedures

### To make compressible bones (bones with end effectors)



1. Create a bones system with end effectors.
2. Link the root of the bone to the appropriate biped object.
3. On the Motion Panel specify an End Effector Parent.  
Specify another part of the biped.
4. On the Motion Panel set the Position Threshold to 0.
5. On the Hierarchy panel > IK > Sliding Joints rollout, turn on all the sliding parameters.

6. Animate the character, the bones expand and compress with the motion of the character.

### To add a bone after Physique is applied using Reinitialize




This bone is used to animate the characters nose

1.  Turn on Figure mode.
2. Add a bone where it is needed.
3. Link the root node of the bone to the biped.
4.  On the Physique rollout click Reinitialize.
5. On the Physique Initialization dialog, click Initial Skeleton Pose and then click Include New Bones.  
Vertex Link Assignment turns on also.
6. Click Initialize.
7. Adjust envelopes in Sub-Object Envelope.
8. Turn off Figure mode.

If you want the end of the bone to follow the biped, select the bone and delete the bone end effector in the Motion Panel

### To add a bone after Physique is applied using Add (Add Bone)

1.  Turn on Figure mode.
2. Add a bone where it is needed.
3. Link the root node of the bone to the biped.
4. On the Physique Bones rollout click Add.
5. In the viewports click a bone.  
Repeat until all bones are added.
6. Adjust envelopes.
7. Turn off Figure mode.

## Merging and Cloning a Biped Character

At some point you will need to use the 3DS MAX Merge command to merge a character into your scene; this is also the way to clone a biped and a mesh with the Physique modifier applied.

Note: For the biped currently in the scene, change the biped name on the Structure rollout if the file to be merged has a biped with the same name.

For a tutorial that teaches you to merge and clone a biped character, see *Merging and Cloning Characters* (see page 442).

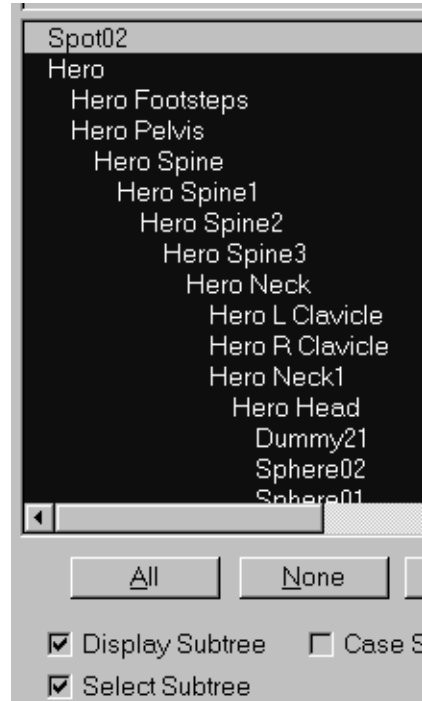
### Procedure

#### To merge a skinned biped

1. Choose File > Merge from the 3DS MAX menu, then select a .MAX file.  
The Merge dialog displays.
2. Click to turn on Select Subtree in the Merge dialog.
3. Locate and click the center of mass name in the Merge dialog list.  
The default name for the biped center of mass is *Bip01*. If the biped was renamed on

the Structure rollout, find the renamed center of mass in the list.

Note: The center of mass is the root object in the biped hierarchy, if this is selected with Select Subtree turned on, then all the child links are selected including extra bones and the mesh skin (the mesh skin is linked when Attach to Node is used in Physique).

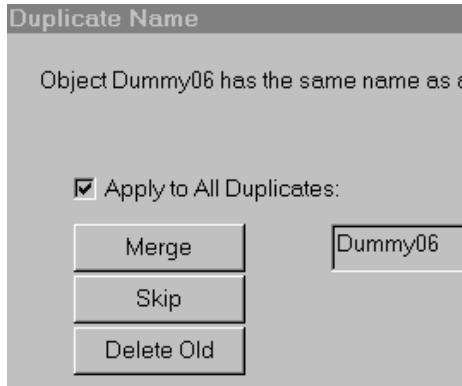


The center of mass object is named *Hero*, in the image of the Merge dialog. With Select Subtree active, all the children are also selected when you click *Hero*, including the Physique mesh.

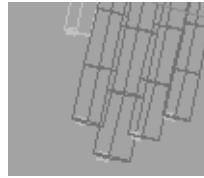
4. On the Duplicate Name dialog, select Apply To All Duplicates, and select Merge.

Even though the characters in your scene may have different names than the character you are merging, the biped finger/

toe/head dummies in the file to be merged may have the same names as other biped dummies in your scene. This step merges these dummy objects, dummies with duplicate names appear in Select by Name dialogs; this is not a problem.



5. On the Display rollout, toggle Objects to hide the biped finger-toe-head dummy objects.  
Normally the biped finger-toe-head dummies are hidden from sight, they are used by Physique to create envelopes for all the finger tips, toe tips, and head; these dummies display when a character is merged. The quickest way to hide them is to simply toggle the Objects button on the Display rollout.



In this image of the biped hand, dummies are visible as blue cubes. The number of biped dummies varies according to how many fingers and toes the character has.

6. The character animation is merged along with the character.

## Clones

If the merged character is a clone of the character in your scene, and both of their motions are identical, they will overlap. On the General rollout, turn on Footstep mode, select all the footsteps for one of the characters, and move them to separate the characters.

Use the Merge procedure to clone a character. Change the name of one of the clones on the Structure rollout before using File > Merge.

## Main Menu

### Preset Set Keys & Convert Bips

Customize Menu > Load Custom UI > CS3Tools.cui

After loading the Preset Set Keys custom UI you can record parameters and then use six set key buttons to apply the recorded parameters. The preset keys then provide one touch access to 6 key frame settings you use most when animating a biped. Only Key Info attributes of a Biped's motion are stored. Transformations are not stored. You can for example create a set key that has IK Blend set to 1 with Object space turned on and the interpolation tension set high.

Convert Bips allows you to preview or to convert many old *.bip* files from previous releases of the software into the current release. Conversion is done by simply reading and then writing each file. This process is identical to opening and saving Biped files through the standard UI.

## Procedure

### Using Custom Keys

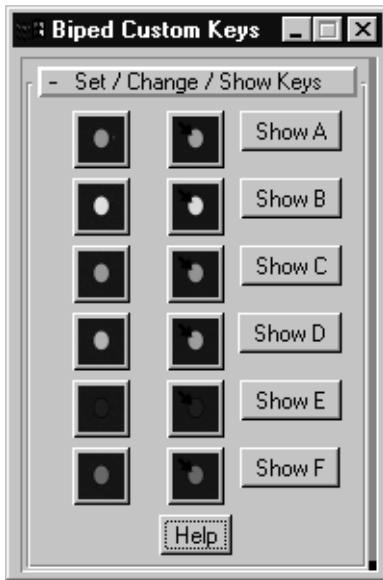
1. On the Customize menu click Load Custom UI.
2. Choose *CS3Tools.cui*  
Twelve colored buttons appear, six for recording parameters and six that apply the stored parameters.

3. Select a single biped bone at a frame where a key exists and change parameters in the Key Info and IK Key Info rollouts.
4. Click one of the six record icons.  
The record icons are circles with arrows. All parameters except position and rotation will be stored.
5. Scrub to another key and click the matching set key button to apply the stored key parameters.  
Optionally hold down the SHIFT key when pressing the set key button to set only the selected keys on the selected biped bones.

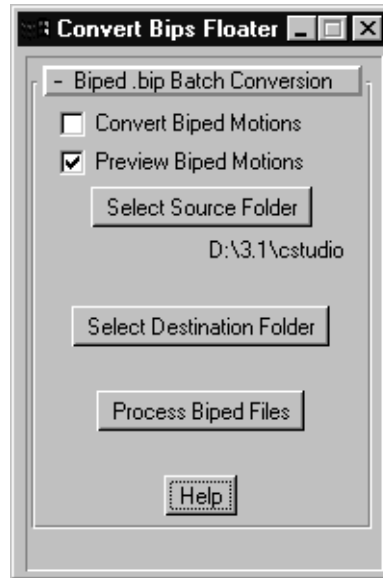
## Interface



**Record/Set Keys.** Icons with arrows record biped parameters in the Key Info and IK Key Info rollouts. The other icons apply the recorded parameters to the biped. By selecting multiple keys in the Track Bar and holding down the shift key and then clicking the Set Key button you can paste parameters onto selected multiple keys. Transformations are not recorded.



**Launch Preset Floater.** Displays the preset floater. This is functionally identical to the Record/Set Keys control described above. The Show Presets show you the recorded parameters.



#### Convert Bips Floater

**Convert Biped Motions.** Turn on to convert *.bip* files.

**Preview Biped Motions.** Turn on to preview biped motions.

**Select Source Folder.** Browse for a folder with *.bip* files.

**Select Destination Folder.** Browse for a destination folder



**Process Biped Files.** Starts the conversion or preview process.

## Create Panel

The first step in creating a character is to use controls on the Create panel to *create a biped* (see page 143). Once you've created a biped, open the Motion panel with the biped selected to display Biped controls, which allow you to animate the biped, load and save biped files, and to fit the biped to a mesh.

### Procedure

#### To create a biped

-  Open the Create panel.
-  On the Create panel, click Systems.
- Click the Biped button.
- Click and drag in a viewport.  
A blue bounding box appears, and after a brief pause, Biped displays a standing biped figure.
- Go to the Motion panel to display all the Biped controls.  
Some part of the biped must be selected for Biped controls to display on the Motion panel.  
If you are using **character studio** for the first time, you'll see the *Authorization dialog* (see page 146). You'll need to authorize your copy of **character studio** before continuing.

### Interface

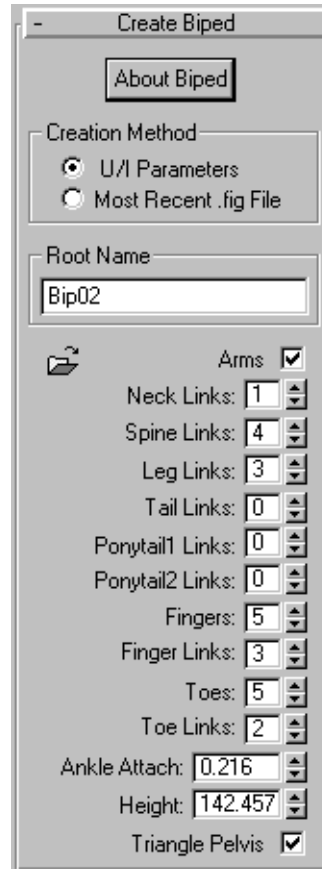
Biped

**Create Biped.** Lets you create a biped by dragging out a rectangle in any viewport. A biped is created to match the height of the rectangle.

After initially creating a biped, you can adjust the height and structure parameters with the

controls on the Create Biped rollout. Biped height and structure can also be altered, at any time, by turning on Figure mode and adjusting parameters on the Structure rollout on the Motion panel.

### Create Biped rollout



**About Biped.** Displays the About Biped dialog, showing version, copyright, and creator information.

## Creation Method Group

**UVI Parameters.** Uses the displayed parameters to create the biped. Click and drag in a viewport to create the biped.

**Most Recent .fig File.** Creates a biped, which has the proportion and structure of the most recently loaded figure file but the height of the dragged box. Click and drag in a viewport to create a biped.

If no figure file has been loaded since MAX was started, the software will look in the *biped.ini* file. This file contains the line:

FigureFile=c:\3dsmax\cstudio\default.fig.

This is the default value in the *.ini* file, but each time you load a new figure file, either from the Create or the Modify panel, the *.ini* file gets rewritten with that figure file's name. If the figure file listed in the *.ini* file is not found a biped of a height of 100 and all the parameters currently in the user interface will be created.

**Root Name.** Displays the name of the biped center of mass object. The center of mass (COM) is the root or parent of the biped hierarchy, and is visible as an octagonal object in the biped pelvis area. The Root Name is appended to all the links of the biped hierarchy.

To simplify merging a character, or to simplify selecting biped links in the Select by Name dialog on the 3DS MAX toolbar, change the default center of mass name *Bip01* to the name your character will use. If you change the Root Name from *Bip01* to *John*, for example, then all of the links will have the name *John* appended to them: *John Pelvis*, *John L Thigh*, and so on. You can also change the Root Name on the Structure rollout on the Motion panel.

*Bip01* is the default name given to the center of mass object of the first biped created. If multiple

bipeds are created, the Root Name increments and they will become *Bip02*, *Bip03* and so on.

**Load .fig File.** Load a figure file.

**Arms.** Specifies whether or not arms will be generated for the current biped. Turn this off to create a biped without arms.

**Neck Links.** Specifies the number of links in the biped neck. Range=1 to 5.

**Spine Links.** Specifies the number of links in the biped spine. Range=1 to 5.

**Leg Links.** Specifies the number of links in the biped legs. Range=3 to 4.

**Tail Links.** Specifies the number of links in the biped tail. A value of 0 specifies no tail. Range=0 to 5.

**Ponytail1 / 2 Links.** Specifies the number of Ponytail links. Range=0 to 5.

Animate hair with ponytail links. Ponytails are linked to a character's head and can be used to animate other appendages. Reposition Ponytails in Figure mode and use them to animate a character's jaw, ears, and nose. Use Ponytails to animate any appendage that should move with the head.

Note: Ponytails can only be keyed using rotations, they will not respond to move transformations.

**Fingers.** Specifies the number of biped fingers. Range=0 to 5.

**Finger Links.** Sets the number of links per finger. Range=1 to 3.

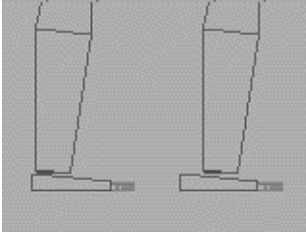
**Toes.** Specifies the number of biped toes. Range=1 to 5.

**Toe Links.** Specifies the number of links per toe. Range=1 to 3.

**Tip:** For characters wearing shoes you only need one toe with one link.



**Ankle Attach.** Specifies the right and left ankles' point of attachment along the corresponding foot block. The ankles may be placed anywhere along the center line of the foot block from the heel to the toe.



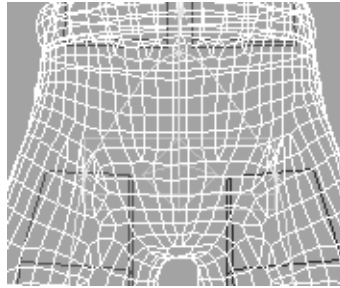
#### Ankle Attach Adjusted

A value of 0 places the ankle attachment point at the heel. A value of 1 places the ankle attachment point at the toes. Click the spinner up arrow to move the ankle attach toward the toes. Range=0 to 1.

**Height.** Sets the height of the current biped. The biped feet stay in the same place as the biped changes height.

**Triangle Pelvis.** Select this control to create links from the upper legs to the lowest biped spine object when Physique is applied. Normally, the legs are linked to the biped pelvis object.

The pelvis area can be a problem when the mesh is deformed with Physique. Triangle Pelvis creates a more natural spline for mesh deformation.



#### Triangle Pelvis

Triangle Pelvis creates two links that extend from the legs to the lowest biped spine object. A link from the biped pelvis to the lowest spine object is also created. This provides natural deformation to this area once Physique is applied and the character is moving. If you are working on a new character, select Triangle Pelvis before applying Physique. If Bones is selected (on the Display rollout on the Motion panel), links from the legs to the lower spine object are visible.

## Authorization

Unlike 3D Studio MAX, which provides a grace period of 30 days, you must authorize **character studio** in order to use it.

The first time you activate any of **character studio's** tools, such as Biped, Crowd or Physique a dialog box will appear asking you to enter your authorization code.

Also displayed in the dialog is your application ID number. You will use this number to obtain your authorization code.

## Internet Registration

You can obtain your authorization code by going to the web site: <https://register.autodesk.com> and selecting 3D Studio MAX from the product list and your country from the country list. Be sure to add the "s" in https. Follow the instructions on the pages that follow.

## Phone or Email Registration

In most worldwide locations, you can also call or email your application ID number to obtain the authorization code. Follow the instructions found in your product box to obtain your local phone number to call.

Note: The authorization code is specific to your 3DS MAX 3.1 hardware lock. If you change to a different hardware lock, you will need a different authorization code. It's a good idea, when you receive your authorization code, to write it on a piece of paper and tape it to your hardware lock. If for any reason you need to reinstall the software, you'll have the code readily available.

## Motion Panel

Once you have created a biped, use the Biped controls on the Motion panel to animate the biped, load and save Biped files, and fit the biped to a mesh representing your character.

Biped controls only appear on the Motion panel if any part of a biped is selected in the viewports. *Create a biped (see page 143)* if one does not exist. The Motion panel comprises the following rollouts:

*General Rollout (see page 147)*

*Track Selection Rollout (see page 160)*

*Key Info Rollout (see page 161)*

*IK Key Info Rollout (see page 167)*

*Keyframing Rollout (see page 173)*

*Display Rollout (see page 176)*

*Layers Rollout (see page 178)*

*Motion Capture Rollout (see page 186)*

*Animation Properties (see page 181)*

*Structure Rollout (see page 184)*

## Procedure

### To create a Named Selection for the biped center of mass

1. Select the biped center of mass (diamond shaped object in the pelvis) in the viewports.
2. Type a name in the Named Selection Sets field on the 3DS MAX toolbar.
3. Click ENTER.

Choose this selection if the biped is hidden and you want to view the biped and edit biped parameters on the Motion panel.

**Tip:** Give the center of mass selection the same name as your character in the Named Selection Sets field, "John," for example.

With multiple characters, you can quickly select the character you want to edit by choosing the character's name in the Named Selection Sets dropdown.

## General Rollout

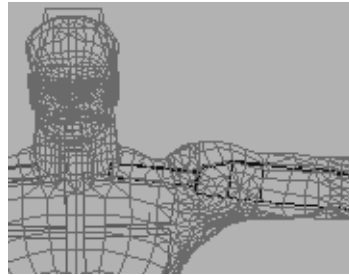
Select the Bipeds > Motion Panel > General rollout

Use controls on the General rollout on the Motion panel to put the biped into Figure, Footstep, or Motion Flow modes, and to load and save *.bip*, *.stp*, and *.fig* files. You'll find other controls on the General rollout, as well. For example, the General rollout provides controls for converting footsteps into a freeform animation, or a freeform into a footstep animation, and for turning on In Place mode.


### Procedures

#### To change the biped name (center of mass name)

1. Select a biped.
2. On the Structure rollout, enter a name in the Root Name field at the top of the rollout. This actually renames the biped center of mass object and appends the name to the rest of the biped links.  
Naming a biped is important when there are multiple bipeds in a scene, or if a biped is to be merged into a scene.



#### To create and fit a biped to a mesh for use with Physique in Figure mode

1. Create a character (3D geometry) in a reference pose: arms out, legs parted.
2. Create an appropriate biped skeleton.
3.  Turn on Figure mode and set parameters on the Structure rollout to suit your character.  
Keep in mind the number of finger, toe and leg links you will need.
4. Align the biped to your model in Figure mode using Height on the Structure rollout, Rubber Band mode to position the knees and elbows and Non-Uniform Scale on the 3DS MAX toolbar.

Keep the lowest spine object of the biped at the mesh belt-line and use Non-Uniform Scale on the biped finger objects to extend them slightly beyond the mesh fingers.

**Tip:** Scale the biped limbs to approximate the dimensions of the mesh. Envelopes created by Physique will be sized based on the scale of the biped objects if the Object Bounding Box option is used on the Physique Initialization dialog.

5. Select the mesh and add the Physique modifier on the Modify panel.



6. On the Physique rollout, first turn on Attach to Node and then select the biped pelvis.
7. Click Initialize on the Physique Initialization dialog.
8. Physique creates links and associated envelopes.
9. Fine-tune the Envelopes using the Envelope sub-object controls.
10. In Envelope Sub-Object, turn off Deformable and turn on Rigid for the head link. This will prevent distortion of the face and skull.

**Tip 1:** If your model consists of more than one object, select all the objects and *then* apply the Physique modifier.

**Tip 2:** If you want to link eyeballs to the biped head, do so *after* you apply the Physique modifier to your model to prevent Physique from creating extra links to the eyeballs.

**Tip 3:** Envelopes for finger links are bound to influence vertices on adjacent fingers. In Vertex Sub-Object, exclude vertices from influence by a particular link by first selecting them, turning on Remove from Link, and then selecting links in the viewports. The envelopes of the selected links will not influence the selected vertices. Click Lock Assignments in Vertex Sub Object, otherwise any modification to the finger envelopes will re-assign all vertices within the outer bound of an envelope to that envelopes link.

### To link a mechanical character to the biped (without Physique)

With mechanical or jointed characters, you can simply link objects to the biped without using the Physique modifier.

1. Load or create a mechanical or jointed model; body parts should be separate 3DS MAX objects.
2. Create a biped.



3. On the General rollout, click Figure mode, then position and fit the biped to your mesh objects.

4. Go through the mesh objects and use Link on the 3DS MAX toolbar to link each object to its corresponding part on the biped.

All of the keyframe animation applied to the biped will also animate your model.

If 3DS MAX Linking and Physique are used together, link the body parts to the biped after applying Physique. Physique will not create superfluous links to the mechanical body parts if they are not linked to the biped when Physique is applied.

### To fit biped legs to the skin



1. Turn on Figure mode on the General rollout.
2. Choose Non-Uniform Scale, turn on the Local Z axis constraint, and scale the biped's pelvis so the biped's leg links are centered in each leg of the skin.
3. Select the biped's two thigh links (LLeg and RLeg), turn on the Local X axis constraint, and scale the thigh links so they end at the knees of the skin.
4. Select the biped's two lower leg links (LLeg1 and RLeg1), leave Local X as the axis constraint, and scale the lower leg links so

the biped's ankles are level with the ankles of the skin.

**Tip:** When the two upper legs are selected, you can press PAGE DOWN to select the lower legs.


5. In a left or right viewport, scale the biped's feet so their profile roughly matches the profile of the feet of the skin.
6. Scale toes or move them along their local X axis so each toe is aligned with the corresponding toe in the skin.

The ends of the final toe links should go through the tips of the skin's toes.

You might have to change the number of biped toes to match the number of skin toes. A biped must have at least one toe on each foot. If the skin has no toes or the character is wearing footwear, the position and number of biped toes doesn't matter—but they should still extend beyond the skin.

**Tip:** During the fitting process, try Freezing the skin object to prevent accidental selection.

### To fit the spine to the skin torso

1.  Turn on Figure mode on the General rollout.
2. Select the lowest link of the spine (Bip01 Spine). Choose the Move transform, constrain movement to the spine link's local X axis, and move it vertically to the waist of the skin, just below the navel.
3. Non-uniformly scale the other spine links in their local X axis so they fit the upper part of the skin's torso.

The neck link should begin where the skin's neck begins.


If the torso of the skin curves, you should also rotate spine links about their local

Z-axis, to align the spine with the longitudinal center of the torso.

4. Non-uniformly scale the biped's neck in its local X axis to match the length of the neck of the skin. The top of the last neck link (also the base of the head link) should be where you want the head to pivot. This is usually just below the ears, centered with the spine.

Leave the head in its default position relative to the spine and neck links.

### To fit biped arms to the skin

1.  On the General rollout turn on Figure mode.
2. Rotate one upper arm (R Arm1 or L Arm1) in its local Y axis to center its link in the upper arm of the skin.
3. Non-uniformly scale the upper arm so its link ends at the elbow of the skin.
4. Non-uniformly scale the lower arm (RArm2 or L Arm2) so its link ends at the wrist of the skin.


If the skin's arms are bent, rotate the lower arm to center its link as well.

5. Scale fingers, or move them along their local X axis so each finger is aligned with the corresponding finger of the skin.


The ends of the final finger links should go through the tips of the skin's fingers. You might have to change the number of biped fingers to match the number of skin fingers.

6. When the arm is completely fitted to the skin, select all of it and click Copy Posture.
7. Click Paste Posture Opposite to pose the opposite arm.



**To scale a biped and a mesh with the Physique modifier applied**

1.  Turn on Figure mode.
2. Select the biped and change the Height parameter on the Structure rollout.

**To use In Place mode to adjust keyframes**



1. Select a biped that has animation.
2.  Click In Place mode on the General rollout.
3. Turn on the Animate button.
4. Find a frame where the biped needs adjustment and modify or add keyframes.

**To save a biped Figure file**

1. Select a biped.
2.  Turn on Figure mode.
3.  Click Save File.
4. Enter a name for the figure file, then click OK.



**Tip:** The figure file can be reloaded if you accidentally move the biped in Figure mode after it is fitted to your character.

**To load a biped Figure file**



1. Select a biped.
2.  Turn on Figure mode.
3.  Click Load File.
4. Choose a figure file (*.fig*) to load, and then click OK.

**Warning:** Loading a figure file will replace the selected biped's pose and structure.

**To use Rubber Band to position the elbows and knees**


1.  On the General rollout, turn on Figure mode.
2. Select an arm or leg on the biped.
3. Turn on the Move transform.  
Use Local coordinates with an X axis constraint.
4.  Turn on Rubber Band mode.  
This button is unavailable if you are not in Figure mode or if a part of the biped other than the upper or lower arm or leg or the center of mass is selected.
5. Move the selected arm or leg link.  
As you move the arm or leg link, the hands and feet are stationary as the knees or elbows are positioned.

**To adjust the biped center of mass with Rubber Band**


1.  Turn on Figure mode.
2. Select the center of mass object (diamond shape) on the biped.
3. Turn on the Move transform.
4.  Turn on Rubber Band mode.  
This button is unavailable if you are not in Figure mode or if a part of the biped other than the upper or lower arm or leg or if the center of mass is selected.
5. Move the center of mass.  
While Rubber Band mode is active, the center of mass moves independently from the rest of the biped, which remains in the same position.

**Tip:** Another way to adjust the biped's balance is to change the value of Balance Factor. Balance Factor displays on the Key Info rollout when a Body Horizontal key is current. Balance Factor can be keyframed.


#### To save biped motion file (.bip)

1. Select a biped.
2.  Click Save File on the General rollout.
3. Enter a name in the dialog, then click OK.  
**Tip:** Optimize raw motion capture data and save the motion in a .bip file.

#### To load a biped motion file (.bip)


1. Make sure Figure mode on the General rollout is inactive and select a biped.
2.  On the General rollout Click Load File.
3. Choose a .bip motion file, and then click OK.  
Note: If the .bip file is a raw motion capture file, use Load Motion Capture File on the Motion Capture rollout.

#### To save footstep data

1. Make sure Figure mode is inactive, then select a biped.
2.  On the General rollout, click Save File.
3. Choose Step Files (.stp) in the Files of Type dropdown.
4. Enter a name for the footstep file, and then click OK.  
Note: This only saves footstep data, not biped limb keys. Save a .bip file to save all footsteps and biped limb keys.

#### To load footstep data


1. Select a biped, and make sure Figure mode is inactive.

2.  Click Load File.
3. Choose Step Files (.stp) as the file type to load.
4. Choose a footstep file, and then click OK.  
New default keys will be generated to match the footsteps.


#### To preview biped motion using 3DS MAX

- Drag the Time slider or click Play.


#### To preview biped motion using Biped controls

1. Activate a viewport.
2.  Click Biped Playback on the General rollout.  
The preview plays back repeatedly, showing white biped skeletons against a gray background.
3. Click Biped Playback again to end the preview.

#### To edit the footstep buffer before you insert it by splicing


1.  Click Buffer mode on the General rollout.  
This button is active only when there are footsteps in the buffer. To place footsteps in the buffer, turn on Footstep mode, select some footsteps, and click Copy Footsteps in the Footstep Operations rollout.  
The viewports now display the footsteps in the footstep buffer, rather than the footsteps in the currently activated footstep sequence.
2. Edit footsteps or motion keys as you normally would. You can't, however, splice (copy and paste) footsteps while in Buffer mode.  
Biped automatically updates the footstep buffer to reflect all key and footstep editing.

If you load a biped (.bip) motion file by clicking Load File on the General rollout while Buffer mode is active, Biped replaces the footsteps and other motion in the footstep buffer, with the motion in the biped file.

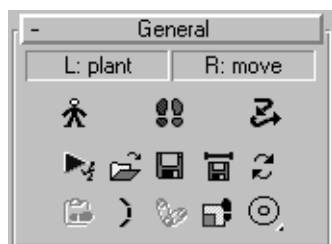
3.  Click to turn off Buffer mode.

Now you can splice the edited buffer by using Paste Footsteps on the Footstep Operations rollout.

### To turn off Scale Stride mode

-  On the General rollout, click Scale Stride mode to turn it off.
- By default footstep stride length and width are scaled to match the stride length and width of the biped figure. The button changes to indicate stride scaling has been turned off.

## Interface



### Foot States

**Display.** Displays the state of the left and right biped feet at the current frame in a footstep animation. Foot states change as the biped foot moves from being in contact with a footstep to being airborne. An asterisk appearing after the Foot State indicates that the foot is selected. At

any frame, a biped foot may be in one of the following four states:

- Plant.** The biped foot state in full contact with the footstep.
- Lift.** The biped foot state just before leaving a footstep.
- Move.** the biped foot state between footsteps; it is an airborne period.
- Touch.** The biped foot state at which a biped foot first contacts a footstep.

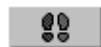
**Tip:** Set Multiple Keys on the *Keyframing* rollout (see page 173) can select keys according to foot states. Selecting periodic keys (for example, all of the foot Move State keys) and incrementing all of them is the key to perfecting rhythmic motions, such as running.



**Figure mode.** Use *Figure mode* (see page 156) to fit a biped to the mesh or mesh objects representing your character. Leave Figure mode on when you attach the mesh to the biped with Physique. Figure mode is also used to scale a biped with a mesh attached, to make biped “fit” adjustments after Physique is applied, and to correct posture in motion files that need a global posture change.

When Figure mode is turned on, the biped jumps from its animated position to its Figure mode pose. Animation is preserved when you exit Figure mode.

All of the parameters on the Structure rollout are active in Figure mode.



**Footstep Mode.** Create and edit footsteps; generate a walk, run, or jump footstep pattern; edit selected footsteps in space; and append footsteps using parameters available in Footstep mode.



Two additional rollouts display on the Motion panel when *Footstep mode* (see page 247) is active:

- *Footstep Creation rollout* (see page 248)
- *Footstep Operations rollout* (see page 258)



**Motion Flow Mode.** Create scripts and use editable transitions to combine *.bip* files together to create character animation in *Motion Flow mode* (see page 227). After creating a script and editing transitions, use Save Segment on the General rollout to store a script as one long *.bip* file. Save a *.mfe* file, this enables you to continue Motion Flow work in progress.

**Tip:** Use Motion Flow mode to cut motion capture files together.

Two additional rollouts display when Motion Flow mode is active:

- *Motion Flow rollout* (see page 229)
- *Motion Flow Script rollout* (see page 234)

Note: Some **character studio** controls are disabled when Motion Flow mode is active.



**Biped Playback.** Plays the animation for all bipeds unless they are excluded on the Display Preferences dialog. This playback mode usually gives real-time playback, which you may not get if you use Play on the 3DS MAX animation toolbar.

In Biped Playback mode, the biped is displayed as bones only, with no other scene objects visible.



**Load File.** Displays a standard Open dialog to load *.bip*, *.fig*, and *.stp* files.

**Restructure biped to match file (.bip files only).** Turn on to change the biped structure to match the structure stored in the *.bip* file. The file loads with the stored biped structure. This is an option in the Load File dialog.

This option is unavailable when you load a *.bip* file into a clip or onto a biped that is in Edit Clip mode because all the clips in the motion flow would have to be adapted.

**Set lowest starting foot height to Z=0 (.bip files only).** Sets the lowest starting foot height to Z=0. This is an option in the Load File dialog. Default=On.

In **character studio 3** the height of a motion clip can be retained. This is important if you want to retain the height of a motion clip for motions adapted to characters of different sizes. If, for example, the character is jumping of a rock and you want to retain the Z position of the character, you would turn this option off. Leave this option off if Motion Flow motions must be blended that begin and end at different heights, such as three clips that have the character mounting a bicycle, riding the bicycle, and dismounting the bicycle.

Turning off this option can, however, cause a jump in the motion during motion flow transitions. Turn this on for smooth transitions in Motion Flow mode. If adaptation takes place, the height is set so that the lowest foot at frame 0 starts at the Z=0 height. This lines up clips along the Z axis and creates smooth transitions.

Note: Use Load Motion Capture File on the *Motion Capture rollout* (see page 186) to load the raw version of the motion capture *.bip* files that come bundled with **character studio**. These files have no footsteps and keys at every frame. Loading files using Load Motion Capture File allows you to filter the data and extract footsteps.

- **Biped file (.bip).** Load a biped motion file. Motion files include, footsteps, keyframe settings, the biped scale, and the active gravity value (GravAccel). IK Blend values for the keys

and Object Space Object information are also loaded.

- **Figure file (.fig).** Load a Figure file. Figure mode must be active to load a Figure file. Figure files allow you to apply the structure of one biped to another. Reload a Figure file if you accidentally lose your biped Figure mode pose; this pose is the biped fitted to a mesh.
- **Step file (.stp).** Load footsteps without body keyframes. Using this ASCII file format enables developers to write programs that generate step files for biped motion. Biped will generate body keys for the loaded steps. See *stp.rtf* in the *cstudio\docs* directory for more details on the step file format.



**Save File.** Save Biped files. You can save Biped files (.bip), figure files (.fig), and step files (.stp) files.

- **Biped file (.bip).** Save a biped motion file. A .bip file includes footsteps and keyframe data. Biped files store the complete movement and allow you to create libraries of motion. Create your own .bip library by animating the biped and saving a .bip file.
- **Figure file (.fig).** Save the structure and position of a biped in Figure mode. After fitting the biped to a mesh in Figure mode, save a figure file. If the biped is accidentally moved in Figure mode, reload this file.
- **Step file (.stp).** Save footstep timing and location data in an ASCII format. Step files contain no body keys (spine, arms, etc.).



**Save Segment.** Save a segment of your animation as a .bip file or save a Script from Motion Flow mode as a .bip file. Displays a *modified Save As dialog* (see page 241).



**Convert.** Convert a footstep animation to a freeform animation. This works in both directions. Displays the *Convert to Freeform dialog* (see page 158) or *Convert to Footsteps dialog* (see page 158) depending on the direction.

- Convert uses biped foot IK Blend values to extract footsteps.
- Use Convert to extract footsteps from an animation saved using Save Segment in Motion Flow mode.
- Convert the animation in either direction depending on how you like to work. Convert to freeform for unrestricted key editing. Convert to footsteps to take advantage of footsteps.



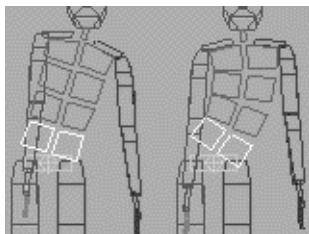
**Buffer Mode.** Edit segments of an animation in Buffer mode. Copy footsteps and associated biped keys into the buffer using Copy Footsteps on the Footstep Operation rollout first, then turn on Buffer mode to view and edit the copied segment of your animation.

**Tip:** Paste buffered motion back to the original animation repeatedly to create looping motions.

Edit footstep and biped animation that have been copied into the buffer using Copy Footsteps on the Footsteps Operation rollout. The changes can be pasted back by turning off Buffer Mode, turning on Paste Footsteps on the Footstep Operation rollout and overlapping the buffered footsteps with the original footsteps. The buffered motion is spliced into the original animation.



**Bend Links Mode.** Bend all the biped spine objects naturally by rotating a biped spine object. Bend Links also works for the biped tail and ponytail links.



**Bend Links inactive (left) and Bend Links active (right)**



**Rubber Band Mode.** Use this to reposition the biped elbows and knees without moving the biped hands or feet in Figure mode. Reposition the biped center of mass to simulate the physics of wind or weight pushing against the biped. Figure mode must be turned on to enable Rubber Band Mode.

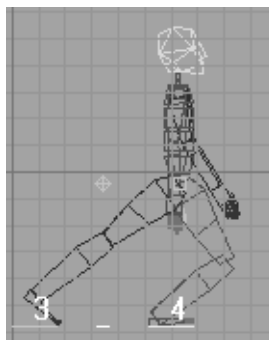
To reposition biped knees and elbows, turn on Figure mode and turn on Rubber Band mode. Select the move transform tool, then select and drag a biped upperarm or thigh in the viewports. Use this as an aid to fitting a biped to a mesh.

To reposition the biped center of mass, turn on Figure mode and turn on Rubber Band mode. Select the Move transform tool, choose an axis on the 3DS MAX toolbar, then select and drag the center of mass in the viewports. Use this to account for wind force or pushing against a heavy object.

Use Rubber Band mode in Calibrate Figure mode on the Motion Capture rollout to position the biped knees and elbows to marker positions when a marker file is loaded.

Note: Rubber Band mode behaves differently than Non-Uniform Scale. For example, if you “Rubber-Band” the biped thigh, the thigh and biped calf objects scale proportionally to keep the biped foot stationary. Using Non-Uniform

Scale, the calf retains its scale and the foot moves.



**Moving the biped center of mass (blue diamond) behind the character, turns this default walk cycle into a struggle against a high wind.**



**Scale Stride Mode.** Footstep stride length and width are scaled to match the stride length and width of the biped figure. Scale Stride mode is on by default.



Displays when Scale Stride mode is *off*.

Scale Stride mode is on by default, so scaling occurs automatically when you load a *.bip*, *.stp*, or *.fig* file. Scaling occurs when you paste footsteps and when you scale the biped's legs or pelvis.

For example, if you load a *.bip* file that was saved from a larger biped, the footsteps come into your current scene scaled to match the selected smaller biped. If Scale Stride mode is off, the footsteps come into the current scene without being scaled down.

If you turn off Scale Stride mode and then go into Figure mode and scale the biped up or down, the footstep stride width and length remains the same when you exit Figure mode.



**In Place Mode.** Use In Place Mode to keep the biped visible in the viewports while the animation plays. Use this for biped key editing or adjusting envelopes with Physique. It prevents XY movement of the biped center of mass during animation playback; motion along the Z axis is preserved. This is a three-button fly-out. In Place mode is stored with the 3DS MAX file.



**In Place X Mode.** Lock center of mass X axis motion. Use this for game export where the character stays in place but the swinging motion of the hips and upper body along the Y axis is preserved.



**In Place Y Mode.** Lock center of mass Y axis motion. Use this for game export where the character stays in place but the swinging motion of the hips and upper body along the X axis is preserved.

Biped keys for limbs, footsteps, and center of mass can be adjusted using In Place mode. When the center of mass is moved on the XY-axes in this mode, the footsteps move. View biped playback without requiring a follow camera. In this viewing mode, visible footsteps “slide” under the biped.

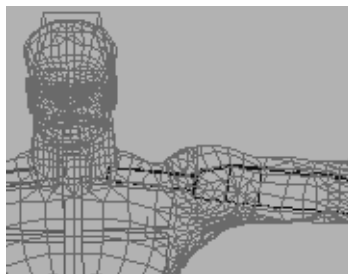
For export to games, this feature is valuable since many game engines intelligently move the character's center of mass laterally according to game play. In Place mode makes it easy to view, tune, and export animation in a manner that is complimentary to game engine playback.

**Tip:** Another way of following a moving character is to link a camera and camera target to the center of mass shadow, which is the disc between the biped's feet.

Note: Trajectories do not display when In Place mode is active.

## Figure Mode

Select the Biped > Motion Panel > General rollout > Figure mode



While Figure mode is active, you can change biped structure and fit that structure to a character mesh. It can be used for a variety of other procedures as well.

- **Figure mode is a reference** position to fit a biped to a mesh. Use Figure mode to fit a biped to the mesh representing your character. This “reference” or Figure mode position, in which the biped is aligned to the mesh, is necessary when a mesh is linked or attached to the biped with Physique. After the biped is positioned to fit within a mesh, leave Figure mode on during the process of attaching a mesh to the biped with Physique, or when using Link on the 3DS MAX toolbar to link the mesh objects of a character to the biped.

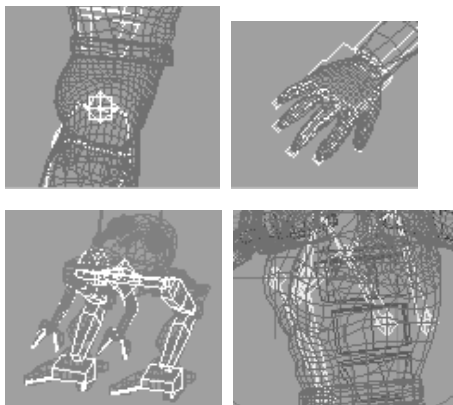
The relationship or “fit” position between the biped and the mesh can always be restored by turning on Figure mode, regardless of which motion file happens to be loaded. After fitting a biped to a mesh in Figure mode, use Save on the General rollout to save a figure file (.fig). If you

accidentally reposition the biped in Figure mode, load the figure file.

- **Figure mode is used for biped adjustment after a mesh is attached to correct biped joint location.** After using Physique to attach a mesh character to the biped, you may want to reposition a biped limb relative to the mesh. For example, if the biped shoulder joint is too far out relative to the mesh shoulder, then the Physique modifier must be inactivated, the biped limbs adjusted, and then a Reinitialize in Physique must be performed before reactivating the Physique modifier.
- **Figure mode is used for biped adjustment after a mesh is attached to correct posture in a motion file.** Figure mode is also used to make adjustments after a character is attached or linked to the biped. After loading a *.bip* motion file, for example, you may find that the character is hunched too far forward during the entire animation. Rotating the biped's spine objects in Figure mode will correct the character's posture for the entire animation. This is a basic procedure where you simply rotate the biped limbs in Figure mode and then exit Figure mode; the posture will be corrected for the entire animation.
- **Figure mode is used to define biped structure.** All the parameters on the Structure rollout are activated in Figure mode allowing you to tailor the biped to your mesh character. After creating a biped, make all of your biped structure changes on the Structure rollout. For example, you may want to use one toe with one toe link if your character is wearing shoes or if your character's toes do not need to be keyframed individually. Set the biped structure before "fitting" the biped to the your mesh character.
- **Turn Figure mode on to scale a character.** Use the height spinner on the Structure rollout to scale a complete character (a complete character has a biped and mesh attached with Physique).
- **Rename a Character.** Figure mode is not necessary for naming a character, but while you are editing the biped structure, you may want to name the biped. Naming a character is necessary when multiple bipeds are merged into one scene. Rename the character in the Name field on the Structure rollout. This renames the center of mass object and appends the name to all the links in the biped hierarchy. For example, renaming Bip01 to John will rename the right thigh object to John R Thigh.
- **Reverse-Knee Characters.** If your character mesh has reverse knees, rotate the biped calves or thighs along the local X axis 180 degrees in Figure mode; the biped local X axis is along the *length* of the limb. **character studio** assumes you want a reverse knees character if the calves or thighs are rotated past 90 degrees in the local X axis. When Figure mode is turned off, the biped walks, runs, and jumps with reverse knees.  
Note: See *Tutorial 7* (see page 543) for a step by-step procedure on fitting a biped to a mesh in Figure mode, and using Physique to attach a mesh to the biped.

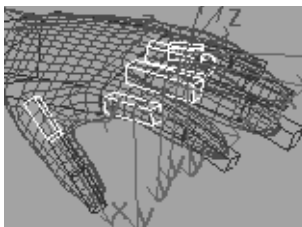
## Notes on Fitting the Biped to a Mesh in Figure Mode

These are quick notes designed to give you a general sense of the issues involved when a biped is fitted to a mesh.



- Use the Structure rollout to set the number of toes and fingers; specify the number of links per finger and toe. One toe with one toe link is often sufficient if your character wears shoes, or if animating individual toes will not be necessary.
- Put the lowest biped spine object at the character's belt-line.
- Non-uniform scale the biped fingers to slightly protrude from the character's hand.
- Rubber Band mode and Non-Uniform scale are used to size the biped limbs to fit the biped to a mesh.
- Use Link on the 3DS MAX toolbar to link non-deformable (mechanical) objects to the biped. Do this after Physique is applied to prevent Physique from generating extra links (Envelopes). Superfluous envelopes (links) can be turned off in Physique however, so this is not critical.
- Reposition and use Ponytails on the Structure rollout to animate a character's jaw, ears, hat, hair and ponytails.
- Objects like eyeballs and weapons should be linked to the biped after Physique is applied; otherwise links (Envelopes) will extend to these objects when Physique is applied.

- A saved .fig file can be reloaded if the biped is repositioned in Figure mode by mistake.



Move the first link on each finger to position the fingers relative to the mesh; use local and world coordinate systems for this. Use Non-Uniform Scale and scale the finger links to position the joints. After positioning the thumb, rotate the first thumb link around the local x axis until the local Z axis creates a natural rotation for the thumb (refer to the image). A User view and toggling back and forth between a shaded and wireframe display is helpful when fingers are positioned.

---

## Convert to Freeform or Footsteps Dialogs

Select the Biped > Motion panel > General rollout > Convert

When you click Convert on the General rollout of the Motion panel, a Convert To dialog displays: *Convert to Freeform* or *Convert to Footsteps*, depending on the animation method of the currently loaded motion. Use Convert to Freeform for unrestricted key editing or Convert to Footsteps to take advantage of footsteps.

Note: By default, Jumping or airborne periods between footsteps are calculated by **character studio** based on gravity strength (GravAccel) and time between footsteps. Biped elevation during these jumping and airborne periods will be lost when converting to freeform from footsteps

(unless you create a vertical COM key in the airborne period in the footstep animation), a freeform animation uses spline interpolation for the center of mass position and does not account for gravity. Elevation in airborne periods is restored if you convert back to footsteps.

## When to Use Convert

- Use Convert after using Save Segment in Motion Flow Mode to save a script as a *.bip* file. Exit Motion Flow Mode, load the *.bip* file and click Convert to extract footsteps. Save Segment applies IK Blend values of 1 to the biped foot keys for the keys at footsteps. If a freeform animation is part of the script in Motion Flow mode, and you want to convert the entire script motion to footsteps, then use the Save Keys at every Frame option in Save Segment and use Load Motion Capture File on the Motion Capture rollout. This will allow you to extract footsteps using the proximity method.
- Use Convert if you are working on a footstep animation and want to switch to a freeform animation.
- Use Convert if you have started a freeform animation and want to convert to a footstep animation. In order to convert a freeform animation to a footstep animation, the file must be properly set up by locking the feet to world space using IK Blend before converting to footsteps. When creating freeform animations, you should set your leg keys to IK Blend=1.0 during the periods you want the feet to be planted, and set the move keys to IK Blend=0.0. This will insure that the feet are locked and rid of unnecessary foot sliding as the body is moved. When converting, if the leg keys have been set up this way, Biped will extract

footsteps during any duration where IK Blend=1.0.

## Interface

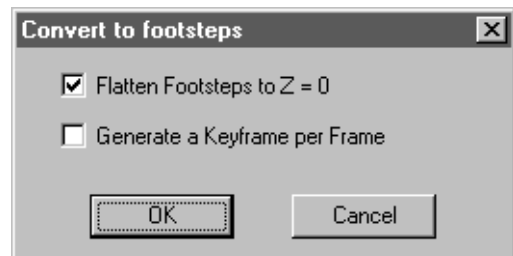
### Convert to Freeform dialog



**Generate a keyframe per frame.** Creates keys at every frame.

Note: Converting footsteps to freeform creates foot IK Blend values of 1 for the biped feet for the original footstep keys. This simplifies keyframing by putting the feet into world coordinate space, which prevents them from sliding when the biped is moved. These foot IK Blend values are also used when you click Convert (on the General rollout) to convert back to footsteps.

### Convert to Footsteps dialog



**Generate a key per frame.** Creates a key at every frame, and extracts footsteps based on foot IK Blend values equal to 1. Save Segment in Motion Flow mode stores the active script as a *.bip* file without footsteps. The biped foot keys are assigned IK Blend values of 1 for the original

footstep keys. After loading a *.bip* file saved using Save Segment in Motion Flow mode, use Convert (on the General rollout) to extract footsteps.

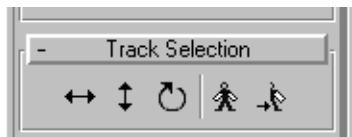
Convert is also useful if you have already converted from footsteps to freeform in which case the feet are assigned an IK Blend value of 1 for the keys that represented footsteps.

**Flatten Footsteps to Z=0.** Flatten the footsteps to Z=0. The entire biped is repositioned to place the footsteps at Z=0.

Note: An IK Blend value of 1 for the feet puts them into world coordinate space and prevents them from slipping while setting biped keys in a freeform animation.

## Track Selection Rollout

Select the Biped > Motion panel > Track Selection rollout



Use the controls on the Track Selection rollout to select one of the three center of mass tracks for editing and to select biped limbs.

### Procedures

#### To locate vertical center of mass keys

1. On the Track Selection rollout, click Body Vertical.
2. Use the Next and Previous Key buttons on the Key Info rollout to jump to the next or previous center of mass Body Vertical keyframes.

**Tip:** If the selected vertical COM key is a touchdown key from an airborne period,

you can change the Ballistic Tension parameter on the Key Info rollout to control knee bend. Turn on Trajectories on the Display rollout to view how changing parameters affects the trajectory.

#### To use Symmetrical Tracks

1. Select any limb on the biped.
  2. On the Track Selection rollout, click Symmetrical Tracks.
- The selected biped limbs remain selected, opposite limbs are selected also.

#### To use Opposite Tracks

1. Select a limb or limbs on the biped.
  2. On the Track Selection rollout, click Opposite Tracks.
- Selected biped limbs are deselected; opposite biped limbs are selected.

#### To set a Body Vertical key

1. On the Track Selection rollout click Body Vertical.
2. Turn on the Animate button.
3. Click Select and Move on the 3DS MAX toolbar.
4. Click and drag the center of mass up or down in the viewports.

The biped is repositioned vertically, a key is created in the center of mass Body Vertical track.

#### To set a Body Horizontal key

1. On the Track Selection rollout, click Body Horizontal.
2. Turn on the Animate button.
3. Click Select and Move on the 3DS MAX toolbar.
4. Click and drag the center of mass laterally in the viewports.



The biped is then positioned laterally, a key is created in the center of mass Horizontal track.

#### To set a Body Rotation key

1. On the Track Selection rollout, click Body Rotation Track.
2. Turn on the Animate button.
3. Click Select and Rotate on the 3DS MAX toolbar.
4. Select an Axis to rotate around.
5. Click and drag the center of mass.

The biped then rotates around the selected axis, a key is created in the center of mass Body Rotation track.

#### Interface



#### Body Track Tools (Center of Mass Tracks)

Biped's center of mass (COM) object is the root of the biped hierarchy, and uses three animation tracks to position and rotate the biped: They are Horizontal, Vertical, and Rotation. The Horizontal and Vertical tracks also contain Biped Dynamics parameters.

Using these tools, select and edit one of the three center of mass tracks. Open the *Key Info rollout* (see page 161) to view Biped Dynamics parameters and to adjust Tension, Continuity, and Bias for the selected Body track.

**Tip:** Turn on Trajectories to view the way a parameter change affects the animation.

Note: Tension, Continuity, and Bias only affect the Center Of Mass Body Vertical keys if the value of Dynamics Blend is less than 1. To turn gravity off at a vertical center of mass key, set the value of Dynamics Blend to 0.

**Body Horizontal.** Selects the center of mass to edit horizontal biped motion.

Displays Balance Factor parameters on the *Key Info rollout* (see page 161), and can be animated.

**Body Vertical.** Selects the center of mass to edit vertical biped motion.

It displays Dynamics Blend and Ballistic Tension Parameters on the Key Info rollout. These can be animated. Ballistic Tension is only available at keys just before or just after an airborne state, as between footsteps in a run or jump.

**Body Rotation.** Selects the center of mass to edit biped rotational motion.



**Symmetrical Tracks.** Selects the matching object on the other side of the biped. For example, if the right arm is selected, clicking Symmetrical Tracks selects the left arm too. You can then make changes to both sides of the body at once. Symmetrical Tracks works for single and multiple biped objects.



**Opposite Tracks.** Selects the matching object on the other side of the biped, and deselects the current object. For example, if the right arm is selected, clicking Opposite Tracks selects the left arm. Opposite Tracks can be used for single or multiple objects.

## Key Info Rollout

Select the biped. > Motion panel > Key Info rollout

Tools in the Key Info rollout allow you to do the following:

- Find the next or previous key for the selected biped body part.
- Use the Time spinner to slide a key back and forth in time.

- Change Tension, Continuity, and Bias for a key and display trajectories.
- Adjust biped dynamics.

When the Vertical Center of Mass track is selected you can change the vertical dynamics of the motion, on a key-by-key basis. When the Horizontal Center of Mass track is selected you can change the balance factor for shifts in weight distribution.

## Activating Parameters

Groups of the Key Info rollout are unavailable depending on what part of the biped is selected and if a key is current. Body Vertical, Body Horizontal and Body Turning refers to the three tracks used to animate the biped center of mass. Select one of the three center of mass tracks on the Track Selection rollout, then use Next Key or Previous Key to find a key to edit.

- If Body Vertical is selected and a key is current, then parameters for Dynamics Blend, Ballistic Tension, Z Position, Time, and TCB parameters (Tension, Continuity and Bias) are active. Ballistic Tension is only available at keys just before or just after an airborne state, as between footsteps in a run or jump.

Note: TCB controls are not effective at Body Vertical keys just before and just after an airborne period, between footsteps, if Dynamics Blend=1. Biped Dynamics calculates the airborne trajectory, in this case lower the value of Dynamics Blend to use the TCB controls. In a walk sequence, where footsteps overlap, Dynamics Blend has no effect and TCB controls can be used.

- If Body Horizontal is selected and a key is current, the Balance Factor parameter, XY Position, Time, and TCB parameters are

active. Z Position, Dynamics Blend, Ballistic Tension are grayed.



- If Body Rotation is selected and a key is current, only the Time and TCB parameters are active.
- If a biped hand is selected and a key is current, then all parameters are active except parameters in the Body Dynamics group.
- If a biped foot key is selected and current, then all parameters are active except for parameters in Body Dynamics.
- If a biped leg is selected and a key is current, then Time and TCB parameters are active. XYZ Position and Body Dynamics parameters are made unavailable. In a footstep animation, Time is made unavailable at a Touch and Lift key.
- If a biped arm is selected and a key is current, then Time, and TCB parameters are active. XYZ Position and Body Dynamics parameters are made unavailable.

## Tension Continuity and Bias (TCB)

Rather than creating extra keys to fine-tune the motion of the biped limbs, you can use the TCB controls to adjust ease in, ease out, and limb trajectory on keys that already exist.

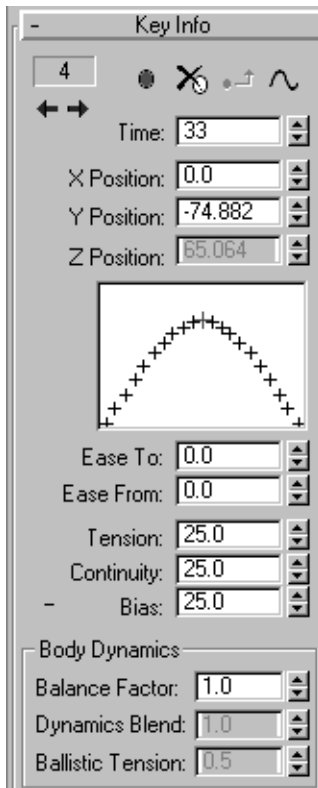
## Procedures


### To change TCB for a biped arm


1.  Turn on Trajectories.
2. Select a biped arm.
3.  Use Next or Previous key to locate an arm key.
4. Change the Tension, Continuity, and Bias spinners.

The trajectory changes to reflect the new parameters. Play the animation.

## Interface



 **Next Key-Previous Key.** Find the next or previous keyframe for the selected biped part. The field displays the key number.

 **Set Key.** Use either Set Key or turn on the 3DS MAX Animate button to create keys when you are moving biped objects. Using the Set Key method, you can experiment with different biped poses without updating the motion until you find the desired pose. You can also quickly fine tune your motion by setting a key and

adjusting the key parameters on the Key Info rollout without having to transform the biped in the viewports.

0 is a keyboard shortcut for setting keys.

Note: If a biped key is current, then TCB, XYZ position spinners, and IK Blend parameters can be updated without using Set Key or having the Animate button turned on.

Note: If 3DS MAX bones using the IK Controller or 3DS MAX Particle Emitters are linked to the biped, or if you are displaying 3DS MAX trajectories or ghosting, the Animate button *must* be on while the biped is positioned. These objects update their parameters in real time as they are positioned.



**Delete Key.** Deletes the key of the selected object at the current frame.

By default biped arm, hand, and finger keys are stored in the clavicle track. If you delete keys for any one of these objects, you lose positions for the rest of the arm objects at that frame. If you plan on extensive hand animation, turn on Arms in the Separate Tracks group of the Animation Properties rollout. This creates separate tracks for each biped arm object. Deleting an upper arm key will preserve hand and finger keys.



**Set Parents.** Turn on to create keys for the parent objects as well as the selected object only when Separate Tracks are turned on in the Animation Properties rollout.

Set Parents is used to store the position of the entire limb when a biped limb is moved using inverse kinematics, instead of rotated using forward kinematics. For example, if Set Parents is turned off and Separate Tracks are turned on in the Animation Properties rollout for the biped

arms, then the arm will snap back to its original position if you transform the biped hand.

If Separate Tracks are turned on for a biped body part, then turn on Set Parents. This allows you to use the Move transform to position the biped limbs.

If a hand key is created, then forearm and upper arm keys are also created.



**Trajectories.** Shows and hides trajectories for the selected biped object. You can edit keys on the bipeds horizontal and vertical track by turning on Trajectories, turning on Sub-Object, selecting the horizontal or vertical center of mass track and transforming keys in the viewports.

- You can bend the horizontal center of mass trajectory around selected horizontal keys by using the Bend Horizontal spinner in the keyframing rollout.
- Display trajectories to view how parameter changes in the Key Info rollout affects the biped motion. Changing Tension, Continuity, and Bias will affect the trajectory around the current key. Changing the value of IK Blend for a hand or foot will affect the trajectory between keys.
- Leave Trajectories on and turn on Show Buffer Trajectories on the Motion Capture rollout to compare a raw motion capture trajectory with the filtered trajectory on the biped. This is if a motion capture file has been loaded.
- Changing Dynamics Blend for a center of mass vertical key or changing the value of GravAccel will change gravity in a footstep animation and will therefore affect the trajectory.

**Time.** Enter a value to specify when in time the key occurs.

Use this to fine tune keyframe timing on a character by moving a key backwards and forwards in time.

**XYZ Position.** Reposition the selected biped part using these spinners.

A hand or foot can be repositioned in world coordinate XYZ. The biped center of mass can also be positioned using these spinners.

**Ease To.** Slows the velocity of the animation curve as it approaches the key. Default=0.

High Ease To causes the animation to decelerate as it approaches the key.

The default setting causes no extra deceleration.

**Ease From.** Slows the velocity of the animation curve as it leaves the key. Default=0.

High Ease From causes the animation to start slow and accelerate as it leaves the key.

The default setting causes no change of the animation curve.

**TCB Graph.** Charts the effect that changing the controller properties will have on the animation. The red mark at the top of the curve represents the key. The marks to the left and right of the curve represent an even division of time to either side of the key.

The TCB graph is a stylized representation of the animation around a single key.

**Tension.** Controls the amount of curvature in the animation curve.

High Tension produces a linear curve. It also has a slight Ease To and Ease From effect.

Low Tension produces a very wide, rounded, curve. It also has a slight negative Ease To and Ease From effect.

The default value of 25 produces an even amount of curvature through the key.

**Continuity.** Controls the tangential property of the curve at the key. The default setting is the

only value that produces a smooth animation curve through the key. All other values produce a discontinuity in the animation curve causing an abrupt change in the animation. Default=25.

High Continuity values create curved overshoot on both sides of the key.

Low Continuity values create a linear animation curve. Low continuity creates a linear curve similar to high tension except without the Ease To and Ease From side effect.

The default setting creates a smooth continuous curve at the key.

**Bias.** Controls where the animation curve occurs with respect to the key. Default=25.

High Bias pushes the curve beyond the key. This produces a linear curve coming into the key and an exaggerated curve leaving the key.

Low Bias pulls the curve before the key. This produces an exaggerated curve coming into the key and a linear curve leaving the key.

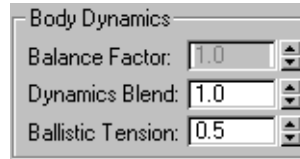
The default setting distributes the curve evenly to both sides of the key.



**Simple.** Collapses the lower part of the Key Info rollout, for simplified viewing.

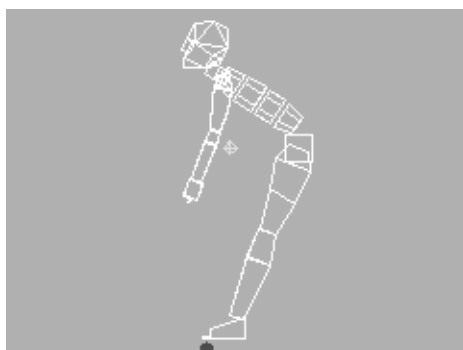
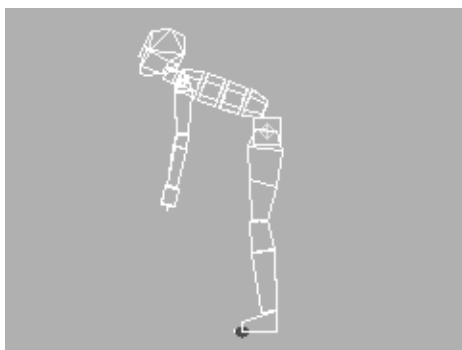
## Biped Dynamics Parameters

These parameters apply to the biped center of mass and are used by **character studio** to calculate the biped's airborne trajectory based on gravity (GravAccel) and time between footsteps, the amount of knee bend on landing (Ballistic Tension) and how the biped objects adapt to maintain balance (Balance Factor).



**Balance Factor.** Position the biped's weight anywhere along a line that extends from the center of mass to the biped's head. This center of mass (Body Horizontal track) parameter can be keyframed. To activate Balance Factor select the Horizontal Track (in the Track Selection rollout), set a key and enter a value in the Balance Factor field.

- For example, to create a *sit then walk* sequence, you could shift the biped's weight (balance) between 0 (the character is supported by the chair) for the sit key and 1 for the stand key (the character's pelvis shifts to maintain balance).
- If a character is seated, and reaches across the table, leave Balance Factor at 0; however the character leans, he will pivot from the center of mass, the pelvis will not move back to maintain balance.
- In a walking motion, a value of 2 will swing the hips and keep the biped head steady; a value of 0 will keep the hips steady and swing the upper body.



A value of 0 in the first image causes the biped not to compensate for weight. A value of 2 in the second image causes the biped pelvis to move away from the Center of Mass to compensate for weight.

The Balance Factor determines how far the biped's hips will shift forward or backward to compensate for forward or backward bending of the spine. When the biped has a normal weight distribution between the upper and lower body, the default value of 1 causes the hips to swing backward as the biped bends over to compensate for the forward weight.

At times, when the biped leans, you will want the biped's hips to refrain from shifting to compensate for the forward weight. This would be when the biped is sitting or falling down. A Balance Factor of 0 (the minimum value) causes

the hips to stay still when the biped leans forward or backward.

A value of 0 places the biped's weight at the center of mass. A value of 1 places the biped's weight above the center of mass. A value of 2 places the biped's weight in the head. Click the spinner up arrow to move the biped's weight distribution toward the head. Range=0.0 to 2.0; Default=1.

Note: You can also shift the center of mass by turning on Figure mode, selecting the center of mass object, and using *Rubber Band mode* (see page 147) to move the center of mass to a new position. This method cannot be keyframed, but it allows you to move the center of mass outside the biped body. To simulate pushing a heavy object, move the center of mass behind the biped for example.

**Dynamics Blend.** Select the Body Vertical track (center of mass vertical track) and control the amount of gravity in an airborne period, as in a running or jumping motion. This parameter has no effect on a walking motion where footsteps overlap.



The vertical center of mass track Dynamics Blend parameter is set to 1 for both keys (white squares) in the first jump, and to .5 in the second jump.

A value of 1 uses the *GravAccel* (see page 181) value to calculate gravity. A value of 0 removes the effects of gravity calculation and flattens a jumping or airborne motion.

**Ballistic Tension.** Select the Body Vertical track (COM) and control the amount of spring or

tension when the biped lands or takes off from a jump or run step. The change is subtle.

A walk cycle will not activate this value. The biped has to be airborne, *then* the Lift and Touch vertical keys will display a Ballistic Tension value.

If there are more than three vertical keys during a support period, you can also edit Ballistic Tension for the lift-off key; otherwise Biped uses the same value for touchdown and lift-off, since it is assumed that there is only one vertical dip in the motion. Low values are high tension (less dip in the trajectory). Default=0.5; Range=0 to 1;

## IK Key Info Rollout

Select the Biped > Motion Panel > IK Key Info rollout

This rollout is the heart of the IK system. Controls here allow you to set IK constraints and pivots for the biped hands and feet. There are three preset set key buttons that apply the most common IK constraints. A biped limb can be put into the coordinate space of the world or an object in the scene as well as body space.

### Inverse Kinematics

Footstep and freeform animations use similar IK constraints and extensions. All edits become unified as keyframe based in **character studio 3**. This means that in a footstep animation you can edit keys to change footstep duration. Deleting and inserting keys or changing IK space or IK blending will alter footstep duration also. By definition a footstep in both a Freeform and Footstep animation is the start and end of a sequence of IK constraints in World Space with an IK Blend value greater than 0. If this rule is adhered to then you can convert easily between

footstep and freeform animation using Convert on the General rollout.

Sliding footsteps are understood as footsteps with moving IK constraints. A sliding footstep is created by using the Sliding Key set key button, which automatically sets IK Blend to 1 and turns off Join to Previous IK Key.

In cases involving edits that alter the length of ballistic intervals (biped is airborne), the software insures that there is a vertical key occurring at the liftoff and touchdown frames in order to calculate the correct ballistic motion, so vertical keys are inserted if needed.

Footstep constraints are implemented with a pivot-based IK system. This allows you to pivot a hand or foot around a selected pivot. In a walking motion you can select a pivot on the ball of a foot and rotate the foot around it for example.

Since both footsteps and freeform use the same IK system:

- Conversions between Footstep and Freeform IK are identical.

Converting from freeform to footstep will always preserve the IK constraints. In fact, if there is no Biped dynamics at work during ballistic motion (that is, the Dynamics Blend for vertical keys is zero), the conversion to footsteps will not change the motion at all. Therefore, it is often desirable in the workflow to convert to footsteps to simply move clusters of IK pivots around using footstep edits. You can then convert back to freeform without losing the coherence of your work.

- Footsteps now serve as “gizmos” for positioning the underlying IK pivots.

Unlike previous releases of the software, you are free to edit the IK pivot constraints of the footstep. This means that a footstep is best

thought of as a convenient “gizmo” for moving IK pivots. The IK pivots will move as if they are “linked” in the coordinate space of the footstep, but they no longer need to be rigidly locked onto the footstep’s plane. In cases where you want to eliminate foot sliding, the “Join to Prev IK Key” (Join to previous IK key) in the IK system itself provides for precise control.

- The ability to set IK pivots of extracted footsteps dramatically improves motion capture editing.

When motion capture files are imported, the IK pivots can now be computed from the footsteps. Using the default sliding footstep parameters, the extracted footsteps will not alter the coherence of the raw data coming in. In cases where the motion capture data is already of high quality, the footstep extraction will retain its precise motion. This allows you to edit the motion with footstep operations, even if the data is not key reduced.

## Procedures

### To put the biped legs into world space

In a freeform animation (no footsteps), putting the character’s feet into the coordinate space of the world during footsteps will simplify the process of animating the biped. Setting a planted key anchors the foot to world coordinate space and prevents it from sliding.

If you are manually creating a walk or run then you should alternate the IK constraints for the biped feet. If a foot is on the ground use Set Planted Key or Set Sliding Key. If a foot is in the air between footsteps then use Set Free Key.

If you plan on converting a freeform animation to a footstep animation later, the keys representing footsteps should have IK blend equal to 1. IK constraints are used by the

Convert command in the general rollout to extract footsteps.

1. Select a leg on the biped.



2. On the IK Key Info rollout click Set Planted Key.

This turns on Object, sets IK Blend to 1.0, and turns on Join to Previous IK Key.

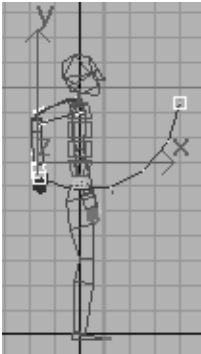
3. Test the leg by moving the center of mass up and down, the foot should stay planted on the ground as the biped body moves up and down.

### To set a pivot point for a hand or foot rotation

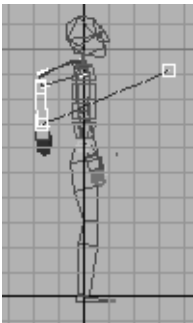
1. Select a hand or foot.
2. Find or set a keyframe that is in world or object space.  
A key that has an IK Blend of 1 with object selected is world space, if an object is also selected and displayed in the Object Space Object field then the hand or foot is in object space. You can click Set Planted Key to place a limb into world space.
3. Turn on Select Pivot and select one of the displayed pivots in the viewports.
4. Turn off Select Pivot and rotate the limb about the pivot.
5. Set a key.
6. Click Select Pivot to turn it off.



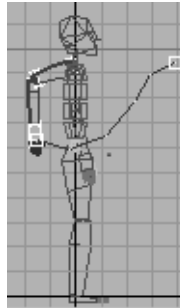
## Using IK Blend to vary limb trajectory



Default trajectory for the hand between two keys. In the Kinematics group, Body is selected and IK Blend=0 for both keys.



Trajectory for the hand when both keys are set with IK Blend=1. In the Kinematics group, Body is selected and IK Blend=1 for both keys.



With IK Blend=0 for the first key and 1 for the second key, the trajectory starts as a forward kinematic motion (curve) and ends with inverse kinematics.

## To link the biped hand to an object using Object Space Object

This attachment is set on a per key basis and can be blended. Blending would work well if the character was catching a ball, as the blend value increases between two keys the hand moves to match the velocity of the ball.

1. Select the biped's hand and move it to the correct position, relative to the ball.
2. Click Set Key, or turn on the Animate button before positioning the hand.

A key must exist to store Object Space parameters.

3. Turn on Select Object Space Object (button with the arrow).

Select the ball in a viewport.

In the viewports, the cursor changes to a plus sign.

4. On the IK Key Info rollout turn on Object in the Kinematics group; set IK Blend to 1.

Anywhere the ball moves, the hand will follow at this keyframe. The duration of the attachment is created when another key is set with similar parameters at a different frame or time.

Turning on the anchor, on the Keyframing rollout, for the selected biped limb will lock the hand to the object until further keys can be set to establish the period of attachment. Anchors are an interactive aid for keeping attachments in place while you are moving either the object, the body of the biped, or the time-slider, which can change both elements simultaneously. You should turn anchors off once your keys have been set that perform the desired attachments. The anchor buttons are on the Keyframing rollout, on the Motion command panel.

Anchors are no longer critical. By using Set Planted Key on two consecutive key frames the limb will snap to the first keys location.

**To activate an anchor, do the following steps in either order**

- Select the appropriate hand or foot and move or rotate it into the desired attached posture in relation to the “Object Space” object. If there is no object space object specified (in the Object Space field on the Key Info rollout), the hand/foot will then become anchored in World space.
- On the Keyframing rollout, click the button for the limb you want to anchor: Anchor Right Arm, Anchor Left Arm, Anchor Right Leg, or Anchor Left Leg.

Note: Anchors are no longer critical. By using Set Planted Key on two consecutive key frames the limb will snap to the first keys location.

Note: The arm or leg you select beforehand does not actually have to be the same as the arm or leg you are anchoring.

Note: You could use the Link tool on the toolbar or the Link Controller in 3DS MAX to attach an object to the biped; in which case the object follows the motion of the

hand. If the character is running and a pistol has to follow the swinging motion of the hand, then linking the pistol to the hand using the 3DS MAX Link tool would be the logical choice.

**To set keys that define animatable IK attachments**

1. At the frame for the start of the attachment, move your hand or foot to the desired location in relation to the Object (or World Space, if no object is specified)
2. On the Keyframing rollout, set an anchor for that limb so that the hand or foot remains attached in the same place.  
This step is optional since you can also snap to the same pivot point by checking the Join to Prev Key option later on in step 5.
3. On the IK Key Info rollout, click Set Key with IK Blend=1.0 and Object Space chosen or, as a shortcut, click the more convenient Set Planted Key button.
4. Click on Select Pivot Point and then select one of the end effector’s red spheres displayed in the viewpoint
5. Move to the frame of detachment and reposition the hand or foot (if desired).
6. Click Set Key with IK Blend=1.0 and Object Space chosen or, as a shortcut, if you wish to snap to the same pivot point as the previous key, click Set Planted Key button again. This will turn on the Join to Prev Key checkbox. If you wish to allow the pivot point to move, click Set Sliding Key. This will turn off the Join to Prev Key checkbox.
7. If you used anchors, turn off the anchors and move between the attachment and detachment keyframes, noting whether the hand or foot is attached for the desired interval.

8. If the hand or foot detaches during the interval you should delete the default keys for the attached limb in the interval between the start and end attachment keys.

Note: The attached hand or foot may drift from its attach point as the object tries to maintain a smooth, continuous trajectory. To insure linear motion between keys, you should also set Continuity to 0 for the keys defining the attach interval. You can also adjust the tension to 50 to eliminate any undesired overshooting.

You can continue to animate each limb's IK attached object conventionally—moving or otherwise transforming it in different keyframes. The arm or leg moves to follow it using the kinematic solution specified by the IK Blend setting.

Note: The procedure above can be embellished by setting more than two consecutive keys with IK Blend=1.0 in order to move the hand or foot over the object in some specific way during the attachment. Also, subtle changes in the dynamics of the attachment are possible by manipulating the IK Blend values.

An eased detachment can be created, for example, by setting the key that follows the detach key to be in Object space with IK Blend=0.0. This will cause the blended IK values to remain in Object space while the IK Blend values ramp gradually down from 1.0 to 0.0.

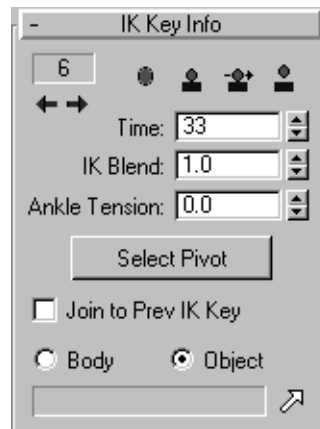
Saving a *.bip* file from a motion containing IK Blend keys will store the IK Blend information in the file. When the *.bip* file is loaded onto a new biped, the IK Blend settings and the body/object state are mapped onto whatever Object Space Object is already defined for the existing biped. If no Object Space Object is assigned, then the IK Blend setting will be stored in World space.

Note: You can change the position of the elbow or knee without affecting the attach point of the hand or foot. While the anchors are turned on, select the forearm or a calf and rotate it around its X axis. The hand and foot stays put while the limb is rotated.

**Warning:** You must save the animation as a 3DS MAX file in order to save the animation keys for the Object Space Object.

## Interface

Footsteps will adapt automatically to changes made in IK values of leg keys. In Freeform and Footstep animation a footstep interval is the start and end of a sequence of IK constraints in World Space with IK Blend > 0.



**Next Key-Previous Key.** Find the next or previous keyframe for the selected biped part.

The field displays the key number.

**Set Key.** Use either Set Key or turn on the 3DS MAX Animate button to create keys when you are moving biped objects.

**Set Planted Key.** Set a biped key with IK Blend set to a value of 1, Join to Previous Key turned on and Object selected.

In a Footstep or Freeform animation all footsteps that do not slide should have Join to Previous Key turned on.

**Set Sliding Key.** Set a biped key with IK Blend set to value of 1, Join to Previous Key turned off and Object selected. This creates a sliding footstep. Sliding footsteps display in the viewports with a line running through the middle of the footstep. Sliding footsteps are understood as footsteps with moving IK constraints.

In a Footstep or Freeform animation if the foot slides rather than being planted then use Set Sliding Key.

**Set Free Key.** Set a biped key with IK Blend set to a value of 0, Join to Previous Key turned off and Body selected.

In a Footstep or Freeform animation a biped leg in a move state should have a “free” key.

**Time.** Enter a value to specify when in time the key occurs.

Use this to fine tune keyframe timing on a character by moving a key backwards and forwards in time.

**Ankle Tension.** Adjusts the precedence of the ankle joint over the knee joint. When set to 0, the knee takes precedence. When set to 1, the ankle takes precedence.

This effect is only visible between keyframes.

**Select Pivots.** Turn on to select pivots on the biped hands and feet to rotate around. After a pivot is selected in the viewports turn off Select Pivots and rotate the hand or foot.

Pivots are only available if the biped hand or foot is in world or object coordinate space.

**Join to Previous Key.** Turn on to put the biped foot in the coordinate space of the previous key. Turn off to put the biped foot into a new reference position.

Turn off and move the biped foot to create a sliding footstep for example.

**IK Blend.** Determines how forward kinematics and inverse kinematics are blended to interpolate an intermediate position. Using an arm to move a hand is an example of forward kinematics. Using the hand to move the arm is an example of inverse kinematics.

Activates when a biped arm or leg (hand and foot) key is current.

- 0 with Body turned on is normal biped space (forward kinematics).
- 1 with Body turned on is Inverse Kinematics, creates more straight-line motion between biped keys.
- 1 with Object turned on, but no Object Space Object specified, puts the limb fully into world space.
- 1 with Object turned on *and* an Object Space Object specified puts the biped limb into the coordinate space of the selected object; the biped limb follows the specified object.

**Body.** The biped limb is in biped coordinate space.

**Object.** Object Space, the biped limb is either in World coordinate space or the coordinate space of the selected object. Coordinate space can be blended between keys.

**Select Object Space Object (button with an arrow in it).** Link a biped hand or foot to an object in your scene.


## Keyframing Rollout

Select the biped. > Motion panel >  
Keyframing rollout


Use controls on the Keyframing rollout to set and delete keyframes, copy and paste biped posture, mirror a biped animation, and select or change keys using Set Multiple Keys. You can also bend the horizontal center of mass track around a selected horizontal key.

### Procedures

#### To create a “high step” type walk from a default biped walk cycle using Set Multiple Keys


1. Create a walk cycle for the biped in Footstep mode.
2.  Click Set Multiple Keys to display the Set Multiple Keys dialog.
3. Turn on Left Leg, Right Leg, and Move, then click Select.  
All the Move keys are selected for both legs. Move is the leg state between footsteps.
4. Rotate either of the biped upper legs up, then click Apply Increment.  
The rotation is applied to all the selected biped leg keys.  
Close the dialog and click Play; the biped now has a high step walk. Open Track View and look at the left and right biped leg tracks; all leg keys in the Move state are white (selected).

#### To bend the horizontal center of mass track


1. On the Track Selection rollout select the horizontal track.
2.  On the Display rollout turn on Trajectories.

3. Turn on Sub-Object Trajectories.
4. Select a key on the trajectory in the viewports.
5. Use the Horizontal Bend spinner to bend the horizontal path.


#### To edit keys on the Center of Mass trajectory

1. Load a Freeform animation.
2.  On the Key Info rollout turn on Trajectories.
3. On the Track Selection rollout click Body Horizontal or Body Vertical.
4. Turn on Sub Object Trajectories at the top of the Motion panel.
5. In the viewports select and move keys on the Center of Mass trajectory.

#### To copy and paste an entire biped track


1.  Make sure that Copy Track is active. This is a three button fly-out, choose the Copy Track button.
2. Select the part(s) of biped corresponding to a track or set of tracks that you wish to copy.
3. Click on Copy Track.  
The track(s) are copied to memory.
4. Click on another biped.
5. Click on Paste Track or Paste Opposite Track.  
The entire animation track is pasted to the selected object(s).

#### To switch the left and right arm tracks


1. Select both arms (any part of the arm).
2.  Click Copy Track.
3. Click Paste Opposite Track.

## Interface



 **Set Key.** Use either Set Key or turn on the 3DS MAX Animate button to create keys when you are moving biped objects. You must use Set Key for certain operations such as anchoring arms and legs or specifying an IK Blend. 0 is a keyboard shortcut for setting keys.

Note: If 3DS MAX bones using the IK Controller or 3DS MAX Particle Emitters are linked to the biped, or if you are displaying 3DS MAX trajectories or ghosting, the Animate button *must* be on while the biped is positioned. These objects update their parameters in real time as they are positioned.

 **Delete Key.** Deletes keys at the current frame for the selected biped object(s).

By default biped arm, hand and finger keys are stored in the clavicle track. If you delete any one of these objects, you lose positions for the rest of the arm objects at that frame. If you plan on extensive hand animation, turn on Arms in the Separate Tracks group of the Animation Properties rollout, this creates separate tracks for each biped arm object. Deleting an upper arm key will preserve hand and finger keys.



**Anchor Right Arm-Left Arm-Right Leg-Left Leg.** Allows you to fix the location and orientation of hands and feet temporarily. Use anchors when you are setting up animation with

inverse kinematics Object space, in which the arm or leg follows an object in the scene. Anchors ensure that the arm or leg keeps its alignment until you set the second key that establishes the Object-space sequence.

Note: Anchors are no longer necessary because of improvements to the IK system. When you use Set Planted Key in the IK Key Info rollout the limb is positioned to the previous IK key (join to previous IK Key).



### Three button flyout

**Copy Posture-Copy Pose-Copy Track.** Use Copy Posture to copy selected biped limb(s) posture. Use Copy Pose to copy the entire biped posture. Use Copy Track to copy the entire animated track from a biped object. You can then paste this posture, pose or track onto the same or a different biped. Copy Track can be used to copy tracks from one biped to another, from one limb to another, and from the left to right or the right to the left side of the biped.

A handy technique is to create an extra biped for posture storage. Postures that you will use frequently are applied to this biped at different frames. Then, when you need a particular posture, use the time slider to move to the appropriate frame, select the biped limbs to copy, and click Copy Posture. Then select the biped you are animating and click Paste Posture.

**Tip:** Use Copy Posture and Paste Posture to copy the first frame to the last frame in a looping animation.



**Paste Posture/Pose/Track.** Pastes the position and orientation from the posture buffer onto the currently selected biped. Pastes an entire animation track onto the biped.

This is useful for copying a pose from one biped to another. You don't have to select the same objects on the biped you want to paste the pose onto. The posture buffer retains information on which objects were copied. Paste Posture will automatically paste the pose onto the appropriate biped objects, even though they may not be selected.

Note: Turn on the Animate button before pressing Paste Posture to set a key. If you don't have the Animate button turned on, you will need to select the appropriate object(s) and press Set Key to store the new posture.



**Paste Posture/Pose/Track Opposite.** Pastes the position and orientation from the posture buffer onto the biped objects on the opposite side of the selected biped. Pastes an entire animated track onto the opposite side of the biped.

This helps you create symmetrical poses and movements. Use Paste Posture Opposite in Figure mode to speed up the process of aligning the biped to the mesh. Perfect the fit of the right arm, hand, and fingers; copy them, then use Paste Posture Opposite to copy the changes to the opposite arm.



**Mirror.** Mirror the entire biped animation.



**Set Multiple Keys.** Select keys using filters or apply a rotational increment to selected keys. Use this to change periodic motion keys in Track View. Displays the *Set Multiple Keys dialog* (see page 269).

Use Set Multiple Keys to edit a default walk or run motion created in Footstep mode. Open Track View, select all the leg keys in the "move" state using the State Filters in the Set Multiple

Keys dialog and then apply an increment to all of these keys, for example.



**Set Parents.** When a limb key is created, keys are created for the parent objects also if Set Parents is turned on. Use Set Parents with Separate Tracks turned on on the Animation Properties rollout.

Set Parents is used to store the position of the entire limb when a biped limb is moved using inverse kinematics instead of rotated using forward kinematics. For example, if Set Parents is turned off and Separate Tracks are turned on in the Animation Properties rollout for the biped arms, then the arm will snap back to its original position if you transform the biped hand.

If Separate Tracks are turned on for a biped body part, then turn on Set Parents. This allows you to use the Move transform to position the biped limbs.

Note: Separate Tracks, on the Animation Properties rollout, adds biped object transform tracks.

**Bend Horizontal.** Bend the horizontal center of mass track around a selected horizontal key.

Select the center of mass horizontal track, turn on trajectories, turn on Sub-Object, then select a key on the trajectory and use Bend Horizontal to bend the path (trajectory) about the selected key. Bend Horizontal works in both footstep and freeform animation.

Note: You can also move vertical and horizontal Center of Mass keys on the trajectory.


## Display Rollout

Select the biped. > Motion panel > Display rollout

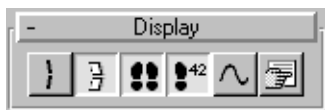
Using controls on the Display rollout on the Motion panel, you turn on or off the display of biped, bones, footsteps, footstep numbers, and trajectories. You can also change footstep colors and set the number of bipeds you want to play back.


### Procedure

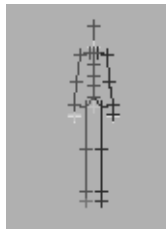
#### To display footsteps

- Select a biped that has footstep animation.
-  On the Display rollout turn on Footsteps.  
Footsteps will display in the viewports.  
Note: If a biped has only freeform animation or no animation, no footsteps will appear.

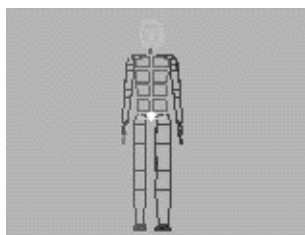
### Interface



 **Bones.** Displays biped bones. Bones are represented as the color of the corresponding links, which do not render. Selecting Bones is useful for seeing exactly where the joints fall in relation to the biped objects.

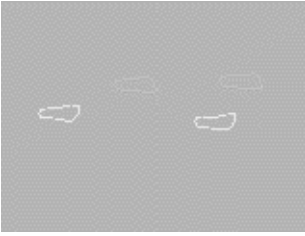


**Objects.** Displays biped body objects; these will render if you do not select Hide before rendering. Hide the biped objects before scene rendering. You can also hide individual body objects by using the standard 3DS MAX Hide controls found in the Display panel and Display Floater.

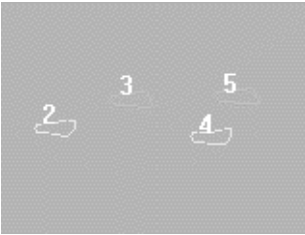


**Footsteps.** Displays biped footsteps in the viewport. Footsteps are represented as green and blue foot-shaped outlines by default; these are also visible in preview renderings. Turning off the Footsteps button also turns off the footstep numbers and the center of mass shadow (the disc between the biped feet).





**Footstep Numbers.** Displays biped footstep numbers. Footstep numbers specify the order in which the biped will move along the path created by the footsteps. Footstep numbers are displayed in white and do not render, but do appear in preview renderings.



**Trajectories.** Displays trajectories for selected biped limbs.

You can edit keys on the bipeds horizontal and vertical track by turning on Trajectories, turning on Sub-Object, selecting the horizontal or vertical center of mass track and transforming keys in the viewports. Use Trajectories when editing keyframe parameters to visualize their influence, and to compare raw and filtered motion capture data.



**Display Preferences.** Shows the *Display Preferences dialog* (see page 177) which is used to change footstep colors, trajectory parameters, and to set the number of bipeds to be played back when you use Biped Playback on the

General rollout. Footstep color preference is a good way to distinguish between the footsteps of two or more bipeds in a scene.

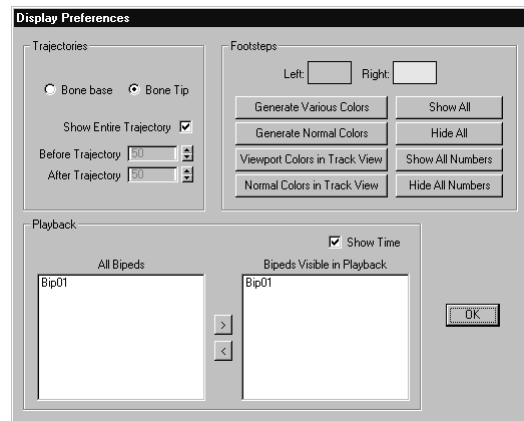
## Display Preferences Dialog

Select a Biped > Motion panel > Display rollout > Display Preferences > Display Preferences dialog

Use the controls in the Display Preferences dialog to change footstep colors and trajectory parameters, and to set the number of bipeds to play back when you use Biped Playback on the General rollout.

You access the Display Preferences dialog by pressing the Display Preferences button on the Display rollout on the Motion panel.

## Interface



### Trajectories group

Note: Trajectories do not display using In Place mode.

**Bone base.** Displays bone base trajectories.

**Bone Tip.** Displays bone tip trajectories (default).

**Show Entire Trajectory.** Displays trajectories for all animated frames.

**Before Trajectory.** Sets the number of frames to display trajectories before the current frame.

**After Trajectory.** Sets the number of frames to display trajectories after the current frame. Using Before and After Trajectory will result in a “traveling” trajectory display that will move with the biped through space.

### Footsteps group

**Left.** Selects the color for left footsteps. Double-click the color swatch next to this selection to display the Color Selector and set the color for left footsteps.

**Right.** Selects the color for right footsteps. Double-click the color swatch next to this selection to display the Color Selector and set the color for right footsteps.

**Generate Various Colors.** Asks whether you want different colors for left and right footsteps. Based on your response, generates various colors for footsteps in the viewports. This works with multiple bipeds.

**Generate Normal Colors.** Changes right footsteps to blue and left footsteps to green, the default. Applies to all bipeds in the viewports.

**Viewport Colors in Track View.** Displays viewport footstep colors in Track View.

**Normal Colors in Track View.** Displays normal footstep colors in Track View.

**Show All.** Shows all footsteps.

**Hide All.** Hides all footsteps.

**Show All Numbers.** Shows all footstep numbers.

**Hide All Numbers.** Hides all footstep numbers. Show and hide are useful when there are multiple bipeds in the viewports.

### Playback group

Controls in this group limit the number of bipeds to play back when you use Biped Playback on the General rollout on the Motion panel.

**Show Time.** Displays frame numbers in the viewport during playback.

**All Bipeds.** Lists all bipeds in the scene. Select a biped name in the window and click the right arrow to move it into the list of bipeds that will be visible during playback with Biped Playback.

**Bipeds Visible in Playback.** Lists bipeds visible during playback with Biped Playback. Select a biped name and click the left arrow to *exclude* it from this list.

---

## Layers Rollout

Select the Biped > Motion panel > Layers rollout



Controls in the Layers rollout allow you to add layers of animation above the original biped animation. This is a powerful way of making global changes to your character animation. For example, simply add a layer and rotate the spine forward at any frame, and a run cycle becomes a crouched run. The original biped motion is kept intact and can be viewed by switching back to the original layer. Layers can be viewed individually or as a composite of all the animation in all the layers. Layers behave like a freeform animation; the biped can adopt any position.

**Tip:** You can globally translate both a footstep and freeform animation by doing a layered edit on the Center of Mass. For example, by adding a layer and moving the Center of Mass you can move a freeform or footstep animation.

Layers will allow you to easily adjust raw motion capture data, which contains keys at every frame. Simply add a layer and keyframe the biped. The original layer is displayed as red bones.

**Tip:** To compare subtle motion between layers, turn off Object and turn on Bones on the Display rollout. You can compare the layer to the original with the red bone display.

Layers can be named for clarity.

## Layers versus Set Multiple Keys


Use Layers for global changes or for longer periods of interpolation. Use Set Multiple Keys for changing a group of selected keys in Track View. Set Multiple Keys is accessed from the *Keyframing rollout* (see page 173).

With Layers, you can interpolate the position of the biped over a wide range of frames. To change the biped posture over a period of one hundred frames during a run cycle, simply create a new layer and create two spine keys one hundred frames apart; during the run the posture changes during that period.

Set Multiple Keys is useful to correct for periodic keys. If a character bobs his head from left to right and all the “head right” keys need to be exaggerated, then open Track View, select all the “head right” keys, rotate the biped head and click Apply Increment in the *Set Multiple Keys dialog* (see page 269).

## Procedure

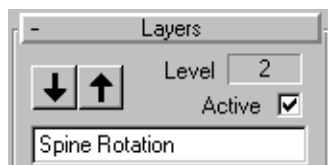
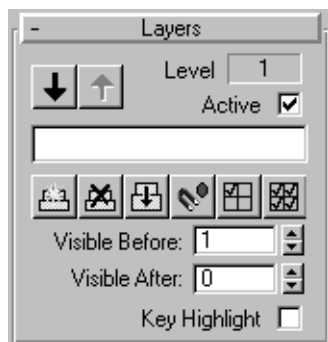
### To make a global posture change to the biped

1.  Select a biped, then click Create Layer.
2. Turn on the Animate button, then rotate a spine object on the biped forward.

The biped leans forward through the animation.

Use Set Snap Key at a frame where the character's posture should return to its original motion.

## Interface



**Next-Previous Layer.** Navigate through the layers using the up and down arrows.

**Level.** This field displays the current layer (Level).

**Active.** Toggles the displayed layer on and off.

**Name Field.** Type a name to easily identify a layer.



**Create Layer.** Creates a layer, and the Level field increments.

Position the biped to create keys in a layer.



**Delete Layer.** Deletes the current layer.

All layer numbers 'above' the one deleted are decremented by one.



**Collapse Layers.** Collapses all the layers into layer 0.

Legs that stray from the original footsteps in higher layers are “pulled in” to the original footsteps.



**Snap Set Key.** Snaps the Selected biped part to its original position in layer 0 and creates a key.

Use this in higher layers to return the selected biped part to the original motion. If a layer has a posture key that bends the character forward at frame 2 and you want to return the biped to its original posture motion at frame 50, use Snap Set Key at frame 50 with a spine object selected. The character will interpolate from its forward posture position to its original posture position between frame 2 and 50.



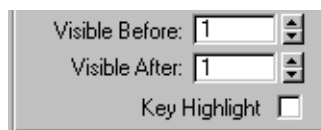
**Activate Only Me.** View the animation in the selected layer.

Select Play after turning on Activate Only Me to view a layer's keys in motion.



**Activate All.** Activates all the layers.

Playing the animation shows a composite of all the layers.



**Visible Before.** Sets the number of preceeding layers to display as stick figures.

**Visible After.** Sets the number of succeeding layers to display as stick figures.

**Key Highlight.** Displays keys by highlighting the stick figures.

## Animation Properties Rollout

Select the biped. > Motion panel > Animation Properties rollout

Use these controls to specify the way you want to create biped animation. The parameters let you modify gravity strength, dynamic properties for keys generated by newly created footsteps, the number of transform tracks available on the biped, and prevent key adaptation using parameters in the Animation Properties rollout.

### Biped Dynamics and Spline Dynamics

These parameters specify how new biped center of mass keys are created and therefore how you want to work with the biped. Turning on Spline Dynamics will create center of mass keys, without gravity and balance calculation, for newly created footsteps. This may feel more familiar to new users who are already familiar with spline interpolation.

Biped Dynamics calculates biped airborne trajectory, knee bend on landing and positions the biped to maintain balance when the spine is rotated. When parameters change, the biped adapts. Turn on Biped Dynamics and use this adaptation to your advantage.

You can always change from one method to the other on a per-key basis or for the entire animation at any time.

### Separate Tracks

By default, biped uses an optimized method for key storage. For example, keys for the fingers, hand, forearm, and upper arm are stored in the Clavicle transform track. If you are more comfortable knowing that a transform track is available for each arm object, then use the Separate Tracks group to make these transform

tracks available; transform tracks are displayed in Track View.

**Tip:** Separate Tracks are designed to be used with forward kinematics rotations. Using the Move tool to change the position of a biped limb with Separate Tracks turned on will require that you turn on Set Parents. If Set Parents is not turned on, then the limb you move will only be stored relative to its parent objects, and the new parent positions will not be stored. This will result in those parent objects snapping back to their originally stored location. Turning on Set Parents will insure that the position of the entire biped limb is stored.

Note: Separate Tracks are intended to be used as a preference. Turning Separate Tracks off and on will result in a key being stored for every biped object in the limb for the frames where any biped keys previously existed.

## Procedures

### To create separate tracks for biped arms

- Turn on Arms in the Separate Tracks group. If Track View is open, notice that fingers, hands, forearm, and upper-arm now have a transform track. A deleted upper arm key will not destroy palm and finger keys.

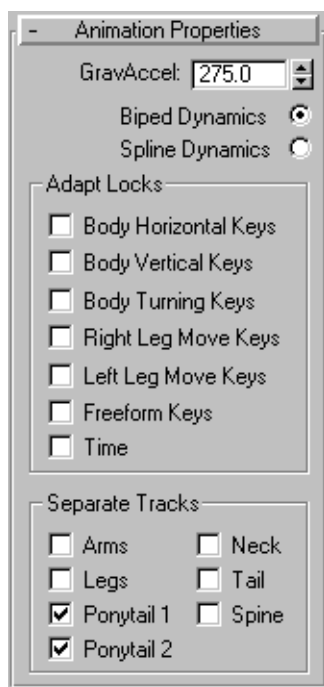
### To turn on Spline Dynamics for newly created keys

1. On the Animation Properties rollout, turn on Spline Dynamics.
2. Turn on Footstep Mode, if you haven't already.
3. Create new footsteps and create keys for inactive footsteps.

Balance Factor and Gravity are turned off. A jump created in Footstep mode has a flat trajectory; spine rotation does not alter the biped body to maintain balance.

**To use Adapt Locks to prevent keyframe adaptation**

1. On the Animation Properties rollout, select a track to lock.
2. Edit footsteps.  
The tracks you selected as locked are unaffected by footstep editing.

**Interface**

**GravAccel.** Sets the strength of the gravitational acceleration used to calculate the biped's motion.

By default, this parameter is set to accurately simulate Newtonian gravity as found on the Earth's surface.

At a value of 0, the biped still runs but the feet hardly get off the ground.

**Biped Dynamics.** Creates new center-of-mass keys using Biped Dynamics.

Keys for the center of mass Balance Factor and Dynamics Blend parameters are set to a value of 1. Biped calculates airborne trajectories and biped balance.

Note: Use Dynamics Blend on the Key Info rollout to set a vertical COM key that is a blend between Dynamics and Spline Interpolation; a value of 1 is full Dynamics, a value of 0 is full Spline interpolation.

**Spline Dynamics.** Creates new center-of-mass keys using full spline interpolation. All new vertical COM keys are created with Dynamics Blend=0, and all new horizontal COM keys are created with Balance Factor=0.

**Adapt Locks group**

Lock specified tracks to prevent automatic adjustments being made to those tracks when footsteps are moved in space or edited in time. All the locks except for Time work for footstep editing in space. Time, locks upper body keys when footsteps are edited in time (Track View). Adapt Locks only applies to a Footstep animation, not a freeform animation.

When you move a footstep in space or adjust footstep timing, Biped automatically adapts existing keyframes to match the new footsteps. Adapt locks allows you to preserve the exact position of already created keys for a selected track.

Adapt Locks does not need to be on all the time. For example, if you want to raise all the footsteps along the world Z axis, without changing the upper body position, turn on Adapt Locks Body Vertical Keys, turn on Footstep mode, select all the footsteps and move them up along the world Z axis. The footsteps are repositioned, the legs are adapted, but the upper body retains the same

motion rather than being raised with the footsteps. Now turn off Adapt Locks Body Vertical Keys, the upper body still retains its original motion.

Use Adapt Locks Time to retain upper body motion while editing footstep duration in Track View. When the duration of a footstep is changed, the biped leg will adapt by re-timing the touch, plant and lift keys. The biped upper body keys will retain their exact motion.

**Body Horizontal Keys.** Turn on to prevent adaptation of body horizontal keys when footsteps are edited in space.

**Body Vertical Keys.** Turn on to prevent adaptation of body vertical keys when footsteps are edited in space.

**Body Turning Keys.** Turn on to prevent adaptation of body turning keys when footsteps are edited in space.

**Right Leg Move Keys.** Turn on to prevent adaptation of right leg move keys (a leg move key, is a leg key between footsteps) when footsteps are edited in space.

**Left Leg Move Keys.** Turn on to prevent adaptation of left leg move keys (a leg move key, is a leg key between footsteps) when footsteps are edited in space.

**Freeform Keys.** Turn on to prevent adaptation of a freeform period in a footstep animation. The biped's position during a freeform period will not move if footsteps after the freeform period are moved further away.

**Time.** Turn on to prevent adaptation of upper body keys when footstep duration is changed in Track View.

Note: Leg and foot keys for frames in which the foot is in contact with the ground are always automatically adapted.

## Separate Tracks group

By default **character studio** stores a finger, hand, forearm, and upper-arm key in the clavicle track. The toe, foot, and calf keys are stored in the thigh track. This optimized approach to key storage works well in most cases. If you need extra tracks, turn them on for a specific biped body part.

For example, turn on Arms if you plan on extensive finger-hand animation; if an arm key is deleted, it will not affect the finger-hand keys. Notice that in Track View a transform track is now available for the first link of the thumb (stores all finger keys), hand, forearm, and upper-arm.

**Arms.** Turn on to create separate transform tracks for the finger, hand, forearm and upper-arm.

There is one finger track per hand. All finger keys are stored in the "Finger0" transform track, the first link of the biped thumb.

**Legs.** Turn on to create separate toe, foot, and calf transform tracks.

**Ponytail 1.** Turn on to create separate ponytail 1 transform tracks.

**Ponytail 2.** Turn on to create separate ponytail 2 transform tracks.

**Spine.** Turn on to create separate spine transform tracks.

**Neck.** Turn on to create separate transform tracks for the neck links.

**Tail.** Turn on to create separate transform tracks for each tail link.

## Structure Rollout


Select the biped. > Motion panel > General rollout > turn on Figure mode > Structure rollout

Turn Figure mode on to enable parameters on the Structure rollout. The Structure rollout provides parameters for changing the biped's skeletal structure to match your character mesh (dinosaur, robot, human and so on) and a name field to rename the biped center of mass object and append this name to all the links in the biped hierarchy.

After setting parameters to suit your character, use the Height parameter to scale the biped to the size of the mesh representing your character. This is the first step in fitting a biped to a mesh in Figure mode.

### Procedures

#### To scale a biped and a Physique mesh

-  Select the biped, turn on Figure mode, and then change the biped height on the Structure rollout.

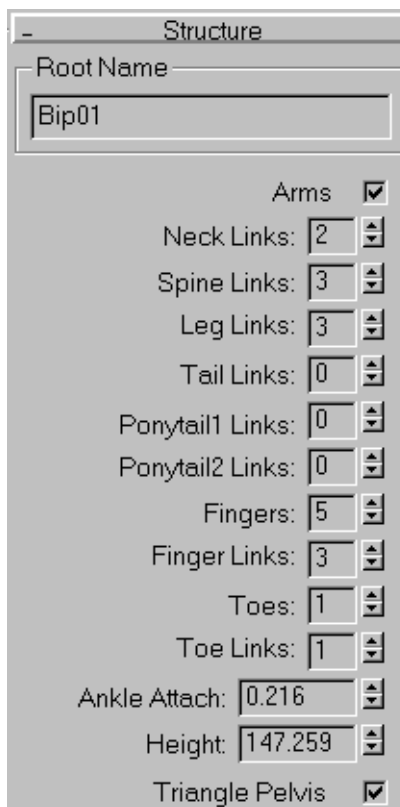
The biped and mesh scale together.

#### To name the biped

This actually names the biped center of mass and appends the name to all the links in the biped hierarchy.

1. Select a biped in the viewports.
2. On the Motion panel, on the Structure rollout, type a name in the Root Name field. Identifying the links of the character in any name dialog is simplified.

## Interface



**Root Name.** Displays the name of the selected biped's center of mass object. The center of mass (COM) is the root of the biped hierarchy; it is visible as a diamond shaped object in the pelvis area. The Root Name is appended to all the links of the biped hierarchy.

To simplify selecting biped links in the Select by Name dialog on the 3DS MAX toolbar, change the default center of mass name "Bip01" to the name your character will use. If you change the Root Name from Bip01 to John, for example, all of the links will have the name "John" appended to them, for example "John Pelvis" and "John L thigh."



## Structure parameters

**Arms.** Specifies whether or not arms will be generated for the current biped.

**Neck Links.** Specifies the number of links in the biped neck. Range=1 to 5.

**Spine Links.** Specifies the number of links in the biped spine. Range=1 to 5.

**Leg Links.** Specifies the number of links in the biped legs. Range=3 to 4.

**Tail Links.** Specifies the number of links in the biped tail. A value of 0 specifies no tail. Range=0 to 5.

**Ponytail1 / 2 Links.** Specifies the number of Ponytail Links. Range=0 to 5.

Animates hair with ponytail links. Ponytails are linked to a character's head and can be used to animate other appendages. Reposition ponytails in Figure mode and use them to animate a character's jaw, ears, nose, or anything that should move with the head.

Unlike the process of selecting the biped hand and dragging to reposition the entire arm, ponytails must be keyed using rotational transformations.

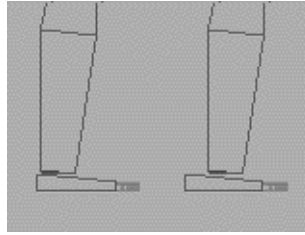
**Fingers.** Specifies the number of biped fingers. Range=0 to 5.

**Finger Links.** Sets the number of links per finger. Range=1 to 3.

**Toes.** Specifies the number of biped toes. Range=1 to 5.

**Toe Links.** Specifies the number of links per toe. Range=1 to 3.

Characters wearing shoes may only need one toe with one link.



**Ankle Attach=0.25 and Ankle Attach=0.5**

**Ankle Attach.** Specifies the right and left ankles' point of attachment along the corresponding foot block. The ankles may be placed anywhere along the center line of the foot block from the heel to the toe.

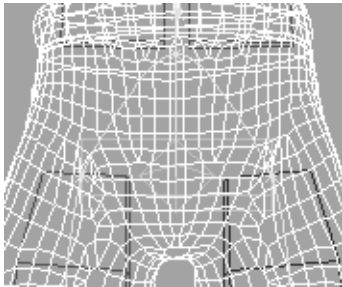
A value of 0 places the ankle attachment point at the heel. A value of 1 places the ankle attachment point at the toes. Click the spinner up arrow to move the ankle attach point toward the toes. Range=0 to 1.

**Height.** Sets the height of the current biped.

Use to size the biped to your mesh character before Physique is attached. This parameter is also used in a procedure to scale your character after Physique is attached.

**Triangle Pelvis.** Turn on to create links from the upper legs to the lowest biped spine object when Physique is attached. Normally the legs are linked to the biped pelvis object.

The pelvis area can be a problem when the mesh is deformed with Physique. Triangle Pelvis creates a more natural spline for mesh deformation.



Triangle Pelvis creates links

Triangle Pelvis creates two links that extend from the legs to the lowest biped spine object. A link from the biped pelvis to the lowest spine object is also created. This provides natural deformation to this area after Physique is applied and the character is moving. If you are working on a new character, turn this on before applying Physique. If Bones is turned on in the Display rollout, links from the legs to the lower spine object are visible.

## Motion Capture Rollout

Create or select a biped. > Motion panel > Motion Capture rollout

The tools on the Motion Capture rollout on the Motion panel are typically used for working with raw motion capture data. You can also load standard *.bip* files using Load Motion Capture File. For example, you might do this if you want to loop the motion. This rollout includes tools for:

- Batch conversion of motion capture files.
- Converting the motion capture file stored in the motion capture buffer.
- Pasting one frame of motion capture data from memory to selected biped limbs.
- Displaying raw motion capture data as a stick figure.
- Displaying raw motion capture trajectories.

The buttons in the bottom row are used mainly with marker files, although the calibration controls also work with raw *.bvh* files. Import, calibrate, and filter marker files (*.csm*) using tools on the Motion Capture rollout. Markers are placed on an actor during motion capture to identify joints; calibration lets you adjust the biped relative to the original marker positions if necessary. Load only raw marker files with no key reduction or footsteps to enable marker calibration controls.

Note: See *BVH File Specification* (see page 199) and *CSM File Specification* (see page 213) for the BVH and CSM file specifications, respectively.

## Motion Capture Buffer

Raw motion data is automatically stored in the motion capture buffer when files (.csm, .bvh, and .bip) are loaded using *Load Motion Capture File* (see page 189). This buffered raw motion data is independent of the biped motion in your scene and can be used in various ways:

- Use *Convert from Buffer* (see page 190) to try alternate filter settings quickly; this saves you from having to browse for the same file.
- After importing a .bvh or .bip file using footstep extraction and key reduction, you can use *Paste from Buffer* (see page 190) on selected biped limbs (and COM) to paste keys from the raw motion capture data to the filtered data; do this if critical motion has been lost in the filtering process.
- If you specify Load Buffer Only in the *Motion Capture Conversion Parameters dialog* (see page 192), the motion file is loaded into the motion capture buffer without altering the biped animation. Use this to paste posture and limb keys from any file onto the biped animation in your scene.

When you load a motion capture file, the motion capture buffer is loaded with motion capture data from that file. This buffer is altered during calibration. It is also used to show the motion capture markers and trajectories.

Internally, there is only one motion capture buffer. It is often large, so its contents are not saved before a file load or a calibration.

Therefore, if you undo a motion capture file load, the contents of the motion capture buffer does not change. That's why you'll still see old markers and trajectories. It is not possible to undo calibration.

## Marker Files

Unlike a .bip or .bvh file that contains limb rotation data, a .csm marker file (see page 198) contains only marker position data. When a raw marker file is imported, only marker position data is buffered in the motion capture buffer. Character Studio uses the marker data to extract limb rotation data to position the biped. After using the calibration controls on the Motion Capture rollout to correct biped scale and posture relative to the markers, use *Convert from Buffer* (see page 190) to filter the raw marker data to key reduce and extract footsteps.

Note: A .csm marker file (see page 198) is an ASCII file.

## See also

*Importing Motion Capture Data* (see page 83)

## Procedures

### To use Convert From Buffer

A Motion Capture file should already be in memory. Use *Load Motion Capture File* on the Motion Capture rollout to import a motion capture file if one is not already be in memory.

1. Select a biped.
2. On the Motion Capture rollout, click *Convert From Buffer* to display the Motion Capture Conversion Parameters dialog.
3. Adjust parameters and click OK.

### To compare raw and filtered trajectories

1. Select a biped and Turn on Show Buffer Trajectory on the Motion Capture rollout.
2. Click Trajectories on the Display rollout.

As you select various biped parts, two trajectories are displayed. The yellow trajectory represents raw motion capture data in the motion capture buffer; the

purple trajectory represents the filtered data.

### To use Show Buffer

1. Create a biped (see page 143).
2. Use Load Motion Capture File on the Motion Capture rollout to import a motion capture file.
3. Turn on Show Buffer on the Motion Capture rollout.

A red stick figure appears, representing the raw motion capture data.

Play the animation; the animation of the biped representing filtered motion capture data and the red stick figure play back together.

**Tip:** For a very accurate visual comparison between raw motion capture data and filtered data, toggle Show/Hide Objects on the Display rollout to hide the biped. Toggle Show/Hide Bones in the same rollout to display only biped bones (yellow stick figure) then play the animation with Show Buffer turned on. The two stick figures move together, and any discrepancies are easily spotted.

To learn how to use Show Buffer with the Fit to Existing parameter in Motion Capture Conversion Parameters rollout, see *To use Fit to Existing to import a motion capture file* (see page 193).

### To import a motion capture file

1. Select a biped in the viewpoints.
2. Click Load Marker Name File on the Motion Capture rollout to load a marker name file (.mmm).

This step is not required if the marker or joint names in the motion capture file adhere to the Character Studio marker naming convention.

3. Select Load Motion Capture File.
4. Choose a file type: .bvh, .bip or .csm.
5. Select a file and click Open.

The *Motion Capture Conversion Parameters* dialog (see page 192) displays.

6. Select the filter options you want and click OK.

Note: Load raw marker data (No Key Reduction, Freeform) to enable the marker calibration buttons.

The biped adapts itself to the motion data. If Footstep Extraction is turned on, footsteps appear.

**Tip:** Use a biped that does not have a mesh attached with Physique. Import motion capture data with the idea of then saving a .bip file that can be used for any character. If skeletal scale information is loaded from a motion capture file, a mesh with the Physique modifier applied may deform unnaturally.

7. If a marker file was loaded, turn on Show Markers as a visual aid for biped scale and limb correction.

If correction is necessary, adjust biped scale first. Keyframe adaptation takes place in order to accommodate a biped scale change. The remaining steps in this procedure are optional, unless you need to calibrate motion capture files.

8. Click Talent Figure Mode and use Non-Uniform Scale or Rubber Band Mode (on the General rollout) to size the biped to the displayed markers.
9. Click Talent Figure Mode again to exit the mode.

Key Adaptation takes place when you exit Talent Figure mode. Now, biped limb

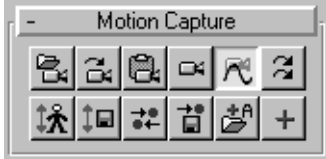
positions relative to the markers can be adjusted.


10. Align the biped limbs to the markers, if necessary, and then click Adjust Talent Pose to compute the offset for the entire animation.
11. Use Save Talent Figure Structure and Save Talent Pose Adjustment to save a size and position offset to a *.fig* and a *.cal* file, respectively.

Load these files in the Motion Capture Conversion Parameters dialog when you import similar marker files in the future.

At this point, you can use Convert from Buffer to extract footsteps and reduce keyframes. Both scale and position adjustments will be incorporated. Save the motion as an optimized *.bip* file.

## Interface



 **Load Motion Capture File.** Key reduce and extract footsteps from raw motion capture data. Load *.bip*, *.csm*, or *.bvh* files. After a file is selected, the *Motion Capture Conversion Parameters dialog* (see page 192) displays. Select filter parameters in this dialog.

**Set lowest starting foot height to Z=0 (.bip files only).** Sets the lowest starting foot height to Z=0. This is an option in the Load File dialog. Default=On.

In **character studio 3** the height of a motion clip can be retained. This is important if you want to retain the height of a motion clip for motions

adapted to characters of different sizes. If, for example, the character is jumping of a rock and you want to retain the Z position of the character, you would turn this option off. Leave this option off if Motion Flow motions must be blended that begin and end at different heights, such as three clips that have the character mounting a bicycle, riding the bicycle, and dismounting the bicycle.

Turning off this option can, however, cause a jump in the motion during motion flow transitions. Turn this on for smooth transitions in Motion Flow mode. If adaptation takes place, the height is set so that the lowest foot at frame 0 starts at the Z=0 height. This lines up clips along the Z axis and creates smooth transitions.

### **Restructure biped to match file (.bip files only).**

Turn on to change the biped structure to match the structure stored in the *.bip* file. The file loads with the stored biped structure. This is an option in the Load File dialog.

This option is unavailable when you load a *.bip* file into a clip or onto a biped that is in Edit Clip mode because all the clips in the motion flow would have to be adapted.

Note: *.csm* and *.bvh* files always load with the biped structure stored in the file.

**Set lowest starting foot height to Z=0 (.bip files only).** Sets the lowest starting foot height to Z=0. This is an option in the Load File dialog.

In **character studio 3** the height of the unadapted motion is normally retained. This is important if you want to retain the height of a motion clip for motions adapted to characters of different sizes. This can, however, cause a jump in the motion during motion flow transitions. Turn this on for smooth transitions in Motion Flow mode. If adaptation takes place, the height is set so that the lowest foot at frame 0 starts at the Z=0 height. This lines up clips on the Z axis.

**.bip:** Filters the raw version of the motion capture data that ships with Character Studio. These are in a *.bip* format. Filter standard *.bip* files to convert footstep animation to a freeform animation; extract footsteps from a freeform animation; and to loop a *.bip* file.

**.bvh:** Biovision motion capture data file. Contains the “actor’s” skeletal and motion information. Once the motion capture data is filtered and adjusted, save it as a *.bip* file for later use.

**.csm:** Imports a Character Studio marker file (ASCII format file). Optionally load a Marker Name file (*.mmm*), a Talent Structure file (*.fig*), and Talent Pose file (*.cal*) before loading a *.csm* file. Marker files should be loaded with no key reduction and no footstep extraction to enable the calibration controls.

First load raw marker data and turn on Show Markers to help you decide if calibration is necessary. If both scale and position calibration are necessary, calibrate scale first (Talent Figure mode) and then calibrate limb position. Character Studio adapts biped keys after biped scale is changed in Talent Figure mode, then orient the biped limbs relative to the markers and click Adjust Talent Pose to apply this offset to the entire animation. Save Talent Figure Structure and Talent Pose to a *.fig* and a *.cal* file respectively. Load a *.fig* and a *.cal* before loading a marker file that requires the scale and position offsets contained in these files.

Note: Calibration files can be loaded in the Motion Capture Conversion Parameters dialog before filtering marker files.



**Convert From Buffer.** Filters the most recently loaded motion capture data. This data is stored in the motion capture buffer. Displays the Motion Capture Conversion Parameters dialog.

The most recently imported motion capture file is stored in its raw form in the motion capture buffer. Convert From Buffer provides a quick way to try new conversion parameters in the Motion Capture Conversion Parameters dialog.



**Paste From Buffer.** Pastes a frame of raw motion capture data to the selected parts of the biped.

After importing a motion capture file, you may discover a subtle movement has been lost in the process of reducing keyframes. Paste From Buffer can add a keyframe from the raw motion capture data to a selected biped body part to restore this motion. Turn on the Animate button before using Pasting From Buffer, or press Set Key after using Pasting From Buffer to store the new position in a key.



**Show Buffer.** Displays raw motion capture data as a red stick figure.

Compare raw and filtered motion capture data using Show Buffer. Ideally, the motion of the biped and the red stick figure are very similar. If this is not the case, alter the filter parameters and import the motion capture file again or select a biped object and use Paste from Buffer at selected frames to restore the lost motion.

Raw motion capture data is buffered for the currently loaded or most recently imported motion capture file, allowing easy comparison of the raw and filtered motion data. Show Buffer displays a red stick figure representing the raw buffered data; compare this to the filtered motion of the biped during playback.



**Show Buffer Trajectory.** Displays buffered raw motion capture data as yellow trajectories for the selected biped body parts.

Use Show Buffer Trajectory to display a trajectory based on the buffered raw motion capture data for any biped body part. Use this in combination with Show/Hide Trajectories on the Display rollout to see how closely the raw and filtered data match.

This assumes a motion capture file has been imported.



**Batch File Conversion.** Converts one or more *.csm* or *.bvh* motion capture files to filtered *.bip* files. Displays the *Motion Capture Batch File Conversion* dialog (see page 197).



**Talent Figure Mode.** After loading a raw marker file, turns on Talent Figure mode to scale the biped relative to the markers. Calibration for the entire marker file takes place when you exit Talent Figure mode.

Keyframe adaptation takes place in order to accommodate the new biped scale; because of this, you should adjust the biped scale before adjusting the biped position relative to the markers.

Use Rubber Band Mode on the General rollout and Non-Uniform Scale to size the biped in Talent Figure mode.

Ideally, you will not need to use this feature. When loading a motion capture file, Biped attempts to extract the appropriate figure scale from the given data. Use Talent Figure mode only if the extracted scale of the biped doesn't match the scale of the original talent. Minor differences in scale will alter the motion.

Note: Calibration controls are only enabled when a marker or *.bvh* file is imported in its raw form. Do not use key reduction or extract footsteps when you import a marker file for the first time.



**Save Talent Figure Structure.** After changing the biped scale in Talent Figure mode, stores the changes into a *.fig* file. Use this file in the Motion Capture Conversion Parameters dialog to adjust marker files created by the same actor.



**Adjust Talent Pose.** After loading a marker file, use Adjust Talent Pose to correct the biped position relative to the markers. Align the biped limbs to the markers then click Adjust Talent Pose to compute this offset for all the loaded marker data.

Note: Calibration controls are only enabled when a marker or *.bvh* file is imported in its raw form. Do not use key reduction or extract footsteps when you import a marker file for the first time.



**Save Talent Pose Adjustment.** Saves a Talent Pose adjustment as a *.cal* file.

Save a *.cal* file after adjusting the biped relative to the markers. A *.cal* file is used for processing marker files that require the same adjustment. A *.cal* file can be loaded in the Motion Capture Conversion Parameters dialog during marker file importation.



**Load Marker Name File (.mmm).** Load a Marker Name file to map incoming marker names in motion capture files (*.bvh* or *.csm*) to the Character Studio marker naming convention. Displays the Marker Name File dialog.

**Load a CSM marker file.** Browses for a marker file for use with a CSM file.

**Load a BVH marker file.** Browses for a marker file for use with a BVH file.

When a BVH file is loaded, checks for and reports unknown track names - but loads the file. Reports if any required tracks were not in the file and if so aborts the file load.

**Use.** Uses the marker name file when importing motion capture files

If necessary, load a Marker Name File before loading a BVH or CSM file. Marker Name files are bundled with Character Studio to map marker names in popular third party marker files. Edit these ASCII files if the marker files you have use unique names for markers.

Note: See *BVH File Specification* (see page 199) and *CSM File Specification* (see page 213) for more information on the Marker Name File format and Character Studio-supported makers and joint names.



**Show Markers.** Opens the *Marker Display dialog* (see page 197), with settings for specifying how markers are displayed.

Marker and marker names are displayed around the biped. You can use these to spot and adjust discrepancies like the biped elbow position relative to the elbow marker. For information on how to correct these discrepancies, see *Talent Figure mode* (see page 191) and *Adjust Talent Pose* (see page 191).

## See also

*Character Studio Marker Files* (see page 198)

---

## Motion Capture Conversion Parameters Dialog

Create or select a biped. > Motion panel > Motion Capture rollout > Load Motion Capture File button > Open a file.

Load motion capture data. > Motion panel > Motion Capture rollout > Convert from Buffer button

Motion capture and marker data typically have keys at every frame. Filtering motion capture data reduces keys, making the job of altering or personalizing the motion data much simpler. Other filtering options include footstep extraction, applying the skeletal structure stored in the motion capture file to the biped, looping the data, importing a portion of the motion capture file and selecting tracks to load.

## Motion Capture Buffer

Any file imported using Load Motion Capture File is stored in its raw (pre-filtered) form in the motion capture buffer. This buffer is used to try new filtering options with the Convert from Buffer command, and to paste keys from the raw motion capture data to the biped using Paste from Buffer on the Motion Capture rollout. The Show Buffer command displays a stick figure that represents the buffered data.

Create your own library of imported and optimized motion capture data by saving *.bip* files for use with other characters or as part of a longer script in Motion Flow mode. Use a biped that has no mesh attached with Physique. You import the data, adjust it to your liking, and save it as a *.bip* file. You can also run standard *.bip* files through this filtering process to create loops or to extract footsteps from a freeform animation.



Note: Marker files contain **position** data. Regular motion capture files contain joint **rotation** data.

## Procedure

### To use Fit to Existing to import a motion capture file

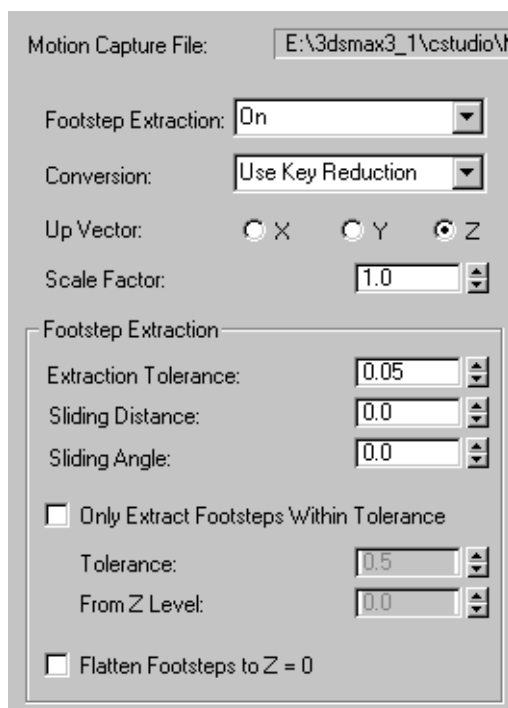
Use a motion capture file that contains footsteps and other motion. A handspring motion would be ideal for this example.

1. Select a biped.
2. Click Motion panel > Motion Capture rollout > Load Motion Capture File.
3. Select a motion capture file.  
The Motion Capture Conversion Parameters dialog displays.
4. Set Footstep Extraction to On, then click OK.  
The motion capture file loads.
5. Select Motion Capture rollout > Show Buffer to display raw motion capture data as a red stick figure, then select Play.  
During the “hand spring” period of the playback, the hands on the red stick figure representing the raw motion data touch the ground. The biped, using the filtered data, is positioned higher and cannot reach the ground.  
Dynamics calculates the biped higher than the raw motion data because the footsteps before and after the handspring are so far apart in time. Creating a freeform period in Track View and reloading the same file using the Fit to Existing option will match the biped position to the motion capture position during the handspring.
6. Find the Footsteps track for the biped in Track View.
7. Right-click in the footsteps area of the Track View Edit window, then select Edit Free

Form (No Physics) in the Footstep Mode dialog.

8. Click the “handspring” area between the footsteps. It turns to a solid yellow.  
Motion capture data will replace the data in this freeform period.
9. Reload the same motion capture file using the Fit to Existing option in the Motion Capture Conversion Parameters dialog.  
The freeform area created in Track View is replaced with motion capture data. The biped closely matches the red stick figure during the handspring part of the playback.

## Interface



**Motion Capture File.** Displays the file to be imported.

**Footstep Extraction.** Motion capture data can be applied to the biped in one of three ways:

- **None: Freeform.** No footsteps are extracted. For swimming or flying motion data, footstep extraction is not necessary. For a traditional approach to character keyframing, use this option to keyframe the biped without footsteps or Biped Dynamics; this is essentially a freeform animation.
- **On.** Extracts footsteps. Direction and style of the motion capture data are easily edited. Allows changes to the toe structure of the biped after import; footsteps will readjust the character's motion to maintain correct foot-toe-ground contact at all times, a common problem associated with motion capture import. Inappropriate "sliding feet" in the motion data are corrected.
- **Fit to Existing.** Fits to existing footsteps. Use with motion data that has both footstep motion and flying, swimming, falling, or tumbling motions. First load the motion data using Perform Footstep Extraction. Create a freeform period for the flying, swimming, or falling portion of the data in Track View, and then reload the same motion capture file using the Fit to Existing option. The freeform area is loaded with data from the motion capture file without the influence of biped dynamics.

**Conversion.** Chooses the type of key processing.

- **Use Key Reduction.** Reduces keys for simpler key editing.
- **No Key Reduction.** Does not reduce keys. Use this on files that are already key reduced

or if you want to work with all the data in a raw motion capture file.

Note: Marker files imported for the first time should be loaded with no key reduction or footstep extraction to enable the calibration controls on the Motion Capture rollout.

- **Load Buffer Only.** Does not apply the data to the biped, but loads the data to the motion capture buffer only. Use this either to compare your edited version with the original or to paste postures from the motion capture buffer to the biped in the scene.

Note: To compare the filtered data with raw motion capture data, use Show Buffer on the *Motion Capture* rollout (see page 186) rather than turning off key reduction.

**Up Vector.** Sets the vertical axis used in the motion capture data.

**Scale Factor.** Multiplies the stored talent size by this value and size the biped accordingly.

## Footstep Extraction group

Options here are active when Footstep Extraction is on.

**Extraction Tolerance.** Sets the sensitivity of footstep extraction. Biped determines if the footstep is there by checking that the foot does not move beyond the distance determined by the Extraction Tolerance value. Smaller numbers are more sensitive and extract more footsteps. The value is a percentage of foot length.

The default value is 0.15. Increase this value to .2 or .25 if too many footsteps are generated.

**Sliding Distance.** Creates a *sliding footstep* (see page 62) when positional tolerance is reached. This value is a percentage of foot length. By default the foot must slide its own distance (100), before a sliding footstep is created.

Use this with motion capture files that contain sliding feet. A sliding footstep can be created manually by setting IK Blend > 0 for a biped foot at a “touch” state key (biped foot first touches a footstep).

Note: Sliding footsteps display in a gold color.

**Sliding Angle.** Creates a *sliding footstep* (see page 62) when rotational tolerance is reached. This value is in degrees. The default is set high (360 degrees), the foot must make a complete turn before a sliding footstep is created.

Use this with motion capture files that contain feet that pivot, as in a dance motion.

Note: Sliding footsteps display in a gold color.

**Only Extract Footsteps Within Tolerance.** Turns on Z axis Tolerance. These controls filter out footsteps that do not fall within a given range of the ground plane. Use this when filtering motions, such as hopping or pitching a baseball, in which a foot may come off the ground and remain stationary, but its position is not intended as a footstep.

**Tolerance.** Value is a percentage of leg length.

**From Z Level.** Set a Z value (ground).

**Flatten Footsteps to Z=0.** Moves extracted footsteps to Z=0. Use this to flatten out minor differences in the height of the extracted footsteps.



**Start.** Start importing at this frame. Default is frame 0, the first frame.

**End.** Stop importing at this frame. Default is the last frame of the clip.

**Loop.** Loop the data by the value set here.

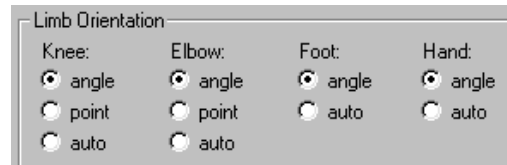
This is relative. Succeeding loops start where the previous loop left off. The clips are not blended and may require editing unless the original clip was designed to loop.

Use this for clips designed to loop.

Note: This often works best if Footstep Extraction is tuned off.

## Limb Orientation

The biped elbow and knee hinge joints are perpendicular to the triangles formed by the shoulder-elbow-wrist and hip-knee-ankle respectively. Resolve errors in the motion capture data that break this rule by using either the angle or point method.



**Angle.** Moves the knee or Elbow position to create the biped joint key.

**Point.** Rotates the shoulder-elbow-wrist or hip-knee-ankle to create the biped joint key.

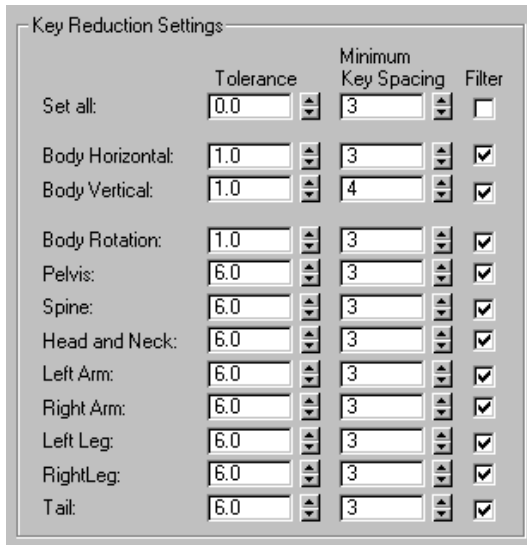
**Auto.** Auto reads exact hand and foot positions from the motion capture data, Character Studio then places the knees and elbows in a natural position. For marker files involving running and walking, this option can clean up the data nearly instantly, regardless of how many markers were used and where they were placed.

## Key Reduction Settings group

Key reduction keeps the original motion intact and intelligently filters out more than 80% of the keys in the motion capture file, making the process of altering the biped animation much simpler.

Use the Tolerance and Minimum Key Spacing settings to fine-tune key reduction for a specific track (body part). Specify whether or not to filter a track in the Filter column.

These settings are available only when Conversion is set to Use Key Reduction.



The Key Reduction Settings dialog box contains a table with columns for Tolerance, Minimum Key Spacing, and Filter. The rows represent different body parts. The 'Set all' row has values 0.0, 3, and an unchecked checkbox. The 'Body Horizontal' row has 1.0, 3, and a checked checkbox. The 'Body Vertical' row has 1.0, 4, and a checked checkbox. The 'Body Rotation' row has 1.0, 3, and a checked checkbox. The 'Pelvis' row has 6.0, 3, and a checked checkbox. The 'Spine' row has 6.0, 3, and a checked checkbox. The 'Head and Neck' row has 6.0, 3, and a checked checkbox. The 'Left Arm' row has 6.0, 3, and a checked checkbox. The 'Right Arm' row has 6.0, 3, and a checked checkbox. The 'Left Leg' row has 6.0, 3, and a checked checkbox. The 'Right Leg' row has 6.0, 3, and a checked checkbox. The 'Tail' row has 6.0, 3, and a checked checkbox.

	Tolerance	Minimum Key Spacing	Filter
Set all:	0.0	3	<input type="checkbox"/>
Body Horizontal:	1.0	3	<input checked="" type="checkbox"/>
Body Vertical:	1.0	4	<input checked="" type="checkbox"/>
Body Rotation:	1.0	3	<input checked="" type="checkbox"/>
Pelvis:	6.0	3	<input checked="" type="checkbox"/>
Spine:	6.0	3	<input checked="" type="checkbox"/>
Head and Neck:	6.0	3	<input checked="" type="checkbox"/>
Left Arm:	6.0	3	<input checked="" type="checkbox"/>
Right Arm:	6.0	3	<input checked="" type="checkbox"/>
Left Leg:	6.0	3	<input checked="" type="checkbox"/>
Right Leg:	6.0	3	<input checked="" type="checkbox"/>
Tail:	6.0	3	<input checked="" type="checkbox"/>

**Tolerance.** Sets the maximum angular or positional deviation for a track.

Values are in units of translation for position tracks, and in degrees for rotation tracks.

**Minimum Key Spacing.** Set the minimum number of frames between keys.

Tolerance is computed first, then Minimum Key Spacing computes further key reduction.

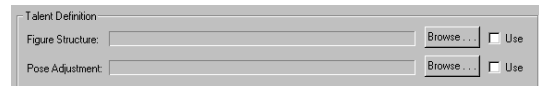
A Minimum Key Spacing value of 10 for the head track ensures that no two keys are closer than 10 frames for this track.

**Set All.** Forces all tracks to the values set in these fields.

Higher values here can determine how much key reduction is possible while preserving the original motion.

**Filter.** Turn off to prevent filtering of the motion capture data into a track. When this is off, there is no key reduction for the track.

## Talent Definition group



The Talent Definition dialog box has two rows. The first row is 'Figure Structure' with a 'Browse...' button and a 'Use' checkbox. The second row is 'Pose Adjustment' with a 'Browse...' button and a 'Use' checkbox.

Loads a Figure Structure File (.fig) and a Pose Adjustment file (.cal) prior to importing a marker file. Typically you correct a marker file by importing it and adjusting the biped scale and limb positions relative to the markers, then saving .fig and a .cal file using Save Talent Figure Structure and Save Talent Pose Adjustment on the Motion Capture rollout. These files can then be loaded in the Talent Definition area when importing marker files created by the same actor in a motion capture session.

**Figure Structure.** Load a .fig file.

**Pose Adjustment.** Load a .cal file.

**Browse.** Browse for a .fig and .cal file.

**Use.** Use either or both the .fig and .cal files to adjust marker files during a marker file import procedure.

**Load Parameters.** Loads a motion capture parameter file (.moc).

**Save Parameters.** Saves a motion capture parameter file (.moc).

## Motion Capture Batch File Conversion Dialog

Create or select a biped. > Motion panel > Motion Capture rollout > Batch File Conversion button

Convert one or more *.csm* or *.bvh* motion capture files to filtered *.bip* files.

**Source File Selection.** Selects motion capture files to convert. Use Shift-click to select all of the files between two selected files. Use Ctrl-click to add individual files to the selection.

**Destination Directory.** Specify the directory in which the software is to store the filtered files.

**Specify Conversion Parameters Once.** Imported files will use a single set of conversion parameters.

**Specify Parameters For Each File.** Specify conversion parameters for each file.

## Marker Display Dialog

Create or select a biped. > Motion panel > Motion Capture rollout > Show Markers button

The Marker Display dialog lets you specify how markers from *.csm* files are displayed in the viewports. For further information on markers and *.csm* files, see *Character Studio Marker Files* (see page 198).

## Interface



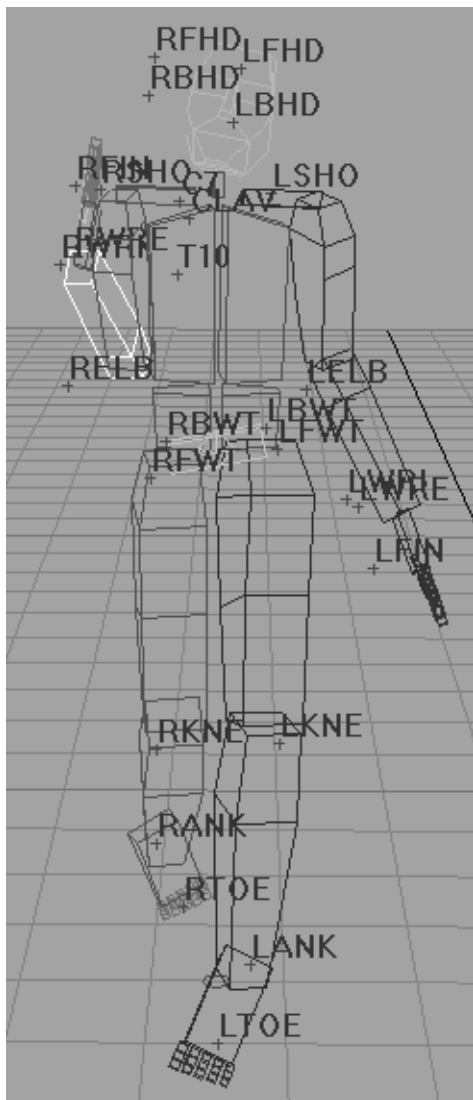
**Show Recognized Markers.** Enables the display of markers that Character Studio recognizes.

**On Selected Objects/On All Objects.** Determines whether markers appear on selected objects or on all objects.

**Show Prop Markers.** Enables the display of markers on *prop bones* (see page 198).

**Show Unrecognized Markers.** Enables the display of markers that Character Studio does not recognize.

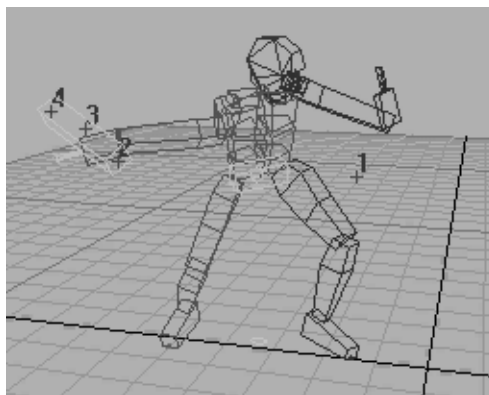
## Character Studio Marker Files



The Show Markers command displays marker positions and names.

The CSM format is an ASCII file used to import positional marker data from motion capture systems onto a biped. See *CSM File Specification* (see page 213) for the .csm file specification.

## Prop Bone



The .csm marker file format now supports a prop bone in either or both hands. There are six additional markers for the top, bottom, and middle of the two props. If these tracks are detected, a 3DS MAX dummy object is created.

The length of the prop is the average distance between the top and bottom prop marker during animation. The prop will be oriented in the plane of the three prop markers, and its origin will be at the bottom prop marker.

The dummies will be bip01prop, bip02prop, and so on. If a prop of that name already exists, its animation and size is reset upon loading of the .csm file. There is a checkbox in the Marker Display dialog for displaying the prop markers. The new prop track names are LPRPB, LPRPM, LPRPT, RPRPB, RPRPM, and RPRPT. These stand for left prop bottom, left prop top, left prop middle, right prop bottom, right prop top, and right prop middle. Track names can be altered via the .mmm

file. The props don't have to have any relation to the left and right hands. For example, a prop could be a hat.

---

## BVH File Specification

This document describes the Character Studio BVH motion capture file format. The BVH BioVision format is an ASCII file that is used to import rotational joint data from various motion capture systems into Character Studio to animate bipedal characters.

Biped provides direct input of *.bvh* files from disk, including comprehensive keyframe reduction and footstep extraction to provide a fast and accurate means of import for large volumes of rotational data stored in the BVH format.

Biped uses rotational data stored in the BVH format to pose a Biped character on a frame-by-frame basis and to move it forward in time. During import, XYZ Euler joint rotations stored in the BVH file are used to derive Quaternion bone rotation data for the biped at each frame.

Once imported, BVH-based animations can be saved out as native Biped *.bip* files, providing access to a comprehensive set of animation, motion mapping, and structural modification features that are built directly into Biped.

## Overview of BVH-supported Hierarchy and Naming

Biped supports a specific BVH hierarchy and naming scheme, as generated by BioVision Motion Capture Studios. This specific hierarchy allows for automatic mapping of BVH data to the Biped character.

Parsing order of sibling nodes in the hierarchy is arbitrary. That is, the order of sibling nodes can be altered as long as their parent-child relationships do not change.

The supported hierarchy and required node/bone names are listed below. Note that an alternative naming scheme for some nodes is shown in (). The {} notation is used to clarify the parent-child hierarchy between nodes.





This list can be easily copied and modified with the right column for each unique motion capture session sample.

Hips	
LeftHip	femur
LeftKnee	tibia
LeftAnkle	foot
RightHip	rfemur
RightKnee	rtibia
RightAnkle	rfoot
Chest	upperback
chest2	thorax
LeftCollar	shoulderjoint
LeftShoulder	humerus
LeftElbow	radius
LeftWrist	wrist
RightCollar	rshoulderjoint
RightShoulder	rhumerus
RightElbow	rradius
RightWrist	rwrist
head	head
neck	neck

## Establishing a Correct Neutral Pose

In creating a BVH file that can be read into Character Studio Biped and displayed correctly, it is critical that the effective neutral pose of the BVH match Biped's neutral pose. That is, the hierarchy, bone lengths, and initial rotations specified in the HIERARCHY section of the BVH file must create a figure that matches the default pose of a newly-created Biped figure. The initial pose for Biped places the biped figure upright along the +Z axis, facing "forward" along the -Y axis, left hand on the +X axis. Hands are oriented with palms against the outside thighs, with thumb on the forward side; fingers and thumb of each hand outstretched

(open) facing "down" along the -Z axis. Legs are together and straight at the knees.

A correctly oriented BVH file can be verified by setting all rotational values in the first frame of the MOTION section to zeroes. When all joint rotation values are set to zero for the first frame of the BVH file, the results displayed at frame zero in Biped, after import, should match the default position of the Biped exactly.

## General BVH File Structure

The BVH file is divided into two major sections: HIERARCHY and MOTION.

The HIERARCHY section describes the joint-to-joint connections and offsets for the sampled motion data.

The MOTION section describes the movement of these individual joints on a per-sample basis.

CHANNELS information listed within a specific node of the HIERARCHY section signals the existence of corresponding XYZ data streams in the MOTION section which follows.

CHANNELS information represents the bulk of the BVH file data.

In the MOTION section, each row contains data values for all CHANNELS which were specified in the HIERARCHY. Each successive row contains a single time sample for all CHANNELS. The listing order of CHANNELS values in each row in the MOTION section is implicitly assumed to match their listed order from the HIERARCHY section (top down).

All string names must appear exactly as shown below, with the exception of nodes names (for example, LeftHip, LeftKnee) for which alternative naming schemes have been listed above.

CHANNELS data that corresponds to unrecognized JOINT names will be ignored.

The [] notation indicates values or strings that should be used in the actual file.

```

----- File Starts Here -----
HIERARCHY
ROOT Hips
{
    OFFSET          [x_float]          [y_float]          [z_float]
    CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
    JOINT LeftHip
    {
        OFFSET          [x_float]          [y_float]          [z_float]
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT LeftKnee
        {
            OFFSET          [x_float]          [y_float]          [z_float]
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT LeftAnkle
            {
                OFFSET          [x_float]          [y_float]          [z_float]
                CHANNELS 3 Zrotation Xrotation Yrotation
                End Site
                {
                    OFFSET [x_float]          [y_float]          [z_float]
                }
            }
        }
    }
    JOINT RightHip
    {
        OFFSET          [x_float]          [y_float]          [z_float]
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT RightKnee
        {
            OFFSET          [x_float]          [y_float]          [z_float]
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT RightAnkle
            {
                OFFSET          [x_float]          [y_float]          [
z_float]

                CHANNELS 3 Zrotation Xrotation Yrotation
                End Site
                {
                    OFFSET
[x_float]          [y_float]          [z_float]
                }
            }
        }
    }
    JOINT Chest

```

```

{
    OFFSET [x_float y_float z_float]
    CHANNELS 3 Zrotation Xrotation Yrotation
    JOINT LeftCollar
    {
        OFFSET [x_float y_float z_float]
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT LeftShoulder
        {
            OFFSET [x_float y_float z_float]
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT LeftElbow
            {
                OFFSET [x_float y_float z_float]
                CHANNELS 3 Zrotation Xrotation Yrotation
                JOINT LeftWrist
                {
                    OFFSET [x_float y_float
z_float]
                    CHANNELS 3 Zrotation Xrotation
Yrotation
                    End Site
                    {
                        OFFSET [x_float y_float
z_float]
                    }
                }
            }
        }
    }
}
JOINT RightCollar
{
    OFFSET [x_float y_float z_float]
    CHANNELS 3 Zrotation Xrotation Yrotation
    JOINT RightShoulder
    {
        OFFSET [x_float y_float z_float]
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT RightElbow
        {
            OFFSET [x_float y_float z_float]
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT RightWrist
            {
                OFFSET [x_float y_float z_float]
                CHANNELS 3 Zrotation Xrotation
Yrotation
                End Site
                {

```



```

17.950001      0.000000      OFFSET      0.000000      -
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                End Site
                                {
                                OFFSET 0.000000      -
3.119996      0.000000
                                }
                                }
                                }
                                }
                                JOINT RightHip
                                {
                                OFFSET      -3.430000      0.000000      0.000000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT RightKnee
                                {
                                OFFSET      0.000000      -
18.809999      0.000000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT RightAnkle
                                {
                                OFFSET      0.000000      -
17.570000      0.000000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                End Site
                                {
                                OFFSET      0.000000      -
3.250000      0.000000
                                }
                                }
                                }
                                }
                                }
                                JOINT Chest
                                {
                                OFFSET      0.000000      4.570000      0.000000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT RightCollar
                                {
                                OFFSET      -
1.060000      15.330000      1.760000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT RightShoulder
                                {
                                OFFSET      -
6.060000      0.000000      0.000000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT RightElbow
                                {

```

```

OFFSET          0.000000      -
11.900000      0.000000

CHANNELS 3 Zrotation Xrotation Yro-
tation

JOINT RightWrist
{
OFFSET          0.000000      -
9.520000      0.000000

CHANNELS 3 Zrotation Xro-
tation Yrotation

End Site
{
OFFSET          0.000000      -
7.140012      0.000000

}

}

}

JOINT LeftCollar
{
OFFSET          1.060000      15.330000      1.76
0000

CHANNELS 3 Zrotation Xrotation Yrotation
JOINT LeftShoulder
{
OFFSET          5.810000      0.000000      0.000000
CHANNELS 3 Zrotation Xrotation Yrotation
JOINT LeftElbow
{
OFFSET          0.000000      -
12.080000      0.000000

CHANNELS 3 Zrotation Xrotation Yro-
tation

JOINT LeftWrist
{
OFFSET          0.000000      -
9.820000      0.000000

CHANNELS 3 Zrotation Xro-
tation Yrotation

End Site
{
OFFSET          0.00
0000      -7.369996      0.000000

}

}

}

}

```

```

        JOINT Neck
        {
            OFFSET      0.000000      17.620001      0.000000
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT Head
            {
                OFFSET      0.000000      5.190000      0.000000
                CHANNELS 3 Zrotation Xrotation Yrotation
                End Site
                {
                    OFFSET      0.000000      4.140008      0
                    .000000
                }
            }
        }
    }
}
MOTION
Frames:      222
Frame Time: 0.033333
  0.00 39.88-0.01-1.79-18.43-1.74 5.02-0.34 0.03 6.61 42.19 7.67-3.87-7.61 1.40 3.65
15.11-0.95 2.33 11.06-15.20-7.25-10.08 1.61 4.89 18.33 11.12-17.68 0.00 0.60 40.98 9.99
22.36 0.00-30.82 11.92 0.00 0.00 0.00 10.97 0.00 12.96-37.45-2.92 9.36-180.00-80.59
157.44 0.00 0.00 0.00-35.59 52.18 0.00 11.29-39.90 61.74
  0.11 39.87-0.01-2.31-17.29-3.05 3.19-4.22 0.24 7.38 40.49-1.01-3.87-7.94 1.38 3.46
13.75-0.82 2.47 11.44-14.02-7.25-10.10 1.61 5.82 17.00 12.55-17.36 0.00 2.00 42.28
10.77 34.97-0.85-22.68 14.90 0.00 0.00 0.00 10.01 0.00 11.77-40.78-3.61 6.09-178.56-
87.20 158.39 0.00 0.00 0.00-36.14 51.71 0.00 10.04-38.83 61.13
  0.22 39.88 0.02-2.83-16.47-4.56 1.98-6.94 0.24 6.27 37.48-5.85-3.87-7.23 1.43 3.26
12.76-0.72 2.52 11.87-12.69-7.25-10.11 1.60 6.78 15.96 14.27-16.96 0.00 3.28 44.28
12.02 51.41-0.44-17.87 20.64 0.00 0.00 0.00 9.19 0.00 10.84-44.55-3.26 2.86-0.03-85.90
-23.56 0.00 0.00 0.00-36.74 50.88 0.00 9.12-37.86 60.76
  0.32 39.92 0.19-3.51-17.17-7.03 1.79-6.29 0.20 2.67 31.81-3.40-3.86-5.65 1.54 2.98
13.46-0.69 2.23 12.43-10.34-7.26-10.29 1.58 7.48 16.60 16.65-16.32 0.00 4.70 47.12
13.31 65.41 0.63-16.14 26.83 0.00 0.00 0.00 8.37 0.00 9.94-48.32-2.20-0.50 0.41-78.48-
23.57 0.00 0.00 0.00-37.20 49.91 0.00 8.35-37.07 60.23

```

## BVH Guidelines for Using the Biped MoCap Converter

### 1. SUPPORT FOR 3-LINKED SPINE

Biped supports both 2- and 3-linked spines in BVH format. The third spine link is called "Chest2".

Below is a simple 10-frame BVH sample file with the correct BVH syntax for the additional chest link: "Chest2". It is the same as the Chest syntax; it is simply added

as an additional child link bone going down vertically in Y before reaching the LeftCollar and RightCollar bones.

### 2. SUPPORT FOR NON-SYMMETRICAL LIMB BONE LENGTHS

Biped supports non-symmetrical arm and leg bone sizes. The example below is non-symmetrical.

### 3. GENERAL NOTES ABOUT GETTING PRECISE BVH CONVERSION OF LIMBS

BIPED has built in anthropomorphic constraints that require that:

Elbows and Knees are one degree-of-freedom hinge joints. The axis of rotation of the hinge should ideally always be perpendicular to the arm's shoulder-elbow-wrist triangle and the leg's hip-knee-ankle triangle. Therefore, it's ideal if the BVH Euler angles for the leg and arm bones match this orientation constraint with no added twists or deviations.

If the hinge joint constraint is violated, BIPED's converter must decide whether to satisfy the orientation data or satisfy the elbow/knee positioning based solely on the point locations.

In the Mocap Conversion Parameters Dialogue Box:

If the "Angle" radio button is specified, the limb's triangle attempts to match the hinge to the Euler matrix axes. This may deviate from the position posture.

If the "Point" radio button is specified, the limb's triangle matches the positions of the actual points. This may deviate from the specified Euler angle matrix.

However, in both cases, the limb always moves to hit the IK position of the wrist or ankle, so if Euler angles are given that are not aligned with the natural hinge joint, the axis is projected to nearest axis that satisfies the wrist/ankle IK constraint. That is, one axis must be normal to the line joining the arm's shoulder-wrist or the leg's hip-ankle.

The following can be detached and written as "any\_filename.bvh". This file is also included as *chest2.bvh*.

```
HIERARCHY
ROOT Hips
{
    OFFSET          0.00      0.00      0.00
    CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
    JOINT LeftHip
    {
        OFFSET          3.430000      0.000000      0.000000
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT LeftKnee
        {
            OFFSET          0.000000      -
            CHANNELS 3 Zrotation Xrotation Yrotation
            JOINT LeftAnkle
            {
                OFFSET          0.000000      -
                CHANNELS 3 Zrotation Xrotation Yrotation
                End Site
                {
                    OFFSET          0.000000      -
                    18.469999      0.000000
                    17.950001      0.000000
                    3.119997      0.000000
                }
            }
        }
    }
}
```



```

    }
  }
  JOINT RightHip
  {
    OFFSET      -3.430000      0.000000      0.000000
    CHANNELS 3 Zrotation Xrotation Yrotation
    JOINT RightKnee
    {
      OFFSET      0.000000      -18.469999      0.000000
      CHANNELS 3 Zrotation Xrotation Yrotation
      JOINT RightAnkle
      {
        OFFSET      0.000000      -17.950001      0.000000
        CHANNELS 3 Zrotation Xrotation Yrotation
        End Site
        {
          OFFSET      0.000000      -
3.119997      0.000000
        }
      }
    }
  }
  JOINT Chest
  {
    OFFSET      0.000000      4.570000      0.000000
    CHANNELS 3 Zrotation Xrotation Yrotation
    JOINT Chest2
    {
      OFFSET      0.000000      6.570000      0.000000
      CHANNELS 3 Zrotation Xrotation Yrotation
      JOINT LeftCollar
      {
        OFFSET      1.060000      15.330000      1.760000
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT LeftShoulder
        {
          OFFSET      5.810000      0.000000      0.000000
          CHANNELS 3 Zrotation Xrotation Yrotation
          JOINT LeftElbow
          {
            OFFSET      0.000000      -
12.080000      0.000000
            CHANNELS 3 Zrotation Xrotation
Yrotation
            JOINT LeftWrist
            {
              OFFSET      0.000000
-9.820000      0.000000
            }
          }
        }
      }
    }
  }

```

```

Xrotation Yrotation

CHANNELS 3 Zrotation
End Site
{
    OFFSET 0.0
}
00000 -7.369996 0.000000
}
}
}
}
JOINT RightCollar
{
    OFFSET -1.060000 15.330000 1.760000
    CHANNELS 3 Zrotation Xrotation Yrotation
    JOINT RightShoulder
    {
        OFFSET -
6.060000 0.000000 0.000000
        CHANNELS 3 Zrotation Xrotation Yrotation
        JOINT RightElbow
        {
            OFFSET 0.000000 -
11.080000 0.000000
        CHANNELS 3 Zrotation Xrotation Yro-
tation
        JOINT RightWrist
        {
            OFFSET 0.000000 -9.820000 0.000000
        CHANNELS 3 Zrotation Xro-
tation Yrotation
        End Site
        {
            OFFSET 0.000000
0 -7.140010 0.000000
        }
        }
        }
        }
    }
}
JOINT Neck
{
    OFFSET 0.000000 10.620001 0.000000
    CHANNELS 3 Zrotation Xrotation Yrotation
    JOINT Head
    {
        OFFSET 0.000000 5.190000 0.000000

```

MOTION

0.0 5.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 40.0 0.0		
0.0 0.0 0.0	-40.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -8.0 0.0	0.0 -8.0 0.0		
6.0 0.0 0.0	40.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -40.0 0	0.0 0.0 0.0	0.0 0.0 0.0	0.0 0.0 5.0
0.0 6.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 50.0 0.0		
0.0 0.0 0.0	-50.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -10.0 0.0	0.0 -10.0 0.0		
7.5 0.0 0.0	50.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -50.0 0	0.0 0.0 0.0	0.0 0.0 0.0	0.0 0.0 8.0
0.0 7.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 60.0 0.0		
0.0 0.0 0.0	-60.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -12.0 0.0	0.0 -12.0 0.0		
9.0 0.0 0.0	60.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -60.0 0	0.0 0.0 0.0	0.0 0.0 0.0	0.0 0.0 10.0
0.0 8.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 70.0 0.0		
0.0 0.0 0.0	-70.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -14.0 0.0	0.0 -14.0 0.0		
10.5 0.0 0.0	70.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -70.0 0	0.0 0.0 0.0	0.0 0.0 0.0	0.0 0.0 12.0
0.0 9.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 80.0 0.0		
0.0 0.0 0.0	-80.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -16.0 0.0	0.0 -16.0 0.0		
12.0 0.0 0.0	80.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -80.0 0	0.0 0.0 0.0	0.0 0.0 0.0	0.0 0.0 14.0
0.0 10.0 0.0	0.0 0.0 0.0		
0.0 0.0 0.0	0.0 90.0 0.0		
0.0 0.0 0.0	-90.0 0.0 0.0		
0.0 0.0 0.0	0.0 0.0 0.0		
0.0 -18.0 0.0	0.0 -18.0 0.0		

```

13.5 0.0 0.0      90.0 0.0 0.0
0.0 0.0 0.0      0.0 0.0 0.0
0.0 0.0 0.0      0.0 0.0 0.0
0.0 -90.0 0      0.0 0.0 0.0      0.0 0.0 0.0      0.0 0.0 16.0

```

## CSM File Specification

This document describes the Character Studio CSM Motion Capture File format. The CSM format is an ASCII file that is used to import positional marker data from various motion capture systems into Character Studio to animate bipedal characters.

Biped provides direct input of CSM files from disk, including comprehensive keyframe reduction, footstep extraction, and talent figure and pose calibration to provide a fast and accurate means of import for large volumes of positional data stored in the CSM format.

Biped uses positional data stored in the CSM format to pose a Biped character on a frame-by-frame basis and to move it forward in time. During import, xyz marker positions stored in the CSM file are used to derive bone rotation data for the biped at each frame, eliminating the need to convert positional marker data into rotational form prior to import.

Once imported, CSM-based animations can be saved out as native Biped *.bip* files, providing access to a comprehensive set of animation, motion mapping, and structural modification features that are built directly into Biped.

Note: The *.csm* marker file format now supports a prop bone in either or both hands. See *Prop Bone* (see page 198).

## Overview of CSM Workflow and General Use

Character Studio computes a joint-centered skeleton so that the hands and feet are positioned precisely over world-space hand and

feet markers, with inverse kinematics used to solve for the required joint angles.

In this approach, the hands and feet may be tracked with much higher accuracy and quality, but the calibration is much more critical since errors in positioning and bone size will propagate and accumulate through the hierarchical figure structure from the root to the hands/feet. For example, if the spine bones are longer, the shoulder will be higher, so the arm will have to stretch further to place the hand at a low world space height.

There are three distinct phases for processing CSM mocap data effectively:

1. Create "Talent Definition" files (*.cal* and *.fig*) for a given performer from a specially recorded "calibration pose motion."
2. Create *.bip* motions from the talent's *.csm* motions using the same Talent Definition files from step 1)
3. Adapt or "motion map" the talent's *.bip* motions to synthetic characters that are different than the talent

### MOTION CAPTURE CONVERSION WORKFLOW

```

-----
|-----|
|1) CALIBRATION | <-- Short calibration
|-----| pose motion .CSM file
|
|.CAL and .FIG files used as "Talent Defi-
nition")
|
v
|-----|
|2) PROCESSING | <-- Motion .CSM files
|-----| captured from person

```

with markers in Talent  
Definition

```

      |
(Talent figure's .BIP files)
      |
      v
|-----|
| 3) MOTION MAPPING |
|-----|
      |
      v
(Final target character's .BIP files)

```

### 1) CALIBRATION: CREATE TALENT DEFINITION FILES (.cal and .fig)

The calibration pose motion should ALWAYS be created with the feet flat and body in the vertical position. Ideally the legs should be straight with the arms outstretched to the side. There should be no marker drop-outs in this first frame (the user can skip bad data by setting the start frame at the first good frame), since frame 0 is used to get the default bone lengths. Note that (as always), calibration may only be employed with no key reduction or footstep extraction. It's advisable to use short segments for this stage (under 200 frames). The advantage to using a motion segment is that the user can see the effects of interactive manual calibration over the range of frames that are loaded. So, for example, a short motion that shows the arms slowly falling to the sides is best.

By default, the system automatically calibrates to make the root (center-of-mass) upright and the feet flat. The user should employ them in any order:

- “Talent Figure Mode” to make adjustments to bone length as needed by adjusting the bone lengths by non-uniformly scaling in x. In this mode, using Copy Posture/Paste Opposite Posture or the just using symmetrical adjustment is one way to insure that the right and left leg/arms are

the same (as with regular Figure Mode).

Upon leaving “Talent Figure Mode”, the figure will recalibrate to the motion with the new bone sizes. To repeat this, just enter Talent Figure Mode again and iterate until satisfied.

- “Adjust Talent Pose” allows the user to rotate and position body parts, including the root center-of-mass, to better fit the data. Because the calibration is hierarchical, rotations of the body may need some adjustments to the spine and feet. It should be pretty intuitive in any case, since what you see is what you get.

There are two modes for interactively calibrating the Talent Pose that are based on the settings in the `3DSMAX\Plugcfg\Biped.Ini` file.

- `MocapUseLegIK=1` means that “Adjust Talent Pose” will always snap the feet back to the world space markers.
- `MocapUseArmIK=1` means that “Adjust Talent Pose” will always snap the hands back to the world space markers.

If either value is 0, the hands or feet will not snap back. You may want to use this to offset hand and foot positions from their markers. Otherwise, at a value of 1, it's easier to fine tune other body parts with the assurance that the hands and feet will snap back in place when the talent pose is adjusted. Note that you may change the *biped.ini* file with a text editor such as Notepad or WordPad, save the file, and reload the *.csm* file with the new settings without rebooting MAX.

Once the Talent figures bone sizes and pose adjustments look good, that is, the figure fits correctly inside the markers, the user *must*:

- “Save Talent Figure Structure” to insure that this figure can be applied other files that use the same performer/marker\_placement.

This is saved as a *.fig* file format and will automatically appear with the correct path name in the mocap import dialogue box on the next mocap import.

- “Save Talent Pose Adjustment” to insure that the pose calibration can be applied other files that use the same performer/ marker\_placement. This is saved as a *.cal* file and will also automatically appear (with the correct path name) in the mocap import dialogue box on the next mocap import. Special Note: Saving the *.cal* is *required*, even if no interactive use of “Adjust Talent Pose” has been done in order to save automatic calibration information about the body’s waist and feet markers.

## 2) PROCESSING: CONVERT ALL MOTIONS THAT USE THE SAME TALENT DEFINITION STRUCTURE (*.fig*) and CALIBRATION (*.cal*)

Now use the *.cal* and *.fig* files in the Talent Definition section of the Mocap Conversion Parameters Dialogue Box to process all the files of that performer that use the marker placement of the calibration motion. For new marker placements or performers, the process should begin with PHASE 1 again. Of course, additional tweaking of the calibration may be performed for a specific file if desired.

Note: One advantage of using the same *.fig* talent structure is that no adaptation is computed if these files are loaded onto the same figure. This means that long keyframe/frame *.bip* files will load instantly. It will also guarantee consistency that one figure is performing all the moves.

Note that Batch mode may be used for this stage, especially if the goal is just to convert the raw files (frame per frame). Otherwise, each file will probably warrant fine tuning on the keyframe reduction/footstep extraction parameters. For cases with sliding motion, and sliding angle of 15 degrees is usually effective.

## 3) MOTION MAPPING: NOW APPLY MOTIONS TO SYNTHETIC CHARACTERS WITH DIFFERENT DIMENSIONS AND/OR LINK STRUCTURE

Once a clean *.bip* has been created with a figure that matches the actual performer in the mocap session, it may then be mapped onto other characters. Mapping motion to other characters will work with or without keyframe reduction or footstep extraction. In fact, raw data (keyframe per frame) with no footsteps maps extremely well in most cases.

## Summary of New Biped Variables in 3dsmax\plucfg\biped.ini for CSM Processing (as of R2.2)

Parameter (=default)	Range	Function
MocapUseLegIK=1	0 or 1	Snaps feet to markers during Adjust Talent Pose CSM calibration
MocapUseArmIK=1	0 or 1	Snaps hands to markers during AdjustTalent Pose CSM calibration
MocapVariableDynamics=1	0 or 1	Turns on automatic determination of Dynamics Blend for vertical keyframes
MocapHeightSmoothing=0	0 to 20	Smooths out captured body vertical height values to prevent jitter
MocapBodyHorzSmoothing=0	0 to 20	Smooths out captured body horizontal translation values to prevent jitter
MocapBodyRotSmoothing=3	0 to 20	Smooths out captured body rotation captured values to prevent jitter
MocapSpineSmoothing=6	0 to 20	Smooths out captured spine captured values to prevent jitter
MocapHeadSmoothing=3	0 to 20	Smooths out captured head captured values to prevent jitter

The MocapUseLegIK and MocapUseArmIK parameters are explained in the previous

paragraph.

MocapVariableDynamics will introduce gravity based biped dynamics for vertical keyframes that surround ballistic jumps between footsteps, but only when both Footstep extraction and Keyframe reduction are on, otherwise this parameter has no effect. If the gravity-based jump deviates significantly from the captured data, spline dynamics (that is, dynamics blend = 0) is used. This typically happens for falling down motions.

If MocapVariableDynamics is set to zero, the conversion will use either spline or biped dynamics for all vertical keyframes in accordance the user's settings in the Animation Properties in the Biped Motion Panel.

The Mocap\*\*\*\*\*Smoothing parameters smooth out jitters in the data, with higher values increasing the degree of smoothing. Values higher than 10 will begin to alter the more gross aspects of the motion, but values less than seven should only remove minor wobbles that sometimes occur due to small fluctuations in marker relationships. The defaults are the recommended values.

## CSM.MAX for Placement and Export of CSM Markers

An example of desired CSM marker placements may be seen by viewing the *csm.max* file in the CSTUDIO\DOCS folder on the release CD and after installation. This file may also be used as a starting point for importing csm files in cases where you wish to export edited csm motions using the *csmexport.ms* MAXScript utility found in the same folder. For export, you should

1. Load *csm.max*. You may adjust the marker positions as desired for your export to another application.



2. Load a mocap file and edit it with the MAX marker objects attached.
3. Run the *csmxport.ms* MAXScript utility for the active frames you want to export.

## Overview of CSM-supported Marker Attachment Positions

CSM marker values are time-varying xyz locations which are typically generated by reflective markers and optical motion capture systems. These xyz values may also be derived from other motion capture sensing devices that generate positional data in a world-coordinate system.

Biped supports a fixed set of known markers that correspond to reflective markers or sensors attached to different parts of a Human “talent” subject during a motion capture session.

Biped’s “superset” of supported markers was chosen to cover typical marker/sensor configurations used by various motion capture service companies and hardware vendors.

The complete set of Biped-supported marker names and attachments are listed below. Optional Markers are noted; these optional markers are not needed for import, but do contribute to a more accurate solution if they are present. All other makers are required for Biped to read a CSM file successfully.

## Character Studio Supported Marker Names and Attachments

Head	LFHD	Left Front Head
	LBHD	Left Back Head
	RBHD	Right Back Head
	RFHD	Right Front Head
Chest	CLAV	top chest
	STRN	center chest
Waist	LFWT	Left Front Waist
	LBWT	Left Back Waist
	RBWT	Right Back Waist
	RFWT	Right Front Waist
Spine	C7	Top of Spine
	T10	Middle of Back
	SACR	Lower Back (optional)
Left Leg	LKNE	Left Outer Knee
	LKNI	Left Inner Knee (Optional)
	LANK	Left Outer Ankle
	LHEL	Left Heel (Optional)
	LMT5	Left Outer Metatarsal
	LMTI	Left Inner Meta-tarsal (Optional)
	LTOE	Left Toe
Right Leg	RKNE	Right Outer Knee
	RKNI	Right Inner Knee (Optional)
	RANK	Right Outer Ankle
	RHEL	Right Heel (Optional)
	RMT5	Right Outer Metatarsal
	RMTI	Right Inner Meta-tarsal (Optional)

	RTOE	Right Toe
Left Arm	LSHO	Left Shoulder
	LELB	Left Outer Elbow
	LILB	Left Inner Elbow (optional)
	LWRE	Left Wrist Stick End
	LWRI	Left Wrist Stick Base
	LWRA	Left Wrist Inner near thumb (alternative to LWRE)
	LWRB	Left Wrist Outer opposite thumb (alternative to LWRI)
	LFIN	Left Hand
Right Arm	RSHO	Right Shoulder
	RELB	Right Outer Elbow
	RILB	Right Inner Elbow (optional)
	RWRE	Right Wrist Stick End
	RWRI	Right Wrist Stick Base
	RWRA	Right Wrist Inner near thumb (alternative to RWRE)
	RWRB	Right Wrist Outer opposite thumb (alternative to RWRI)
	RFIN	Right Hand

CSM files that utilize Biped marker names exactly as shown above may be read into Biped directly. Additionally, for convenience, the CSM format supports custom marker names for the known marker positions described above. These custom marker names must be correlated to the

preset marker names listed above using a Marker name file (MNM). That is, the Left Shoulder marker need not be named “LSHO”, as long as the CSM string “myleftshoulder” is matched to “LSHO” using the separate Marker name file MNM. For more details on the MNM file format, see below.

### Understanding Wrist Marker Alternatives

For specification of wrist markers, the CSM file may contain either of two types of wrist marker attachments, but not both.

For the first method, the wrist position/orientation may be specified by the use of a “stick” attached to the wrist at a 90 degree angle to the forearm, centered at the wristwatch position. These markers are used to describe the stick position:

LWRE	Left Wrist Stick End
LWRI	Left Wrist Stick Base
RWRE	Right Wrist Stick End
RWRI	Right Wrist Stick Base

For the second method, The RWRA--RWRB and LWRA--LWRB markers define the alternative markers that are on the inner and outer positions of the wrist (via a wristband). The RWRA/LWRA markers are on the inside (closer to thumb)

LWRA	Left Wrist Inner near thumb (alternative to LWRE)
LWRB	Left Wrist Outer opposite thumb (alternative to LWRI)
RWRA	Right Wrist Inner near thumb (alternative to RWRE)
RWRB	Right Wrist Outer opposite thumb (alternative to RWRI)

## Understanding Marker Placement for C7, CLAV, STRN, and T10

In general, the midpoints of the upper C7-CLAV and the lower STRN-T10 are used to guide the vertical direction of the chest bones, so it's important that these marker pairs reside at about the same body height in the right location.

The *fashion1.csm* file in the *motions\mocap\csm\Vicon* folder on the release CD is an example of where these markers should be. See below:

```
SIDE VIEW
=====
.----
|      \  -" head"
----- /
      | - "neck"
C7-CLAV ---- midpoint near top of spine
      | - "spine2"
T10-STRN --- midpoint at top of 2nd chest
link (2/3rds way up spine)
      | - "spine1"
      | - "spine"
WTB-WTF-- midpoint of 4 waist markers at
bottom of spine
```

## Overall CSM File Structure

The CSM file is broken down into the following sections, each introduced by a \$section title. These sections can occur in the any order, with the exception of \$order, which must precede \$points, at some location in the file. The \$section titles may be upper or lowercase. The [ ] notation shown below indicates a required numeric value or string of the specified type. The [ ] notation gets replaced by an actual value or string in the file. The [ ] information may occur on the same line as the \$section title.

```
$comments      [string]      OPTIONA
L
```

Comments. Continued until the next \$section title.

```
$filename      [string]      OPTI
ONAL
```

Name of file. String with normal Windows 95/NT naming do's and don'ts.

```
$capturedate    [yyyy/mm/dd]  OPTIONAL
```

Date of capture session for this file. Month and date can be one or two digits (i.e. with or without leading zeros.). Year is always four digits long. (i.e. 1998/01/31)

```
$capturetime     [hh:mm:ss]    OPTIONAL
```

Time of capture session for this file. Uses 24 format (i.e. 23:02:59).

```
$actor          [string]
OPTIONAL
```

The name of the actor used to capture the motion. Useful to know so that actor-to-data map files can be automatically located. Useful also from a data management point of view. This does mean that each file will contain the data for only one actor/creature. For multiple characters multiple files will be used.

```
$firstframe     [ integer
]                OPTIONAL
```

The first frame of point data present.

```
$lastframe      [ integer
]                OPTIONAL
```

The last frame of point data present.

```
$rate           [ integer
]                OPTIONAL
```

The sample rate of the point data contained in the \$point section below, computed in samples per second.

\$Talent <units>

head <HEAD\_LENGTH>

neck <NECK\_LENGTH>

clavicle <CLAV\_LENGTH>

upperarm <UPPERARM\_LENGTH>

forearm <LOWERARM\_LENGTH>

spine <TORSO\_LENGTH>

pelvheight <PELVIS\_HEIGHT>

pelvheight <PELVIS\_LENGTH>

thigh <UPPERLEG\_LENGTH>

calf <LOWERLEG\_LENGTH>

footheight <FOOT\_HEIGHT>

footlength <FOOT\_LENGTH>

where <units> is either millimeter OR inches  
(one MUST be specified)

and

HEAD\_LENGTH is measured from the top of the  
head to bottom of the chin

NECK\_LENGTH is measured from the base of the  
head to top of the spine

CLAV\_LENGTH is the length of the clavicle or  
collar bone

UPPERARM\_LENGTH is measured from the  
shoulder to the elbow

LOWERARM\_LENGTH is the length from the  
elbow to the wrist

TORSO\_LENGTH is the length of the entire  
spine, which will be broken into three links

PELVIS\_HEIGHT is used to specify the distance  
between the base of the spine and the pelvis

PELVIS\_LENGTH is measured from the right hip  
to the left hip

UPPERLEG\_LENGTH is measured from the hip  
to the knee

LOWERLEG\_LENGTH is measured from the  
knee to the ankle

FOOT\_HEIGHT is measured from the ankle to  
the bottom of the foot

FOOT\_LENGTH is measured from the heel to the  
base of the big toe

The names for body parts now correspond to  
biped naming conventions (that the user sees  
when clicking Biped parts).

All 12 body parts must be delineated.

Talent size information in a CSM file will be  
ignored if a Figure Structure file (\*.fig) is specified  
in the Talent Definition area of the Biped  
Motion Capture Conversion Parameters dialog.

The <body part> <number> pairs may be  
specified in any order.

If a body part is missing, it will receive a large  
default dimension so that the user can visually  
see that these missing part dimensions need to  
be provided.

If a body part name cannot be identified due to  
a misspelling, its numerical value is skipped over  
as if it was a missing body part

```
$spinelinks      [ integer ]      OPTIONAL
```

Indicates how many links are desired in  
converted biped (default is 3, range is 2-3).

```
$order           [name1 name2 name3  
.....nameN ]    REQUIRED
```

The order of the point/marker data is contained  
in the point section which follows. Each string  
name corresponds to a capture marker/point  
name. It is assumed that a value is provided for  
every point, not necessarily a measured marker,  
in every frame that is present. If there is any  
mismatch between the number of items  
contained in this \$order section and the number  
of items in the \$point section then it is assumed  
that the file has been incorrectly generated and  
the import will be aborted. Each marker is  
assumed to have 3 values X,Y and Z, specified in  
that order.

Note: Marker names can be of any length but  
must not contain commas (,).

Marker names should match Biped supported  
marker names by default. If custom marker

names are used, they must be correlated to \$points  
 Biped-supported marker names using the Marker  
 Name Mapping file (MNM), see below.

```
[ firstframe#_integername1_float_x      name1_float_y      name1_float_z
      name2_float_x      name2_float_y      name2_float_z
      name3_float_x      name3_float_y      name3_float_z
...   nameN_float_x      nameN_float_y      nameN_float_z ]
[ secondframe#_integer      name1_float_x      name1_float_y
      name1_float_z      name2_float_x      name2_float_y
      name2_float_z      name3_float_x      name3_float_y
      name3_float_z
      nameN_float_x      nameN_float_y      nameN_float_z ]
[ Nthframe#_integername1_float_x      name1_float_y      name1_float_z
      name2_float_x      name2_float_y      name2_float_z
      name3_float_x      name3_float_y      name3_float_z
...   nameN_float_x      nameN_float_y      nameN_float_z ]
]      REQUIRED.
```

A list of the X Y Z translations of the markers defined in the \$order section above, where the first item is the frame number automatically.

All data is specified in XYZ order.

All data is Z-up right-handed axis system.

The first frame number cannot be assumed to always be 1 (see firstframe= field).

The data section will be complete: it will contain a line for every frame between \$firstframe and \$lastframe, and frames will be ordered in ascending order.

Each frame is delimited with a carriage return or line feed (CR or LF).

The end of the \$point section is reached either when the end of file is found or a new section indicator (\$) is found.

Units are millimeters.

## Handling Missing “dropout” marker data

Where a missing or invalid coordinate value exists in the captured data, the *csn* file should contain an empty field designated by the symbol DROPOUT. The presence of the DROPOUT

symbol will not stop the import; the reader will simply skip to the next field. The DROPOUT symbol will replace the 3 XYZ values for the related marker at the current frame. That is, you do not need to use 3 DROPOUT symbols for X, Y, and Z.

Example:

\$Order

```
RFIN RWRA RWRB LWRA RTHI RFWT RELB
RKNE LWRB RTOE STRN LFWT LELB RANK
CLAV RBWT RHEE RFHD RSHO LFHD LUPA
LBWT LKNE LSHO C7 LTOE LANK LHEE LFIN
RBHD LBHD T10
```

\$Points

```
1  -615.870710 347.817731 951.389115  -
597.964337 393.216718 1065.157889  -
697.263315 439.700939 866.198188  -
560.523739 543.521728 1064.977016  -
704.859958 499.569721 726.202908  -
769.069680 537.733809 983.041794  -
827.491483 488.174757 1111.280365  -
786.614308 501.559318 490.345226  -
559.619377 667.057615 884.285434  -
619.849904 416.911009 81.392605  -
628.712654 649.874732 1246.934706  -
565.769040 753.152904 953.378712  -
472.438853 777.570685 1128.282376  -
```

```

756.046863 511.326431 137.824811 -
673.930768 684.421371 1359.618246 -
917.385093 708.115662 955.730053 -
740.491832 603.390511 50.282543 -
766.175720 617.136817 1644.673235 -
877.231408 652.045201 1457.289372 -
643.363323 677.729090 1664.569206 -
536.467702 871.443489 1305.718254 -
770.878404 896.584761 960.613610 -
721.861969 886.636776 478.769389 -
644.810303 875.241811 1451.501453 -
815.373028 832.194167 1505.763190 -
717.521030 822.607926 82.477840 -
829.481080 922.449522 142.889240 -
930.950527 892.967312
58.602675 DROPOUT DROPOUT DROPOUT DRO
POUT

```

### Additional Syntax Rules

- Any unrecognized \$sections are ignored.
- All blank lines are ignored.
- Leading white space on any line is ignored.
- Spaces or line feeds are used as separators.
- The reader allows any combination of upper and lower case letters.

### Character Studio Marker Name File (MNM)

The Character Studio Marker Name file is used to match custom marker names in the CSM file with Biped's preset list of known, supported marker names. For a complete list of supported marker names, see above.

The general syntax of the MNM file consists of two columns of names.

The left column corresponds to Biped's known marker names.

The column on the right corresponds to the custom marker names stored in a specific CSM file.

There should be an entry for every custom CSM marker name in the CSM file. For completeness,

all CSM markers can be listed and correlated to internal Biped names, even if the names are identical.

The listing below is an "identity" listing of fixed versus custom marker names files. The left and right columns are identical.

Optional (missing) marker name entries not in the CSM file can be omitted from the MNM marker file.

This list can be easily copied and modified with the right column for each unique motion capture session sample.

```

LFHD LFHD
LBHD LBHD
RBHD RBHD
RFHD RFHD
C7 C7
T10 T10
SACR SACR
CLAV CLAV
STRN STRN
LSHO LSHO
LELB LELB
LILB LILB
LWRE LWRE
LWRI LWRI
LFIN LFIN
RSHO RSHO
RELB RELB
RILB RILB
RWRE RWRE
RWRI RWRI
RFIN RFIN
LFWT LFWT
LBWT LBWT
RBWT RBWT
RFTW RFTW
LKNE LKNE
LKNI LKNI
LANK LANK
LHEL LHEL
LMT5 LMT5
LMTI LMTI
LTOE LTOE
RKNE RKNE
RKNI RKNI

```

RANK RANK  
 RHEL RHEL  
 RMT5 RMT5  
 RMTI RMTI  
 RTOE RTOE

## Sample CSM and MNM

This section provides examples of a corresponding pair *.csm* and *.mnm* files: *fashion1.csm* and *fashion.mnm*.

It also provides a brief working sample, *wendy.csm*, which illustrates how to specify the talent size syntax.

```
-----
fashion1.csm
-----
$filenamefashion1.csm
$capturedate1998/01/31
$capturetime23:31:59
$actorBogy
$comments
This is a Character Studio CSM File
$firstframe 1
$lastframe 3
$spinelinks 3
$rate 60
$order
C7 CLAV LANK LBHD LBWT LELB LFHD LFIN LFWT LKNE LSHO LTOE LUPA LWRA LWRB RANK RBHD RBWT
RELB RFHD RFIN RFWT RKNE RSHO RTHI RTOE RWRA RWRB STRN T10
$points
1 980.6 -2365.8 1541.3 1030.6 -2239.9 1492.1 967.3 -2427.5 181.8 936.2 -2330.1 1672.9
939.4 -2359.1 1109.3 782.9 -2263.3 1175.7 959.0 -2196.9 1753.8 762.3 -2135.1 862.6
949.3 -2187.2 1082.1 934.2 -2343.2 583.1 870.4 -2246.7 1525.4 1031.7 -2339.7 83.1 814.6
-2218.7 1318.8 799.4 -2132.3 993.7 729.2 -2230.2 966.5 1110.1 -2060.1 197.3 1066.8 -
2351.6 1670.5 1114.0 -2391.4 1116.4 1290.5 -2574.6 1396.5 1105.7 -2231.2 1740.1 1217.5
-2369.8 1149.5 1159.8 -2233.6 1093.8 1138.8 -2119.2 605.9 1136.6 -2342.8 1564.9 1081.3
-2126.7 775.0 1065.2 -1944.5 112.2 1276.8 -2398.1 1271.2 1294.0 -2490.1 1183.6 1044.2 -
2199.9 1395.9 988.1 -2404.3 1417.4
2 979.8 -2358.9 1540.1 1029.4 -2232.4 1489.5 966.5 -2426.7 181.8 936.2 -2322.0 1671.1
940.7 -2351.6 1107.3 783.9 -2249.8 1173.7 959.4 -2189.0 1752.6 767.9 -2113.2 863.4
948.9 -2177.9 1080.7 934.4 -2338.9 583.3 871.7 -2242.9 1523.4 1031.7 -2339.5 82.9 814.6
-2207.6 1317.8 803.3 -2113.8 994.5 732.4 -2209.8 964.1 1108.3 -2039.1 197.5 1067.2 -
2343.2 1669.3 1115.0 -2382.1 1115.0 1292.6 -2563.1 1400.3 1106.5 -2223.7 1738.5 1212.0
-2367.2 1155.5 1162.2 -2222.1 1093.8 1140.2 -2119.4 601.9 1135.4 -2334.3 1563.9 1079.9
-2109.1 774.4 1064.5 -1921.3 116.0 1278.4 -2396.2 1273.8 1284.5 -2483.4 1186.2 1043.6 -
2192.5 1394.0 987.5 -2397.7 1415.2
3 979.6 -2351.7 1539.3 1029.2 -2224.5 1487.1 966.1 -2426.1 181.6 936.4 -2313.5 1669.3
942.7 -2343.0 1105.1 784.3 -2236.2 1171.7 959.4 -2180.8 1750.6 773.8 -2090.6 864.4
949.3 -2168.6 1078.7 934.2 -2334.3 582.5 871.3 -2235.2 1521.8 1031.3 -2339.1 82.5 814.4
-2196.5 1315.4 808.1 -2095.0 995.7 736.1 -2189.2 962.5 1107.1 -2019.5 197.9 1067.8 -
2334.5 1668.0 1116.2 -2372.6 1113.2 1294.4 -2555.6 1402.1 1105.7 -2214.4 1736.9 1206.2
-2358.5 1162.0 1163.4 -2214.2 1091.6 1140.2 -2115.6 598.3 1134.8 -2325.0 1562.3 1080.3
```

```

-2088.1 776.8 1058.9 -1897.7 119.9 1279.2 -2387.4 1276.6 1275.4 -2474.9 1189.0 1042.8 -
2185.0 1391.8 987.7 -2391.0 1415.2
-----
fashion.nmn
-----
LFHD LFHD
LBHD LBHD
RBHD RBHD
RFHD RFHD
C7 C7
T10 T10
SACR SACR
CLAV CLAV
STRN STRN
LSHO LSHO
LELB LELB
LILB LILB
LWRE LWRA
LWRI LWRB
LFIN LFIN
RSHO RSHO
RELB RELB
RILB RILB
RWRE RWRA
RWRI RWRB
RFIN RFIN
LFWT LFWT
LBWT LBWT
RBWT RBWT
RFTW RFTW
LKNE LKNE
LKNI LKNI
LANK LANK
LHEL LHEL
LMT5 LMT5
LMTI LMTI
LTOE LTOE
RKNE RKNE
RKNI RKNI
RANK RANK
RHEL RHEL
RMT5 RMT5
RMTI RMTI
RTOE RTOE
-----
wendy.csm
-----
$Filename wendy.wave.csm
$Date 1998/06/11
$Time 11:49:30

```



```

$Actor wendy
$Comments
Motion capture data from Vicon
Talent measurements are:
HEAD_LENGTH
NECK_LENGTH
CLAV_LENGTH
UPPERARM_LENGTH
LOWERARM_LENGTH
TORSO_LENGTH
PELVIS_HEIGHT
PELVIS_LENGTH
UPPERLEG_LENGTH
LOWERLEG_LENGTH
FOOT_HEIGHT
FOOT_LENGTH
$FirstFrame 12
$LastFrame 300
$Rate 25.000
$Talent inches head 6.9 neck 2.6 clavicle 4.0 upperarm 10.7 forearm 10.7 spine 15.8
pelvheight 4.0 pelvlength 8.0 thigh 18.0 calf 16.5 footheight 5.0 footlength 7.0
$Order
LFHD RFHD LBHD RBHD C7 T10 CLAV STRN LSHO RSHO LELB RELB LWRI RWRI LWRE RWRE LFIN RFIN
LBWT RBWT LFWT RFWT LKNE RKNE LANK RANK LTOE RTOE
$Points
12 -1085.844924 34.741794 1419.824995 -1083.878407 -104.061504 1430.149208 -
1230.220018 34.577917 1352.144048 -1217.929290 -110.780436 1357.224216 -1273.155631
-49.490668 1238.741590 -1353.455060 -52.768196 1114.359415 -1121.897728 -46.213140
1155.984016 -1124.355874 -51.457185 1048.972737 -1184.826260 118.646502
1217.929290 -1177.779575 -202.387335 1229.072884 -1445.061958 188.293966
1038.648525 -1211.538111 -313.823276 915.085731 -1504.549086 192.882505
814.465631 -1027.341054 -303.499064 794.472712 -1454.075160 231.885084 815.612766 -
992.599261 -327.425016 844.946638 -1515.364927 169.448182 714.992666 -926.884831 -
271.543169 793.489454 -1439.981791 72.105609 847.404784 -1462.924484 -133.559253
853.304334 -1274.794395 69.647463 786.934398 -1301.178493 -170.595316 793.653330 -
1072.079307 97.506449 481.796570 -1454.402912 -175.183855 341.354509 -1278.563552
46.049264 147.324870 -1801.329218 -116.188357 136.345152 -1137.302108 8.029943
66.533812 -1686.779626 -114.057964 34.086288
13 -1014.067067 35.069546 1436.868139 -1011.608921 -102.422740 1444.078700 -
1155.328511 36.216681 1368.859440 -1145.823680 -107.994537 1371.809215 -1198.919629
-52.768196 1255.456982 -1275.941530 -61.453644 1129.763795 -1049.300490 -40.149714
1174.502048 -1047.170097 -45.557635 1067.654645 -1115.834302 118.318750
1238.905467 -1092.399979 -195.668403 1253.654341 -1356.896464 174.200597
1085.025542 -1149.592837 -335.127206 952.777299 -1439.490161 188.949471
860.187142 -914.921855 -279.245359 833.639168 -1400.487582 238.604016 862.317535 -
905.744777 -316.609175 876.247028 -1447.356228 176.494866 755.634009 -844.127257 -
260.891204 823.314956 -1356.240958 68.500329 872.641747 -1373.447978 -137.492287
878.377421 -1196.461483 64.239543 805.780183 -1223.664963 -178.625259 802.338778 -
999.973698 98.981336 493.595670 -1435.557128 -173.708968 365.116585 -1139.596378

```

```

35.233423 128.151333 -1777.403266 -122.087906 150.930150 -988.993980 2.785899
72.433362 -1686.124120 -114.057964 33.758535
14 -939.994941 36.544434 1454.566789 -938.028425 -100.128471 1458.335946 -
1078.306610 38.838703 1386.066460 -1073.226442 -104.225381 1388.852359 -1122.880987
-53.423701 1273.483384 -1195.969854 -69.155834 1144.512669 -972.934095 -32.119771
1194.658843 -965.559657 -36.708310 1090.269586 -1046.022962 118.154873
1257.915127 -1008.495270 -184.360933 1280.366192 -1299.539729 165.187396
1114.359415 -1126.486267 -369.869000 1006.201001 -1361.321126 192.554752
882.965960 -840.358100 -265.643619 847.404784 -1325.104445 242.864802 894.928936 -
833.475292 -308.087603 892.470790 -1360.501744 192.554752 781.854230 -780.379343 -
249.583734 829.870011 -1270.697485 60.962015 887.882251 -1282.004956 -145.194477
893.454048 -1112.065145 57.028982 822.495574 -1136.646603 -185.016438 813.646249 -
931.309493 99.636842 495.726063 -1398.848818 -171.742451 395.761469 -1035.862626
25.400840 114.713469 -1732.828890 -131.756613 168.464923 -878.705174 2.785899
73.088867 -1676.291537 -114.057964 29.169996
15 -863.300794 38.674827 1468.496281 -863.628546 -97.834202 1469.151787 -998.498811
42.116231 1399.504324 -1000.301451 -98.817460 1401.306964 -1046.514591 -52.112690
1285.610236 -1115.834302 -73.908249 1152.378736 -893.454048 -22.942694
1210.718729 -880.671690 -28.350615 1108.623741 -977.522634 119.957513 1266.764452 -
926.065449 -173.872844 1297.900965 -1240.871983 158.468464 1115.834302 -1105.346213
-398.219614 1047.661726 -1269.058722 199.273684 881.654949 -794.636588 -263.840979
844.946638 -1236.283445 247.125588 902.958879 -776.446310 -305.793333 891.487532 -
1257.915127 213.039300 787.753780 -731.708057 -243.356431 816.596024 -1182.859743
52.440443 899.517475 -1188.267664 -154.043801 904.925395 -1027.013302 52.440443
833.803044 -1045.531333 -188.621719 823.806585 -869.200343 98.325831 486.221233 -
1332.315006 -175.019979 416.737646 -974.408982 15.896009 104.225381 -1663.181426 -
144.211218 187.966213 -813.482373 1.147135 62.600779 -1648.760304 -107.502908
26.711851
16 -786.115016 42.116231 1475.215213 -788.737039 -93.901168 1473.740326 -918.199382
47.032522 1405.567750 -923.771179 -90.951393 1403.437357 -964.740276 -48.343533
1286.921247 -1036.518132 -75.547013 1153.361994 -812.499114 -15.732133
1215.798897 -796.275352 -22.451065 1114.359415 -909.350058 123.890547 1263.159172 -
844.782762 -169.939811 1300.195235 -1162.375195 155.518689 1086.008800 -1018.491730
-387.567649 1046.678468 -1156.803398 204.845481 855.106974 -764.974963 -273.673562
827.739618 -1131.238682 252.533509 882.474331 -732.691315 -312.512265 871.002984 -
1134.352333 229.754691 770.055131 -698.768903 -251.714127 786.770522 -1099.446664
44.574377 905.253148 -1099.118911 -162.565373 909.677810 -943.927975 51.129432
837.408325 -955.563198 -189.604977 828.395124 -818.398664 92.754034 465.736685 -
1229.892266 -182.066663 411.493601 -951.794041 8.849325 95.703809 -1569.116382 -
158.468464 211.236660 -792.342319 -8.029943 47.032522 -1586.651155 -99.636842
40.641343
17 -708.437610 46.540893 1474.887460 -713.190025 -89.312630 1471.609933 -837.080572
53.259825 1402.126346 -845.438268 -82.593698 1396.390672 -883.621465 -46.213140
1279.219057 -957.365838 -74.727631 1150.084466 -730.724798 -11.799100 1210.227100 -
713.517778 -19.829043 1107.148854 -837.572201 127.331951 1250.049061 -763.827828 -
171.250822 1289.543270 -1066.343634 160.434980 1035.043244 -897.550958 -354.300743
1021.769257 -1017.672348 206.156492 815.121136 -737.607607 -289.733448 800.536138 -
999.645945 255.974913 845.438268 -692.869354 -326.441758 837.572201 -979.325274
233.032219 739.246370 -678.284355 -278.262101 751.373223 -1022.752516 38.838703
905.253148 -1017.836224 -170.267563 904.269890 -865.922816 50.146174 833.475292 -

```

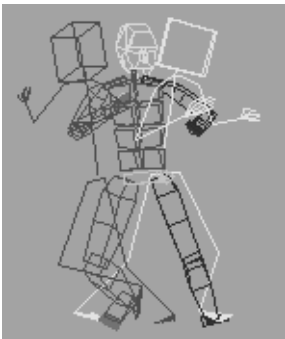
```

869.200343 -189.113348 825.117596 -773.824288 80.791058 439.516463 -1097.971776 -
188.457842 386.912144 -948.188761 3.769157 91.115270 -1455.550047 -172.397956
240.570532 -788.900915 -17.043144 39.494209 -1482.589651 -111.927571 68.336452
18 -630.596328 50.146174 1470.135045 -636.332001 -85.871225 1465.710383 -756.125638
58.012240 1392.457639 -765.630468 -77.513530 1386.721966 -804.305295 -44.410500
1268.894845 -877.558039 -71.941733 1145.168175 -648.786606 -12.454605 1199.575135 -
632.890597 -22.123312 1094.530372 -759.567042 129.462344 1234.480804 -685.167164 -
174.036720 1277.088664 -953.596681 170.595316 979.653027 -806.763441 -332.013555
1007.184259 -848.551919 200.093065 782.181983 -707.782105 -310.873501 779.068332 -
835.933437 251.714127 813.974002 -656.324920 -345.451419 805.452430 -790.703555
219.758232 719.745081 -666.485256 -315.953669 727.283394 -951.466288 33.922412
902.631126 -942.125334 -178.133630 892.143037 -799.061251 50.801679 824.953720 -
788.409286 -188.621719 817.251529 -728.922158 68.008700 419.523544 -947.041626 -
196.323909 362.986192 -945.402862 1.474887 92.754034 -1316.746749 -184.360933
265.151990 -787.589904 -19.501290 38.019321 -1338.542309 -143.719589 89.640382

```

## Motion Flow Mode

Select a biped > Motion panel > General rollout > Motion Flow Mode



The Transition Editor uses **ghosts** to represent the source (yellow) and destination (red) clips. Use these stick-figure ghosts to judge body position and set a likely start frame in both clips for the transition.

In Motion Flow mode, you combine *.bip* files, using either velocity-interpolated transitions or optimized transitions that compute minimum foot sliding, to create longer character animation. First, clips are added and referenced to *.bip* files in the Motion Flow Graph, these are then selected to create a Script on the Motion

Flow Script rollout. The Transition Editor is used to adjust transitions between *.bip* files. Scripts, transitions, and clip references are saved in a Motion Flow Editor file (*.mfe*) for later editing or by using Save Segment on the General rollout to create one long *.bip* file based on the script.

Save Segment in Motion Flow mode does *not* save footsteps in the *.bip* file. Instead, feet are assigned an IK Blend value of 1 with the Object parameter on for the keys representing a footstep period (this puts the feet in world space). To extract footsteps from these *.bip* files, exit Motion Flow mode, load the *.bip* file and use Convert on the General rollout. Convert bases footstep extraction on foot IK Blend values.

## Random Motion and Crowds

Random motions can be created for one or more bipeds using the Create Random Motion command. You can animate a crowd of bipeds for example. For a crowd you must share one motion flow among many bipeds.

If you are driving a crowd using delegates and behaviors, then rather than a completely random motion, the software picks appropriate clips based on the delegates speed and direction. If the delegate slows to a stop the software will find and use a clip that slows to a stop if one exists.

In all crowd simulations you must load clips and create transitions before synthesizing the crowd motion. Often many clips are used to synthesize crowds. Automatic transitions relieve you of having to create transitions between clips manually.

Two additional rollouts display in Motion Flow mode:

- **Motion Flow rollout** (see page 229). Display the Motion Flow Graph; load and save motion flow files (*.mfe*) on the Motion Flow rollout.
- **Motion Flow Script rollout** (see page 234). Create scripts, edit transitions, create a unified motion and create random motions for the biped(s) using commands in the Motion Flow Script rollout.

Note: In Motion Flow mode some **character Studio** controls are disabled.

## Workflow: Getting Started with Clips and Transitions in Motion Flow mode

Create your own library of *.bip* files from imported motion capture data and from your own character animation. With a biped selected, turn on Motion Flow mode. Select Show Graph on the Motion Flow rollout to open the Motion Flow Graph. On the Motion Flow Graph, click Create Clip, and add clips in the Motion Flow dialog. Associate the clips to *.bip* files by first turning on Select Clips-Transitions on the Motion Flow Graph toolbar and then right-clicking a clip icon; a clip dialog appears, allowing you to browse for a clip.

Click Define Script on the Motion Flow Script rollout, then select a series of clips on the Motion Flow Graph. By left clicking on a clip it will appear in the list on the Motion Flow Script rollout. The clips are connected visually in the Motion Flow Graph with red arrows (active

script) representing default transitions (Minimum Motion Loss). The default duration for a transition is 25 frames, which provides good results in many cases. Transitions by default use velocity interpolation between clips.

Use Edit Transition on the Motion Flow Script rollout to fine-tune the transitions between clips. Select a clip in the list of the Motion Flow Script rollout, then on the Motion Flow Script rollout, click Edit Transition; the *Transition Editor* (see page 237) displays, and the source and destination clip names are displayed at the top of the Transition Editor.

In the Source Clip area, Start Frame represents the frame in the source file where the transition starts. For example, if Source Clip Start Frame is 60, the transition *from* the source clip starts at frame 60. The *duration* of the transition is set in the Length field at the upper left; Length values are in frames. In this example, if Length is set to 10, the transition to the destination clip takes 10 frames. In the Destination Clip area, Start Frame represents the frame in the *destination* clip that the transition starts; a value of 80, for example, starts the transition at frame 80 of the *destination* clip. In this example, the source clip plays from 0 to 60, there is a 10 frame transition from frame 60 of the source clip to frame 90 of the destination clip (frames 80 to 90 cover the destination clip transition period), then the rest of the destination clip plays.

When the Transition Editor displays, the first things to try, before manually editing the transition, are the Optimized Transitions. Optimized transitions use a minimum foot sliding algorithm. The top right hand corner of the Transition Editor dialog is the icon for optimized transitions. If the optimized transition is not satisfactory try editing the transition manually.

Manual transition editing offers the most control; the Frame spinners in the Ghost areas of the Transition Editor allow you to scrub the source and destination clips while viewing two stick figures. Find a good start frame in both clips using the Frame spinners. Things to look for in both clips are similar supporting feet, body momentum that will appear natural, and arm motion similarities. If velocity changes between the clips are too abrupt, use the Length field to adjust the duration of the transition.

On the Motion Flow rollout, click Save to save your work as a *.mfe* file; transitions and scripts are saved. Use Save Segment on the General rollout to save a *.bip* file based on the current script. These *.bip* files contain no footsteps. Biped foot keys are saved with an IK Blend value of 1 for footsteps. To extract footsteps, exit Motion Flow mode, use Load File on the General rollout, and then click Convert on the General rollout. Convert looks at foot IK Blend values of 1 to extract footsteps.

**Tip:** The location of the referenced *.bip* files is saved in the *.mfe* file. If the *.bip* file cannot be found, the program looks to the motion flow directory specified in `\plugcfg\biped.ini`. By default, this setting is

`MoFlowDir=<maxdir>\cstudio\scripts`

If a referenced *.bip* file cannot be found in its current location, you will need to move it to the specified Motion Flow directory. You can change the location of this directory at any time by editing your *biped.ini* file with a text editor. The new directory will be used the next time you restart 3D Studio MAX. You can also add multiple search paths to your *biped.ini* file by repeating the `MoFloDir=` line multiple times. The program will search the directories in the order they appear and will use the first instance of the file that it finds. When network rendering the filenames need to be UNC compatible.

## Motion Flow Rollout

Select a biped > Motion panel > General rollout > Motion Flow Mode > Motion Flow rollout

The Motion Flow rollout displays when Motion Flow mode is turned on in the General rollout. Load and save motion flow editor files (*.mfe*), display the *Motion Flow Graph* (see page 230) and display the Shared Motion Flow dialog using controls on this rollout.

### Procedure

To load a *.mfe* file

1. Select a biped.



2. Open the Motion panel.



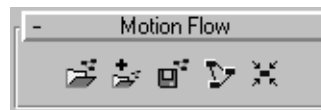
3. Click Motion Flow mode on the General rollout.



4. Click Load File on the Motion Flow rollout.

5. Select a *.mfe* file.

### Interface



**Load File.** Load a Motion Flow Editor file (*.mfe*). Motion Flow Editor files include:

- **Clips.** references to biped animation files.
- **Transitions.** The names, attributes, and connections between clips.
- **Scripts.** different paths through a set of connected clips and transitions.

A number of sample Motion Flow Editor files and new biped animations have been included with **character studio**. These files are located in the `cstudio\scripts` directory.

**Tip:** The location of the referenced *.bip* files is saved in the *.mfe* file. If the *.bip* file cannot be found, the program looks to the motion flow directory specified in `plgcfg\biped.ini`. By default, this setting is:

MoFlowDir=<maxdir>\cstudio\scripts

If a referenced *.bip* file cannot be found in its current location, you will need to move it to the specified Motion Flow directory. You can change the location of this directory at any time by editing your *biped.ini* file with a text editor. The new directory will be used the next time you restart 3D Studio MAX. You can also add multiple search paths to your *biped.ini* file by repeating the `MoFloDir=` line multiple times. The program will search the directories in the order they appear and will use the first instance of the file that it finds. When network rendering the filenames need to be UNC compatible.

If you load a *.mfe* file onto a biped using a shared motion flow, you will get a warning and the biped will be removed from the shared motion flow. The biped will get the newly loaded motion flow and all its scripts.



**Append File.** Append a *.mfe* file to the *.mfe* that is already loaded. Displays a load file dialog.

The appended graph will appear directly below the bottom of the existing graph in the graph window, so you may have to scroll down to see it.



**Save File.** Save a Motion Flow Editor file (*.mfe*).

Saving a *.mfe* file from a biped having a shared motion flow will save the motion flow and its scripts as if it were not shared. It will be a normal *.mfe* file.

Note: To save a script as a *.bip* file use Save Segment on the General rollout. You can also click Unified motion to have the scripted motion available when you exit Motion Flow mode.



**Show Graph.** Opens the *Motion Flow Graph* (see page 230). The first step in script creation is to add clips to the Motion Flow Graph.



**Shared Motion Flow.** Displays the *Shared Motion Flows dialog* (see page 243). Allows you to create, delete, and modify shared motion flows. Shared motion flows are used to animate one or more bipeds to simulate a crowd. If the current biped is using a shared motion flow then the icon has a white circle around it.

---

## Motion Flow Graph Dialog

Select a biped > Motion panel > General rollout > Motion Flow Mode > Motion Flow rollout > Motion Flow Graph > Motion Flow Graph dialog

Use tools in the Motion Flow Graph dialog to add clips to the dialog window, calculate optimized transitions, set random script transition values, move and delete clips and display clip dependencies. Clips and transitions display as icons in the Motion Flow Graph dialog. The Motion Flow Graph displays when you click Show Graph on the *Motion Flow rollout* (see page 229) on the Motion panel.

The first step in Motion Flow mode is to add clips in the Motion Flow Graph for use in scripts.

Clips represent all or part of a *.bip* file. Scripts represent different paths through the clips in the Motion Flow Graph. The first clip in the current script is red. Transitions are shown as arrows between clips, red arrows represent the path through the active script. Black transition arrows indicate unloaded scripts. A transition looping back to the same clip represents a cycle or loop. If the biped is using a shared motion flow then the title of the graph dialog will say “Shared Motion Flow Graph”, followed by the name of the shared motion flow. Shared Motion Flows are used to control multiple bipeds with one shared motion flow.

By default minimum motion loss is used to compute transitions. Optimized transitions use an algorithm that uses minimum foot sliding. Optimized transitions take longer to compute but yield very high quality results.

## Random Scripts for One or More Bipeds

You can create random scripts by using the Create Random Motion command in the Motion Flow Script rollout. Random scripts are created by randomly traversing clips in a motion flow graph. To use Create Random Motion each biped must be in the same shared motion flow.

To use a shared motion flow to create random scripts for multiple bipeds, the first step is to click Shared Motion Flow on the Motion Flow rollout, and add bipeds that will share one shared motion flow. On the Motion Flow Graph, clips are added and transitions are created between all the clips. Then Create Random Motion is used to compute a random motion for all the bipeds.

Clips and transitions are given percentages that are used by the software to generate random scripts. Percentages for clips and transitions are

set in the clip dialogs and in the transition editor dialog.



You can load many clips and use Synthesize Motion Flow Graph to create all possible transitions between the clips. You can then use Optimize Selected Transitions to create optimized transitions. Once the transitions are created you can quickly create scripts with optimized transitions or generate random motions for a crowd of bipeds.

A *.mfe* file stores pointers to the clips, transition parameters, and scripts.


See the *Motion Flow Script rollout* (see page 234) to create scripts and transitions.

## Procedures





### To create clips in the Motion Flow Graph

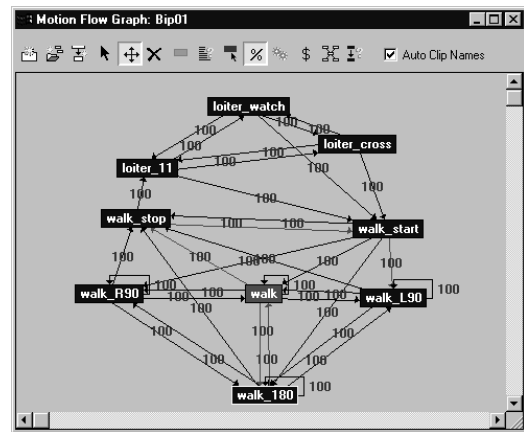
1.  Click Create Clip on the Motion Flow Graph toolbar.
2. Click a few times inside the Motion Flow Graph window.
3.  Click Select Clip/Transition on the Motion Flow Graph toolbar.
4. Right-click over a clip in the window. A Load File dialog displays.
5. Click Browse in the load dialog and choose a *.bip* file.
6. Optionally, set the start and end frame and then click OK.

The selected icon displays the clip name in the Motion Flow Graph; this clip can now be used for script creation.

7.  To create transitions between clips turn on Create Transition on the Motion Flow Graph toolbar and then click and drag from one clip to another clip.

**To load clips and create optimized transitions**

1.  On the General rollout turn on Motion Flow Mode.
2.  On the Motion Flow rollout click Show Graph.  
The Motion Flow Graph dialog displays.
3.  On the Motion Flow Graph toolbar click Create Multiple Clips.
4. Select multiple clips in the load file dialog.  
Clips appear on Motion Flow Graph window.
5.  On the Motion Flow Graph toolbar click Synthesize Motion Flow Graph.  
Every possible transition is added.
6. Select all the transitions in the Motion Flow Graph.
7. Click Optimize Selected Transitions.  
Optimized transitions yield high quality transitions but they take time to compute. You can now create scripts or crowd simulations that use these transitions.

**Interface**

**Create Clip.** Select and click in the dialog window to create clips.

The clips are empty, right-click a clip using the Select Clip tool to display the *Clip Properties dialog* (see page 246); select a *.bip* file and set its duration in the Clip Properties dialog. You can also set the Random Start Probability here. Random Start Probability is used when multiple clips are selected as possible start clips when you generate a random motion flow.

Note: Setting clip duration is not critical for transitions; the *Transition Editor* (see page 237) allows you to start and end a transition on any frame of a clip.



**Create Multiple Clips.** Load multiple motion files.

Displays an open file dialog. Select multiple files and click OK, multiple files are loaded into the Motion Flow Graph window.

**Set lowest starting foot height to Z=0 (.bip files only).** Sets the lowest starting foot height to Z=0. This is an option in the Load File dialog. Default=On.



In **character studio 3** the height of a motion clip can be retained. This is important if you want to retain the height of a motion clip for motions adapted to characters of different sizes. If, for example, the character is jumping a rock and you want to retain the Z position of the character, you would turn this option off. Leave this option off if Motion Flow motions must be blended that begin and end at different heights, such as three clips that have the character mounting a bicycle, riding the bicycle, and dismounting the bicycle.

Turning off this option can, however, cause a jump in the motion during motion flow transitions. Turn this on for smooth transitions in Motion Flow mode. If adaptation takes place, the height is set so that the lowest foot at frame 0 starts at the Z=0 height. This lines up clips along the Z axis and creates smooth transitions.



**Create Transition.** Create a transition between two clips.

Click+drag from one clip to another in the Motion Flow Graph window. Click and then mouse up to create a loop transition. It is necessary to have this capability in order to create random scripts. You can create transitions which are not included in a script.



**Move Clip.** Moves clips within the dialog window, this does not affect the animation.



**Select Clip/Transition.** Selects a motion clip or transition.

Right-click a clip to display the *Clip Properties dialog* (see page 246). Right-click a transition to display the *Transition Editor* (see page 237).



**Delete Clip/Transition.** Deletes a clip or transition.

If a script is dependent on the clip, a dialog displays a warning; clicking OK on the dialog deletes the clip and the script that is dependent on it.

If you delete the selected clips and transitions from a shared motion flow, it will delete all the scripts from all the bipeds sharing that motion flow which are dependent on those clips and transitions.



**Clip Mode.** Edit biped footsteps and limbs for the selected clip. Use Set Key on the Keyframing rollout to set biped limb keys. Clips turn green in the Motion Flow Graph window in Clip Mode.



**Show Script Dependencies.** Displays the scripts that use the selected clip.

If you push the Show Script Dependencies button on a shared motion flow graph, it will check all the bipeds sharing that motion flow for scripts dependent on the selected clips and transitions.



**Select Random Start Clips.** Turn on and select clips in the Motion Flow Graph window. Press CTRL+click to add clips. Selected clips are used by Create Random Motion in the Motion Flow Script rollout to start on one of the selected clips based on percentage. If three clips are selected using the default weighting of 100 then each clip has an equal chance of being the start clip.



**Show Random Percentages.** Displays clip and transition percentages in the Motion Flow Graph window.

Random start clips display in purple and display their probability of starting a random script. This also shows the probability (0 - 100) that

each transition will be chosen. Create Random Motion in the Motion Flow Script rollout uses clip and transition percentages to generate random scripts.



**Optimize Selected Transitions.** Select one or more transitions and then click Optimize Selected Transition to optimize them. Displays the *Transition Optimization dialog* (see page 245) to set the location of the transition.

Optimized transitions take time to compute. A progress bar is displayed when you use this feature. Minimum foot sliding is the method used to compute an optimized transition.



**Show Optimal Transition Costs.** Displays costs in the Motion Flow Graph window. The lower the number the better the transition.



**Synthesize Motion Flow Graph.** Creates transitions between every clip. The transitions are not optimized. Use Optimize Selected Transitions to optimize the transitions. Optimized transitions take time to compute but are high quality.



**Check All Transitions.** Checks the graph for overlapping transitions and transitions whose length is too long for the clip. It informs you of any problems, or tells you that none have been found.



**Auto Clip Names.** Names the clip based on the name of the motion file. Turn off to name a clip yourself.

## Motion Flow Script Rollout

Select a biped > Motion panel > General rollout > Motion Flow Mode > Motion Flow Script rollout

Create and delete scripts, name scripts, edit transitions, edit clips and position the entire animation using tools on the Motion Flow Script rollout. Create random motion for one or more bipeds using controls in this rollout. The Motion Flow Script rollout on the Motion panel is only available when *Motion Flow mode* (see page 227) is active.

### Scripts

A Script is a list of clips (.bip files) that you constructed and are executed as you designed to animate a character. To create a script, add clips to the Motion Flow Graph, then select Define Script on the Motion Flow Script rollout and click a sequence of clips in the Motion Flow Graph window. Default transitions are assigned if no transitions exist between the clips. The clip names and starting frame numbers display in the list on the Motion Flow Script rollout.

Different scripts vary the order of the clips found in the Motion Flow Graph. Scripts are run in a top-down order to animate the character. Click Play to view script motion on the biped.

**Tip:** As the length of a script varies, use ALT+R to set the Active Time in 3DS MAX to the length of the script. Turn on the Plug-in Keyboard Shortcut Toggle near the 3DS MAX status line to enable **character studio** shortcuts.

Note: Use Save Segment on the General rollout to output a script as one long .bip file. You can also create a unified motion that will make the script motion available when you exit motion flow mode.

## Transitions Between Clips

By default, when a script is created, Minimum Motion Loss is used to find likely start frames for the source and destination clips.

Editing transitions using the *Transition Editor* (see page 237) allows you to determine where a transition occurs in the source and destination clip. Transition duration and the orientation of the destination clip can also be adjusted in the Transition Editor.

## Random Motion



The Create Random Motion command traverses clips in the Motion Flow Graph based on transition percentages. Transitions are given percentages and the software creates random scripts based on the transition percentages. This is a quick way to create crowd scenes or to try out different scripts on a single biped. For example, if five clips are present in the Motion Flow Graph and transitions exist between all the clips and each clip has a percentage or probability of being used, you can use Create Random Motion to create a script that is comprised of the five clips that are selected at random.

## Position the Entire Animation

Use the Position, Rotation, and Start Frame controls to position the entire animation. If you are editing the script for a character in a scene with other objects or characters, use these controls to position the animation relative to the rest of the scene.

## Procedures

### To create a script

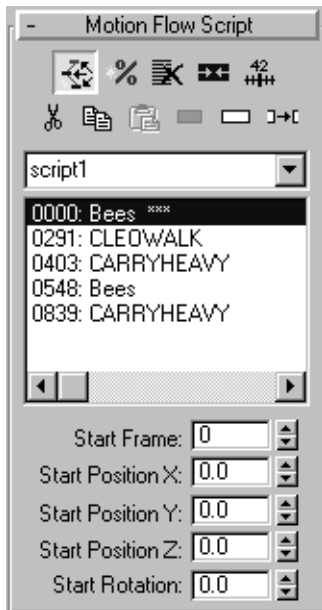
1. *Add clips* (see page 231) to the Motion Flow Graph window and associate these clips to stored *.bip* files.
2.  On the Motion Flow Script rollout, click Define Script, and then select a sequence of clips in the Motion Flow Graph. The selected clips appear in the list on the Motion Flow Script rollout.
3.  Edit Transitions by selecting a clip in the list of the Motion Flow Script rollout and click Edit Transition. Change parameters in the *Transition Editor* (see page 237).  
Click Play to view the script
4. Save a *.mfe* file to work on later.
5. Use Save Segment on the General rollout to output the script as a *.bip* file.  
Click Create Unified Motion to make the motion available when you exit Motion Flow mode.


### To move footsteps to line up clips

If you find that one clips z position is higher than another clips z position and there is a jump during the transition, you can simply change one clips z position by moving all of its footsteps along the z-axis.

- Go into footstep mode, select all of the footsteps and drag them along the z-axis to match them up with the other clip.

## Interface




 **Define Script.** Display the Motion Flow Script dialog (no dialog displays if there are no scripts, in this case simply select clips in the Motion Flow Graph).

The Motion Flow Script dialog has the following options:


- **Create New Script.** Names a new script. Select clips in Motion Flow Graph to create the clip list for the new script.
- **Redefine Script.** Keeps the script name and removes clips in the list. Select clips in Motion Flow Graph to create a new clip list.
- **Insert Above Selected Clip item.** Inserts a clip above the selected clip in the list.


First select a clip in the list, choose Insert Above Selected Clip item, and then click a clip in the Motion Flow Graph.


- **Insert Below Selected Clip item.** Inserts a clip below the selected clip in the list.
- **Append to End of Script.** Appends a clip to the end of the clip list.

 **Create Random Motion.** Displays the *Create Random Motion dialog* (see page 241).


Controls in the Create Random Motion dialog allow you to create random scripts to animate one or more bipeds. Random motion on multiple bipeds can be used to create a crowd scene.

 **Delete Script.** Deletes the current script, displays the previous script if one is present.

 **Create Unified Motion.** Converts a script into a Freeform unified motion. The created motion replaces animation present when Motion Flow Mode is turned off.

 **Edit Clip.** Displays the *Clip Properties dialog* (see page 246). Change the start and end frame for the current clip or replace the current clip with another one. You can also set the Random Start Probability here. Random Start Probability is used when multiple clips are selected as possible start clips when you generate a random motion flow.

The clip name in the script list and the icon in the graph window are updated if the clip is replaced.

 **Edit Transition.** Displays the *Transition Editor* (see page 237) for the selected clip.

Edit the transition for the selected clip and the clip following it. By default, when a script is created, Minimum Motion Loss is used to find start frames for the source and destination clips.

Use Edit Transition to select your own start frames or to try out optimized transitions.

Note: Right-clicking a transition “arrow” in the Motion Flow Graph also displays the Transition Editor, but it will only give you the basic transition editing tools. To edit the transition’s Source Clip-Frame Start and the Destination Clip use the Edit Transition button.



**Cut.** Removes the selected clip from the script list and creates a default transition to the next clip on the list.



**Copy.** Copy the selected clip to the clipboard.



**Paste.** Paste a clip from the clipboard.



**Clip Mode.** Edit biped footsteps and limbs for the selected clip. Use Set Key on the Keyframing rollout to set biped limb keys.



**Go to Frame.** Make the first frame of the selected clip the current frame.

## Move the Entire Animation

**Start Frame.** Set the start frame for the first clip in the script.

**Start Position X.** Move the entire script along a world X-axis.

**Start Position Y.** Move the entire script along a world Y-axis.

**Start Position Z.** Move the entire script along a world Z-axis.

**Start Rotation.** Rotate the entire script around the world Z-axis.

All transformation and rotation is based on the original position.

## Transition Editor

Select a biped > Motion panel > General rollout > Motion Flow Mode > Motion Flow Script rollout > Select a clip in the list window > Transition Editor

Display the Transition Editor by selecting a clip in the list of the Motion Flow Script rollout and clicking Edit Transition on the same rollout or right-click a transition arrow in the Motion Flow Graph.

## Transitions

A good transition links two clips together seamlessly, the motion through the transition should appear natural, as though the motion was captured as one long motion sequence. Like an AB roll transition in video editing, an appropriate section in both clips is selected for the transition (dissolve) from the source clip to the destination clip. Velocity differences between the source and destination clips are matched during the period of transition producing a seamless result. By default, Minimum Motion Loss is used to find likely start frames in the source and destination clips when clips are appended to a script. Optimized transitions can be computed by using Optimize Transition in the upper right hand corner of the Transition Editor dialog. Optimized transitions use a minimum foot sliding method to compute transitions. Optimized transitions take longer to compute, but yield better results.

Although only one arrow is used to represent a transition between two clips in the Motion Flow Graph, any number of transitions can be named and stored in the Transition Editor representing that transition. If, for example, you create 5 different transitions between two clips for one script, all of these transitions are available in a

new script that uses the same two clips. Think of Motion Flow Graph as a data storage area; if all of the scripts are deleted, the transitions are preserved and can be stored in a *.mfe* file.

## Automatic Transitions

When you create a script, default transitions are used between the clips. Default transitions use minimum motion loss and are quick to compute. However, the best quality transitions are the optimized transitions. Once the Transition Editor is open, the first thing to try, before manual editing, are the optimized transitions (upper-right corner of the dialog). Optimized transitions use a minimum foot sliding algorithm to compute the transition and yield very good results.

## Length (Transition Duration)

Set the duration of a transition in the Length field. A value of 10, for example, creates a transition of 10 frames between the source and destination clips. During the period of transition, the velocity of the source clip is interpolated to the velocity of the destination clip. If the transition takes place at the last frame of the source clip and the first frame of the destination clip, and Length is set to 10, then the last frame of the source clip is interpolated with the first 10 frames of the destination clip.

## Editing Transitions Manually (Ghosts)

Manually setting the Start Frame for the source and destination clips offers the most control. Unwanted motion in either clip can be avoided and judging the best Start Frames for both clips is left to you.

The Ghost area Frame spinners allow you to view and scrub the source and destination clips by displaying stick figures (ghosts); yellow and

red stick figures represent the source and destination clips. When a suitable Start Frame is located for both clips, use Set Start Frame in the Ghost areas to copy the Frame values to the Start Frame fields in the Source and Destination Clip areas.

Scrubbing the time slider over the transition period enables you to view the biped's transition from the yellow stick figure (source) to the red stick figure (destination).

## Other Transition Editor Features



Rolling and Fixed specify whether a clip is rolling (in motion) or fixed (single frame) during the transition. Change the direction of the destination clip using the Angle field.

Other parameters in the Transition Editor allow you to create and name new transitions, scroll through the saved transitions, Jump to the transition-starting frame, set automatic transition parameters and go to the next transition in the script

All transitions and their attributes are saved with a *.mfe* file.

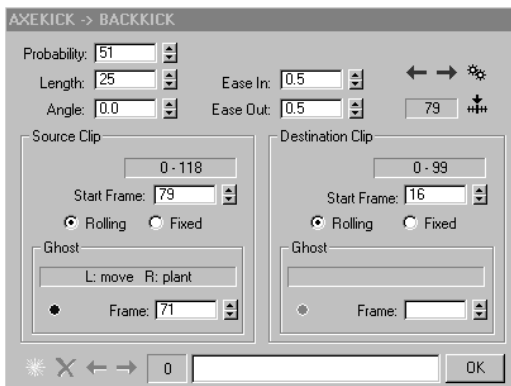
## Procedure

### To create a manual transition between two clips

1.  Select a biped and turn on Motion Flow mode on the General rollout.
2. *Create* (see page 235) or load a script with at least 2 clips.
3. Select a clip in the clip list on the Motion Flow Script rollout.
4.  Click Edit Transition to display the Transition Editor for the selected clip and the clip following it in the list.

5. Scrub the Ghost area Frame spinners in the Source and Destination areas to find a place in both clips to start the transition.  
Two stick figures appear to help you find an appropriate start frame.  
The stick figures may be positioned far apart during this process, this does not matter, look for body motions in the two clips that will transition well.
6. Click Set Start Frame in both Ghost areas when a good Start Frame is located.  
**character studio** repositions the destination clip for a best body fit between the two clips.
7. Set clip duration in the Length field.  
Clip duration of 10 to 25 frames is normal.
8. Click OK.
9. Click Play or scrub the time slider to view the transition. You should see one red bone biped (the destination clip), one yellow bone biped (the source clip), and a biped that has the motions of the synthesis of the two clips.

## Interface



**Probability.** Set a probability value for the transition. This is used by Create Random Motion when a random script is generated.

**Length.** Sets the number of frames for the duration of the transition.

Transitions are calculated by matching velocities in both clips. Smooth out abrupt velocity changes using longer transitions.

**Angle.** Sets the direction of the destination clip.

The angle of the destination clip is automatically set for best body fit between the two clips when the Start Frame values change. Use Angle to change the direction of the destination clip.

**Ease In.** Ease in value for the source clip.

**Ease Out.** Ease out value for the destination clip.



**Previous Transition.** Go to the previous transition in the script.

Displays the previous transition in the Transition Editor, moves the time slider to the start frame of the previous transition and highlights the previous clip in the Motion Flow Script list.

**Next Transition.** Go to the next transition in the script.

Displays the next transition in the Transition Editor, moves the time slider to the start frame of the next transition and highlights the next clip in the Motion Flow Script list.

**Optimize Transition.** Displays the *Transition Optimization dialog* (see page 245).

Options in the Transition Optimization dialog allow you to search for the location for the transition.

**Go To Start Frame.** Moves the time slider to the first frame of the transition.

The number field shows the start frame number.

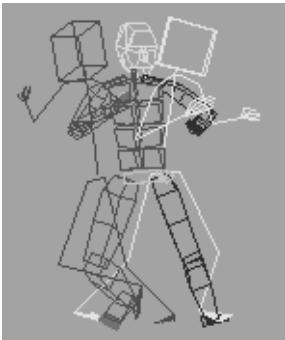


**Start Frame.** Set the transition start frame for the source and destination clips in their respective fields. Duration for the source and destination clips display above the Start Frame fields.

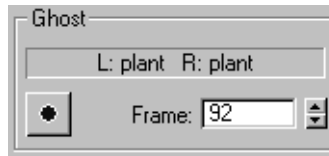
**Rolling.** Keep the clip in motion during the transition.

**Fixed.** Freeze the biped at the Start Frame position during the transition.

### Ghost group



The Ghost area Frame spinners allow you to view and scrub the source and destination clips by displaying stick figures (ghosts); yellow and red stick figures represent the source and destination clips. The source and destination bipeds may not be close to each other during this scrubbing process; the destination clip will be repositioned when Set Start Frame is clicked. When a suitable start frame is located, click Set Start Frame to copy the values in the Frame field to the Start Frame field. Monitor foot position status in the field provided.



**Set Start Frame.** Copy the value in the Frame field of the Ghost area to the Start Frame field in the Clip area. The position of the destination clip changes to match the biped body in the destination clip to the biped body in the source clip.

Locate an appropriate start frame for the source and destination clips by using the Frame spinner and viewing the positions of both stick figures, then click Set Start Frame.

The destination clip is rotated and positioned to match both bipeds. Use the Angle spinner to reorient the destination clip.

**Frame.** Use the Frame spinner to scrub a stick figure back and forth, which allows you to determine a start frame for the source and destination clips. Visual feedback of the stick figures is a good way to judge which start frames are needed for the source and destination clips.



**Create Transition.** Click to create a new transition. The transition number field increments. Edit and name the new transition. Any number of transitions can be stored.

Note: Clicking OK saves the displayed transition. Create Transition is used only if you want to work on a new transition.

**Delete Transition.** Click to delete a transition.

**Previous Transition.** Go to the previous stored transition. This button is grayed if no previous transition exists.



**Next Transition.** Go to the next stored transition. This button is grayed if no next transition exists.

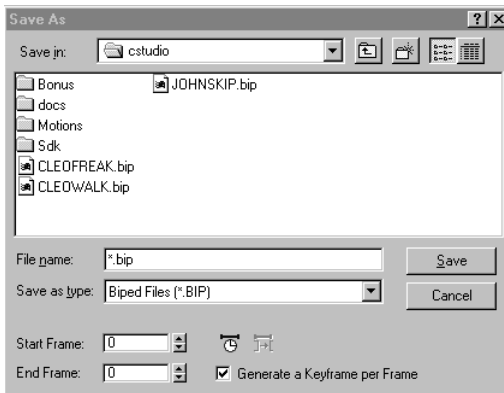
**Ok.** Store transitions and exit the dialog.

## Save Segment Dialog

Select a biped > Motion panel > General rollout > Save Segment

A modified Save As dialog displays when you click Save Segment on the General rollout of the Motion panel. Special controls are at the bottom of the dialog.

### Interface



**Start Frame.** Starts saving at this frame of the named animation.

**End Frame.** Stops saving at this frame of the named animation.

**Active Time Segment.** Click to save animation in the active time segment.

**Edited Transition Interval.** Saves frames in a transition only. Select a clip on the Motion Flow Script rollout, open the Transition Editor and then select Save Segment to activate this control.

Game developers can save only the motion in the transition.

**Generate a Keyframe per Frame.** Saves a key at every frame. Select this check box if you plan to extract footsteps using Load Motion Capture on the Motion Capture rollout. The footstep proximity procedure for extracting footsteps needs keys at every frame to judge foot motion and extract footsteps.

### Save Segment in Motion Flow Mode

Use Save Segment from Motion Flow mode, *turning off* the Generate a Keyframe per Frame check box, to create one *.bip* file that bridges all of the transitions in a Motion Flow script. Footsteps are not saved; feet are assigned an IK Blend value of 1, with Object turned on for the keys that represent footstep duration. This IK constraint prevents the feet from slipping in a freeform animation. After loading this file, use Convert on the General rollout to extract footsteps. Convert bases footstep extraction on foot IK Blend values.

Note: You can also use Create Unified Motion on the Motion Flow Script rollout to transfer all the animation in Motion Flow Mode into your scene. The animation will then be available when you turn off Motion Flow Mode.

## Create Random Motion Dialog

Select a biped > Motion panel > Motion Flow Mode > Motion Flow Script rollout > Create Random Motion > Create Random Motion dialog

You can randomly traverse clips in a motion flow graph to animate one or more bipeds using controls in the Create Random Motion dialog.




Parameters for random motion are set in the Motion Flow Graph, in the clip and transition dialogs, as well as in the Create Random Motion dialog.

Random motion is created by first adding clips to the Motion Flow Graph window and adding transitions between the clips. Clips and transitions are then given percentages, which are used to create random motion for one or more bipeds. You can manually control the “weighting” for possible start clip, transitions and frame range. This allows you to animate multiple bipeds in a crowd scene for example. A different script is created for each biped.


If you want to create random motion for multiple bipeds, they must be sharing a motion flow.

## Procedure

### To create a random script for one biped

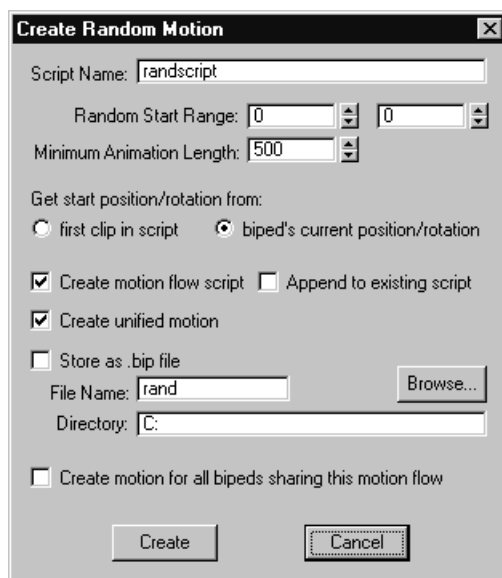
1.  On the Motion Flow rollout click Show Graph  
This displays the Motion Flow Graph.
2. On the Motion Flow Graph window add clips.  
The Create Multiple Clips command on the Motion Flow Graph allows you to add multiple clips.
3.  On the Motion Flow Graph window click Synthesize Motion Flow Graph.  
Every possible transition is computed; this can take some time.
4.  On the Motion Flow Graph window click Select Random Start Clips.
5. On the Motion Flow Graph window region-select all the clips.

The clips turn a purple color. All the clips have an even chance of starting first.

6.  On the Motion Flow Script rollout click Create Random Motion.
7. On the Create Random Motion dialog set a value for Minimum Animation Length.  
This should be long enough to include the clips you want in the animation.
8. Click Create.

A random script is created based on clips in the Motion Flow Graph window. You can vary clip and transition percentages in the clip dialog or transition editor to favor a clip or transition if you like.

## Interface



**Script Name.** Type a name for the script to be generated.

**Random Start Range.** Set the start and end frame range over which the new script(s) will start.

**Minimum Animation Length.** Set the minimum animation length.

When a random motion is created, it is done by making a motion flow script which traverses the clips in the motion flow graph, adding clips based on random calculations. It will add clips until the length of the script is greater than or equal to the minimum animation length, specified here.

**first clip in script.** Gets the start position and rotation from the first clip.

**biped's current position/rotation.** Uses the biped's current position to start the script.

**Create motion flow script.** Creates a script after computing the motion.

If this is on, a motion flow script will be created. If not, a script will still be created internally in order to generate the random motion, but it will be deleted after the random motion is generated.

**Append to existing script.** Appends random motion to the existing script.

**Create unified motion.** Creates a unified motion. The generated motion will be available when you exit Motion Flow mode.

If multiple bipeds are in the random calculation then the motion is unified for each biped.

**Store as .bip file.** Stores the random motion as a .bip file.

If multiple bipeds are in the random calculation they are saved separately with incrementing numbers.

**File Name.** Type a name for the .bip file.

A .bip extension will be added automatically.

**Directory.** Type a directory path or browse for the path.

**Browse.** Browse to a directory.

**Create motion for all bipeds sharing this motion flow.** Turn on to create a random script for each biped sharing the current biped's motion flow.

You can create a shared motion flow by using the Shared Motion Flow command on the Motion Flow rollout and adding bipeds in the Shared Motion Flow dialog.

**Create.** Creates random motion for the selected biped or all the bipeds in the shared motion flow.

**Cancel.** Cancel and close the dialog.



## Shared Motion Flow Dialog

Select a biped > Motion panel > General rollout > Motion Flow Mode > Motion Flow rollout > Shared Motion Flow > Shared Motion Flow Dialog

Controls in the Shared Motion Flow dialog allow you to assign one motion flow to multiple bipeds. Rather than building a motion flow network of clips for each biped you can create a motion flow with all the clips and transitions to animate multiple bipeds. Random motion creation will use each bipeds' own motion flow. If a biped's motion flow happens to be a shared motion flow, then the shared motion flow will be used to compute random motion.

A biped that shares a motion flow shares only the graph. Its scripts are unique to that biped, although the scripts point to the clips of the shared motion flow. You can manipulate that biped's motion flow and scripts in the usual ways. You can create random motion on a biped that shares a motion flow, or create a motion flow script via the crowd system.

There are a few indicators in the user interface that show if a biped is using a shared motion flow. If it is, a white circle will surround the



Shared Motion Flow icon when you edit that biped. If you edit that biped's motion flow graph, the title of the graph dialog will say "Shared Motion Flow Graph", followed by the name of the shared motion flow.

Bipeds in a shared motion flow should have the same lower body scale and structure. Adaptation for differently sized bipeds does not occur in a shared motion flow. If you want differently sized bipeds in a crowd, then create a shared motion flow for *each* size.

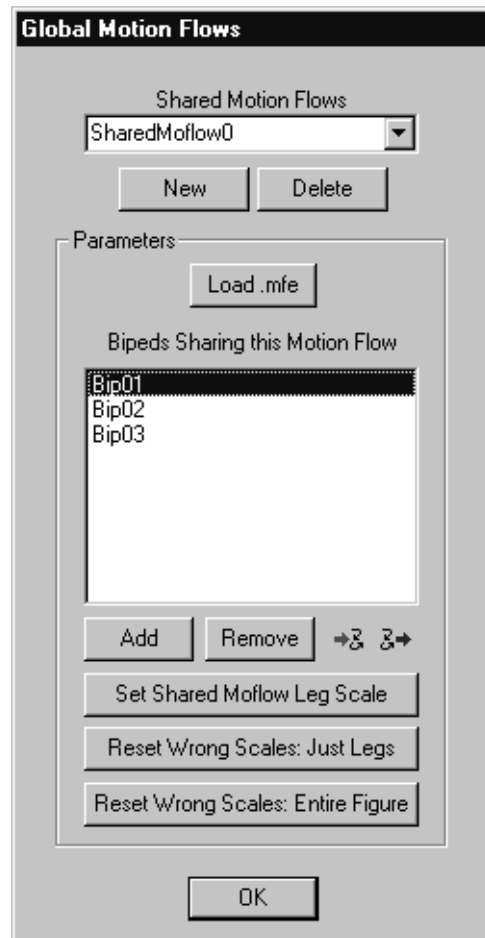
**Tip:** The only way to save a shared motion flow, along with the bipeds sharing it, and keep everything hooked up correctly is to save it all in a *.max* file.

## Procedure

### To share a motion flow among multiple bipeds

1.  Turn on Motion Flow Mode.
2.  On the Motion Flow rollout click Shared Motion Flow.  
The Shared Motion Flows dialog displays.
3. On the Shared Motion Flows dialog click New.  
A new shared motion flow is created.
4. On the Shared Motion Flows dialog click Add.
5. On the Select dialog choose the bipeds that will share the motion flow.  
The bipeds are added to the list. If you add clips to the motion flow graph they will be shared by the bipeds. If you create a random motion, all the bipeds will use the same motion flow graph.

## Interface



**Shared Motion Flows List.** Lists shared motion flows.

**New.** Creates a new shared motion flow.

**Delete.** Deletes the current shared motion flow. The scripts of the bipeds sharing the deleted motion flow will be deleted. Those bipeds will have an empty motion flow graph and no scripts.

**Load .mfe.** Displays a load file dialog. Load a .mfe file into the shared motion flow.

Note: If you load a .mfe file in the usual way via the Motion Flow rollout into a biped using a shared motion flow, you will get a warning and the biped will be removed from the shared motion flow. The biped will get the newly loaded motion flow and all its scripts. The shared motion flow will remain the same.

**Bipeds Sharing this Motion Flow List.** Lists bipeds that share this motion flow.

**Add.** Displays a dialog where you can choose bipeds to add.

Add bipeds to the list of bipeds that share a motion flow. When you add a biped, its current motion flow graph and motion flow scripts will be deleted. It will now have the shared motion flow graph. A biped can only share one motion flow graph. You must remove a biped from its shared motion flow in order to add it to a different shared motion flow.

**Remove.** Removes the selected biped(s) in the list from the current shared motion flow.

These bipeds' scripts will be deleted. They will have an empty motion flow graph.



**Put Multiple Bipeds in Motion Flow.** Put the bipeds in the list into Motion Flow Mode.

The crowd system needs Motion Flow Mode to be turned on to perform calculations for motion. This is a convenient way of turning on Motion Flow Mode for multiple bipeds.



**Take Multiple Bipeds out of Motion Flow.**

Take the bipeds in the list out of Motion Flow Mode.

**Set Shared Moflow Leg Scale.** Adapts the shared motion flow to the scale of the biped currently selected in the list. After this operation, the

selected biped will have the correct leg scale, although other bipeds may not.

**Reset Wrong Scales: Just Legs.** Reset the leg scale only of the bipeds who have the wrong scale, so that they adapt appropriately to the shared motion flow.

**Reset Wrong Scales: Entire Figure.** Resets the entire figure structure of the bipeds who have the wrong scale, to match the figure structure of the correctly scaled biped.

When you add the first biped to the shared motion flow, the system adapts to the size of that biped. If the leg scale of a biped you add subsequently doesn't match that of the first biped, it will be marked in the list with "wrong scale" after it.

All the bipeds you plan on using in a shared motion flow must have the same lower body structure and scale.

---

## Transition Optimization Dialog


Motion Flow Mode > Motion Flow Graph > Select transitions in the graph window. > Optimize Transitions > Transition dialog

Options in the Transition Optimization dialog allow you to select the range over which the optimize algorithm will search for the transition. It can search either the whole clip, or it can search near the existing transition. If it searches the whole clip, you must specify the preferred length of the optimized transition. The software will try to get as close to that length as possible, still opting to give you the best length. If it searches about the existing transition, you must specify the number of frames about which it will search before and after the existing transition. Optimized transitions compute for minimum

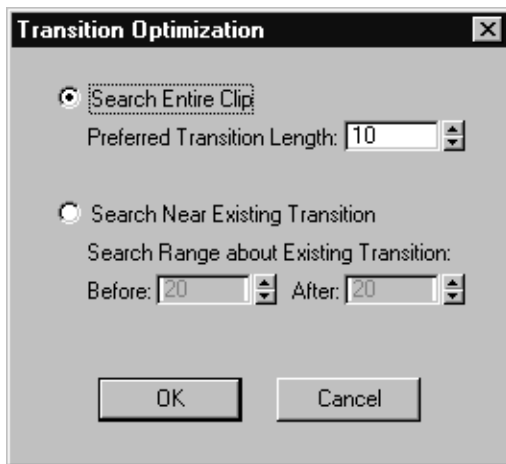
foot sliding over the range of the transition. This method yields very high quality results.

## Procedure

### To use the Optimize Transition Range dialog

1. Select transitions in the Motion Flow Graph.
2.  Click Optimize Selected Transitions.
3. Enter a transition length.
4. Click OK.  
Optimized transitions that use minimum foot sliding are computed.

## Interface



**Search Entire Clip.** Search the entire clip for an optimized transition start frame.

**Preferred Transition Length.** Specify the length of the optimized transition.

**Search Near Existing Transition.** Create an optimized transition near the existing transition.

Search Range about Existing Transition

**Before.** Set a frame value to search before the existing transition.

**After.** Set a frame value to search after the existing transition.

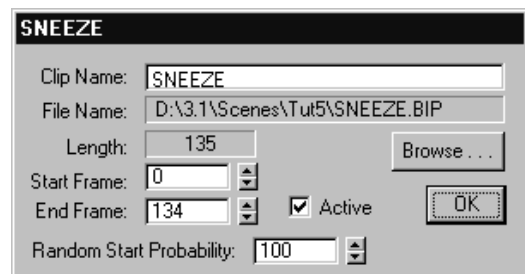
## Clip Properties Dialog

Motion Flow Mode > Motion Flow Graph > Right-click a clip in the Motion Flow Graph window. > Clip Properties dialog

Motion Flow Mode > Motion Flow script rollout > Select a clip in the list > Edit Clip > Clip Properties dialog

Parameters in the Clip Properties dialog allow you to browse for a clip, set a start and end frame for the clip and set a random start probability for the clip. Random start probability is used when you use Create Random Motion to generate a random script for a biped.

## Interface



**Clip Name.** The clip name as it appears in the Motion Flow Graph.

**File Name.** The path and file name of the motion clip.

**Length.** The length of the clip in frames.

**Browse.** Displays a load file dialog. Browse for a motion file.

**Set lowest starting foot height to Z=0 (.bip files only).** Sets the lowest starting foot height to Z=0. This is an option in the Load File dialog. Default=On.

In **character studio 3** the height of a motion clip can be retained. This is important if you want to retain the height of a motion clip for motions adapted to characters of different sizes. If, for example, the character is jumping a rock and you want to retain the Z position of the character, you would turn this option off. Leave this option off if Motion Flow motions must be blended that begin and end at different heights, such as three clips that have the character mounting a bicycle, riding the bicycle, and dismounting the bicycle.

Turning off this option can, however, cause a jump in the motion during motion flow transitions. Turn this on for smooth transitions in Motion Flow mode. If adaptation takes place, the height is set so that the lowest foot at frame 0 starts at the Z=0 height. This lines up clips along the Z axis and creates smooth transitions.

**Start Frame.** Sets the start frame for the clip.

**End Frame.** Sets the end frame for the clip.

**Active.** Activates the clip. Inactive clips display as a green color in the Motion Flow Graph.

**Random Start Probability.** Set a percentage for random start probability. This is used when multiple clips are selected as possible starting clips in a random motion flow. The Create Random Motion command allows you to generate random motion for one or more bipeds.

---

## Footstep Creation

---

### Footstep Mode

Select the Biped > Motion panel > General rollout > Footstep Mode

When Footstep mode is active, you can create or edit footsteps to generate a walk, run, and jump footstep pattern. You also edit selected footsteps in space and append footsteps using parameters available in Footstep mode.

If footsteps are extracted during motion capture import, turn on Footstep Mode to edit footsteps in the viewports.

A powerful feature in Biped is the ability to adapt keyframes when footsteps are edited in space or time. The following tracks are influenced in the vicinity when a footstep is edited in space:

- Body Horizontal keys change to step or hop within range of new footstep locations.
- Body Vertical keys change to match possible changes in stride length, since the body must be lower in order to step longer distances.
- Body Rotation keys change to bank into turns created by changes in path curvature or body speed.
- Right or left leg keys in a move state must be adapted to step between new locations.

Note: If for some reason you do not want the adaptation to occur, use Adapt Locks on the *Animation Properties* rollout (see page 181) to keep the biped from correcting the body position.

Two additional rollouts display when Footstep mode is active: *Footstep Creation* rollout (see page 248) and *Footstep Operations* rollout (see page 258).

## Footstep Creation Rollout

Select a Biped with footsteps > Motion panel > General rollout > Footstep Mode > Footstep Creation rollout


The Footstep Creation rollout, available on the Motion panel when Footstep mode is on, provides controls for creating and editing footsteps. Create a walk, run, or jump footstep pattern using these controls.

**Tip:** All footsteps created here are inactive; you activate them using Create Keys for Multiple Footsteps on the *Footstep Operations* rollout (see page 258).

The timing parameters at the bottom of the rollout work with Create Footsteps (append) and Create Footsteps (at current frame) to change timing for newly created footsteps. These change depending on whether you select Walk, Run, or Jump mode.

## Procedures



### To create multiple footsteps

1.  On the General rollout, click Footstep Mode.  
You are now in Footstep mode, and can create, activate, or edit footsteps.
2. On the Footstep Creation rollout, choose the gait you want to create — Walk, Run, or Jump.
3. Click Create Multiple Footsteps.  
The Create Multiple Footsteps dialog displays for the selected gait.
4. Set multiple footstep parameters, and then click OK.


5.  On the Footstep Operations rollout, click Create Keys for Inactive Footsteps to activate the footsteps.

**Tip:** Use Set Multiple Keys on the Keyframing rollout, then turn on Track View to edit this motion.

### To create footsteps manually, beginning at the current frame

1. On the Footstep Creation rollout, click Walk, Run, or Jump and set timing parameters for that gait.
2.  Click Create Footsteps (at current frame).
3. Click in a viewport to create a footstep. Continue clicking to create more footsteps.
4.  On the Footstep Operations rollout, click Create Keys for Inactive Footsteps.
5. Play the animation.


### To append footsteps onto the existing footsteps

1.  On the Footstep Creation rollout, click Create Footsteps (append).
2. Click in a viewport to create a footstep. Continue clicking to create more footsteps  
By default, footsteps alternate from foot to foot. The first click creates a right footstep, the next click creates a left footstep, and so on. Look at the prompt line and the cursor to see which type of footstep will be created next.



## Interface



 **Create Footsteps (append).** Turn on Create Footstep mode. Manually place footsteps by clicking in any viewport. Hold the mouse button down after clicking to move a footstep. Release the mouse button to place the footstep.

Each new footstep is appended to the end of the biped's footstep sequence. Create Footsteps alternates right and left footsteps as you create new ones. Press Q to toggle between a left and right footstep.

Newly created footsteps are bright green for right footsteps and bright blue for left footsteps. Once the footsteps have been activated, the footsteps change color to pastel green and pastel blue.



**Create Footsteps (insert at current frame).** Create footsteps at the current frame. Footstep creation alternates between left and right footsteps.



**Create Multiple Footsteps.** Create a walk, run, or jump footstep pattern automatically. Select a gait type before using Create Multiple Footsteps. Displays the Create Multiple Footsteps dialog. The dialog differs slightly depending on the gait (walk, run, jump) you have selected.

## See Also

*Create Multiple Footsteps Dialog (Walk) (see page 250)*

*Create Multiple Footsteps Dialog (Run) (see page 253)*

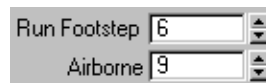
*Create Multiple Footsteps Dialog (Jump) (see page 255)*



**Run.** Sets the biped gait to Run. Any footsteps you add will have run characteristics until you change to another mode (walk or jump). Each new footstep will start after the end of the previous footstep on the opposite side.



**Jump.** Sets the biped gait to Jump. Any footsteps you add will have jump characteristics until you change to another mode (walk or run). Each new footstep will start at the same time as the most recent footstep on the opposite side. Alternately, it may start after the end of the previous footstep.



## Timing parameters

Use these parameters with Create Footsteps (append) or Create Footsteps (at current frame) to apply timing to newly created footsteps. These parameters are different for each gait and change as you select a different gait.

**Walk Footstep (Walk only).** Specifies the number of frames a new footstep will be on the ground during a walk.

**Double Support (Walk only).** Specifies the number of frames when both feet will be on the ground at the same time during a walk.

**Run Footstep** (Run only). Specifies the number of frames a new footstep will be on the ground during a run.

**Airborne** (Run and jump only). Specifies the number of frames when the body will be in the air during a run or a jump.

**2 Feet Down** (Jump only). Specifies the number of frames when two sequential footsteps, on opposite sides, will be on the ground during a jump.

## Create Multiple Footsteps Dialog: Walk

Select the Biped > Motion Panel > General rollout > Footstep Mode > Footstep Creation rollout > Walk > Create Multiple Footsteps > Create Multiple Footsteps Walk dialog

The Create Multiple Footsteps dialog for the walk gait allows you to create a sequence of walking footsteps by setting a series of parameters. This dialog displays when you've selected Walk on the Footstep Creation rollout on the Motion panel, and then click Create Multiple Footsteps.

## Procedures

### To walk up stairs

- In the First Step group, set Actual Stride Height to a value greater than zero.

### To walk down stairs

- In the First Step group, set Actual Stride Height to a value less than zero.

### To walk in place

- In the First Step group, Set Parametric Stride Length to zero.

### To walk backwards

- In the First Step group, set Parametric Stride Length to a value less than zero.

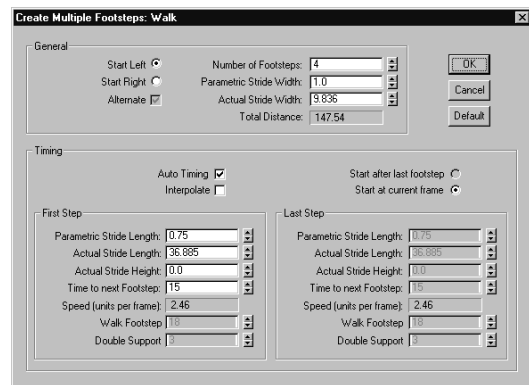
The absolute value of the Parametric Stride Length is still the length of the stride.

### To make the biped speed up as it walks

- In the Timing group, click Interpolate. The controls in the Last Step group are enabled.
- In the Last Step group, set Time to Next Footstep to be a value less than Time to Next Footstep in the First Step group.

You can adjust the values in either group, or both. The important thing is to make the Last Step a shorter time than the First Step.

## Interface



**Start Left.** Starts the footstep sequence with a left step.

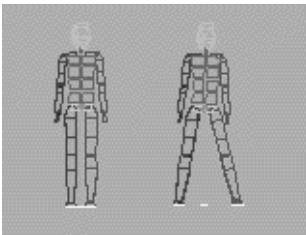
**Start Right.** Starts the footstep sequence with a right step.

**Alternate.** Footsteps will alternate between right and left. Alternate is always selected when the Walk gait is selected. You can only clear Alternate when Run or Jump gaits are selected.

**Number of Footsteps.** Determines the number of new footsteps to be created.

**Parametric Stride Width.** Sets the stride width as a percentage of the pelvis width. A value of 1.0 produces a stride width equal to the pelvis width. A value of 3.0 produces a wide, waddling stride. Changes to this setting automatically change the Actual Stride Width.

Parametric describes the parameter in terms of biped anatomy, and Actual describes the value in 3DS MAX units.



Stride Width=1 and Stride Width=3

**Actual Stride Width.** Sets the stride width in modeling units. Changes to this setting automatically change the Parametric Stride Width.

## Timing

**Auto Timing.** Sets timing parameters automatically. Auto Timing affects the following timing parameters for the Walk gait:

- Walk footstep, Double Support

When Auto Timing is selected, these parameters are automatically adjusted to reasonable values. Control the footstep sequence by adjusting the Stride Length and Time to Next Footstep parameters.

When Auto Timing is off, you can control the footstep sequence by adjusting the gait timing parameters, but you can't change the Time to Next Footstep parameter.

**Interpolate.** Control acceleration or deceleration of the series of footsteps. When this box is selected, a second set of step parameters under Last Step is enabled.

Biped creates footsteps starting with the values of the parameters under First Step and ending with the values of the parameters under Last Step.

By interpolating between the two, Biped produces a footstep series that changes over time.

When Interpolate is cleared, the Last Step parameters are grayed out. Biped creates all the footsteps using only the parameters under First Step.

**Start After Last Footstep.** Appends the newly created footsteps to the end of the existing footstep sequence.

**Start at Current Frame.** Inserts the newly created footsteps at the current frame after the existing footstep sequence allowing you to make a gap in time before the footsteps start again.

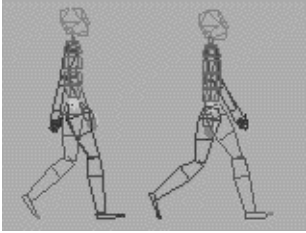
## First Step and Last Step

**Parametric Stride Length.** Sets the stride length for the new footsteps as a percentage of the length of the biped's leg. The default value of 0.75 gives an average stride of normal proportions.

A value of 1.0 will produce a stride length equal to the leg length, which makes the biped stretch slightly to reach the next step. A value of 0.0 will make the biped walk in place. A negative stride length will make the biped walk backwards.

When a biped walks backwards, it does not simply reverse the forward movement but maintains the correct foot-state sequence with the toe touching the ground first, followed by the heel.

Adjusting Parametric Stride Length automatically changes the value for Actual Stride Length.



**Stride Length=0.75 and Stride Length=1**

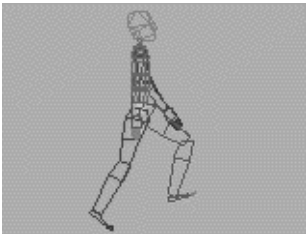
**Actual Stride Length.** Sets the stride length for new footsteps in 3DS MAX units.

The same rules apply as for Parametric Stride Length (described above).

Adjusting Actual Stride Length automatically changes the value for Parametric Stride Length.

**Actual Stride Height.** Sets the rise or fall between footsteps. You can use this parameter to create a set of footsteps going up or down a slope or a stairway.

The value for Actual Stride Height is the difference in height in units between each of the new footsteps. Positive values step up and negative values step down.



**Stride Height=5 units**

**Time to Next Footstep.** Specifies the number of frames in each foot movement cycle. A cycle starts with the frame that a foot touches the

ground, continues as the foot lifts and moves, and ends with the frame before the foot touches the ground again. This parameter is only enabled if Auto Timing is on.

**Speed.** Displays the number of units the biped will move per frame. It changes in response to changes in the other parameters but cannot be adjusted directly.

The following two parameters are only enabled when Auto Timing is off.

You can use these parameters instead of Auto Timing to control the speed of the forward motion over the series of footsteps. However, because these parameters both affect the footsteps' time in contact with the ground, using them to slow down a walk gives the walk a hesitant, 'stop-go' quality.

**Walk Footstep.** Specifies the number of frames each footstep will be on the ground during a walk.

The higher the number, the longer each biped foot remains in contact with the ground and, consequently, the slower the speed of the walking motion.



**Footsteps 3 through 5 are on the ground for 22 frames each**

**Double Support.** Specifies the number of frames both feet will be on the ground at the same time during a walk.

The higher the number, the longer the period during which both feet remain in contact with the ground during each walk cycle and, consequently, the slower the speed of the walking motion.



The dotted line surrounds the double-support period (6 frames)

## Create Multiple Footsteps Dialog: Run

Select the Biped > Motion Panel > General rollout > Footstep Mode > Footstep Creation rollout > Run > Create Multiple Footsteps > Create Multiple Footsteps Run dialog

The Create Multiple Footsteps dialog for the run gait allows you to create a sequence of running footsteps by setting a series of parameters. This dialog will display when Run is selected on the Footstep Creation rollout first, and then Create Multiple Footsteps is clicked.

### Interface

**Start Left.** Starts the footstep sequence with a left step.

**Start Right.** Starts the footstep sequence with a right step.

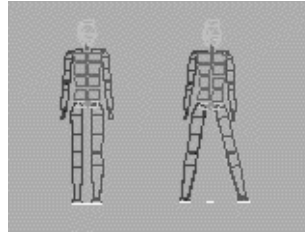
**Alternate.** Turn on to alternate between right and left footsteps. When this box is cleared, all of the footsteps will be either right or left, producing the effect of a hopping run on one foot.

**Number of Footsteps.** Determines the number of new footsteps to be created.

**Parametric Stride Width.** Sets the stride width as a percentage of the pelvis width.

A value of 1.0 produces a stride width equal to the pelvis width. A value of 3.0 produces a wide,

waddling stride. Changes to this setting automatically change the Actual Stride Width. Parametric describes the parameter in terms of biped anatomy, and Actual describes the value in 3DS MAX units.



Stride Width=1 and Stride Width=3

**Actual Stride Width.** Sets the stride width in modeling units. Changes to this setting automatically change the Parametric Stride Width.

### Timing

**Auto Timing.** Sets timing parameters automatically.

Auto Timing affects the following timing parameters for the Run gait:

- Run footstep, Airborne

When Auto Timing is selected, these parameters are automatically adjusted to reasonable values. You control the footstep sequence by adjusting the Stride Length and Time to Next Footstep parameters.

When Auto Timing is off, you can control the footstep sequence by adjusting the gait timing parameters, but you can't change the Time to Next Footstep parameter.

**Interpolate.** Control acceleration or deceleration of the series of footsteps. When this box is selected, a second set of step parameters under Last Step is enabled.

Biped creates the footsteps starting with the values of the parameters under First Step and ending with the values of the parameters under Last Step.

By interpolating between the two, Biped produces a footstep series that changes over time.

When Interpolate is cleared, the Last Step parameters are grayed out. Biped creates all the footsteps using only the parameters under First Step.

**Start After Last Footstep.** Appends the newly created footsteps to the end of the existing footstep sequence.

**Start at Current Frame.** Inserts the newly created footsteps at the current frame after the existing footstep sequence allowing you to make a gap in time before the footsteps start again.

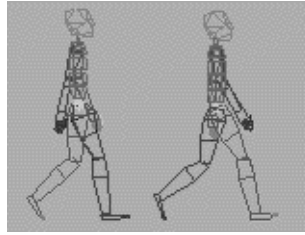
## First Step and Last Step

**Parametric Stride Length.** Sets the stride length for the new footsteps as a percentage of the length of the biped's leg. The default value of 0.75 gives an average stride of normal proportions.

A value of 1.0 will produce a stride length equal to the leg length, which makes the biped stretch slightly to reach the next step. A value of 0.0 will make the biped run in place. A negative stride length will make the biped run backwards.

When a biped runs backwards, it does not simply reverse the forward movement but maintains the correct foot-state sequence with the toe touching the ground first, followed by the heel.

Adjusting Parametric Stride Length automatically changes the value for Actual Stride Length.

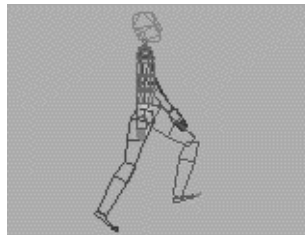


**Stride Length=0.75 and Stride Length=1**

**Actual Stride Length.** Sets the stride length for the new footsteps in 3DS MAX units.

The same rules apply as for Parametric Stride Length (described above).

Adjusting Actual Stride Length automatically changes the value for Parametric Stride Length.



**Stride Height=5 units**

**Actual Stride Height.** Sets the rise or fall between footsteps. You can use this parameter to create a set of footsteps going up or down a slope or a stairway.

The value for Actual Stride Height is the difference in height in units between each of the new footsteps. Positive values step up and negative values step down.

**Time to Next Footstep.** Specifies the number of frames in each foot movement cycle. A cycle starts with the frame in which a biped foot touches the ground, continues as the foot lifts and moves, and ends with the frame before the

foot touches the ground again. This parameter is only enabled if Auto Timing is on.

**Speed.** Displays the number of units the biped will move per frame. It changes in response to changes in the other parameters but cannot be adjusted directly.

The following two parameters are only enabled when Auto Timing is off. You can use these parameters instead of Auto Timing to control the speed of the forward motion over the series of footsteps.

**Run Footstep.** Specifies the number of frames each footstep will be on the ground during the run.

The higher the number, the longer the biped foot remains in contact with the ground and, consequently, the slower the speed of the running motion.



**Footsteps 2 and 3 are on the ground for 5 frames each**

**Airborne.** Specifies the number of frames the body will be in the air between footsteps.

The higher the number, the longer the biped hangs in the air for each step and, consequently, the slower the speed of the running motion.

## Create Multiple Footsteps Dialog: Jump

Select the biped. > Motion Panel > General rollout > Footstep Mode > Footstep Creation rollout > Jump > Create Multiple Footsteps > Create Multiple Footsteps Jump dialog

The Create Multiple Footsteps dialog for the jump gait allows you to create a sequence of jumps by setting a series of parameters. It displays when you've selected Jump on the Footstep Creation rollout on the Motion panel, and then click Create Multiple Footsteps.

### Interface

**Start Left.** Starts the footstep sequence with a left step.

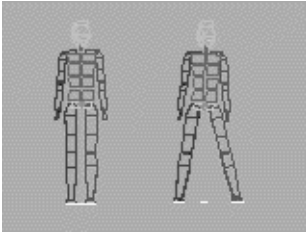
**Start Right.** Starts the footstep sequence with a right step.

**Alternate.** When this box is selected, the footsteps will alternate between right and left. When this box is cleared, the footsteps will be either right or left steps, causing the biped to hop on one foot.

**Number of Footsteps.** Determines the number of new footsteps to be created.

**Parametric Stride Width.** Sets the stride width as a percentage of the pelvis width. A value of 1.0 produces a stride width equal to the pelvis width. A value of 3.0 produces a wide, waddling stride. Changes to this setting automatically change the Actual Stride Width.

Parametric describes the parameter in terms of biped anatomy, and Actual describes the value in 3DS MAX units.



Stride Width=1 and Stride Width=3

**Actual Stride Width.** Sets the stride width in modeling units. Changes to this setting automatically change the Parametric Stride Width.

## Timing

**Auto Timing.** Sets timing parameters automatically.

Auto Timing affects the following timing parameters for the Jump gait:

- 2 Feet Down, Airborne

When Auto Timing is selected, these parameters are automatically adjusted to reasonable values. You control the footstep sequence by adjusting the Stride Length and Time to Next Footstep parameters.

When Auto Timing is off, you can control the footstep sequence by adjusting the gait timing parameters, but you can't change the Time to Next Footstep parameter.

**Interpolate.** Allows you to control acceleration or deceleration of the series of footsteps. When this box is selected, a second set of step parameters under Last Step is enabled.

Biped creates the footsteps starting with the values of the parameters under First Step and ending with the values of the parameters under Last Step.

By interpolating between the two, Biped produces a footstep series that changes over time.

When Interpolate is cleared, the Last Step parameters are grayed out. Biped creates all the footsteps using only the parameters under First Step.

**Start After Last Footstep.** Appends the newly created footsteps to the end of the existing footstep sequence.

**Start at Current Frame.** Inserts the newly created footsteps at the current frame after the existing footstep sequence allowing you to make a gap in time before the footsteps start again.

## First Step and Last Step

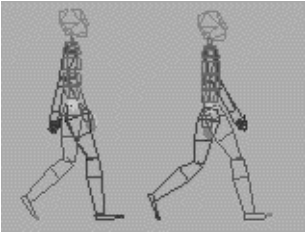
**Parametric Stride Length.** Sets the stride length for the new footsteps as a percentage of the length of the biped's leg. The default value of 0.75 gives an average stride of normal proportions.

A value of 1.0 will produce a jump length equal to the leg length, which makes the biped stretch slightly to reach the next step. A value of 0.0 will make the biped jump in place. A negative stride length will make the biped jump backwards.

When a biped jumps backwards, it does not simply reverse the forward movement but maintains the correct foot-state sequence with the toe touching the ground first, followed by the heel.

Adjusting Parametric Stride Length automatically changes the value for Actual Stride Length.





**Stride Length=0.75 and Stride Length=1**

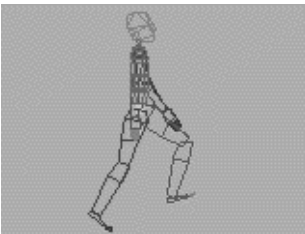
**Actual Stride Length.** Sets the stride length for the new footsteps in 3DS MAX units.

The same rules apply as for Parametric Stride Length (described above).

Adjusting Actual Stride Length automatically changes the value for Parametric Stride Length.

**Actual Stride Height.** Sets the rise or fall between footsteps. You can use this parameter to create a set of footsteps going up or down a slope or a stairway.

The value for Actual Stride Height is the difference in height in units between each of the new footsteps. Positive values step up and negative values step down.



**Stride Height=5 units**

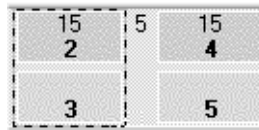
**Time to Next Footstep.** Specifies the number of frames in each foot movement cycle. A cycle starts with the frame a particular foot touches the ground, continues as the foot lifts, moves, and ends with the frame before the foot touches

the ground again. This parameter is only enabled if Auto Timing is on.

**Speed.** Displays the number of units the biped will move per frame. This changes in response to changes in the other parameters but cannot be adjusted directly.

The following two parameters are only enabled when Auto Timing is off. You can use these parameters instead of Auto Timing to control the speed of the forward motion over the series of footsteps.

**2 Feet Down.** Specifies the number of frames that the left and right footsteps will be on the ground during the jump. The higher the number, the longer the pause between each jump and, consequently, the slower the speed of the jumping motion.



**The dotted line surrounds the 2 feet down period (15 frames)**

**Airborne.** Specifies the number of frames the body will be in the air during the jump.

The higher the number, the longer the biped hangs in the air for each jump and, consequently, the slower the speed of the jumping motion.



**The dotted line surrounds the airborne period (13 frames)**


## Footstep Operations Rollout

Select the Biped > Motion Panel > General rollout > Footstep Mode > Footstep Operations rollout

Once footsteps are created on the Footstep Creation rollout, use parameters on the Footstep Operations rollout to activate and deactivate footsteps, and to adjust the footstep path.

### Procedures

#### To move footsteps

1.  On the General rollout, click Footstep Mode.
2. Click Select and Move on the 3DS MAX toolbar.
3. Select footsteps in the viewports.
4. Click and drag to move the footsteps.

#### To rotate footsteps

1. Click Rotate on the 3DS MAX toolbar.
2. Click to select a single footstep, or use CTRL+click, or use a window to select multiple footsteps.
3. Click and drag to rotate the footsteps.  
**Tip:** Rotating a footstep about its local X or Y axis creates the effect that the biped is walking on uneven terrain.

#### To scale the stride length or width

1. User CTRL+click or use a region to select multiple footsteps.
2. On the Footstep Operations rollout, choose Length, Width, or both.
3. Increase Scale to a value greater than one to increase the stride length or width. Decrease it to less than one to decrease the stride length or width.

Immediately after you set the value of Scale, Biped scales the stride and then resets Scale to its default value of one. To put it another way, stride scaling is always based on the current footstep spacing; the Scale parameter is not an absolute value.

#### To bend the footsteps' path

1. Use CTRL+click or use a region to select multiple footsteps.
2. Increase Bend to a value greater than zero to bend the footsteps toward the left. Decrease it to less than zero to bend the footsteps to the right.

Immediately after you set the value of Bend, Biped bends the footstep path and then resets Bend to its default value of zero. To put it another way, bending footsteps is always based on the footstep's current location; the Bend parameter is not an absolute value.

#### To delete footsteps

1. Click to select a single footstep. Use CTRL+click or use a window to select multiple footsteps.
2. Click DEL or click Delete Footsteps on the Footstep Operations rollout.  
Biped deletes the selected footsteps and the movement keys they generated.  
**Note:** If inactive footsteps are present, you are allowed to delete only inactive footsteps, but are not allowed to delete all the inactive footsteps.

#### To activate footsteps

- Click Create Keys for Inactive Footsteps on the Footstep Operations rollout.  
Biped generates the motion keys.

#### To deactivate footsteps

1. Select the footsteps to deactivate.




The footsteps you select must be in sequence. Biped doesn't allow you to deactivate a non-sequential set of footsteps.

2.  Click Deactivate Footsteps.

The footsteps are deactivated. Viewports and Track View will display footsteps in saturated colors again.

Note: Deactivating footsteps does not delete the keys generated when you activated the footsteps. The biped still moves as it did before, and will continue this movement until you click Create Keys again. However, editing inactive footsteps doesn't affect the keys, and activating the footsteps again generates new default keys that replace the previous keys.

### To splice biped motion

1.  On the General rollout, click Footstep Mode to activate that mode.
2. Select the footstep sequence you want to copy.  
Selecting the footsteps by dragging a region is the easiest way to ensure you've selected all of a footstep sequence.
3.  Click Copy Footsteps on the Footstep Operations rollout.  
Biped copies the footsteps and all associated biped motion keys to the footstep buffer. The Buffer Mode button on the General rollout is now enabled.
4.  Click Paste Footsteps on the Footstep Operations rollout.  
The footsteps reappear in the viewports, alongside (or near) the activated footsteps.

The footsteps pasted from the buffer appear in their saturated colors.

5. Use the Move transform to position the first pasted footstep at the point where you want to splice. When you move the first pasted footstep over an activated footstep of the same side—right or left—the activated footstep turns red to indicate splicing is possible.

This step is usually easiest to do in a Top viewport.

Note: If you deselect the pasted footsteps by clicking elsewhere in a viewport, no paste occurs and the pasting process ends. Steps 6 and 7 will not occur. If you accidentally deselect the pasted footsteps, you can paste again by starting over from step 4.

6. Release the mouse button when you see that a splice is possible.

Biped splices the footstep buffer over the active footsteps. The first footstep in the buffer replaces the footstep that turned red and the rest of the buffer footsteps follow.

7. The footstep that turned red and the remaining original footsteps now appear in the viewports. They will be near the activated footsteps, in their saturated colors. They can now be pasted onto the end of the pasted motion, to perform an insert. Or they can also be pasted anywhere in the footstep sequence. You can now follow steps 5 and 6 again, as if these footsteps were in the buffer. (If you pasted onto the last footstep, this step through the end does not occur.)


If you don't want to append the leftover footsteps, simply click anywhere in the viewport to dismiss them.


**Warning:** If any footstep in the buffer overlaps in time with a footstep previous to

the one onto that you are pasting, a message appears and the paste will not be performed.

**Tip:** To create a cycle of a motion with alternating footsteps, you must copy and paste at least three footsteps.

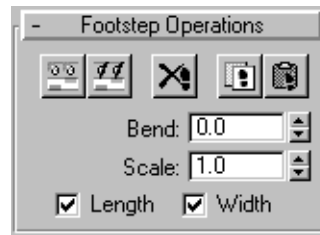
### To edit the footstep buffer before you insert its contents by splicing

1.  On the General rollout, click Buffer Mode.  
This button is active only when there are footsteps in the buffer, after you've clicked Copy Selected Footsteps.  
The viewports now display the footsteps in the Footstep buffer, instead of the footsteps in the currently activated footstep sequence.
2. Edit footsteps or motion keys as you normally would. However, you can't splice (copy and paste) footsteps while in Buffer mode.  
Biped updates the Footstep buffer to reflect your edits.  
If you load a biped (*.bip*) file by clicking Load File in the General rollout while Buffer mode is active, Biped replaces the footsteps and other motion in the Footstep buffer with the motion in the biped file.

3.  Click to turn off Buffer Mode.

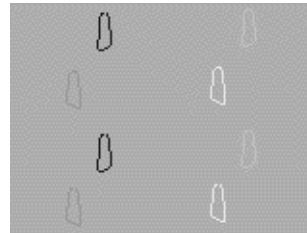
Now you can splice the edited buffer as you would a buffer you copied. See the previous steps in this topic.

## Interface



### Create Keys for Inactive Footsteps.

Activates all inactive footsteps. Activation creates default keys for any footsteps that do not have them. If a footstep does not have keys, it is displayed as bright green (right foot) or bright blue (left foot). After keys are created for the footsteps, the footsteps change color to pastel green and pastel blue.



Inactive footsteps on the left and activated footsteps on the right



**Deactivate Footsteps.** Removes the body keys assigned to the selected footsteps, making those footsteps inactive. The footsteps themselves remain in the scene.



**Delete Footsteps.** Deletes the selected footsteps.

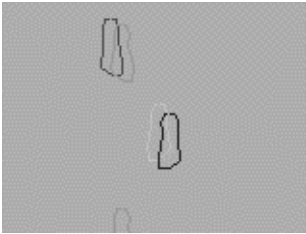


**Copy Footsteps.** Copies the selected footsteps and biped keys to the footstep buffer.

Biped will only copy a continuous sequence of footsteps (2,3,4,5...). You can't copy discontinuous footsteps (3,4,7,8...).

If any footsteps exist that have not been activated, the Copy button is grayed. Activate the footsteps first, then try again.

**Tip:** Turn on Buffer mode on the General rollout to view and edit only the buffered footsteps and biped motion.



A pasted footstep is inactive until it overlaps a like foot and turns red. Mouse up to activate the new footsteps.



**Paste Footsteps.** Pastes footsteps from the footstep buffer into a scene. The footsteps appear inactive in the scene. Move them so they overlap the active footsteps. When a footstep turns red, mouse up and the pasted footsteps will activate.

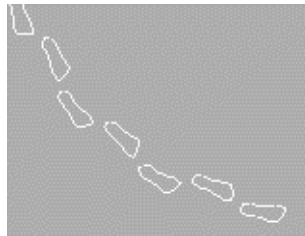
**Warning:** If any footstep in the buffer overlaps in time with a footstep previous to the one onto which you are pasting, a message appears and the paste is not performed. This can occur if the first footstep you are pasting and the original footstep you are pasting onto both have *double support* periods during the same duration of the footstep. The second pasted footstep and the footstep prior to the one you are pasting onto may overlap in time.

To correct the problem, you may want to move the right edge of the preceding footstep from the

original scene to the left, or move the left edge of the second pasted footstep to the right while in buffer mode. Which option you choose depends on what footstep timing and support relations you want to end up with.

**Tip:** To create a cycle of a motion with alternating footsteps, you must copy and paste at least three footsteps. All body keys between the start of the first step and end of the last step are also copied and pasted.

**Bend.** Bends the path for the selected footsteps. The path is bent to the left or right as you move the spinner. Other footsteps after the selected footsteps will be moved to maintain their positions relative to the repositioned footsteps. The Bend spinner is grayed if inactive footsteps are present.

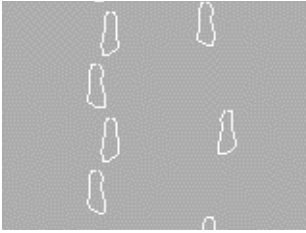


**Bend=0.5**

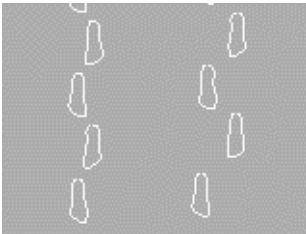
**Scale.** Changes the width or length for the selected footsteps. The selected footsteps are scaled around the first footstep in the selection. First select the width or length box (or both), then use the spinner to set the amount of scaling.

If any footsteps exist that have not been activated, the Scale spinner is grayed out. Activate the footsteps first, then try again.

**Length.** When Length is selected, the Scale spinner changes the stride length of the selected footsteps. Length and Width may both be active at the same time.

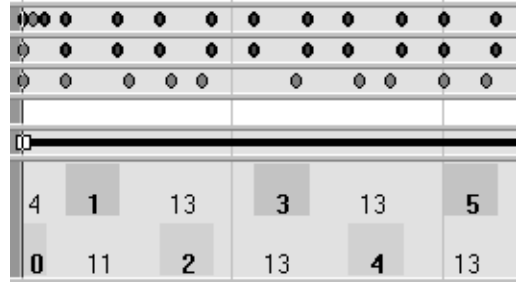
**Length=1 and Length=2**

**Width.** When Width is selected, Scale changes the stride width of the selected footsteps. Length and Width may both be active at the same time.

**Width=1 and Width=2**

## Track View

Main Toolbar > Track View



In this Track View of a running motion, footsteps are represented by colored squares, footstep numbers appear inside the squares, other numbers represent foot-air duration. Red keys in this image show the center of mass vertical and horizontal tracks. Red keys are associated with the center of mass and only appear at touch down keys from an airborne period, or lift keys just before an airborne period. Red keys can only be moved in time by moving the associated footstep.

## Biped and Track View

Track View provides a way to create and adjust biped keyframes, edit biped footsteps, and specify a freeform period. Biped keys display in a familiar “dots on a track.” Footsteps display as squares that can be moved in time. Biped dynamics and footsteps work together; if footsteps are moved in time, Biped adapts leg keys and vertical positions to account for the editing. There are a couple of restrictions, you can’t overlap footsteps on the same track, or move footsteps into negative time.

**Tip:** Select a biped object and click Zoom Selected at the bottom left corner of the Track View dialog to place the selected biped object at

the top of the Track View Hierarchy window. This is a quick way to locate a biped object in Track View.

## Footsteps in Track View

By default, left leg footsteps are blue and right leg footsteps are green. Inactive footsteps are more saturated values of blue and green, active footsteps are pale blue and green. The left edge of each footstep block indicates when a foot touches a footstep (Touch State). The right edge of each footstep block indicates the last frame the foot is on a footstep (Lift State). The space between two footsteps indicates an airborne state for the foot (Move State). The period between the Touch and Lift States is the Plant State.

- If footsteps are hidden in the viewports, turn off Visible Objects in the Show Only group of the Track View Filters dialog; this will allow footsteps to be displayed in Track View.
- State Filters on the *Set Multiple Keys dialog* (see page 269) can select keys based on the “State” a foot is in.

## Separate Tracks

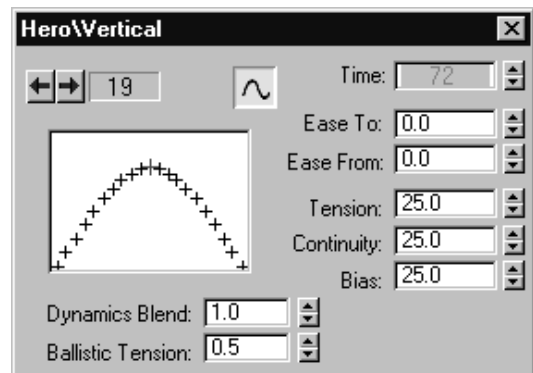
By default, Biped stores all of the toe, foot, and calf keys in the thigh track. The finger, hand, forearms, and upper-arm keys are stored in the clavicle track. All the spine keys are stored in the spine 01 track. Although you can see all of these objects in the Track View hierarchy, they have no transform track, unless you enable them in the Separate Tracks group of the *Animation Properties rollout* (see page 181).

For example, if you rotate a biped foot, a key is created in the biped thigh track. This optimized approach works well in many cases. If your animation requires extensive hand and finger

keyframing, turn on Arms on the Animation Properties rollout, all of the arm transform tracks are now enabled, down to the first finger link on each finger. Now if you delete an upper arm key, your finger-hand animation is preserved.

## Editing Biped Keys in Track View

Track View is often used to change multiple keys simultaneously. For example, to change Dynamics Blend for all the vertical center of mass keys, select all the vertical keys in the Track View window, display the TCB (Tension, Continuity, and Bias) dialog by right-clicking over the selected keys and change the Dynamics Blend parameter.



Right-click a key in Track View to display the TCB dialog

## Edit Keys

If Edit Keys is enabled on the Track View toolbar, the following functions are possible:

- **Clone Keys.** Use SHIFT+drag to clone and position selected key(s) in time.
- **Scale Keys.** On the Track View toolbar, turn on Scale Keys and scale selected keys. The scale center is the current frame.

- **Add Keys.** Turn on Add Keys on the Track View toolbar and add keys to a biped track.

## Edit Time

If Edit Time is enabled on the Track View toolbar, the following functions are possible:

- **Scale Time.** If yours is a purely freeform animation, you can scale time for all tracks of the biped. If it is a footstep-driven animation, you are restricted to scaling time for the footstep track. Turn on Scale Time on the Track View toolbar, select the Footstep transform track in the Track View Hierarchy window, then select and drag to scale footsteps time in the Track View Edit window.
- **Insert Time.** If yours is a purely freeform animation, you can insert time for all tracks of the biped. If it is a footstep-driven animation, you are restricted to scaling time for the footstep track. Turn on Scale Time on the Track View toolbar, select the Footstep transform track in the Track View Hierarchy window, then select and drag to scale footsteps time in the Track View Edit window.
- **Copy and Paste Time.** You can copy and paste time for all biped tracks other than the Footstep track. You could, for example, copy time from the left arm track and paste it to the right arm track. This will copy the left arm keys to the right arm. You can also paste time onto another biped in the scene.

## Set Multiple Keys

Use *Set Multiple Keys* (see page 173) on the Keyframing rollout to automatically select keys for your animation (all of the leg keys in a move state (airborne), for example). This works well with Track View open, all the keys selected by

the state filters in the Set Multiple Keys dialog display as white keys in Track View. Use Apply Increment in the Set Multiple Keys dialog after adjusting a biped limb to change the selected keys.

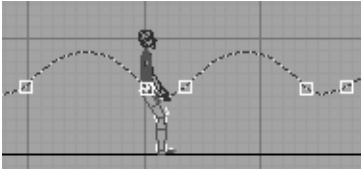
## How Dynamics and Footsteps Relate

When footsteps are created, a footstep track displays in Track View. Footsteps appear as blue and green blocks, laid out to show their exact placement in time. If you examine the center of mass tracks with footsteps, you will notice that frames where leading and trailing edges of the footsteps occur also have keys for the center of mass Body Vertical and Horizontal tracks. These keys contain dynamics information used by **character studio** to calculate airborne body position relative to gravity, and leg bend on landing and balance. Biped Dynamics is the reason you do not need a vertical center of mass key at the top of a jumping motion or at the bottom of dip when the biped lands from an airborne period.

Click and drag the middle of a footstep to move it in time. Click and drag one edge of a footstep to stretch the footstep in time.

Note: Changing the duration of footsteps or moving them relative to one another may change the 'support relationships' of the footsteps. Whenever the support relationships change, Biped generates new keys and deletes any existing leg keys in the airborne period between the edited footsteps.

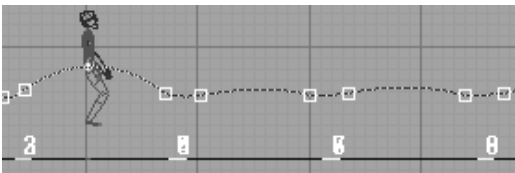




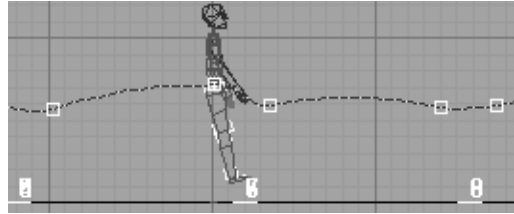
Because of Biped Dynamics, no keys are necessary for the highest part of this jumping motion or for the dip when the biped lands; character studio calculates the trajectory of the Body. This image shows the center of mass trajectory. Keyframes are white squares on the trajectory.



Simply change gravity (GravAccel on the Animation Properties rollout), and the jumping motion is flattened. The biped looks like it's hopping on the moon.



By selecting half the footsteps in Track View and moving them to the right in time, the biped has to jump higher to account for the added time to the next footstep. Notice the yellow dots, representing frames, are tightly bunched together; there are more frames in this airborne period.



Here, the center of mass is moved in the Z plane. Now the biped heel never hits the ground; the biped appears to do a little jump using just his toes. Here, character studio understands that you want to override the calculated trajectory and position the keyframe yourself.

## Freeform Animation

It is left to you to create all the keys in a freeform animation; Biped Dynamics is not active and does not recalculate body position. Balance Factor is active in a freeform animation. A completely freeform animation contains no footsteps.

To start an entirely freeform animation, simply create a biped and begin keyframing. A dialog warns that you are starting a freeform animation, and the Footsteps Mode button on the General rollout grays out, indicating that you cannot create footsteps.

You will often want a freeform period in a footstep sequence; for a walk then fall type of motion for example. In cases like this, a freeform period is specified between footsteps in Track View using the Footstep Mode dialog. A combination of footsteps and freeform is often required when motion capture data is imported. A freeform period is created using controls in the Footstep Mode dialog and display as a yellow boxes between footsteps in Track View.

Right-click the footstep track to display the *Footstep Mode dialog* (see page 267); here you specify a freeform period, select multiple

footstep edges, and set footstep numbering display options.

## Procedures

### To copy keys in the left arm track to the right arm track

1. Click Edit Time on the Track View toolbar.
2. Click the left Clavicle Transform track in the Track View Hierarchy window.
3. Click and drag on the Clavicle track to select time in the Track View Edit window.
4. Click Copy Time on the Track View toolbar.
5. Select the right clavicle transform track.
6. Click Paste Time. A dialog gives you a choice between an Absolute and Relative paste. An Absolute paste will paste over the same frames; a relative paste will paste starting where you clicked in the track before using the Paste Time command.

Note: Keys cannot exist past the end of footsteps. Copy and paste a shorter period of time if a warning displays.

### To select a footstep

- Click the center of a footstep.  
The footstep's border turns white to indicate it's selected.

### To select multiple footsteps, do one of the following

- Click to select a footstep, then use CTRL+click near the center of other footsteps to add them to the selection set.
- Click and Drag a window over footsteps in the footstep track.  
Dragging creates a selection box that selects each footstep it touches.

### To move footsteps in the time

- Drag selected footsteps forward or backward.

### To change vertical dynamics (gravity) for multiple center of mass (COM) keys in Track View

1. Open Track View and select all the vertical COM keys (or any combination that you want to change).
2. Right-click over one of the selected keys to display the TCB dialog.  
Change the value of Dynamics Blend in the TCB dialog. This changes it for all selected keys.

Note: This will only affect the biped during airborne periods in a footstep animation. Changing Dynamics Blend for all center of mass keys in a walk motion will not affect the walking motion.

### To change the duration of a footstep

1. Click the left or right edge of a footstep.  
In addition to a white border, a small white key appears to indicate the border is adjustable.
2. Drag to make the footstep longer or shorter. You can adjust a selection of multiple footsteps this way.

### To clone biped keys in Track View

- Select a key or group of keys and Shift-drag them to the left or right.  
When you release the mouse, you will have created new cloned keys that are exact duplicates of the originals.  
Note: You can drop a cloned key over any existing key to replace it. If the key is locked, it remains locked, and the X,Y,Z value is updated.

## Footstep Mode Dialog

Right-click anywhere in the Footstep track in Track View to display the Footstep Mode dialog. This is a modeless dialog and can remain visible while you continue to work in Track View.

This dialog allows you to:

- Control the duration information displayed with the footsteps.
- Create a freeform period between footsteps.
- Select the right or left edges of footstep blocks or the whole block.

### Procedures

#### To create a freeform period

1. In Track View, right-click the footstep track to display the Footstep Mode dialog.
2. Choose Edit Free Form.

In the footstep track, the footsteps become narrow, and most numerical information for the footsteps is hidden. Outlines of yellow boxes appear between footsteps in the airborne periods to indicate potential freeform areas.

3. Click a yellow box.

The box turns to a solid yellow color, showing that freeform editing is in effect in that area. Click the box again, and the box becomes hollow, showing that dynamics are again active in that area.

At the moment when you set a freeform area, the vertical keys are maintained but the interpolation becomes spline interpolation. At the moment when you unset a freeform area, the vertical keys are recomputed based on vertical dynamics.

**Tip:** If you want to start working on a freeform area using the vertical positions computed with dynamics, then set some

vertical keys to record some vertical positions before making the area a freeform area.

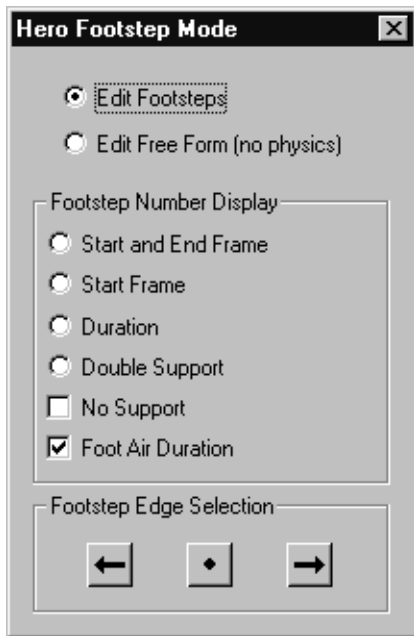
Note: Freeform periods during a footstep animation can occur only between footsteps. Biped does not allow you to begin or end an animation with the biped in midair. To create an animation that begins with the biped in mid-air, start the biped on a footstep, but begin rendering from a frame in the mid-air period.

#### To create a purely freeform animation in Biped

1. Create a biped.
2. Move a biped limb.
3. Answer Yes to the message asking if you want to create a purely freeform animation (without footsteps).

Note: Once you initiate a freeform animation, you cannot add footsteps to it in Footstep mode. You can, however, convert your freeform animation to a footstep animation using Convert on the General rollout.

## Interface



**Edit Footsteps.** In this mode (which is the default), you can edit the biped's footsteps to change their duration; start and end frames; airborne duration; and so on.

**Edit Free Form (no physics).** In this mode, you can edit the biped's body keys for the frames at which the biped is airborne.

Free Form suspends the physically based dynamics that normally control biped motion. This is essential when you want to make the biped fly, or sit down, or fall over. Note that the freeform period must be between footsteps.



The freeform area appears as a yellow block between the footsteps.

## Footstep Number Display group

The footstep blocks can have any one of four time settings displayed for them. You can only show one of these at a time.

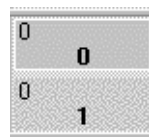
The footstep step number is always displayed on the footstep blocks (in boldface).

**Start and End Frame.** Displays the start and end frames of the footstep (from Touch to Lift). Biped displays just the start frame if the footstep block is too small to show both. Zoom in to see both start and end numbers.



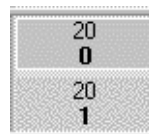
Both footsteps 0 and 1 begin at frame 0 and end at frame 19.

**Start Frame.** Displays just the start frame number (Touch).



Both footsteps 0 and 1 begin at frame 0.

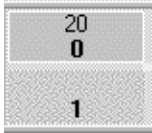
**Duration.** Displays the number of frames that the foot is in contact with the ground (from Touch to Lift).



Both footsteps 0 and 1 are in contact with the ground for 20 frames.

**Double Support.** Displays the number of overlapping frames in which both feet are in contact with the ground.

You can also turn on the following two numbers for the intervals between the footsteps. You can display both numbers at the same time by selecting both boxes.



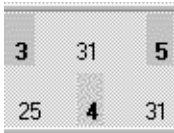
Footsteps 0 and 1 share a double-support period of 20 frames.

**No Support.** Displays the number of frames that the whole biped is airborne; that is, the frames in which neither of the feet have any contact with the ground.



The no-support period between frames 3 and 4 and 4 and 5 is 12 frames.

**Foot Air Duration.** Displays the number of frames that each foot has no contact with the ground.



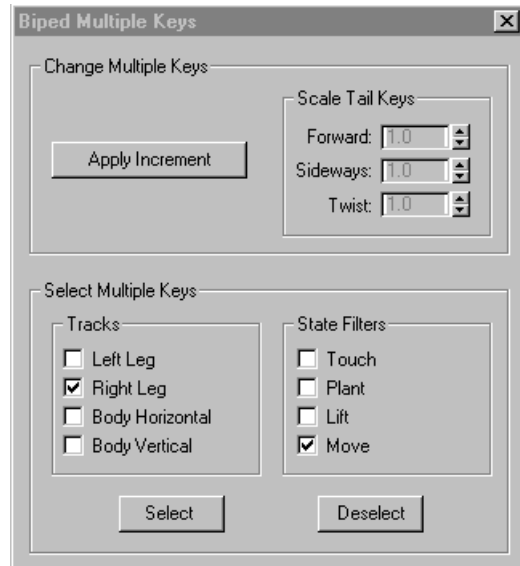
The foot air duration for the left footstep track between frames 3 and 5 is 31 frames.

## Set Multiple Keys Dialog

Select the Biped > Motion panel > Keyframing rollout > Set Multiple Keys > Set Multiple Keys dialog

Keys can be selected manually in Track View or the Track Bar and an increment applied to the selected keys. State Filters in this dialog select certain biped keys for you, based on foot states (Touch, Plant, Lift, and Move). Select Left Leg and the Move state filter, then click Select to select all the left leg keys in a Move state for example (Move is the leg state between footsteps).

### Interface



### Change Multiple Keys group

These controls allow you to apply to a set of keys the rotation or IK translation of a limb at the current frame. First select the keys in Track View,

then rotate or move the limb, then click Apply Increment.

**Apply Increment.** Adjusts the rotation and/or position of a limb at the selected keys. Use this feature when you need to set the position of a limb over multiple keys.

**Scale Tail Keys.** Exaggerates or tones down the default motion applied to the biped's tail for three types of rotation. To activate all three spinners, select tail keys in the biped Tail track in Track View.

**Forward.** Sets the amount of forward and backward swing in the tail.

**Sideways.** Sets the amount of side-to-side swing of the tail.

**Twist.** Sets the amount of local X axis rotation of each tail object.

### Select Multiple Keys group

These controls allow you to select keys according to the foot state at that frame. This is very helpful when you want to apply an increment to all keys of a particular type in a particular track.

- First, select the tracks you want: Left Leg, Right Leg, Body Horizontal, Body Vertical.
- Then select the foot states: Touch, Plant, Lift, Move
- Then click Select to select all the matching keys. Selected keys are highlighted in white in Track View.



## Synthesis Dialog

Open Track View > Global Tracks > Block Control > GlobalClip Properties > Synthesis dialog

Select a Crowd object. > Modify panel > Global Clip Controllers rollout > New > Choose a GlobalClip object. > Select the object in the list. > Edit > Synthesis dialog

The Synthesis dialog has three tabs to split up the workflow. On the Motion Clips tab you specify which object is the global object that contains all the animation; you also setup clips derived from the global object here. Controls on the State tab allow you to link clips to a state, if the bird is accelerating use a fast flap clip for example. Controls on the Synthesis tab allow you to select the objects to synthesize and then synthesize the motion for all of them.

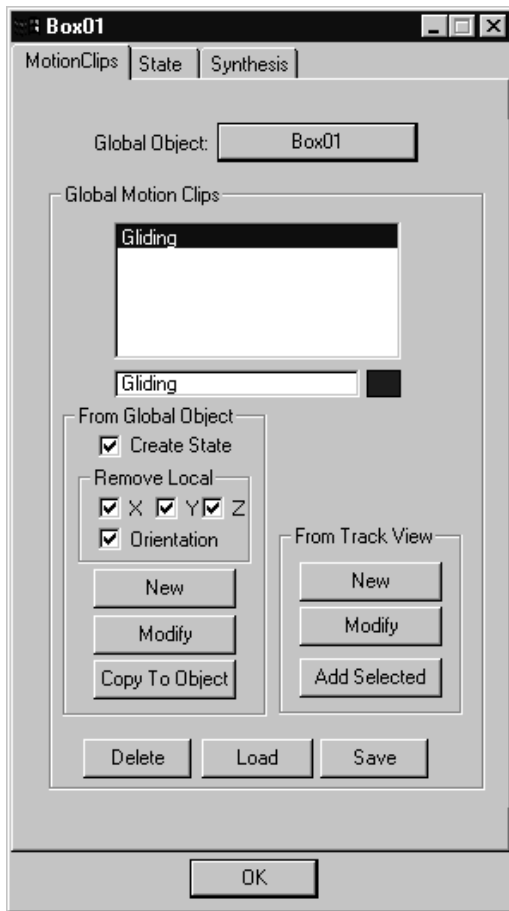
### See also

*Clip State Dialog (see page 275)*

### Procedure

Usage of the Synthesis dialog is closely associated with usage of the ClipState dialog. For a procedure that combines both, see *To use the Synthesis and ClipState dialogs (see page 275)*.

## Interface



### Motion Clips tab

**Global Object.** Click Global Object and pick the object that contains the animation (all the clips) in the Select Global Objects to Copy dialog.

### Global Motion Clips group

**List Window.** The list of motion clips.

The clips that you create appear in this list.

**Edit Window.** Rename or change the color for the selected motion clip.

### From Global Object group

**Create State.** Create a new state with parameters specific to the motion clip like speed, heading, acceleration and so on.

The software evaluates the motion and orientation of the object and creates a new state with parameters set accordingly.

**Remove Local X, Y, Z, Orientation.** Turn options on to strip out transformation or orientation data from the motion clip.

The idea is to animate the global object with full lateral motion. The software then creates states based on the actual motion of the global object. After the states are created the animation is stripped from the object. When delegates linked to clones of the global object come close to the actual motion recorded in the state then the appropriate state is used to trigger the motion clip.

This technique is used to minimize sliding feet. If you animate a creature with many legs you should animate lateral motion as well a leg motion. Then you create clips that record and then strip out the lateral motion that you created. When the delegate approaches the speed and direction you created originally on the global object then the leg motion clip will be activated by the state; leg motion will be accurate relative to speed preventing "sliding feet".

**New.** Lets you setup a new clip. Displays the *MotionClip Parameters dialog* (see page 283) where you can specify the name and duration of the clip.

This selects all active controllers.

**Modify.** Lets you modify a clip in the list. Displays the *MotionClip Parameters dialog* (see page 283) where you can specify the name and duration of the clip.

**Copy to Object.** The selected motionclip keys will be copied back to the GlobalObject. Deletes the animation keys that are there already.

This allows you to edit a motion clip. Select a motionclip, click CopyToObject, change it on the global object, and then click Modify and this will replace the global object animation. The change will affect all the objects being synthesized.

### From Track View group

**New.** Lets you setup a new clip. Specify a track in the *Track View Pick dialog* (see page 283).

The *MotionClip Parameters dialog* (see page 283) displays upon exiting the Track View Pick dialog.

**Modify.** Lets you modify a clip in the list. Specify a track in the *Track View Pick dialog* (see page 283).

The *MotionClip Parameters dialog* (see page 283) displays upon exiting the Track View Pick dialog.

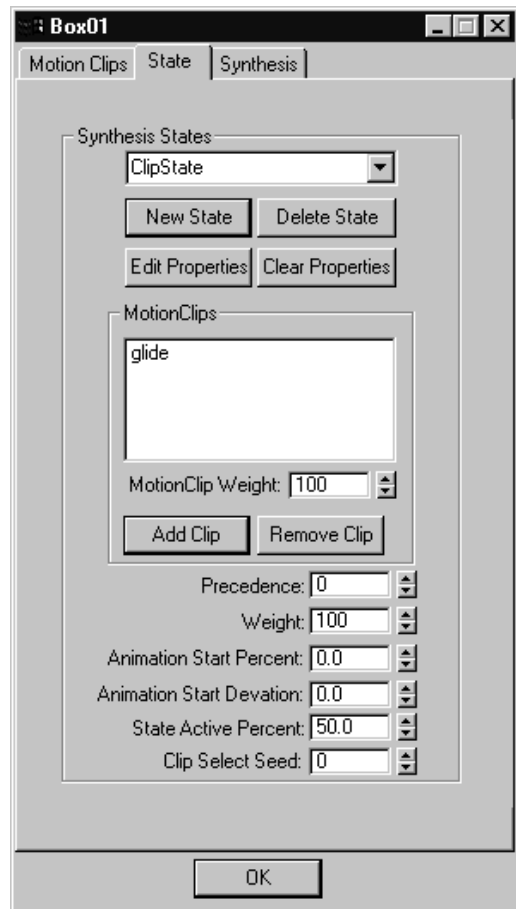
**Add Selected.** Creates a motion clip from only the selected tracks in track view.

**Delete.** Deletes a clip. Select a clip and click Delete.

**Load.** Loads a motionclip file (.clp). Displays the Load Motion Clip dialog.

**Save.** Saves a motionclip into a .clp file. Displays the Save Motion Clip dialog.

## State Tab



Create and define states to determine how the synthesis will occur.

**State Drop Down List.** Displays the current state. Select a state to modify. You can change the state name by editing the text in the window.

**New State.** Displays the *ClipState dialog* (see page 275). Creates a new state.

Choose clip state parameters in the ClipState dialog.

**Delete State.** Deletes the selected state.



**Edit Properties.** Lets you modify the current state. Displays the *ClipState dialog* (see page 275) for the selected state.

**Clear Properties.** Modifies the state back to its default and removes clips from the clip window.

**MotionClips Window.** Displays clips for the current state. Use Add to select which clip or clips should be associated with the current state. If more than one clip is displayed, then the clips are picked randomly during synthesis.

**MotionClip Weight.** Increases the random probability that a clip will be chosen. Range 0-1000.

A higher weight means that a clip will have a better chance of randomly being chosen.

**Add Clip.** Displays the Select MotionClip dialog. Select a clip and click OK to add a clip to the current state.

If there is more than one motionclip for a state, the state will randomly select which one to use. The user can influence this random selection by modifying the Weight parameter.

**Remove Clip.** Removes the selected clip from the current state.

**Precedence.** Sets the precedence for the selected state. Range 0-1000.

If more than one state is active, the one with the higher precedence is played first. If more than one state has the same precedence, then the one with the greater weight is played first.

**Weight.** Specifies a weight value for a state.

If two states have the same precedence, the state with a greater weight will be considered more heavily during random selection.

**Animation Start Percent.** Specifies how far into the animation you want it to have played when the state is active.

For the default value of 0 the animation will start once the state is active. If the value is 100 the animation will be done playing once the state becomes active. You can also randomly modify where the animation starts by specifying a nonzero Animation Start Deviation value.

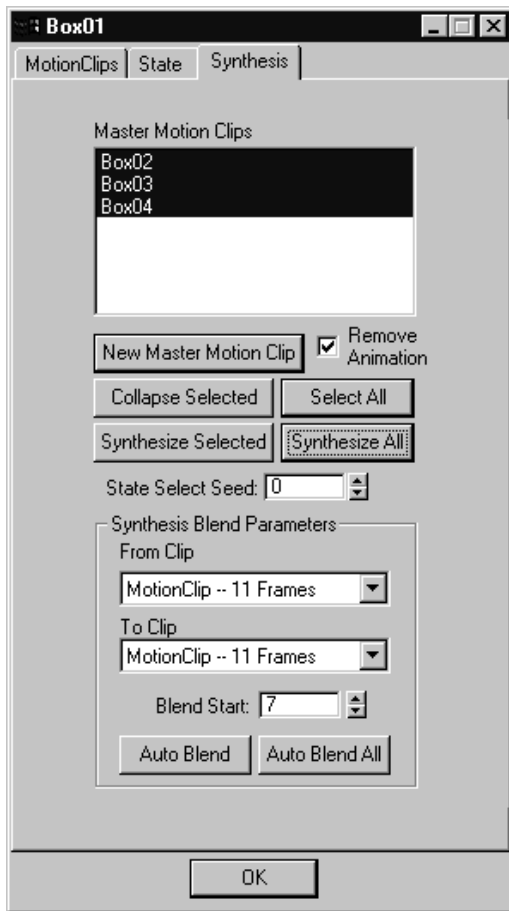
**Animation Start Deviation.** Randomly modifies where the animation starts by specifying a nonzero Animation Start Deviation value. Range 0.0-1.0.

**State Active Percent.** Specifies the percentage of time the state needs to be valid over its interval in order for it to be selected. Range 0-100, Default=50.

**Clip Select Seed.** Changes how the random selections occur.

If the value stays the same, you are guaranteed to get the same random selections for that state.

## Synthesis tab



Controls in the Synthesis tab are for adding objects to be synthesized, selecting blend transition points and performing the synthesis.

**New Master Motion Clip.** Displays the Select Objects to Copy dialog. Lets you select objects onto which the motion will be copied to.

**Remove Animation.** Strip the animation from the clones.

After making clones of the original animated object you can strip the animation from the

clones. During synthesis motion is applied based on which state is active.

**Collapse Selected.** Collapses keys on selected objects.

Keys are collapsed to the objects in the scene. This deletes the MasterMotionClip for that object. You can then edit keys and make changes to the animation manually in the scene.

**Synthesize Selected.** Creates keys for the selected objects.

**Select All.** Selects all of the objects in the list.

**Synthesize All.** Synthesizes the motion for all the objects. Depending on the active state appropriate motion is applied to the clones of the original object.

A progress bar displays during synthesis.

**State Select Seed.** Set a seed for random state selection.

During synthesis more than one state may become active at the same time, in which case one state is randomly chosen. The seed is used to modify the random value selected when determining which state to select.

### Synthesis Blend Parameters group

Blending between clips can be either user specified or algorithmic. Drop down lists are used to specify a current clip and the next clip to be blended to. You can specify, in frames, when the next clip should be blended. Or you can toggle Auto Blend All to have the system determine the best blend point.

**From Clip List.** Lets you select the From clip.

**To Clip List.** Lets you select the To clip.

**Blend Start.** Sets the frame to start a transition manually.

**Auto Blend.** Creates automatic transitions between current clips.

**Auto Blend All.** Creates automatic transitions between all clips.

## ClipState Dialog

Track View > Global Tracks > Block Control > GlobalClip Properties > Synthesis dialog > State Tab > New State > Edit Properties > ClipState dialog

Select a Crowd helper > Modify panel > Global Clip Controllers rollout > New > Choose GlobalClip object. > Select object in list. > Edit > Synthesis dialog > State Tab > New State > Edit Properties > Clip State dialog

Use the ClipState dialog to associate states with clips. A clip can be selected to play based various aspects of a delegate's motion: speed, acceleration, pitch, pitch velocity, and/or heading velocity. You can also define a state with a script. You can turn each state aspect on and off individually. For example, you can have speed and pitch turned on to include these parameters when the motion is synthesized. These parameters all depend upon motion trajectory information and specify whether or not a state will be active.

For each state's active aspect(s), you specify a range of values between which a state will be active. Alternatively, you can specify a single, unique motion value for when a state is active. In addition, you specify the in and out values of the parameter as it approaches and then passes through that unique value. These values are analogous to the tangents of a curve. You can pick Anything, Decreasing, Increasing, or Constant. For example, a landing animation

can have a Unique speed value of 0.0, a Decreasing in value and an Anything out value. Taking off, on the other hand, would have an Increasing out value.

If no state parameters are used, the state is a default state. For example, you have clips to be randomly chosen for an object, and you don't care about what the speed or pitch is. You can create default states and the synthesis engine will randomly pick which clip is active.

Note: "ClipState" is the default name of this dialog, because it's the default name of the first state created in the Synthesis dialog > State tab. It's the dialog invoked by clicking the Edit Properties button. If you rename the state, the dialog assumes the changed name.

## Automatic State Creation

A state can be created automatically if Create State is turned on in the Motion Clips tab of the Synthesis dialog. This is used if you created a creature that contains lateral motion as well as looping motion. If all the options are turned on in the Remove Local group of the MotionClips tab then a state is created that reflects the actual heading, speed and acceleration of your creature. When a delegate approaches the heading and speed contained in this state it triggers the appropriate motionclip. This method prevents sliding feet in a multi legged creature animation.

## See also


*Synthesis Dialog (see page 270)*

## Procedure

### To use the Synthesis and ClipState dialogs

This procedure assumes that the global object is static and has animation that loops. For

creatures with many legs you can animate lateral motion on the global object and then strip out the lateral motion in the synthesis dialog. This latter approach is to minimize foot sliding in a multi-legged creature.

1. Animate an object.  
Create animation in one position, like a bird's beating wings. Create a variety of animation like a gliding motion, wings beating slowly, and so on.
2. Use Shift-Move to create as many clones of the animated object as necessary.
3. In Create > Helpers > Object Type rollout use Crowd and Delegate to create a crowd object and a delegate object.  
Create the crowd and delegate objects in the top viewport.
4. Use Crowd's *Scatter Objects* (see page 362) to clone the delegate and distribute the clones.  
Make sure you end up with an equal number of delegates and animated object clones.  
Next, associate and link the objects to the delegates.
5.  On the Setup rollout, click the Object/Delegate Associations button.  
The *Associate Bipeds With Delegates* dialog (see page 370) appears.
6. Add the objects and delegates into their respective columns.
7. Click Align objects with Delegates and Link Objects with Delegates, and then click OK to exit the dialog.  
The objects align themselves with the delegates and are linked to the delegates.  
Next, animate the delegates using a behavior. See the *Crowd documentation* (see

page 341) to animate the delegates using behaviors.

At this point, when you solve the simulation, the objects follow the delegates that are guided by behaviors.

8. Select the Crowd object. On the Modify panel > Global Clip Controllers rollout click New, and use the Select dialog to select the original animated object.  
The object appears in the Global Clip Controllers rollout list.
9. In the list, click the original object and then click the Edit button.  
The Synthesis dialog displays.
10. On the Motion Clips tab turn off Create State, Remove Local XYZ and Orientation.  
Use these options only if your original object has lateral motion to coordinate with footsteps.
11. On the Motion Clips tab click New.
12. Choose a descriptive name and a frame range for the motion clip.  
At this point you need to specify a frame range for a part of the animation. For example, frames 0 through 10 might be your glide animation.
13. Continue to create new clips using different frame ranges. Give the clips descriptive names.
14. Open the State tab and click New State. Give it a descriptive name. In many cases, the state can use the same name as the motion clip that's to be associated with it.  
At this stage you should have two or three motion clips. Now you need to associate a clip with this state. For example, your glide motion should be active only if acceleration is less than 0.

15. Click Add Clip and select a clip in the Select MotionClip dialog.
16. Click Edit Properties and activate the clip based on any combination of speed, acceleration and so on. Open the appropriate tab(s), turn on its Use ... check box, and set appropriate parameters. Click Exit to exit the dialog.
17. On the Synthesis tab click New Master Motion Clip and add all of the cloned objects.
18. Click Auto Blend All and then click Synthesize All. Exit the dialog.
19. Click Play. It's not necessary to re-solve the simulation.

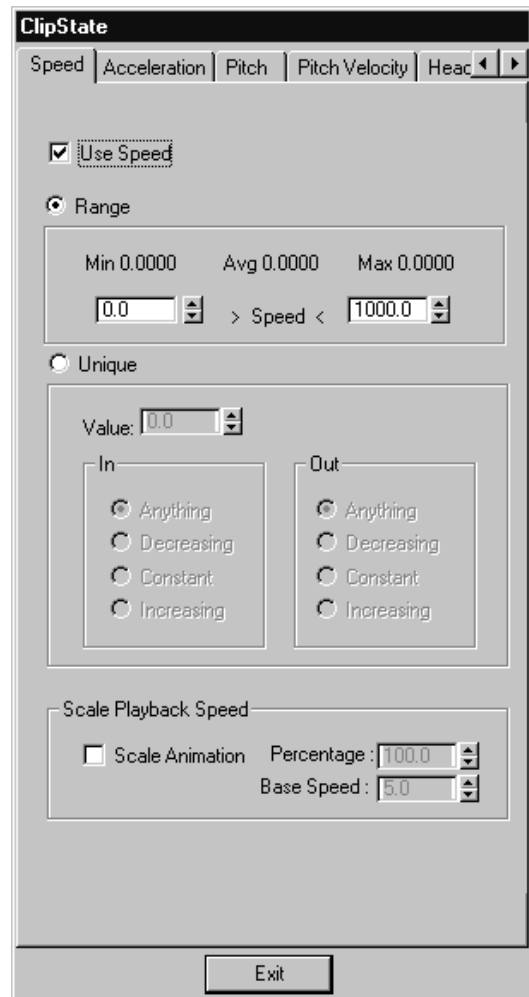
The objects follow the delegates and are animated using clips that are activated according to the states you created.

For a tutorial covering this procedure, see *Using Crowd with Animated Non-Biped Objects* (see page 567).

## Interface

Make sure Min settings are smaller than Max settings. For example, when using a negative range (-180 to -10), enter the number with the larger absolute value (-180) as the Min setting.

## Speed



**Use Speed.** Use speed to make a clip active.

**Range.** Use a speed range to make a clip active.

**Range Display.** Displays minimum, average, and maximum clip speed.

**Min.** Set a minimum speed value.

**Max.** Set a maximum speed value.

**Unique.** Use a unique speed to make a clip active.

**Value.** Set a unique speed value.

### In/Out

These conditions specify what the value of the parameter is before and after the unique value is met. It can be Increasing, Decreasing, Constant or Anything -- in which case it can be either of the first three. This allows you to specify under what conditions that value is occurring. For example, say the unique value is a speed of 0, the In value is Decreasing and the Out value is Constant. This means that the object has decreased its speed to zero and is now stationary, a perfect state to trigger a landing animation (clip).

**Anything.** Speed into or out of the target value is not important.

**Decreasing.** Speed into or out of the target value is decreasing.

**Constant.** Speed into or out of the target value is constant.

**Increasing.** Speed into or out of the target value is increasing.

### Scale Playback Speed group

**Scale Animation.** Scale the clips animation based on speed.

As a bird flies faster its wings beat faster for example. Scaling an animation scales the keys of the animation. The closer the keys are to each other the faster the animation plays.

**Percentage.** Specify how much to alter the playback speed based upon the difference from the Base Speed.

For example, if an object is moving 50% faster than it's base speed, and the Scale Percentage value is 50%, then the playback speed is scaled down by 25%.

**Base Speed.** Specify at what speed should the animation be played back at its normal rate.

**Exit.** Exit the ClipState dialog.

### Acceleration

The screenshot shows the 'ClipState' dialog box with the 'Acceleration' tab selected. The 'Use Acceleration' checkbox is checked. Under the 'Range' radio button, there are three input fields: 'Min' (0.0000), 'Avg' (0.0000), and 'Max' (0.0000). Below these is a range selector with '0.0' and '1000.0' values and a '> Rate <' label. The 'Unique' radio button is unselected. Under 'Unique', there is a 'Value' field set to '0.0'. Below this are two columns of radio buttons for 'In' and 'Out' conditions, each with four options: 'Anything', 'Decreasing', 'Constant', and 'Increasing'. At the bottom, there is a 'Scale Playback Speed' section with a 'Scale Animation' checkbox (unchecked), a 'Percentage' field (100.0), and a 'Base Acceleration' field (0.0). An 'Exit' button is at the very bottom.

**Use Acceleration.** Use Acceleration to make a clip active.

**Range Display.** Displays minimum, average, and maximum acceleration.

**Range.** Use an acceleration range to make a clip active

**Min.** Set a minimum acceleration value.

**Max.** Set a maximum acceleration value.

**Unique.** Use a unique acceleration to make a clip active.

**Value.** Set a unique acceleration value.

### In/Out

**Anything.** Acceleration into or out of the target value is not important.

**Decreasing.** Acceleration into or out of the target value is decreasing.

**Constant.** Acceleration into or out of the target value is constant.

**Increasing.** Acceleration into or out of the target value is increasing.

### Scale Playback Speed Group

**Scale Animation.** Scale the clips animation based on acceleration.

As a bird accelerates its wings beat faster for example.

**Percentage.** Specify how much to alter the playback speed based upon the difference from the Base Acceleration.

**Base Acceleration.** Specify at what acceleration should the animation be played back at its normal rate.

### Pitch

The screenshot shows the 'ClipState' dialog box with the 'Pitch' tab selected. The dialog has a title bar 'ClipState' and a tab bar with 'Speed', 'Acceleration', 'Pitch', 'Pitch Velocity', and 'Head'. The 'Pitch' tab contains the following controls:

- ☐ Use Pitch
- ☒ Range
  - Min 0.0000 Avg 0.0000 Max 0.0000
  - Min: 180.0 (spin box) > Pitch < Max: 180.0 (spin box)
- ☐ Unique
  - Value: 0.0 (spin box)
  - In:
    - ☒ Anything
    - ☐ Decreasing
    - ☐ Constant
    - ☐ Increasing
  - Out:
    - ☒ Anything
    - ☐ Decreasing
    - ☐ Constant
    - ☐ Increasing
- Scale Pitch Orientation
  - ☐ Scale Pitch
    - Percentage: 100.0 (spin box)
    - Base Pitch: 0.0 (spin box)
- Scale Playback Speed
  - ☐ Scale Animation
    - Percentage: 100.0 (spin box)
    - Base Pitch: 0.0 (spin box)
- Exit button

**Use Pitch.** Use pitch to make a clip active.

**Range Display.** Displays the minimum, average, and maximum pitch of the clip.

**Range.** Use a pitch range to make a clip active.

**Min.** Set a minimum pitch.

**Max.** Set a maximum pitch.

**Unique.** Use a unique pitch value to activate a clip.

**Value.** Set a unique pitch value

### In/Out

**Anything.** Pitch into or out of the target value is not important.

**Decreasing.** Pitch into or out of the target value is decreasing.

**Constant.** Pitch into or out of the target value is constant.

**Increasing.** Pitch into or out of the target value is increasing.

### Scale Pitch Orientation

The Scale Pitch parameter lets you scale the pitch the animation is played at depending upon the pitch of the object. In the default case when the Scale Pitch is off, the object will pitch normally. If it is turned on you can select a Base Pitch for a pitch for which the animation should play at, plus a percentage to specify how much the BasePitch is modified by the true pitch of the objects direction. If the Percentage is 100 then the BasePitch will be active; anything else will scale between the BasePitch pitch and the true pitch. This is useful for when birds and other objects elevate, but don't pitch up.

**Scale Pitch.** Scale the clips pitch based on pitch.

**Percentage.** Specify how much to alter the pitch based upon the difference from the Base pitch.

**Base Pitch.** Specify a base pitch.

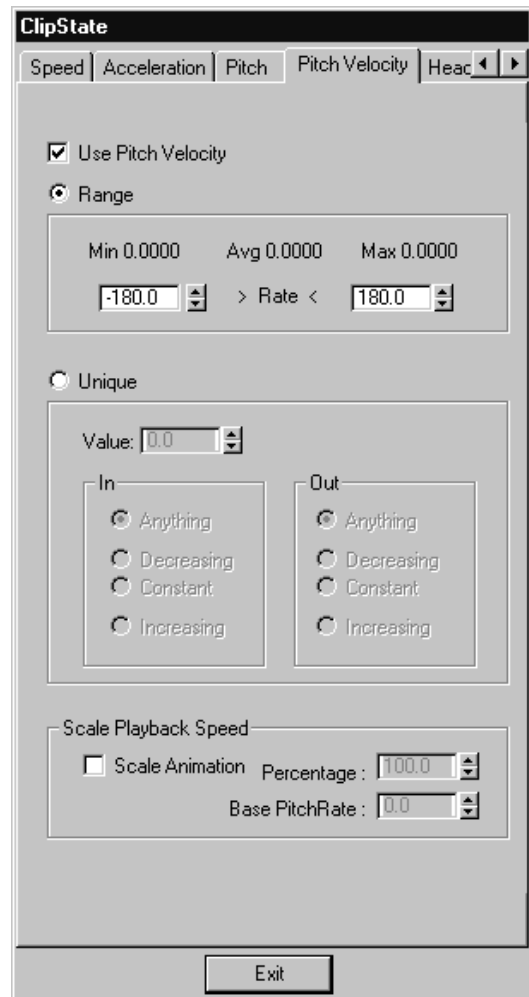
### Scale Playback Speed

**Scale Animation.** Scale the animation based on pitch.

**Percentage.** Specify a percentage to scale speed.

**Base Pitch.** Specify at what pitch should the animation be played back at its normal rate.

## Pitch Velocity



**Use Pitch Velocity.** Use pitch rate to make a clip active.

**Range Display.** Displays the minimum, average, and maximum pitch rate of the clip.

**Range.** Use pitch rate range to make a clip active

**Min.** Set a minimum pitch rate.

**Max.** Set a maximum pitch rate.



**Unique.** Use a unique pitch rate value to activate a clip.

**Value.** Set a unique pitch rate value.

### In/Out

**Anything.** Pitch rate into or out of the target value is not important.

**Decreasing.** Pitch rate into or out of the target value is decreasing.

**Constant.** Pitch rate into or out of the target value is constant.

**Increasing.** Pitch rate into or out of the target value is increasing.

### Scale Playback Speed

**Scale Animation.** Scale the animation based on pitch rate.

**Percentage.** Enter a percentage of pitch rate to scale speed.

**BasePitchRate.** Specify at what pitch rate should the animation be played back at its normal rate.

## Heading Velocity

The screenshot shows the 'ClipState' dialog box with the 'Heading Velocity' tab selected. The dialog has four tabs: 'Pitch', 'Pitch Velocity', 'Heading Velocity', and 'Script'. The 'Heading Velocity' tab is active. It contains the following controls:

- ☐ Use Heading Velocity
- ☒ Range
  - Min 0.0000 Avg 0.0000 Max 0.0000
  - Range: [-180.0] > Rate < [180.0]
- ☐ Unique
  - Value: [0.0]
  - In:
    - ☒ Anything
    - ☐ Decreasing
    - ☐ Constant
    - ☐ Increasing
  - Out:
    - ☒ Anything
    - ☐ Decreasing
    - ☐ Constant
    - ☐ Increasing
- Scale Playback Speed
  - ☐ Scale Animation Percentage: [100.0]
  - Base HeadingRate: [0.0]

An 'Exit' button is located at the bottom right of the dialog.

**Use Heading Velocity.** Use heading rate to activate a clip.

**Range Display.** Displays the minimum, average, and maximum heading rate of the clip.

**Range.** Use a heading rate range to activate a clip.

**Min.** Set a minimum heading rate value.

**Max.** Set a maximum heading rate value.

**Unique.** Use a unique heading rate value to activate the clip.

**Value.** Set a unique heading rate value.

#### In/Out

**Anything.** Heading rate into or out of the target value is not important.

**Decreasing.** Heading rate into or out of the target value is decreasing.

**Constant.** Heading rate into or out of the target value is constant.

**Increasing.** Heading rate into or out of the target value is increasing.

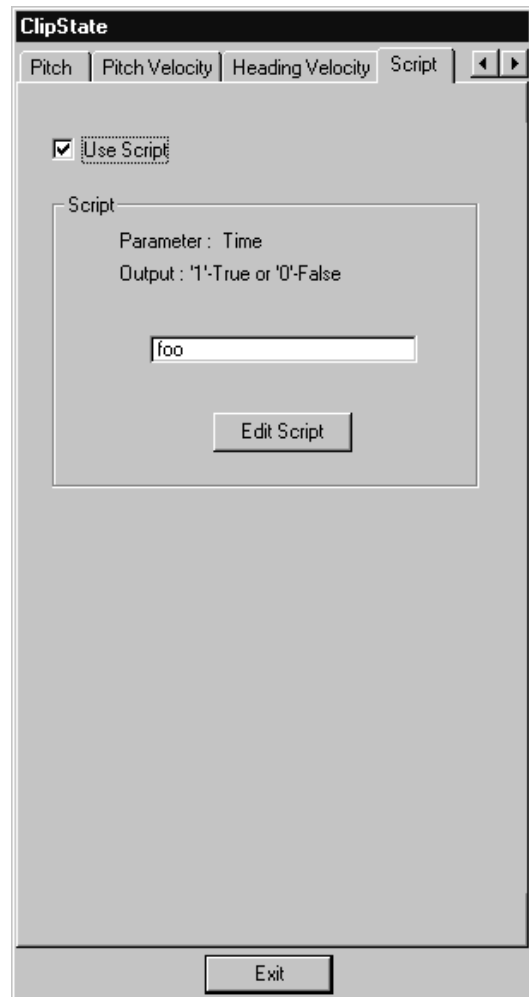
#### Scale Playback Speed

**Scale Animation.** Scale the animation based on heading rate.

**Percentage.** Enter a percentage of heading rate to scale speed.

**BaseHeadingRate.** Specify at what heading rate should the animation be played back at its normal rate.

## Script



Allows you to create a MAXScript script that takes two parameters: node and time. The script typically tests one or more values, and then returns 1 to indicate that the state is active or 0 if it's inactive. Based upon the evaluation of this MAXScript function, the State is then determined to be active or not.

Scripts used by the clip controller are similar to those used by the *cognitive controller* (see page 382), with the exception that a special time-related statement is required.

In the following sample script, “del” is the delegate’s node, and “t” is the time. The name of the scripted function, “stoppedScript,” would also need to be entered into the name field in the Script panel of the ClipState dialog. Unlike cognitive controller scripts, the statement “at time t” needs to be invoked because the animation is not running when the synthesis takes place.

```
fn stoppedScript del t = (
  at time t
  if del.pos.z < 65 then 1 else 0
)
```

**Use Script.** Turn this on to use a MAXScript script to control a clip.

**Script Window.** Enter the name of the function defined by the script, also found at the start of the script.

**Edit Script.** Edit the script.

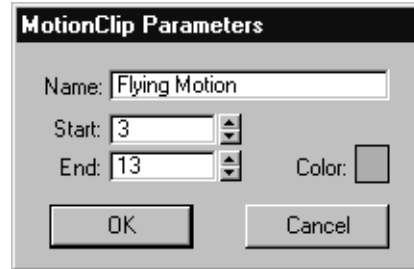
## MotionClip Parameters Dialog

Open Track View. > Global Tracks > Block Control > GlobalClip Properties > Synthesis dialog > Motion Clips tab > Click New (in the From Global Object group). > Motion Clip Parameters dialog

Select a Crowd helper. > Modify panel > Global Clip Controllers rollout > New > Choose a GlobalClip object. > Select the object in the list. > Edit > Synthesis dialog > Motion Clips tab > Click New (in the From Global Object group). > Motion Clip Parameters dialog

Enter a clip name, frame range, and color using controls in the MotionClip Parameters dialog. Clips are triggered to play by states that you define in the State tab

## Interface



**Name.** Type a name for the new motion clip.

**Start.** Enter the clip start frame.

**End.** Enter the clip end frame

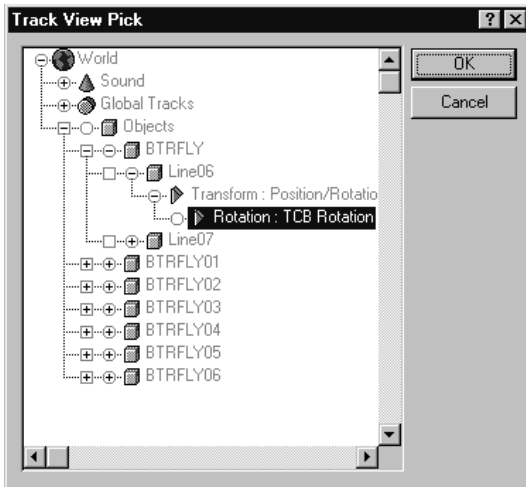
**Color.** Click on the color swatch and choose a color in the color picker

## Track View Pick Dialog

Open Track View > Global Tracks > Block Control > GlobalClip Properties > Synthesis dialog > Motion Clips tab > Click New (in the From Track View group). > Track View Clip dialog

Select a Crowd helper. > Modify panel > Global Clip Controllers rollout > New > Choose a GlobalClip object. > Select the object in the list. > Edit > Synthesis dialog > Motion Clips tab > Click New (in the From Track View group). > Track View Clip dialog

Open the hierarchy and pick a track to use in the motion clip.



## Animations

### Motion Library

**character studio** ships with a library of motions that you can apply to any biped. Some of these motions have been created using Freeform or Footstep methods. Other motions have been created using motion capture.

Since these are motion files, not *.max* files, they do not include geometry. Many of these files demonstrate motions that happen with imaginary objects, but the objects are not visible in the scene. You'll have to supply your own, and animate them to match the motions.

Here is a catalog and brief description of some of these motion files.

### Footsteps

File	Description
About face.bip	Biped takes two steps, then does an about face.
ActorCalibration.bip	Biped stands still with arms and legs spread apart.
AthleteStretch.bip	Biped does 3 stretches in one sequence.
AxeKick.bip	Biped runs up, axe kicks with left leg, then returns to position.
BackFlip.bip	Biped does a back flip.
BackKick.bip	Biped does a backward spin kick with right leg.
BackLegKick.bip	Biped does a single high kick with right leg.
BackWalkLoop.bip	Biped turns its head as it walks backwards.
Ballet.bip	Biped performs ballet techniques.
Banana.bip	Biped walks, then slips, and does a pratfall.
BangThumb.bip	Biped stands and bangs its thumb while hammering a nail.
Barbecue.bip	Biped adds fuel, lights a match, and reacts to explosion.
BarStoolStnd.bip	Biped sitting on a stool, steps off, and walks away.
Bees.bip	Biped walks and swings its arms as if it's swatting bees.
BlownBack.bip	Biped is blown back and onto its back.
Bow.bip	Biped bows twice.
BroadJump.bip	Biped completes a long jump and lands on both feet.
CallDog.bip	Biped whistles, then bends down, and opens its arms.

File	Description
CarryHeavy.bip	Biped picks up, carries, and sets down a heavy suitcase.
CartWheel.bip	Biped does a single cartwheel.
CartWhlPunch.bip	Biped does a single cartwheel and ends in a low position.
Catwalk.bip	Biped walks like a fashion model on the runway.
Chacha.bip	Biped dances the cha-cha.
ChairSit.bip	Biped reaches for chair, sits down, and scoots forward.
ChairStand.bip	Biped stands from sitting position.
Cheer.bip	Biped cheers, waving its hands and clapping.
Cigarette.bip	Biped puts a cigarette to mouth, lights it, takes a puff.
Creep.bip	Biped creeps in a straight line.
Cripple.bip	Biped limps like Igor in "Young Frankenstein"
Cs1dance.bip	Biped dances.
DbIRndKick.bip	Biped completes a double roundhouse kick.
Dodge.bip	Biped dodges to its left, its right and then down.
DogDoo.bip	Biped steps in something, wipes its foot, then walks on.
DoubleTake.bip	Biped walks straight line and looks back twice.
DragByDog.bip	Biped is taken for a walk by a dog on a leash.
Drunk.bip	Biped walks like a drunk, then holds its aching head.
EatDrink.bip	Biped in sitting position eats and has a drink.

File	Description
Exercise.bip	Biped does squats and jumping jacks.
FaceHit.bip	Biped takes a hard hit to its face.
Fishing.bip	Biped casts fishing line and reels in a fish.
FlySpinKick.bip	Biped jumps in the air and does a spin kick.
Fridge.bip	Biped opens the refrigerator, takes something out to drink.
FrntHndSpring.bip	Biped executes a front flip.
FrontSweepKick.bip	Biped Drops down, sweeps its right leg in a circle twice, then returns to standing position.
GetPunched.bip	Biped takes a punch to the left side of its face.
GetSlapped.bip	Biped reacts to a slap across both sides of its face, then holds its face.
GrabReach.bip	Biped steps up, reaches for an object, stands on tiptoes.
Grenade.bip	Biped pulls grenade pin out with its teeth, throws grenade, and then covers itself from the explosion.
GunDive.bip	Biped rolls into a kneeling position with gun drawn.
GunFighter.bip	Biped steps forward, draws, and fires its gun.
HandPlant.bip	Biped does a single handstand with right arm.
HatsOff.bip	Biped looks right and left, then looks at an object in its right hand.
HopScotch.bip	Biped plays hopscotch, hopping one way, then the other.
HulaHoop.bip	Biped picks up hula-hoop and begins to gyrate.

File	Description
Joe run.bip	Biped runs in a straight line, jumps and land on both feet.
JogLoop.bip	Biped runs in a straight line.
JogTurnL.bip	Biped tuns left while running.
JumpOver.bip	Biped jumps over an object and lands with both feet.
JumpRndKick.bip	Biped executes a high kick.
JumpSplit.bip	Biped jumps with arms and legs spread apart.
KickPunch.bip	Biped jumps and kicks in the air.
Kiddie.bip	Biped walks like a toddler.
KOed.bip	Biped takes a punch and falls to the ground.
Laugh.bip	Biped laughs standing in place.
LegHookKick.bip	Biped executes a hook kick with right leg.
Limbo.bip	Biped walks leaning way back, then stands up straight.
LimpLoop.bip	Biped limps in a single step.
LiteLimpLoop.bip	Biped limps in a single step.
MarchLoop.bip	Biped marches in a straight line.
MarchStart.bip	Biped begins to march.
MarchStop.bip	Biped stops marching and stomps right leg.
MarchTurnL.bip	Biped begins to march, then stops and turns left
MarchTurnR.bip	Biped begins to march, then stops and turns right
Military.bip	Biped marches, stops and stomps right leg, then turns left and salutes.

File	Description
MotorCycle.bip	Biped walks over to motor-cycle, raises right leg over the seat, then kick starts it.
MoveHeavy.bip	Biped picks up an object, walks forward as if carrying something heavy, then lays it down.
Muscle.bip	Biped flexes in a few positions.
Plie.bip	Biped walks in a straight line waving its arms and bending its legs, then stops in a pose.
Plie2.bip	Biped walks in a straight line waving its arms and bending its legs, then stops in a pose.
Politician.bip	Biped waves its arms.
Prance.bip	Biped prances in a straight line.
Punch.bip	Biped executes a single punch with its right hand.
Robot.bip	Biped moves like a robot.
RopePull.bip	Biped walks as if using a rope to pull its self up a mountain.
Roundkick.bip	Biped kicks with right leg.
RunLoopC.bip	Biped runs in straight line.
Saw.bip	Biped steps up, plants its left leg, and begins to saw.
ScissorKick.bip	Biped spins and kicks with its right leg.
Shocked.bip	Biped reaches out with its right hand, then shakes as if shocked.
ShootArrow.bip	Biped reaches for an arrow from behind its back, then places the arrow on the bow, and shoots.
SideKick.bip	Biped executes a sidekick with its right leg.

File	Description
SkipLoop.bip	Biped skips in a straight line.
SkipRope.bip	Biped skips rope different ways.
Slap.bip	Biped slaps twice with its right hand.
SledgeHammer.bip	Biped slams down as if using a sledgehammer.
Slip.bip	Biped slips and falls on its back.
Sneeze.bip	Biped sneezes once.
SpillCoffee.bip	Biped is in sitting position, then reacts to hot coffee being spilled on its lap.
SpillDrink.bip	Biped is in sitting position, then reacts to spilling its drink.
SpinBackKick.bip	Biped spins and kicks with its right leg.
SpinFall.bip	Biped walks forward hunched down, then turns around and falls backward.
SpinKickHi.bip	Biped walks back, then executes a high kick.
Sprint.bip	Biped runs in a straight line, jumps and land on both feet.
Stagger.bip	Biped staggers forward.
StraightKick.bip	Biped executes a single with its right leg.
StrutLoop.bip	Biped struts in a straight line.
SwingAxe.bip	Biped swings both arms twice as if holding an axe.
Tantrum.bip	Biped throws a tantrum.
ThruDoor.bip	Biped opens door, walks through, and then shuts door.
TightRope.bip	Biped walks in a straight line with both arms held out.

File	Description
TornadoKick.bip	Biped executes a spin kick with its right leg.
Trip.bip	Biped walks straight and trips forward.
Vault.bip	Biped jumps over an object.
Walk.bip	Biped walks in a straight line.
WalkIntoWall.bip	Biped reacts as if walking into a wall, then holds its nose.
WalkLoop.bip	Biped walks in a straight line.
WalkLoopC.bip	Biped walks in a straight line.
WalkStart.bip	Biped walks in a straight line.
WalkStop.bip	Biped walks then stops.
WalkTurnL.bip	Biped walks in a straight line, then turns to its left.
WalkTurnR.bip	Biped walks in a straight line, then turns to its right.
Wave.bip	Biped waves its right hand.
Whip.bip	Biped snaps its arms as if cracking a whip.
YawnStretch.bip	Biped stretches both arms out.

## Freeform

File	Description
Drive.bip	Biped drives, switches gears with right hand and feet.
SkateBack.bip	Biped slides backwards as if skating backwards.
SkateSpin.bip	Biped slides with arms and legs spread out.
SkateStart.bip	Biped skates forward.
SkateStop.bip	Biped skates forward, then stops in a half circle.
SkateTrnArnd.bip	Biped skates forward, then turns to skate backwards.
Skateup.bip	Biped gets up from the ground.
Unicycle.bip	Biped balances itself as if it were on a unicycle.
UnicycleLoop.bip	Biped balances itself as if it were on a unicycle.
UniOff.bip	Biped jumps off the unicycle.
UniOn.bip	Biped jumps on the unicycle.

## Motion Capture

The Motion capture files provided with the software are designed to be used in motion flow networks for crowd simulations, and for general animation use. The motions were designed to fit together, and represent variations to build a motion flow network. Files that share similar names represent a range of motions meant to be joined in a motion network for use with the crowd system.

To open the motion capture files, use motion capture import. If you want to use them in motion flow networks, save them as BIP files. You can also use just a segment of a captured motion by saving a segment of the file to create a new BIP file.

## Naming Conventions

The files have been named to indicate direction and foot placement of the motion.

Cyclic locomotion has been captured at turning angles of either 45 or 90 degree increments in both the left and right direction. In the case of walking, fast walking and jogging, there are also comprehensive coverage for turning on both the left or the right foot. For example, *walk\_L90\_Rfoot* means walking with a left 90 degree turn, and the turn occurs on the right foot.

## Motion Capture Studios

The BVH files were captures at Modern Uprising Studios ([www.modernuprising.com](http://www.modernuprising.com)). The CSM files were captured at House of Moves ([www.moves.com](http://www.moves.com)).

## Motion Capture Talent

Jodie Melnick performed the female motions at the Modern Uprising Studios. Kim Weild performed the female motions at the House of Moves. Michael Girard performed all the male motions.



## Motion Files

### Modern Uprising, Male Motions

Motion	File
Walking Straight	walk1.bvh
	walk2.bvh
Walking Turns	
Turning Left on Right Foot	walk_L45_Rfoot.bvh
	walk_L90_Rfoot.bvh
	walk_L135_Rfoot.bvh
	walk_L180_Rfoot.bvh
Turning Left on Left Foot	walk_L45_Lfoot.bvh
	walk_L90_Lfoot.bvh
	walk_L135_Lfoot.bvh
	walk_L180_Lfoot.bvh
Turning Right on Right Foot	walk_R45_Rfoot.bvh
	walk_R90_Rfoot.bvh
	walk_R135_Rfoot.bvh
	walk_R180_Rfoot.bvh
Turning Right on Left Foot	walk_R45_Lfoot.bvh
	walk_R90_Lfoot.bvh
	walk_R135_Lfoot.bvh
	walk_R180_Lfoot.bvh
Jog to Walk Transitions	jog_2_walk_Lfoot.bvh
	jog_2_walk_Rfoot.bvh
	walk_2_jog_Lfoot.bvh
	walk_2_jog_Rfoot.bvh
Jogging Straight	jog1.bvh
	jog2.bvh
Jogging Turns	
Turning Left on Right Foot	jog_L45_Rfoot.bvh
	jog_L90_Rfoot.bvh
	jog_L135_Rfoot.bvh
	jog_L180_Rfoot.bvh
Turning Left on Left Foot	jog_L45_Lfoot.bvh

Motion	File
	jog_L90_Lfoot.bvh
	jog_L135_Lfoot.bvh
	jog_L180_Lfoot.bvh
Turning Right on Right Foot	jog_R45_Rfoot.bvh
	jog_R90_Rfoot.bvh
	jog_R135_Rfoot.bvh
	jog_R180_Rfoot.bvh
Turning Right on Left Foot	jog_R45_Lfoot.bvh
	jog_R90_Lfoot.bvh
	jog_R135_Lfoot.bvh
	jog_R180_Lfoot.bvh
Loitering Motions	Loiter_1.bvh
	Loiter_2.bvh
	Loiter_cross.bvh
	Loiter_cross2.bvh
	Loiter_watch.bvh
	Loiter_shoes.bvh
	Loiter_point.bvh
	Loiter_point.bvh
	Loiter_sunblock.bvh
	Loiter_gesture.bvh
	Loiter_twist.bvh
	Loiter_heart.bvh
	Loiter_scratch.bvh
	Loiter_scratch1.bvh
	Loiter_scratch2.bvh
	Loiter_stretch.bvh
	Loiter_celphone.bvh
Walk starts in different directions	walk_start_L45.bvh
	walk_start_L90.bvh
	walk_start_L135.bvh
	walk_start_L180.bvh
	walk_start_R45.bvh
	walk_start_R90.bvh

Motion	File
	walk_start_R135.bvh
	walk_start_R180.bvh
Walk facing one direction and moving in another	walk_facesame_L45.bvh
	walk_facesame_L90.bvh
	walk_facesame_L135.bvh
	walk_facesame_R45.bvh
	walk_facesame_R90.bvh
	walk_facesame_R135.bvh
Walking backwards	walk_backward1.bvh
	walk_backward2.bvh
Wild Dance Motion	Dynamic_1.bvh
	Dynamic_2.bvh
	Dynamic_3.bvh

## Sample Animations

The software ships with a collection of files to assist your learning and animation practice. Some of these files will be copied to your hard disk when you first install the software. Other files remain on the **character studio** Installation CD. You will need to load these from the CD when you want to use them.

The software ships with a set of sample characters, and a library of motion files. There are character FIG files, and MAX files as well. The footstep and freeform animations are available as BIP files, and the motion capture files are either BVH or CSM.

## Directory Structure

When you first install **character studio**, the installer will create a *Cstudio* directory under your 3D Studio MAX root, which contains all the character and motion files, as well as the tutorial files.

Beneath *Cstudio*, you will find six sub-directories: Charactr, Motions, Docs, Images, Scripts and Tutorials.

**Charactr** has two subdirectories, Figures and Meshes. Here you find the 3D Studio geometry (*.max*) and the biped figure files (*.fig*)

**Motions** has three subdirectories, Footstep, Freeform and Motion Capture. Here you will find motion files that can be used to animate the biped characters. For a description of these files, see *Motion Library* (see page 284).

**Tutorials** directory has subdirectories for each of the Online Tutorials. Go to the appropriate tutorial subdirectory to find all the files needed for a particular tutorial.

## Sample Files

Some of the features and parameter settings are best explained by viewing sample animation. The character studio CD contains a directory of AVI files that demonstrate some of these features. The scene files used to generate the animation are also included on the CD so you can experiment with changing the settings, see the movement from other views, and so on.

## Procedures

### To play a sample AVI files

1. Place the **character studio** CD-ROM in the drive.
2. From this page, click the help topic you're interested in and note the name of the sample AVI file.
3. Start the Windows Media Player (or select View File from within 3DS MAX).
4. Navigate to the *cstudio\samples* folder and open the AVI file.

Note: The 3DS MAX file used to generate the animation is also included on the CD-ROM so

you can experiment with changing the settings, see the movement from other views, and so on.

*Ballistic Tension (see page 291)*

*Gravitational Acceleration (see page 291)*

*Stride Length (see page 291)*

*Time to Next Footstep (see page 292)*

*Center of Mass Position (see page 292)*

*Dynamics Blend (see page 292)*

---

## Ballistic Tension

This animation shows three bipeds with different settings for the Ballistic Tension parameter.

This parameter controls the amount of spring or tension when the biped lands or takes off from a jump or run step. You can see this effect especially in the difference in the amount of bend in the knees between frames 53 and 63 and frames 96 and 106.

**Left biped:** Ballistic Tension=0.0

**Center biped:** Ballistic Tension=0.5

**Right biped:** Ballistic Tension=1.0

Animation file name

balltens.avi

3DS MAX file name

balltens.max

---

## Gravitational Acceleration

This animation shows three bipeds with different settings for the GravAccel parameter.

This parameter controls how high the biped jumps and how fast it falls.

**Left biped:** GravAccel=624

**Center biped:** GravAccel=412

**Right biped:** GravAccel=206

Animation file name

gravacl.avi

3DS MAX file name

gravacl.max

---

## Stride Length Sample Animation

This animation shows a biped whose stride length is increasing as it walks.

Interpolating the Parametric Stride Length of the walk from .25 for the First Step to 1.25 for the Last Step in the Create Multiple Footsteps dialog created this animation.

Animation file name

stride.avi

3DS MAX file name

stride.max

---

## Time to Next Footstep

This animation shows a running biped whose speed gradually increases during the run, although the footsteps are evenly spaced.

Interpolating the Time to Next Footstep parameter from 28 for the First Step to 18 for the Last Step in the Create Multiple Footsteps dialog created this animation.

### Animation file name

cycrun.avi

### 3DS MAX file name

cycrun.max

---

## Center of Mass Position

This animation shows two bipeds that differ in the position of the center of mass object in relation to the rest of the body.

**Left biped.** Center of mass correctly positioned inside the pelvis.

**Right biped.** Center of mass incorrectly positioned in front of the body.

Note how the walk of the biped on the right is jerky while the walk of the biped on the left is smooth.

### Animation file name

com pos.avi

### 3DS MAX file name

com pos.max

---

## Dynamics Blend

This animation shows two bipeds with different settings for the Dynamics Blend parameter.

This animation shows a case in which turning off full dynamics at the peak of the vertical motion creates a more graceful effect.

**Left biped:** Dynamics Blend=0.0

**Right biped:** Dynamics Blend=1.0

### Animation file name

dynblend.avi

### 3DS MAX file name

dynblend.max

## Physique User Interface

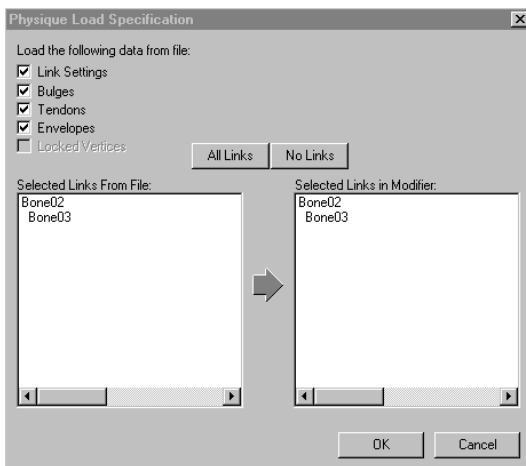
Controls in the Physique and Physique Floating Bones rollouts are for attaching the mesh to the biped, splines, or bones. Controls in the Level of Detail rollout are for troubleshooting envelopes, bulges, and tendons. The Sub-Object controls are for fine tuning envelopes, creating and adjusting tendons and bulges, and for vertex editing.

### See also

*Introduction to Physique (see page 11)*

## Physique Load Specification Dialog

Select an object with the Physique modifier. > Modify Panel > Physique rollout > Open Physique (.phy) File. > Choose File. > Physique Load Specification Dialog



At the upper left of the dialog are check boxes that let you specify which kind of data to load:

**Link Settings.** Load link parameters. Default=selected.

**Bulges.** Load cross sections and bulge angles. Default=selected.

**Tendons.** Load tendons. Default=selected.

**Envelopes.** Load envelopes. Default=selected.

**Locked Vertices.** Load locked vertices.

**All Links.** Select all links to load.

This button is enabled only if the number of saved links equals the number of open links. Click to select all links in both lists. If All Links is disabled, you must choose links by hand. Click the name of a link in a list to select it. The number of links to load must equal the number of links to update before you click OK.

**No Links.** Deselect all links in both lists.

After deselecting all links, you must select the links to load and to update, before you click OK. Turn off a box to avoid loading that kind of data.

**List Windows.** At the bottom left of the dialog is a list of links that were saved in the file, and at the bottom right is a list of links in the currently open Physique skin and skeleton. You can choose which links to update with the saved data.

## Box Generator Utility

Use Box Generator, on the Utilities panel, when using Physique with 3DS MAX bones. Physique uses the boxes to determine the radial scale for its default envelopes.

### Procedures

To create bounding boxes on a non-Biped hierarchy

1. Open the Utility panel and select Box Generator.
2. Set the name of the boxes in the Box Name field.
3. Set the Width parameters of your boxes so they roughly fit your mesh.  
You can either base the box width on a percentage of the link length, or use an absolute value where all the boxes will have the same width.
4. With your box parameters specified, click Pick Root Bone.
5. Click the root bone of your hierarchy either by selecting it in the viewport, or selecting it from the 3DS MAX Select By Name dialog.
6. After this utility generates the boxes for you, non-uniformly scale the box width and length to snugly fit your mesh. You can also adjust the object parameters on the Modify panel to fit the boxes in the mesh.

When attaching your Physique mesh to your bones, choose the Object bounding box option for generating default cross sections, and your Physique mesh should deform fairly well by default.

### To disable the Non-Uniform Scale warning

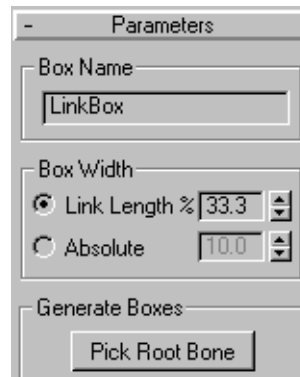
When you use Non-Uniform Scale to scale an object in 3D Studio MAX, a warning appears

that the scale information is not contained in the Modifier stack. This warning does not apply to biped objects, and can be disabled, as follows:

1. Choose Customize > Preferences.
2. On the General panel, in the UI Display area, clear the Display NU Scale Warning check box.

This setting is saved in the *3dsmax.ini* file.

### Interface



**Box Name.** Type a name to append to the created boxes.

**Link Length.** Size the width of the created boxes on a percentage of the link length.

**Absolute.** Size the width of the created boxes on an absolute value.

## Floating Bones Rollout

Select an object with the Physique modifier. > Modify panel > Floating Bones rollout

Use splines, a bones hierarchy, or unattached bones to deform a mesh with controls in the Physique Floating Bones rollout. By animating spline vertices you can animate a mesh for example. Envelopes are created automatically for the selected splines or bones.

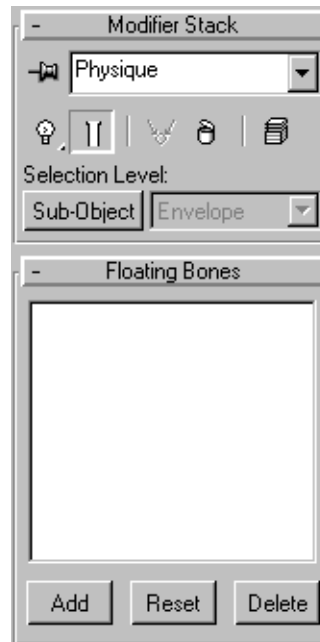
### Procedure

#### To have a spline influence a mesh

1. Create a spline and place it within a mesh that has Physique applied.
2. On the Floating Bones rollout click Add.
3. Select the spline in the Select Bones dialog.
4. Animate the spline vertices to animate the mesh.

You can substitute the spline for 3DS MAX bones if you want to animate bones.

### Interface



**Add.** Click Add and select splines or bones in the viewports.

**Reset.** Resets the initial position of splines or bones, vertices are reassigned but envelopes are left as is.

**Delete.** Select a spline or bone in the list window and click delete to delete it.

## Physique Rollout

Select an object with the Physique modifier. > Modify panel > Physique rollout



Use controls on the Physique rollout for linking a mesh to a biped or 3DS MAX bones hierarchy, to load and save Physique files, and to initialize or reinitialize the Physique parameters on a mesh. Open the Bulge Editor to create and edit Bulge Angles.

Control Physique viewport display using parameters on the *Physique Level of Detail rollout* (see page 298).

Both rollouts display when Sub-Object is not active.

### Procedures

#### To attach a mesh to a biped using Physique


1.  On the Motion panel, On the General rollout, click to activate Figure mode.
2. Fit a biped to your character mesh.  
If the mesh is made of multiple objects, select all of them.
3. With the mesh or mesh objects selected, choose Physique on the Modify panel.  
Click More on the Modify panel to locate the Physique modifier.
4.  Click Attach to Node and select the biped Pelvis object.  
*The Physique Initialization dialog* (see page 335) displays.
5. Click Initialize, accepting the default settings.

By default, the Object Bounding Box option is selected. Envelopes will be sized to approximate the biped limbs.


6. Click to turn on Sub-Object selection, and choose Envelope to edit the newly created links and envelopes.

Adjust envelopes with the biped in an animated position. Adjust envelopes around problem areas.

#### To attach a mesh to a 3DS MAX bones hierarchy using Physique

1. Create a bones hierarchy.
2. Place the bones hierarchy inside a mesh.
3. Optionally, use the *Box Generator utility* (see page 294) to create boxes around the bones.  
Resize the boxes to closely fit the mesh to help Physique give better default envelopes.
4. Select the mesh.
5. Select Physique on the Modify panel.
6.  Turn on Attach to Node, then select the root node in the bones hierarchy.  
The Physique Initialization dialog displays.
7. Click Initialize.  
Click to turn on Sub-Object selection, and choose Envelope to edit envelope parameters.  
Note: Use frame 0 as a “figure mode” when a bones hierarchy is used. Frame 0 should be used as the fit position of the bones to the mesh.

#### To add a bone after Attach to Node is used

1.  Link the bone to the biped, then click Reinitialize.
2. Select Include New Bone in the Physique Initialization dialog, and click Initialize.



### To save Physique data

1. With Physique active on the Modify panel, click Save Physique File on the Physique rollout.
2. Enter a name for the new Physique file, and then click OK.

### To load Physique data

1. With Physique active on the Modify panel, click Open Physique File on the Physique rollout.
2. Choose the Physique file to open, and then click OK.

A Physique Load Specification dialog is displayed.

3. Use the dialog to choose the kind of data and then select links in the source file in the list on the left of the dialog. Select links you want to copy to in the list on the right of the dialog.

Note: The number of links selected from the file (left column) must match the number of links in the current modifier (right column).

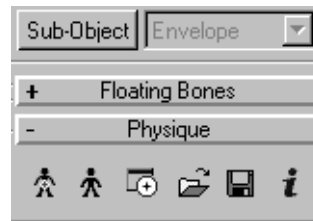
4. Click OK.

Physique updates the links you chose with the data that was saved.

**Warning:** The number of selected input links must match the number of selected target links.

Physique updates the links you chose with the data that was saved.

### Interface



**Attach to Node.** Attach the selected mesh or mesh objects to the biped or to a 3DS MAX bones hierarchy.

The biped should be in Figure mode and fitted to the mesh character before Attach to Node is used.

Turn on Attach to Node, then in the viewports click the biped pelvis or the root node in a 3DS MAX bones hierarchy.

After selecting a node in the viewports, the *Physique Initialization dialog* (see page 335) displays. Select envelope creation parameters, then click Initialize in the dialog. Physique follows the hierarchy and creates envelopes for all the found links (includes 3DS MAX bones linked to the biped). The envelopes influence the vertices on the mesh to follow the animated hierarchy.

Note: If the center of mass is selected by mistake, Physique corrects this by attaching to the biped pelvis instead.



**Reinitialize.** Displays the Physique Initialization dialog and resets any or all of the Physique attributes to the default values. For example, *reinitializing* (see page 338) with the Vertex Settings option selected reestablishes the relationship of a vertex and its original position relative to the Physique deformation spline.

Settings for vertex-link assignments, bulges, and tendons can be reset from this dialog.



**Bulge Editor.** A graphical alternative to Bulge sub-object for creating and editing bulge angles. Displays the *Bulge Editor* (see page 318).



**Open Physique File.** Loads a saved Physique file (.phy), which contains envelope, bulge angle, link, tendon and vertex parameters.

When you click Open Physique File to load a .phy file, the *Physique Load Specification dialog* (see page 293) displays. Select the links you want to import in the list on the left of the dialog and apply them to the links you select in the list on the right of the dialog. Pay particular attention to the link names that you want to copy, the name for the biceps link created by Physique on a default biped is actually called “Bip01 L Forearm,” for example.

Any aspect of your Physique work can be loaded into any other character. You can choose from Link Settings, Bulges, Tendons and/or Envelopes. You can also apply links from the file to differently named links in the current scene. For example, save and reload a .phy file to copy a bulge angle created on one character to a link on another character.

**Tip:** A bulge created on a bones hierarchy can be loaded and applied to Physique links created using a biped.

Note: Loading a .phy file will not overwrite locked vertices. If you want to overwrite them, you must go to Vertex sub-object level and unlock them, or reinitialize with the Link-Vertex Assignments option selected.



**Save Physique File.** Save a Physique file (.phy), which contains envelope, bulge angle, link, and tendon parameters.



**About Physique.** Displays version and copyright information about Physique.

---

## Physique Level of Detail Rollout

Select an object with the Physique modifier. > Modify panel > Physique Level of Detail rollout

Controls in the Physique Level of Detail rollout not only optimize the viewports, but also affect the rendered result. The primary purpose of this rollout is for troubleshooting. After creating bulge angles and tendons you can turn off their influence to see exactly what they add to the deformation of the skin. This area also has controls for how changes in the modifier stack below Physique are handled.

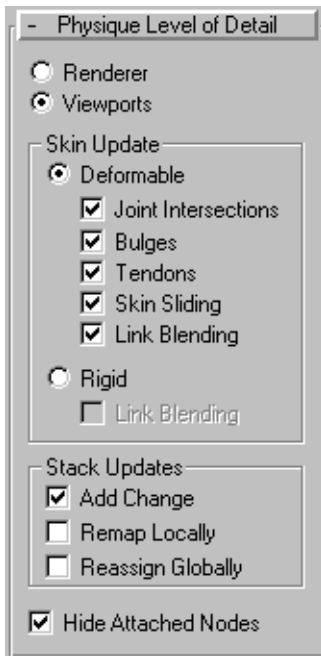
**Tip:** Select Deformable and all its parameters during editing; this allows you to spot problem areas. Select Rigid for the fastest viewport redraw.

## Procedure

### To troubleshoot bulges and tendons

1. Select a character that has bulges or tendons.
2. Toggle Cross Sections and Tendons on and off.
3. Examine the mesh with and without the influence of bulges and tendons.
4. Make appropriate changes to bulges and tendons.

## Interface



**Renderer.** Select and choose skin parameters in the Skin Update area to included in rendered images.

**Viewports.** Select and choose skin parameters in the Skin Update area to include in viewport images.

### Skin Update group

**Deformable.** Turn off all use of the Physique Deformation Spline. Turn on for highest quality rendering.

**Joint Intersections.** Turn off to remove joint intersection influence. To allow the mesh to overlap itself; for example, at the elbow and knee joints.

**Cross Sections.** Turn off to remove any bulge angle influence.

**Tendons.** Turn off to remove any tendon influence.

**Skin Sliding.** Turn off to remove skin sliding influence.

**Link Blending.** Turn off to remove the influence of link blending.

**Rigid.** When selected, forces all vertices to use Rigid assignments rather than Deformable. This is an easy way to isolate deformation spline problems. It also provides the quickest redraw speed. You might choose this if you were adjusting the animation of your skeleton.

**Link Blending.** Blend rigid links.

### Stack Updates group

If the vertex count changes, vertices are reassigned globally regardless. Add Change adds in changes based on the vertices initial position. The other options reset the initial position each frame to do the remapping and reassigning. For this reason choosing Add Change or making non-animated stack changes should always be done at the initial position (Figure mode or frame 0).

**Add Change.** Adds in changes from the stack and then applies Physique deformation. No vertex remapping or reassigning is performed.

This option will generally give you the deformation you want. There is no performance penalty from physique when this option is used.

**Remap Locally.** For deformable vertices this resets vertex position on the Physique deformation spline used for bending and the link position used to interpolate twist. For rigid vertices this option resets the link position used to interpolate twist.

When vertices are sliding along the length of the spline and you want them to bend and twist

based on the spline position but don't want vertex weights to change, turn this option on.

**Reassign Globally.** Re-weights, and resets the position on the spline used for bending for moved vertices globally. The vertex link assignment, weighting, and spline position are reset for all moving points on every frame. This is equivalent to Physique 2.2. This option is like reinitializing on every frame.

When vertices are moving to different envelopes and you want them reassigned to the new envelopes use this option.

**Hide Attached Nodes.** Toggles the display of the underlying skeletal system. This allows you to hide and unhide the biped, for example.

---

## Envelope Sub-Object

Select an object with the Physique modifier. > Modify panel > Sub-Object Envelope

After Attach to Node is used to create links and envelopes in a hierarchy or if splines and bones are added using the Add command in the Physique Bones rollout, envelopes are adjusted using the tools in Envelope Sub-Object to correct the way skin behaves. Usually the first adjustment is to assign a rigid envelope to the characters head. The head should not deform with the deformation spline, a rigid envelope blends with other envelopes but retains the shape of the head.

Tools on the 3DS MAX toolbar can also be used. For example you can use the Non-Uniform Scale tool on envelopes and cross sections or use the Select and Move tool to move envelopes, cross sections, or control points on cross sections.

Envelopes are the basis of Physique's Blending. They provide smooth deformation of the skin across the joints. Envelopes are used to

determine which links have influence over which vertices. Vertices falling between overlapping envelopes receive influence from each, thus creating a smoothly blended transition. Envelope sub-object level is where you can adjust envelopes to encompass more or fewer vertices or you can adjust the characteristics of an envelope such as its relative strength. The goal is to modify the envelopes so each vertex is encompassed by at least one link's envelope. The outer bounds should have sufficient overlap between links to provide a smooth blend at the joints.

## Patches

Patch tangent handles do not need to be encompassed by an envelope to work. Only a patch vertex needs to be encompassed by an envelope. This relieves you having to scale the envelopes to a large size to encompass tangent handles.

## Workflow

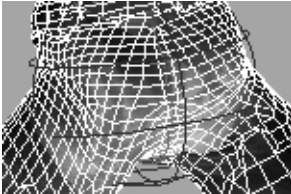
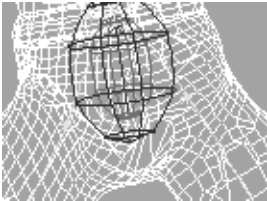
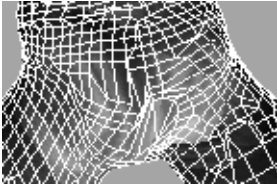
With Initial Skeletal Pose turned on, check that the envelopes cover all areas of the model. If you turn off Initial Skeletal Pose, the character adopts its animated position. Use an animation that stretches the character around such as a run or dance file, for example. Find a frame where the envelopes need adjusting and edit the envelope parameters. Changing the envelopes with the character in an animated position will always reference the Initial Skeletal Pose.

Indications that envelopes need adjustment include vertices that don't follow the skeleton. To correct for this increase the radial scale of an envelopes outer boundary. Vertices at joints get pushed or dented in. Too many envelopes may be affecting these vertices – try reducing the inner bound radial scale and overlap values.


Bending appears too linear or broken at the joint, the envelope's outer overlap values should be increased.

## Procedure

**To adjust envelopes around the biped's pelvis with Physique**



Biped's Triangle Pelvis option on the Structure rollout was used here. The deformable envelope from the pelvis to the lower spine object is too small to correct the mesh problem in the pelvis area.

1.  Turn on Link and select the middle link.
2. Select Both and increase the Radial Scale value until the outer envelope just surrounds the pelvis area.  
Both envelope bounds scale together.

3. Increase the Parent Overlap value until the red envelope appears through the mesh.  
Decrease the Falloff value to strengthen this envelope if necessary. Adjust the thigh and buttock envelopes for overlap to finish correcting this area.

**To select cross sections in the inner or outer bounds**

Click the Inner or Outer button in the Envelope Parameters area if you want to work with cross sections solely in that part of the envelope. Click Both to be able to work with both Inner and Outer cross sections at the same time.

- Click Select Object on the 3DS MAX toolbar, and click the cross section you want to select. Use CTRL+click to add another to the selection set.

As you move the cursor over a selectable cross section, it changes to a plus shape, letting you know the object at that location is selectable.

A color you can specify indicates which cross section is selected. By default, selected cross sections are yellow.

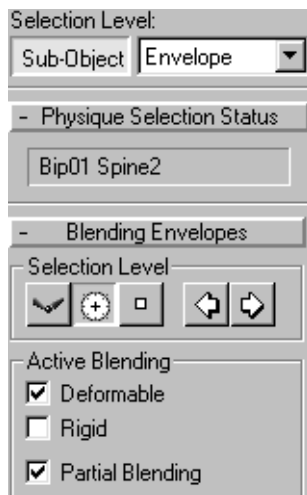
Use Radial Scale to resize the selected cross section, and the 3DS MAX transform tools to move, rotate, and scale it.

To move to the adjacent cross section within the current level, press Next or Previous.

**To copy an envelope and its settings to the mirrored link**

1. With the source link selected, click Copy to save the settings to the buffer.
2. Select the mirror link and click Paste.
3. Click Mirror and the selection immediately flips about the mirror axis.
4. Use Rotate on the 3DS MAX toolbar to rotate the selection to the final orientation.

## Interface



**Link.** Turn on to select links in the viewports and edit the selected links envelope parameters.

Turn on link, select a biceps link, CTRL+Click the opposite bicep to add it to the selection, then edit envelope parameters for both links at the same time for example.



**Cross Section.** Edit envelope cross sections to change envelope shape and area of influence.

Select a cross section on the inner or outer boundary of an envelope and then move or scale it. Non-Uniform Scale the cross section of the collar envelope to avoid vertices in the chest area, for example.



**Control Point.** Edit the control points on a cross section.

Select a point on a cross section of an envelope and reposition it to change an envelopes shape and area of influence.



**Next/Previous.** Move to the next or previous link, cross section or control point. This depends on Selection Level status.

**Deformable.** Enable or disable deformable envelopes for the selected link(s).

By default, deformable envelopes are red.

**Rigid.** Enable or disable rigid envelopes for the selected link(s).

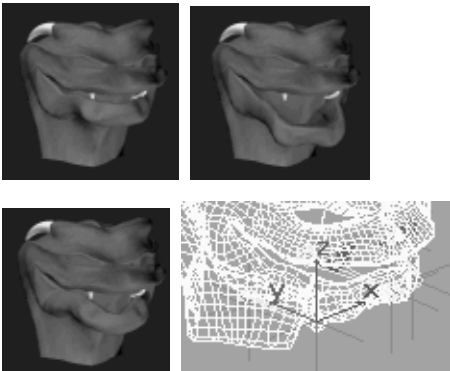
By default, rigid envelopes are green.

Note: You can have deformable and rigid envelopes turned on for the same link.

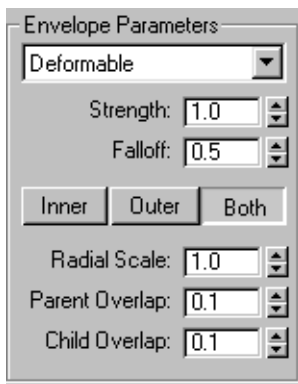
Normally, you use one or the other. By non-uniform scaling the rigid and deformable envelopes for one link, you could position one envelope on top of the other. For example, you could deform the shoulder with a rigid envelope and the armpit with a deformable envelope. Both envelopes can also be turned off for a link.

**Partial Blending.** Turns on partial blending for the selected link(s). Leave either a deformable or rigid envelope on and then turn on Partial Blending.

Physique calculates the weights of each link on a given vertex. If Partial Blending is *off*, and the total weight is less than 1, Physique normalizes the combined weight to 1. If Partial Blending is *on*, the weights are allowed to remain at a value of less than 1. The remainder is filled in by the root node, the equivalent of no influence. See *How Partial Blending Calculates Weights* (see page 306) for a detailed explanation of how the program calculates these values.



**Partial Blending on all the deformable envelopes for the bones in this character's jaw allows smooth mesh deformation when the bones are moved.**



## Envelope Parameters

The Strength and Falloff values apply to both the Inner and Outer Envelopes bounds.

**Strength.** Changes the strength of the envelopes. Range=0 to 100; Default=1.

Used primarily for areas where envelopes overlap and you want one to be more influential than the other.

**Falloff.** Changes the rate of falloff between the Inner and Outer bounds of an envelope. This is

a Bezier curve function. Range=0 to 1.0; Default=0.5.

Vertices within the inner bound are fully influenced (weight=1) and those beyond the outer bound get no influence from the link (weight=0). Falloff determines the rate at which the influence falls off from 1 to 0.

**Inner.** Turn on to change the values of the Inner bound.

**Outer.** Turn on to change the values of the Outer Bound.

**Both.** Turn on to change Values for both Inner and Outer Bounds at the same time. Changes to this value are equally applied to both the Inner and Outer Bounds.

Displayed values for Radial Scale and Overlap reflect the values for the Inner Bound when Both is selected.

**Radial Scale.** Scales envelope bounds radially. Range=0 to 100.

**Parent Overlap.** Changes the envelope overlap to the parent link in the hierarchy. Range= -1 to 10; Default=0.1.

A value of zero causes the end of the envelope to fall on the joint. Values less than 0 bring the envelope inside the link and values above 0 will overlap onto the adjoining link.

**Child Overlap.** Changes the envelopes overlap to the child link in the hierarchy. Range= -1 to 10; Default=0.1.

A value of zero causes the end of the envelope to fall on the joint. Values less than 0 bring the envelope inside the link and values above 0 will overlap onto the adjoining link.



## Edit Commands

Command functions here will vary depending on whether Links, Cross Sections, or Control Points are turned on in the Selection Level area.

**Insert.** Inserts a cross section or control point on a cross section.

**Delete.** Deletes a Cross Section or Control Point.

**Copy.** Copies an Envelope or Cross Section.

**Paste.** Pastes an Envelope or Cross Section.

**Mirror.** Mirrors the envelopes on a selected link or mirrors selected cross sections in an envelope.

After a Mirror operation, click Rotate on the 3DS MAX toolbar, turn on Local coordinates, then click and drag over a Link or Cross Section to adjust orientation.

Note: To copy an envelope to its mirror, the sequence would be Copy, select the opposite link, Paste, then Mirror.

**Exclude.** Excludes links from influencing a selected link(s); a link will not blend with any link in its exclusion group. For example, exclude the right thigh link from influencing the left thigh link. Displays the modeless Exclusion dialog.

For example, rather than scaling the index finger envelopes to avoid vertex influence on the middle finger, exclude the middle finger links from the index finger links.

## Exclude Envelopes dialog

**Link Envelopes List (left side).** Displays available links to exclude, selected links are not in this list.

**Exclude Envelopes for Selected Links List (right side).** Envelopes to exclude from the current link selection appear in the list. The text field displays the name of the selected link.

**Right Arrow.** Select links in the Link Envelopes List on the left, then click the right arrow to add them to the Exclude Envelopes for Selected Links list on the right.

**Left Arrow.** Select links in the Exclusion Envelopes for Selected Links list on the right of the dialog, then click the left arrow to remove them from the exclusion list.

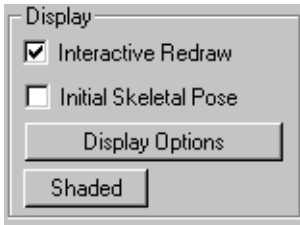
**Clear.** Clear the Exclude Envelopes for Selected Links List. Selected links do not exclude other links.

**Display Subtree.** The Link Envelopes list increments links according to their level in the hierarchy. This visual aid can help you to find and select links.

Leave this modeless dialog open, choose a link or links in the viewports, select links to exclude in the Link Envelopes List on the left and use the Right Arrow to add them to the Exclusion list.

Note: To get the new exclude functionality in **character studio 3** you must reinit old files with initial skeleton pose, and vertex assignments checked.





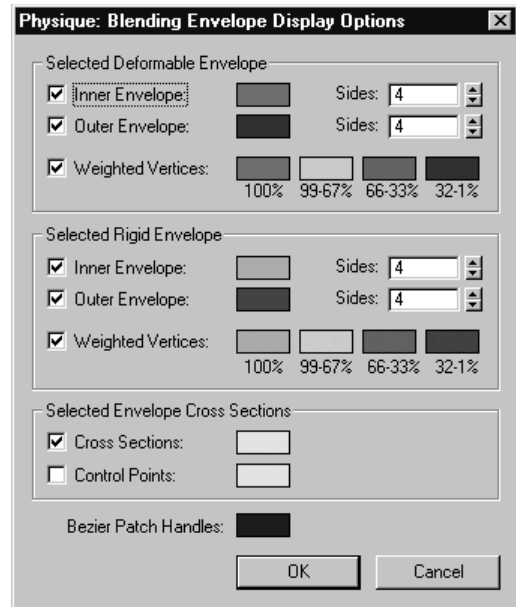
**Interactive Redraw.** Turn on to dynamically update the mesh when adjusting envelopes.

When this is off the mesh updates at the end of envelope size change.

**Initial Skeletal Pose.** Puts the mesh character in the pose it was in just before Physique was applied.

**Shaded.** Toggles shaded display of vertex weights in the viewports.

**Display Options.** Customize the envelope display. Displays the Blending Envelope Display Options dialog. As well as selecting what is displayed, the color for each parameter can be selected.



### Selected Deformable/Rigid Envelope groups

All parameters except Control Points are on by default.

**Inner.** Toggle viewport display of inner envelopes. Change the viewport colors of the inner envelope boundary.

**Outer.** Toggle viewport display of outer envelopes. Change the viewport colors of an outer envelope boundary.

**Weighted Vertices.** Toggle viewport display of weighted vertices. Change viewport colors of weighted vertices.

**Sides.** Number of envelope sides in the viewport display.

### Selected Envelope Cross Sections

**Cross Sections.** Toggle viewport display of cross sections. Change viewport colors of selected cross sections.

**Control Points.** Toggle viewport display of control points. Change viewport colors of selected control points.

**Bezier Patch Handles.** Set the viewport color of Bezier patch handles.

## Partial Blending and Weight Assignments

### In cases where no envelopes use Partial Blending (the default)

Vertex  $v$  is assigned to links  $l1$ ,  $l2$ , and  $l3$ , and the weights for these links are:  $w1 = .2$ ,  $w2 = .3$  and  $w3 = .4$

In a non-partial blended case, the sum of these vertex weights is  $w1 + w2 + w3 = .9$  (less than 1.0).

With partial blending off, Physique will normalize the weights so they sum to 1.0, for example:

$$\begin{aligned} w1' &= w1 / w1+w2+w3 = .2/.9 = .222222... \\ w2' &= w2 / w1+w2+w3 = .3/.9 = .333333... \\ w3' &= w3 / w1+w2+w3 = .4/.9 = .444444... \\ \text{so } w1'+w2'+w3' &= 1.0 \end{aligned}$$

### In cases where all envelopes use Partial Blending

The weight fill-in for vertices with a weight less than 1 will always fill with the weight of the root. Whenever a vertex is assigned to a group of links, all of which are partially blended, then the remaining weight will be assigned and blended with the root link.

This works well for a case where you want partial deformation falling off to no deformation. An example would be a static head, where you are deforming the head for facial expressions, but the head itself remains in place.

Example: Vertex  $v$  is assigned to links  $l1$ ,  $l2$ , and  $l3$ . The weights for these links are:  $w1 = .2$ ,  $w2 = .3$ , and  $w3 = .4$ .

The sum of these vertex weights is  $w1 + w2 + w3 = 0.9$  (less than 1.0). We need an additional fill-in weight ( $wf$ ). The fill-in weight,  $wf$ , is determined by  $1.0 - (w1+w2+w3) = 0.1$  or 10%. Physique fills in with weight  $wf$  from the root (leaving the vertex partially undeformed).

The resulting deformation will be  $w1*l1 + w2*l2 + w3*l3 + wf*root$ . The root portion of this deformation is essentially an undeformed portion that simply follows the root of the skeleton.

### In cases where some envelopes use Partial Blending and some do not

The vertex weight fill-in in overlap areas will be based on the percentage of partial and non partial weights. For example, If the total weight of non partial links is 80% of the total summed partial and non partial weight, then 80% of the fill-in will be more of the non partial deformation. The remaining 20% fill-in will come from the root.

Example: If Vertex  $v$  is assigned to links  $l1$ ,  $l2$ , and  $l3$ , the weights for these links are:  $w1 = .2$ ,  $w2 = .3$ , and  $w3 = .4$ .

Let's assume  $l1$  and  $l2$  are Non Partial, and  $l3$  is Partial. The Non Partial weight is  $w1 + w2 = .5$ ; the Partial weight is  $w3 = .4$ ; the Non Partial weight is  $.5/(.5+.4) = .555555$ , or 56%; and the fill-in weight is still  $(1.0 - .9)$  or  $wf = .1$ .

The program fills in with 56% of  $wf$  with more of the Non Partial Blended Links. The remaining 44% of  $wf$  is filled in with the root as in the partial blended case. This provides a smooth transition between the partial and non-partial links.

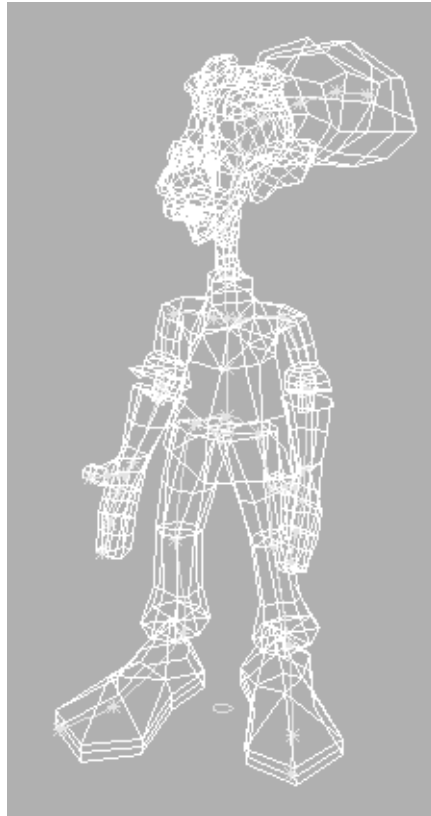
---

## Link Sub-Object

Select an object with the Physique modifier. > Modify panel > Sub-Object Link

Use parameters at the Link sub-object level to change how deformation around joints occurs. When a joint in the skeleton bends or rotates, Physique, by default, deforms vertices uniformly on either side of a joint. Change these defaults using the tools at the Link sub-object level. For example, adjust the amount of skin sliding that occurs along a limb as the limb bends, or change the angle of the crease between the upper arm and chest.

## The Physique Deformation Spline



Like a 3DS MAX spline, the *deformation spline* created by Physique is a continuous curve through several points. While a 3DS MAX spline runs continuously from vertex to vertex, the deformation spline is a smooth curve running from joint to joint. The Bend, Bias, and Tension spinners change the shape of the curve, analogous to rotating or scaling the spline handles at a 3DS MAX spline vertex.

The deformation spline also takes into consideration twisting and scaling of the skeleton's links. At the Link sub-object level, you take control of the behavior of the deformation

spline, and subsequently gain full control of the skin's behavior relative to the skeleton's movement.

## Procedure

### To adjust joint intersection parameters

1. With Physique active on the Modify panel, click to turn on Sub-Object on the Modifier Stack rollout.
2. Choose Link from the Selection Level dropdown.
3. Click the link you want to work with.
4. Select Active for the crease you want to manipulate; choose Crease at Parent's Joint or Crease at Link's Joint.
5. Change parameters on the Physique Joint Intersections rollout.

Assuming that you are working with the joint between the selected link and its child (the biceps link, for instance), select Crease At Link's Joint and change parameters. You then need to select the child link (the forearm) and activate its Crease At Parent's Joint. Adjustments to the Parent's Joint for this link will affect the other side of the same crease.

## Interface

### Link Settings rollout



**Active.** Activates the selected link when selected. Default=selected.

Turning off Active makes the link unavailable for vertex assignment, meaning that the link has no influence on *any* vertices. Any vertices within range of this link are not influenced by it and may be picked up by other nearby envelopes, or may be manually assigned without blending to any other link. Clearing Active makes Physique ignore that link as if it were never part of the skeleton.

**Continuity.** Maintains a smooth transition across the joint from the parent link to the current link. Default=selected.

When this box is selected, the effect of the link parameters passes smoothly across the joint to the connected link. When this is cleared, Bend, Twist, and Radial Scale are limited to the current link, which produces an abrupt transition across the joint. This is analogous to breaking the spline handles on a 3DS MAX spline.



**Bulge Editor.** Displays the Bulge Editor. The Bulge Editor represents bulge cross sections graphically.



**Reinitialize Selected Links.** Reinitializes the link and its vertex parameters based on the current link parameter settings. This does not change vertex assignments or manual reassignments.

For example, increasing the tension of a link can cause the spacing to change for the link's cross sections. Reinitializes Selected Links smooths the spline, and makes cross section spacing even again.

Think of the skin being snugly stuck to the link. As the link parameters change, the vertices may bunch up or stretch out. Clicking Reinitialize lets the skin “slip” to it's original shape, establishing a new relationship to the deformation spline

## Bend group

Bend tension affects the curvature of deformation spline through the joint. A low value, near 0, makes the spline linear and creates a sharp angle at the joint, like bending a hinge. High values, near 2, make the spline smooth through the joint, creating a rounded joint, like bending a firm hose. Bend bias pushes the effect

along the spline toward one side of the joint or the other.

**Tension.** Sets the smoothness of a joint. A value of 0.0 produces a sharp joint. A value of 2.0 produces a rounded joint. Range=0.0 to 2.0; Default=0.5.

**Bias.** Displaces the pivot point about which vertices are bent. The default value of 0.5 centers the bend at the joint. Values lower than 0.5 move the pivot onto the child link. At 0.0, the bend effect is limited to the selected link.

## Twist group

Twist parameters control the way the skin deforms when a joint rotates along its length, as in turning a doorknob.

**Use Twist for Rigid.** Turn on twist for rigid envelopes. Default=Off.

Turn off for the rigid envelope for a characters head, the head should not twist along the length of the link. You may want to turn this on for the forearm or upper-arm link if you're using rigid envelopes.

**Tension.** Values lower than 1.0 concentrate the effect closer to the joint. Values higher than 1.0 move the effect away from the joint. Range=0.0 to 2.0; Default=1.0

**Bias.** Shifts the distribution of the twist from one side of the joint to another. The default value of 0.5 twists the selected link and the child link equally. Values greater than 0.5 shift the twist to the child link. Default=0.5

## Sliding group

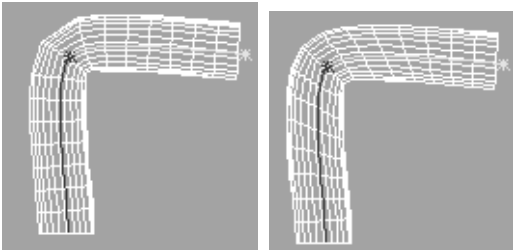
Skin sliding parameters control the amount of skin sliding that occurs when a joint rotates. Range=0 to 1. Both Inside and Outside are set to 0.0, off by default.

**Inside.** As values increase, skin moves away from the joint. Default=0.0

**Outside.** As values increase, skin moves towards the joint. Default=0.0

**FallOff.** As values increase, the effect is localized to the joint. Default=0.5

**Tip:** Use sliding for knees and elbows.



*Inside and Outside set to a value of 0 in the image on the left, and 0.25 in the image on the right.*

Without skin sliding, vertices closest to the joint tend to compress on the inside and stretch apart on the outside, generally revealing the segments of the mesh. Outside sliding causes the vertices around the joint to move toward the joint, preventing localized stretching on the side that is greater than 180 degrees. Inside sliding causes the vertices to relax and slip away from the joint, preventing bunching of vertices on the side having an angle less than 180 degrees.

#### Radial Scale group

Radial Scale parameters expand or contract the skin by scaling the radial distance perpendicular to the link. They apply to any combination of user definable scale, bulge settings, or link length.

**Tension.** Values lower than 1.0 concentrate the effect closer to the joint as Tension approaches 0.0. Values greater than 1.0 move the effect away from the joint. Range=0 to 2; Default=1.0.

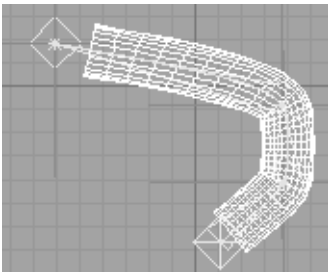
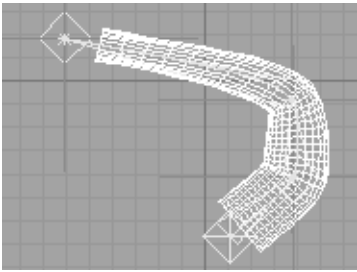
**Bias.** Shifts the effect of radial scaling. At the default value of 0.5, scaling affects both the selected link and the child link.

Values lower than 0.5 shift the scaling effect onto the selected link and values greater than 0.5 shift scaling onto the child link. At 0.0, expansion and contraction are limited to the selected link. At 1.0, expansion and contraction are limited to the child link. Default=0.5.

**Link Scale.** Scales the entire link radially, independent of the effect of any cross sections. At 1.0, the link is 'actual size' and this parameter has no effect. Other values increase or decrease the radial scale of the link. Range=0 to 10; Default=1.0.

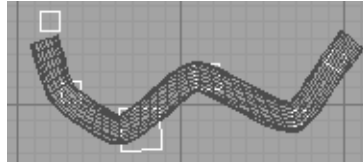
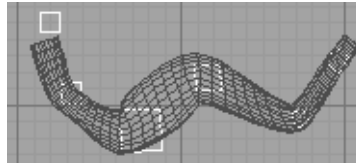
**CS Amplitude.** Cross section amplitude has no effect unless the link has bulge angle cross sections. At 0.0, cross section deformation is turned off. Values greater than 1.0, up to the maximum of 10.0, exaggerate the effect of cross sections. Default=1.0.

**Stretch.** When selected, preserves the volume of the link's skin when the length of the link changes. The effect is similar to stretching or squeezing a material that is only partly elastic, such as a tough rubber hose. Default=cleared.



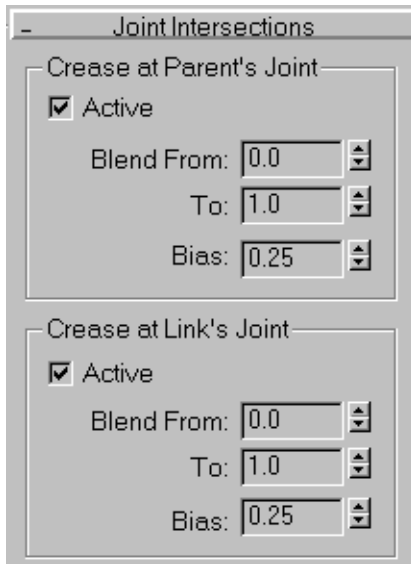
Stretch is turned on in the image on the left, and turned off in the image on the right. As the 3DS MAX bones change in length, the mesh expands and contracts when Stretch is selected.

**Breathe.** When selected, scaling a skeleton node changes the radial scale of the link's skin. When cleared, scaling a node has no effect on the scale of the skin. Default=cleared.



Linked 3DS MAX objects are shown in white. Notice that one of the objects is scaled larger than the others. In the left image, Breathe is selected, the mesh is scaled as the object is scaled. Breathe is cleared in the image on the right.

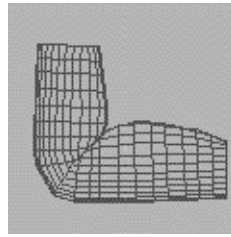
**Note:** The 3DS MAX objects linked together are used as the hierarchy for Physique.



### Joint Intersections rollout

When a joint bends, the skin can “collide.” Without collision detection, it can overlap unrealistically. This is especially likely when one or both of the links have bulges. Physique’s joint intersection controls can detect skin collisions and correct overlap by creasing the skin.

By default, Physique draws a crease plane that bisects the joint and prevents vertices from each link from crossing the crease plane. The result can be a somewhat unnatural flattening of the bulges along the plane. Physique therefore provides controls that allow you to modify the effect of the crease plane.



Properly adjusted crease plane

You can modify the effect of the joint creases at both ends of the selected link. These joints are the *parent’s joint* and the *link’s joint*.

- The parent’s joint is between the selected link and its parent. The parent’s joint is numbered 0.
- The link’s joint is between the selected link and its child. The link’s joint is numbered 1.

### Crease at Parent’s Joint group

The Blend From and Blend To points define the distance along the link that is affected by the crease plane. For the parent joint, the joint is at 0.0 and the crease plane has an effect from the joint to the Blend From point.

**Active.** Turns off the effects of the joint intersection controls. When this box is cleared, Physique makes no compensation for overlapping bulges.

**Blend From.** The area between Blend From and Blend To contains vertices that are partially affected by the crease plane. These vertices will be shifted, but not as much as those between the crease plane and the Blend From point. The distance of the vertices from the crease plane determines the exact amount of shifting. Vertices within the blend region that are closer to the intersection plane are shifted more than the vertices further away from it.



**Blend To.** Indicates the distance beyond which the crease plane has no effect.

**Bias.** Sets the strength of the crease plane effect within the blended region. A value of 0.0 means that the intersection plane will have no effect within the blended region; a value of 1.0 means that it will have a full effect within the blended region.

### Crease at Link's Joint group

For the link's joint, the joint is at 1.0 and the crease plane has an effect from the joint to the Blend To point. Vertices that are at locations less than the Blend From point are not affected by the crease plane; vertices between the Blend From and Blend To points are partially affected by the intersection plane.

---

## Bulge Sub-Object

Select an object with the Physique modifier. > Modify panel > Sub-Object Bulge

After you edit envelope parameters for good overall mesh deformation, create bulge angles to simulate muscle contraction and expansion when a character's limb rotates.

A bulge angle requires two links, the selected link and the child link of that selection (upper arm and forearm, for example). The joint that separates the two is referred to as the *bulge joint*. Any rotation applied to this joint becomes the angle used to center the effect of a bulge angle.

Physique creates a single default bulge angle for each bulge joint in the attached skeleton. The angle of this joint is that of the initial skeleton pose, created when the Physique modifier was applied to the skeleton, or when the Reinitialize command is used with Initial Skeleton Pose selected.

### Workflow to Create a Biceps Bulge

To create a biceps bulge angle, activate Bulge sub-object and click Link. Select the upper arm link, then click Insert Bulge Angle; the bulge angle name increments from 0 to 1 in the Current Bulge Angle field, indicating a new bulge angle has been created. Type a descriptive name for the new bulge angle in the Current Bulge Angle field, like *Arm at 90*, for example.

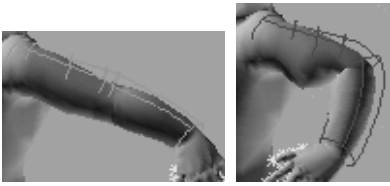
Click Bulge Angle Color and select a color for the newly created bulge angle. When Select Nearest Bulge Angle is active, the bulge angle color changes as the joint bends; this is a good visual indicator of which angle has the most influence at any given frame.

Scrub the time slider to a frame where the arm is bent to 90 degrees, then click Set Bulge Angle (you can also rotate the joint). Click Cross Section and Insert, then position the cursor over the upper arm link in the viewports to place a cross section for the biceps.

Click Control Points and move the cross section control points in the viewports to edit the cross section and bulge the biceps. Click Play, the biceps continues to grow as the angle of the character's arm approaches 90 degrees. If Select Nearest Bulge Angle is active, the bulge angle gizmos change colors, depending on which angle has the greatest influence.

**Note:** The Set Bulge Angle command only records the angle between the biceps and forearm, not the frame number where that angle occurs. Click Link and use Copy, Paste, and Mirror to create an identical bulge angle for the opposite arm.

**Tip:** Load a *.bip* file that will rotate the character's limbs. Turn on In Place mode before you scrub the time slider to locate appropriate limb angles. This keeps the character (biped) in the viewports.




After creating a new bulge angle with the Insert Bulge Angle command, scrubbing to frame 3 of this running motion puts the character's arm at a 90-degree angle, perfect for using the Set Bulge Angle command. Refer to the images above. The three cross sections that control the biceps "bulge" are created by clicking Cross Section and Insert, then clicking the upper arm link in the viewports. Click Control Point and drag the cross section control points to "bulge" the mesh or use the Cross Section view in the Bulge Editor to drag control points to "bulge" the mesh. Give each newly created bulge angle a name and color to make identification simpler.



## Bulge Editor

The *Bulge Editor* (see page 318) duplicates many of the controls in Bulge sub-object. The benefit of using the Bulge Editor is that bulge angle data is represented graphically and provides an alternative way of creating, selecting, and editing cross sections. Rather than editing cross section control points in the viewports to "bulge" the mesh, you may prefer to open the Bulge Editor and move control points in the Bulge Editor's Cross Section view.

## Procedures

### To create a new bulge angle on a selected link

1.  With Bulge sub-object active, click Link in the Selection Level area.
2. Select a link in the viewports.

3.  Click Insert Bulge Angle.  
This creates a new bulge angle. The name in the Current Bulge Angle field increments. Give the new bulge angle a descriptive name.
4. In the Current Bulge Angle field, type a name, such as "Arm at 90".
5. Click the Bulge Angle Color swatch and choose a color in the color dialog.  
This makes identification much simpler. Now a joint angle must be set for the newly created bulge angle.
6. Exit Bulge sub-object level.
7. Select and rotate the bone at the joint where you want the bulge. Select the mesh again and return to Bulge sub-object level.
8.  Click Set Bulge Angle. The actual joint angle between the link and its child is recorded (for example, the angle created by the biceps and forearm is the actual joint angle, if the biped upper arm link is selected).
9. Click Cross Section and use Insert to create and position a cross section, if one does not exist where the mesh should bulge.
10. Scale a cross section or move control points on a cross section to create the bulge: click Cross Section or Control Point in the Selection Level area, depending on how you want to deform the mesh.
11. Enter **80** in the Influence field.

If the bulge angle is for a biceps bulge when the forearm and upper arm create a 90-degree angle, the bulge begins to appear at 10 degrees.

Note: By default one bulge angle is created by Physique in the initial skeletal pose; the arm is usually straight in this pose.

### To copy all Bulge angles for one link to it's opposite

1. Turn on Bulge sub-object.
2. Select a Link to copy in the viewports.
3. Choose Entire Link in the Current Bulge Angle dropdown.
4. Click Copy.
5. Select the opposite link in the viewports.
6. Click Paste, and then click Mirror.  
All the bulge angles from the first link are pasted to the opposite link.

### To choose a specific bulge angle for editing

1. Click the arrow by the Bulge Angle dropdown.  
The full list of bulge angle names appears for the current link.
2. Click the name of the bulge angle you want to edit.  
The viewports update to show the cross sections associated with the newly selected bulge angle.

### To change a bulge angle value

1. In the Bulge rollout, make sure the bulge angle's name is displayed in the Current Bulge Angle dropdown.
2. In a viewport, scrub to another frame displaying the angle you want to record, or use the 3DS MAX transforms to change the angle between the active link and its child link.
3. In the Bulge rollout, click Set Bulge Angle.

### To delete a bulge angle

1. Click the downward pointing arrow of the Bulge Angle dropdown.

The full list of bulge angle names is displayed.

2. Click the name of the bulge angle you want to delete.



3. Click Delete Bulge Angle.

You can't delete the default bulge angle—a link must always have at least one bulge angle defined.

### To use Select Nearest Bulge Angle

The Select Nearest Bulge Angle button can be useful to help you identify the bulge angle that has the greatest effect at the current pose.

1. Click Select Nearest Bulge Angle.
2. Scrub the time slider or rotate the joint.  
The bulge angle named in the dropdown list changes to the most influential bulge.

## Interface



### Selection Level group

The Insert, Delete, Copy, Paste, and Mirror commands work according to what is turned on in the Selection Level area.



**Link.** Click to select links in the viewports. The Copy, Paste, and Mirror commands are enabled for links.



**Cross Section.** Edit cross sections to “bulge” the mesh. Click to select cross sections in the selected link in the viewports. The Insert, Delete, Copy, Paste, and Mirror commands are enabled for cross sections.



**Control Point.** Click to edit cross section control points on the selected link.



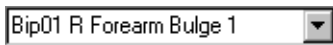
**Next/Previous Link.** Selects the next or previous link, cross section, or control point depending on the selection level.



**Bulge Editor.** Create and edit bulge angles using the Cross Section and Profile views in the Bulge Editor. Displays the *Bulge Editor* (see page 318).



**Select Nearest Bulge Angle.** Turn on to select the bulge angle nearest to the current joint angle. If a joint bends over time, you can use the time slider to select a bulge angle. If you click Play, you can see the Current Bulge Angle field change to reflect the nearest bulge angle to the current angle of the selected link and its child (if two or more bulge angles for the limb exist).

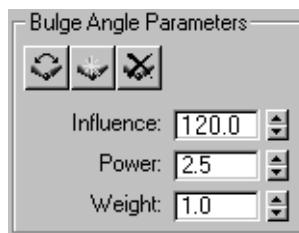


**Current Bulge Angle (field).** Displays the current Bulge Angle. Type a descriptive name for a newly created bulge angle. When you click Insert Bulge Angle, this field displays the new bulge angle name. Use the dropdown to select other bulge angles.

**Entire Link.** in the Current Bulge Angle dropdown selects all cross sections for all bulge

angles in the selected link. Use this to change cross section parameters globally for a link. This also affects the Copy and Paste commands. While Entire Link is selected, Copy Cross Section copies all cross sections for all bulges, and Paste Cross Section pastes all cross sections for all bulges. Entire Link is useful for copying all the bulges from one arm or leg to another.

**Bulge Angle Color.** Changes the current bulge angle color. Giving each bulge angle a different color is a good way to distinguish between them.



**Set Bulge Angle.** Changes the angle value to the skeleton's current joint angle on the current bulge angle (visible in the Current Bulge Angle field). First use Move or Rotate in a viewport to change the angle between the selected link and its child, then click Set Bulge Angle.



**Insert Bulge Angle.** Adds a new bulge angle for the selected link. The Current Bulge Angle field displays a bulge angle name with the number incremented. Type a descriptive name for any new bulge angle, such as “Arm at 90”.

By default, one Bulge Angle is created in the Figure mode pose when Physique creates links. Only one additional bulge angle will allow you to bulge the mesh. You can create more bulge angles for further control, if you want.



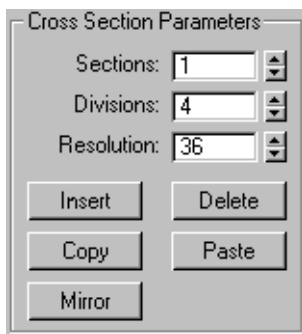
**Delete Bulge Angle.** Removes the current bulge angle at the selected link.

**Influence.** The range of angles through which the bulge influences the skin. Range =0 to 180; Default=90 degrees.

For example, if you've set a bulge angle for the joint at 90 degrees, an Influence value of 40 means that the bulge effect begins to appear when the rotating joint reaches 50 degrees (90-40) or 130 degrees (90+40).

**Power.** Controls how smoothly or abruptly the bulge takes effect. At 0, the bulge takes effect immediately, without interpolated easing. As values increase, the bulge eases in gradually. A value of 10 will bulge the mesh abruptly when the set angle is reached. Range=0 to 10; Default=2.5.

**Weight.** Increases the effect of the current bulge angle relative to the effect of any other bulges. Range=0 to 100; Default=1.0.



**Sections.** Sets the number of cross sections for the selected link.

**Divisions.** Sets the number of control points evenly distributed around the cross section.

**Resolution.** Sets the number of radial divisions around the cross section. Control points snap to the nearest resolution line.

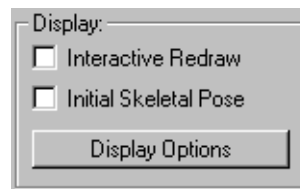
**Insert.** Creates a new cross section or control point; depends on what is on in the Selection Level area.

**Delete.** Deletes a cross section or control point.

**Copy.** Copies bulge angles and cross sections, depending on which is active in the Selection Level area.

**Paste.** Pastes links and cross sections, depending on which is active in the Selection Level area. To copy and paste all the bulge angles from one link, select Entire Link in the Current Bulge Angle field, and then click Copy. Select the opposite link and click Paste. Click Mirror to mirror the pasted link parameters.

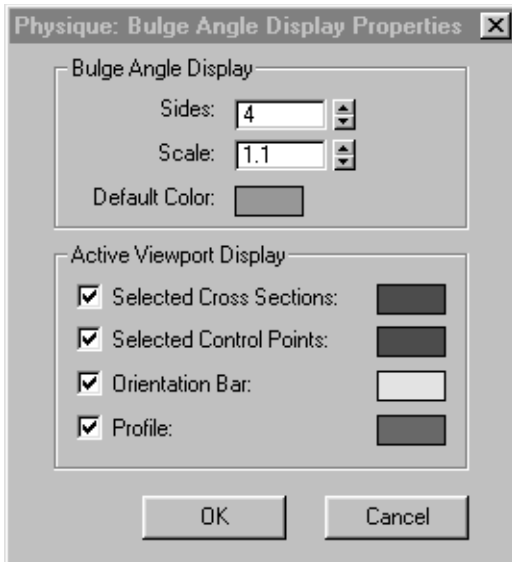
**Mirror.** Mirrors bulge angles and cross sections on bulge angles. Use Select and Rotate, then click and drag over a link or cross section to orient bulge angles and cross sections, if necessary.



**Interactive Redraw.** Select to deform the mesh in real time as you edit cross sections and control points in the viewports. Clear to update the mesh on mouse-up. Clear Interactive Redraw on slow systems or with dense meshes that take too long to compute in real time.

**Initial Skeletal Pose.** Put the mesh into its original pose when Physique was first applied.

**Display Options.** Sets viewport display options.



## Envelope

**Sides.** Specifies the number of sides the bulge angle envelope displays in the viewports.

**Scale.** Specifies the scale of envelope display in the viewports.

**Default Color.** Changes envelope color in the viewport display.

## Active Viewport Display

**Selected Cross Sections.** Changes the color of selected cross sections in the viewports.

**Selected Control Points.** Changes the color of selected Control Points in the viewports.

**Orientation Bar.** Changes the color of the Orientation Bar in the viewports.

The Orientation Bar displays in the viewports when the Bulge Editor is open.

**Profile.** Changes the color of the bulge profile.

The bulge profile displays in the viewports when the Bulge Editor is open.

## Bulge Editor

Select an object with the Physique modifier. > Modify panel > Physique rollout > Bulge Editor




The Bulge Editor is an alternative to using Bulge Sub-Object to create and edit bulge angles. The difference is that the Bulge Editor allows you to create, select, and edit cross sections in a Cross Section and Profile view. Creating and editing cross sections allows you to “bulge” the mesh. Using *Bulge Sub-Object* (see page 313), creating and editing bulge angles will take place in the viewports. However, all of the parameter changes you make in the Bulge Editor, are also reflected on the mesh in the viewports.

On the Modify panel, you access the Bulge Editor from either the Physique rollout or from the Bulge rollout when Bulge Sub-Object is active. The Bulge Editor works exclusively with bulge angles. Tendons are created using *Tendons Sub-Object* (see page 326) and are edited in the viewports.

## Procedures


**Tip:** For easier creation of bulge angles, you should create a simple animation that moves the limb into extreme orientations. In the case of the human arm, you might set keyframes with the arms down against the body, extended to the sides, bent at the elbows, and finally more relaxed with the hands touching the shoulders. By scrubbing the time slider, you can easily choose one of many intermediate or extreme orientations. This will save time when creating Bulge Angles because you won’t need to exit the Physique sub-object level to change the orientation of the mesh. You can create and set multiple Bulge Angles without ever leaving the Bulge Editor or Bulge Sub-object level.

### To create a new bulge angle using the Bulge Editor

1.  Click the Bulge Editor button at the top level of the Physique modifier, or in Link Sub-Object, or in Bulge Sub-Object to display the Bulge Editor.  
Being in Bulge sub-object level provides the added benefit of letting you work in either the Bulge Editor or Bulge Sub-Object interchangeably.
2. Select a link in the viewports.
3.  Click Insert Bulge Angle on the toolbar.  
Physique creates a new bulge angle. The number of the bulge angle name in the Current Bulge Angle field increments.
4. Type a descriptive name in the Current Bulge Angle field.  
It is a good idea to change the color of the newly created bulge angle. Do this by clicking Bulge Angle Color in Bulge Sub-Object and selecting a color in the Bulge Angle Color dialog.
5.  Click Insert CS Slice on the Bulge Editor toolbar and click in the Profile view to create and position a cross section.  
Do this if no appropriate cross sections exist.
6. Rotate the joint to the desired angle in the viewports.  
You will have to exit Bulge Sub-Object, select the appropriate biped limb and rotate it. For biceps, rotate the biped forearm to ninety degrees, for example. If an animation is loaded, rather than exiting Bulge Sub-Object, simply scrub the time slider to a frame that has the limbs at an appropriate angle.


7. Edit cross section control points in the Bulge Editor Cross Section view to “bulge” the mesh.

As control points are scaled or moved in the Cross Section view, the mesh “bulges” in the viewports.

8.  Click Set Bulge Angle on the Bulge Editor toolbar. This records the current angle of the joint.

When the joint angle is reached, the mesh bulges. By default, Physique creates one bulge angle when Physique is first used to attach a mesh to the biped in Figure mode. So, you only need to create and set one additional bulge angle to have an arm or leg that bulges when the biped joints are rotated.

### To change a bulge angle value

1. In the Bulge Editor, make sure the bulge angle you want to reset is displayed in the Current Bulge Angle field.
2. In a viewport, change the angle between the active link and its child link.  
Rotate a biped limb or use the time slider to move to a frame that has the limb rotated correctly.
3.  In the Bulge Editor, click Set Bulge Angle.  
The angle of the joint is set for the displayed bulge angle. When the biped limb rotates to this angle, the mesh bulge effect is at full strength.

### To choose a specific bulge angle for editing

1. Click the downward pointing arrow on the Current Bulge Angle dropdown.  
The full list of bulge angle names appears.

- Click the name of the bulge angle you want to edit.

The Bulge Editor's Cross Section and Profile views update to show the cross sections associated with the newly selected bulge angle.

#### To delete a bulge angle

- Click the downward pointing arrow of the Bulge Angle dropdown.  
The full list of bulge angle names appears.
- Click the name of the bulge angle you want to delete.



- Click Delete Bulge Angle.

You can't delete the default bulge angle; a link must always have at least one bulge angle defined.


#### To add a cross section

- In the Bulge Editor, make sure that the bulge angle you want to edit is displayed in the Current Bulge Angle field.



- Turn on Insert CS Slice.
- Click the link in the Profile view at the point where you want the cross section to be.  
You can also add cross sections to the child link.

#### To delete a cross section

- Select a cross section in the Profile view.  
The selected cross section turns red.
-  Click Delete CS Slice.  
You cannot delete the default cross section at the joint between the link and its child.

#### To make a cross section the active cross section

- Select a cross section in Profile View.

The selected cross section displays in Cross Section view. Change its shape by selecting and editing control points in Cross Section view. This will bulge the mesh at the location of the cross section.

#### To select multiple cross sections, do one of the following

Change parameters for multiple cross sections by selecting them and changing the influence and power parameters for example.



- Turn on Select and Translate CS.
- Select a cross section in the Profile view.
- Use CTRL+click to add cross sections to the selection.

Or, click Select and Translate CS then drag a rectangular region in the Profile view. All cross sections that the region surrounds or crosses are selected.

ALT+click removes a cross section from the selection.

#### To move cross sections along the link

Move a cross section to reposition where on the mesh a bulge occurs.

- Select the cross sections to move.



- Click Select and Translate CS on the toolbar.



If you are moving a single cross section, you can skip step 1 and click the cross section after you click Select and Translate CS Slice.

- Drag left or right in the Profile view to move the cross section (or cross sections) left or right.

You cannot move cross sections through each other.



### To copy and paste cross sections

1. Select the cross section to copy.
2.  Click Copy Selected CS.
3. Select a cross section to paste over.
4.  Click Paste Selected CS.

The cross section you pasted now has the shape of the cross section you copied.

Note: You can copy a cross section and paste it to a similar cross section on another link. Create a bulge angle and an appropriate cross section on the target link before pasting. Use Mirror selected CS after pasting if necessary.

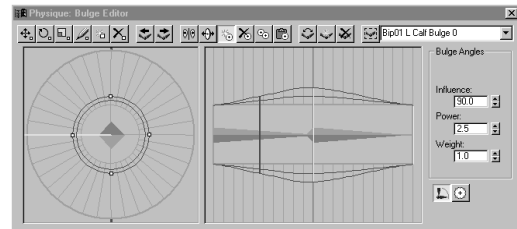
### To change the shape of a cross section

1. Select a cross section in the Profile view.
2. Use the selection and control point tools to move, scale, or rotate control points on the cross section in the Cross Section view.

### To change the Profile view orientation

1. In the Cross Section view, move the cursor to the outer end of the orientation bar.  
The cursor changes to a shape like the Cross Section view.
2. Drag to rotate the orientation bar.  
When you release the mouse, the Profile view will automatically update.

## Interface



## Toolbar



**Select Scale Rotate Control Points.** Select, scale, and rotate selected control points of a cross section in the Cross Section view window. This is the same as using the MOVE tool in the 3DS MAX viewports. Control points are simultaneously scaled and rotated around the link which in effect “moves” them.

Select a cross section in the Profile view first. Then select and move control points in the Cross Section view to deform the mesh.



**Select and Rotate Ctrl Pts.** Select and rotate control points in and around the center of the link in Cross Section view.

This does not twist the mesh, if the cross section is not a circle, a bulge migrates around the mesh.



**Select and Scale Ctrl Pts.** Select and scale control points about the link center in the Cross Section view.

If all cross section control points are scaled, the mesh bulges uniformly.



**Draw Ctrl Pts.** Draw a cross section in the cross section window. The mesh bulges based on the shape you draw.

In the Cross Section view, you can draw the cross section interactively, placing control points at each resolution step.

The number of control points created depends on the cross section resolution. You can freeze existing control points by selecting them in the Cross Section view and holding down the CTRL key as you draw.

Note: In the Profile view, you can draw the profile which places a control point on each cross section in the bulge angle. This is most useful when you have multiple cross sections in the bulge angle. You can rotate the orientation bar to draw whichever profile you desire. (Imagine the Cross Section view being like drawing around the arm, and the Profile view like drawing along the length of the arm.)



**Insert Ctrl Pts.** Insert a single control point on a cross section in the cross section window.

Add and position control points to sculpt the cross section and control exactly where a bulge occurs on the mesh.



**Delete Ctrl Pts.** Delete control points. First select the control points in the Cross Section view, and then click Delete Ctrl Pts.



**Previous Link/Next Link.** Move to the next or previous link in the hierarchy. The Cross Section and Profile views update to display the appropriate cross sections.

The Profile view displays the parent link on the left and the child link on the right.



**Mirror Selected CS.** Mirror the cross section across the vertical plane running between the green profile reference points at the top and bottom of the Cross Section View.

The mesh bulges on the opposite side of the link.



**Select and Translate CS.** Select and move a cross section along the link.

The bulge on the mesh migrates up and down the link as the cross section is moved.



**Insert Cross Section Slice.** Click this button and then click in the Profile view to create and locate a new cross section on either the parent or child link.

Extra cross sections give you more control of how and where the mesh bulges.



**Delete Cross Section Slice.** Deletes the currently selected cross section.

Select a cross section in the Profile view then click Delete Cross Section Slice.



**Copy Selected CS.** Copies the selected cross section.

You can copy a cross section from one link to another link. Create a new bulge angle with an appropriate cross section in the target link first, then copy the cross section parameters from the source cross section, select the target cross section and click Paste Selected CS.



**Paste Selected CS.** Pastes copied cross section parameters onto another cross section.

First select a cross section in the Profile View and click Copy. Then select another cross section in the Profile View and click Paste.



**Set Bulge Angle.** Sets the effect of the current Bulge Angle to the skeleton's current joint angle.

First set the joint to the orientation at which you want the bulge, then click this button. During an animation, whenever the joint rotates near this angle, the mesh will bulge.



**Insert Bulge Angle.** Adds a new bulge angle for the selected link.

The Current Bulge Angle field displays a bulge angle name with the number incremented.

Type in a descriptive name for any new bulge angle, such as “Arm at 90”.

By default, one Bulge Angle is created in the Figure mode pose when Physique creates links. Only one additional bulge angle will allow you to bulge the mesh. You can create more bulge angles for further control, if you like.



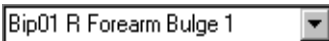
**Delete Bulge Angle.** Deletes the current bulge angle. The current bulge angle displays in the Current Bulge Angle field.

First select a bulge angle in the Current Bulge Angle dropdown, and then click Delete Bulge Angle.



**Select Nearest Bulge Angle.** Turn on to select the bulge angle nearest to the current joint angle.

If a joint bends over time, you can use the Time slider to select a bulge angle. If you click Play, you can see the Current Bulge Angle field change to reflect the nearest bulge angle to the current angle of the selected link and its child (If two or more bulge angles for the limb exist).



**Current Bulge Angle (field and dropdown).** Displays the current Bulge Angle. Select a bulge angle in the dropdown for editing or renaming.

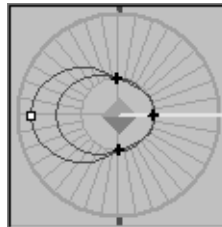
Type a descriptive name for a newly created bulge angle.

When a new bulge angle is created with the Insert Bulge Angle button, this field displays the new bulge angle name. Use the dropdown to select other bulge angles.

## Cross Section View

The Cross Section view displays an outline of the active cross section. In the Profile view the active cross section is a bright red color. It is possible to select multiple cross sections in the Profile view using either a window selection or CTRL+click to add cross sections. This allows you to dial in cross section parameters for all of them at once. Only one “active” cross section, however, can be viewed and edited in the Cross Section view. Selected but inactive cross sections display in a dark red color in the Cross Section and Profile views.

Edit cross sections to bulge the mesh.



**Gray Square.** Represents the link at the center of the cross section.

**Control Points.** Display as small black crosses on the control spline, or as black squares with a white center, when they are selected.

**Resolution Lines.** Display as gray lines that surround the link radially. Control points “snap” to these lines as they are positioned.

Resolution can be increased using the Resolution parameter on the Bulge rollout in

**Bulge Sub-Object.** The Cross Section view, however, always displays 36 resolution lines.

**Red Line.** The control spline for the selected cross section.

The shape of the spline determines where mesh deformation occurs.

**Green Line.** Represents mesh deformation.

This updates in Play mode according to the angle of the limb and the bulge angle parameters.

**Orientation Bar.** Select the end of this bar in the Cross Section view and rotate it to change the orientation of the section and profile views.

This yellow bar is a visual reference at a fixed location in the viewports. You can determine the orientation of the Cross Section view by comparing the placement of the yellow bar.

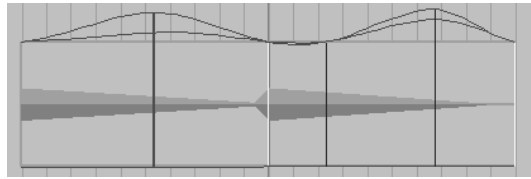
## Profile View

Use Profile View to select, move, and copy cross sections on the selected link and its child. Select a cross section in Profile View to display and edit a cross section in Cross Section view. Window select or use CTRL+click to select multiple cross sections. Although only one cross section at a time is active in Cross Section view, you can select multiple cross sections and change parameters for all of them at the same time. Insert a new cross section by clicking Create CS Slice and clicking on the Profile View to position it.

The Profile View is a schematic profile of two links. The currently selected link is on the left, and its child link is on the right. If the selected link is an end link, the outline of the right half of the Profile view turns gray.

Cross sections are shown as vertical bars across the profile. The active cross section is red. Unselected cross sections are white. Cross

sections that are selected but not active are dark red.



The control spline is red and the deformation spline is green. This is similar to Cross Section view.

## Draw in Profile View

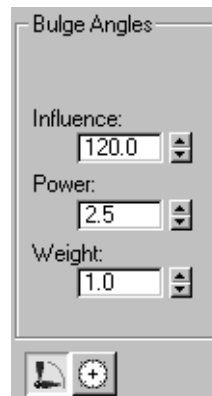
Use the Draw tool in Profile view to change the control spline for the selected link and its child. This changes the cross section shapes as you “draw” in the Profile view.

## Profile View Orientation

Use the Cross Section view orientation bar to change the orientation of the Profile view.

## Bulge Angle parameters

Use the Bulge panel to change bulge parameters for the currently active bulge angle.



Set parameters for bulge angles



**Bulge Angles.** Displays bulge angle parameters in the panel.

**Influence.** The range of angles through which the bulge influences the skin. Range=0 to 180; Default=90 degrees.

For example, if you've set a bulge angle for the joint at 90 degrees, an Influence value of 40 means that the bulge effect begins to appear when the rotating joint reaches 50 degrees (90-40) or 130 degrees (90+40).

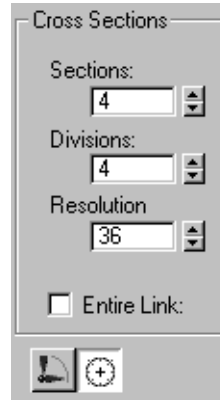
**Power.** Controls how smoothly or abruptly the bulge takes effect.

At 0 the bulge takes effect immediately, without interpolated easing. As values increase, the bulge eases in gradually. A value of 10 will bulge the mesh abruptly when the set angle is reached. Range=0 to 10; Default=2.5.

**Weight.** Increases the effect of the current bulge angle relative to the effect of any other bulges. Range=0 to 100; Default=1.0.

### Cross Section parameters

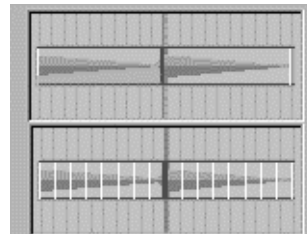
Use the Cross Section panel to change the cross section settings for a particular link. The Bulge Editor and Bulge sub-object are synchronized so you can work in either or both and get the same results. Value change in the Bulge Editor are reflected in Bulge sub-object and vice versa.



**Cross Sections.** Displays cross section parameters in the panel.

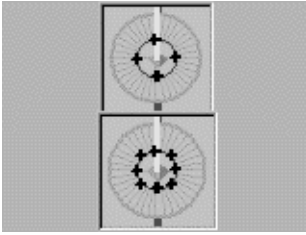
**Sections.** Sets the number of cross sections for the selected link.

Rather than manually inserting cross sections in the Profile view, use Sections to create cross sections. For a thigh or biceps bulge, you may only need one cross section in the middle of the link.



Sections=1 above and Sections=8 below

**Divisions.** Sets the number of control points evenly distributed around the cross section control spline.



**Divisions=4 above and Divisions=8 below**

**Resolution.** Sets the number of radial divisions around the cross section. Control points snap to the nearest resolution line.

**Entire Link.** Selects all cross sections for all bulge angles in the selected link. Use this to globally change parameters for all cross sections on a link. This also affects the Copy Selected CS and Paste Selected CS commands.

While Entire Link is selected, Copy Selected CS copies all cross sections for all bulges, and Paste Selected CS pastes all cross sections for all bulges. Entire Link is useful for copying all the bulges from one arm or leg to another.

Note: Selecting Entire Link here will also display “Entire Link” in the Current Bulge Angle name field on the Bulge rollout in Bulge sub-object.

## Tendons Sub-Object

Select an object with the Physique modifier. > Modify panel > Sub-Object Tendon



After adjusting envelope parameters for good mesh deformation, use tendons to control the amount of skin stretching across multiple links. While envelopes provide smooth skin deformations, tendons provide additional stretching in much the same way that actual human tendons might create pulling in the wrist (several joints away) when the fingers are moved.


## Workflow

To create a tendon from the spine to the upper arm, turn on Link sub-object and select a spine link. Click to turn on Insert and click the selected link to add tendon cross sections, then turn off Insert. Click Cross Section in the Selection Level area. Using the Rotate tool on the 3DS MAX toolbar, rotate a cross section so its control points are in useful locations. Use the Radius spinner in the Tendon Parameters area to scale the cross section radius so the control points fall close to the surface of the skin. Click Control Point, select a control point on the spine cross section, click Attach, and select the arm link. A tendon stretches from the spine to the arm. Finally, adjust Pull, Pinch, and Stretch to refine the movement of the skin.


## Procedures

### To create and attach a tendon

-  Activate Tendons sub-object, and click Link in the Selection Level area.
- Select a link in the viewports.
- Click Insert.
- Position the cursor over the link in the viewports.  
The cursor changes to a small star.
- Click to create cross sections on the link. Right-click to deactivate Insert; the Insert button changes color from green to gray.
-  Click Cross Section in the Selection Level area.
- Use the 3DS MAX Rotate tool to rotate the cross section so the control points are positioned in useful places.

8. Use the Radius spinner to scale the cross section radially so the control points fall close to the surface of the skin.
9.  Click Control Point in the Selection Level area, then select one or more control point(s) on a cross section in the viewports.
10. In the Edit Commands area, click Attach and select a link in the viewports.  
Tendons span the area between the links.
11. Change the Pinch, Pull, and Stretch parameters as necessary to adjust skin behavior.
12. Change Upper and Lower boundary parameters as necessary to control the extent of the effect on neighboring links.

#### To attach a tendon to another link

1.  Click Control Point to make it the active selection level.
2. In a viewport, select one or more of the tendon's attach points.  
Click a point to select it; use CTRL+click to add points to the selection, or drag a rectangular region to select multiple points.
3. Click to turn on Attach in the tendon Edit Controls area.
4. In a viewport, click the link to which you want to attach the tendon.

In the viewports, red lines appear showing the tendon's connection from its base to the other link. The Physique skin might also deform as a result of attaching the tendon.

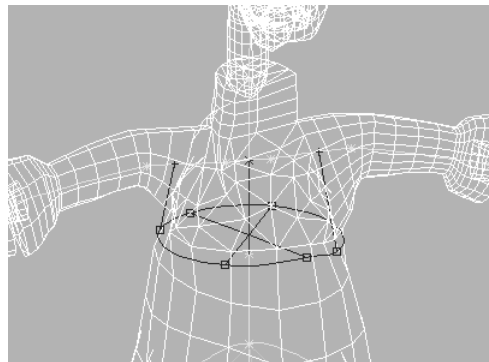
**Tip:** The tendon can have fixed attach points that are not connected to another link. These are useful for giving some rigidity to the skin, as when (in an actual body) a bone lies close to the skin's surface. For example, you might leave two fixed

attach points on either side of a character's chest area, to simulate the effect of the sternum. When all tendons are attached to other links, the skin over the base link can have a "squishy" appearance when its animated. This is appropriate for some animated characters, but not for others.

After you create a tendon and attach it, you can adjust it using the parameters in the Bulge Editor Tendon command panel and the boundary condition parameters in the Physique Tendons rollout.

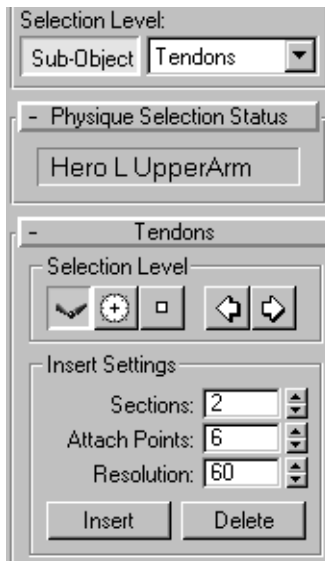
#### To delete a tendon

1. At the Link sub-object selection level, click the link that has the bad tendon.
2. Click Delete in the Physique Tendons rollout.



Tendons connecting upper chest to arms

## Interface



**Link.** Click to select a link.



**Cross Section.** Click to edit cross sections.



**Control Point.** Click to edit control points on cross sections.



**Previous/Next Link.** Go to the previous or next link, cross section, or control point, depending on the selection level.

**Sections.** Set the number of base cross sections for the tendon.

**Attach Points.** Set the number of Attach Points around the tendon's base cross section.

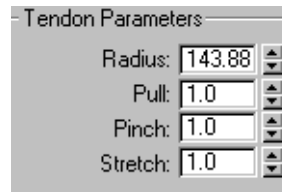
**Resolution.** Sets the radial resolution around the base cross section. Attach points are constrained to one of the radial resolution lines.

**Insert.** Click to insert new cross sections on the selected link, or to insert a control point on a

cross section. The action depends on the selection level: Link, Cross Section, or Control Point.

**Delete.** Click to delete cross sections or control points.

The action depends on the selection level: Link, Cross Section, or Control Point.



**Radius.** Scale Attach Points relative to the center of the cross section. Default=the average circumference of the skin where the tendon base is located.

**Pull.** Defines the strength of pull *along the length of the link*. Default=1.0.

**Pinch.** Defines the amount of pinch *around the link circumference of the tendon base*. Default=1.0.

**Stretch.** Defines the amount of stretch *toward the attached link*. Default=1.0.

The Pull, Pinch, and Stretch values work together to control, along a particular dimension, the strength of the one link (the attached link: call it link B) on vertices assigned to another link (the link where the tendon cross sections are inserted: call it link A). Vertices on link A behave as if they were under the influence of link B.

### Pull

Defines the strength of pull along the length of the link. Default=1.0.

### Pinch

Defines the amount of pinch around the link circumference of the tendon base. Default=1.0.



## Stretch

Defines the amount of stretch toward the attached link. Default=1.0.

Pull, Pinch, and Stretch can all range from -2.0 to 2.0. At the default value of 1.0, the tendon's effect is full strength. Reducing the value of these parameters to a value between 0.0 and 1.0 reduces the strength of the tendon's effect. At values below 0.0, the tendon deforms the skin in the opposite direction. Pull, Pinch, and Stretch values between 1.0 and 2.0 cause an exaggerated effect, which is not usually needed for realistic effects.

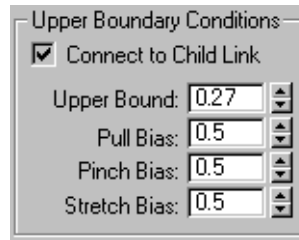
**Tip:** Setting Stretch less than 0.0 can be useful in regions where movement typically pulls the skin inwards, as in the collarbone area or the buttocks.

**Tip:** As a general rule, leave the tendon values at 1.0 when the tendon is attached to a nearby link. For example, leave them at 1 between the upper spine (at the chest) and the arms. Reduce the Pull, Pinch, and Stretch values to reduce the tendon's effect when the tendon is attached to a more distant link. For example, reduce them between a lower part of the spine and the arms.



**Attach.** To attach a point, select the point, click **Attach** to turn it on, then select a different link in the viewports.

**Detach.** Detach a tendon on the selected control point at the Control Point selection level. At the Cross Section selection level, all attached tendons on the selected cross section are detached. At the Link selection level, all attached tendons are detached on all cross sections of the selected link.

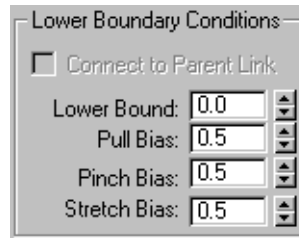


**Connect to Child Link.** Allow tendons to affect the child link. This allows you to connect tendons across several links. Otherwise, there is a boundary between the links where no tendon effect occurs. Default=off.

**Upper Bound.** Set the upper boundary overlap. Upper Boundary values greater than 1.0 effect the child link.

**Warning:** Tendons that span several links can conflict with joint intersection parameters. In these cases, clear Active to turn off the joint intersection parameters for the joints that the tendon spans.

**Pull Bias, Pinch Bias, Stretch Bias.** Set Upper Boundary falloff effect. Values of 0.0 have no effect. Increasing the value shifts the Pull, Pinch, or Stretch effect toward the child. Default=0.5.

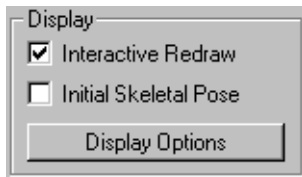


**Connect to Parent Link.** Allow tendons to affect the parent link. This allows you to connect tendons across several links. Otherwise, there is a boundary between the links where no tendon effect occurs. Default=off.

**Lower Bound.** Set the lower boundary overlap. Lower Boundary values less than 0.0 affect the parent link.

**Warning:** Tendons that span several links can conflict with joint intersection parameters. In these cases, clear Active to turn off the joint intersection parameters for the joints that the tendon spans.

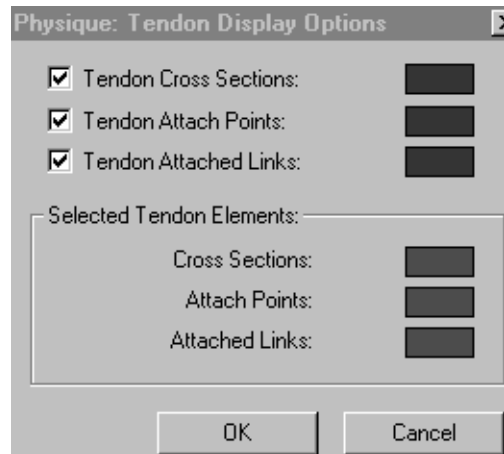
**Pull Bias, Pinch Bias, Stretch Bias.** Set Lower Boundary fall off effect. Values of 0.0 have no effect. Increasing the value shifts the Pull, Pinch, or Stretch effect toward the parent. Default=0.5.



**Interactive Redraw.** Select to deform the mesh in real time, as cross sections and control points are edited in the viewports. Clear to update the mesh on mouse-up. Clear Interactive Redraw on slow systems or with dense meshes that take too long to update.

**Initial Skeletal Pose.** Position the mesh to its pose when Physique was applied.

## Display Options



**Tendon Cross Sections.** Toggle viewport display and change color for tendon cross sections.

**Tendon Attach Points.** Toggle viewport display and change colors for tendon attach points.

**Tendon Attached Links.** Toggle viewport display and change colors for tendon attached links.

Selected Tendon Elements area

**Cross Sections.** Change viewport color of selected cross section.

**Attach Points.** Change viewport color of selected control point.

**Attached Links.** Change viewport color of selected link.

## Vertex Sub-Object

Select an object with the Physique modifier. > Modify panel > Sub-Object Vertex

Typically, you will want to adjust envelopes to correct the way skin behaves as the biped moves. You can, however, manually assign vertex properties to override envelopes. Remove the influence of inappropriate links from selected vertices for example. You can also change the weight distribution between links for a single vertex using Type in Weights.

### Procedures

#### To remove a links influence on vertices

An example of this would be to remove the influence of the index finger links from the vertices of the middle finger.

1. Turn on Vertex Sub-Object.
2. Turn on Select and select vertices in the viewports.
3. Turn on Remove from Link and select links in the viewports.
4. Turn on Lock Assignments.

The selected vertices are no longer influenced by the selected links. You must use Lock Assignments to ensure that these envelopes will not influence these vertices if envelope parameters are adjusted later.

#### To check vertex assignments

1. Click to turn on Select by Link.
2. Make sure all three Vertex Type buttons are on.
3. Click a link.

Physique displays the vertices assigned to that link. If any vertices assigned to the link

are out of place, reassign them to a different link, as described below.

4. Repeat the previous steps for other links in the skeleton.

#### To remove deformable vertices from a link's influence

1. Make sure all three Vertex Type buttons are on, and then click Select By Link. Observe the assignments and determine which vertices are incorrectly influenced. Also note if the vertices are red (deformable) or green (rigid.) Blue vertices do not fall within the influence of any link and are thus assigned to the root.
2. Click Select.
3. Select only the out of place vertices.  
Note: Alternatively, you can use ALT+click to deselect the vertices that are in the correct place.
4. Click to turn on Deformable, the left, red Vertex Type button. (The other two Vertex Type buttons turn off.)
5. Click Remove From Link.
6. Click each link that you want no longer to influence the selected deformable vertices.

#### To assign deformable blended vertices manually

Note: This procedure is analogous to the manual assignment method in **character studio 1.x**, except that envelopes determine assignment in **character studio 3**, whereas cylindrical capping regions were used in R1.x. This technique is considered an expert-level procedure to be used for specific manual reassignment tasks. Users should consider the generally more appropriate method of reinitializing with Vertex-Link Assignments checked. See *Reinitializing Physique Settings* (see page 105).

1. Verify that Deformable envelopes are active for each link to which you want to manually assign deformable blended vertices.

This can be done in Envelope sub-object level, by selecting each link and confirming that Deformable is selected and that each envelope is sized appropriately for the link.

2. Go to Vertex sub-object level and click Select.
3. Confirm that all three vertex type buttons are active and selected.
4. Choose all the vertices.
5. Select Remove From Link and select all the links by a window selection.  
All the vertices should turn blue.
6. Verify that the Red, Deformable vertex type is selected and choose Assign To Link.
7. Pick a link.  
Vertices falling within the link's envelope turn red, indicating successful assignment. Remaining vertices stay blue.
8. Use CTRL+click to select the next link.  
Unassigned (blue) vertices in the current selection are considered for assignment. Any vertices falling within the link's envelope will be assigned and turn red.
9. Repeat step 8 for each link, until all the vertices are manually assigned.

#### To override vertex assignments manually

This technique is used when envelope assignments are inappropriate and you want to have specific vertices influenced by a specific link.

1. Make sure all three Vertex Type buttons are on, and then select the vertices you want to reassign.
2. Click Assign To Link.

3. From the Vertex Type buttons choose the type of vertex assignment you want to use, deformable (red) or rigid (green.)

4. In the Blending Between Links list, choose No Blending.

This setting will disregard the effects of blending envelopes and vertex weights and allow you to assign the selection to any link manually.

5. Click the link you want to influence the vertex.
6. Click Lock Assignments to preserve the manual assignment.

#### To make vertices rigid

1. Click Select and select the vertices you want to make rigid.
2. Click to turn on Assign to Link.
3. Select Rigid Vertices (green cross) in the Vertex Type area.
4. Select No Blending in the Blending Between Links list.
5. Select the link these vertices are to be assigned to.
6. Click Lock Assignments.  
Small squares around the selected vertices indicate the vertices are locked.

When you assign deformable vertices, some vertices might turn blue; they are assigned to the root instead of the link you clicked. To assign these vertices as correctly deformable vertices, simply use CTRL+click with the neighboring parent or child link. If the blue vertices now turn red, they are deformable.

**Tip:** You don't have to work locally, one link at a time. You can use 3D Studio MAX multiple selection tools to select groups of vertices or groups of links, working with areas of the body and its skeleton all at once.

## Interface



**Physique Selection Status.** This text display shows “Select Vertices” when you are able to select vertex sub-objects directly. When you use Select by Link or Assign to Link, this display shows the name of the currently selected link, or “None” if no links are selected, or “Multiple Links Selected” when you’ve selected more than one link.

### Vertex Type

There are three vertex types, differentiated by color:

- **Red.** Deformable vertices that follow Physique’s deformation spline.
- **Green.** Rigid vertices that do not deform but just follow the link they are assigned to.
- **Blue.** Vertices attached to the root node. Physique uses this color when it isn’t sure which link to assign the vertices to. These vertices don’t deform but follow the center of mass object. You should reassign blue vertices as rigid or deformable.

### Blending Between Links

**N Links.** Vertices within all overlapping envelopes are influenced.

While this is selected in Vertex level, vertices may only be assigned to envelopes they fall within. Envelopes are used by Physique in this case to define the blended weight of each vertex.

**No Blending.** Vertices are influenced by only one link, as in **character studio 1**.

This is useful as a general default if you are developing characters for a game engine that doesn’t support blending. At vertex level it is useful to override blending weights defined by envelopes and assign vertices as either Rigid or Deformable, to any link. For example, after selecting vertices of the skull and face, turn on No Blending, turn on Rigid vertices, turn on Assign to Link and click the head link. This will override the head envelope vertex assignments and assign these vertices as rigid to the head link. Use Lock Assignments to lock any changes.

**2,3,4 Links.** The set number of links (closest links) influences Vertices. Game developers can choose the number of links that can influence vertices.

### Vertex Operations

**Select.** Select vertices using the selection tools on the 3DS MAX toolbar.

**Select by Link.** Select vertices by link.

**Assign to Link.** Assign the selected vertices to a link.

Turn on this button and then click a link to assign the currently selected vertices to that link. Assign to Link has a dependency on the envelopes defined for each link, and you must set Blending Between Links to “No Blending” to override this. After vertices are assigned manually in this way, you use Lock Vertices to keep them from being reassigned inadvertently with subsequent operations.

Note: You must select or assign a link by clicking the link; you can’t use the Select by Name dialog to select or assign a link while you use Physique to work with vertices.

**Remove from Link.** Remove the selected vertices from a link.

Note: Select, Select by Link, Assign to Link, and Remove from Link will act only on vertices specified by the red, green, and blue Vertex Type (“+”) icons above.

**Lock Assignments.** Lock vertex assignments.

Lock will prevent any changes from being made to the weights and blending presently assigned to the selected vertices. After ANY manual assignment of vertices, they should be locked. Before using Type-In Weights, vertices must be locked. When manipulating envelopes in difficult areas of the character, you might choose to lock vertices that are working well before changing the envelopes to affect other vertices.

Note: If additional data, such as more vertices, comes up the Modifier stack, Physique notes the proximity of new vertices to the locked vertices. It then *locks* the new vertices based on proximity. Already locked vertices will not be unlocked.

**Unlock Assignments.** Unlocks vertex assignments.

Click Unlock Assignments on selected locked vertices prior to reassigning them to another link, or changing their blending settings.

**Type-In Weights.** Displays the Type-In Weights dialog, which you use to enter a weight for selected locked vertices; only the weight for locked vertices can be changed.

To use this feature: select one or more vertices, then lock them, click this button to display the Type-In Weights dialog, select a link in the dropdown, and change the weight to that link to position the vertex (vertex updates in the viewports as the weight changes).

Note: you’ll primarily use Type-In Weights to correct flaws on low- to medium-resolution meshes. On a high-resolution mesh, adjusting envelopes should be used to correct deformation.

This can be used for stubborn vertices that fall in blended regions between two or more envelopes. Subtle adjustments can be made to the vertex weights of specific links that would be difficult to achieve by adjusting envelopes alone.

### Type in Weights dialog

**Link Name.** Use the dropdown to select different links and display vertex weight to that link.

When more than one vertex is selected, each is likely to have slightly different weight. Because of the lack of commonality, the weight field will be blank when displaying absolute weights. Entering a value will set all selected vertices to the same weight.

**Currently Assigned Links Only.** When selected, displays only those links that presently influence a vertex.

**All Links.** When selected, displays all available links.

**Weight.** Displays vertex weight for the link displayed in the dropdown. Use the spinner to change the value, if necessary. When more than one vertex is selected, each is likely to have slightly different weight. Because of the lack of commonality, the weight field will be blank when displaying absolute weights. Entering a value will set all selected vertices to the same weight.

**Absolute.** Use an absolute value for vertex weight.

**Normalized.** Feature not implemented in **character studio 3**.

**Relative Scale.** This is the mode of choice when working on multiple vertices. If a particular link is not having a strong enough effect on the selected vertices, you should lock them, open Type-In Weights, choose the Link and enter a relative scale. 1.0 would keep the weights the same, 2.0 would double the effect of the selected link on the selected vertices.

**Hide.** Hide selected vertices

**Unhide All.** Unhide all vertices.

**Initial Skeletal Pose.** Put the character into the initial pose it was in when the Physique modifier was applied, usually this is also Figure mode pose.

---

## Physique Initialization Dialog

Select an object with the Physique modifier. > Modify panel > Physique rollout > Attach to Node > Select node in viewports. > Physique Initialization dialog

Use the Physique Initialization dialog to specify link parameters and the type and size of envelopes to create for Physique links. This dialog will display under these two circumstances:

- When you apply Physique initially, using Attach to Node from the *Physique rollout* (see page 296) on the Modify panel.
- When you click the Reinitialize command on the Physique rollout.

## Creating Physique Links and Envelopes for the First Time

When Attach to Node is used to attach the mesh to the biped for the first time, use parameters on the Vertex-Link Assignment rollout to specify envelope parameters. In many cases, the default parameters on the Vertex-Link Assignment rollout work well as a starting point for envelope type, envelope sizing, and blending between links.

You normally want deformable envelopes. Blending Between Links is set to N Links (a vertex can be influenced by all envelopes that overlap it). The Object Bounding Box option is selected by default, and will size the envelopes based on the size of the biped limbs. In the case of a 3DS MAX bones hierarchy, you can use the *Box Generator utility* (see page 294) to create bounding boxes around the bones; in this case envelopes are sized to the bounding boxes.

Note: If Object Bounding Box is selected, but a bone does not have a connected object that

defines a bounding box, Physique will automatically use the link length for that link to determine the default envelope size. The envelope radial scale is set to 1/3 of the link length in this case. The only case you should *not* use the Object Bounding Box option is if you specifically want to ignore the size of any bounding boxes on the skeleton.

### Link Settings, Joint Intersections, and Cross Section Rollouts

Controls on these rollouts match those found in *bulge sub-object* (see page 313) and *link sub-object* (see page 307) and are used here to set default values.

Unless you want to change default settings for all link parameters and bulges on Initialization or Reinitialization, you do not need to change parameters on the Link Settings, Joint Intersections, and Cross Section rollouts in this dialog.

## Interface



### Initialization group

The Initialization check boxes are normally unavailable and not changeable during initialization. Although these options are unavailable, they are all turned on by default (except Include New Bones) for initialization, since ALL these settings are new when Attach to Node is used for the first time to link a mesh to the biped or 3DS MAX bones hierarchy.

These check boxes become available when re-attaching to the same skeleton or when the *Reinitialize* (see page 338) command is used on the Physique rollout. During re-attachment, the first two and lower two are selected and unavailable, and the middle three are available to set if desired. During reinitialization, all the controls are available.

**Initial Skeleton Pose.** Use this to change the biped position relative to the original mesh position.



**Include New Bones.** Creates new links for any new object (bone) linked to the biped.

**Link and Joint Settings.** Resets link parameters and joint intersection parameters to their default values.

**Bulges.** Reset bulge angles to the default values.

**Tendons.** Reset tendons to the default values.

**Vertex-Link Assignments.** This essentially re-evaluates which vertices fall within each envelope. (This may vary from the original default settings depending on whether changes have been made to the envelopes.) It also removes any custom vertex assignments.

**Vertex Settings.** Recalculates vertex parameters at the link each vertex is assigned to. This does not reassign manually reassigned vertices.

### Vertex-Link Assignment Rollout

On the Vertex-Link Assignment rollout you can choose to create new envelopes and pick the type of blending between links.

**Deformable.** Create Deformable Envelopes. It's on by default.

Deformable envelopes determine vertex-link assignment based on the deformation spline that Physique creates through the links.

**Rigid.** Create Rigid Envelopes for Physique links.

Rigid Envelopes determine vertex-link assignment based upon the linear 3DS MAX links in the hierarchy.

**Blending Between Links.** Normally leave this at N Links.

**N Links.** Vertices within all overlapping envelopes are influenced.

**No Blending.** Vertices are influenced by only one link, as in **character studio 1**. This allows a mesh with the Physique modifier applied in **character studio 1** to work with **character studio 3**.

Use No Blending if you are developing characters for a game engine that doesn't support blending or if you intend to use strictly **character studio 1**- style vertex-link assignments.

**2,3,4 Links.** Limit the number of links that can influence a vertex. These options are mainly for game engine restrictions.

### Radial Falloff Envelopes

**Create Envelopes.** Create envelopes based on parameters set in the Radial Falloff Envelopes area. On by default.

**Object Bounding Box.** Size the envelopes based on the size of the biped limbs or the bounding boxes of any objects in the 3DS MAX hierarchy. If you are attaching to a 3DS MAX bones hierarchy, you can use the *Box Generator utility* (see page 294) to create box objects around bones. These boxes are then used by Physique to size the envelopes.

**Link Length.** Size envelope radial scale to 1/3 of link length.

**Overlap.** Sets how far the envelopes overlap. Default=0.1.

**Smooth.** Sets the distance between the Inner and Outer bounds of an envelope by scaling the outer boundary. Range=0 to 10; Default=.75.

**Falloff.** Sets the rate of falloff between the inner and outer boundary of an envelope. This is a Bezier function. Range=0 to 10; Default=0.5.

### Link Settings rollout

The Link Settings rollout contains the default link values that will be assigned to all links. See *Link Sub-Object* (see page 307) for a command reference.

## Joint Intersections rollout

The Joint Intersections rollout contains default joint intersection values will be assigned to all links. See *Link Sub-Object* (see page 307) for a command reference.

## Cross Sections rollout

The Cross Sections rollout sets the default Bulge Angle parameter settings for new bulges. See *Bulge Sub-Object* (see page 313) for a command reference.

## Reinitialize Physique

Select an object with the Physique modifier. > Modify panel > Physique rollout > Reinitialize

The Reinitialize command on the Physique rollout is used mainly for resetting a particular attribute of your character's skin behavior. Among common uses are resetting bulges, deleting tendons, reassigning vertex-link assignments, or repositioning the biped relative to the attached mesh. Essentially, it can be used to reset any single item in the Initialization area, or all items.

Check boxes in the Initialization area of the Physique Initialization dialog become available when reinitializing using the Reinitialize command or for reattaching (using Attach to Node on the same skeleton). During reattachment, the first two and lower two are checked and unavailable, and the middle three are available. Reattaching can be used to select a new root node for a mesh that is already attached to a skeleton with Physique. During reinitialization, all the controls are available.

## Interface



## Initialization group

**Initial Skeleton Pose.** Uses the current biped position as the default pose for defining how the skeleton fits inside the skin. The current pose becomes the new initial pose, replacing the pose used when you first attached the mesh to the biped. This does not reassign manually reassigned vertices unless you also select Vertex-Link Assignments in the Initialization area.

Use Initial Skeleton Pose if you want to reposition the biped relative to the mesh in Figure mode, to reposition the biped shoulder joints for example. It is helpful first to turn off the Physique modifier, using the Active/Inactive Modifier toggle. This lets you scale and rotate the biped limbs in Figure mode, independent of the mesh skin. Use this after scaling a character. See the procedure below on scaling a biped with a mesh attached.

**Include New Bones.** Creates new links for any new object (bone) linked to the biped. Use this to add a 3DS MAX Bone to a mesh with the Physique modifier applied.

**Link and Joint Settings.** Resets link parameters and joint intersection parameters to their default values. Link parameters include Bend, Twist, Sliding, and Radial Scale values. For Radial Scale, this includes Scale, Amplitude, Stretch, and Breath.

**Bulges.** Reset bulge angles to the default values. One bulge angle is created per link by default. This will delete any new bulge angles you have created.

**Tendons.** Reset tendons to the default values. Deletes any tendons you have created.

**Vertex-Link Assignments.** Re-evaluates which vertices fall within each envelope. (This may vary from the original default settings, depending on whether changes have been made to the envelopes.) It also removes any custom vertex assignments. Use this if you want to replace all manual vertex reassignments with the default vertex assignments.

**Vertex Settings.** Recalculates vertex parameters at the link to which each vertex is assigned. This does *not* reassign manually reassigned vertices. Use this when you have changed the link parameters and want to recompute the vertex parameters based on these changes. This check box is automatically selected if either Initial Skeleton Pose or Vertex-Link Assignments is selected.

## Default Settings rollouts

You can change values in these rollouts to apply new default settings, when paired with the appropriate selection in the Initialization area.

**Link Settings Rollout.** Contains the default values that will be assigned to all links if Link

and Joint Settings is selected in the Initialization area. These settings are explained in the *Link Sub-Object* (see page 307) topic.

**Joint Intersections Rollout.** Contains the default joint intersection values that will be used if Link and Joint Settings is selected in the Initialization area. These settings are explained in *Link Sub-Object* (see page 307).


**Cross Sections Rollout.** Sets the default Bulge Angle parameter settings for new bulges if Bulges is selected in the Initialization area. These settings duplicate ones explained in *Bulge Sub-Object* (see page 313) and *Bulge Editor* (see page 318).

**Vertex-Link Assignment Rollout.** Sets the default Blending Envelope parameter settings if Vertex-Link Assignment is selected in the Initialization area. See *Physique Initialization Dialog* (see page 335).

---

## Scaling a Character

### To scale a biped and mesh attached with Physique

1.  Turn on Figure mode.
2. Change the biped height on the Structure rollout.

The biped and mesh scale together.

### To reinitialize a scaled mesh

Reinitializing a scaled mesh may be necessary if you use Initial Skeleton Pose in Physique. In this case the mesh appears at its size before scaling. To correct for this perform the following steps:

1. Disable Physique or drop down in the modifier stack below Physique.
2. Scale and move the mesh until it matches the biped or bones.

3. Re-enable or go back to Physique. The mesh will get large because it is doubly scaled.
4. Reinitialize with Initial Skeleton Pose. The mesh will shrink to the size you set during scaling.

---

## Physique and Free Form Deformations (FFDs)

Physique can be applied to a Free Form Deformation object (FFD), which in turn can animate a mesh that is bound to the FFD. For example, you could use this technique to animate a credit card or a box of cereal.

### Procedures

#### To apply Physique to an FFD to animate the entire mesh

1. Place a FFD (Box) space warp around the mesh to deform. The box should be large enough to encompass the mesh. The number of control points you use for the FFD can be fine-tuned later by going back down in the stack to the FFD to adjust the number of control points.
2. Select the FFD and apply the Physique modifier.
3. Click Attach to Root and choose the biped pelvis or the root node of your bone structure.
4. Select the FFD space warp and adjust the vertex assignments of the control points so they are assigned to the proper links if necessary. This can be done in the same way as done using Physique with a mesh, only there are fewer assignments to deal with. Although fewer assignments provide smoother surface deformation with the FFD, control points and their link assignments must be thoughtfully placed.
5. Use Bind To Space Warp on the 3DS MAX toolbar to bind the mesh to the FFD space warp.
6. Link the mesh to the biped pelvis, or root node of the bone structure, so it follows the skeleton and FFD as they move around the scene.

#### To use an FFD to complement the effects of Physique on a portion of a character mesh

Note: Use this procedure to animate clothes or amorphous shapes.

1. Place a FFD (Box) space warp around the mesh to deform. The box should be large enough to encompass the mesh. The number of control points you use for the FFD can be fine-tuned later by going back down in the stack to the FFD to adjust the number of control points.
2. Select the FFD and the mesh and apply a Physique modifier to both.
3. Click Attach to Root and choose the biped pelvis.
4. Select the mesh only and add a Mesh Select modifier to the mesh, *above* Physique in the modifier stack.
5. Go to Vertex sub-object level, and select the set of vertices that lay inside the FFD space warp you are using.  
Note: If you are using a NURBS model, you will want to use a NsurfSel modifier to select sub-object control points that lay inside the FFD lattice.
6. While in Vertex sub-object level, bind the mesh to the FFD space warp. Now only the selected vertices or CVs will be affected by the FFD.

7. With the mesh selected, go down in the stack to the Physique level and assign all vertices or CVs affected by the space warp as blue (rigid). These should be the same vertices or CVs selected in step 5, above.
8. Assign the rest of the vertices or CVs that fall outside the FFD as deformable, in the same way you normally assign vertices to links with Physique.
9. Select the FFD space warp. If necessary, adjust the vertex assignments of the control points so they are assigned to the proper links.

You do this in the same way as you do when using Physique with a mesh, except there are fewer assignments to deal with. Although fewer assignments provide smoother surface deformation with the FFD, control points and their link assignments must be thoughtfully placed.

## Crowd Animation User Interface



Crowd animation, one of the most important new features in **character studio 3**, lets you simulate the behavior of crowds of people, animals, or other beings parametrically using several different types of objects. The following topics contain explanation of and procedures for using the Crowd system features.

- The *Crowd helper object* (see page 346) includes facilities for replicating and grouping objects and applying Crowd\_Behaviors to objects and groups. The Crowd object works directly on *delegate* helper objects, and indirectly on bipeds and other objects via delegates.
- The *Delegate helper object* (see page 342) lets you control individual delegate behavior, such as speed and turning characteristics.
- The *Vector Field space warp* (see page 417) lets you control crowd behavior with respect to obstacles as well as with surrounding objects such as a room.

## Delegate Helper Objects

Create panel > Helpers > Object Type rollout > Delegate

The Delegate helper object is a special object used in crowd animation. It serves as an agent for motion created by a Crowd object and its behaviors. The Crowd object controls a delegate or delegates, whose motion can then be imparted to a biped or other object. Delegates cannot be rendered.

The delegate object takes the shape of a pyramid. By default, the point of the pyramid indicates the forward direction.

The delegate object includes the following rollouts:

*Geometry Parameters Rollout (see page 342)*

*Motion Parameters Rollout (see page 342)*

Also, you can set parameters for multiple delegates simultaneously, optionally with random variation, using the Crowd object's *Edit Multiple Delegates* dialog (see page 372).

### Procedure

#### To add a delegate object

1. Click in a viewport to set the delegate position, and then drag vertically to set the size.
2. Release to finish.

**Tip:** The delegate always points “up” in the viewport you add it in. So if you want it to point “forward,” that is, toward the positive Y direction in the World coordinate system, then add it in the Top viewport.

## Geometry Parameters Rollout

Create panel > Helpers > Object Type rollout > Delegate > Geometry Parameters rollout

Select a Delegate object. > Modify panel > Geometry Parameters rollout

Use these parameters to modify the delegate object's size.

### Interface

**Width, Depth, Height.** Sets the width, depth, and height of the Delegate object. These fields also act as readouts when you create the delegate.

Note: Delegates are helper objects, and thus cannot be rendered. Thus the size of the Delegate object is primarily for use in scene setup, and for determining bounding box extents.

## Motion Parameters Rollout

Create panel > Helpers > Object Type rollout > Delegate > Motion Parameters rollout

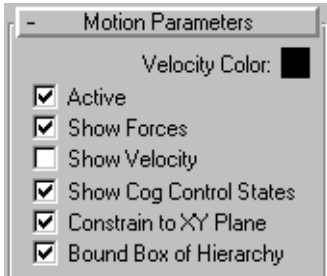
Select a Delegate object. > Modify panel > Motion Parameters rollout

The Motion Parameters rollout lets you specify a Delegate object's characteristics, including speed, acceleration and other factors. It also lets you associate the delegate with a biped.

Important note: When using delegates with bipeds, only the settings in the Biped group have any effect in the simulation, because Biped gets all its speed and turning information from motion flow clips and behaviors.

Note: You can set any or all motion parameters simultaneously for a number of delegates with the *Edit Multiple Delegates* dialog (see page 372).

## Interface



**Velocity Color.** When Show Velocity is on, uses the specified color to draw a vector in the delegate's center during the simulation solution. The vector length indicates the delegate's relative speed. Default=black.

**Active.** The delegate object is subject to control by a Crowd object. Default=on.

**Show Forces.** The forces being applied to a delegate by any applicable behaviors are drawn as vectors whose length indicate the extent of the forces and whose orientation shows the direction in which the behavior is influencing the delegate to move. For example, if the delegate is affected by a *Space Warp* behavior (see page 404) and a *Wander* behavior (see page 416), the vectors (using default colors) are yellow and blue-green, respectively. These vectors are visible only during solution of the crowd simulation. Default=on.

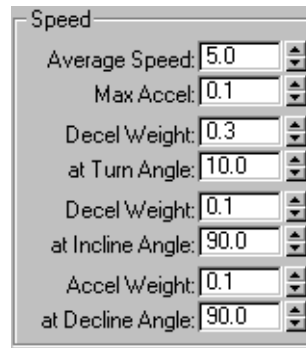
**Show Velocity.** Uses the Velocity Color (see above) to draw a vector whose length depicts the delegate's relative speed. This vector is visible only during solution of the crowd simulation. Default=off.

**Show Cog Control States.** During a solve, a text label appears next to the delegate showing the name of the *cognitive controller* (see page 382) state or transition that currently directs its behavior, if any. Default=on.

**Constrain to XY Plane.** The delegate remains at its initial height (position on the world Z axis) throughout the simulation. When off, the delegate's height can change during the simulation, for example when seeking an object at a different height. Default=on.

**Bound Box of Hierarchy.** When on, the *Avoid* behavior (see page 392) uses the bounding box of the delegate and all of its children to perform its behavior. Default=on.

## Speed group



**Average Speed.** Specifies the delegate's baseline velocity in 3DS MAX units (or the current unit type) per frame. This can be modified during the simulation by a variety of factors, such as a linked biped's built-in speed and Deviation settings in a behavior. Default=5.0.

**Max Accel(eration).** Multiplied times Average Speed to determine the maximum acceleration. Default=0.1.

For example, given the defaults of 5.0 for Average Speed and 0.1 for Max Accel, the

acceleration or deceleration at any moment can be no greater than 0.5 units/frame/frame.

**Decel(eration) Weight.** Specifies how much a delegate should slow down when turning. The higher this setting, the more the delegate slows down when it reaches the turn angle (see following parameter). A value of 0 specifies no slowdown; a value of 1 tells the delegate to stop. Default=0.3.

The algorithm computes a value,  $d$ , which goes linearly from 0 to  $(1 - \text{Decel Weight})$  as the turn angle of the delegate goes from 0 to the Turn Angle specified by the user. The speed of the delegate is then multiplied by  $d$ . For example, when the delegate turns at the Turn Angle or greater, its speed will be multiplied by  $1 - \text{Decel Weight}$ , slowing it down as much as possible based on this parameter. When the delegate is not turning at all, its speed is not affected by the Decel Weight. When the delegate is turning at half the specified Turn Angle,  $d = \text{Decel Weight} / 2$ , so its speed will be multiplied by  $(1 - \text{Decel Weight} / 2)$ .

As a practical example, take a delegate traveling at 10 units/frame, Decel Weight is set to 0.4, and At Turn Angle is set to 30. When the delegate has turned 15 degrees (half the At Turn Angle), the effective deceleration weight is 0.2. Subtract that quantity from 1 to get 0.8, and then multiply that times the delegate's speed to get 8 units per second halfway into the turn. At the full turn (30 degrees), the delegate travels at 6 units per frame.

**At Turn Angle.** Specifies the turn angle at which Decel Weight's full slowdown effect is applied. If the current turn angle is less than At Turn Angle, the algorithm divides the latter by the former, and then divides the Decel Weight setting by the result to derive the effective decel weight. Default=10.0.

**Decel(eration) Weight.** Specifies how much the delegate should slow down when moving at an upward slant. Default=0.1.

See *Decel Weight* (see page 344) for a full explanation.

**At Incline Angle.** Specifies the upward slant angle at which Decel Weight's full slowdown effect is applied. Default=90.0.

**Accel(eration) Weight.** Specifies how much the delegate should speed up when moving at a downward slant. Default=0.1.

See *Decel Weight* (see page 344) for a full explanation, taking into account that Accel Weight produces a speedup effect rather than a slowdown. Thus, the effective acceleration weight is added to 1, not subtracted from it.

**At Decline Angle.** Specifies the downward slant angle at which Accel Weight's full speedup effect is applied. Default=90.0.

### Turning group

These parameters affect the delegate's turning behavior; that is, how it changes direction in response to forces applied by crowd behaviors.

You can use the *Orientation Behavior* (see page 396) to affect how a delegate turns and banks, independently of the actual path taken.

Turning applies to objects moving both on the ground and in the air.

**Max Turn Velocity.** Specifies the maximum number of degrees a delegate can turn per frame.



Applies both to heading and pitch.  
Default=30.0.

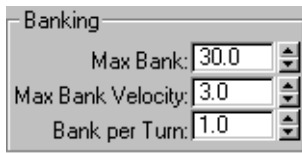
**Max Turn Accel.** Specifies how much the delegate's heading or pitch angle can change per frame. This controls angular acceleration and deceleration. For smooth turns, keep it relatively low. Default=3.0.

**Tip:** If a delegate exhibits sluggish turning behavior during a simulation, try increasing Max Turn Velocity or Max Turn Accel, or both.

**Max Incline.** Specifies the maximum number of degrees a delegate can turn upward at any given frame. For example, most birds can't fly straight up, so you might set this to 45 for a bird. Default=90.0.

**Max Decline.** Specifies the maximum number of degrees a delegate can turn downward at any given frame. For example, for a bird that can't fly straight up but can fly straight down, you might set to Max Incline to 45 and Max Decline to 90. Default=90.0.

### Banking group



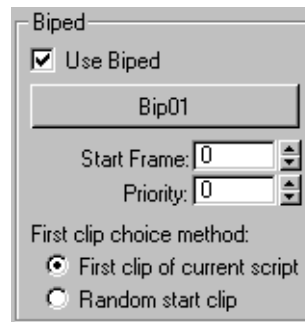
These parameters affect the delegate's banking behavior; that is, how it tilts around its front-back axis as it changes direction while moving. Banking typically applies to objects moving in the air or on water, but can also apply to ground-based objects such as one- or two-wheeled vehicles.

**Max Bank.** Specifies the maximum number of degrees a delegate can bank. Default=30.0.

**Max Bank Velocity.** Specifies the maximum number of degrees a delegate can bank per frame. Default=3.0.

**Bank per Turn.** The number of degrees the delegate will bank as a function of the turn angle at the current frame. For example, if Bank per Turn=1, the delegate will bank one degree for every degree it is turning at a given frame. Default=1.0.

### Biped group



These parameters relate to the use of bipeds associated with delegates. In order to have a biped exhibit character animation as it follows the delegate's course, you must use motion flow methods. For procedures, see:

*To use bipeds in a crowd simulation (see page 347)*

**Also, for tutorials, see:**

*Creating a Crowd of Swimming Bipeds (see page 573)*

*Advanced Crowd/Bipeds (see page 577)*

**Use Biped.** Associates the delegate with a biped (specified with the None button), and causes the delegate's speed to be determined by that of the biped's existing motion. Its behavior (for example, seeking another object) remains defined by a crowd object.

Note: This button is not available until a biped is designated via the None button.

**None (label).** Click this button and then select a biped to be associated with the delegate's motion. If selecting a biped in a viewport, you must click the biped's center of mass object (e.g., bip01). Thereafter, the name of the biped object appears on the button.

**Start Frame.** Specifies the frame at which the biped's first clip will begin to play. If, when several bipeds share the same starting clip, you vary this setting per biped, they won't walk in lockstep formation. This is most useful when you take advantage of the ability of the *Edit Multiple Delegates dialog* (see page 372) to randomize the start frame for each delegate.

**Priority.** Sets the delegate priority, which determines the order of solution in biped/ delegate simulations. For details, see *Priority Rollout* (see page 354).

First clip choice method

Determines which motion clip in the shared motion flow graph Crowd initially uses to animate the biped linked with the delegate.

**First clip of current script.** Uses the first clip in the biped's motion flow script, if a script exists. If this option is chosen, but there is no script, an error message is generated.

**Random start clip.** Uses the random start clip or clips specified in the shared motion flow graph, if random start clips have been designated. If this option is chosen, but no random start clips have been designated, an error message is generated.



## Crowd Helper Object

Create panel > Helpers > Object Type rollout > Crowd

The Crowd helper object acts as the command center for controlling crowd simulations in **character studio**. In most cases, you won't need more than one Crowd object per scene.

The Crowd object provides the following rollouts:

*Setup Rollout* (see page 360)

*Solve Rollout* (see page 352)

*Priority Rollout* (see page 354)

*Smoothing Rollout* (see page 357)

*Collisions Rollout* (see page 358)

*Geometry Rollout* (see page 359)

*Global Clip Controllers Rollout* (see page 359)

## Crowd Behaviors

The Crowd object also lets you add behaviors to the scene, choose the current behavior from a list, and provides a rollout for modifying that behavior. Behaviors provided with **character studio** include:

*Avoid Behavior* (see page 392)

*Orientation Behavior* (see page 396)

*Path Follow Behavior* (see page 398)

*Repel Behavior* (see page 400)

*Scripted Behavior* (see page 402)

*Seek Behavior* (see page 403)

*Space Warp Behavior* (see page 404)

*Speed Vary Behavior* (see page 405)

*Surface Arrive Behavior* (see page 406)

*Surface Follow Behavior* (see page 409)

*Wall Repel Behavior* (see page 411)

*WallSeek Behavior (see page 414)*

*Wander Behavior (see page 416)*


## Procedures

### To add a Crowd helper object

- Click Crowd and then drag the helper object to any convenient size.

### To use Crowd and Delegate helper objects together

This example shows how to create a basic crowd simulation with delegates and behaviors.

- Run 3DS MAX or reset the program.
- Add a Crowd object and one or more *Delegate objects (see page 342)* to the scene. In general, add delegates in the Top viewport so that they point forward. The Crowd object's location is immaterial.
- Select the Crowd object and open the Modify panel.
- On the Setup rollout > Behaviors group, click the New button.
- In the Select Behavior Type dialog, click a behavior and then click OK to close the dialog.  
A Behavior rollout appears for the behavior you chose.
- If the behavior requires a target object or objects, such as Seek, click the None button and then select an object, or click Multiple Selection and select several objects.
- Change other behavior settings as necessary.
- Create and modify additional behaviors as necessary.
-  On the Setup rollout, click the Behavior Assignment button.
- In the Assignment Design group, the two upper lists should each contain a single

entry: the delegate on the left, and the behavior on the right. Select both items.

- Click the New Assignment button to the right of the Assignment Design group. It's a vertical button with five rightward-pointing arrows.

This adds the new assignment to the list in the Behavior Assignments group.

- Accept the changes and close the Behavior Assignments and Teams dialog by clicking the OK button.

- On the Modify panel, scroll down to the Solve rollout and click the Solve button.  
Keys are created as follows: The delegate turns to point toward the sphere, banking as it turns, and then moves directly toward the sphere. When it reaches its target, it moves slightly beyond the sphere, and then repeats the turn-and-move motion until the end of the simulation. To prevent this, try starting with the two objects farther apart, or animating the sphere's position.

The next procedure builds on this one by assigning a biped to the delegate's motion.

### To use bipeds in a crowd simulation

*Motion synthesis* is an important new feature in **character studio 3**. In motion synthesis, the software uses a "game engine" that determines procedurally when to apply various motion clips to a biped during solution of a crowd simulation, based on the delegate's activity. For example, when one delegate stops to wait for another to get out of its way, using the Avoid behavior, the biped attached to the stopped delegate could cross its arms and tap its foot impatiently.





This procedure gives the basic steps for creating and saving a motion flow graph, assigning several bipeds to delegates, and then sharing the

motion flow graph among the bipeds so that the software automatically creates separate motion flow scripts each biped, based on the behaviors assigned to their delegates.

For tutorials covering crowd/biped methods, see *Creating a Crowd of Swimming Bipeds* (see page 573) and *Advanced Crowd/Bipeds* (see page 577). Also, for an important overview of using bipeds with Crowd, see *Using Bipeds with Crowd Delegates* (see page 116).

Note: This procedure assumes you know how to animate bipeds with footsteps and keyframe methods, and save the animations as *.bip* files. To prepare for using Shared Motion Flow, you should create and save a range of biped motions such as start, walk, turn right, turn left, stop, and wait.


The first step is to create an appropriate graph and save it to disk.

1.  Add a biped, and then open the Motion panel.
2.  Click the General rollout > Motion Flow Mode button.
3.  Click the Motion Flow rollout > Show Graph button.
4.  In the Motion Flow Graph toolbar, click the Create Multiple Clips button. This displays an Open dialog that lets you select any number of *.bip* files from the same directory to add simultaneously to the motion flow graph. Use Click+Shift-click to choose several contiguous files, and Ctrl-click to choose non-contiguous files.
5. Choose the *.bip* files you want the software to use for motion synthesis, and then click the OK button.


The files are added to the graph as clips. Each clip is automatically named after the file from which it's derived.


Note: For best results, especially with simulations in which bipeds are to turn at different angles, use as many different turning clips as possible. A minimal setup would include separate left-turn and right-turn walks at angles of 45, 90, 135, and 180 degrees.

The next step is to add transitions among the clips so **character studio** knows which actions can proceed to and from other actions. You can do this manually for greater control, but for initial testing, you can save time by letting the software add and optimize transitions automatically.


6.  Click the Synthesize Motion Flow Graph button. This uses the first 30 percent and the last 30 percent of each motion to create transitions.

The graph now shows arrows to and from each clip, as well as from each clip to itself. If you like, delete transitions that obviously don't belong, such as the ones from the stop and start clips to themselves.

 Alternatively, you can use Create Transition to set up a custom graph.

7.  Optimize the transitions. See *Transition Optimization Dialog* (see page 245).

When you solve the simulation, Crowd automatically generates a motion flow script for the biped, based on this graph. When you have a graph with multiple clips, as in this case, it chooses the starting clip for the script from one or more clips you designate as random start clips.

8.  Click the Select Random Start Clips button, and then click a clip.

This tells **character studio** to start the script with this clip, and uses the default probability of 100% that the clip will be chosen.

If you want the various bipeds to start with different clips, select multiple random start clips by pressing and holding the Ctrl key as you click. The default Random Start Probability setting of 100 for all clips means that the software will choose randomly among them for a starting clip for each biped's script.



To change the likelihood of starting with specific clips, right-click a clip and modify its Random Start Probability setting. For example, say you want to start each biped's script with any of three clips: clips A, B, and C. You want clip A to be chosen twice as often as clip B or C. In that case, using the Random Start Clips tool, you'd first click clip A, and then Ctrl-click clips B and C. Then you'd right-click each in turn, assigning a Random Start Probability of 60 to clip A, and 30 to both clips B and C.



Note: The probability values are arbitrary; what counts is their ratios. For example, values of 80/40/40 or 20/10/10 would work the same.

Note: You can also set and change random starting clips and start probabilities in the Motion Flow graph after loading the *.mfe* file into the Global Motion Flows dialog, described later in this procedure.

Note: If a motion flow script already exists for a biped, for example after you've solved a crowd simulation, the software can use the


first clip in the script for subsequent solutions.

9.  In the Motion Flow rollout, click the Save File button, and save the graph in the *.mfe* format.
10. Reset the software.
11. Set up a crowd simulation with any number of delegates, using behaviors appropriate to the crowd scene you want to create.  
**Tip:** When first starting out with motion synthesis, use smaller crowds of eight or so delegates.
12. Solve the simulation and adjust the settings as necessary to obtain the desired activities.
13. Add as many bipeds as you have delegates, and link each delegate to a different biped using the delegate's *Motion Parameters rollout* (see page 342) > Biped group settings.  
You must select the biped's center of mass (COM) object (typically named Bip0#), as indicated by the mouse cursor turning into a crosshairs icon when it's over the COM in the active viewport.
14. Turn on Use Biped for each delegate.  
 **Tip:** To link up any number of delegates with bipeds, all at the same time, use the Crowd helper object. Click the Setup rollout > *Biped/Delegate Associations* (see page 374) button, and then use the dialog to connect the pairs and turn on Use Biped.
15. Set the biped/delegates to use a random start clip as the first clip. You can set this simultaneously for multiple delegates with the *Edit Multiple Delegates dialog* (see page 372).  
Next, you use the Shared Motion Flow function to apply the saved motion flow graph to the bipeds.

16. Select any biped and open the Motion panel.
17.  Click the General rollout > Motion Flow Mode button.
18.  Click the Motion Flow rollout > Shared Motion Flow button.
19. In the Shared Motion Flow dialog, click the New button.  
This creates a new shared motion flow and assigns it a default name. You can change the name if you like.  
Next, load a motion flow file.
20. Click the Parameters group > Load .mfe button, and use the Open dialog to load a motion flow file. Typically, this would be the one you saved earlier in the procedure. Next, specify the bipeds that will share this motion flow.
21. In the Parameters group, click the Add button, and use the Select dialog to specify the bipeds that will share the motion flow. For your convenience, the Select dialog shows only center of mass objects for the bipeds in the scene.  
After you click the Select button, the bipeds appear in the dialog, in the list under “Bipeds Sharing this Motion Flow.”
22. To correctly share a motion flow, bipeds’ legs must be scaled the same. If any of the bipeds are scaled differently than the one you started with, an alert appears, and then, when you click the OK button in the alert box, the wrong-scale bipeds are noted as such in the list. At this point, you can select one of the bipeds in the list, and then click the Set Shared Motion Flow Scale button to match the others’ scale to that biped. Or you can click one of the Reset Wrong Scales

buttons to rescale the wrong-scale bipeds or just their legs. Be sure to take one of these measures before proceeding.

One more step in the Shared Motion Flow dialog is necessary: You must activate Motion Flow mode for all the bipeds sharing the motion flow. A special button in the dialog lets you perform that action in one step.

23.  Click the Put Multiple Bipeds in Motion Flow button.  
This activates Motion Flow mode for all the bipeds sharing the motion flow.
24. Click the OK button to exit the dialog. Delegate-controlled bipeds can begin their animation with their motion flow scripts’ first clip, if it exists, or with a random motion clip. But when you load a motion flow file into the Shared Motion Flow dialog, any scripts in the file are ignored. Thus, delegate-associated bipeds using motion flow in an unsolved crowd simulation have no existing scripts, and you must specify that they use the random start clip that you set in the motion flow. You do this via the delegates.
25. Use the Edit Multiple Delegates dialog > Biped group to specify Random Start Clip for all the delegates.  
Solve the simulation.
26. Select the Crowd object, and go to the Modify panel.
27. In the Solve rollout, set the desired End Solve frame.
28. Click Solve to run the simulation.  
Crowd solves the simulation.

29. Check the solved simulation by dragging the frame slider and/or playing back the animation.  
Chances are adjustments will be required. You can resolve problems in a number of different ways: Change the behavior and/or motion flow setups, change delegate parameters, and so on.  
If you find that bipeds are colliding and interpenetrating, you can take advantage of Crowd's special Priorities and Backtracking features. In fact, it is strongly recommended that you use both options for most Crowd/Biped simulations.
30. Use the *Priority rollout* (see page 354) controls for assigning different priorities to your delegates. Typically, delegates at the head of the crowd should have the highest priorities (that is, the lowest Priority settings).
31. In the Solve rollout > Bipeds group, turn on Biped/Delegates Only, then turn on Use Priorities, and then turn on Backtracking. If you've solved previously, it's a good idea to turn on Delete Keys Before Solve as well.  
Now, when it solves the simulation, Crowd solves for one biped/delegate at a time, starting with the lowest-priority biped/delegate. As it solves for each subsequent biped/delegate, it looks for collisions, and when they occur, it backs up the solution to the end of the previous clip, and if necessary, previous clips, and then tries different paths through the motion flow graph. This method can take longer, which is why Backtracking is off by default, but it's often the best way to resolve problems with colliding bipeds.



## Behavior Rollout

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button

Select a Crowd object. > Modify panel > Setup rollout > New button

Select a Crowd object. > Modify panel > Setup rollout > Choose a behavior.

The Behavior rollout appears in the Crowd object command panel after you first add a behavior to the scene using the Setup rollout. The rollout's full name (for example: Avoid Behavior or Seek Behavior) depends on the current behavior, displayed in the text box in the Setup rollout. To display the rollout for a different behavior, choose the behavior from the list.

Note: If you add the first behavior to the scene from the *Behavior Assignments and Teams Dialog* (see page 375), a behavior rollout does not automatically appear in the Crowd command panel. You must first choose the behavior from the list at the bottom of the Setup rollout.

Use the controls in this rollout to modify the behavior. Following is a list of available behaviors:

*Avoid Behavior* (see page 392)

*Orientation Behavior* (see page 396)

*Path Follow Behavior* (see page 398)

*Repel Behavior* (see page 400)

*Scripted Behavior* (see page 402)

*Seek Behavior* (see page 403)

*Space Warp Behavior* (see page 404)

*Speed Vary Behavior* (see page 405)

*Surface Arrive Behavior* (see page 406)

*Surface Follow Behavior (see page 409)*

*Wall Repel Behavior (see page 411)*

*WallSeek Behavior (see page 414)*

*Wander Behavior (see page 416)*

For a detailed description of specific behaviors, refer to the above topics. For an overall look at behaviors, see *Crowd Behaviors Overview (see page 112)*.

## Solve Rollout

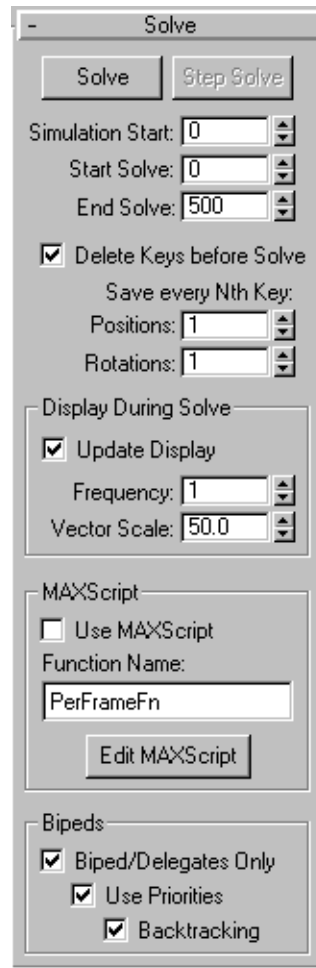
Create panel > Helpers > Object Type rollout > Crowd > Solve rollout

Select a Crowd object. > Modify panel > Solve rollout

Once you've set up the crowd simulation, use this rollout to set solution parameters and to solve the simulation. You can solve continuously or a frame at a time, starting at any frame.

In order to generate the simulation as quickly as possible, delegate keys are saved *after* the solution is run, so there might be a pause between the end of the solution and return of control of the software to you. Also, any objects linked to delegates are hidden during the simulation.

## Interface



**Solve.** Runs the crowd simulation continuously, applying all specified behaviors to delegates to which they are assigned. Solving a simulation overwrites any previous solutions.

To abort a solution in progress and save all keys generated up to that point, press the ESC key. Alternatively, with complex simulations, you can save time by pressing Shift-ESC to abort a solution without saving keys.





**Tip:** To save time, turn on the Plug-in Keyboard Shortcut Toggle and then press the S key to run a solution.

**Tip:** With simulations using many bipeds or mesh objects, you can speed up the solution significantly by increasing the Solve rollout > Display During Solve group > Frequency setting to **100** or so.

**Step Solve.** Runs the crowd simulation one frame at a time, starting at the current frame as specified by the time slider position. Press the spacebar to advance one frame.

To abort a solution in progress and save all keys generated up to that point, press the ESC key. When you do so, the software disregards any non-default settings for Save Every Nth Key. Alternatively, with complex simulations, you can save time by pressing Shift-ESC to abort a solution without saving keys.

Note: Step Solve always starts at the current frame; it disregards the Simulation Start setting.

**Tip:** Use Step Solve to analyze your simulation when things don't go as expected. You can start at any frame. Zoom in to examine the vectors of misbehaving delegates.



**Tip:** To save time, turn on the Plug-in Keyboard Shortcut Toggle and then press the T key to run a solution in step mode.

**Simulation Start.** The first frame of the simulation. You should set this and keep it the same, in order to make the solution repeatable. Default=0.

**Start Solve.** The frame at which you begin solving. This must be greater than or equal to Simulation Start. If greater, the solve will begin in the middle of the simulation. Default=0.

Start Solve should equal Simulation Start the first time you solve, so that when you set Start Solve to start in the middle of the simulation, the simulation up to that point will be correct.

Note: If you set Start Solve to a frame number lower than the first frame of the active time segment, **character studio** changes the first frame of the time segment to the Start Time value.

**End Solve.** Specifies the last frame considered for the solution. Default=100.

Note: If you set End Solve to a frame number higher than the last frame of the active time segment, **character studio** changes the last frame of the time segment to the End Solve value.

**Delete Keys before Solve.** Deletes the keys of active delegates in the range over which the solution takes place. Default=off.

This option leaves the first two keys so that the delegate doesn't end up with no keys and then pop to its current position. This is a useful feature for biped-crowds; it lets you watch each biped compute, one after another, without the ones not yet computed still performing their old animation.

**Save every Nth Key.** Lets you specify the number of position and rotation keys saved after the solution.

**Positions/Rotations.** The frequency with which keys are saved for delegate positions and rotations. If 0, no keys are saved. If 1, a key is saved every frame. If 2, a key is saved every other frame, and so on. Defaults=1.

**Tip:** To save memory and time, save fewer than one key per frame.

**Display During Solve group**

**Update display.** When on, motion produced during solution of a crowd simulation appears in the viewports. Default=on.

**Frequency.** How often the display is updated during the solution. If 1, the update occurs every frame. If 2, the update occurs every other frame, and so on. Default=1.

**Tip:** Increase Frequency to speed up the solution. For the fastest results, if you don't need to see interactions during the solution, set Frequency to a relatively high number, such as 100.

**Vector Scale.** Globally scales all force and velocity vectors that are displayed during the simulation. Scaling vectors up helps to see them better when they are very small. It does not effect the simulation. Default=1.0.

### MAXScript group

This feature lets you execute a MAXScript script at each frame. Its primary purpose is for working with bipeds; specifically, to take advantage of available MAXScript calls to Biped that let you specify which clip the biped will be likely to choose next when using the shared motion flow feature. With this feature, you can write a script which, based on what clip is currently being used, the frame number, the proximity of other bipeds, or anything else that you can find out in a script, the biped's next clip can be dynamically selected during the Crowd simulation. Of course, this scripting feature can be used for other purposes as well.

**Use MAXScript.** When on, a user-specified script is executed at each frame during the solution. Default=off.

**Function Name.** The name of the function to be executed. This name must also be specified in the script.

**Edit MAXScript.** Click this button to open a MAXScript window for displaying and modifying the script.

### Bipeds group

When solving simulations that use biped/delegates, it is strongly recommended that you use all three options in this group.

**Biped/Delegates Only.** When on, only biped/delegates are included in the computation. Also, the options to use priorities and backtracking become available. These options are available only for biped-only computations. Default=off.

**Use Priorities.** When on, biped/delegates are computed one delegate at a time, in order of their Priority values, from lowest to highest. Also, backtracking becomes available and Step Solve becomes unavailable. Default=off.

**Backtracking.** Turns on backtracking functionality when solving a crowd simulation that uses bipeds. Default=off.

When Backtracking is on during the solution, in the case of an impending collision between bipeds, the Crowd system will back up the simulation to the beginning of the current clip, and then try a different traversal of the lower-priority delegate/biped's motion flow graph. If necessary, the system will back up two or more clips.

---

## Priority Rollout

Create panel > Helpers > Object Type rollout > Crowd > Priority rollout

Select a Crowd object. > Modify panel > Priority rollout

The crowd system uses the Priority rollout settings when solving a simulation involving bipeds associated with delegates. The Priority parameter is a positive integer assigned by the user to a delegate. When priorities are used, one biped at a time is computed, based on its priority

setting *from lowest to highest*; that is, a lower Priority setting means a higher priority. If the priorities of two biped/delegates are the same, the computation order of those two biped/delegates is randomly determined.

This topic describes the six different ways of setting a delegate's priority, and how priorities might be put to use in different situations.

## Using Priorities

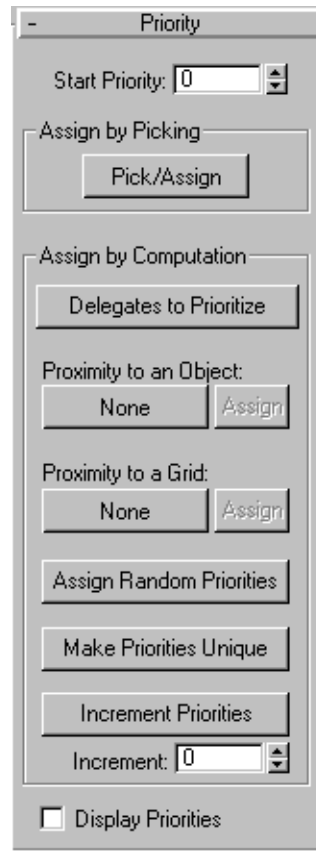
If you have a large crowd all going in one direction, you would typically want the delegates in front to solve first. In that case, using Proximity to an Object or Proximity to a Grid would be useful in setting priorities.

In a case where you start with a circle of bipeds, and you want them all to wander and mingle, you might not care about the bipeds' priorities. You could let them all have the same priority and let the system decide which goes first. However, it might be better to assign random priorities or make priorities unique, so that you are guaranteed the same order each time, and you can read the priority numbers to know what will happen next. This also lets you change the order if you need to.

Suppose you have two groups of bipeds, all of which are assigned random priorities. If you wanted to keep the priority relationships within each group, but make one group start after or before the other, you could use Increment Priorities to increment or decrement all the priorities in one group.

Obviously, you need some way to set the priorities by hand, if none of the algorithms applies to your situation. It's useful to be able to set them visually. That's what the Assign by Picking method is for.

## Interface



**Start Priority.** Sets the initial priority value. Applies to the first four methods of setting priorities: Assign by Picking, Proximity to an Object, Proximity to a Grid, and Assign Random Priorities. Default=0.

Note: Priority decreases as the setting increases. Thus, a delegate with Priority value 0 goes before a delegate with Priority 1, 1 goes before 2, and so on.

### Assign by Picking group

**Pick/Assign.** Lets you assign successively higher Priority values to any number of delegates by

selecting each in turn in the viewport. The first delegate you select is assigned the Start Priority value. The Priority value assigned to each succeeding delegate you select is incremented by one.

To stop assigning priorities, right-click in a viewport or click the Pick/Assign button again.

Delegate priorities appear in viewports as black numerals attached to each delegate; they're usually most easily seen in Wireframe views.

Note: You can undo and/or redo assignments during the process.

**Tip:** It's possible to assign two or more delegates the same priority value using this method. In such a case, for more predictable behavior, use Make Priorities Unique so that delegates don't share priorities.

### Assign by Computation group

This group provides five different methods for assigning priorities to delegates, plus a button for selecting delegates to be affected by these methods.

**Delegates to Prioritize.** Lets you use the Select dialog to specify delegates to be affected by subsequent use of other controls within this group. Select the delegates with the Select dialog, and then click Select to exit the dialog. This selection applies only to Proximity assignments (that is, Proximity To An Object and Proximity To A Grid).

**Proximity to an Object.** Lets you assign priorities based on delegates' distance from a specific object. To specify the object, click the None button, and then select the object on which priorities are to be based. Lastly, click the Assign button to compute and assign priorities. The delegate closest to the object is assigned the Start Priority value, and each successively farther delegate is assigned the next highest priority.

For any delegates that are equidistant from the object, the software assigns priorities randomly.

**Proximity to a Grid.** Lets you assign priorities based on delegates' distance from an infinite plane defined by a specific grid object. To specify the grid object, click the None button, and then select the grid object on which priorities are to be based. Lastly, click the Assign button to compute and assign priorities. The delegate closest to the grid object is assigned the Start Priority value, and each successively farther delegate is assigned the next highest priority.

For any delegates that are equidistant from the plane, the software assigns priorities randomly.

**Assign Random Priorities.** Assigns random priorities to the selected delegates. The range of priority values assigned lies between the Start Priority value and that value plus the number of selected delegates.

**Make Priorities Unique.** Ensures that all delegates have unique priority values. If two delegates share the same priority, one of them will be given a new priority value that differs from the rest.

**Increment Priorities.** Increments the priorities of all selected delegates by the Increment value.

**Increment.** Sets the value by which the Increment Priorities button adjusts delegate priorities. Use a negative Increment value to decrement priorities. Default=0.

**Display Priorities.** Enables the display of assigned priority values as black numerals attached to the delegates. Default=on.

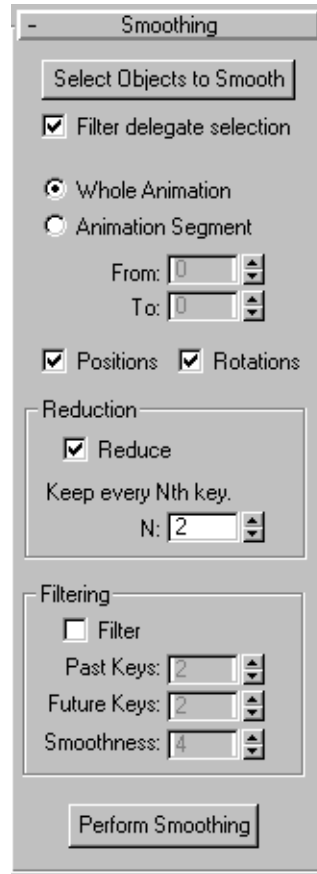
## Smoothing Rollout

Create panel > Helpers > Object Type rollout  
> Crowd > Smoothing rollout

Select a Crowd object. > Modify panel >  
Smoothing rollout

Smoothing works on existing animation keys (that is, a solved simulation) to create more natural-looking animation. Use these controls if solving a crowd simulation results in abrupt position or rotation changes of animated objects. By default, smoothing works by reducing keys. As an option, it can also filter (average) existing animation keys to make changes more gradual, resulting in more natural motion.

### Interface



**Select Objects to Smooth.** Opens the Select dialog, which lets you specify which objects' positions and/or rotations to smooth.

**Filter Delegate Selection.** When on, the Select dialog opened by the Select Objects to Smooth button shows only delegates. When off, it shows all scene objects. Default=on.

**Whole Animation.** Smoothes all animation frames. This is the default option.

**Animation Segment.** Smoothes only the frame ranges specified in the From and To fields.

**From.** When Animation Segment is chosen, specifies the first animation frame for smoothing.

**To.** When Animation Segment is chosen, specifies the last animation frame for smoothing.

**Positions.** When on, selected objects' animation paths generated via the simulation are smoothed after the simulation has finished. Default=on.

**Rotations.** When on, selected objects' rotations generated via the simulation are smoothed after the simulation has finished. Default=on.

### Reduction group

**Reduce.** Reduces the number of keys by keeping only every Nth key.

**Keep Every Nth Key: N.** Limits the amount of smoothing by keeping every other key (N=2), or every third key (N=3), and so on. Default=2.

### Filtering group

Smoothing works by averaging the delegate's current position and/or orientation with those several keyframes ahead and behind. All keyframes used in the calculation can be affected. These settings let you control the number of keyframes used and the extent of smoothing performed.

**Filter.** When on, smoothing is performed using the remaining settings in this group.

**Past Keys.** The number of keys prior to the current frame used for averaging position and/or rotation. Default=2.

**Future Keys.** The number of keys after the current frame used for averaging position and/or rotation. Default=2.

**Smoothness.** Determines the degree to which smoothing is performed. The higher the setting,

the closer all keys involved in the calculation are moved toward the average value. Default=4.

The highest available Smoothness value is always 6. The lowest available value depends on the Past Keys and Future Keys settings.

**Perform Smoothing.** Click this button to carry out the smoothing operation.

Note: If neither the Reduce nor the Filter check box is turned on, no smoothing is performed.

---

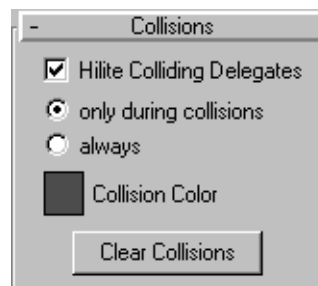
## Collisions Rollout

Create panel > Helpers > Object Type rollout > Crowd > Collisions rollout

Select a Crowd object. > Modify panel > Collisions rollout

During the simulation, you can use this rollout to keep track of collisions defined by Avoid behaviors. A delegate whose hard radius as defined by the Avoid behavior intersects with the hard radius of anything it is avoiding is marked as having collided at that frame. If too many collisions occur, the simulation might not provide satisfactory results; in such cases, you might want to alter the simulation setup.

### Interface



**Hilite Colliding Delegates.** When on, delegates that collide are highlighted in the collision color.

**Only During Collisions.** Colliding delegates are highlighted only in frames in which they actually collide.

**Always.** Colliding delegates are highlighted in frames in which they collide and all subsequent frames.

**Collision Color.** The color swatch indicates the color used to highlight colliding delegates. To change the color, click the swatch and use the Color Selector dialog to set a new color.

**Clear Collisions.** Clears collision information from all delegates.

If you re-solve part of the simulation, new collision information for the recomputed frames will be computed. However, if you move a delegate manually, its collision information will also remain the same, and may be incorrect. In such cases, use Clear Collisions to correct the collision information.

---

## Geometry Rollout

Create panel > Helpers > Object Type rollout > Crowd > Geometry rollout

Select a Crowd object. > Modify panel > Geometry rollout

Use this parameter to modify the crowd object's size.

### Interface

**Icon Size.** Determines the size of the Crowd helper object's icon. This setting is primarily for visibility; it has no effect on the crowd simulation.

---

## Global Clip Controllers Rollout

Create panel > Helpers > Object Type rollout > Crowd > Global Clip Controllers rollout

Select a Crowd object. > Modify panel > Global Clip Controllers rollout

Use global clip controllers when assigning non-biped animated objects (such as a bird flapping its wings) to delegates in a crowd simulation. Applications include synthesis of animation activity based on a variety of criteria, such as an object's speed, acceleration, and pitch. This group of controls replicates the Track View controls for the same functionality. For an in-depth discussion of global clip controllers and related topics, see:

*Clip Controllers (Track View) (see page 119)*

*Synthesis Dialog (see page 270)*

*State Dialog (see page 275)*

Also, for a procedure and a tutorial covering clip controller usage, see:

*How to use the Synthesis and ClipState dialogs (see page 275)*

*Using Crowd with Animated Non-Biped Objects (see page 567)*

### Interface



**(List).** Lists objects designated as Global Objects, whose controllers can be used as animation clips to control other objects (typically clones).

To designate an object as a Global Object, click the New button, and then select the object in the Select dialog.

**New.** To designate a Global Object and add it to the list, click this button, and then select the object in the Select dialog.

**Edit.** To modify a Global Object's properties, click its name in the list, and then click this button. This opens the *Synthesis Dialog* (see page 270).

**Load.** Loads a previously saved Global Motion Clip (.ant) file from disk.

**Save.** Stores the current Global Motion Clip settings on disk in the .ant file format.



## Setup Rollout

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout

Select a Crowd object. > Modify panel > Setup rollout

The Setup rollout of the Crowd helper object contains controls for the setting up crowd functions.

### Interface



A number of functions are available from a row of buttons at the top of the Setup rollout:



**Scatter.** Opens the *Scatter Objects Dialog* (see page 362).



**Objects/Delegate Associations.** Opens the *Object/Delegate Associations dialog* (see page 370).



**Biped/Delegate Associations.** Opens the *Associate Bipeds With Delegates dialog* (see page 374).



**Multiple Delegate Editing.** Opens the *Edit Multiple Delegates dialog* (see page 372).





**Behavior Assignments.** Displays the *Behavior Assignments and Teams dialog* (see page 375).



**Cognitive Controllers.** Displays the *Cognitive Controller editor* (see page 382).

### Behaviors group

Use these controls for adding, removing, and renaming behaviors. You can also convert a behavior to a script.

**New.** Launches the *Select Behavior Type dialog* (see page 380). Select a behavior and then click OK to add the behavior to the scene. Then use the *Behavior Assignments and Teams dialog* (see page 375) to assign the behavior to a delegate or delegates in the scene.

The first time you add a behavior to the scene using this command panel, a new rollout appears for this behavior below the Setup rollout. This rollout lets you change settings for the behavior. To display the rollout for a different behavior in the scene, choose it from the list below the Convert to MAXScript button.

Following is a list of available behaviors:

*Avoid Behavior* (see page 392)

*Orientation Behavior* (see page 396)

*Path Follow Behavior* (see page 398)

*Repel Behavior* (see page 400)

*Scripted Behavior* (see page 402)

*Seek Behavior* (see page 403)

*Space Warp Behavior* (see page 404)

*Speed Vary Behavior* (see page 405)

*Surface Arrive Behavior* (see page 406)

*Surface Follow Behavior* (see page 409)

*Wall Repel Behavior* (see page 411)

*WallSeek Behavior* (see page 414)

*Wander Behavior* (see page 416)

**Delete.** Deletes the current behavior.

If the behavior is currently in use, that is, it's assigned to a delegate or team, either directly in the *Behavior Assignments and Teams dialog* (see page 375) or indirectly through a cognitive controller, a small dialog appears asking you to confirm the deletion. If you delete a directly assigned behavior, its assignment is removed from the scene. If you delete a behavior used in a cognitive controller, it is removed from the state to which it was assigned.



**Convert to MaxScript.** Converts the current behavior to MAXScript. When you click this button, the *Scripted Behavior rollout* (see page 402) appears. Available only for the Seek behavior.

**Behaviors List.** Lists all behaviors in the current scene (added with New). Select a behavior from the list to have its rollout appear below the Setup rollout.

Note that behaviors that appear in this list aren't necessarily assigned to any delegates active in the crowd simulation. Likewise, a behavior whose rollout appears below the Setup rollout isn't necessary active or assigned. To assign delegates and/or activate behaviors, use the *Behavior Assignments and Teams dialog* (see page 375).

You can rename a behavior by first selecting it from the list, and then clicking its name and entering a new one from the keyboard. It's a good idea to give descriptive names to behaviors; for example, Avoid Red Team.

Note: If you add the first behavior in the scene from the *Behavior Assignments and Teams dialog* (see page 375), the text box remains empty and no rollout for the behavior appears. To edit the behavior, choose it from the list.

## Scatter Objects Dialog


Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Scatter Objects

Select a Crowd object. > Modify panel > Setup rollout > Scatter Objects

The Scatter Objects dialog of the Crowd helper object includes facilities for creating crowds by cloning objects as well as distributing clones and other objects within a radial area, along a shape, across a grid object or surface, or within a box or sphere. You can also specify various orientation and scaling options for objects.

### Procedure

#### To produce multiple clones of an object

1. Create an object to clone.
2. Decide how to distribute the clones.
3. Depending on your choice in the previous step, create an object to define how the clones are to be positioned: a grid object, a primitive box or sphere, a shape, or any object to serve as a surface. Alternatively, you can distribute clones in a circular area without the use of a distribution object.
4. Add a Crowd object.
5. With the Crowd object selected, switch to the Modify panel.
6.  Click the Scatter Objects button to open the Scatter Objects dialog.
7. On the Clone tab, set the number of clones to create, and select the object to clone.
8. Click Generate Clones.  
This produces the specified number of clones in the same location. To vary

positions, orientations, and/or sizes, proceed with the following steps.

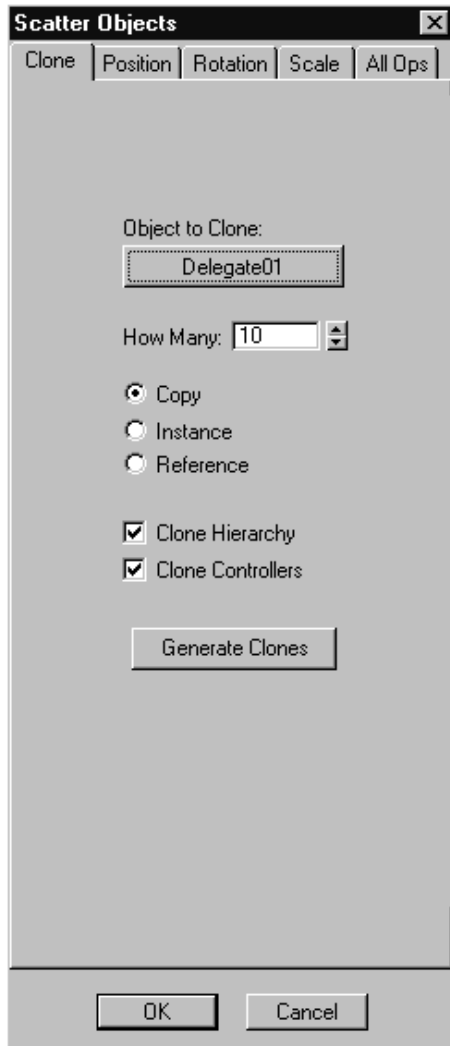
9. On the Position tab > Placement Relative to Object group, choose a distribution option. Alternatively, choose Placement in Area group > In Radial Area and set a center and radius, and then skip to step 11.
10. Click the None button, and then select the distribution object from step 3.
11. Click Generate Locations.  
This distributes the clones on or within the reference object.
12. On the Rotation tab, choose which of the cloned object's local axes are to look forward and up. Optionally, specify source and target objects for the clones' orientations, as well as limits for randomized deviations from the calculated orientation.
13. Click Generate Orientations.
14. On the Scale tab, for each axis, specify Average, Deviation, and, optionally, Same As settings and a random seed.
15. Click Generate Scales.  
At this point, you're basically finished. However, to create a series of randomized positions, orientations, and/or sizes for clones, follow the next two steps and repeat as necessary.
16. On the All Ops, turn on Positions, Rotations, and/or Scales. To vary the randomized positions, rotations, and/or scales, turn on the corresponding Inc Seed check boxes.
17. Click the Scatter button.

You can combine the cloning, position, rotation, and scale functions in one operation by setting the options on their respective tabs without clicking the Generate buttons, and then using the All Ops tab to apply any or all scatter

operations simultaneously. Also, you can use the All Ops tab > Select Objects to Transform to specify any objects to scatter; not just clones.

## Interface

### Clone panel



Contains the basic options for cloning an object.

**Object to Clone (None).** Click this button, and then select an object in the scene to be cloned.

**How Many.** Specifies the number of clones to be generated.

**Copy/Instance/Reference.** Lets you specify how the object is cloned. It can be cloned as a copy, an *instance* (see page 727), or a *reference* (see page 731).

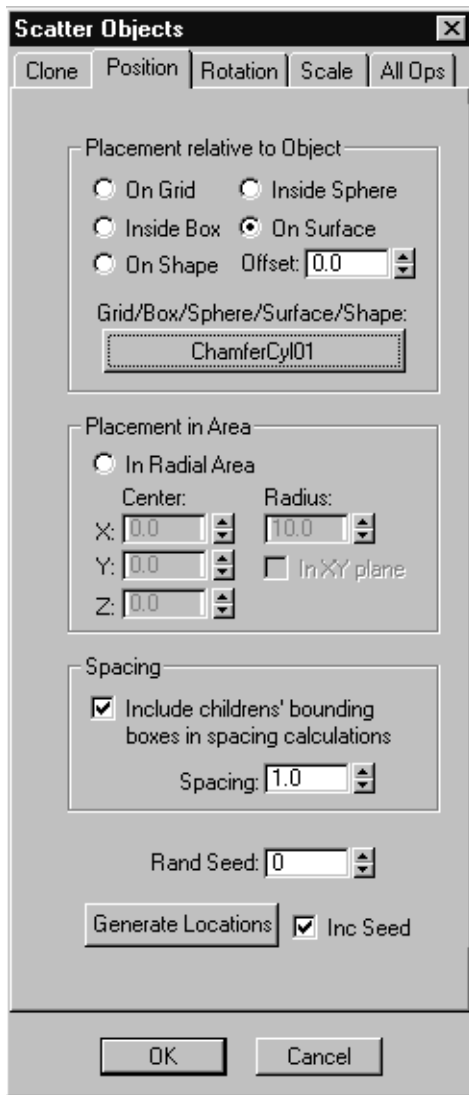
**Clone Hierarchy.** When on, all objects linked to the selected object are cloned as well, with the hierarchical structure retained intact for each clone.

**Clone Controllers.** When on, any controllers (i.e., animation) associated with the selected object are cloned as well.

**Generate Clones.** Click this button to create the specified number of clones of the object whose name appears on the Object to Clone button. Cloned objects are all created in the same location; to distribute them, you must set distribution options on the Position panel, and then click the Generate Locations button.

After you generate clones, the original object and its newly created clones are scatter objects; that is, they're selected in the list in the All Ops panel (Select Objects to Transform), and are thus subject to subsequent operations on the Position, Rotation, and Scale panels. You can change this selection via the All Ops panel > Select Objects to Transform function or by generating clones again.

## Position panel



Contains options for positioning objects using a reference (distribution) object. You can distribute objects randomly over the surface of a grid or other object, along a shape, or within the

volume of a box or a sphere. You can choose only one option from the first two group boxes.

## Placement Relative to Object group

**On Grid/Inside Sphere/Inside Box/On Surface/On Shape.** Choose the appropriate item before selecting the reference object.

- On Grid distributes the scatter objects over the surface of a grid object.
- Inside Box and Inside Sphere distribute the scatter objects within the volume of a primitive box or sphere object, respectively.
- On Surface distributes the scatter objects over the surface of any renderable object. For example, you can create a landscape object for use as a distribution surface by applying a Noise modifier to a patch grid.
- On Shape distributes the scatter objects along a shape object: a spline or NURBS curve. If the shape consists of more than one curve, Scatter uses the lowest-numbered curve (typically the first one added).

**Offset.** When using On Surface, specifies a consistent distance above the surface (using surface normals) for distribution. Available only when On Surface is chosen.

**Grid/Box/Sphere/Surface/Shape (None).** Click this button, and then select an object in the scene to be used as a reference object. Listed objects are limited to the chosen category.

Note: With the Grid, Box, and Sphere choices, you can use only a grid helper object, a primitive sphere, or a primitive box as a reference object. A primitive sphere or box that has been converted to an editable mesh object can't be used as a reference object. Also, the appropriate radio button (see the first item in this group) must be chosen before you select a reference object.

### Placement in Area group

Contains options for positioning scatter objects in a radial area, without using a reference object.

**In Radial Area.** Distributes the scatter objects randomly in a spherical or circular arrangement, using the remaining controls in this group box.

**Center.** Specifies the center of the distribution in world coordinates.

**Radius.** Specifies the maximum distance from the center within which objects are to be positioned.

**In XY Plane.** Specifies that objects are to be distributed on the world XY plane only, resulting in a disc-like array.

### Spacing group

**Include children's bounding boxes in spacing calculations.** When on, all of a hierarchical scatter object's sub-objects are considered when determining spacing. When off, only the selected object is considered.

**Spacing.** Specifies the minimum distance between scatter objects. The Spacing setting is multiplied by the size of the object's bounding sphere to determine how close objects can get. If Spacing is left at 1.0, the default, objects normally cannot be positioned within each others' bounding spheres. If Spacing is set to 2.0, objects are separated by a distance equal to or greater than the size of the bounding sphere.

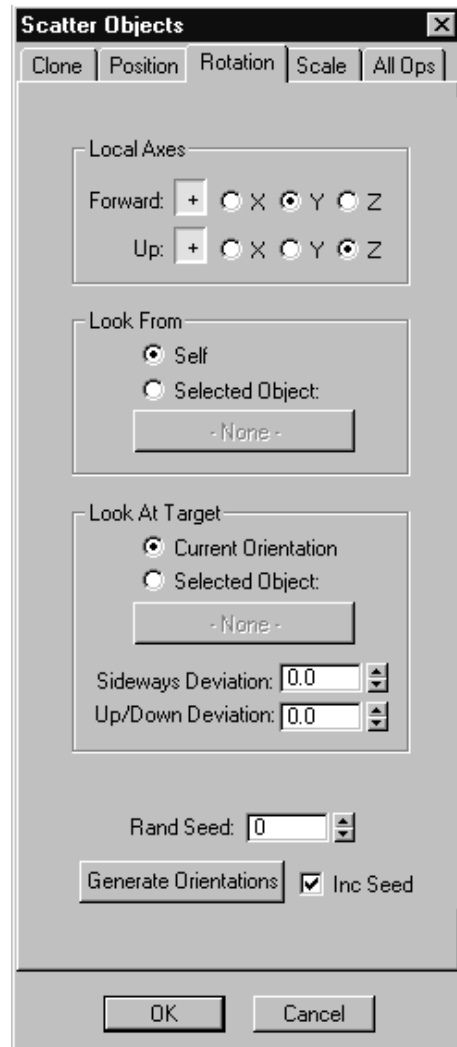
**Rand Seed.** Specifies a seed value for randomizing scatter objects' locations. If, for example, you use the same value for clones of a delegate and then a master object, each pair ends up in the same place. If a scene has more than one crowd, each should use a different seed to avoid having identical configurations.

**Generate Locations.** Click this button to produce a set of locations for all scatter objects;

that is, objects selected with the *Select Objects to Transform* (see page 369) button.

**Inc(rement) Seed.** When on, and you click the Generate Locations button, Scatter adds 1 to the Rand Seed value, and redistributes the objects using the new random seed. Default=on.

### Rotation panel



Contains options for orienting scatter objects. You can specify alternative forward and up axes, plus a target object toward which the objects will point. In addition, you can specify a source object; when using both source and target objects, the objects are rotated so they're parallel to the line between the two.

### Local Axes group

Use these settings to designate alternative forward and up axes. The default settings match the delegate axes.

**Forward:** +/-X/Y/Z. Specifies which axis of the objects points forward, for use with the Look At Target option. When the + button is active, the default condition, the positive chosen axis is used. Click the + button to use the negative axis.

**Up:** +/-X/Y/Z. Specifies which axis of the objects points upward; this axis is aligned with the world Z axis. When the + button is active, the default condition, the positive chosen axis is used. Click the + button to use the negative axis.

Note: You cannot specify the same axis as Local Forward and Local Up simultaneously. If you choose an axis for one that's already chosen for the other, the software switches the other to a different axis.

### Look From group

**Self/Selected Object.** Determines the direction from which the objects look. By default, each object looks from its own position (Self), so that when several objects are looking at a single target, each is oriented differently. To orient each object so that it's parallel to an imaginary line between two objects (the "from" object and the "to" object), choose Selected Object and specify the object with the (None) button.

**None (label).** When choosing Selected Object as the Look From object, use this button to specify the "from" direction. Click the button, and then

select an object from which the objects are to look.

### Look At Target group

#### **Current Orientation/Selected Object.**

Determines the direction toward which the scatter objects look. By default, each object retains its current orientation. To orient each scatter object so that it's parallel to an imaginary line between two objects (the "look from" object and the "look at target" object), choose Selected Object and specify the object with the (None) button.

**None (label).** Use this button to specify the "to" direction. Click the button, and then select an object toward which the scatter objects are to look.

**Sideways Deviation.** Sets a maximum deviation angle in degrees for the objects' sideways orientation. If the scatter objects should look in an object's general direction but may look at a spot to either side of the target, use Sideways Deviation to set the maximum amount by which they can deviate from the calculated angle. The actual deviation amount for each object is calculated at random, based on the Deviation settings and the Rand Seed setting. Range=0.0 to 180.0.

**Up/Down Deviation.** Sets a maximum deviation angle in degrees for the objects' up/down orientation. If scatter objects should look in an object's general direction but may look at a spot above or below the target, use Up/Down Deviation to set the maximum amount by which they can deviate from the calculated angle. The actual deviation amount for each scatter object is calculated at random, based on the Deviation settings and the Rand Seed setting. Range=0.0 to 180.0.

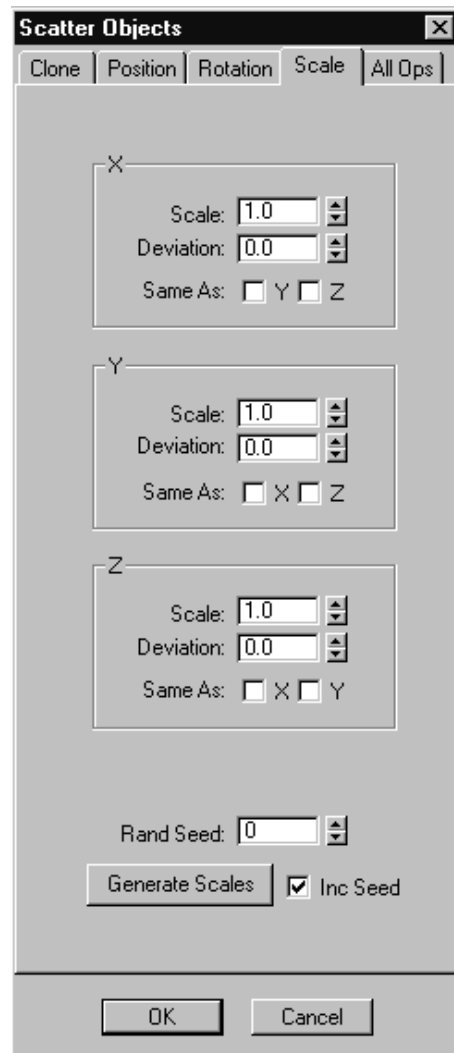
**Rand Seed.** Specifies a seed value for randomizing the scatter objects' orientations,

based on the Deviation settings. If you use the same value for clones of a delegate and then a master object, each pair ends up with the same orientation. If a scene has more than one crowd, each should use a different seed to avoid having identical configurations.

**Generate Orientations.** Click this button to produce a set of orientations for all scatter objects; that is, objects selected with the *Select Objects to Transform* (see page 369) button.

**Inc(rement) Seed.** When on, and you click the Generate Orientations button, Scatter adds 1 to the Rand Seed value, and reorients the scatter objects using the new random seed. Default=on.

### Scale panel



Contains options for scaling scatter objects. You can apply uniform or non-uniform scaling, with optional per-axis deviation for scaling variation.

Each axis group has a “Same As” option that lets you scale that axis by the same amount as another. To prevent non-uniform scaling, set two axes to be the same as the third. For example, set scaling in the X group, and then in the Y and Z groups, turn on Same as X.

**Warning:** These controls can apply non-uniform scaling to objects, which may produce unexpected results when performing other operations within 3DS MAX.

#### X group

**Scale.** Sets scaling on the X axis as a multiplier. Default=1.0.

**Deviation.** Sets the maximum factor for randomization of scaling. For each scatter object, Deviation is multiplied by a random number between 0.0 and 1.0, and then added to the Scale multiplier.

**Same As Y/Z.** Lets you use the same scaling as on the Y or Z axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

#### Y group

**Scale.** Sets scaling on the Y axis as a multiplier. Default=1.0.

**Deviation.** Sets the maximum factor for randomization of scaling. For each scatter object, Deviation is multiplied by a random number between 0.0 and 1.0, and then added to the Scale multiplier.

**Same As X/Z.** Lets you use the same scaling as on the X or Z axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

#### Z group

**Scale.** Sets scaling on the Z axis as a multiplier. Default=1.0.

**Deviation.** Sets the maximum factor for randomization of scaling. For each scatter object, Deviation is multiplied by a random number between 0.0 and 1.0, and then added to the Scale multiplier.

**Same As: X/Y.** Lets you use the same scaling as on the X or Y axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

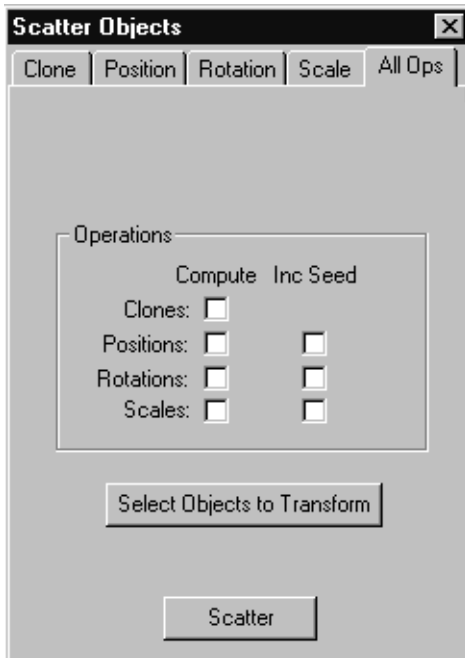
**Rand Seed.** Specifies a seed value for randomizing the clones' scales, based on the Deviation settings. If you use the same value for clones of a delegate and then a master object, each pair ends up with the same scaling factor.

**Generate Scales.** Click this button to produce a set of scales for all scatter objects; that is, objects selected with the *Select Objects to Transform* (see page 369) button.

**Inc(rement) Seed.** When on, and you click the Generate Scales button, Scatter adds 1 to the Rand Seed value, and re-scales the scatter objects using the new random seed. Default=on.



## All Ops panel



This panel lets you perform various permutations of cloning and transform operations in a single step, with or without successive randomization.

### Operations group

**Clones.** Turn on to clone an object. When you click the Scatter button, the object is cloned, and then any specified transforms are applied to the clones.

Turning on Clones makes the Select Objects to Transform button unavailable. The object to clone and cloning parameters must be specified on the *Clone panel* (see page 363).

**Compute Positions/Rotations/Scales.** Any options in this column that are turned on when you click the Scatter button cause the respective transforms to be applied to the current selection

(see Select Objects to Transform, below)

according to the settings in the *Position panel* (see page 364), *Rotation panel* (see page 365), and *Scale panel* (see page 367).

### Inc(rement) Seed Positions/Rotations/Scales.

Any options in this column that are turned on cause the respective Rand Seed settings to be incremented by 1 each time you click the Scatter button.

Use this option to experiment with various randomized transform combinations for your clones.

**Select Objects to Transform.** Lets you designate objects to be affected by clicking the Scatter button.

Clicking this button opens a version of the Select dialog that's unique to Scatter Objects functionality. If you've performed one or more cloning operations during the current session, the results of the most recent cloning are selected by default, including the original cloned object. For example, if you created 10 clones, 11 objects are selected. You can use the Select dialog to alter or replace this selection.

Note: The results of the most recent cloning operation remain selected even if you close and later reopen the Scatter Objects dialog.

**Scatter.** Performs any cloning and/or transforms that are turned on.

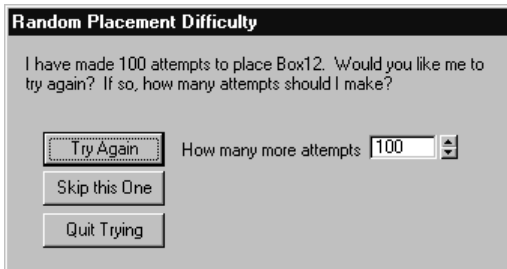
**OK.** Retains all changes and closes the dialog.

**Cancel.** Forgets any changes and closes the dialog.

## Random Placement Difficulty Dialog

This dialog appears when the software encounters difficulty placing cloned objects without overlapping using the *Scatter Objects dialog* (see page 362). The dialog text tells you how many attempts the software has made to place a specific object, and asks you if you want to try again.

### Interface



**Try Again.** Click this button to force the software to make N more attempts, where N is set in the How Many More Attempts field.

**Skip This One.** Instructs the software to stop trying to place the current object and proceed to the next.

**Quit Trying.** Aborts the Generate Locations process; no more objects will be placed.

## Object/Delegate Associations Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Associate Objects with Delegates

Select a Crowd object. > Modify panel > Setup rollout > Associate Objects with Delegates

You can use this dialog to link any number of delegate-object pairs. You can also use this dialog to align objects with delegates, optionally matching scaling factors as well. Lastly, you can specify that the objects use the delegates' controllers.

### Procedure

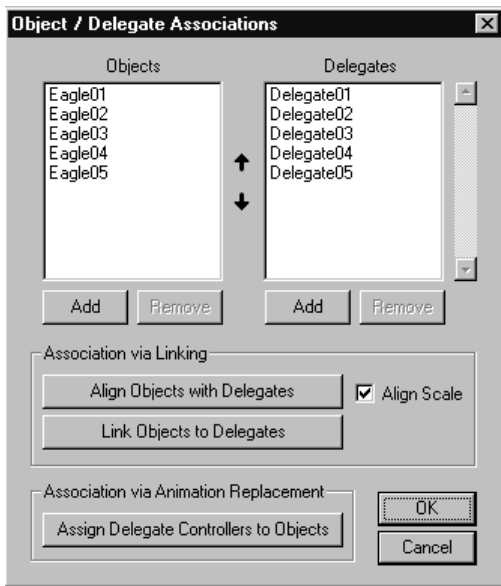
#### To link objects and delegates

1. Use the Add button under the Objects list to add objects to link with delegates.
2. Use the Add button under the Delegates list to add delegates to link with the objects you added in the previous step.

Each object in the Objects list will be associated with the delegate in the same position in the Delegates list. If necessary, reorder either list manually by highlighting entries and using the Shift Up/Shift Down buttons, which are the arrow buttons between the two lists.

3. Click any combination of buttons in the lower area of the dialog.
4. Click OK to exit.

## Interface



**Objects.** Lists objects available for linking, specified using the Add function (see following item). You can select any number of objects from this list for subsequent removal.

**Add.** Click this to open the standard 3DS MAX Select dialog, which lists all objects in the scene, including delegates. Make your selection, and then click the Select button to add the objects to the Objects list.

**Remove.** Deletes the selected object or objects from the list.

**Shift Up/Shift Down.** Use the arrow buttons between the two lists to move highlighted items higher or lower in the list. When Make Specified Associations is on and you click OK, associations are created between pairs of items at matching positions in the lists.

**Delegates.** Lists delegates available for linking, specified using the Add function (see following

item). You can select any number of delegates from this list for subsequent removal.

**Add.** Click this to open the standard 3DS MAX Select dialog, which lists all delegates in the scene. Make your selection, and then click the Select button to add the delegates to the Delegates list.

**Remove.** Deletes the selected delegate or delegates from the list.

### Association via Linking group

**Align Objects with Delegates.** Aligns each object with its corresponding delegate by moving the object to the delegate and rotating the object as necessary to juxtapose their pivot points. Does not link objects hierarchically with delegates.

**Align Scale.** When on, clicking Align Objects with Delegates sets each object's absolute scaling factor to that of its corresponding delegates. This is useful if, for example, you've randomized delegates' sizes with the Scatter Objects *Scale panel* (see page 367), and want the associated objects to match.

**Link Objects to Delegates.** Creates a hierarchy for each object-delegate pair, with the delegate as parent.

### Association via Animation Replacement group

**Assign Delegate Controllers to Objects.** Copies each delegate's controllers to the paired objects as an instance. This is the same as using Track View > Copy Controller from the delegate, and then pasting the controller as an instance to the object. Does not link objects hierarchically with delegates.

Once you've set up the delegate animation the way you want it, if you want to then apply the animation to an object or objects *en masse*, use this function. You can then delete the delegates if you like.

**OK.** Implements any changes and closes the dialog.

**Cancel.** Eliminates any changes and closes the dialog.

Note: Clicking OK has no intrinsic effect; in order to implement any of the dialog functions, you must first click at least one of the Align/Link/Assign buttons.

---

## Edit Multiple Delegates Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Multiple Delegate Editing

Select a Crowd object. > Modify panel > Setup rollout > Multiple Delegate Editing

The Edit Multiple Delegates dialog lets you define groups of delegates and set parameters for them. You can create and store up to 10 different configurations or settings combinations; each consists of one or more delegates and settings for the delegates. The parameters are mostly the same as those found in the delegate object's *Motion Parameters Rollout* (see page 342), with the following exceptions and additions.

First, each setting has an associated SET check box, which lets you determine whether the setting has any effect. When off (the default), the setting has no effect. When on, the setting affects the specified delegate(s).


Second, each numeric parameter has two Value settings and an associated Random check box, which lets you specify a random value within a specified range for each member of the group. By default, Random is off, and the Value 1 setting is applied for all parameters with SET turned on. If you turn on Random for a parameter, its Value 2

setting becomes available. If you then specify a different setting for Value 2, the software calculates a different random number between Value 1 and Value 2 for each delegate in the group.

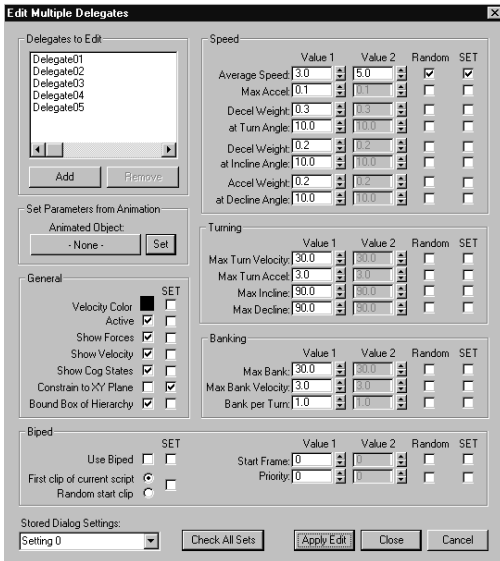
**Tip:** To reset a parameter to a specific value for all delegates in a group after it's been set to a random value within a range, turn off Random and turn on SET for the parameter, then set Value1 to the desired value, and then click OK.

## Procedure

### To edit multiple delegates

1. Select the crowd object and open the Modify panel.
2.  On the Setup rollout, click the Multiple Delegate Editing button.
3. In the dialog, choose a stored dialog setting to use from the 10 available settings.
4. If necessary, use the Delegates to Edit group box controls to add delegates to or remove them from the current setting.
5. Modify the remaining parameters as necessary. Be sure to turn on the SET check box for any parameters that are to change.
6. Click the Apply Edit button to make the changes and exit the dialog.

## Interface



### Delegates to Edit group

This group shows delegates belonging to the current settings combination in a list box and lets you add and delete members.

**Add.** Click this button, and then choose delegates to add from the Select dialog.

**Remove.** To remove delegates from the list, first choose the names of those to delete in the list box (drag to choose two or more contiguous names, or use CTRL + Click to choose non-contiguous names), and then click Remove.

### Set Parameters from Animation group

Use this function to obtain motion parameters from an animated object and apply them to all specified delegates. It affects only Average Speed, Max Accel, and the Turning parameters.

Because this one animation will set most of the parameters of the delegate, it should be representative of a whole range of motion of the

delegate. For example, the object should turn and accelerate. The animation should be somewhat lengthy so that averages are calculated correctly.

**Animated Object.** Specifies an animated object. Click this button, and then select the object from the list in the Select dialog.

**Set.** After specifying the animated object, click this button to apply its parameters to the delegate settings. Also turns on the SET check box for any affected parameters.

### General group

Rather than numeric values, the settings in this group are on-off switches, except for the first, Velocity Color. To change Velocity Color, click the color swatch, use the Color Selector dialog to pick a new color, and then turn on the Velocity Color SET check box. To change any other setting in the General group, click the off-on check box to the right of the setting, and then turn on the setting's SET check box.

### Speed group

These parameters are the same as those found in the delegate object's *Motion Parameters rollout* (see page 342). For an explanation of the Random and SET check boxes, see the introduction to this topic.

### Turning group

These parameters are the same as those found in the delegate object's *Motion Parameters rollout* (see page 342). For an explanation of the Random and SET check boxes, see the introduction to this topic.

### Banking group

These parameters are the same as those found in the delegate object's *Motion Parameters rollout* (see page 342). For an explanation of the Random

and SET check boxes, see the introduction to this topic.

### Biped group

These parameters are the same as those found in the delegate object's *Motion Parameters rollout* (see page 342). For an explanation of the Random and SET check boxes, see the introduction to this topic.

**Stored Dialog Settings.** Use this list to specify up to 10 different combination of delegates and settings. To store a combination, choose a name from the list, and then specify the delegates and settings. To recall a combination, choose its name from the list. To rename a combination, choose its name from the list, and then highlight the name and edit it using the keyboard.

**Check All Sets.** Click this button to turn on all SET check boxes. This ensures that any changes you make in the dialog take effect when you click the Apply Edit button.

**Apply Edit.** Click Apply Edit to implement all changed settings and exit the dialog.

**Close.** Click Close to remember, but not implement, all changed settings and exit the dialog.

**Cancel.** Click Cancel to forget all changed settings and exit the dialog.

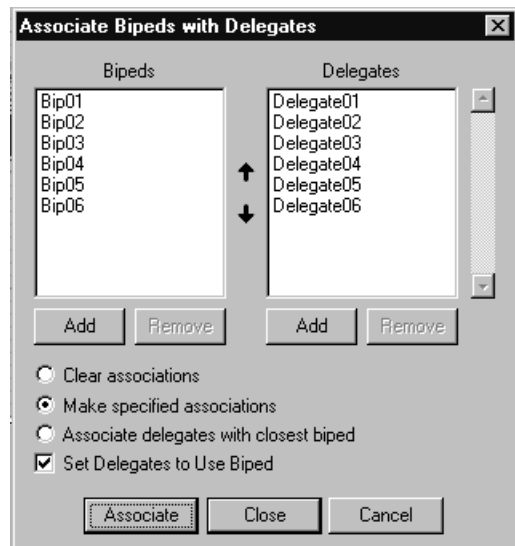
## Associate Bipeds With Delegates Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Edit Multiple Delegates > Biped Associations button

Select a Crowd object. > Modify panel > Setup rollout > Edit Multiple Delegates > Biped Associations button

Use this dialog to associate any number of delegates with an equal number of bipeds. Add delegates and bipeds to the two lists, and order them so the desired pairs are across from each other. Then choose Make Specified Associations and click the Associate button. Alternatively, you can clear existing delegate-biped associations, or simply associate each delegate with the biped nearest it in the scene.

### Interface



**Bipeds.** Lists bipeds available for linking, specified using the Add function (see following item). You can select any number of objects from this list for shifting up or down, or deleting.

To clear highlighted items, control-click them.

**Add.** Click this to open the standard 3DS MAX Select dialog, which lists all bipeds in the scene that are not currently listed in the Associate Bipeds with Delegates dialog. Make your selection, and then click the Select button to add the delegates to the Objects list.

**Remove.** Removes the selected biped or bipeds from the list.

**Shift Up/Shift Down.** Use the arrows between the two lists to move highlighted items higher or lower in the lists. When Make Specified Associations is chosen and you click the Associate button, associations are created between pairs of items at matching positions in the lists.

**Delegates.** Lists delegates available for linking, specified using the Add function (see following item). You can select any number of objects from this list for shifting up or down, or deleting.

To clear highlighted items, control-click them.

**Add.** Click this to open the standard 3DS MAX Select dialog, which lists all delegates in the scene that are not currently listed in the Associate Bipeds with Delegates dialog. Make your selection, and then click the Select button to add the delegates to the Delegates list.

**Remove.** Removes the selected delegate or delegates from the list.

**Clear Associations.** When this is on, and you click the Disassociate button, **character studio** eliminates any delegate-biped associations. Makes the Bipeds list and buttons unavailable.

**Make Specified Associations.** When chosen, and you click the Associate button, **character studio**

associates each parallel delegate-biped pair in the two lists. That is, the first delegate is associated with the first biped, the second delegate with the second biped, and so on.

**Associate Delegates With Closest Biped.** When chosen, and you click the Associate button, **character studio** calculates the biped nearest each delegate in the scene and links that biped to the delegate.

**Set Delegates to Use Biped.** When on, and you click the Associate button, the software turns on the *Use Biped* (see page 345) option for all delegates listed in the dialog.

**Associate/Disassociate.** Implements specified changes, calculates any random values, and closes the dialog. Button text changes to “Disassociate” when the Clear Associations option is chosen.

**Close.** Remembers any changed settings and closes the dialog. No new delegate settings are calculated or applied.

**Cancel.** Closes the dialog and ignores changes.

---

## Behavior Assignments and Teams Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Behavior Assignments

Select a Crowd object. > Modify panel > Setup rollout > Behavior Assignments

The Behavior Assignments and Teams dialog lets you group *delegates* (see page 342) into *teams* (see page 379), and assign *behaviors* (see page 361) and *cognitive controllers* (see page 382) to individual delegates and teams. It also lets you modify existing assignments.

The dialog is modeless; while it's open, you can use the Modify panel to adjust behaviors and set up new behaviors, as well as animate assignments' Weight settings.

Note: Crowd doesn't let you use multiple cognitive controllers with a delegate. You can assign them, but when you solve, the software notifies you that it will use only the first assigned cognitive controller.

Note: The Enable Flashing option helps you see which delegates are affected by different actions in this dialog. When it's turned on, and you perform any of the following, the relevant delegates "flash" (highlight briefly) in the viewports:

- Click a delegate or team in the Assignment Design group
- Click an assignment in the Behavior Assignments group
- Add members to a team in the Teams group
- Remove members from a team in the Teams group

## Procedures

### To group delegates into a team

1. In the Teams group, click the New Team button.
2. Use the Select Delegates dialog to designate the delegates in the team, and then click the OK button.
3. To change a team name, choose it from the drop-down list at the top of the Teams group, click on its name, and then use the keyboard to edit the text.
4. To remove team members, choose the team from the drop-down list at the top of the Teams group, select the members to remove from the lower list, and then click the Remove Members button.
5. To add team members, choose the team from the drop-down list at the top of the Teams group, click the Add Members button, and then use the Select Delegates dialog to designate the delegates to add.

### To create a new behavior assignment

This procedure gives the basic method for assigning a behavior or cognitive controller to a delegate or team.

1. If you want to assign the same behavior to more than one delegate, use the *Teams group* (see page 379) to collect delegates into teams. Note: You can still assign behaviors to an individual delegate, even if it belongs to one or more teams.

2. Make sure no existing assignments in the Behavior Assignments group are highlighted. If any are, Ctrl-click them to clear the selections.

If assignments are highlighted, the software assumes you want to modify the existing assignment(s).

3. In the Assignment Design group, select one delegate or team, and one or more behaviors or one cognitive controller.

You can select only one item from either side of this group, with the exception of behaviors. If you choose multiple behaviors, the software creates a separate assignment for each.

Note: When you select a delegate or team, it briefly highlights in the viewport(s) to indicate the affected delegate(s).

4. Click the New Assignment button. This is the vertical button to the right of the assignment Design group, with five rightward-pointing arrows.

This adds the assignment(s) to the list in the Behavior Assignments group.



5. At this point, you can highlight an assignment, and then change its Weight setting, its Active status, delete it, or change the assignee and/or behavior/cognitive controller.
6. Click OK to accept the changes and close the dialog.

#### To modify an existing behavior assignment or assignments

1. In the Behavior Assignments group, select the assignment(s) to change.  
You can select multiple assignments by Ctrl-clicking for non-contiguous items or Shift-clicking for contiguous items, and then change the assignees or behaviors for all of them at once.
2. To change assignees, in the Assignment Design group, select a delegate or team.
3. To change the assigned behaviors, in the Assignment Design group, select a behavior or cognitive controller.
4. Click the Reset Assignment button. This is the vertical button to the right of the assignment Design group, with five rightward-pointing arrows.
5. Change the Weight setting and Active status as necessary.
6. Click OK to accept the changes and close the dialog.

## Interface

### Assignment Design group



Lets you set up assignments by choosing a behavior or cognitive controller and a delegate or team to assign it to. Choose one item from the upper or lower list on the left, and one item from the upper or lower list on the right. Then click the New/Reset Assignment button immediately to the right of the Assignment Design group (vertical button with five rightward-point arrows).

Note: With the exception of Behaviors, you can choose only one item from either side of this group. To assign the same behavior to more than one delegate, the most efficient method is to use the Teams group to gather delegates into teams.

You can select multiple behaviors for a new assignment to a delegate or team. When you click New Assignment, the software creates a separate assignment for each highlighted behavior. For changing assignments, you're still restricted to choosing one behavior at a time. If you choose an existing assignment and multiple

behaviors, the Reset Assignment button becomes unavailable.

**Delegates.** Lists delegates in the scene.

**Behaviors.** Lists existing behaviors. To use a behavior that hasn't been added to the scene yet, click the New Behavior button at the bottom of this group.

**Teams.** Lists teams in the scene. To create a new team, use the controls in Teams group box.

**Cognitive Controllers.** Lists existing cognitive controllers. To create a new controller, click the Setup rollout > *Cognitive Controllers* (see page 382) button. You needn't first exit the Behavior Assignments and Teams dialog; when you close the editor, the new controller(s) are added to the Cognitive Controllers list.

**Clear Selections.** Deselects all highlighted items in the Assignment Design and Behavior Assignments groups. Use this before modifying an assignment, to avoid possible confusion.

**New Behavior.** Opens the *Select Behavior Type dialog* (see page 380), which lets you add a behavior to the scene for use in an assignment.

To modify a new behavior, use the facilities available in the Crowd object's rollouts.

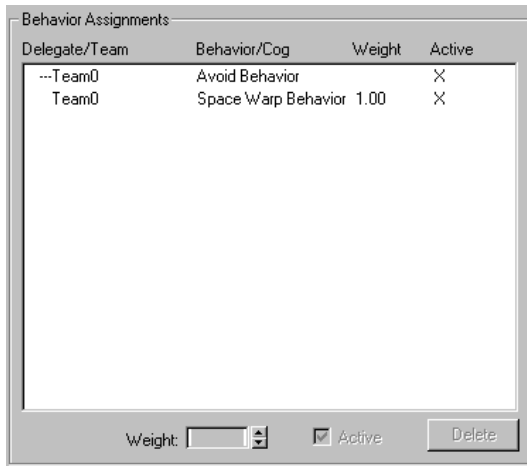
Note: If you add the first behavior in the scene from this dialog, the text box in the Crowd object > Setup rollout remains empty and no rollout for the behavior appears. To edit the behavior, choose it from the list.

**New Assignment/Reset Assignment.** Click to assign a behavior or behaviors or a cognitive controller to a delegate or team.



This vertical button with five right-pointing arrows on it is situated between the Assignment Design and Behavior Assignments group. It's available only when two items in the Assignment Design group are highlighted (exception: Multiple behaviors can be highlighted). If no item in the Behavior Assignments group is highlighted, clicking the button creates a new assignment and adds it to the assignments list. If one or more items in the Behavior Assignments group are highlighted, clicking the button sets the highlighted assignments to use the highlighted delegate/team and behavior/cognitive controller combination.

## Behavior Assignments group



Lets you create and modify behavior assignments.

**List box.** Displays all current behavior assignments, including team or delegate name, assigned behavior or cognitive controller, weight setting, and active status. Items are sorted in alphabetical first by Delegate/Team name, and then by Behavior/Cog name. A dashed line appears before a list entry if it's the first item for that delegate or team.

To modify or delete an assignment, choose it from the list, whereupon the software highlights the assigned components in the Assignment Design group. Make the changes using the remaining controls in this dialog.

Note: You can select multiple assignments from the list by Ctrl-clicking for non-contiguous items or Shift-clicking for contiguous items. To clear an item, Ctrl-click it.

**Weight.** The relative effect of the assigned behavior or cognitive controller. The higher an assignment's Weight setting is than others', the greater relative effect it will have. This setting is animatable. Default=1.0.

In most cases, you should keep Weight within a range of 0.0 to 1.0. Higher settings are available but shouldn't be used unless absolutely necessary.

Note: The Weight setting is not relevant to the *Avoid* (see page 392), *Orientation* (see page 396), and *Surface Follow* (see page 409) behaviors, and is thus unavailable for assignments using those three.

**Active.** When on, the assignment is currently in effect. When off, the assignment has no effect. This check box is animatable. Default=on.

**Delete.** Deletes the highlighted behavior assignment.

## Teams group



Lets you define, modify and delete teams of delegates.

Note: You can toggle the display of this group box with the No Teams/Teams button below the Behavior Assignments group.

**Drop-down list.** Displays the name of the current team. To view a different team, choose it from the list. To change a team name, click in the box and then use the keyboard to edit the text.

**List box.** Displays delegates in the current team.

**New Team.** Adds a team to the list, and opens the *Select Delegates dialog* (see page 381) to let you specify new team members. The default team name is “Team,” followed by a number, starting with “0” and counting up.

**Delete Team.** Deletes the current team.

Team members are not deleted from the scene.

**Add Members.** Lets you add members to the current team. Use the *Select Delegates dialog* (see page 381) to specify new team members.

**Remove Members.** Removes selected members from the team.

Removed members are not deleted from the scene.

**Create/Change Selection Set.** Adds the current team to the list of selection sets, accessible from the Named Selection Sets list on the Main toolbar.

If the current team already is a selection set, and you subsequently changed the team’s makeup, click this button to update the members in the Named Selection Sets list.

**Enable Flashing.** When on, and you click a list item in the dialog or create/modify a team, the relevant objects highlight briefly in the viewports to indicate which are affected. See the *introductory note* (see page 376) for details.

Default=On.

**OK.** Click this button to accept all changes and close the dialog.

**No Teams/Teams.** Toggles display of the Teams group box. By default, the Teams group is displayed; click the No Teams button to turn it off. When it’s off, click the Teams button to turn it on. The state of the toggle persists only during the current session.

---

## Select Behavior Type Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Behavior Assignments > New Behavior

Select a Crowd object. > Modify panel > Setup rollout > Behavior Assignments > New Behavior

Use this dialog to select the type of behavior to be added to a Crowd object. The choices are:

*Avoid Behavior* (see page 392)

*Orientation Behavior* (see page 396)

*Path Follow Behavior* (see page 398)

*Repel Behavior* (see page 400)

*Scripted Behavior* (see page 402)

*Seek Behavior* (see page 403)

*Space Warp Behavior* (see page 404)

*Speed Vary Behavior* (see page 405)

*Surface Arrive Behavior* (see page 406)

*Surface Follow Behavior* (see page 409)

*Wall Repel Behavior* (see page 411)

*WallSeek Behavior* (see page 414)

*Wander Behavior* (see page 416)

## Select Delegates Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Behavior Assignments > Choose or add a team. > Add Members

Select a Crowd object. > Modify panel > Setup rollout > Behavior Assignments > Choose or add a team. > Add Members

The Select Delegates dialog lets you designate delegates to be assigned to teams using the *Behavior Assignments and Teams dialog* (see page 375) for assigning crowd behaviors.

**List box.** Lists all delegates in the scene.

**All.** Selects all delegates in the list.

**None.** Deselects all delegates.

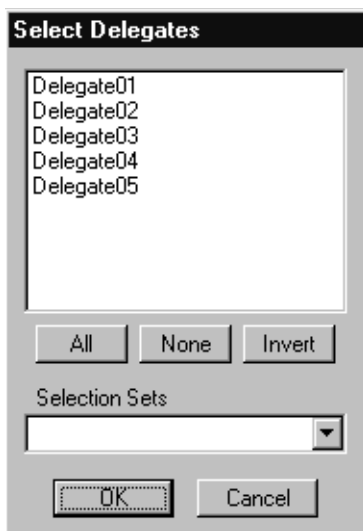
**Invert.** Inverts the current selection.

**Selection Sets.** Choose an item from this to select all members of the selection set. You must first have created a selection set from one or more delegates.

**OK.** Closes the dialog and implements changes.

**Cancel.** Closes the dialog and ignores changes.

### Interface



## Cognitive Controller Editor

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Cognitive Controllers

Select a Crowd object. > Modify panel > Setup rollout > Cognitive Controllers

The Cognitive Controller editor lets you combine behaviors into states. More importantly, it lets you sequence different behaviors (see page 112) and behavior combinations using state diagrams, where conditionals written in MAXScript impose changes in behavior. For example, you can specify that a character or object is to wander aimlessly until it comes within a certain distance of another object, whereupon it heads straight for that object. Or you can specify that one character is to avoid another only when the second character is avoiding the first.

Note: The MAXScript conditionals used in the cognitive controller typically consist of a single line of code. You can load and save them separately as *.ms* files, but they are also stored within the *.max* scene file in which they reside.

The editor interface consists of an icon-based toolbar above a window that contains the state diagram. When you first open the editor, no state diagrams exist. Begin by clicking the New button to create a new state diagram.

**Tip:** If you find yourself consistently assigning two or more behaviors to delegates or teams, you can save time by combining the behaviors into a single-state cognitive controller, or “behavior module,” and assigning that instead. The only disadvantage is that you can’t animate the weights of behaviors used in the cognitive

controller, but you can work around that by using transitions.

Note: Crowd doesn’t let you use multiple cognitive controllers with a delegate. You can assign them, but when you solve, the software notifies you that it will use only the first assigned cognitive controller.

### See also

*State Dialog* (see page 385)


*State Transition Dialog* (see page 386)

*Applying Logic to Crowd Behavior* (see page 564)

## Procedure

### To set up and use a cognitive controller

This procedure describes a typical setup routine for creating and using a cognitive controller. The procedure assumes basic knowledge of crowd simulation setup. For more information about crowd setup, see *Crowd Helper Object* (see page 346) and *Setup Rollout* (see page 360).

1. Create a scene containing a crowd object and one or more delegates.
2. Create at least two behaviors.
3.  Open the Cognitive Controller editor.
4. Click the New button to create a cognitive controller.
5. **character studio** gives the controller the default name of CogControl. It’s recommended that you give more descriptive names to cognitive controllers, such as “Seek/Wander”. Do this by clicking on the name in the text box and editing it from the keyboard.




Creating a new cognitive controller automatically places you in Create State mode.

6. Click in the editor window to create and place a state. Continue clicking in different places to add as many states as necessary.
7. Right-click a state to open the *State editor* (see page 385).
8. Again, it's recommended that you give more descriptive names to states, which you can do in the State editor. Click the name (State or State#) in the text box and edit it from the keyboard.

Next, define a behavior or behaviors for each state.

9. Click the Add button.
10. In the Select Behaviors dialog, choose one or more behaviors.  
If you choose multiple behaviors, you can specify different weights for each in the State editor. For example, you can combine a Seek behavior at full weight with a Wander behavior at half weight, so that the delegate will meander slightly as it seeks the target.
11. Close the Select Behaviors dialog, and then close the State editor.
12. Repeat steps 6-9 as necessary to define behaviors for the other states in the controller.

Next, use Create Transition to define the sequence of states during the simulation.

13. Decide on the sequence in which the states are to occur.
14.  Click the Create Transition button.
15. Drag a line from one state to the next in the order that they are to execute. Click a state to create a transition from itself to itself.

An arrow appears, pointing from the “source” state to the “destination” state.

Each state can have any number of incoming and outgoing transitions. Specify

different transition conditions for each to create as complex a state diagram as necessary.

Next, use the State Transition editor to define a conditional for each transition.

16. Right-click a transition line.
17. In the *State Transition editor* (see page 386) dialog, enter the name of the transition condition, and then click the Edit MAXScript button.
18. Use the MAXScript editor window to enter or load a script that defines the condition(s) under which the transition is to occur.  
Typically, this is a function that tests a condition and returns 1 (if true) or 0 (if false), for example:

- ```
fn test1 del t = (
```
- In plain English, the above statement says that if the delegate's position on the X axis is less than or equal to 40, and it has been in the current state for more than 50 frames, then the transition should occur. However, if either condition is false, or both are, then the delegate should stay in the current state (or test any other transitions). Following is a list of its keywords:
- fn - what follows is a MAXScript function.
  - test1 - the function name; this should also appear in the Transition dialog. This function is executed first when the transition is tested. The script may contain any number of additional functions to be called from within a function in the script.
  - del - refers to the delegate to which the script is currently being applied. The transition script is executed once per frame for each delegate/team member the cognitive controller is assigned to.

Thus, if you use “del” in the script rather than the name of a specific delegate, all delegates to which the cognitive controller is assigned are tested.

- t - the current time (frame number) in the simulation.
- del.simpos.x - the delegate’s current position on the X axis. The special function “simpos” is used to determine a delegate position during a simulation solution. This is necessary because delegate positions aren’t available to MAXScript via the standard “[node].pos” function during a simulation.
- del.duration - the number of frames the delegate has been in the current state.

You can see a complete list of delegate-specific parameters that can be checked in the script by opening a MAXScript Listener window (press function key F11) and entering:

```
ShowProperties $delegate01
```

And because the delegate is a node, it also responds to standard MAXScript node-related functions, with the exception of “simpos,” as noted above. Also, for information on how to access the transition properties, such as duration and priority, see the MAXScript reference.

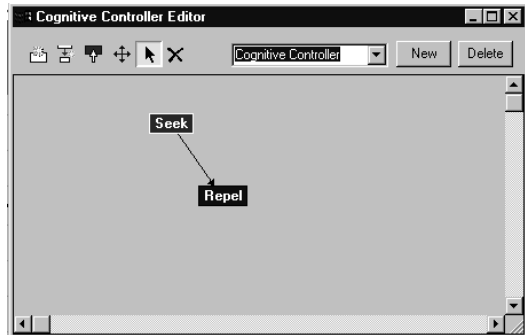
You can find more examples of MAXScript conditionals that can be used with cognitive controllers in the *State Transition editor* (see page 386) section.

19. Use the State Transition editor to set other parameters such as priority and duration.
20. Use the *Behavior Assignments and Teams dialog* (see page 375) to assign the cognitive controller to delegates or teams.

Note: Crowd doesn’t let you use multiple cognitive controllers with a delegate. You

can assign them, but when you solve, the software notifies you that it will use only the first assigned cognitive controller.

## Interface



**Create State.** Lets you create new states in the diagram. Click this button, and then click in the state diagram area to add states. A state appears as a rectangular box containing the name of the state.

The first state you add is, by default, the first state in the controller that executes when the simulation is run. This is indicated by its red color; states you add subsequently are colored blue. To set a different state to execute first, use the Set Start State function.

You specify a state’s name and behaviors by editing the state. To edit a state, right-click it. This opens the *State dialog* (see page 385).



**Create Transition.** Lets you link states with transitions. Click this button, and then drag between two states to create the transition, starting with the earlier state. The transition appears as a black arrow pointing from the first state to the second. Alternatively, if you click a state with the Create Transition tool active, you



create a transition that loops back to the state itself.

Right-click a transition to specify its characteristics and conditions by editing the transition. This opens the *State Transition dialog* (see page 386).



**Set Start State.** Normally the state that executes first in a cognitive controller is the one that was added first. Use this tool to choose a different state to execute first. The start state is red; the rest are blue.

Typically you would use this when you have a circular sequence of states, and you want to change which state executes initially.



**Move State.** Lets you move states around in the window by dragging them.



**Select State/Transition.** Lets you select states and transitions for subsequent deletion. Selected states have white outlines, and selected transition lines are white.

You can select multiple states by dragging a box around them. You can select multiple states and transitions by holding the Ctrl key as you click.



**Delete State/Transition.** Lets you delete states and transitions. First select the state(s) and/or transition(s) to delete, and then click this button.

**(Name).** Shows the name of the current state diagram. To display and/or edit another, choose it from the list.

To change a state diagram's name, click the name in the box and use the keyboard to edit the text.

**New.** Adds a new cognitive controller. By default, cognitive controllers are named

CogControl followed by a number, but you can change this to anything you like.

**Delete.** Deletes the current cognitive controller. This is an undoable operation.

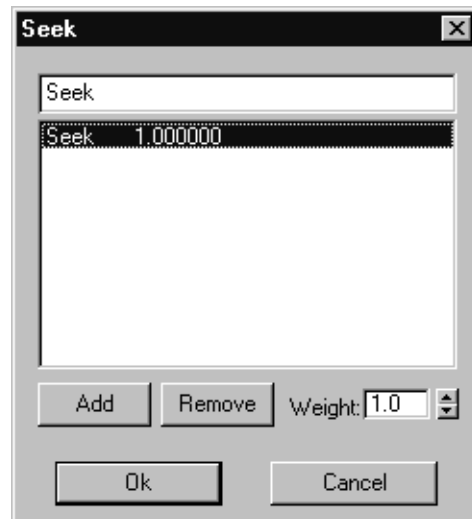
## State Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Cognitive Controllers > right-click a state icon.

Select a Crowd object. > Modify panel > Setup rollout > Cognitive Controllers > right-click a state icon.

The State dialog lets you add and remove behaviors to and from a state, and specify weights for each.

## Interface



**State name.** Displays the name of the state. To change the name, click this text and edit from the keyboard.

**List.** Displays the names of all behaviors associated with the state. To add or remove a state, or change its weight, click the state's name, and then make the appropriate changes.

Note: The order of behaviors in this list is immaterial; all behaviors execute simultaneously.

**Add.** Opens the Select Behavior dialog, which displays the names of all behaviors in the current Crowd object but not associated with the current state. Select a behavior (use Shift and/or Ctrl to choose multiple behaviors), and then click OK.

**Remove.** Deletes the selected behavior from the state.

**Weight.** Specifies the selected behavior's weight. The higher the weight in relation to other behaviors' weights, the more evident the results of the behavior in the state. Default=1.0.

In most cases, you should keep Weight within a range of 0.0 to 1.0. Higher settings are available but shouldn't be used unless absolutely necessary.

---

## State Transition Dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Cognitive Controllers > Right-click a transition line.

Select a Crowd object. > Modify panel > Setup rollout > Cognitive Controllers > Right-click a transition line.

These settings control how the software effects a transition from one state to another when using a *cognitive controller* (see page 382). For more detailed information, see *To set up and use a cognitive controller* (see page 382).

## The Transition Script

The most important element of the transition is the MAXScript conditional script. This is a script associated with the controller that is executed once per frame, and can test any aspect or aspects of the scene and cause a transition or not, depending on the result.

Scripts are executed once per frame per assigned delegate, so objects and effects can be animated and still let delegates react with accuracy.

All scripts used in transitions use the following structure:

```
fn [FunctionName] del t = (
    [MAXScript code]
    if [MAXScript conditional] then 1
    else 0
)
```

The opening section contains "fn" (function) followed by the function name, which also must appear in the State Transition dialog, and then the input parameters "del t", and lastly "= ("". Following this there can be any MAXScript code.

The closing section contains a MAXScript conditional, and then "then 1 else 0". This means: If the result of the conditional is true, then return 1 (that is, the transition is to take place), or if the result of the conditional is false, then return 0 (that is, the transition is not to take place). You could reverse the order of the numbers 1 and 0 ("then 0 else 1") so that the conditional being true would cause no transition to take place, and vice-versa. Lastly, the function must end with a close parenthesis: ")".

Following are some examples of scripts that can be used in cognitive controllers, along with brief explanations. These are presented for you to modify and use in your own scenes.

### To test a particle system parameter

This sample script tests the number of particles emitted by particle system Spray01, and returns positive if the number equals 100.

```
fn TestParticles del t = (
  if (particleCount $Spray01) == 100 then 1
  else 0
)
```

Note that, in testing for an equality, MAXScript uses a double = to signify that it's not an assignment.

### To test an object position

This sample script tests the location of the object named Sphere03, and returns positive if its position is (X>=150, Y>=0, Z>=70).

```
fn PositionCheck del t = (
  if ($sphere03.pos.x >= 150
  and $sphere03.pos.y >=0
  and $sphere03.pos.z >=70) then 1
  else 0
)
```

### To test an atmospheric property

This sample script tests the Density parameter of a fog effect, and returns positive if it equals 50.

```
fn TestAtmos del t = (
  atmos_fog = getAtmospheric 1
  print atmos_fog.density -- to:debug
  if (atmos_fog.density == 50) then 1
  else 0
)
```

Note the second line, which assigns the fog atmospheric to a variable named "atmos\_fog". This is necessary only for atmospheric effects; with most standard objects, you simply use the object name preceded by a \$, as in the two previous examples. The "1" following the getAtmospheric command refers to the atmospheric's position in the Rendering Effects dialog > Effects list.

Once you've executed this assignment, you can obtain a list of the atmospheric's properties by

entering this command in the MAXScript Listener:

```
ShowProperties atmos_fog
```

Also, the third line in the sample script isn't necessary for the cognitive controller; it simply prints the result of the test in the Listener window for debugging purposes.

### To test the distance between two objects

This sample script uses MAXScript's Distance function to obtain the distance between a delegate and a scene object, and returns positive if the result is less than 30.

```
fn TestDist del t = (
  get_dist=distance $sphere01.pos
  $delegate02.simpos
  print (get_dist)
  if get_dist < 30 then 1
  else 0
)
```

As in the previous example, a variable is used; this time, just to keep the script simple. In the second line, the Distance function is used to obtain the distance between a sphere and a delegate, and the result is assigned to the variable "get\_dist". If you wanted to test all delegates that use the cognitive controller instead of a specific one, you'd replace "\$delegate02.simpos" with "del.simpos".

Note: The special property "simpos" is used to determine a delegate position during a simulation solution. This is necessary because newly calculated delegate positions aren't available to MAXScript via the standard "[node].pos" property during a simulation. To get other transformation data calculated during a simulation solution, such as rotation, use the simTransform property. The simTransform property is documented in the MAXScript reference.

In the third line of the script, the calculated distance is printed to the Listener window, for debugging purposes. This line is not necessary for the simulation and can be deleted. Lastly, the value is compared with the constant 30, and if it is less, the script returns a 1, indicating that the transition is to take place.

You can draw on this script to create a cognitive controller that uses multiple Seek behaviors/states to move delegates along a path among any number of objects. As soon as the delegate is within a given distance of a specified object, the transition takes place and the delegate starts using the next Seek state, thus moving toward the next object. Because the transition is tested at each frame, the target objects can be moving in any way you like, resulting in a seemingly dynamic animation.

#### To test a modifier parameter

This sample script checks the Angle parameter of a Bend modifier applied to a cylinder, and returns true if it is between 70 and -70, inclusive.

```
fn TestBend del t = (
if ($cylinder01.bend.angle <= 70 and
$cylinder01.bend.angle >= -70 )
then 1
else 0
)
```

Note that the If statement in the second line uses parentheses around the text because two conditions are checked: whether the angle is less than or equal to 70, and whether the angle is greater than or equal to -70. Because of the “and” between them the script returns true only if both are true.

#### To test another delegate’s behavior

You might want to determine in a transition script which behavior is currently influencing a certain delegate. Crowd provides a MAXScript-based method for doing this. You can even check

whether a particular delegate is specified as a target within that behavior. An example would be a cocktail party scene in which Betty avoids Harry if Harry is seeking Sally. But if Harry is avoiding Sally, then Betty will seek Harry.

The following example script is taken from the sample file *party.max*, which you can find in *cstudio\tutorials\tutorial\_8* directory in your 3DS MAX path. The scene uses a more complex scenario than the example described in the previous paragraph. Following is an overview, but to fully understand the setup, you should examine the scene. Study, in particular, the behavior assignments and cognitive controllers, which use a total of eight different transition scripts.

Six delegates are confined in a “room” defined by four grids, using a Wall Repel behavior. Delegates 1, 2, 3, and 5 simply wander at random during the simulation. However, delegate 4 uses a cognitive controller (cc1) that tells it to start wandering, and then switch to one of three Avoid behaviors if members of one of three arbitrary pairs of delegates come within 50 units of each other. Each of the Avoid behaviors targets a different group of three delegates, two of which include delegate 2. Delegate 6 is assigned a second cognitive controller (cc2) that uses the following script to tell it to switch to an Avoid behavior if delegate 4 is avoiding delegate 2. The heart of the script is this line in function *transfunc4*:

```
(isDelAvoid = isDelegateAvoiding
the_current_behavior.name "$Delegate04"
"$Delegate02")
```

Load the file, press F11 to open the Listener window, and then solve. The Listener window displays a message whenever delegate 4 is found to be avoiding delegate 2.

You can use this script as is in your own simulations to check for whether one delegate is

avoiding a second by substituting the delegates' names in the above line, and also substituting the names of your Avoid behaviors in the list in *transfunc4*, adding or deleting lines as necessary.

The example script illustrates a second important point: Cognitive controller transition scripts can contain multiple functions. Crowd first executes the function specified in the State Transition dialog > Transition Condition field, and that function calls one or more additional functions in the script, which, of course, can also call functions. In this case, *transfunc4* calls the first function, *isDelegateAvoiding*, passing it three parameters.

Lastly, the script contains a special function, *getBehaviorType*, that compares an input

behavior against a list of known behaviors, and on a match, returns the known behavior. In this case, *transfunc4* runs through the list of behaviors currently influencing Delegate04, testing each with *getBehaviorType*, and if an Avoid behavior is in effect, proceeds to check whether Delegate02 is an obstacle of that Avoid behavior. Use of this function is more efficient and flexible than testing for specific behaviors, especially if your scene contains many behaviors of the same type, or you're constantly editing behavior settings. You can see the returned behaviors by removing the comment (double hyphen) from the beginning of the following line in *transfunc4*.

```
-- format "Return Behavior: %\n" return_behavior

fn isDelegateAvoiding theCurrentBehavior theCogDelegate theAvoidingDelegate = (

  the_return = 0
  counter = 1
  for the_assignments in $Crowd01.assignments do
  (
    if the_return == 1 then exit

    if the_assignments.delegate != undefined then
    (
      if theCogDelegate == "$"+the_assignments.delegate.name then
      (
        if the_assignments.cogcontrol != undefined then
        (
          for the_cogcontrol_state in the_assignments.cogcontrol.states do
          (
            if the_return == 1 then exit

            for the_cogcontrol_state_behavior in the_cogcontrol_state.behaviors do
            (
              if the_return == 1 then exit

              if the_cogcontrol_state_behavior.name == theCurrentBehavior then
              (
                for the_obstacle in the_cogcontrol_state_behavior.obstacles do
                (
                  if the_return == 1 then exit
```



```

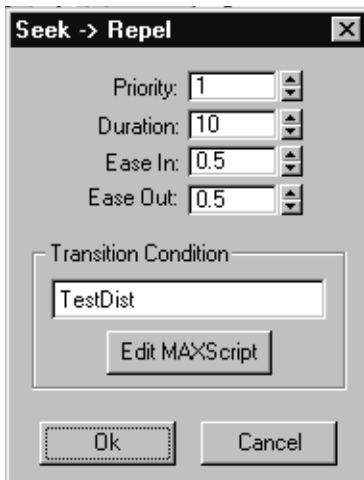
return_behavior = getBehaviorType the_current_behavior
-- format "Return Behavior: %\n" return_behavior
if return_behavior == Avoid_Behavior then
(
    isDelAvoid = isDelegateAvoiding the_current_behavior.name "$Delegate04" "$Deleg
ate02"

    if isDelAvoid == 1 then
    (
        format "$Delegate04 found to be Avoiding $Delegate02\n"
        format "          Starting Transition in frame %:\n" t
        another_the_return = 1
    )
)

counter = counter + 1
)
another_the_return
)

```

## Interface



**Priority.** Sets the transition's precedence.

When more than one transition tests true, the software uses the Priority setting to determine which transition occurs. It performs the transition with the lowest Priority setting. Thus, for example, a transition with a Priority setting

of 0 takes precedence over one with Priority 1, and so on.

**Duration.** The number of frames the software takes to effect the transition between states.

**Ease In.** The rate at which the transition begins. Lower values cause a more abrupt transition, while higher values cause a more gradual transition. Default=0.5. Range=0 to 1.0.

**Ease Out.** The rate at which the transition ends. Lower values cause a more abrupt transition, while higher values cause a more gradual transition. Default=0.5. Range=0 to 1.0.

Note: The sum of the values for Ease In and Ease Out must be less than or equal to 1.0. The software won't let you set a value for either parameter that would cause the sum to exceed 1.0. To increase the value of one parameter when its value equals 1.0 minus the other parameter, decrease the other parameter first.

**Transition Condition.** The name of the MAXScript function that specifies when/how the transition is to occur. This name must also

appear at the beginning of the main function in the script, after “fn”. The script can contain additional functions that are called by the main function and each other.

**Edit MAXScript.** Opens an editor window for editing, saving, and loading the transition’s MAXScript script.

---

## Behaviors

### Behavior Rollout

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button

Select a Crowd object. > Modify panel > Setup rollout > New button

Select a Crowd object. > Modify panel > Setup rollout > Choose a behavior.

The Behavior rollout appears in the Crowd object command panel after you first add a behavior to the scene using the Setup rollout. The rollout’s full name (for example: Avoid Behavior or Seek Behavior) depends on the current behavior, displayed in the text box in the Setup rollout. To display the rollout for a different behavior, choose the behavior from the list.

Note: If you add the first behavior to the scene from the *Behavior Assignments and Teams Dialog* (see page 375), a behavior rollout does not automatically appear in the Crowd command panel. You must first choose the behavior from the list at the bottom of the Setup rollout.

Use the controls in this rollout to modify the behavior. Following is a list of available behaviors:

*Avoid Behavior* (see page 392)

*Orientation Behavior* (see page 396)

*Path Follow Behavior* (see page 398)

*Repel Behavior* (see page 400)

*Scripted Behavior* (see page 402)

*Seek Behavior* (see page 403)

*Space Warp Behavior* (see page 404)

*Speed Vary Behavior* (see page 405)

*Surface Arrive Behavior* (see page 406)

*Surface Follow Behavior* (see page 409)

*Wall Repel Behavior* (see page 411)

*WallSeek Behavior* (see page 414)

*Wander Behavior* (see page 416)

For a detailed description of specific behaviors, refer to the above topics. For an overall look at behaviors, see *Crowd Behaviors Overview* (see page 112).

---

### Avoid Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Avoid Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Avoid Behavior

The Avoid behavior lets you specify any object or objects that delegates must keep away from. As delegates approach designated objects during the crowd simulation, they steer clear of them while turning and/or braking as necessary. This behavior uses three different methods to let delegates avoid each other and other objects: Steer To Avoid, Repel, and Vector Field.

Steer to Avoid behavior is best used for animals that steer around each other at close proximity. Legged animals and fish typically do this.



Steering motion may be sudden since its action is often engaged for relatively short periods of time.

By contrast, Repel avoidance behavior mimics the continuous action of a repellent magnetic field. Birds, bats and flying insects are best animated with large Repel fields so that they can smoothly avoid each other while maintaining a comfortable margin of error. Repel forces prevent intrusion from all sides, regardless of the direction of travel. Thus even animals that rely mainly on Steer to Avoid will also need some degree of Repel avoidance to maintain spatial separation when they are moving through dense traffic. The forces of Repel avoidance are always directed uniformly outward in a spherical shape.

Use Vector Field avoidance for cases where crowd members must avoid hitting the more complex shapes of arbitrary MAX objects. The outward forces of the Vector Field avoidance may be constructed to form the shape of any MAX object. For example, suppose you want to animate a school of fish swimming around a sunken ship. In this case, a vector field can be created so that it extends the shape of the ship into the surrounding space. The field is computed by scanning and converting the ship's surface normals into a 3D lattice that surrounds the ship. These normals will extend into space as "beacons" in the 3D lattice, telling the fish how to best swim away from the shape of the ship. As the fish enters the space of the vector field lattice, it can be precisely repelled along an avoidance force directed away from the ship's surface.

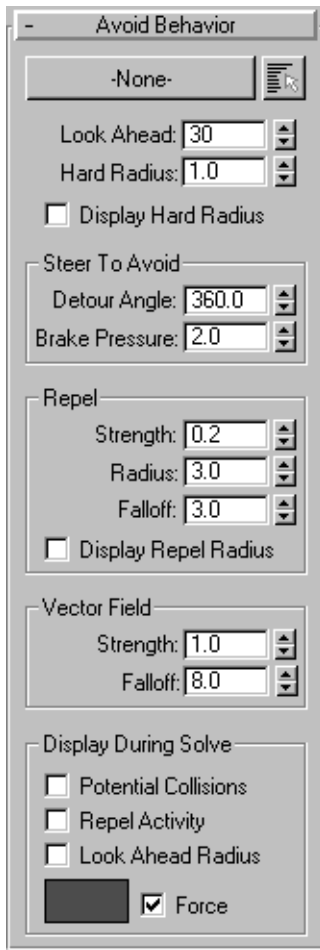
Note: In the explanations that follow, the word "target" is used to refer to the object or objects to be avoided.

## Procedure

### To use the Avoid behavior

1. Add an Avoid behavior to the Crowd object.
2. In the Avoid Behavior rollout, use the None button or the Multiple Selection button to designate the target object or objects to avoid.  
**Tip:** When avoiding multiple objects with a team, it's okay for an object to avoid itself.
3. Change the default settings as desired.
4. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Avoid Behavior rollout

**None (label).** Specifies a single target. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you've selected multiple targets using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are

designated as targets, click the Multiple Selection button.

**Multiple Selection.** Opens the Select dialog to let you designate multiple targets. When you have more than one target, you can set delegates to move toward the closest target in the group, or to a computed average of the target positions.

**Look Ahead.** The number of frames in advance of the current frame that the software looks for potential collisions. Default=30.

**Hard Radius.** Distance from the target's pivot point, in multiples of the delegate's bounding sphere, where no penetration should occur. Default=1.0.

**Tip:** Because the hard radius' center is the pivot point, Avoid may not work as expected with target objects whose pivot point is not centered, such as the box primitive. For best results, use the Hierarchy panel > Adjust Pivot rollout controls to center the pivot to the object.

**Display Hard Radius.** Enables display of a wireframe sphere that depicts the extent of the Hard Radius setting. Default=off.

### Steer To Avoid group

Steer To Avoid is used by delegates to steer precisely around anticipated future collisions based on the delegates' current speed and direction. Delegates using this approach can pass very close to one another.

**Detour Angle.** Maximum necessary turning angle relative to the direction of delegate's goal that delegate will steer to avoid rather than slow down and wait. Default=360. Range=0-360.

**Tip:** To disable turning for avoidance, thus allowing only braking, set Detour Angle to 0. This forces delegates to remain directed toward their goal so that they must slow down and wait until there is a clearing in front of them, much

like an audience queuing to leave through an exit after a concert.

**Brake Pressure.** Determines how strongly the brakes are applied. Higher values induce more gradual and slower stops, but may cause brakes to “pump” in order to slow down. A value of 0 disables the brakes. Default=2.0.

### Repel group

Repel is a general separation force that is based only on the spatial position. Delegates use this to keep from getting into situations where they might side-swipe each other or where they might get so close that Steer To Avoid is too difficult to achieve.

**Strength.** Determines the strength of the repelling force; higher values result in greater repulsion force. Default=0.2. Range=0.0 to 1.0.

**Radius.** Maximum distance from delegate's bounding sphere within which “repel” avoidance is sensed and carried out. Default=3.0.

**Falloff.** The rate at which the strength diminishes between the Repel radius and the hard radius. A value of 1.0 indicates a linear falloff. Higher values cause the strength to fall off to zero more rapidly with distance, thus focusing its effect closer to the delegate's hard radius. Lower values reduce the rate of diminishment, with a Falloff setting of 0.0 indicating that the strength is the same at the Radius distance as it is at the Hard Radius. Default=3.0.

**Display Repel Radius.** Enables display of a wireframe sphere that depicts the extent of the Repel setting. Default=off.

### Vector Field group

If you've applied a *Vector Field space warp* (see page 417) to an object in your scene, you can specify the vector field as an object to avoid. The distinction is this: When used with the *Space*

*Warp behavior* (see page 404), delegates use the vector field to steer around the object by being guided to travel perpendicular to the field's vectors. When used with the Avoid behavior, the delegate simply moves away in the direction of the vectors.

**Tip:** Sometimes when using Avoid with a vector field, the behavior might seem to be “fighting” with other behaviors (such as Seek) over delegate movement, causing a halting and/or wavering motion. In such cases, try reducing Brake Strength and/or increasing Falloff.

**Strength.** Higher values result in more powerful influence. Delegates will be directed to move perpendicular to the field. Default=1.0. Range=0.0 to 1.0.

**Falloff.** Higher values cause vector field influence to fall off to zero more rapidly with distance, thus focusing its effect closer to the delegate's hard radius. Default=8.0.

### Display During Solve group

Use these switches for debugging a crowd simulation. During the solve, they display information about the simulation using graphical metaphors for different aspects of the Avoid behavior.

**Potential Collisions.** Displays a green line from the delegate to the location of a potential collision. Default=off.

**Repel Activity.** Displays a white line between the delegate and target when the repel force is in effect. Default=off.

**Look Ahead Radius.** Displays a sphere that shows the current distance used to check for potential collisions.

**Color Swatch.** Shows the color used to draw the Avoid force vector during the solution. Click the box to choose a different color. Default=red.

**Force.** When on, force exerted on the delegate(s) by the Avoid behavior is drawn in the viewports as a colored line during the simulation solution. Default=on.

## Orientation Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Orientation Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Orientation Behavior

The Orientation behavior lets you control whether and how delegates rotate, independent of their direction of motion. Normally, a delegate always faces in the direction it's moving. You can use the Orientation behavior to specify limits to the delegate's rotational activity without affecting its path, which is generated by other behaviors. Use these settings, for example, to keep delegates facing in one direction while moving in another.

Note: These settings do not affect the path a delegate takes, which is produced by other behaviors such as Seek and Avoid. They influence only the direction it faces as it traverses the path.

## Procedure

### To use the Orientation behavior

1. Add an Orientation behavior to the Crowd object.
2. To restrict the heading orientation with respect to the delegate's current heading, turn on Heading group > Relative. Otherwise, to restrict the heading to a specific direction or range, leave Relative off.
3. To restrict the pitch orientation with respect to the delegate's current pitch, turn on Pitch group > Relative. Otherwise, to restrict the pitch to a specific direction or range, leave Relative off.
4. Change the other default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface

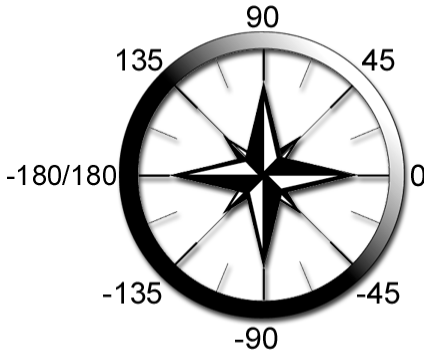


## Orientation Behavior rollout

### Heading group

Use these controls to affect how delegates turn on the vertical axis. By default, heading is absolute, with 0 specifying the positive X axis in World coordinates. Thus, -90 would specify the

negative Y axis, 90 the positive Y axis, and 180 or -180 the negative X axis.



#### Absolute heading as viewed from the top

For example, if you wanted a delegate to be able to turn between the positive X axis and the positive Y axis, you would set Max Heading to 0 and Min Heading to 90.

You can also specify heading limits in amounts relative to the delegate's heading at the time that the Orientation behavior takes effect by turning on the Relative check box.

**Relative.** When on, Heading settings are applied relative to the delegate's heading at the time the behavior takes effect. When off, settings are absolute. Default=off.

**Min Heading.** The minimum permissible heading. This number should be lower than the Max Heading value. Default=180. Range=-180 to 180.

**Max Heading.** The maximum permissible heading. This number should be higher than the Min Heading value. Default=180. Range=-180 to 180.

**Max Heading Vel(ocity).** Specifies how much the delegate's heading can change per frame. This controls angular acceleration and deceleration. Default=180.

**Head. Response:** Determines how quickly the heading follows the direction the object is moving in. A value of 1.0 indicates maximum responsiveness, and will point in the direction the delegate is moving (within the limits) while a lower value means that it is less responsive. Default=1. Range=0 to 1.

#### Pitch group

Use these controls to affect how delegates turn on the left-right axis.

**Relative.** When on, Pitch settings are applied relative to the delegate's pitch at the time the behavior takes effect. When off, settings are absolute. Default=off.

**Min Pitch.** The minimum number of degrees a delegate can incline or decline. This number should be lower than the Max Pitch value. Default=-180. Range=-180 to 180.

**Max Pitch.** The maximum number of degrees a delegate can incline or decline. This number should be higher than the Min Pitch value. Default=180. Range=-180 to 180.

**Max Pitch Velocity.** Specifies how much the delegate's pitch can change per frame. This controls angular acceleration and deceleration. Default=180.

**Pitch Response.** Determines how quickly the pitch follows the direction the object is moving in. A value of 1.0 indicates maximum responsiveness, so that will point in the direction the delegate is moving (within the limits) while a lower value means that it is less responsive. Default=1. Range=0 to 1.

### Banking group

Use these controls to affect how delegates turn on the in-out axis. Banking is primarily a result of heading changes.

**Max Bank.** The maximum number of degrees the delegate can bank. Default=30.0.

**Max Bank Velocity.** The maximum number of degrees the delegate's bank angle can change per frame. This controls angular acceleration and deceleration. Default=3.0.

**Bank per Turn.** The number of degrees the delegate will bank as a function of the turn angle at the current frame. For example, if Bank per Turn=1, the delegate will bank one degree for every degree it is turning at a given frame. Default=1.0.

---

## Path Follow Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Path Follow Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Path Follow Behavior

The Path Follow behavior lets you direct delegates to traverse a specified path during a crowd simulation. Delegates can move forward or backward along paths, and when they reach the end, they can loop back to the start or reverse direction, or even continue in the same general direction.

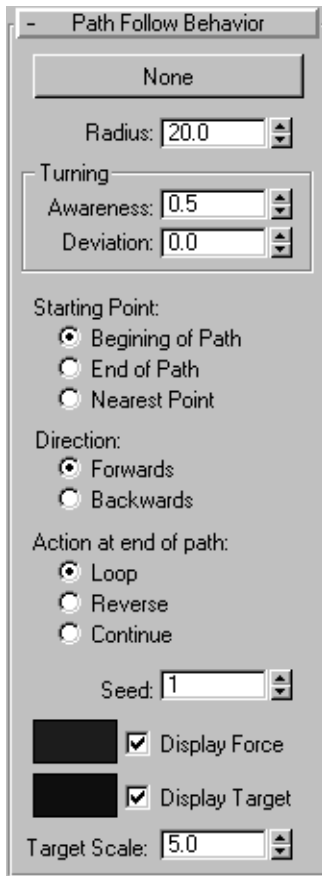
If the delegate's start position isn't on the path at the start of the simulation, it moves to the path before following the path. During the solution, **character studio** intermittently displays an optional *target* icon to show the delegate's immediate goal; this changes as the simulation proceeds.

## Procedure

### To use the Path Follow behavior

1. Add a Shape object to be used as a path for delegates.
2. Add a Path Follow behavior to the Crowd object.
3. In the Path Follow Behavior rollout, use the None button to designate the shape to follow.
4. Change the default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Path Follow Behavior rollout

**None (label).** Click this button, then select a path object. Suitable path objects include splines and NURBS curves. If a path object contains more than one spline or curve, **character studio** uses the lowest-numbered element (usually the earliest created one).

Note: You can assign a path object only from the Modify panel.

**Radius.** The radial distance from the path within which the delegate stays while traversing the path. Default=20.0. Range=0.0 to 99,999.0.

### Turning group

These parameters determine how delegates turn while following the path. Awareness determines how well a delegate anticipates turns in the path as it moves; you can apply random variation to Awareness with the Deviation setting.

**Awareness.** Specifies how “intelligent” the delegate is while traversing this path. A high Awareness setting means that it takes into account the curve of the path while moving and will try to anticipate changes. A low value for Awareness, on the other hand, means that the delegate notices the path only when leaving it. Default=0.5. Range=0.0 to 1.0.

Note: You can randomize awareness behavior with the Deviation and Seed settings.

**Deviation.** Specifies the maximum amount by which Awareness should vary. **character studio** takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Awareness setting, and adds the result to Awareness. Default=0.5. Range=0.0 to 1.0.

Note: You can vary behaviors among different Path Follow behaviors that use the same Awareness and Deviation settings by changing the Seed value.

### Starting Point

Determines where on the path the delegate begins to follow the path. The default choice is Beginning of Path.

**Hint:** To see a selected spline path’s start point, open the Modify panel and turn on any sub-object level; the start point is represented with a unique indicator. Also in the Modify panel, with closed curves, you can see the vertex ordering at

any sub-object level by turning on Selection rollout > Display group > Show Vertex Numbers. To see a NURBS curve's start point, go to the Curve sub-object level; the start point is indicated by a small green circle.

**Beginning of Path.** The delegate first moves to the start of the path before following it.

**End of Path.** The delegate first moves to the end of the path before following it. With closed curves, this is the same point as the beginning of the path.

**Nearest Point.** The delegate first moves to the closest point on the path and then follows the path from there.

#### Direction

Determines the direction the delegate takes initially when following the path. The default choice is Forwards.

**Forwards.** The delegate moves along path vertices in ascending order.

**Backwards.** The delegate moves along path vertices in descending order.

#### Action at End of Path

Determines what the delegate does when it reaches the path end. The default choice is Loop.

**Loop.** The delegate loops around the path, even if it isn't closed. If Beginning of Path or End of Path is chosen, it returns to the path's start or end point each time it finishes traversing the path. If Nearest Point is chosen, it returns to an arbitrary point determined by its position and the path shape.

**Reverse.** The delegate reverses direction at the end of the path. Use this choice to simulate a back-and-forth "patrol" behavior.

**Continue.** The delegate continues moving in the same direction it faced at the end of the path

until the simulation ends or it's acted upon by another force or behavior.

**Seed.** Specifies a seed value for randomizing Awareness. Default=1.

**Color Swatch.** Shows the color used to draw the Path Follow force vector during the solution. Click the box to choose a different color. Default=blue.

**Display Force.** When on, force exerted on the delegate(s) by the Path Follow behavior is drawn in the viewports as a vector during the simulation solution. Default=on.

**Color Swatch.** Shows the color used to draw the target icon. Default=dark blue.

**Display Target.** Enables display of the target icon, which appears during the solution when a new interim goal is calculated for the delegate. Default=on.

**Target Scale.** Specifies the overall size of the target icon. Default=5.0.

---

## Repel Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Repel Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Repel Behavior

The Repel behavior lets you specify any object or objects (sources) that will force delegates to move away from them. This is basically the opposite of the *Seek behavior* (see page 403). If you want delegates to back away from an object, as opposed to turning to face the direction they're moving, use Repel in conjunction with the *Orientation behavior* (see page 396).



Note: Repel is set by default to work only within a specific radius around the source. If you want it to work at any distance, turn off Radius group > Use Radii.

## See also

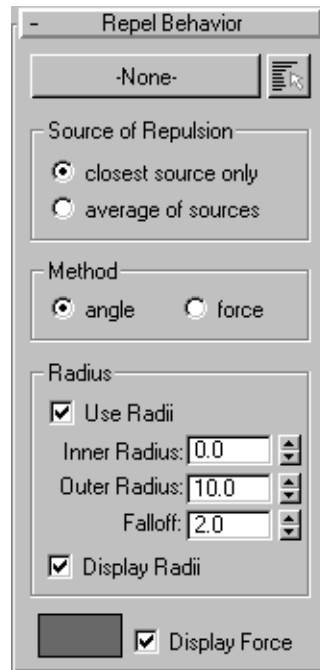
*Wall Repel Behavior (see page 411)*

## Procedure

### To use the Repel behavior

1. Add a Repel behavior to the Crowd object.
2. In the Repel Behavior rollout, use the None button or the Multiple Selection button to designate the object or objects that are to repel delegates.
3. Change the default settings as desired.
4. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



## Repel Behavior rollout

**None (label).** Specifies a single source. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you've selected multiple sources using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are designated as sources, click the Multiple Selection button.

**Multiple Selection.** Opens the Select dialog to let you designate multiple sources. When you have more than one source, you can set delegates to move toward the closest target in the group, or to a computed average of the source positions.

### Source of Repulsion group

Determines repel activity when the behavior uses multiple sources. The default choice is Closest Source Only.

**Closest Source Only.** Each delegate is repelled by the closest of the assigned sources. Use this to have delegates assigned a single Repel behavior move away from sources in different directions.

**Average Of Sources.** All delegates move away from a common point determined by averaging all sources' locations.

### Method group

Determines whether delegate direction as influenced by the behavior is calculated by an angular method or a force method.  
Default=Force.

**Angle.** Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly away from the source.

The magnitude of the force is greatest when the delegate is moving directly towards the source, and needs to turn around. It can be as little as 0 when the delegate is moving directly away from the source.

**Force.** Always applies a force directly away from the source. The magnitude of the force is constant.

### Radius group

Use the Radius settings to activate the Repel behavior only when the delegates are within a specific distance from the target. The relative strength of the force increases from 0 percent at the outer radius to 100 percent at the inner radius.

**Use Radii.** When on, the behavior applies only to delegates closer to the target than the Outer Distance value. Default=on.

**Inner Radius.** The distance from the target at which the force is applied at full strength. Default=0.0.

**Outer Radius.** The distance from the target at which the force begins to be applied. Default=10.0.

**Falloff.** Default=2.0.

**Display Radii.** The radii are displayed when the force is active.

**Color Swatch.** Shows the color used to draw the Repel force vector during the solution. Click the box to choose a different color. Default=violet.

**Display Force.** When on, force exerted on the delegate(s) by the Repel behavior is drawn in the viewports as a vector during the simulation solution.

---

## Scripted Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Scripted Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Scripted Behavior

The Scripted behavior lets you create custom behaviors using MAXScript. A scripted behavior can incorporate one of three behavior types: Force, Constraint, or Orientation.

For detailed information on scripting behaviors, see the MAXScript reference.

## Interface



### Scripted Behavior rollout

**Behavior Type.** Choose Force, Constraint, or Orientation.

**Script Context Name.** Specify a name for the script.

**Edit MAXScript.** Click to open an editor window.

## Seek Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Seek Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Seek Behavior

The Seek behavior lets you specify any object or objects as a stationary or moving target for delegates. Delegates move toward the target during the crowd simulation while turning as necessary.

### See also

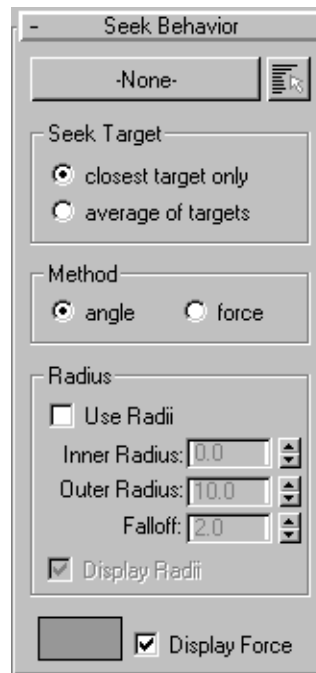
*WallSeek Behavior* (see page 414)

## Procedure

### To use the Seek behavior

1. Add a Seek behavior to the Crowd object.
2. In the seek Behavior rollout, use the None button or the Multiple Selection button to designate the object or objects to be sought.
3. Change the default settings as desired.
4. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Seek Behavior rollout

**None (label).** Specifies a single target. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you've selected multiple targets using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are designated as targets, click the Multiple Selection button.

**Multiple Selection.** Opens the Select dialog to let you designate multiple targets. When you have more than one target, you can set delegates to move toward the closest target in the group, or to a computed average of the target positions.

### Seek Target group

Determines seek activity when the behavior uses multiple targets.

**Closest Target Only.** Each delegate seeks the closest of the assigned targets. Use this to have delegates assigned a single Seek behavior move in different directions.

**Average Of Targets.** All delegates move toward a common point determined by averaging all targets' locations.

### Method group

Determines whether delegate direction as influenced by the behavior is calculated by an angular method or a force method.

Default=Angle.

**Angle.** Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly toward the target.

The magnitude of the force is greatest when the delegate is moving away from the target, and needs to turn around. It can be as little as 0 when the delegate is directly approaching the target.

**Force.** Always applies a force directly towards the target. The magnitude of the force is constant.

### Radius group

Use the optional radius settings to activate the Seek behavior only when the delegates are

within a specific distance from the target. The relative strength of the Seek behavior increases from 0 percent beyond the outer radius to 100 percent at the inner radius.

**Use Radii.** When on, the Seek behavior applies only to delegates less than the Outer Radius distance from the target. Default=off.

**Inner Radius.** The distance from the target at which Seek is applied at full strength. Default=0.0.

**Outer Radius.** The distance from the target at which Seek begins to be applied. Default=10.0.

**Falloff.** Default=2.0.

**Display Radii.** The radii are displayed when the force is active.

**Color Swatch.** Shows the color used to draw the Seek force vector during the solution. Click the box to choose a different color. Default=green.

**Display Force.** When on, force exerted on the delegate(s) by the Seek behavior is drawn in the viewports as a vector during the simulation solution. If Use Radii is turned on, the radii are also displayed when the force is active.



## Space Warp Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Space Warp Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Space Warp Behavior

The Space Warp behavior lets you assign a space warp, such as wind or gravity, to a delegate or delegates. Any space warp that works with dynamics can be used with the Space Warp behavior.

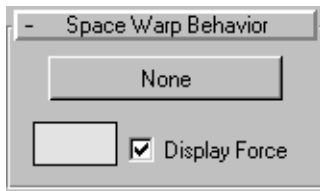
In particular, use the Space Warp behavior to tie delegates to a *Vector Field space warp* (see page 417), so that they avoid an object while following its contours.

## Procedure

### To use the Space Warp behavior

1. Add a space warp to the scene.
2. Add a Space Warp behavior to the Crowd object.
3. In the Space Warp Behavior rollout, use the None button to designate the space warp that will affect delegate motion.
4. Change the default settings as desired.
5. Use *behavior assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Space Warp Behavior rollout

**None (label).** Click this button, then select a space warp object.

**Color Swatch.** shows the color used to draw the Space Warp force vector during the solution. Click the box to choose a different color.

**Display Force.** When on, force exerted on the delegate(s) by the Space Warp behavior is drawn in the viewports as a vector during the simulation solution.

## Speed Vary Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Speed Vary Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Speed Vary Behavior

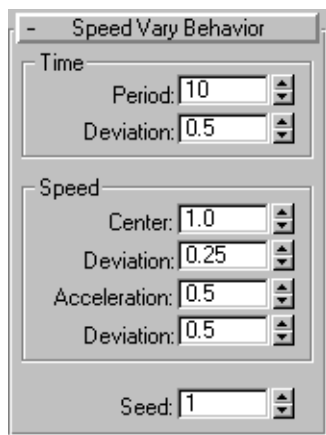
The Speed Vary behavior is useful for objects whose velocity changes at random as they move, such as sightseeing tourists. Its parameters let you specify how often a delegate should change speed, what speed range it should look at for a new speed, and how long it should accelerate to get to that new speed.

## Procedure

### To use the Speed Vary behavior

1. Add a Speed Vary behavior to the Crowd object.
2. Change the default settings as desired.
3. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Speed Vary Behavior rollout

#### Time group

**Period.** Specifies how many frames should elapse before a new speed is chosen.

**Deviation.** Specifies the maximum amount by which Period should vary. Each time a period ends, **character studio** takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Period setting, and adds the result to Period. Default=0.5. Range=0.0 to 1.0.

#### Speed group

The mathematical formula for the new speed is as follows: new speed = (delegate's Average Speed\*Center)\*(1 + RN\*Center Deviation), where RN is a random number between -1 and 1.

**Center.** Specifies the speed the delegate should change to. Center is a multiplier: A value of 0.0 means to stop, a value of 1.0 means to move at its *average speed* (see page 343), and a value greater than 1.0 means to move faster than its average speed. Default=1.0. Range=0.0 to 99,999.0.

**Deviation.** Specifies the maximum amount by which the delegate's calculated speed (Average Speed\*Center) should vary. Each time a period ends, **character studio** takes a random number between the negative and positive values of the Deviation setting, multiplies it by the calculated speed, and adds the result to the calculated speed. Default=0.25. Range=0.0 to 99,999.0.

**Accel Period.** Specifies the rate at which the delegate's speed should change in relation to the period length. A value of 0.0 means that the transition to the new speed will proceed as quickly as possible, and a value of 0.5 means the transition will take half the period. A value of 1.0 means the transition will take the entire period. Default=0.5. Range=0.0 to 1.0.

**Deviation.** Specifies the maximum amount by which Acceleration should vary. Each time a period ends, **character studio** takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Acceleration setting, and adds the result to Acceleration. Default=0.5. Range=0.0 to 1.0.

**Seed.** Specifies a seed value for randomizing the Speed Vary behavior.



## Surface Arrive Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Surface Arrive Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Surface Arrive Behavior

The Surface Arrive behavior is similar to the *Seek behavior* (see page 403) in that it lets you specify an object or objects as a stationary or moving target for delegates. The principal difference is

that you can use the Approach settings to specify an intermediate target. After reaching this location, the delegates will then make their final approach to the ultimate target surface. An example would be birds flying over a row of telephone poles, and then each one dropping to land on top of a different pole.

## Procedure

### To use the Surface Arrive behavior

1. Add a Surface Arrive behavior to the Crowd object.
2. Add an object or objects to serve as the target surface to the scene.

Note: If you use multiple objects, delegates will arrive at the surface of the closest object.

3. In the Surface Arrive Behavior rollout, use the None button or the Multiple Selection button to designate the one or more target objects.
4. Change the default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface

The screenshot shows the 'Surface Arrive Behavior' interface with the following settings:

- Target Selection:** A button labeled '-None-' with a list icon.
- Arrival:**
  - ☒ **Disable After Arriving**
  - Rate: 0.5
  - Deviation: 0.0
  - Speed: 0.0
  - Deviation: 0.0
  - Distance: 999999
  - Deviation: 0.0
- Location:**
  - Offset: 0.0
  - ☐ Facing
  - ☒ Random
  - ☐ Closest ☐ Every Frame
  - ☐ Display Offset
- Approach:**
  - Height: 0.0
  - Deviation: 0.0
  - Descent Start: 0.001
  - Deviation: 0.0
  - ☐ Off This Normal
    - X: 0.0
    - Y: 0.0
    - Z: 1.0
- Seed:** 1
- ☒ **Display Target**
- Target Scale:** 5.0

## Surface Arrive Behavior rollout

**None (label).** Specifies a single target. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you've selected multiple targets using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are designated as targets, click the Multiple Selection button.

**Multiple Selection.** Opens the Select dialog to let you designate multiple targets. When you have more than one target, you can set delegates to move toward the closest target in the group, or to a computed average of the target positions.

### Arrival group

Specifies three aspects of the Surface Arrive behavior: Rate, Speed, and Distance.

**Disable After Arriving.** When on, turns off the Surface Arrive behavior after the delegate arrives at the surface. Default=on.

**Rate.** A multiple of the delegate's *Max Accel* (see page 343) setting that specifies the acceleration with which it will try to arrive. A value of 1.0 means to use the full acceleration of the delegate. Default=0.5.

**Deviation.** Adds random variation to the to the Rate setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Rate setting. Default=0.0.

**Speed.** The speed at which to arrive, relative to the speed of the target. Default=0.0.

**Deviation.** Adds random variation to the Speed setting. The actual deviation is calculated by multiplying the Deviation setting by a random

number between -1 and 1, and then multiplying the result by the Speed setting. Default=0.0.

**Distance.** The maximum radial distance from the target within which the behavior will be active. Until the delegate is within this radius, the behavior has no influence. Default=9999999.0.

**Deviation.** Adds random variation to the to the Distance setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Distance setting. Default=0.0.

### Location group

**Offset.** Specifies a consistent distance from the calculated arrival point, based on the surface normal, for the delegate to use. Default=0.0.

**Facing.** When on, the delegate will try to arrive only at points on triangles on the surface that are facing it. Default=off.

**Random.** The software chooses a random point on the target surface as the arrival point. When using the Random option, the software chooses arrival points for delegates once, at the beginning of the simulation. This is the default choice

**Closest.** The software chooses the closest point on the target surface as the arrival point. If Closest is chosen, but Every Frame is off, the software chooses arrival points for delegates once, at the beginning of the simulation.

**Every Frame.** When on, the software chooses arrival points for delegates at every frame. Available only when Closest is chosen. Default=off.

Every Frame is useful when the target object is rotating during the animation, but requires more time for calculation.



**Display Offset.** When on, shows the Offset distance as lines emanating from each vertex in the surface object, perpendicular to the surface.

### Approach group

The Height and Descent settings together specify the path the delegate will take for its arrival. They allow for a wide range of behavior, from soft, gradual landings to direct helicopter-type descents.

In both cases, the actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and multiplying the result by the relevant value.

**Height.** Specifies a distance from the arrival point along its face normal. This is the point that the delegate will go to first before descending to the arrival point.

**Deviation.** Adds random variation to the to the Height setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Height setting.

**Descent Start.** Specifies the distance between the delegate and the arrival point at which the descent should start.

Note: Be careful that Descent Start is set high enough that the delegate won't overshoot when descending because its speed is too high and deceleration too low, compared to when it should start descending.

**Deviation.** Adds random variation to the to the Descent Start setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Descent Start setting.

**Off This Normal.** When on, lets you set an approach vector to specify the angle at which the final approach occurs. Default=off.

**X/Y/Z.** Use these settings to specify the final approach vector in world coordinates. For example, the vector specified by the default settings of X=0, Y=0, Z=1 means that the delegates will approach the target along the vertical world axis.

**Seed.** Affects the random numbers used to calculate the Deviation settings. For similar randomization among different Surface Arrive behaviors, use the same Seed value.

**Color Swatch.** Shows the color used to draw the target icon. Default=dark blue.

**Display Target.** Enables display of the target icon, which appears during the solution when a new interim goal is calculated for the delegate. Interim goals are created when using the Approach group settings. Default=on.

**Target Scale.** Specifies the overall size of the target icon. Default=5.0.

---

## Surface Follow Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Surface Follow Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Surface Follow Behavior

The Surface Follow behavior moves delegates with respect to object surfaces, which can be still or animated. For example, you can apply an animated Noise modifier to a patch grid to simulate a choppy water surface, and objects guided by Surface Follow will stay on top.

Note: By default, a delegate influenced by Surface Follow picks a direction to move in at any given frame based on its current facing and the plane of the face it's currently over. Thus

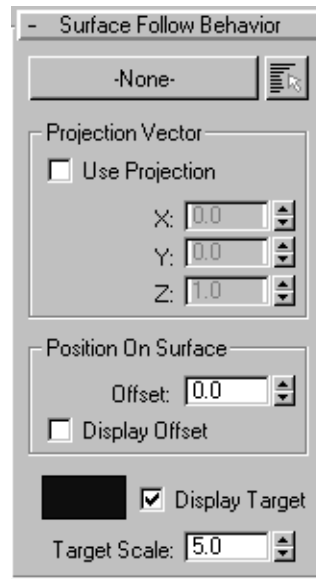
objects moving up a hill, while seeking a point at the bottom of the other side of the hill, tend to turn left or right to skirt the hill, rather than following the upward slope. You can override this with the Projection Vector option.

## Procedure

### To use the Surface Follow behavior

1. Add a Surface Follow behavior to the Crowd object.
2. Add an object or objects to serve as the follow surface to the scene.  
Note: If you use multiple objects, they must intersect to form a contiguous surface. Each delegate will move to the closest surface, follow it to the next closest that it encounters, and then start following that one, and so on.
3. In the Surface Follow Behavior rollout, use the None button or the Multiple Selection button to designate the object or objects whose surface(s) the assignees are to follow.
4. Change the default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Surface Follow Behavior rollout

**None (label).** Specifies a single “target” object to use as a surface. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you’ve selected multiple targets using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are designated as targets, click the Multiple Selection button. The Select dialog appears with designated targets highlighted.

**Multiple Selection.** Opens the Select dialog to let you designate multiple targets. When you have more than one target, delegates initially move toward the closest target in the group, and then move over its surface until they encounter another target, at which point they switch to its surface, and so on.

### Projection Vector group

These controls let you override the default direction calculated by the Surface Follow behavior by describing a virtual plane along which the delegate is to move. You do this by specifying a vector, in world coordinates, that's perpendicular to the desired virtual plane.

For example, if you want the delegate, when it encounters a hill, to keep moving forward, straight up and over the hill, instead of skirting it, you would use the default Projection Vector settings:  $X=Y=0$ ,  $Z=1$ . This vector is aligned with the world Z (vertical) axis, so it specifies a plane parallel to the world XY plane. Thus, the delegate always moves straight ahead while following the surface.

**Use Projection.** When on, Surface Follow calculates delegate direction from the specified vector, rather than using the default.

**X/Y/Z.** Specifies a vector using world coordinates. Default= $X=Y=0$ ,  $Z=1$ . Range=-1.0 to 1.0.

If only one of these settings is not 0, then the projection vector is aligned with the non-zero axis. Combine non-zero settings to create angled planes for Surface Follow. For example, to create a virtual plane that's rotated 45 degrees clockwise about world Y axis, set  $X=Z=1$  and  $Y=0$ . Also, while you can set all three axes to 0, that specifies no vector, and so effectively turns off Use Projection.

### Position on Surface group

**Offset.** Specifies the delegate's distance above the surface, using the surface normal. Recalculated at each frame.

**Display Offset.** When on, shows the Offset distance as lines emanating from each vertex in the surface object, perpendicular to the surface.

**Color Swatch.** Shows the color used to draw the Surface Follow target (see Display Target, next) during the solution. Click the box to choose a different color.

**Display Target.** When on, the interim goal for each delegate influenced by the Surface Follow behavior is drawn in the viewports as a wireframe sphere during the simulation solution.

If the delegate starts out away from the surface to be followed, the target is most visible before the delegate reaches the surface; the target is then positioned along the surface edge. While the delegate is actually following the surface, the target is usually coincident with the delegate, because Surface follow sets a new destination only a frame or two ahead.

---

## Wall Repel Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Wall Repel Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Wall Repel Behavior

The Wall Repel behavior uses one or more grid objects to repel delegates. When influenced by the Wall Repel force, delegates turn until they're heading away from the grid. It's useful for keeping objects inside an enclosed, straight-sided enclosure, such as a room in a building.

You can set the grids to repel from either side or both sides, and optionally specify a maximum distance for repelling. You can also set the behavior to act as though each grid extends infinitely along its plane.

Note: The Wall Repel behavior simply applies a force on the delegate in the direction opposite

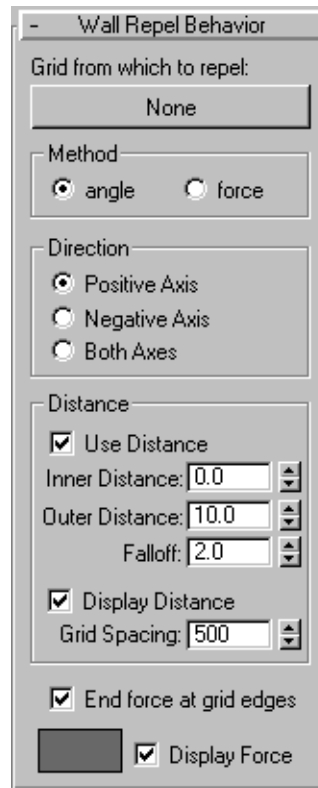
the wall; it does not guarantee that the delegate won't go through the wall. If the delegate does go through the wall, adjust settings such as Inner and Outer Distance, and, in particular, try reducing Falloff.

## Procedure

### To use the Wall Repel behavior

1. Add a Wall Repel behavior to the Crowd object.
2. Add one or more grid objects to the scene, and use the move, rotate, and/or scale functions to set them up.  
**Tip:** For best results, do not use Mirror to copy a grid to be used with Wall Repel. Use Shift-Clone instead.
3. In the Wall Repel Behavior rollout, use the None button to designate the grid to repel, or to specify multiple grids, use Multiple Selection.
4. Change the default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Wall Repel Behavior rollout

**Grid from which to repel.** Set a repelling grid ("source") by clicking the None button and then selecting the grid. Thereafter, the grid's name appears on the button.

To set multiple source grids, click Multiple Selection and use the Select dialog. With multiple source grids, the word "Multiple" appears on the large button.

### Method group

Determines whether delegate direction as influenced by the behavior is calculated by an

angular method or a force method.  
Default=Force.

**Angle.** Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly away from the source.

The magnitude of the force is greatest when the delegate is moving directly towards the source, and needs to turn around. It can be as little as 0 when the delegate is moving directly away from the source.

**Force.** Always applies a force directly away from the source. The magnitude of the force is constant.

### Direction group

Determines whether the grid repels from its positive-axis side, its negative-axis side, or both.

To determine which is the positive-axis side, select the grid, and then set the reference coordinate system to Local (the default system is View). Look at the grid in a viewport in which it appears edge-on. The side with the axis sticking out is the positive-axis side, and the opposite side is the negative-axis side.

**Tip:** For ease of setup, when building a "room" from grids, make sure they all point inward (or outward).

**Positive Axis.** The grid repels only from the positive-axis side.

**Positive Axis.** The grid repels only from the negative-axis side.

**Both Axes.** The grid repels from both sides.

### Distance group

Use the distance settings to activate the Wall Repel behavior only when the delegates are within a specific distance from the target. The relative strength of the behavior increases from

0 percent at the outer radius to 100 percent at the inner radius.

**Use Distance.** When on, the behavior applies only to delegates closer to the target than the Outer Distance value. Default=on.

**Inner Distance.** The distance from the target at which the force is applied at full strength. Default=0.0.

**Outer Distance.** The distance from the target at which the force begins to be applied. Default=10.0.

**Falloff.** The rate at which the repelling force diminishes between the Inner Distance and the Outer Distance. A value of 1.0 indicates a linear falloff. Higher values cause the strength to fall off to zero more rapidly with distance, thus focusing its effect closer to the Inner Distance. Lower values reduce the rate of diminishment, with a Falloff setting of 0.0 indicating that the strength is the same at the Outer Distance as it is at the Inner Distance. Default=2.0.

**Display Distance.** Shows the inner and outer distance settings as grids offset from the target grid in the viewports. The Inner Distance grid is light blue, while the Outer Distance grid is blue-white. Default=on.

**Grid Spacing.** Alters the spacing of grid lines used to draw the Inner/Outer Distance grids. The lower the value, the closer the spacing. Default=500.

**End force at grid edges.** When on, the force emanates only from the grid object. When off, the force emanates from an imaginary infinite grid created by extending the grid plane in all directions.

**Color Swatch.** Shows the color used to draw the Wall Repel force during the solution. Click the box to choose a different color. Default=violet.

**Display Force.** The force, when activated, is drawn in the viewports as a wireframe rectangle during the simulation solution. Default=on.

## Wall Seek Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Wall Seek Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Wall Seek Behavior

The Wall Seek behavior uses a grid object to attract delegates. When influenced by the Wall Seek force, delegates turn until they're heading toward the grid. It's useful for moving objects toward a rectangular area, such as a doorway.

You can set the grid to attract from either side or both sides, and optionally specify a maximum distance for attraction. You can also set the behavior to act as though the grid extends infinitely along its plane.

## Procedure

### To use the Wall Seek behavior

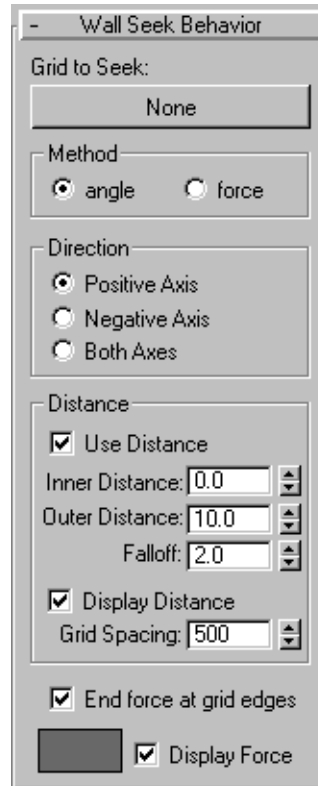
1. Add a Wall Seek behavior to the Crowd object.
2. Add a grid object to the scene, and use the move, rotate, and/or scale functions to set it up.

**Tip:** For best results, do not use Mirror to copy a grid to be used with Wall Seek. Use Shift-Clone instead.

3. In the Wall Seek Behavior rollout, use the None button to designate the grid to be sought.
4. Change the default settings as desired.

5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



## Wall Seek Behavior rollout

**Grid to seek.** Set the target grid by clicking the None button and then selecting the grid. Thereafter, the grid's name appears on the button.

### Method group

Determines whether delegate direction as influenced by the behavior is calculated by an angular method or a force method. Default=Angle.

**Angle.** Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly toward the target.

The magnitude of the force is greatest when the delegate is moving away from the target, and needs to turn around. It can be as little as 0 when the delegate is directly approaching the target.

**Force.** Always applies a force directly towards the target. The magnitude of the force is constant.

### Direction group

Determines whether the grid attracts from its positive-axis side, its negative-axis side, or both.

To determine which is the positive-axis side, select the grid, and then set the reference coordinate system to Local (the default system is View). Look at the grid in a viewport in which it appears edge-on. The side with the axis sticking out is the positive-axis side, and the opposite side is the negative-axis side.

**Positive Axis.** The grid attracts only from the positive-axis side.

**Positive Axis.** The grid attracts only from the negative-axis side.

**Both Axes.** The grid attracts from both sides.

### Distance group

Use the distance settings to activate the WallSeek behavior only when the delegates are within a specific distance from the target. The relative strength of the behavior increases from 0 percent at the outer distance to 100 percent at the inner distance.

**Use Distance.** When on, the behavior applies only to delegates closer to the target than the Outer Distance value. Default=on.

**Inner Distance.** The distance from the target at which the force is applied at full strength. Default=0.0.

**Outer Distance.** The distance from the target at which the force begins to be applied.

Default=10.0.

**Falloff.** The rate at which the attracting force diminishes between the Inner Distance and the Outer Distance. A value of 1.0 indicates a linear falloff. Higher values cause the strength to fall off to zero more rapidly with distance, thus focusing its effect closer to the Inner Distance. Lower values reduce the rate of diminishment, with a Falloff setting of 0.0 indicating that the strength is the same at the Outer Distance as it is at the Inner Distance. Default=2.0.

**Display Distance.** Shows the inner and outer distance settings as grids offset from the target grid in the viewports. The Inner Distance grid is light blue, while the Outer Distance grid is blue-white. Default=on.

**Grid Spacing.** Alters the spacing of grid lines used to draw the Inner/Outer Distance grids. The lower the value, the closer the spacing. Default=500.

**End force at grid edges.** When on, the force emanates only from the grid object. When off, the force emanates from an imaginary infinite grid created by extending the grid plane in all directions.

**Color Swatch.** Shows the color used to draw the Seek force vector (and, if used, the radii) during the solution. Click the box to choose a different color. Default=violet.

**Display Force.** When on, force exerted on the delegate(s) by the Seek behavior is drawn in the viewports as a vector during the simulation solution. If Use Radii is turned on, the radii are also displayed when the force is active.

## Wander Behavior

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Wander Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Wander Behavior

The Wander behavior imparts a random motion to delegates, letting you simulate meandering activity in which delegates move and turn in a haphazard manner. It works by randomly picking a new direction, and then turning and moving in that direction. You can specify how often to pick a new direction, how far to turn, and how fast or slow to turn while moving.

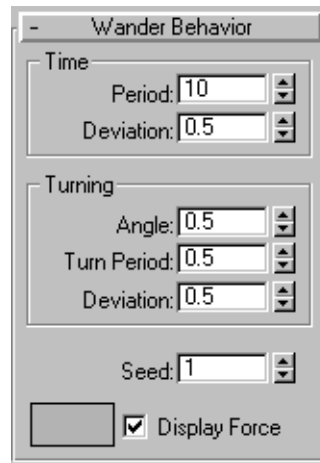
**Tip:** Trajectories calculated by the Wander behavior vary significantly for each object to which it's applied. To create a group of wandering delegates all using the same trajectory, apply the Wander behavior to a delegate, solve the simulation, and then replicate the delegate using the *Scatter Objects* (see page 362) facility with Clone Controllers turned on.

## Procedure

### To use the Wander behavior

1. Add a Wander behavior to the Crowd object.
2. Change the default settings as desired. Probably the most important is Period, which determines the number of frames between changes of direction.
3. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



## Wander Behavior rollout

### Time group

**Period.** Specifies how many frames should elapse before a new direction is chosen. Default=10.

**Deviation.** Specifies the maximum amount by which Period should vary. Each time a period ends, **character studio** takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Period setting, and adds the result to Period. Default=0.5. Range=0.0 to 1.0.

### Turning group

**Angle.** Specifies how far to turn when changing direction. A small value means to change direction only by a small amount, while as the value approaches 1.0 it will randomly turn in any direction. Default=0.5. Range=0.5 to 1.0.

**Turn Period.** Specifies how long over the current period it takes to turn. A value of 0.0 means that it will rotate as quickly as possible to face a direction and then travel in that a direction,



while a value of 1.0 means it will take the entire period to rotate in that direction. Default=0.5. Range=0.5 to 1.0.

**Deviation.** Specifies the maximum amount by which Angle should vary. Each time a period ends, **character studio** takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Angle setting, and adds the result to Angle. Default=0.5. Range=0.0 to 1.0.

**Seed.** Specifies a seed value for randomizing the Wander behavior.

**Color Swatch.** Shows the color used to draw the Wander force vector during the solution. Click the box to choose a different color.

**Display Force.** When on, force exerted on the delegate(s) by the Wander behavior is drawn in the viewports as a vector during the simulation solution.



## Vector Field Space Warp

Create panel > Space Warps > Particles & Dynamics > Object Type rollout > Vector Field

A vector field is a special type of space warp that crowd members use to move around irregular objects such as a curved, concave surface. The Vector Field object, a box-shaped lattice, is placed and sized so that it surrounds the object to be avoided. The vectors are generated from the lattice intersections. These vectors are, by default, perpendicular to the surface of the object to which the field is applied; if necessary, you can smooth them out with a blending function. The crowd members move around the object by traveling perpendicular to the vectors.

You can use the vector field either as a *Space Warp behavior* (see page 404) or as the source object for an *Avoid behavior* (see page 392), or both. When used together, delegates slow down when they approach a complex object, and then go around it. This guarantees that the delegate will not pass through the obstacle's surface.

The vector field includes settings for strength, falloff, and a push/pull effect, as well as options to display the lattice, the effective range, and the vectors.

The Vector Field space warp provides these rollouts:

*Create Method Rollout* (see page 419)

*Lattice Parameters Rollout* (see page 419)

*Obstacle Parameters Rollout* (see page 420)

## Procedures

### To add a Vector Field space warp

Adding a Vector Field space warp object works the same as adding a Box geometry primitive.

1. On the Object Type rollout, click Vector Field.
2. Drag in a viewport to set the first initial dimensions.
  - If using the Cube creation method, this sets all three dimensions simultaneously.
  - If using the Box creation method, release the mouse button, and then move the mouse vertically to set the height.
3. Click to create the space warp.

#### To use a Vector Field space warp with delegates

This procedure presents general guidelines for using the Vector Field space warp with delegates in a crowd simulation.

1. Create an object to serve as an obstacle. This object must be an editable mesh or geometric primitive; it can have modifiers.
2. Add a Vector Field space warp.
3. Position and scale the space warp lattice so that it encloses the obstacle object.  
The lattice should be significantly larger than the object. The object should be located roughly at the lattice center.
4. In the Lattice Parameters rollout, increase the Length Segs/Width Segs/Height Segs settings so that the lattice segments intersect the object at reasonable intervals.  
To determine appropriate Segs settings for your obstacle objects, first examine the object complexity. If the obstacle has a lot of detail, and you want that detail reflected in the vector field, then you need a relatively high lattice resolution.
5. Click the Obstacle Parameters rollout > Compute Vectors group > None button, and then select the object that will act as an obstacle in the crowd simulation.

This specifies the obstacle object. The *range volume grid* appears on the object's surface as an olive-colored wireframe.

6. Increase the Obstacle Parameters rollout > Compute Vectors group > Range setting.  
As you increase this setting, you'll see the range volume grid expand. The range volume should enclose the space in which crowd members need to start turning in order to avoid the object.
7. Turn off Display group > Show Lattice and Show Range so that the vector field will be more easily visible when generated.
8. Turn on Display group > Show Vector Field.
9. In the Compute Vectors group, click the Compute button. This generates the vector field.  
**Tip:** To make the vector lines more evident, increase the Display group > VectorScale setting.  
The vectors appear as blue lines surrounding the obstacle object. One vector is computed for each lattice intersection within the range volume grid. Each vector matches the normal of the object at the point on the object's surface closest to the lattice point.  
The vector force falls off with distance from the object, as shown by the progressively shorter vector lines toward the grid perimeter.
10. Add *Crowd* (see page 346) and *Delegate* (see page 342) helper objects.
11. Select the Crowd object and open the Modify panel.
12. In the Setup rollout > Behaviors group, click New.
13. In the Select Behavior Type dialog that appears, choose Space Warp Behavior, and then click OK.

14. In the Space Warp Behavior rollout that appears, click the None button, and then select the Vector Field space warp.  
You might find it easiest to use Select By Name to select the space warp.
15. In the Setup rollout, click the Behavior Assignments button, and use the *Behavior Assignments and Teams dialog* (see page 375) to assign your delegate or delegates to the space warp behavior.
16. Add any other objects and/or behaviors appropriate to the simulation.
17. Select the Crowd object, and then solve the simulation by clicking the Solve rollout > Solve button.
18. Fine tune the behavior associated with the Vector Field space warp by adjusting the *Lattice parameters* (see page 419) and *Obstacle parameters* (see page 420).
19. Continue computing the vector field and then solving the simulation after each adjustment. In certain cases you might need to animate the vector field parameters to keep objects within the field.

#### To use a Vector Field space warp with a particle system

1. Begin by following steps 1 to 9 in the previous procedure, *To use a Vector Field space warp* (see page 418).
2. Add a particle system in the vicinity of the vector field.
3. Use the Bind to Space Warp tool to bind the particle system to the Vector Field space warp.

The particles should use the vectors to move around the obstacle object. If they don't, you may have to slow them down so that the vector field has a chance to calculate their positions.



## Create Method Rollout

Create panel > Space Warps > Particles & Dynamics > Object Type rollout > Vector Field > Create Method rollout

The Create Method rollout for the Vector Field space warp lets you specify whether to create the vector field using the cube or box method.

### Interface

**Cube.** Forces length, width, and height to be equal. Creating a cube-shaped space warp is a one-step operation. Starting at the center of the cube, drag in a viewport to set all three dimensions simultaneously. You can change the individual dimensions in the Lattice Parameters rollout.

**Box.** Creates a standard box-shaped space warp from one corner to the diagonally opposite corner, with different settings for length, width, and height.



## Lattice Parameters Rollout

Create panel > Space Warps > Particles & Dynamics > Object Type rollout > Vector Field > Lattice Parameters rollout

Select a Vector Field space warp. > Modify panel > Lattice Parameters rollout

Use these parameters to specify the vector field lattice size and number of segments.

### Interface

**Length/Width/Height.** Specify the dimensions of the vector field lattice. The lattice should be larger than the vector field object.

**Length Segs/Width Segs/Height Segs.** Specify the resolution of the vector field lattice. The greater the resolution, the higher the accuracy of the simulation.

## Obstacle Parameters Rollout

Create panel > Space Warps > Particles & Dynamics > Object Type rollout > Vector Field > Obstacle Parameters rollout

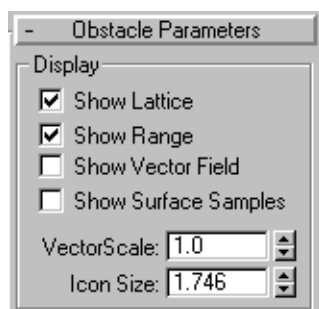
Select a Vector Field space warp. > Modify panel > Obstacle Parameters rollout

A *vector field* (see page 417) serves to let *crowd* (see page 346) members avoid other objects in the scene. It consists of a three-dimensional array of vectors generated around an *obstacle object*, which serve to guide objects around the object. The settings in this rollout help determine how the vectors are generated and displayed, and how they affect other objects.

Note: Objects are subject to a vector field's forces only if they are bound to the field with a Crowd object. For general usage guidelines, see *To use a Vector Field space warp* (see page 418).

## Interface

Display group



The first four controls in this group let you enable and disable display of four different elements of the Vector Field space warp.

Note: You can select a Vector Field space warp in the viewport by clicking on any of its visible elements *except* the range representation. If you've turned off display of all four elements, you can still select the space warp by clicking its gizmo, which, when viewed from the "top" (the orthogonal viewport in which the warp was created), resembles a pair of crossed double-headed arrows in the shape of an X.

**Show Lattice.** Displays the vector field lattice, a yellow wireframe box. The vectors are generated at lattice intersections inside the vector field range. Default=on.

**Show Range.** Displays the volume about the obstacle object within which vectors are generated, as an olive-colored wireframe. The range starts out the same shape and size as the obstacle object, and is typically enlarged with the Compute Vectors group > Range setting. Default=on.

**Show Vector Field.** Displays vectors, which appear as blue lines emanating outward from lattice intersections within the range volume. Default=off.

**Show Surface Samples.** Displays short green lines emanating from sample points on the surface of the obstacle object. These appear only after you've *computed* (see page 422) the vector field. See *Sample Resolution* (see page 422) for more information. Default=off.

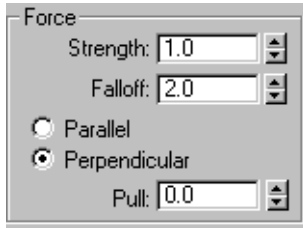
**VectorScale.** Scales the vectors so they're more visible or less obtrusive. Default=1.0.

Note: This setting does not affect the strength of the vectors; only their visibility.

**Icon Size.** Adjusts the size of the Vector Field space warp icon, a pair of crossed double-headed

arrows. Increase the size for easier viewport selection. Default=size originally drawn in viewport.

### Force group



These parameters determine how the vector field affects objects within its volume.

Changing any of the Force group settings does not require that you recalculate the vector field.

Note: The vector field does not assure that delegates or particles remain at a particular distance from the obstacle. In some cases you might have to animate the Strength, Falloff, and/or Pull settings to keep delegates within the vector field.

**Strength.** Sets the degree of effect the vectors have on the movement of an object entering the vector field. If *Show Vector Field* (see page 420) is on as you adjust Strength, you can see the vector lines change size in the viewports in real time. Default=1.0.

Note: Sometimes, after changing strength, vectors will be too large or too small. In such cases, adjust the *VectorScale* (see page 420) parameter so that they display properly.

**Falloff.** Determines the rate at which the strength of the vectors falls off with distance from the surface of the object. A value of 0 will make all the vectors the same size. A value greater than 0 will make them get smaller as they get further away. A value less than 0 will make

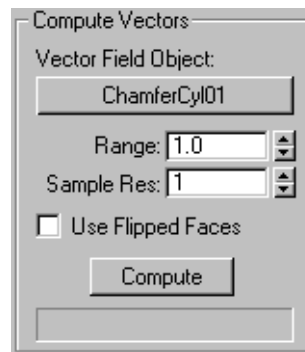
them get larger as they get further away. Default=2.0.

**Parallel/Perpendicular.** Sets whether the force generated by the vectors works parallel or perpendicular to the vector field. Because the vectors are perpendicular to the object surface, and you typically would want delegates to travel parallel to the surface, you would normally use a perpendicular force. Default=Perpendicular.

**Pull.** Adjusts objects' position relative to the field. Available only when Perpendicular is chosen. Default=0.0. Range=-1.0 to 1.0.

Objects moving perpendicular to a vector field sometimes tend to drift away from it, due to lack of subsampling. The Pull parameter helps to pull objects back. Pull values greater than 0 create a pulling force towards the source of the vector field vector. Values less than 0 pull the force towards the direction in which the vector field's vector is pointing. A value of 0.0 produces a force perfectly perpendicular to the vector field's vector.

### Compute Vectors group



**Vector Field Object.** Lets you designate the obstacle object. Click this button, and then select the object around which the vector field is to be generated. Thereafter the object's name appears on the button.

Note: You can use only primitives and unmodified editable mesh objects as obstacles. Also, the object should be fully enclosed in the Vector Field lattice.

**Range.** Determines the volume within which vectors are generated. The Range is represented in viewports as an olive-colored wireframe that starts out the same size and shape as the obstacle object. Increasing the Range setting moves the wireframe away from the obstacle object in the direction of its surface normals. Default=1.0.

In crowd simulations, the Range outline is where the delegates start to “see” the obstacle object, and begin to turn to avoid it. If your crowd members are penetrating the obstacle, or even just coming too close to it before turning, increase the Range setting. Also try increasing the Vector Field lattice resolution and/or the Sample Res setting.

**Sample Res(olution).** Acts as a multiplier of the effective sampling rate used on the obstacle object’s surface to calculate vector directions in the field. Default=1.

The basic sampling rate is determined by the program from the size of the lattice and the size of each polygon.

**Use Flipped Faces.** Causes flipped normals to be used during the computation of the vector field. Default=off.

By default, vectors are generated in the same direction as the obstacle object’s face normals, so that assuming its face normals point outward, objects move around its exterior in a crowd simulation. However, if you want objects to remain within an object’s interior, turn on Use Flipped Faces.

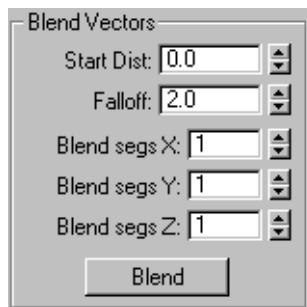
**Tip:** If you run a crowd simulation inside an object that is also being viewed from the inside, such as a room, you’ll probably want the object’s faces to point inward. In that case, use Editable

Mesh/Edit Mesh > Surface Properties to flip the normals, and don’t turn on Use Flipped Faces.

**Compute.** Calculates the vector field.

Always recalculate the vector field after changing any parameters except those in the Force group.

### Blend Vectors group



The Blend Vectors parameters are for reducing abrupt changes in angles of neighboring vectors. For example, if you have a wavy surface, you may get wavy vectors very far out from the surface, which might adversely affect the simulation. Use Blend Vectors to correct this condition.

**Start Dist.** The distance from the object at which you wish to start blending the vectors. Default=0.0.

**Falloff.** The falloff of the blend of the surrounding vectors. Default=2.0.

**Blend segs X.** The number of adjacent lattice points to blend on the X axis. Default=1.

**Blend segs Y.** The number of adjacent lattice points to blend on the Y axis. Default=1.

**Blend segs Z.** The number of adjacent lattice points to blend on the Z axis. Default=1.

**Blend.** Click this button to implement the blending.

---

# Tutorial Introduction

---

## Welcome

Welcome to the **character studio 3** tutorials.

These lessons will acquaint you with the tools and techniques of **character studio**. In these exercises you'll get hands-on experience creating and animating characters. You'll be introduced to the concepts of Biped animation and gain practical knowledge in the tools, thereby developong a knowledge base for doing your own projects.

---

## How to Use These Tutorials

To start learning about the program, read *Using character studio* (see page 1). This gives you an overview of *Biped* (see page 2), *Physique* (see page 9) and *Crowd* (see page 13) features. This will also provide in-depth topics for *understanding Biped* (see page 5) and *Physique* (see page 11), and *workflow* (see page 15).

## Online and Printed Tutorials

You have the choice of using the printed tutorials or following the same lessons online. The content is fundamentally the same, though the online tutorials have the advantage of showing illustrations in color. Occasionally there are last minute corrections that occur to the online documentation, too late for the printed volume. These corrections are noted in the *Readme* file on the CD.

If your system has only a single monitor, you can resize the help window to cover the two upper viewports horizontally. On a dual-monitor system, move the help window to the second monitor, so you can see all four

viewports. Following the printed tutorials also solves this problem.

## Opening the Online Tutorials

To open the help system from within 3D Studio MAX, choose Help > Additional Help > character studio 3 Reference, and click Display Help. As a plug-in to 3DS MAX, the **character studio 3** documentation is not part of the standard 3DS MAX reference.

You can also run the compiled help file, *cstudio.chm*, by clicking the icon in the Windows Explorer. If you have trouble running it, you may need install Internet Explorer 4 or higher. You'll find the install for IE4 on your 3DS MAX R3 CD.

## Finding the Tutorial Files

Each tutorial begins with instructions for loading of necessary files. Files for the tutorials are automatically installed as part of the standard installation. However, if you do a minimal installation, the tutorial support files will not be installed. In that case, you can load the tutorial files directly from the *\tutorials* directory on the CD.

Each tutorial has its own subdirectory located under *\tutorials*. All the support files are in these directories, including textures, motion files, 3DS MAX scenes, and sample AVI files.

## Tutorials and Procedures

Tutorials teach you to use **character studio 3** in the context of creating animation. They focus on learning how to accomplish a task, such as animating a character swimming.

If you need to learn how to use a particular tool or feature, you will also find the steps outlined in the procedures that are found in the User Interface reference descriptions.

When you are in a hurry to learn how to use a tool or feature, go to the Contents panel of the help system and open Procedures > *character studio Procedures* (see page 123).

## MAXScript Tutorial

A tutorial to help you use MAXScript with **character studio** can be found in the MAXScript Reference, updated with this release. Look for the CS3Tools.cui Tutorial topic.

---

## What You Will Learn in These Tutorials

Following is a brief description of the tutorials that ship with **character studio**. They won't teach you everything there is to know, but they will get you started and provide a strong foundation for your knowledge of the software.

- *Tutorial 1: Biped and Physique* (see page 429)

In *Lesson 1* (see page 429), you'll learn how to create a biped and apply different motions to it.

In *Lesson 2* (see page 433), you'll learn how to use Figure mode to change the biped skeleton. You'll see how biped animation adapts to changes in the biped skeleton, and you'll use the biped ponytails to create jaws for an alligator character.

In *Lesson 3* (see page 438), you'll see how Biped with Physique work together to animate the character of Dr. X, the mad scientist.

In *Lesson 4* (see page 442), you'll merge the character of Tubby McChubbs into the scene with Dr. X. You'll make a clone of Dr. X, to create his assistant.



- *Tutorial 2: Freeform Biped Animation (see page 447)*

In *Lesson 1 (see page 447)*, you'll create a simple freeform animation of a biped swimming.

In *Lesson 2 (see page 457)*, you'll use learn to use animated pivot points in Freeform mode to create a walking animation.

- *Tutorial 3: Animating a Biped with Footsteps (see page 469)*

In *Lesson 1 (see page 470)*, you'll modify the default walk cycle to add personality to the movement.

In *Lesson 2 (see page 477)*, you'll learn to use Cut and Paste Footsteps to lengthen a walking animation. You'll also create a walk on uneven terrain and add a jump.

In *Lesson 3 (see page 481)*, you'll animate gymnastic flips.

In *Lesson 4 (see page 494)*, you'll use sliding keys to animate a character taking a pratfall. You'll add a freeform period to a footstep animation, and convert between animation modes.

In *Lesson 5 (see page 500)*, you'll learn to modify footsteps using IK Blend keys.

- *Tutorial 4: Making a Biped Interact with Objects (see page 503)*

In *Lesson 1 (see page 504)*, you'll control a biped's hand with a bouncing basketball using IK Blend.

In *Lesson 2 (see page 507)*, you'll animate a character climbing and slipping off a ladder.

In *Lesson 3 (see page 516)*, you'll use the Link controller to animate a biped picking up a briefcase.

In *Lesson 4 (see page 519)*, you'll learn to create the illusion of weight.

In *Lesson 5 (see page 522)*, you'll learn to use In Place mode follow a biped's animation.

- *Tutorial 5: Motion Flow Mode (see page 525)*

In *Lesson 1 (see page 526)*, you'll learn how to add clips in Motion Flow mode.

In *Lesson 2 (see page 526)*, you'll create motion flow scripts, edit transitions between clips, and generate a unified motion from the script.

In *Lesson 3 (see page 529)*, you'll loop an animation in motion flow and use layers to make changes to a walking animation.

In *Lesson 4 (see page 530)*, you'll create a shared motion flow for multiple bipeds.

In *Lesson 5 (see page 532)*, you'll create random motion using a motion flow network.

- *Tutorial 6: Working with Motion Capture Data (see page 535)*

In *Lesson 1 (see page 536)*, you'll import, filter, and edit a motion capture file.

In *Lesson 2 (see page 538)*, you'll use trajectories with motion capture files.

In *Lesson 3 (see page 539)*, you'll work with high frequency motion capture data. You'll also loop the motion capture file using motion capture import.

In *Lesson 4 (see page 540)*, you'll use layers to change a motion capture file. You'll animate a biped running sideways on a wall.

- *Tutorial 7: Working with Physique (see page 543)*

In *Lesson 1 (see page 544)*, you'll fit a biped to a patch character, Dr. X.

In *Lesson 2 (see page 548)*, you'll apply and adjust a Physique modifier to the character.

In *Lesson 3 (see page 549)*, you'll fine tune the physique envelopes and weighted vertices.

In *Lesson 4* (see page 551), you'll fix problems in the shoulders and pelvis.

In *Lesson 5* (see page 552), you'll animate muscles with the Bulge Editor.

In *Lesson 6* (see page 553), you'll scale a character with a Physique modifier applied.

In *Lesson 7* (see page 554), you'll use the 3DS MAX Link tool to connect a jointed character and a biped.

- *Tutorial 8: Working with Crowds* (see page 555)

In *Lesson 1* (see page 556), you'll create a delegate and assign a behavior to it.

In *Lesson 2* (see page 559), you'll use Scatter Objects tools to create multiple delegates, and then assign behaviors.

In *Lesson 3* (see page 562), you'll prevent collisions using the Avoid behavior, and learn to animate the weighting of behaviors.

In *Lesson 4* (see page 564), you'll use cognitive controllers to add logic to crowd behavior.

In *Lesson 5* (see page 567), you'll use the Clip controller to animate crowds of non-biped objects, a flock of birds.

In *Lesson 6* (see page 573), you'll create a crowd of swimming bipeds that follow an animated surface.

In *Lesson 7* (see page 577), you'll use motion synthesis and backtracking to animate two gangs of bipeds interacting.

- *Tutorial 9: Animating Multilegged Creatures* (see page 585)

In *Lesson 1* (see page 586), you'll animate a quadruped using animated pivot points in freeform mode.

In *Lesson 2* (see page 598), you'll use Snapshot to create extra limbs for a biped with four arms.

## The Cast of Characters

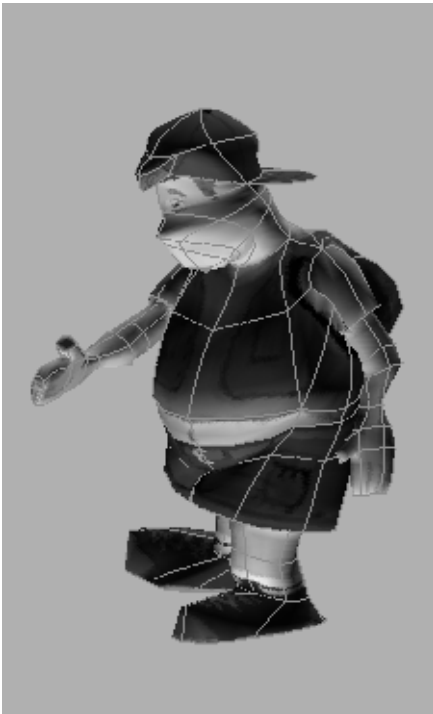
In some tutorials, you will work with the default biped armature that you create quickly in **character studio**. For other tutorials, you will use some actual characters from productions created by animators at discreet.

You'll animate the character of Dr. X., the evil scientist from the game prototype "The Adventures of Tubby McChubb." Dr. X will walk, jump, flip, and slip, all at your command as you proceed through these lessons.



The Mad Scientist

You'll also use Tubby McChubbs to learn how to merge characters into your scene.



**Tubby McChubbs**

There's a Gumby-style construction worker who climbs a ladder. You'll use IK Blending and link controllers to animate the pool guy.



**The pool guy climbing a ladder**

And there is a beetle from the animation *bugsnfrij.avi*, that is used to teach you quadruped animation.



**The beetle in the refrigerator**



---

## Tutorial 1

# Biped and Physique

In this tutorial, you learn about Biped and Physique. You'll do the following:

- Create a default biped character and apply motion to it
- Change the biped's structure to see how Biped animations intelligently adapt to characters of differing proportions to maintain the original, natural-looking motion
- Use biped motions to animate a character (Dr. X) and view the animations

## Lessons

*Lesson 1: Creating a Biped (see page 429)*

*Lesson 2: Modifying the Biped Structure in Figure Mode (see page 433)*

*Lesson 3: Using Biped with Physique (see page 438)*

*Lesson 4: Merging and Cloning Characters (see page 442)*

---

## Lesson 1: Creating a Biped

In this lesson, you will create a default biped: a simple skeleton consisting of bones that are connected in a hierarchy. A default biped is different from a skeleton of 3D Studio MAX bone system objects, because the biped structure automatically has built-in joints like a human being. You can bend your knee so your foot touches the back of your thigh, but you can't bend it forwards so your toe touches the front of your thigh. Biped creates skeletons in the same fashion. They are ready to animate, and work correctly without further setup.



### Setup

- Reset 3DS MAX.

You'll find the files for all tutorials are in the *cstudio\tutorial* directories, in your 3DS MAX path. Each tutorial has its own subdirectory with all the needed files. To load the files for tutorial 1, for example, look in *cstudio\tutorials\tutorial\_1*.

If the tutorial files have not been installed on your system, you can open the files from the Tutorials directory on the **character studio 3** CD.

### Creating a Biped

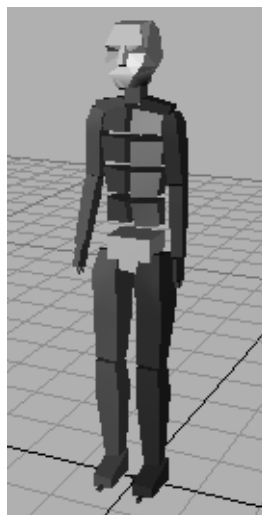
-  On the Create panel, click Systems.
-  On the Object Type rollout, click Biped.  
The Biped button turns green.  
The Create Biped Rollout is displayed on the command panel.
- If you can't see the Height spinner in the Create Biped rollout, scroll to the bottom of the command panel.
- In the Perspective viewport, place your cursor over the center of the grid. Press and drag upwards.  
A blue box appears and grows with your cursor movement.
- Drag upwards until the Height spinner on the Create Biped rollout reads approximately 70 units, then release.  
A biped is created in the viewport.  
The biped is a hierarchy of special objects. Its parent object (*Bip01*) is its center of mass (COM). The COM is displayed in the viewports as a small tetrahedron, initially centered in the biped's pelvis. After you

create a biped, only the center of mass object is selected (not the entire biped).

### Modifying the Biped

- Zoom into the Perspective viewport to get a closer look at the biped.

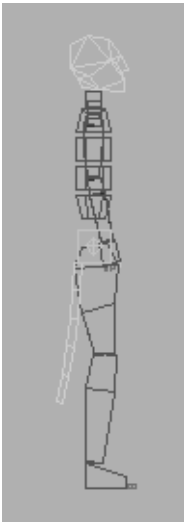
In zoom mode you can switch to pan with the middle mouse button, and rotate by pressing Alt and the middle mouse button.



### Default Biped

- Zoom into each viewport to view the biped.
- Change the Tail Links spinner from 0 to 5, and back to 0.

In the Left view you can see the biped growing with a tail.




#### Biped With tail

As long as this biped remains selected you can make changes to the Create Biped parameters and see the effects.

4. Change the Height of the biped to 50.  
The biped shrinks and its feet stay on the ground.
5. Change the Height back to 70.  
If you deselect and reselect the biped, changes you make in this rollout no longer affect the selected biped. You can change the selected biped's structure parameters by going to the Motion panel (rather than the Modify panel), choosing Figure mode and making changes to the parameters found in the Structure rollout.


Next you'll add a motion file to the biped you just created.

#### Loading and viewing a footstep animation

1.  Click the Motion panel.

The biped rollouts are displayed in the panel.

If the rollout appears empty you have inadvertently deselected the biped. If this is the case, press H and highlight *Bip01*, then click Select to close the dialog.

2.  In the General Rollout click Load File and select *cstudio\tutorial\tutorial 1\baller.bip*. Then click Open.

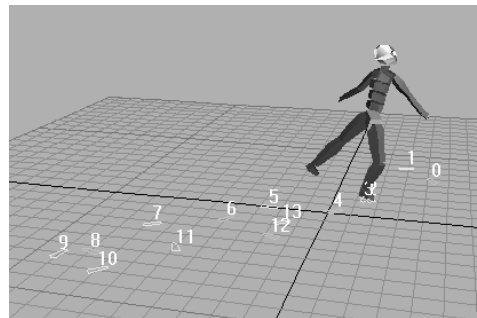
The motion file is loaded. It is a **character studio** animation that uses the footstep method.

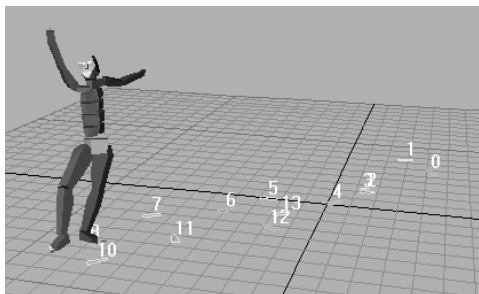
3.  Click Zoom Extents All.

The biped and its footsteps are centered in all viewports. Adjust all your viewports so you can see the biped and the footsteps.

**Tip:** You can switch to User view with the keyboard shortcut U. Use region zoom to frame your shot, then change back to Perspective by pressing P.

4. In the Perspective viewport, use the middle mouse button to pan down.  
Make sure Perspective is the active viewport before you go to the next step.
5. Play the animation in the viewport by clicking Play in the VCR controls. Click Play again to stop the playback.






The animation plays shaded in the viewport

### Biped Playback

Biped has a special playback mode that displays the motion on a stick figure. This helps you focus on the motion only, because the biped geometry and everything else in the scene is hidden in this playback mode.

1.  In the General rollout, click Biped Playback.

The biped becomes a stick figure and the motion preview plays repeatedly.

2. Click Biped Playback again to stop the playback.

The animation stops at the current frame.

**Warning:** Don't use both kinds of playback simultaneously.

**Tip:** By default, 3DS MAX drops frames to maintain a real-time frame rate during playback. Turn off Real Time in the Time Configuration dialog to display every frame during playback.

You just loaded a footstep animation. Now you'll load a freeform animation.

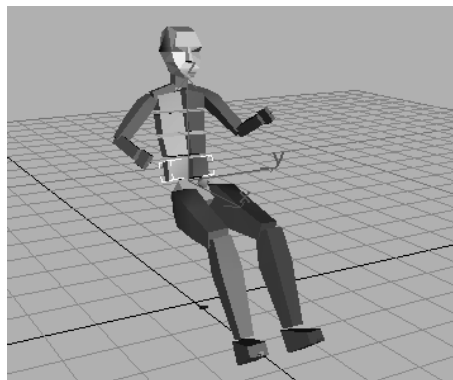
### Loading and Viewing a freeform animation

1. On the Motion panel, click Load file and select *drive.bip*.

This is a freeform animation. There are no footsteps visible, because freeform

animation does not use Biped's footstep method. The biped is driving, not walking: he may move his feet, but he doesn't move on his feet.

2. Again, adjust your perspective viewport and play the animation.
3. Use Ctrl + Alt + middle mouse button to quickly zoom in.



Freeform animation of driving biped

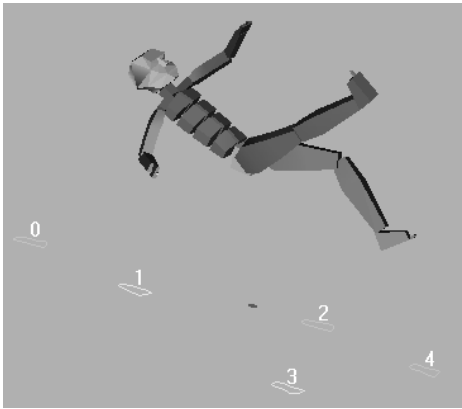
You can find a library of **character studio** motion files in the subdirectories under *cstudio\motions*. For a complete list of motion files see *Motion Library* (see page 284).

### Loading and viewing a combined footstep and freeform animation

1. On the Motion panel, click Load file and select *banana.bip*.
2. Play the animation.

This is an animation that combines footstep and freeform. This animation starts with footsteps and then has a period of freeform when the biped slips on an invisible banana peel and takes a pratfall.





**Footstep animation with Freeform period**

Added a Freeform Period to a Footstep Animation

### Saving the file as a 3D Studio MAX scene

1. From the 3DS MAX menu choose File > Save.
2. Type **mycstut1.max**, and click OK.  
The program saves your scene as a **.max** file in the default scenes directory. This file contains the figure and the animation in a single file. To save and load just the biped animation, use the load and save buttons in the General rollout on the Motion panel.

## Lesson 2: Modifying the Biped Structure in Figure Mode

Figure mode lets you change the structure of the biped, independent of any animation. The biped's basic structure and pose are stored in a figure file (**.fig**). Once you have saved a **.fig** file, you can activate Figure mode and apply that skeleton and pose to any biped. The animation adapts its motion keys to changes in the bipedal structure automatically when you leave Figure

mode. The same animation that works for a tiny or skinny character can work for a big or round one. You can apply any **.bip** file to any biped.

Use Figure mode to fit a biped inside a character's geometry. Position the mesh and skeleton so they overlap at the pelvis, and then rotate, scale or move the biped body parts into place.

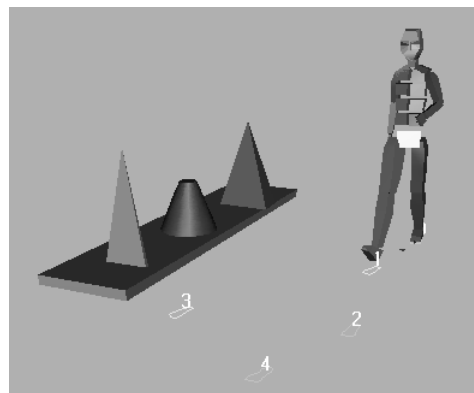
In Figure mode, you can use the same Structure rollout parameters you use during biped creation. You can change the overall height, the number of fingers, toes, and the number of spine links. You can also add a tail, or ponytail. You can specify an armless biped.

In this lesson, you will use Figure mode to change the structure of your biped. Then you will play back the animation to see how Biped has adapted keys to accommodate the changes in the character's proportions.

### Viewing Biped creation parameters

1. In the 3DS MAX main menu, choose File > Open **cs3\_tut01\_mod\_figmode.max**.

This is the same animation you were just using (**banana.bip**) with a few objects in the scene to give a sense of scale.



2. Select any part of the biped in the front perspective viewport.





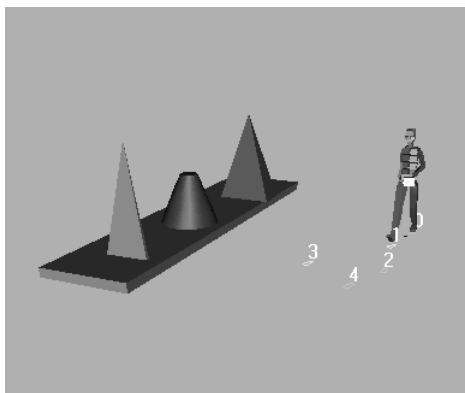
3. Click the Motion panel.

4. Close the Track Selection and Keyframing rollouts, and open the Structure rollout.


As you see, controls on the Structure rollout are the same as on the Create Biped rollout. These are the biped's *creation parameters*. The parameters are not animatable, so their controls are disabled unless you are in Figure mode.

## Changing Biped creation parameters

1.  On the General rollout, click Figure Mode.  
The controls on the Structure rollout become active.
2.  Depending on your zoom factor and viewport layout, the biped figure might disappear from view when you click Figure mode. If necessary, click Zoom Extents All to center the biped in your viewports. Zoom back in as necessary to frame the biped in the viewport.
3. Decrease the Height value by about half, then turn off figure mode.  
The size of footsteps decreases to accommodate the smaller biped. Play the animation, if you like.

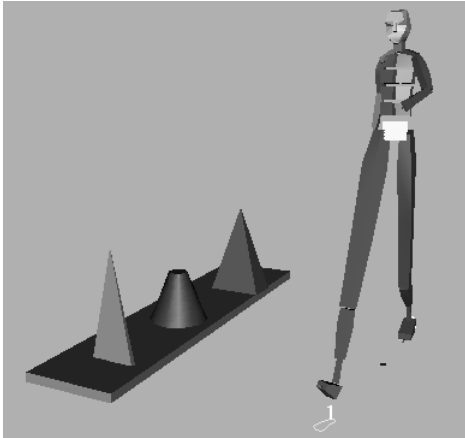


### Footsteps adapt to smaller biped

4. Turn on Figure mode again, and double the biped height. Turn off Figure mode and zoom to see the biped.  
The animation works on the larger biped as well.
5. Turn on Figure mode again, and change the number of Leg links to 4.  
Each leg now has an extra bone that does not correspond to human anatomy.  
You can make any kind of change to the biped body parts: the animation will still work.
6. Select one of the biped's thighs in the perspective viewport.
7.  Open the Track Selection Rollout and choose Symmetrical.  
This adds the other thigh to the selection. Now both thighs are selected.
8. On the main toolbar, click Select and Non-Uniform Scale from the scale flyout.  
A warning appears which cautions about using Non-Uniform Scale.
9. Click Yes to continue.

10. Using the Transform gizmo, scale the thighs in X so they are elongated.
11. Turn off Figure mode.

The biped returns to its animated pose in the current frame. Biped scales footsteps to match the biped's new dimensions. The animation accommodates the double-jointed legs.

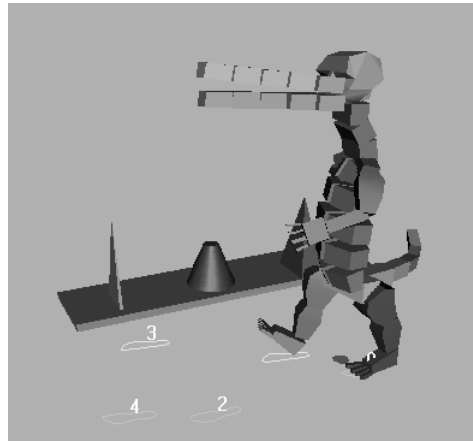


Animation adapts to changes in structure

### Scaling and Rotating parts of the biped

You can make some interesting characters by scaling and rotating the various biped pieces. To get some ideas, examine the sample *.fig* files in *cstudio/characters/figure*.

1. Turn on Figure mode.
2. On the Motion panel click Load File and open *gator\_up.fig*.  
This is the figure file of an alligator that walks upright. It's too small for the scene.
3. On the Structure rollout, increase the Height to 200.  
The gator increases in size.



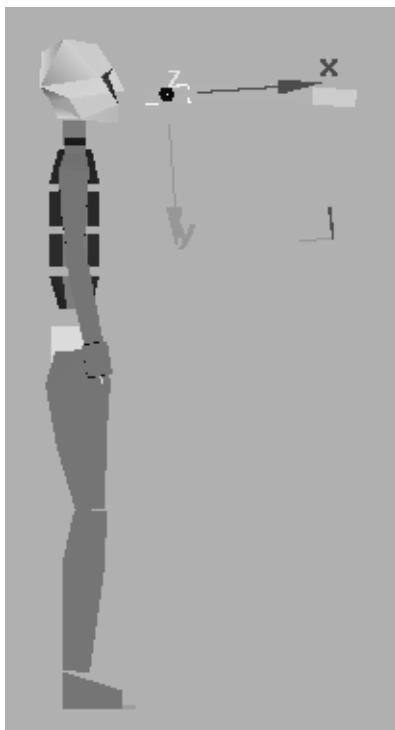
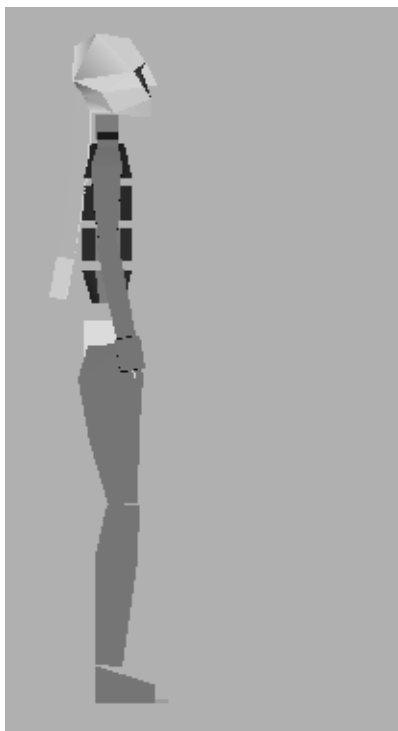
Scaling the biped using figure mode Height

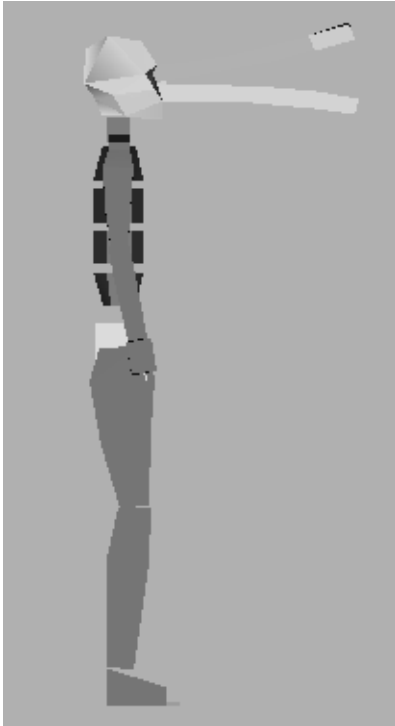
### Making Jaws with Ponytails

You can create bunny ears and alligator jaws by rotating and moving the ponytails. The alligator's jaw uses ponytails that are shifted to the front of the head.

1. Choose File > Reset to reset 3D Studio MAX.
2. Go to Create > Systems and click Biped.
3. In the Front viewport, create a biped.
4. On the Motion Panel, turn on Figure mode.
5. On the Structure rollout, change Ponytail1 Links to 5.
6. Change Ponytail2 Links to 5.  
**Tip:** Press Tab to move to the next field.
7. Zoom the left viewport.
8. Select *Bip01 Ponytail1* and *Bip01 Ponytail2* by pressing H and selecting each name.
9. Move the ponytails so they are in front of the biped's face.
10. Rotate the ponytails about Z, so the ponytails point out from the face.
11. Select one ponytail and rotate it 180 degrees about X.

12. Move the ponytails apart so your biped resembles the one in the illustration.





#### **Ponytails rotated and moved to the front of the head to create alligator jaws**


You can rotate and scale the different links to make the lips for convex or concave. Use PAGE UP and PAGE DOWN to move quickly through the hierarchy.

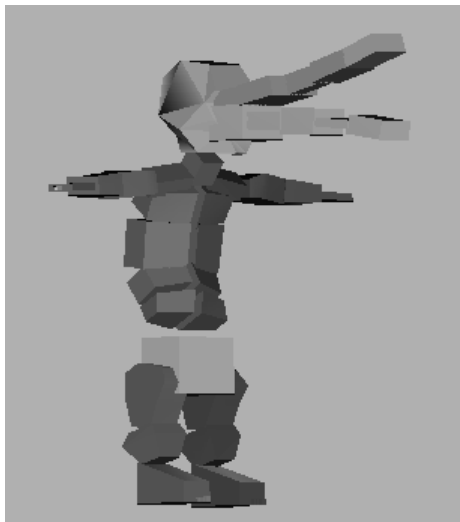
You can even add modifiers to the ponytail links to taper the bones. However, only the biped structure is saved in the *.fig* file. If you add extra bones or add modifiers to the biped objects, save a *.max* file.

Once you leave figure mode, you can animate the jaw using rotations. The ponytails are automatically linked to the head and will move and rotate with its movement.

### **Completing the Gator figure**

Next you'll give the gator a swayback by rotating the spline elements in Figure mode.



1.  In Figure mode, turn on Bend Links mode on the General rollout.  
Bend Links mode passes the rotations of links up the skeleton, affecting several links with one rotation.
2. Select and rotate *Bip01 Spine* about Z a few degrees.
3. Select the next link down (*Bip01 Spine1*) and rotate it a few degrees.
4. Turn Bend Links mode off.
5. Press PAGE DOWN to move down the chain to the next link.  
*Bip01 Spine2* is selected.
6. If any part of the legs becomes selected, press the ALT key and click the legs to remove them from the selection set.
7. Rotate *Bip01 Spine2* up to define the slouching posture.  
Notice the first two links are not affected by the rotation.
8. Finish off the alligator by shortening the spine links, legs, and arms. Rotate the arms up away from the torso.



#### Rotate the arms up away from the body

You use an outstretched pose to adjust the biped file inside a mesh.

#### Save your work

1.  Turn on Figure mode, if it is off.
2.  On the Motion panel, on the General rollout, click Save File.  
When Figure mode is active, Save File saves a *.fig* file. Otherwise Save File saves a *.bip* file.
3. Name the file *mygator.fig*, and then click Save.  
The biped's structure, size, and pose are stored as a biped figure file (*.fig*).
4. From the 3DS MAX menu Choose File > Save to save your 3DS MAX file as *mygator.max*.

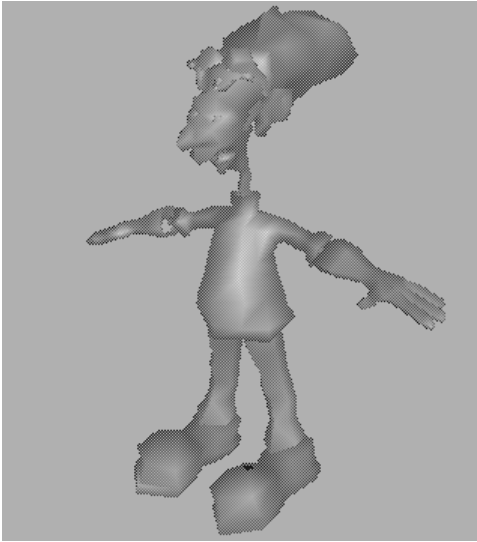
## Lesson 3: Biped with Physique

Use Biped and Physique together to create realistic character animations. You can create an animation using Biped, save the *.bip* animation file, and then apply that motion to any skinned biped. A skinned biped is one connected to a mesh in the form of a two-legged character. The mesh is attached to the biped figure using Physique. You can also use the Skin modifier that ships with 3D Studio MAX, but Physique gives you better control and extra options.

In this lesson, you will use the animation of slipping on a banana peel (*banana.bip*) that you viewed earlier. You'll apply that animation to the Dr. X character, which uses a Biped skeleton and 3DS MAX geometry with a Physique skin modifier.

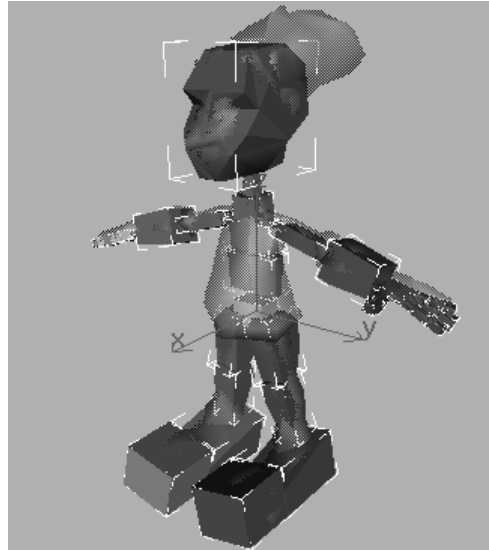
#### Loading a skinned biped

1. On the 3DS MAX menu, choose File > Open.
2. Choose *cstudio\tutorial\tutorial\_1\cs3\_tut01\_drx\_physique.max*. Click Open.  
You see Dr. X in a reference pose.



**The character mesh geometry**

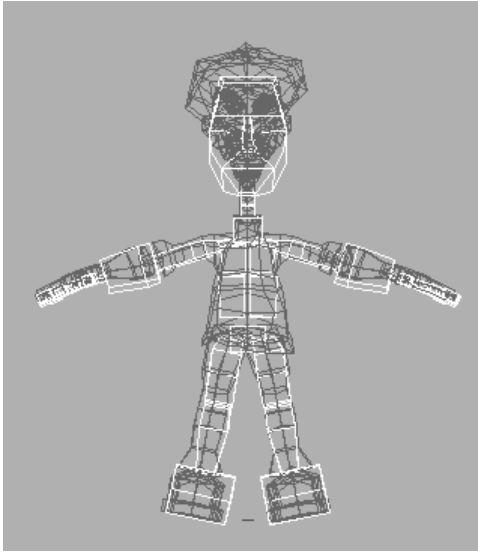
3. On the 3DS MAX toolbar, click Named Selection Sets and choose *biped skeleton*. This file contains several named selection sets, including some hidden objects. Hidden objects cause a warning to be displayed. Click Yes to continue.  
The biped is displayed in the same space as the character's mesh geometry.



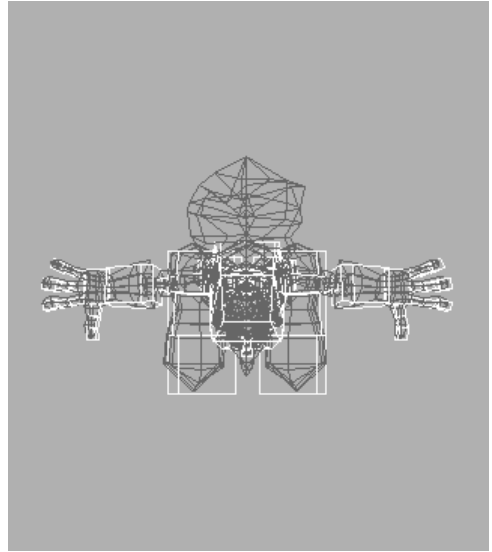
**The biped positioned inside the mesh**

This is a figure pose for Dr. X, but it's missing its ponytail. You can still see how the biped is aligned to the skeleton.

4. Enlarge the Perspective viewport to a single viewport by pressing W.
5. Arc Rotate around the character to see how the biped parts fit inside the mesh.




Dr. X figure pose from Front



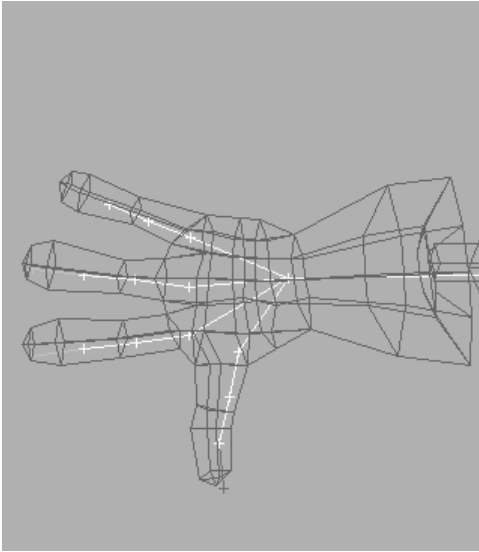
Dr. X figure from the top



Dr. X figure from side

6. On the Motion panel, on the Display rollout, click Show/Hide Objects to turn off display of the renderable biped geometry.
7.  On the Display rollout, click Show/Hide Bones to display the biped's bone structure. The biped skeleton is displayed as simple lines, so you can see how the biped figure was aligned to match the pose of the Dr. X mesh. The biped's fingers and toes are also aligned with those in the Dr. X mesh.








**Biped fingers displayed as links**

In this lesson, we won't go through how to fit the biped inside the mesh. You will do that in *Aligning a Biped to the Mesh Model* (Tutorial 7).

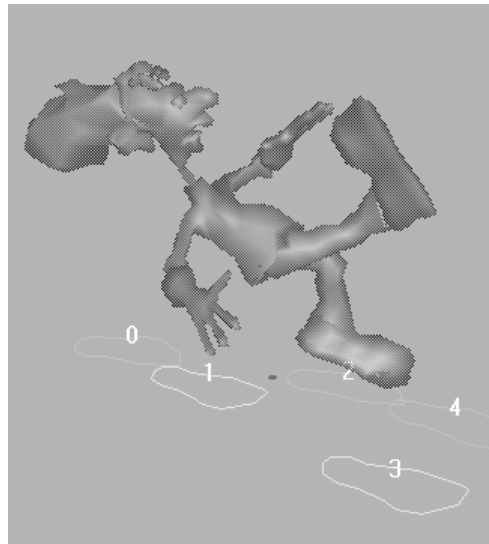
### Applying a motion file and playing the animation

1.  On the Display rollout, click Show/Hide Bones to hide the biped. You can see the mesh of Dr. X in the viewports. The biped is not visible because its renderable elements and bones are hidden. Nevertheless, the biped is still present and can control the motion of the mesh.
2. Turn off Figure mode.
3.  On the Motion panel, on the General rollout, click Load File and choose *cstudio/tutorial/banana.bip*.

The biped is moved to the first frame of the animation.

4.  Click Zoom Extents to center Dr. X and the footsteps.
5. Right-click the viewport label.
6. From the Viewport Properties menu choose Smooth + Highlights.  
Dr. X is displayed as a solid, smoothed figure.
7. Use the time slider to move to different frames in the animation.

In different frames, you see Dr. X walking and slipping on the banana peel. This is the same animation you viewed with the biped.



You can use the 3DS MAX playback button to preview an animated mesh like the Dr. X. If this playback is dropping frames, you can get a better preview of the timing of a motion by making a Preview.

### Creating a Preview

1. Choose Rendering > Make Preview to create a preview animation.
2. On Preview Range, turn on Custom Range. Set it from 0 to 100.
3. Adjust the resolution of the preview under Image Size by setting the Percent of Output spinner. Set the spinner to 24% for a small preview or 50% for a larger preview. The greater the speed of your computer, the larger the preview you can play.
4. Click Create to render the Preview.

### Viewing the animation

1. On the 3DS MAX menu, choose File > View File.
2. Choose `cstudio\tutorial\dr_x_banana.avi` from the list.

The Media Player is loaded and displays the animation at frame 0.

3. Click the Play button.

The Dr. X moves with the motion you applied in previous steps.



---

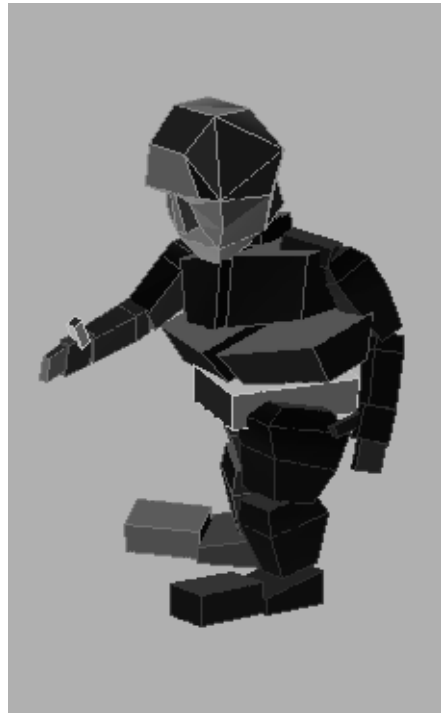
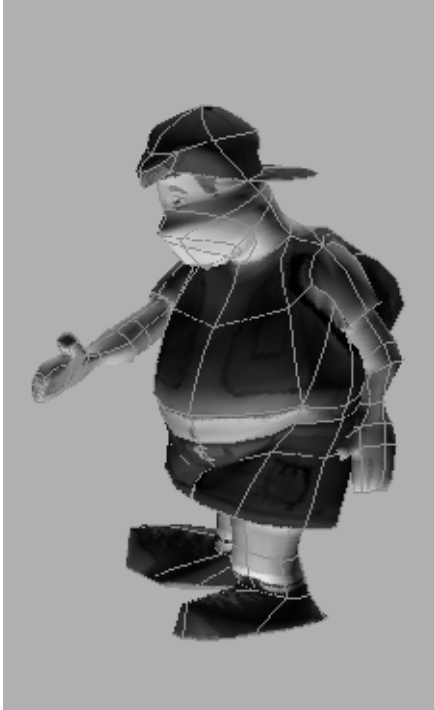
## Lesson 4: Merging and Cloning Characters

If you merge a character into your scene from a different *.max* file, the characters may vary in size and you may have to scale them. In this lesson, you'll merge an existing character into a different scene. Because the character is really a biped linked through Physique to 3D Studio MAX geometry, you'll need to use a particular technique.

You'll merge the character of Tubby McChubbs into a scene with Dr. X and scale him to fit in the scene. You'll also create a sidekick for Dr. X by cloning and scaling a copy of the mad scientist.

### Setup

- Open *cs3\_tut01\_tubby.max*.  
This file contains Tubby McChubbs, the hero of an adventure game.



Tubby McChubbs and his inner biped

**Tip:** You can hide and show the biped skeleton using the Biped Display rollout.

### Changing the Biped Root Name

When you merge a character into a scene, change the name of the biped, to be sure it's unique.

1. Select any part of the biped.
2. In the Motion panel, on the Structure rollout change the Root Name to *tubby bip*.

**Tip:** When you change the root name in the Structure rollout, you change all the biped part names at once. To change the name of one part at a time, use the Name field at the top of the command panel.

3. In the Main toolbar click Select by Name.  
All the biped names start with *tubby bip*.
4. Save the file as *cs3\_tut01\_myhero.max*.

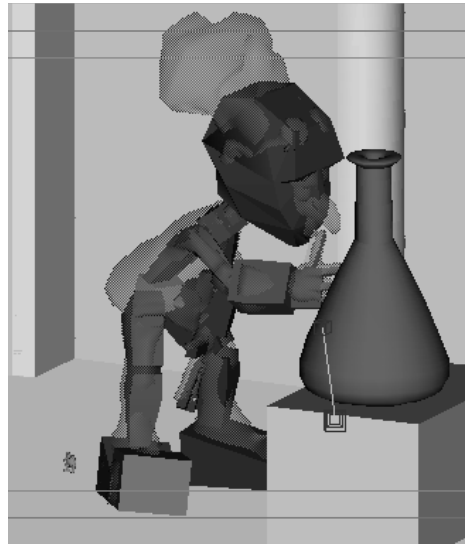
### Merging the character

1. Open *cs3\_tut01\_drx\_beaker.max*.  
Dr. X is peering into a beaker.



**Dr X is alone in the scene**

2. Choose File > Merge, and select *cs3\_tut01\_myhero.max*.  
You'll see a thumbnail image of Tubby when you select the file name.
3. Select All and click OK.  
Note: If any geometry in the incoming file has the same name in the current file, a dialog lets you change the names of the incoming objects.  
The track bar now displays keys.  
The small merged character is near Dr. X's feet.



### Make Named Selection Sets

Make named selection sets for your character after you've merged it.

1. In the Select by Name dialog enter *tubby bip* in the Select Objects field, then click Select.  
All the parts of the tubby biped are selected.
2. In the Named Selection Sets field in the Main toolbar, enter *tubby biped*.  
You've just created a named selection for the merged biped skeleton.



3. In the Display rollout of the Motion panel, turn off Objects.  
This hides the biped skeletons, just as if you hid it using the Display panel.
4. Press H to select by name again.
5. Highlight *tubby patch splines* and then click Select.
6. In the Named Selection Sets field enter *tubby mesh*.

You've just created a second named selection set for the mesh. Next you'll create a selection set that contains both the mesh and the biped.

7. Select *tubby biped* from the Named Selection Sets list.
8. A warning tells you that the biped is hidden. Click Yes.  
The biped appears.
9. Press H to open the Select by Name dialog.
10. Enter *tubby* in the Name field.  
This selects both the biped and the mesh.
11. In the Named Selection Field, enter *tubby all*.

### Finding the Merged Character

When you merge files, you may not be able to find the merged characters because of size or location. You can select them by name, and then find them using Zoom Extents Selected. Then you can scale them using Type-In Transform to adjust their size.

1. Right-click to activate the User View.
2. Choose *tubby all* from the Named Selection Sets list.
3. Choose Zoom Extents Selected from the Viewport Navigation controls.  
Tubby is now clearly visible in the User Viewport.

### Scaling the Character

Scale the merged character so it fits in the scene.

1. Select *tubby bip*.
2. In the Motion panel, turn on Figure mode.
3. Change the Height parameter to 600 on the Structure rollout.  
The biped and the mesh scale in a single movement.

**Tip:** Don't select the mesh and the biped and try to scale it. Change the biped's height instead.

Now Tubby is the right size, but in the wrong place, since he's already animated in this file. It's easy to reposition the entire character and animation by converting to footsteps.

### Repositioning the Character and Animation

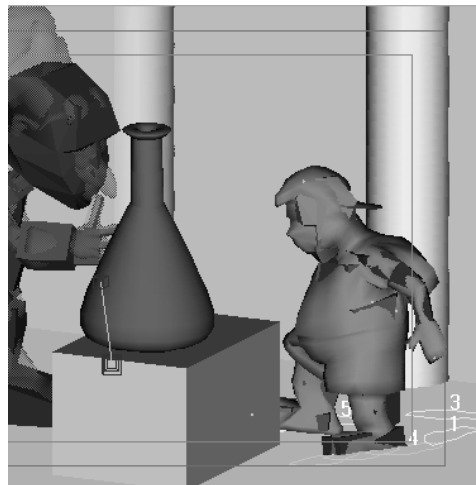
1. Turn off Figure mode.



2. In the General rollout, click Convert to Footsteps.

Accept the default settings in the Convert to Footsteps dialog.

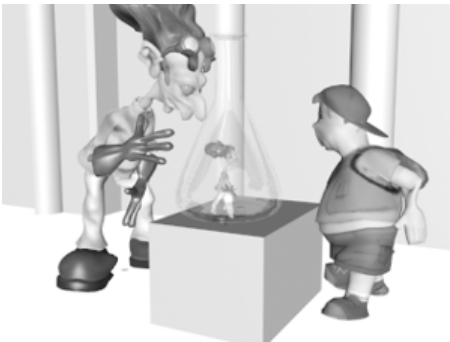
3. Turn on Footsteps mode.
4. Select *tubby's* footsteps in the Top viewport.  
**Tip:** In Footstep mode you can only select footsteps, you can't select geometry.
5. Move the footsteps, so Tubby McChubbs is on the other side of the box.



## Cloning a Character

Create a miniature sidekick (Dr. Y) by cloning and scaling Dr. X.

1. Turn off Figure mode, if it is on.
2. In the Named Selection Sets, choose *Scientist*.  
This selects Dr. X mesh and biped.
3. From the Edit menu choose Clone.  
This clones the mesh. You won't see a change in the viewport, since the clone is in the same location as the original.
4. In the Clone Options dialog name the clone *Dr. Y*.  
Next you'll need to scale the character.
5. Select the biped root of the cloned copy, and turn on Figure Mode.  
Use Select by Name and pick *Bip03* in the list, and click OK to select the root of the cloned copy.
6. On the Structure Rollout change the Biped's height to make the character smaller.  
The clone is smaller than the original.
7. Turn off Figure mode.
8. Select the footsteps of the clone and move them to another location to reposition Dr. Y.  
For fun, try scaling Dr. Y and putting him in the beaker.



---

## Tutorial 2

# Freeform Biped Animation

In this tutorial, you'll learn how to work with freeform Animations. You'll do the following:

- Learn about freeform animation technique
- Create a simple animation of a biped swimming in place
- Create a traditional walk cycle using animated pivot points
- Learn to use planted, sliding, and free keys

## Lessons

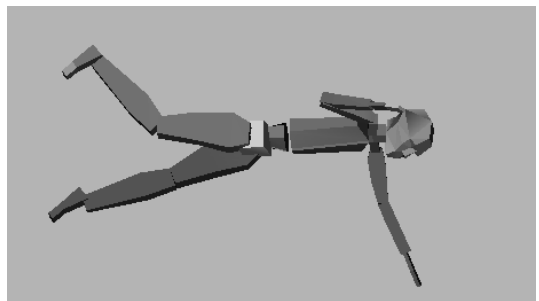
*Lesson 1: Creating a Simple Freeform Animation (see page 447)*

*Lesson 2: Animating a Freeform Walk Cycle (see page 457)*

---

## Lesson 1: Creating a Simple Freeform Animation

This tutorial is an introduction to using freeform animation techniques with Biped.



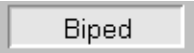

In this tutorial, you will animate a biped swimming in place. You'll use freeform animation methods.

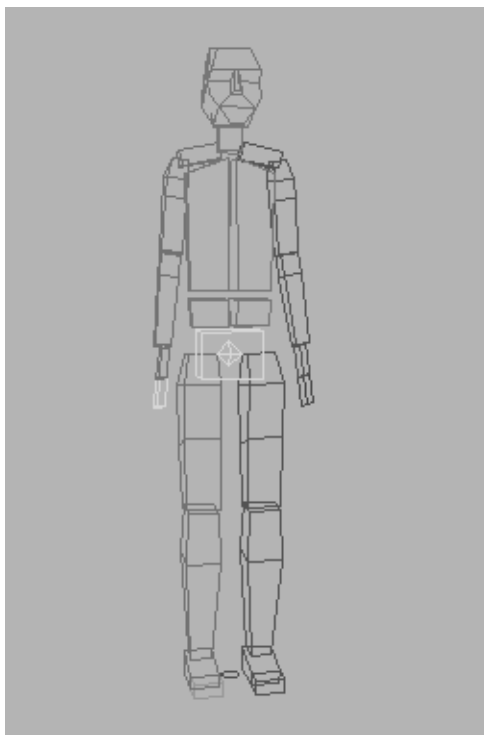
You'll animate the biped kicking his legs using rotations and moves, and Copy and Paste Posture Opposite. You'll animate one arm and copy its tracks to the other arm.

#### Setup


- Reset 3D Studio MAX.

#### Creating a biped and loading a fig file

1.  Create a biped in the Front viewport.
2.  On the Motion panel, turn on Figure mode and load *rtgame.fig*. This file contains a simple figure with 1 large toe per foot and 1 large finger per hand.




Biped with one toe and one finger

3.  Turn off Figure mode.  
**Tip:** You can't animate in figure mode.
4. Select the biped objects and zoom in using Zoom Extents All.
5. Zoom out a little using Zoom all.

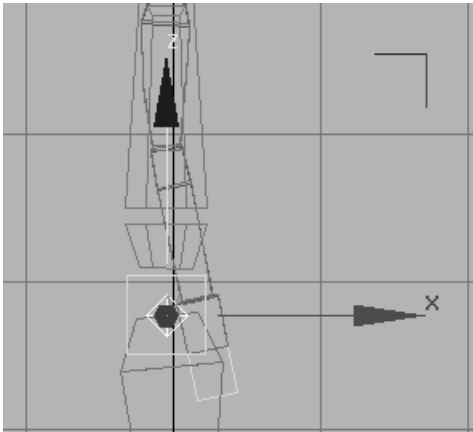
#### Starting a freeform animation

You start a freeform animation by clicking the Animate button and transforming any part of the biped. You'll rotate the biped's root object so it is lying prone.

1. Click in the open space in the Left viewport. This deselects all the biped objects and activates the Left viewport.
2. Press W to maximize the viewport for a closer view of the biped.  
The biped should be in wireframe. Change the shading display of the Left viewport if it is not wireframe.

3.  Turn on Animate .  
The button turns red and the active viewport is outlined in red.
4. Click Select and Rotate on the Main toolbar.  
**Tip:** You can select objects by clicking on them in the viewport, or you can select by name. You'll select the biped center of mass by clicking on it.
5. In the Left viewport, click the small, blue, center of mass object.  
The name *Bip01* appears at the top of the command panel.  
The Transform gizmo appears.





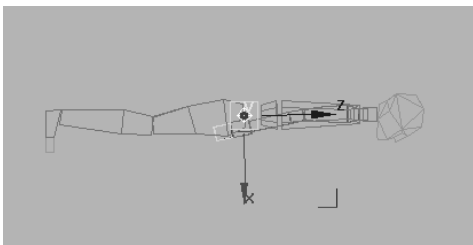
### Transform gizmo

The Transform gizmo lets you easily move, rotate, or scale on a chosen axis. As you move your cursor over the gizmo in the viewport the arrow lines and labels turn yellow.

6. Move your cursor around, over the center of mass until the Y label turns yellow. Since the arrow is pointing at you, you can't see the line.
7. Rotate the center of mass approximately 90 degrees about the Y axis.

The rotational values update in the status line below the viewport as you move your cursor.

The biped rotates so it is lying prone.



### Rotate the biped

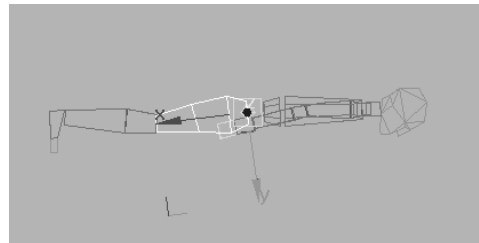
A key appears at the far left of the track bar. You are ready to animate the biped swimming. First you'll position the legs.

### Posing One Leg

You'll work on the right leg first, setting up its position at frame zero.

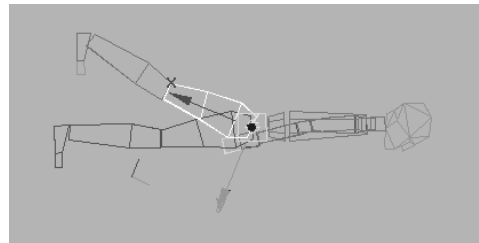
1. Press **W** so you can see four viewports instead of one.
2. Select *Bip01 R Thigh* by clicking the lines of the thigh in the Left viewport.

**Tip:** As you hold your cursor over an object in the viewport, the object's name is displayed in a tooltip.



### Select the right thigh.

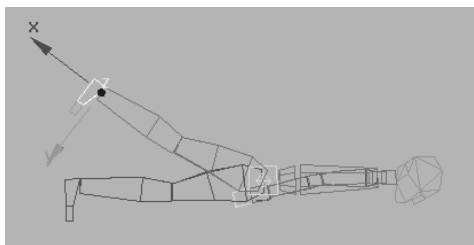
3. Rotate the right thigh approximately -30 degrees about the Z axis.



### Rotate the leg up.

The right foot is pointing straight down. You can rotate it to make it look more natural.

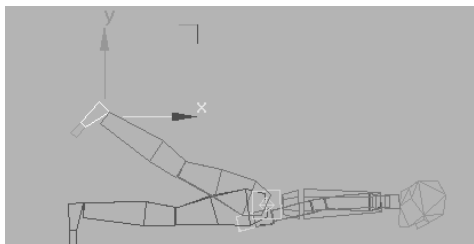
4. Select and rotate the right foot about 40 degrees about the Z axis.



#### Rotate the foot.

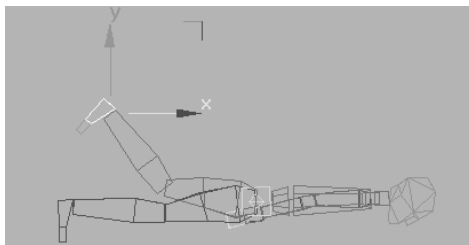
You've just used 3D Studio MAX rotations or "forward kinematics." Next you'll use the Select and Move tool on the foot to move the entire chain of foot, calf and thigh.

5. Right-click the same foot and select Move.  
**Tip:** You can choose the transform tools from the Main toolbar or by right-clicking. The Transform gizmo arrows switch their display. They are displayed at right angles with Y pointing up and X pointing right.



#### Move the foot.

6. In the Left viewport, use the transform gizmo corner to move the foot a little to the right.  
The knee bends to accommodate the new position of the foot.



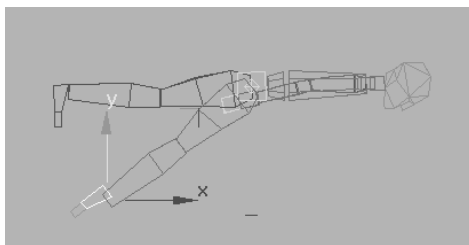
#### The knee bends.

You've just used "inverse kinematics." The foot, calf and thigh are linked together in a hierarchical chain. By moving the end of the chain you rotated the lower and upper leg objects.

### Animating the Leg

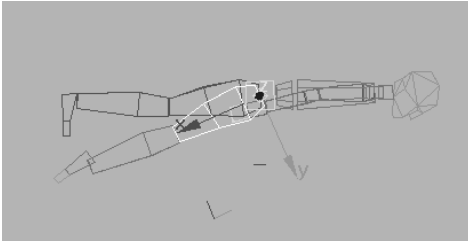
Everything you've done so far has been at frame zero. Now you'll move forward in time and animate the pose at frame 10.

1. Move the time slider to frame 10.
2. Move the foot down in the Y axis until the knee straightens out.



#### Move the foot down.

3. Select the right thigh.
4. Right-click and choose Rotate, then rotate the thigh approximately -13 degrees about the Z axis.



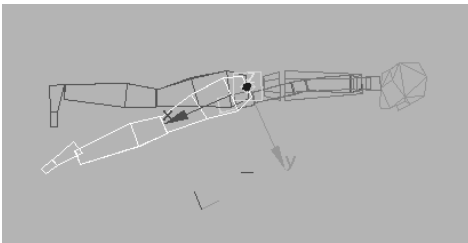
### Rotate the thigh.

Move the time slider back and forth between frame 0 and frame 10. The leg moves up and down.


### Using Copy Posture and Paste Opposite

Now you'll use some specialized Biped tools to pose and animate the opposite leg.

1. Return the time slider to frame 10.
2. Double-click the right thigh.  
This selects the entire leg from the thigh down to the toes.

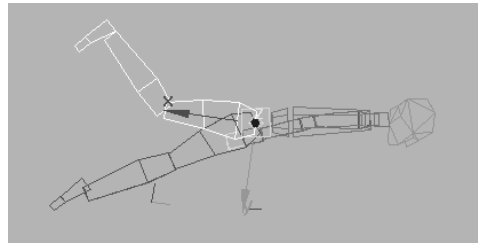


### Double-click to select the leg.

3.  On the Motion panel, under the Keyframing rollout, click Copy Posture. The posture of the right leg is copied into a buffer.





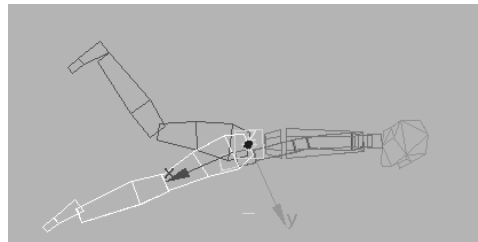
4. Move the time slider back to frame 0 and choose Paste Posture/Pose/Track Opposite.



### Paste Posture Opposite

The left leg rotates downward. The right leg hierarchy is still selected.

4.  At frame 0, choose Copy Posture again.
5. Move the time slider to frame 10.
6.  Click Paste Posture/Pose/Track Opposite again.  
Now the left leg is raised, and the right leg is down.




### Repeat Paste Posture Opposite.

Move the time slider back and forth between frames 0 and 10 and watch the legs kick.

Now you'll repeat this process to make the legs kick several times.

### Using Paste Posture to create multiple kicks

You can use the Copy Posture tools to quickly copy all the leg keys from one frame to another to create repeated kicking motions.

1. Move the time slider to frame 0.  
Be sure that Animate is still on.
2.  On the Track Selection rollout Choose Symmetrical.  
This adds the second leg to the selection. Now both legs are selected.
3. Click Copy Posture at frame 0.
4. Move to frame 20.  
**Tip:** You can type in the frame number in the Current Frame time control.
5. Paste Posture at frame 20.
6. At frames 40, 60, and 80, Paste Posture.  
**Tip:** You can quickly move to these frames by typing in the Current Frame Indicator in the time controls.  
You've just made copies of the leg postures. The legs are now in the same poses at frames 0, 20, 40, and 60.
7. Move to frame 10 and Copy Posture.
8. At frames 30, 50, and 70, Paste Posture.  
The trackbar displays seven keys for the animation of the legs.
9. Save your file as **myswim1.max**.  
**Tip:** If you have AutoBackup turned on, the software will automatically save your work at regular intervals to a file with a *.mx* extension.

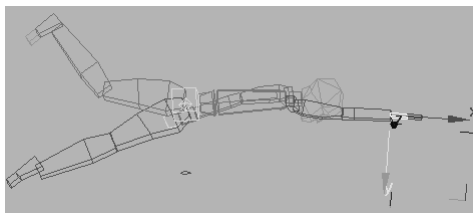
### Animating one arm

When you animated the legs you set two different poses: one with the leg up, and one with the leg down. Animating the arms is more complex. To animate the stroke of an arm, you'll set four poses:

- One for the arm outstretched
- One with the arm down
- One with the arm back
- One with the arm drawn up near the chest

When one arm is animated correctly, you'll use Copy Track and Paste Opposite Track to animate the second arm. You'll adjust the timing of the second arm by sliding the keys in the track bar.

1. Be sure Animate is still on.
2. Move the time slider to frame 0.
3. In the left viewport, select and rotate *Bip01 L UpperArm* approximately -160 degrees about the Z axis, until it is extended in front of the biped.
4. In the Top viewport, select the *Bip01 Left Clavicle* and rotate it -20 degrees about the Y-axis.  
This should prevent the arm from passing through the head.
5. In the same viewport, select and rotate the *Bip01 Left Hand* -90 degrees about the X-axis so the palm is facing down.

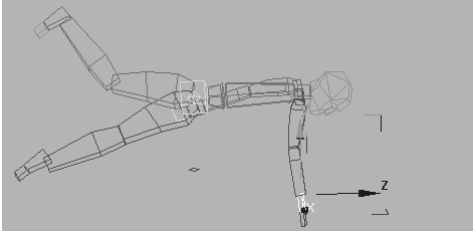


First arm pose

6. Move the time slider to frame 10.
7. On the Main toolbar, choose Select and Move, then change the Reference Coordinate system to World.  
This will facilitate working with the transform gizmo in different viewports.

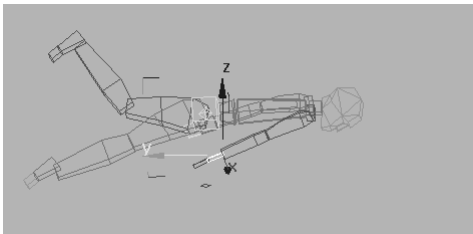
8. In the left viewport, move the hand down in Z and back in Y until it points straight down. Select and rotate the *Bip01 Left Clavicle* approximately 7 degrees about the Y axis.

In the Front viewport move the hand slightly in towards the body.



**Second arm pose**

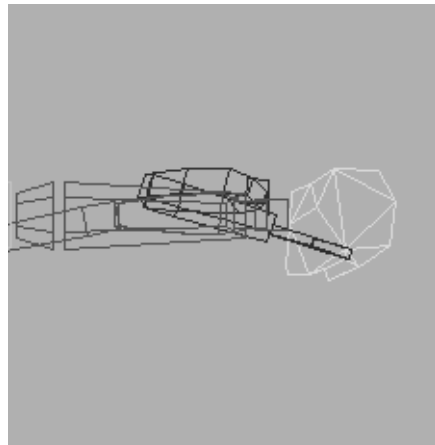
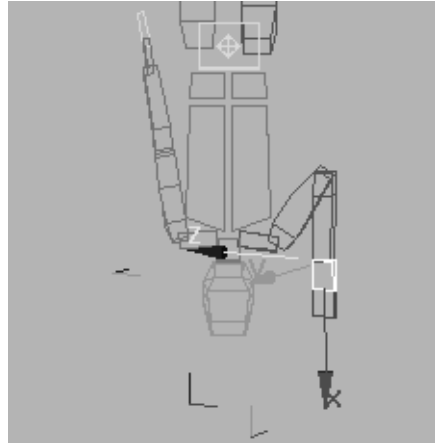
9. Move the time slider to frame 20.
10. Using the Transform gizmo, in the Left viewport move the hand in Y towards the legs.
11. In the Front viewport, select and rotate the clavicle 24 degrees about Z.



**Third arm pose**

12. Move the time slider to frame 30.
13. In the Top viewport, select and move the hand in X and Y until the hand is level with the shoulder.  
Use the Transform gizmo corner to move in both X and Y at the same time.
14. In the Left viewport move the hand up so it is near the ear.

15. In the Top viewport rotate the upper arm around the Z axis approximately 30 degrees.
16. In the Front viewport rotate the hand about X so the palm is flat.



**Fourth arm pose from Top and Left viewports**

## Copying the Arm Pose

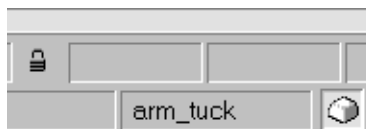
To complete the arm cycle in the next few steps, you'll copy the arm pose from frame 0 to frame 40.

1. Move the time slider to frame 0. In the Top viewport double-click the *Bip01 Left Clavicle*.  
This selects the entire arm hierarchy.
2. On the Keyframing rollout, click Copy Posture.
3. Advance to frame 40 and click Paste Posture.  
If you see any unusual rotations or out of place movements, you can set additional keys to refine the animation.  
Move the time slider and see the animation.

## Adding Time Tags

You can use Time Tags to name the keyframes. This makes it easier to retrieve poses while animating.

1. Double-click the *Bip01 Left Clavicle* to select the entire arm chain.
2. Move the time slider to frame 0.
3. In the blank Time Tag area to the left of the Plug-in Keyboard Shortcut Toggle, click and choose Add Tag.
4. In the Tag Name field, enter **arm\_outstretched**. Click OK.
5. Repeat this process to add a time tag named **arm\_down** at frame 10.
6. At frame 20, add a time tag named **arm\_back**.
7. At frame 30, add a time tag named **arm\_tuck**.





### Use Time Tags to name poses.

Once you have added time tags you can jump to that frame by clicking the Time Tag window and choosing the named time from the list.

Note: Time tags do not get copied with Copy Posture.

## Repeating the animation

If the animation is going to be 80 frames, you'll need to repeat the arm movement.

1. Be sure the animate button is still on.
2. Select the entire arm by double-clicking on *Bip01 Left Clavicle*, if the arm isn't selected already.
3.  At frame 10, Click Copy Posture.
4.  At frame 50, click Paste Posture.  
Do this whenever you need to copy multiple keys for multiple objects at one particular keyframe.
5. Repeat steps 1 through 4, copying the poses from frame 20 to frame 60, 30 to 70, and 40 to 80.
6. Save your work as **myswim2.max**.

## Adding rotation for the spine

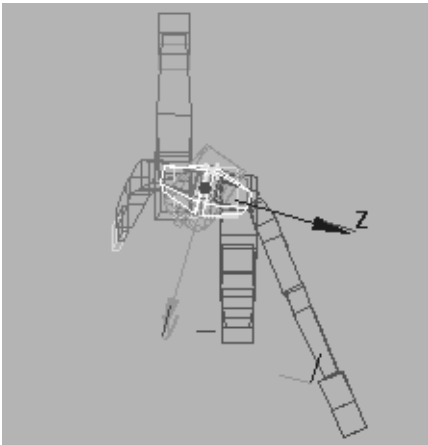
Next you'll add some rotations for the spine to make the animation more convincing. This biped figure (*rtgame.fig*), only has a 2 segment spine. You'll rotate the large section representing the upper torso.

1. Select *Bip01 Spine1*.

Note: The first spine object is *Bip01 Spine*. The large second spine object is *Bip01 Spine1*.

2. At frame 10, in the Front viewport, rotate the large spine object -15 degrees about the X axis.

This will create the appearance that the body follows the movement of the arm.



**Spine rotation**

3. Move the time slider to frame 0 and rotate *Bip01 Spine1* 15 degrees about the X axis. This sets a key for the rotation.
4. Hold down the SHIFT key and drag to copy the track bar key from frame zero to frame 30. Watch the status area to know when you are at frame 30.  
The spine now rotates once in the 40 frame cycle. Next you'll copy these spine rotation keys to repeat the motion.
5. Move the time slider to frame 40.
6. In the track bar, drag a selection rectangle around the three visible keys.
7. Hold down the SHIFT key and drag the keys so the leftmost key is copied to frame 40.

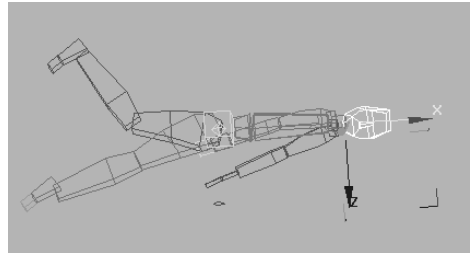
Move the time slider back and forth to see the animation.

8. Copy the key from frame 0 to frame 80.

### Animating the head

The biped can breathe while he swims, if you rotate the head appropriately.

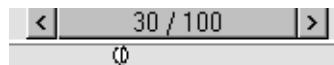
1. At frame 0, rotate the head about the X axis, so the biped's left ear is pointing down.



### Rotate the head for breathing motion.

**Tip:** Look in the Perspective viewport while rotating in the Left viewport.

2. At frame 20, rotate the head back down.
3. Hold down the SHIFT key and drag to move the key on frame 0 to frame 40. Watch the status area to know when you are at frame 40.
4. Move the time slider to observe the head rotation.  
Actually it would look better if the head was turned up at frame 30.
5. Slide the key for the head from 20 to 30.
6. **Tip:** move the time slider to frame 30, then slide the key on top of it.



### Move to the frame, then slide the key




The swimmer lifts and lowers his head once in the 40 frame cycle.

The following steps show you a different way to achieve the same result.

7. Right-click the time slider.  
The Create Key dialog is displayed. This lets you create keys by choosing a source and a destination.
- Tip:** You don't have to turn on Animate, to set keys this way.
8. Choose frame 30 as the Source Time and frame 60 as the Destination Time, then click OK.
9. Right-click the time slider.
10. Choose frame 0 as the Source Time and frame 40 as the Destination Time, then click OK.
11. Right-click the time slider again. Choose 0 as the Source Time and 80 as the Destination Time, and choose OK.  
You still don't have the arm motions completed for the right arm. That comes next.

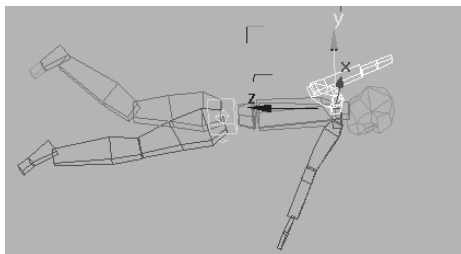
### Animating the other arm with Copy Tracks

Copy Tracks lets you copy and paste the animation tracks of selected objects to other objects or opposite body parts.

1. Double-click the *Bip01 Left Clavicle*.  
This selects the entire arm.
2.  In the Keyframing rollout, press and drag on Copy Posture.
3.  Choose Copy Tracks.
4. Choose Paste Posture/Pose/Track Opposite.
5.  Play the animation.  
The biped is swimming the butterfly stroke. The two arms move together.

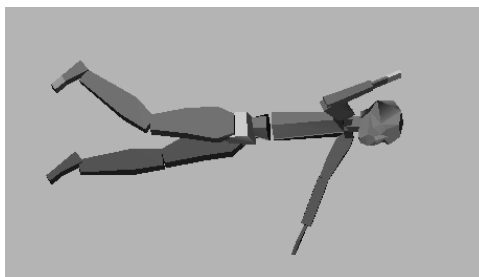
You can change the timing so the arms alternate.

6. Double-click the *Bip01 Right Clavicle*.  
The entire right arm is selected in the viewport.



**Double-click the clavicle to select the whole arm chain.**

7. Drag a box around all the keys in the track bar to select them. Slide all the keys to the right, 20 frames.  
Play the animation.  
Now the beginning and end are not quite right. The easiest way to correct this is to copy and paste poses.
8. At frame 50 click Copy Posture, then at frame 10 Paste Posture.
9. At frame 40 click Copy Posture, then at frame 0 Paste Posture.



**The arms alternate**

To correct the other end of the animation you can crop the animation to 80 frames.



10. In the time controls, click Time Configuration.  
The Time Configuration dialog is displayed.
11. In the Animation group, change the End Time to 80. Click OK.  
**Warning:** Do not click Re-scale Time.  
Play the animation. The swimmer swims endlessly, because the animation is looped. The last frame matches the first frame.

## Perfecting the animation

You can improve the animation by adding some rotation keys to the pelvis and spine and by adding secondary motion to the feet and hands, if you like. Stagger the rotations of the extremities a few frames following the movement of the hands and feet.

## Save your work

Save your BIP file as **myswimmer.bip**. Use Save File in the General rollout to save the BIP file. You can also save your entire MAX file as **myswimmer.max**.

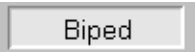

## Lesson 2: Animating a Freeform Walk Cycle

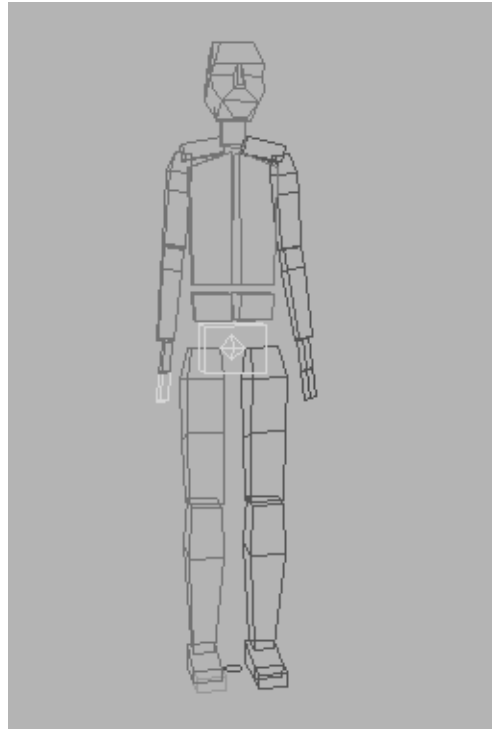
While **character studio** has a entire mode (Footstep mode) for creating quick and easy walking animations, you can also create walk cycles with freeform animation. You'll use animated pivot points and IK blend keys to constrain the feet to the ground plane.

### Setup

- Restart or reset 3D Studio MAX.


### Creating a biped and loading a figure file

1.  Create a biped in the Front viewport.
2.  On the Motion panel, click Figure mode and load *rtgame.fig* for a simple figure that has 1 large toe and 1 large finger.



### Biped with one toe and one finger

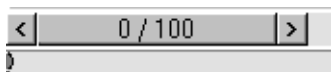
This will work quicker than a file with more toes and fingers.

3.  Click Figure mode again to turn it off.  
**Tip:** You can't animate in Figure mode.

4. Select the entire biped and click Zoom Extents All.
5. Change the Perspective viewport to wireframe and zoom in so the feet are clearly visible.
6. Select *Bip01 R Foot*.



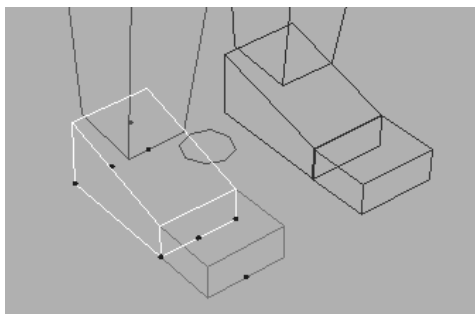
7. On the Motion panel, in the IK Key Info rollout, click Set Key.



#### Track bar key at frame zero

The foot is highlighted in white, and a key appears in the track bar. You have just started a freeform animation.

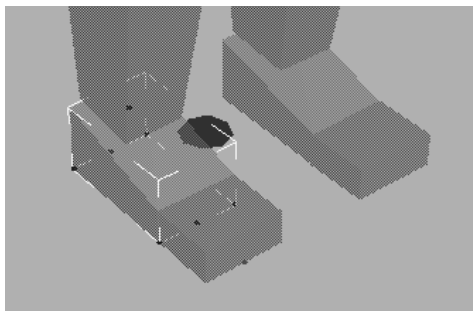
8. On the IK Key Info rollout, click Select Pivot. The pivot points for the foot are now visible. The pivot at the ankle is red, showing that this is the pivot point currently selected.



#### Pivots in wireframe display

Wireframe mode lets you see and select the pivot points.

**Tip:** To select pivot points in Smooth and Highlight shading, use See-Through Display mode (**ALT+X**).



#### Pivots visible in See-Through display

9. In the Perspective viewport, click the pivot point at the heel.

The pivot point at the heel turns red.

**Tip:** You must set a key before you can select a pivot in the viewport.

#### Setting different types of keys at frame zero

There are two ways to set keys in **character studio**. You can turn on Animate and transform objects. This is the standard 3D Studio MAX method of keyframing. It is quick and easy, but if you forget that Animate is on, you may set keys unintentionally.

Instead, you can use the set keys buttons on the Key Info and IK Key Info rollouts. The Set Key buttons set up several parameters at once.

1. Turn off Select Pivot.



2. In Track Selection, choose Body Vertical.



This selects the biped center of mass, and activates the move icon in one step. You've set a key for the foot, but there is a problem. The foot can go through the ground plane. See for yourself in the next several steps.

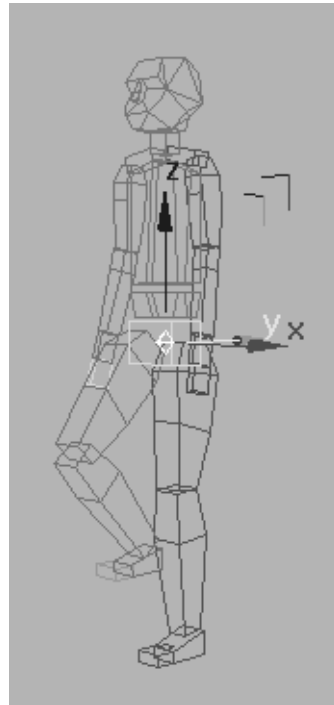
3. Right-click to activate the Left viewport without changing the selection set.

4. Move the center of mass down in the Left viewport.  
The biped moves down through the ground plane (as indicated by the grid in the Perspective viewport).
5. Press **CTRL+Z** to undo.

### Setting Planted Keys

Now, you'll set a planted key. A planted key will do three things: it will set IK Blend to 1, it will turn on Join to Previous IK Key, and it will turn on Object Space. Together, these three settings ensure that the foot will not go through the ground plane.

1. Select *Bip01 R Foot*.
2.  On the IK Key Info rollout, click Set Planted Key.  
The red pivot point becomes more pronounced.
3.  On the Track Selection rollout, click Body Vertical, and move the biped down in the left viewport.  
The foot stays on the ground plane, the knee bends to accommodate the vertical movement of the biped.




**Planted foot stays on ground**

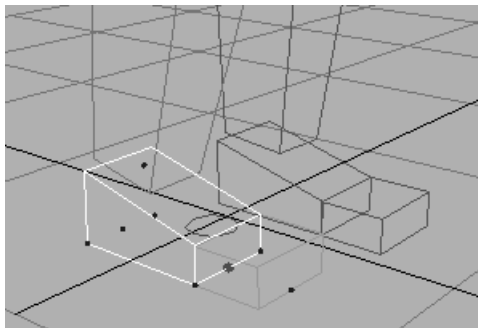
4. Press **CTRL + Z** again, to reverse the movement of the center of mass and return the biped to its original position.

Now you've seen the effect of the planted key on the foot. We can use the same Set Key buttons on pivot points for the feet and hands. We'll replace the key at frame 0 with a new one, changing the pivot point.

### Setting Pivot Keys

1. At frame 0, select the *Bip01 R Foot* in the Perspective viewport.
2.  On the IK Key Info rollout, click Set Planted Key.
3. Turn on Select Pivot.

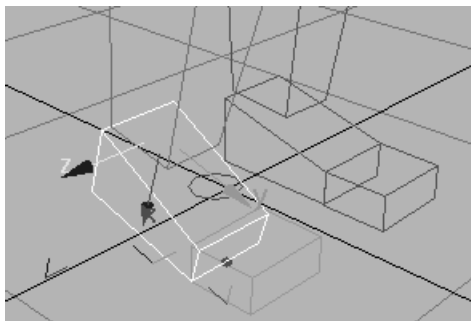
- Choose the pivot point in the middle of the front of the foot, at the base of the toes. The pivot point is displayed in red.



**Ball of the foot pivot selected**

You don't have to set a key after each time you choose the pivot point. However, you should use the set key buttons if you want to change the IK Key parameters.


- Advance the time slider to frame 5, click Set Key, then click Select Pivot again to turn it off.
- Right-click the foot and choose Rotate. Rotate the foot up approximately 15 degrees about the Z axis and click Set Planted Key. The heel lifts off the ground, the foot rotates on the ball, and the toes stay on the ground.

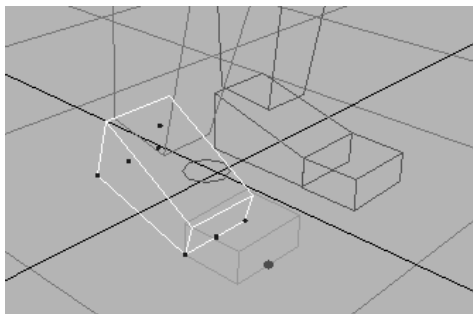


**Rotate the foot on the ball pivot.**

Now, we can animate the pivot point to the toes, as the ball lifts off the ground.

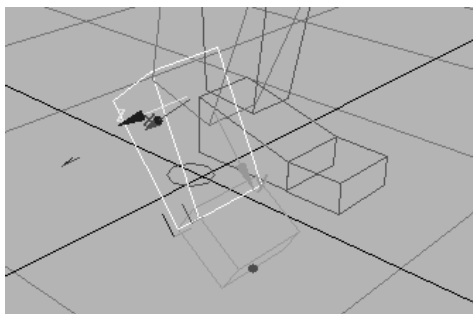
### Animating the Pivot Points

- Go to frame 10, and click Set Key.
- Click Select Pivot and choose the pivot on the end of the toe.
-  Click Set Sliding Key to keyframe the pivot.



**Select the toe pivot.**

- Click Select Pivot again, to turn it off.
- In the Perspective viewport right-click the foot and choose rotate.
- Rotate the foot so the heel continues to raise and roll off the toes.
- Click Set Sliding Key to keyframe the foot rotation.




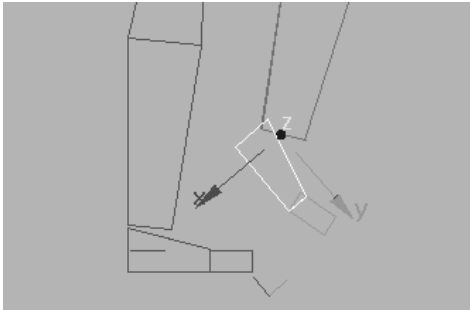
**Rotate the foot.**

The sliding key does not join to the previous IK key, but has IK Blend set to 1, which keeps the foot above the ground plane. If you had set a planted key, the foot would jump to a different location as it attempts to join to the previous IK key.

When the foot lifts off the ground completely, you'll set a free key.

### Lifting the foot off the ground

1. Move the time slider to frame 15.
2. In the Left viewport, move the foot off the ground using the Transform gizmo arrow. Here you're using biped's IK system. By moving the foot, you are also creating rotations for the upper and lower leg links.
3. Move the foot forwards by using the Transform gizmo arrow.
4.  Click Set Free Key to keyframe the position of the foot.
5. Rotate the foot slightly down toward the ground.




**Raise the foot then rotate it.**


6. Click Set Free Key to keyframe the rotation of the foot.  
If you add rotations for the toes, be sure to set Free Keys for them as well.

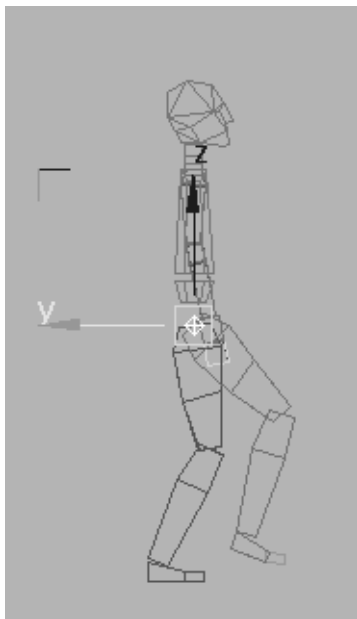
7. Move the time slider back and forth to observe the animation so far.

### Locking down the opposite foot

1. Move the time slider back to frame 0 and select *Bip01 L Foot*.
2. On the IK Key Info rollout, click Set Key.
3. Choose Select Pivot.
4. Select the pivot for the ball of the foot.
5.  Set a planted key for the pivot point.  
The IK Blend changes to 1. Object Space and Join to Prev IK Key are turned on.  
There is now a planted key at frame 0.  
This locks the foot down for any subsequent movement in frames to come.
6. Click Select Pivot to turn it off.




### Keyframing the center of mass

1.  Select the center of mass by choosing Body Horizontal on Track Selection.
2. At frame 0, set a key for the center of mass.  
This will create an initial key for the center of mass.
3. Move the time slider to frame 15.
4. In the Left viewport, use the Transform gizmo to move the center of mass so the torso shifts forwards, and then set a key.




#### Move the center of mass forward.

Since the center of mass is the root node, use Set Key, rather than one of the specialized IK keys.

5. Use the Transform gizmo to move the center of mass downwards so the knee bends, and then set a key.
6. At frame 15, select *Bip01 L Foot*.
7. On the IK Key Info rollout, set a key and choose Select Pivot.
8.  Choose the pivot point at the center of base of the toes, and set a planted key.
9. Turn off Select Pivot.
10.  Rotate the foot so the heel is lifting up off the ground, and set a planted key.
11.  At frame 22, set a key, and turn on Select Pivot.

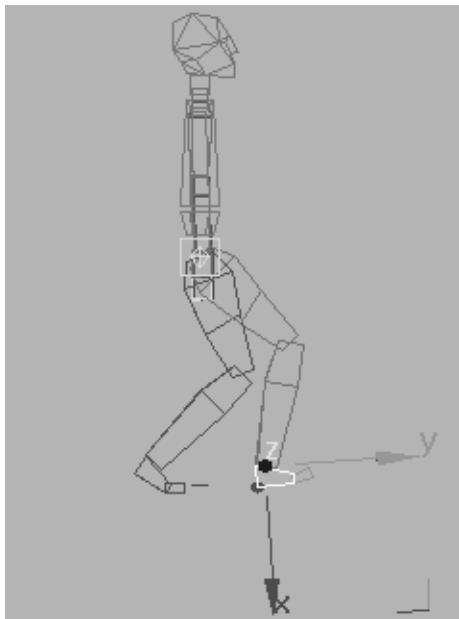
12. In the Perspective viewport, select the pivot at the end of the toes of *Bip01 L Foot*.
13. Set a sliding key.
14. Turn off Select Pivot.
15. Rotate the foot up some more and set a sliding key for the foot.

16.  Move the center of mass forward again, and set a key.

If you move the center of mass down you can change the appearance of your walk. You can put bounce into the biped's step.

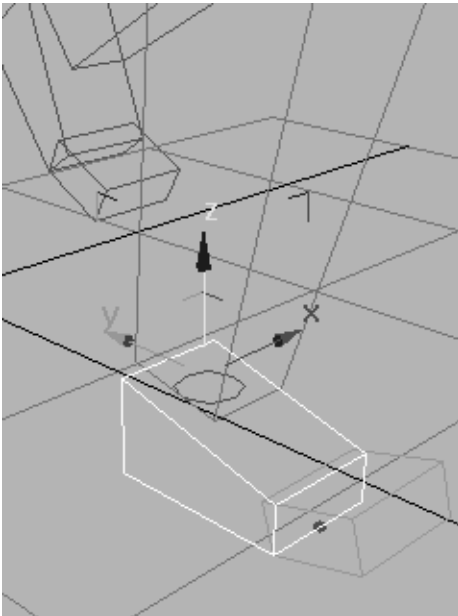
#### Keyframing the heel hitting the floor

1. At frame 22, lower the right foot so the heel is level with the ground and set a sliding key.
2. Select the pivot of the right foot so it's on the heel and set a sliding key.



The heel strikes the ground.

3. Move the foot forwards a bit.
4. Notice the foot moves away from the pivot point in the viewport.
5. Set a sliding key.  
The pivot point in the viewport moves to the heel of the foot.
6. Advance to frame 27 and rotate the right foot so it's flat on the ground.  
Now you can set a planted key.
7. At frame 27, select the pivot on the ball of the right foot.




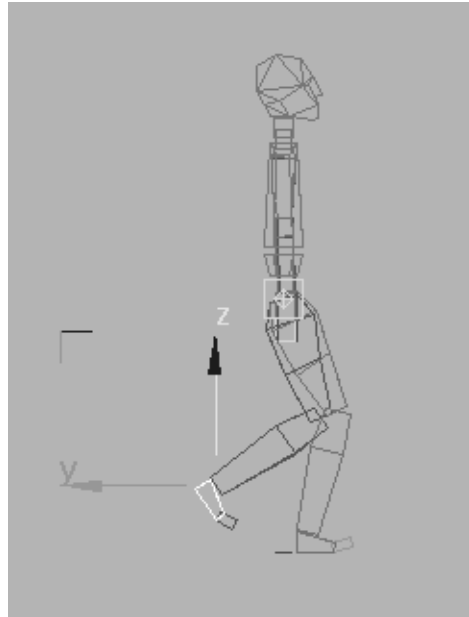
**Animate the pivot to the ball**

8. Move the time slider and see the animation of the foot and the pivot points.

### Keyframing the trailing foot

1. Select the center of mass, move it so it is over the heel of the planted right foot and set a key.


2.  Select the left foot and set a free key.
3. Raise the left foot up off the floor and move it to the right.
4. Rotate the toes down so they look more natural and set a free key for the toes.



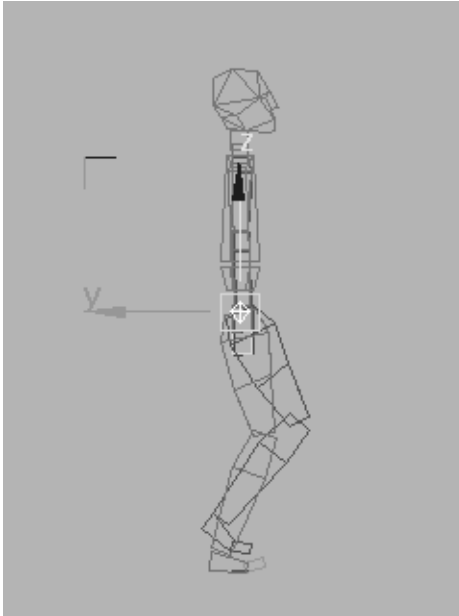
**Lift the trailing foot.**

5. Move the time slider to play the animation.
6. If the foot appears to drag along the floor, go to frame 24, raise the foot off the ground and set a free key.

### Continuing the walk cycle

1. At frame 27, select the center of mass.
2.  Lower the body slightly, so the biped sinks a bit as the right foot flattens onto the floor. Set a key for the center of mass.
3. Move the time slider ahead to frame 32. Move the center of mass so it's over the ball

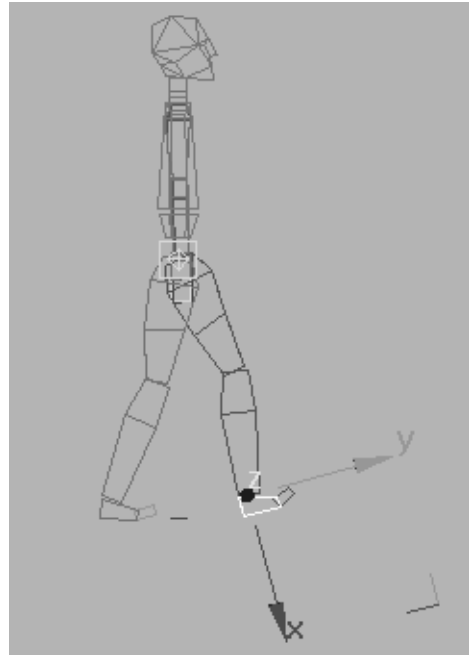
of the right foot. Set a key for the center of mass.



4. Move and rotate the blue foot so the heel swings above the ground. Set a free key. Use this procedure throughout this exercise: lock one foot by setting planted or sliding keys, move the center of mass, then move the other foot and set a key.

### Completing the walk cycle

1. Move the time slider to frame 37, select the center of mass and shift it forward. Set a key.
2. Select the left foot and move it so the leg is fully extended. Set a free key.  
You can rotate the lower leg to make it look more natural.

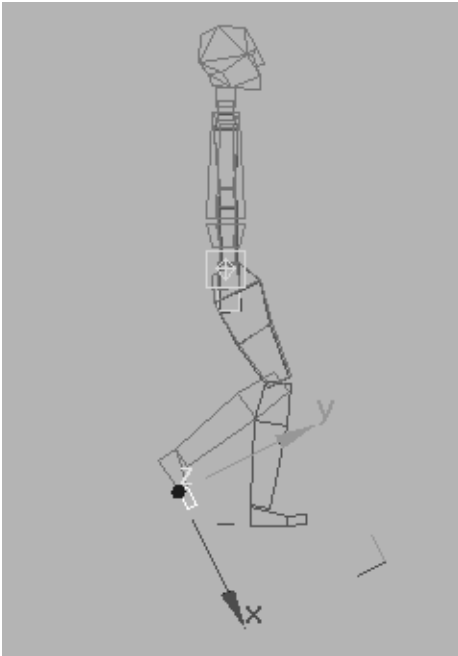


### The Leg extended, but not on the ground

3. Rotate the left foot so the heel is down and the toes point upwards. Set a free key.
4. With the left foot selected, click Select Pivot and select the pivot at the heel. Set a planted key for the pivot.
5. Turn off Select Pivot.  
**Tip:** You can also turn off Select Pivot by clicking Sub-Object.
6. Move to frame 39, and rotate the left foot so it is flat on the ground.
7. Set a planted key for the left foot.
8. Move the center of mass so the body moves forwards.
9. Set a key for the center of mass.
10. At frame 41, rotate the toes so they are flat too, and set a planted key for the toes.





11. For the right foot, be sure there are set keys on the heel pivot at frame 22 and on the ball pivot at frame 27.
12. At frame 32 set a sliding key on the toe pivot.
13. At frame 37, rotate the right foot off the toe pivot and set a planted key.
14. Move the time slider and review the movement. Add rotations for the toes as needed.

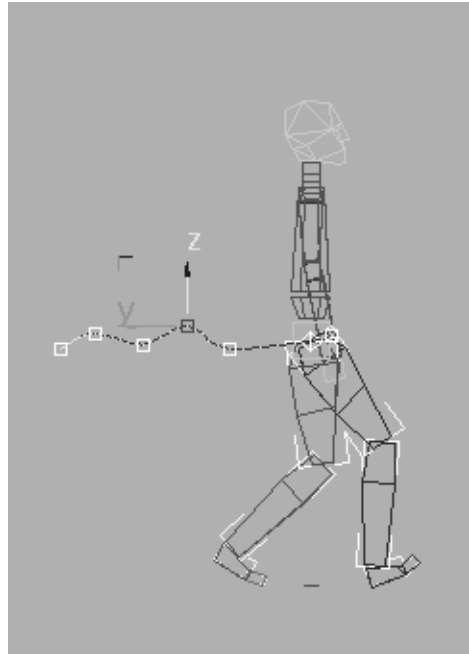


Add Rotations to the toes

### Display Trajectories

Biped has its own trajectory display. You can use it to observe the movement of the center of mass in the walk cycle. You can also edit the keys on the trajectory directly in the viewport.

1.  In the Display rollout turn on Trajectories.
2.  In the Track selection rollout, click Body Horizontal.
3. Scrub the time slider, and see the biped center of mass moving along its trajectories.
4. Choose Select and Move on the Main toolbar, turn on Sub-Object Trajectories, and then click any key on the trajectory.
5. Use the transform gizmo to raise or lower the keys to correct the trajectory.



### Edit keys in Biped Trajectory

**Warning:** Don't use the 3D Studio MAX Trajectories with Biped. Use the Trajectories button in the Biped Display rollout instead.

6. Turn off Biped Trajectories.


### Adding Arm Swings

The character is starting to look like it's walking, but it's still pretty stiff. Adding arm swings will put some life in the animation.

The arms swing opposite to the legs. When the right leg is forwards, the left arm is forwards. Arms bend at the elbow on the forwards swing, and stretch out straight on the backwards swing.

1. Move the time slider to decide where to place the arm swings.  
The right leg stretches out at frame 22, and you'll keyframe the left arm to swing there.



2. Turn on Animate.
3. At frame 0, select and move the left hand slightly, to set a key.
4. At frame 0, select and move the right hand slightly, to set a key.
5. At frame 22, select and move the left hand so it swings forwards.  
Position the arm so there is a slight bend at the elbow. Since Animate is on, you have keyframed the arm by moving it.
6.  On the Track Selection rollout, click Opposite.  
The right hand is selected.
7. Move the right hand so the arm is stretched out.
8. Double-click on *Bip01 R UpperArm*.  
The entire right arm is selected.
9. On the Keyframing rollout, click Copy Posture.
10. At frame 37, click Paste Posture/Pose/Track Opposite.  
The left arm swings behind the body.
11. At frame 22, double-click *Bip01 L UpperArm*.

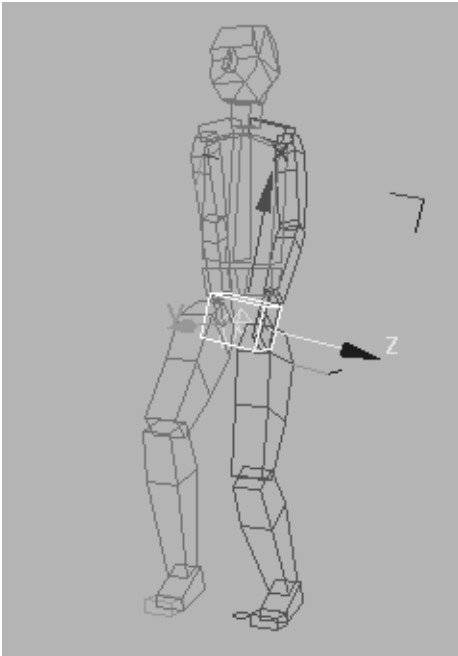
The entire left arm is selected.

12. On the Keyframing rollout, click Copy Posture.
13. At frame 37, click Paste Posture/Pose/Track Opposite.  
The right arm swings in front of the body.
14. Turn off Animate.
15. Move the time slider back and forth to evaluate the animation.

### Adding sway to the shoulders and hips

You've animated the character by moving its hands and feet and center of mass. But the spine, hips and head are still stationary. You'll add some rotations to the shoulders and hips to complete the walk cycle.

1. Select the pelvis and move the time slider to frame 15.  
The left foot is locked at this frame with a planted key.  
Be careful where you add the hip rotations. Don't inadvertently disturb the work you've done on the feet so far.  
As the legs extend and swing forwards, the hips rotate slightly in the direction of the movement.
2. Rotate the pelvis about the Y axis approximately -2 degrees and set a key.



#### Add rotation to the hips.

The pelvis will not accept too much rotation. When you set the key, the pelvis corrects itself to account for the locked foot. You can rotate the pelvis only in Y. You can't rotate it in Z or X.

The right foot is locked down at frame 32, so you'll set a key there as well.

3. Move the time slider to frame 32, and rotate the pelvis a few degrees, and set a key.
4. Move to frame 39 and rotate the pelvis back again and set a key.

The procedure is the same for the spine. At frame 22, the arms swing out in one direction. At frame 37, they swing in the opposite direction.

5. Select the biped spine object (*Bip01 Spine*).

6. At frame 22, rotate the spine in the direction of the arm swing and set a key.
7. At frame 37, rotate the spine in the opposite direction and set a key.

The spine can freely rotate about all three axes. You can make adjustments on each one. Rotate about Z for a more stooped walk. Increase rotation about X to make the walk loose and floppy.

You can also animate the clavicles to raise or lower the shoulders instead of animating the spine.

You have animated a simple walk cycle using totally freeform animation and IK constraints.

You can use the techniques found in the *swimmer tutorial* (see page 447) to refine and finish the animation motions. Be sure to do that tutorial as well, if you've skipped it.

You can use the footstep method of animation, to create a walk cycle automatically. To learn about this, see Tutorial 3 *Animating a Walk Cycle with Footsteps* (see page 469).



---

## Tutorial 3

# Footstep Animation

---

## Animating a Biped with Footsteps

Footstep mode uses a unique footstep gizmo to control the contact of the foot and the ground. When you move the footstep to a new location, the animation updates to match the move.

In this tutorial you will do the following:

- Animate a biped using footsteps
- Make a biped walk, run, jump, and do flips.
- Make a biped take a pratfall
- Add a freeform period to a footstep animation
- Convert a footstep animation to freeform
- Change the duration of a footstep animation using IK keys

## Lessons

*Lesson 1: Creating an Expressive Walk (see page 470)*

*Lesson 2: Modifying Footsteps (see page 477)*

*Lesson 3: Creating Gymnastic Motion Flips with Ballistic Tension (see page 481)*

*Lesson 4: Animating a Pratfall (see page 494)*

*Lesson 5: Changing Footsteps using IK Keys (see page 500)*

## Lesson 1: Creating an Expressive Walk

In this tutorial, you'll use Biped to animate a character walking, using a default motion that is created automatically.

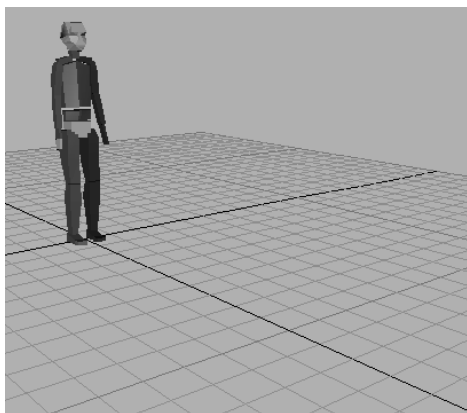
This sophisticated yet simple approach results in a natural-looking motion that is quick to create. You'll refine the motion, changing the default walk into something more expressive.

### Setup

1. Reset 3D Studio MAX.
2. Open *cs3\_tut03\_start\_walk.max*.

The files for this tutorial are in the *cstudio\tutorials\tutorial\_3* directory in your 3DS MAX path. If the files are not installed on your system, you can load them directly from the Tutorials directory of the **character studio 3** CD.


A biped is standing at the origin.



Biped at center of grid

3. Maximize the Perspective viewport by pressing **W**.
4. Click any part of the biped.


A white box outlines the body part you clicked, to show it's selected.

5.  Open the Motion panel.

The Biped controls are displayed. Figure mode is active.

Next you'll turn on Footstep mode. Figure mode turns off automatically.

### Using Footstep Mode

1.  In the General rollout of the Motion panel, turn on Footstep mode. The Footstep mode button turns yellow and the Footstep Creation and Operations rollouts are displayed.

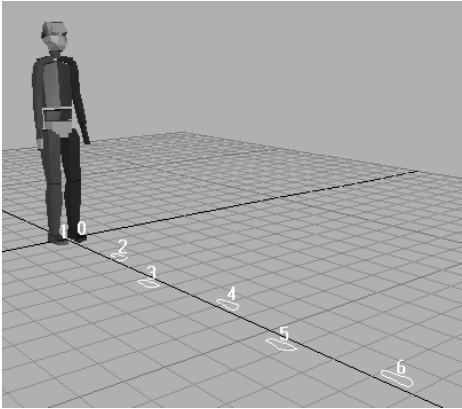
2.  In the Footstep Creation rollout, click Create Multiple Footsteps.

The Create Multiple Footsteps: Walk dialog is displayed.


You'll just change the number of footsteps, leaving the other defaults as they are.

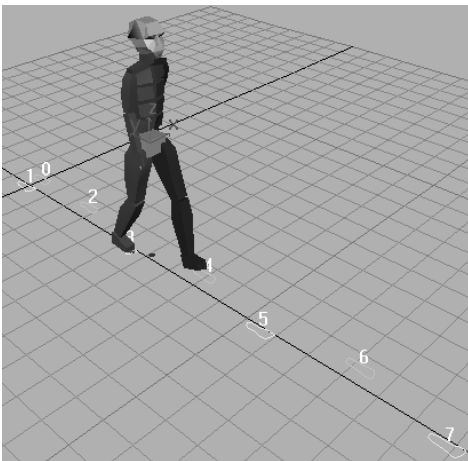
3. In the Create Multiple Footsteps: Walk dialog, change Number of Footsteps to 8 and click OK.

Footprints are displayed in white in the viewport. These are inactive footsteps. They do not yet control any animation for the biped.



### Footsteps

4.  In the Footstep Operations rollout, click Create Keys for Inactive Footsteps. The Footsteps are activated. Animation keys are created for the biped.
5. Play the animation. The biped walks.



### Biped walks.

6. Turn off Footstep mode.


The track bar displays keys for the length of the animation.



### Track bar displays keys.

The biped walks, but without much character. In the next steps, you'll begin individualizing the motion by bending the biped's spine back and forth over the keyed animation frames.

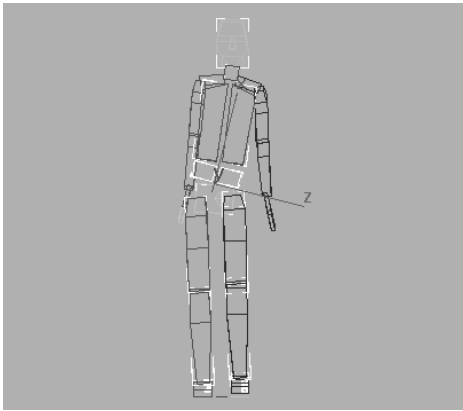
### Adding spine motions

1. In the Front viewport, click one of the biped's spine objects.
2. Open the Motion panel.
3.  On the General rollout, click Bend Links mode. Bend Links mode lets you bend all the spine elements when the first spine link is rotated.

Note: This figure only has two spine elements. The effect is more pronounced on a biped with more spine objects.




4. Turn on Animate and move the time slider to frame 24.
5. Rotate the spine approximately -12 degrees about the Y axis to move the hips down towards the leg that is in motion.



Rotate the spine.

6.  Click the Key Mode button to turn on Key mode.

Key mode lets you use Previous and Next Key buttons to jump between keyframes for the selected object.

7.  Click Next Key to move to frame 40.
8. Rotate the spine 12 degrees about the Y axis.
9. Move to frame 54 and rotate the spine -10 degrees about the Y axis.


Repeat this pattern until you have finished rotating the spine at the end of the animation. Do not make your adjustments too precise. Slight variations from frame to frame make the motion look more natural.

When you are done, play the animation and notice the increased hip swings that result from bending the spine back and forth.

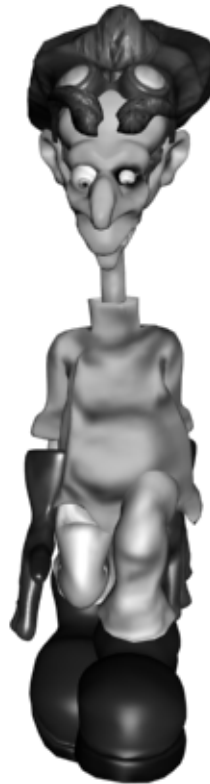
10. Turn off Bend Links modes



11. Turn off Animate.

12.  On the General rollout, click Save File and save the file as *mywalk.bip*.

If you load the newly saved *mywalk.bip* file into a scene containing a skinned character, the character will swing its hips according to the instructions you saved in the file. You may need to adjust it, play the animation to determine if it is necessary.



Animation applied to Dr. X



If you made a mistake during the previous steps: use Load File on the General rollout to load the file *cs3walk1.bip*. Otherwise, just keep working with the file that's open.

Next, you accentuate the biped's hip motion by rotating the pelvis.

### Adding hip motions



1. Turn on Animate, and move the time slider to frame 24.



2. On the Front viewport, use Rotate to select the biped's pelvis.
3. Rotate the pelvis -8 degrees about the Y axis, moving the left leg down.



4. Make sure Key mode is still on, and then go to frame 40.

**Tip:** When Key mode is on, you can use the time slider arrows to jump to the next key.

5. Rotate the pelvis 8 degrees about the Y axis, moving the right leg down.

Continue this pattern until you have finished rotating the pelvis to the end of the animation.



6. Turn off Animate.
7. Play the animation to see the increased hip motion.



Added pelvis rotation

8. Save your file as *mywalk1.bip*.

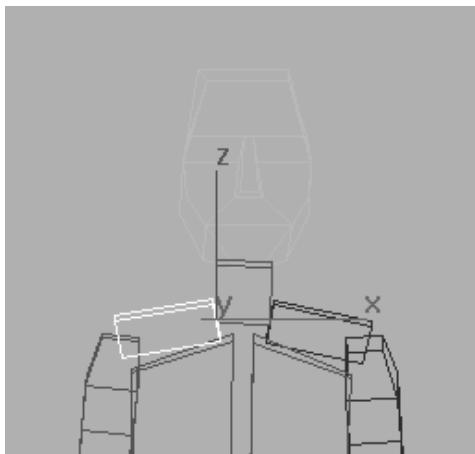
At this point, you can either keep working with the current file or load *cs3walk2.bip* to compare it with what you have done.

### Adding hand motions

Arm and hand motions are an integral part of an individual's movement when it walks. In the following sequence, you'll move the biped's hands at its wrists using Track View and then add irregular motion to each frame to make the movement more life-like.

The keys for the hands, forearms and upper arms are all stored in the Clavicle tracks. You'll use Track View in the next sequence.

1. In the Front viewport select *Bip01 R Clavicle*.



**The right clavicle**

2. Right-click the selected object and choose Track View Selected.
3. Expand the transform track.
4. On the Motion panel, in Track Selection, click Symmetrical.  
This adds the left clavicle to the selection set.
5. Expand the transform track for the left clavicle.



If you can't see all the frames in the animation in the Track View Edit Window, click Zoom Horizontal Extents.

6. Select all the left and right clavicle keys by drawing a selection region around them in the Track View Edit window.

The selected keys turn white.

7. Minimize Track View.



8. On the Keyframing rollout, click Set Multiple Keys to display the Biped Multiple Keys dialog.

Note: Use this dialog to apply a transformation equally to all keys. The change is relative to the original position and any differences between keys are maintained. It operates on existing keys and does not create new keys. Therefore, you do not have to be in Animate mode or use the biped Set Key buttons to use it.

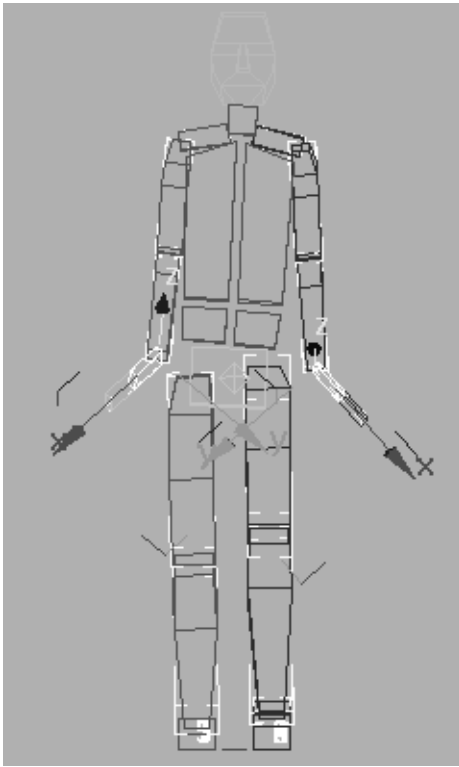


9. On the 3DS MAX main toolbar click Rotate, and select the biped's right hand.



10. On the Motion panel, on the Track Selection rollout, click Symmetrical Tracks to add the left hand to the selection.

11. In the Front viewport, rotate the hands 30 degrees about the Z axis to tilt them away from the body.



**Rotate hands away from body.**

12. On the Change Multiple Keys dialog, click Apply Increment to apply the rotation to all the hand keys.
13. Close the Change Multiple Keys dialog and Track View.
14. Play the animation.

The constant angles of the hands give them a mechanical look. To rectify this you'll adjust each frame in which a hand key is set.

### Adjusting the hand positions

You'll be repeating the procedures again here. There's plenty of repetition in animation work. This should reinforce your learning.

1. Move the time slider to frame 0.



2. Use Rotate to select one of the hands, and click Symmetrical Tracks (on the Track Selection rollout) to add the other hand to the selection.



3. Turn on Animate, make sure Key mode is turned on, and lock the selection.
4. Jump to the next key.
5. Rotate the hands approximately 10 degrees about the Z axis.
6. Jump to each key and rotate the hands slightly at each frame until you've adjusted the last frame.

Note: You can also select the hands separately and adjust each one differently.

7. At this point, you can either save your *mywalk.bip* file or go back and make further adjustments before going on.

**Tip:** For longer animations, Use Change Multiple Keys to add irregularities. Select every other key or select a random series of keys instead of selecting them all.

8. Play the Animation in the Perspective viewport.

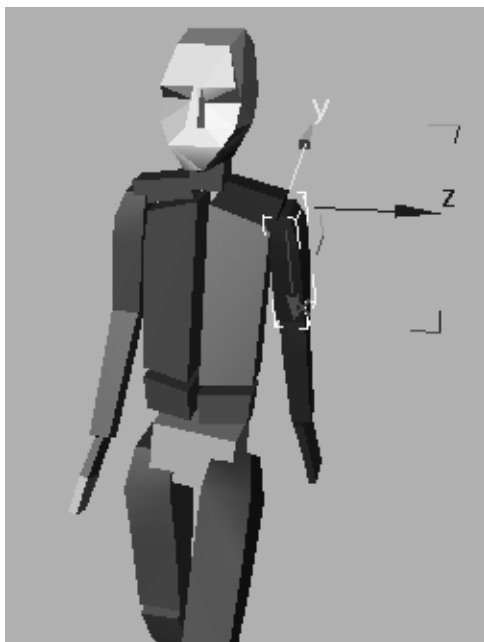
### Keeping the arms from passing through the body

The upper arms might pass through the biped chest (depending on how you rotated the pelvis). You can fix this with a few additional rotation keys.


 Animate

1. Turn on Animate, if it isn't already.
2. Select and rotate the *Bip01 L UpperArm* away from the body (about the Y axis) at the frames where the arms pass through the body. (approximately frames 39, 68, and 99).

**Tip:** You can enter the frame numbers in the Current Frame field in the Time Controls to move directly to the frame you need.






Rotate the arm away from the body.

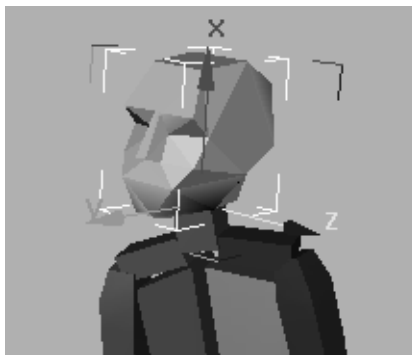
3. Repeat step 1 for the right upper arm. Add rotations at frames 25, 55, 86, and 115.

### Adding head motions

You can edit the biped's head motion to make the character's motion look natural.


 Animate

1. Turn on Animate, if it isn't already.
2.  Turn on Key mode, if it isn't already.
3. Move the time slider to frame 0.
4.  In the Front viewport, select the biped's head using Rotate.
5.  Click Next Key twice to move to frame 40.
6. Rotate the head -20 degrees about the X axis to turn the head to the biped's left, and -8 degrees about the Z axis to make the biped's head turn left and up.
7. Continue to jump through the head's keys, setting rotations to animate the head. Natural head motion is smooth, so the orientations should change gradually from one key to the next.



Rotation of head at frame 69

8. Turn off Animate and Key Mode.
9. Play the animation and notice how much the biped's side-to-side and up and down head movements add to the animation. You can now save your work as *mywalk2.bip*. You can check your file against *cs3walk3.bip*.





## Lesson 2: Modifying Footsteps

The footsteps you've worked with so far gave the biped a default walk. In this lesson, you'll learn how to copy and paste biped motion to extend an animation.

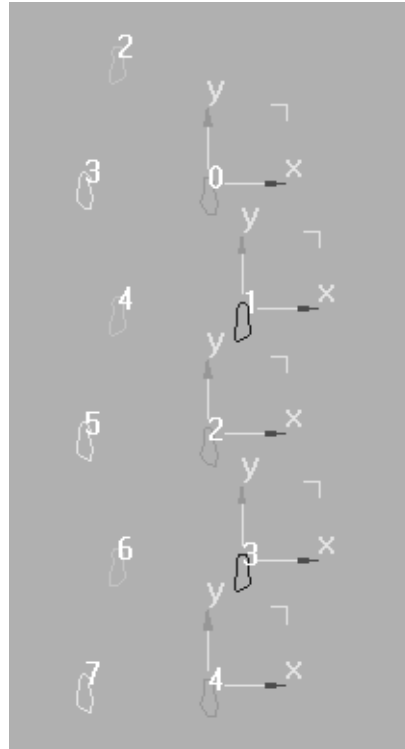
### Setup

- Continue from the previous exercise or open *cs3\_tut03\_start\_run.max*.


### Extending the walk

- Select the biped.
-  On the Motion panel, on the General rollout, turn on Footstep Mode. The Footsteps sub-object level is activated, and only the footsteps can be selected.
-  Right-click to activate the Top viewport, then click the Min/Max button to maximize the viewport.
- Using Select and Move, region-select footsteps three through seven.
-  On the Footstep Operations rollout, click Copy Footsteps to place the selected footsteps into the footstep buffer.
-  Click Paste Footsteps to paste them in the viewport.
- The new footsteps appear next to the biped's current footsteps.

**Tip:** If you have Transform gizmo on, use the minus key (-) to shrink the Transform gizmo, so it doesn't cover up the footsteps.




### Size the Transform gizmo

- Move the new footsteps (they move as a set) so the first footstep of the new set is over footstep 7 of the original set. When footstep 7 of the original set turns red, release the mouse button.  
Footsteps from the original motion are inserted. Now there are 11 footsteps visible.
-  Press W to display four viewports.
- To display the entire animation in the Perspective viewport, zoom out and use Arc Rotate and Pan to adjust your view until the biped and all 11 steps are visible.

11. With the Perspective viewport active, play the animation.

Since you are still in Footstep mode, the Motion panel is available. This is a good time to save your *mywalk.bip* file.

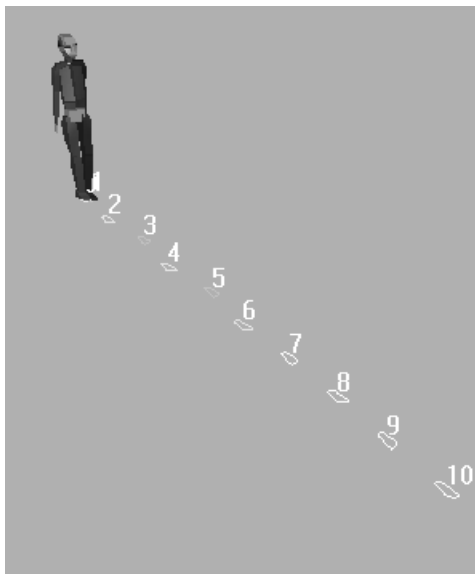
### Scaling the walk

1.  Make sure Footstep mode is active.
2. In the Top viewport, region-select all the footsteps.
3. On the Footstep Operations rollout, clear Length, and leave Width selected.
4. Set Scale = 2.0 to double the spacing between the left and right footsteps.
5. Play the animation.
6. Set Scale to 0.25 to reduce the spacing between the left and right footsteps to half of the original scaling (one-quarter the current setting).

If you hadn't previously doubled this parameter, a setting of 0.5 would have scaled the width by 50%.

Now the biped puts one foot in front of the next.

Use Scale Width and Length to adjust the footsteps if your characters have big feet.

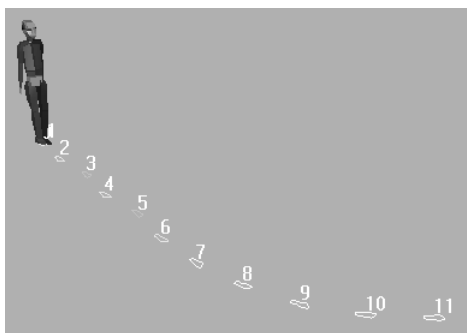


**Scale the width between the steps.**

7. Play the animation.

### Bending the walk

1. In the Top viewport, select all the footsteps from 7 on.
2. On the Footstep Operations rollout, set Bend to 11.0 to bend the footsteps to the left, beginning at footstep 7.
3. Play the animation.

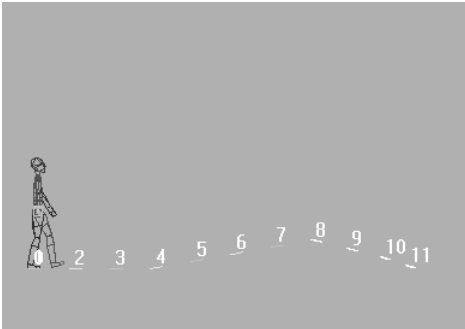


**Bend selected footsteps**

### Walking on uneven terrain

You can raise and rotate the footsteps footprints to create the illusion of walking on uneven terrain.

1. Make sure that Footstep mode is still on.
2. Maximize the Perspective viewport.
3. Use Select and rotate to select all the footsteps from 4 on.
4. Use the transform gizmo arrows to rotate the selected footsteps approximately -8 degrees about the X axis so the footsteps go up a hill.
5. Select footsteps 8 through 11.
6. Rotate the selected footsteps about the X axis approximately 21 degrees, so that the footsteps go back down the hill.
7. Play the animation.



### Walking on uneven terrain by raising and rotating footsteps

You can also create footsteps on uneven terrain by turning on AutoGrid and then setting the footsteps one by one using Create Footsteps (at current frame). You can use mouse and keyboard shortcuts to move over the terrain as you place the footsteps.

### Adding a jump

If there is a period of time during a footstep animation when neither foot is on the ground, it will be interpreted as a jump. There are several different ways to create a jumping animation. In this lesson you'll move footstep keys in Track View to make the jump.

#### Setup

- Open *cs3\_tut03\_start\_jump.max*.

This is a slightly longer version of the same file you've been working on. It has 15 footsteps instead of 11.

#### Moving Footstep Keys in Track View

1. Select *Bip01* and right-click, then choose TrackView selected.
2. Make sure Animated Tracks Only are showing.
3. Right-click on *Bip01* and choose Expand Tracks.
4. The *Bip01 Footsteps* track is displayed in Track View.

**Tip:** If *Bip01 Footsteps* track is not displayed, turn on Footstep mode in the Motion panel and the track will appear.

5. Scroll until you see the footstep track, (the blue and green blocks next to *Bip 01 Footsteps*).

|   |    |    |    |     |     |     |     |     |  |
|---|----|----|----|-----|-----|-----|-----|-----|--|
| 0 | 17 | 31 | 61 | 91  | 121 | 151 | 181 | 211 |  |
| 0 | 0  | 2  | 4  | 6   | 8   | 10  | 12  | 14  |  |
| 0 | 33 | 46 | 76 | 106 | 136 | 166 | 196 | 226 |  |
| 1 | 1  | 3  | 5  | 7   | 9   | 11  | 13  | 15  |  |

#### Footsteps in Track View

Each blue block represents a left footstep and each green block represents a right footstep. The length of the blocks is the period of time the foot is in contact with the ground during the footstep. The spaces between the blue and green blocks represent periods in which the biped is not supported

by the left or right foot. Make sure you can see some other biped tracks as well.

6. Select footsteps 11 through 15 by drawing a box around them in Track View.

Notice that footstep number 11 starts at frame 166.

7. Click in the center of footstep 11 and drag it to the right until the number 166 (indicating the first frame of footstep 11) increments to number 180. Release the mouse button.

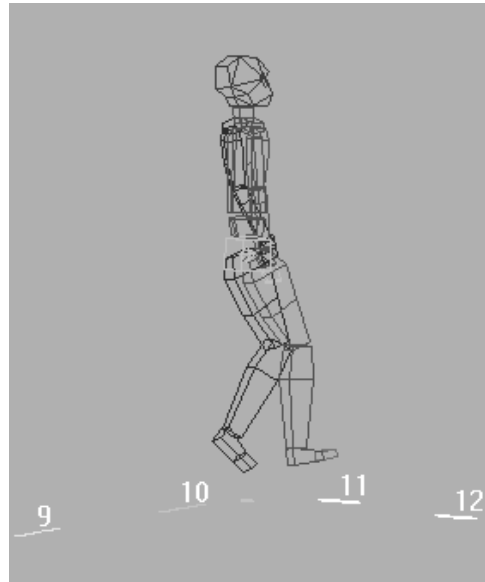
|   |    |    |    |     |     |     |     |     |
|---|----|----|----|-----|-----|-----|-----|-----|
| 0 | 17 | 31 | 61 | 91  | 121 | 151 | 195 | 225 |
| 0 |    | 2  | 4  | 6   | 8   | 10  | 120 | 140 |
| 0 | 33 | 46 | 76 | 106 | 136 | 180 | 210 | 240 |
| 1 |    | 3  | 5  | 7   | 9   | 110 | 130 | 150 |

**Shift the keys to the right to create a gap.**

The keys in the other biped tracks adjust to the change in the footstep track.

You have changed a walking step into a jumping step by creating an area in the footstep track where neither foot is supporting the biped.


8. Close Track View and play the animation.



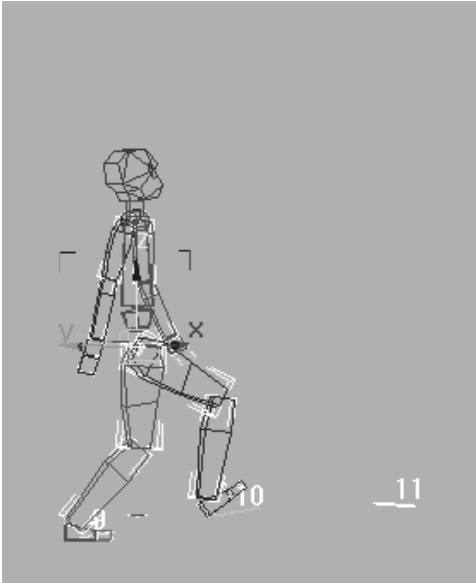
**The gap between footsteps creates a jump.**

### Crouching before the jump


The preparation for the jump, between footsteps 9 and 10, looks a little stiff because the biped is not crouching enough before jumping. Resetting a vertical key will fix this.

1.  On the Track Selection rollout, click Body Vertical.
2. Scroll to frame 153, where there is a Body Vertical track key.
3. Select and move the center of mass down approximately -5 units.






Lower the vertical track.

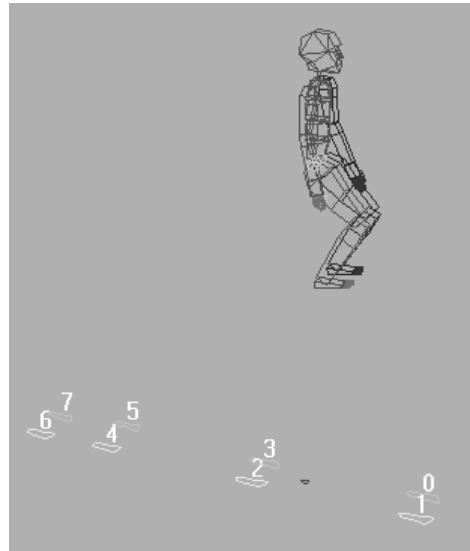
4.  On the Keyframing rollout, click Set Key to set the vertical track key.
5. Move the time slider to view the animation.
6. Select *Bip01 R Foot*. Move the time slider to frame 167 and raise the foot so it's above the ground (approximately 4 units). Set a key to hold it there. Set additional keys on the foot if it swings through the ground.
7. Play back the animation.  
The jump looks smoother. The result should be similar to *cs3walk9.bip*.

## Lesson 3: Gymnastic Motion Flips with Ballistic Tension

The basic jump has already been set up for you. It is a basic eight-footstep backwards jump produced by the Create Multiple Footsteps dialog. The last pair of steps have been moved in to reduce the jump height of the final hop.

### Loading the motion file

1. Open *cs3\_tut03\_flip1.max*.
2. Select the biped.
3.  Go to the Front viewport and play the animation using Biped playback.



### The basic jump

Notice that the upper body appears mechanical, although the biped leans slightly into the jump and bends its knees in anticipation of the jump.




You will add a flip to the motion.

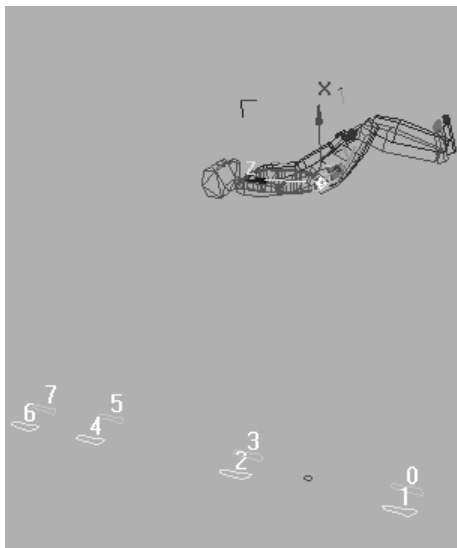
### Adding a flip

Adding a flip involves setting rotation keys for the body by rotating the center of mass object (*Bip01*) while the body is in the air. If the feet are not touching the ground, rotating the center of mass rotates the entire body. If the feet are planted on the ground, rotating the center of mass rotates only the upper body. The feet remain on the ground.


You want the biped to rotate 360 degrees during the flip. You must define at least two turning keyframes in the air to make the biped flip all the way around in the same direction.

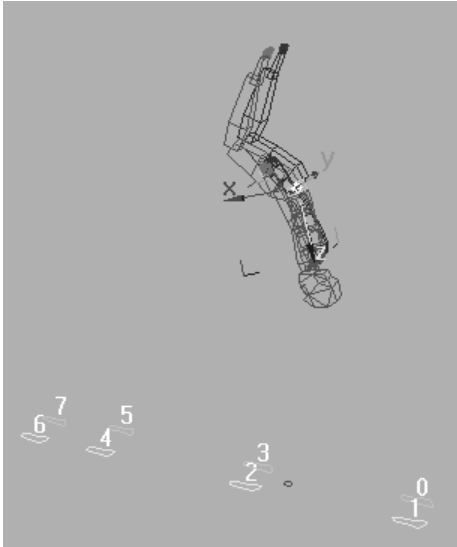
### Producing the first flip


1.  Click Body Rotation on the Track Selection rollout to select the center of mass rotation track.
2.  Click Lock Selection Set.  
When working with objects like the center of mass that are close to other objects, lock your selection to avoid accidentally selecting another object. This is especially true if you will be switching viewports or zooming and panning.
3. Move the time slider to frame 31.
4.  Right-click Select and Rotate.  
The Transform Type-in dialog is displayed. This lets you enter precisely the transform you wish.
5. In the Offset Local area enter a Local rotation of -94 degrees in the Y axis field.  
The biped rotates onto its back.





**Use Transform Type-in to rotate the biped.**

6.  On the KeyInfo rollout, click Set Key to set the turning key.
7. Move the time slider to frame 42.  
The biped rotates back as a result of continuity with other keys. The next step will fix the rotation.
8. Rotate the biped -196 degrees about the Local Y axis.  
The biped is upside down with its feet about to come back down.






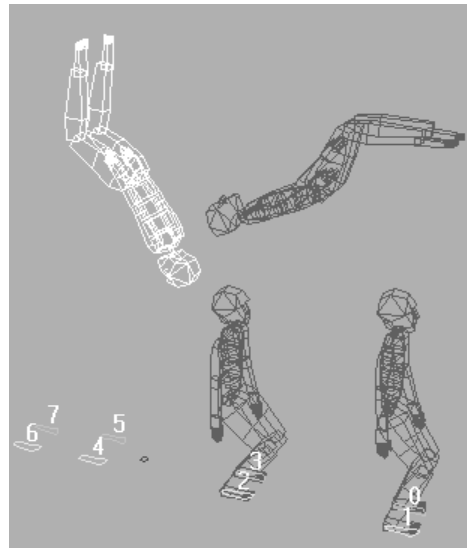
9.  Click Set Key.  
That completes the first flip.
10. Close the Transform Type-In dialog.  
You can use the body keys you just created to produce the second flip. Since the second flip is similar to the first, you can copy body positions from the first flip to equivalent frames in the second flip.

### Producing the second flip


1. Move the time slider to frame 31.
2.  Click Copy Posture on the Keyframing rollout to copy the body rotation to a buffer.
3. Move the time slider to frame 74.
4.  Click Paste Posture to replace the body rotation at this frame with the rotation in the posture buffer.

The biped body rotates into the same position at frame 74 and at frame 31.

5.  Click Set Key.  
Do the same for the rest of the flip frames.
6. At frame 42, click Copy Posture.
7. At frame 85, click Paste Posture.
8.  Click Set Key.
9.  Move the time slider to frame 0, and play the animation.  
Now you have two complete flips in the jump sequence.



Two flips (*shown using ghosting*)

10. Unlock the selection set.
11.  Save your motion data by clicking Save File on the General rollout.
12. Save the file as *myflip2.bip*.

### Verifying your work


1. Click Load File to load another biped motion file, then choose `cstudio\tutorials\tutorial_3\flip2.bip`. This motion file contains the same movements that you created in the last procedure. You can use it to check your rotations.
2. Play this version to verify that your motion sequence is accurate.

You can continue the tutorial with this prepared motion file to insure that you are starting from a known point. If your motion sequence is correct and you would rather use your own file, you can reload *myflip2.bip* and proceed with the tutorial.

In addition to setting keys for the body in the air, we must also alter the rotational keys of the body about its center of mass on the ground to create the impression that the body is preparing for the back flip by whipping itself backwards just before lift-off. Because the feet are planted on the ground, rotating the center of mass rotates only the upper body.

The whipping action will also involve the spine and arms, but you'll begin by setting the body's rotation keys about the center of mass.

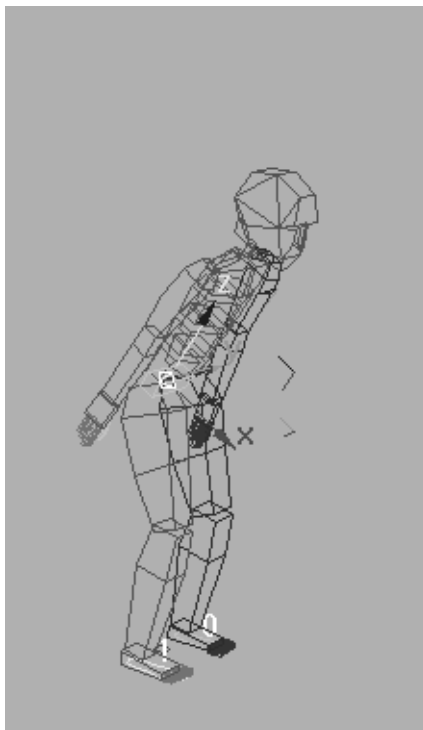
### Creating the basic windup (or angular momentum) for the flip

1. Select the center of mass.
2. Move the time slider to frame 0.
3.  Click Body Rotation Track on the Track Selection rollout.
4. This turns on Select and Rotate in the Main toolbar.



5. Right-click Select and Rotate.
6. In the Rotate Transform Type-In dialog, in the Offset Local group, specify 35 degrees as the rotation for the Y axis.

The biped bends forward. Because the feet are planted on the ground, only the upper body is affected.



**Only the upper body rotates**



7. Click Set Key.  
Don't forget to set the key. If you're used to animating in 3D Studio MAX you can turn on Animate instead to set rotation keys automatically.

Next, you will add rotation keys to the center of mass at five different frames.

**Tip:** You can also use the transform manipulator to select and rotate about a chosen axis. Watch the coordinate display at the bottom as you rotate in the viewport. For entering exact values however, Type-in Transform is faster and precise.

8. Play the animation.



9. Save your motion data by clicking Save File on the General rollout.

10. Name the file *myflip3.bip*, and click Save.

### Verifying your work



- Click Load File to load another motion file, choose *flip3.bip* and click Load. This motion file contains the same movements that you just created in this procedure. You can use it to check your changes. Play this version to verify that your motion sequence is accurate.

## Adjusting the Body Motions

You have made the biped flip. The next steps are designed to improve the quality of the movement and make it look more natural. You do this by working on one body track at a time.

### Adjusting the first flip



1. Click Select By Name, select all the biped spine objects from the selection list, then lock the selection.



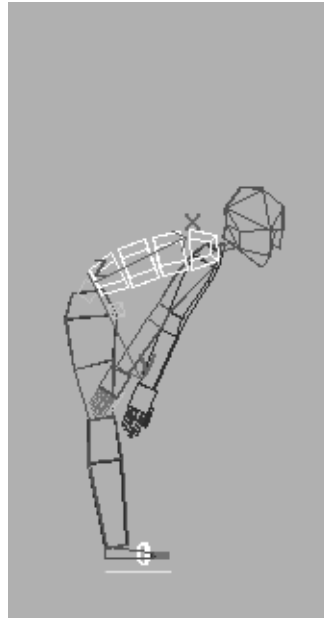
2. Click Rotate.



3. On the General rollout, turn on Bend Links mode.

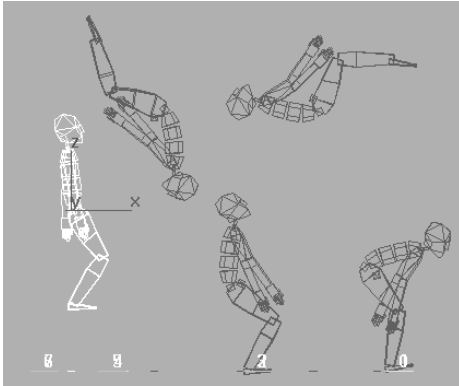
Animate

4. Turn on Animate and go to frame 0.
5. Rotate the spine object 30 degrees about the Z axis to bend the biped forward.



Frame 0

6. Move to frame 20 and rotate the spine -30 degrees about the Z axis. The status of both feet is displayed as Lift in the General rollout.
7. Move to frame 42 and rotate the spine 50 degrees about the Z axis.

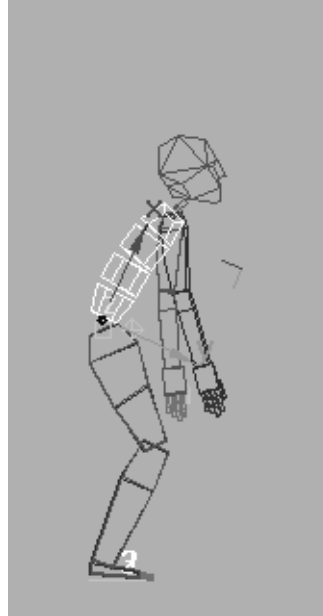


**Frame 42**

Notice that the biped's back tucks in. This alters the biped's position during the first flip. Next, you'll alter its position during the second flip.

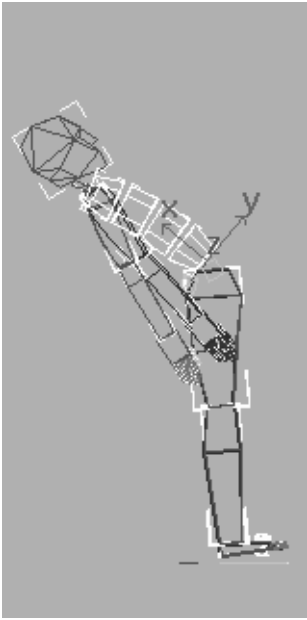
### Adjusting the second flip

1. Move to frame 53 and rotate the spine 20 degrees about the Z axis.  
This is when the biped lands after the first flip. The status of both feet is Touch.






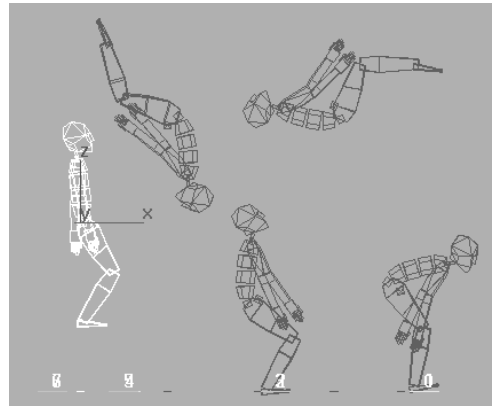
**Frame 53**

2. Go to frame 63, and rotate the spine -20 degrees about the Z axis, so the biped's spine is thrown back in preparation for the next flip.  
At frame 63, the status of both feet is Lift.




Frame 63

3.  At frame 42, click Copy Posture to copy the position of the entire spine at this frame. Note: Copy Posture copies selected biped objects, Copy Pose copies the entire biped.
4.  Go to frame 85 and click Paste Posture. The biped changes from a straight posture to a tucked posture.
5.  Turn off Bend Links mode.
6. Go to frame 0 and play the animation.




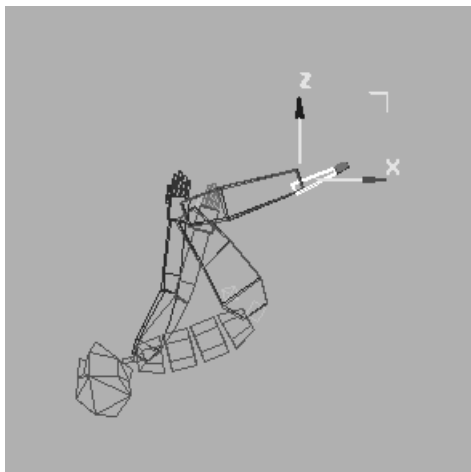
### Upper Body bending added

Notice that the biped's upper body now bends as though it was being used to power the motion.

7.  Save your motion data by clicking Save File on the General rollout and save as *myflip4.bip*.

### Adjusting the legs

1. Unlock the selection and move to frame 37.
2. Be sure Animate is still on.
3. Right-click in the viewport and choose Rotate.
4. Select a biped foot.
5.  On the Track Selection rollout, click Symmetrical Tracks to select both feet.
6. Rotate the feet about the Z axis approximately -50 degrees, so the toes point up a little more.
7. Right-click and select Move. Use the Transform gizmo corners to move the feet so the knees move between the hands.




Move the feet to bend the knees.

8. At frame 80, the equivalent frame in the second jump, move the feet again so the knees tuck and the pose compresses. Use the file *flip4.bip* for reference and backup.

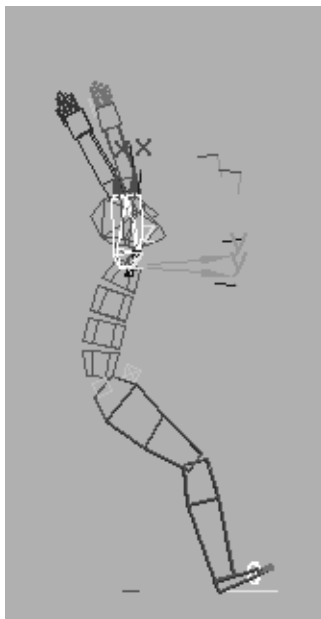
Until now, the biped's arms have been relatively immobile. In a real flip, the arms would begin the flip. In general, arm movement is timed to the spine movement. For the first jump, the arms begin moving at frame 0 and finish moving at frame 20. They accelerate through that period.


### Rotating the arms



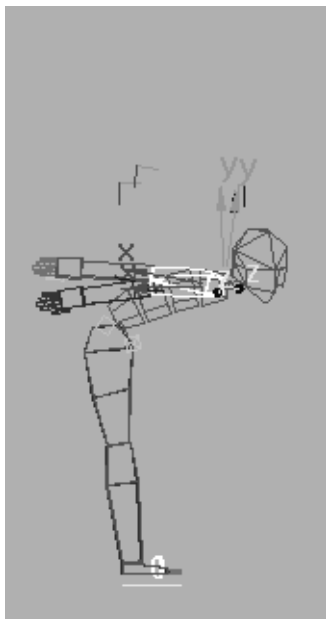
1. Make sure Animate is still on.
2. Right-click and choose Rotate.
3. Select an upper arm (*Bip01 R UpperArm*).
4.  On the Track Selection rollout, click Symmetrical Tracks to select both arms.
5. At frame 20, rotate the arms -150 degrees about the Z axis.

This makes the arms nearly vertical.



6.  At frame 10, the midpoint of the rotation, click Set Key to lock the rotation.
7. Move the time slider to frame 0 and rotate the arms 60 degrees about the Z axis. This places the arms just behind the back.







8. Play the animation.

**Tip:** Turn off Real Time Playback on the Time Configuration dialog to see every frame during playback.

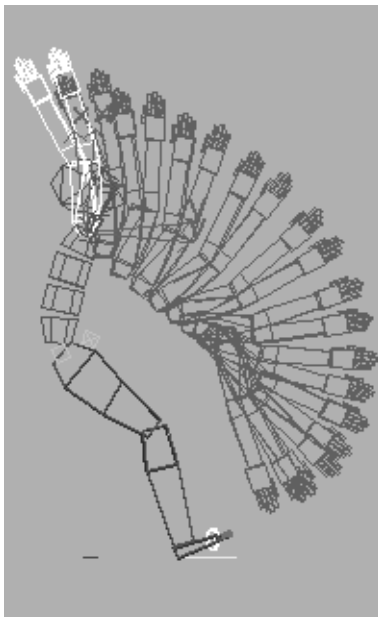
### Adjusting the arm swing

At this point, the motion is fine, but the arms should accelerate more. To do this, you'll copy the arm posture from the first third of the motion onto the arms in the middle of the time sequence. The result will be slower motion in the first half of the sequence and faster motion in the second half.

1. At frame 6, double-click an upper arm to select the entire arm.
2.  Click Symmetrical Tracks to select both arms.
3.  Click Copy Posture.



4. At frame 10, click Paste Posture.
5. Play the animation.



### Swing the arms faster.

The arms now accelerate as they move up. At the beginning of the second flip, the biped is already traveling faster than in the first flip and there is less need for arm movement.

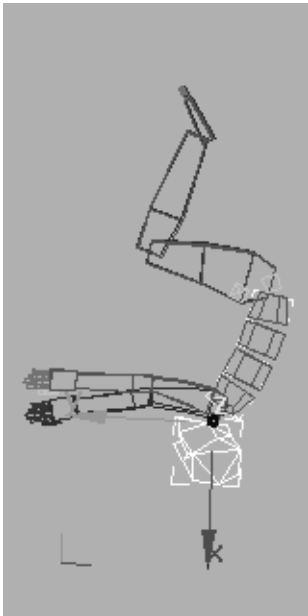


6. At frame 53, click Set Key to lock this arm posture.
7. Deselect the entire arms.
8. Select an upper arm object.
9. Click Symmetrical tracks.
10. At frame 63, the lift-off point, right-click in the viewport, and choose Rotate.
11. Rotate the arms -150 degrees about the Z axis so the arms raise up.

### Moving the head

In general, the head will bend with the spine as it tucks, but will be more vertical when the feet are planted.

1. At frame 20, make sure that Animate is still on.
2. Select and rotate the head -40 degrees about the Z axis.  
As a result of spine and center of mass rotations, the head is thrown back too far at frame 42.
3. At frame 42, rotate the head 40 degrees about the Z axis.

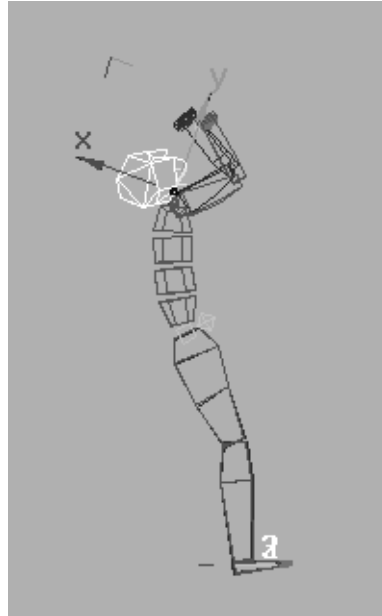


**Biped head rotation at frame 42**

4. At frame 53, rotate the head -12 degrees about the Z axis.  
Frame 53 is the touch down point, where the biped would be looking up from the tuck.

Frame 63 is where the head would be thrown back as part of the jump.

5. At frame 63, rotate the head -40 degrees about the Z axis.

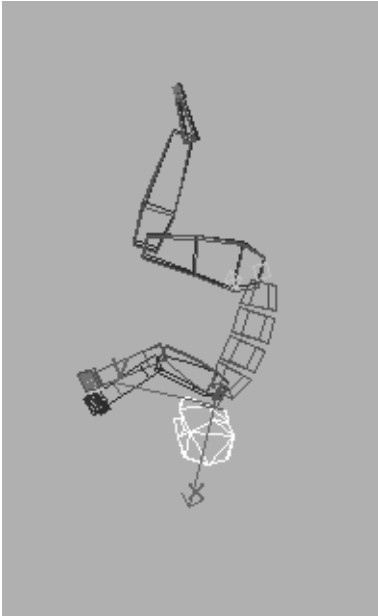


**Head thrown back**

The biped is looking straight up at the sky.

6. At frame 71, rotate the head 15 degrees about the Z axis.
7. At frame 85, the same point in the flip as frame 42, rotate the head 40 degrees about the Z axis.

This tucks the chin into the chest between the arms.



Chin tucked in




8. Turn off Animate, go to frame 0, and play the animation.
9. Save your motion data as *myflip5.bip*.  
You can load *flip5.bip* to check your changes.

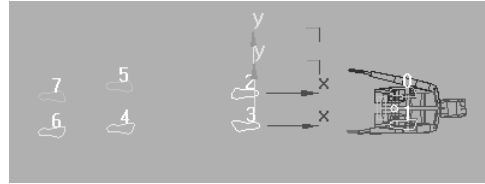
## Adding the Twist

Now that you have a good-looking jump and flip, it's time to add a twist. The easiest way to accomplish this is to rotate selected footprints and let the biped automatically adapt.

### Twisting the footsteps

1.  Turn on Footstep mode on the General rollout.
2. Region-select footsteps 2 and 3.

3. In the Top viewport, drag up on footstep 2 to rotate the footsteps -180 degrees about the Local Z axis.
4. Right-click and choose Move.
5. Move footsteps 2 and 3 about the Y axis, in line with the other footsteps.




Rotate and move the footsteps.

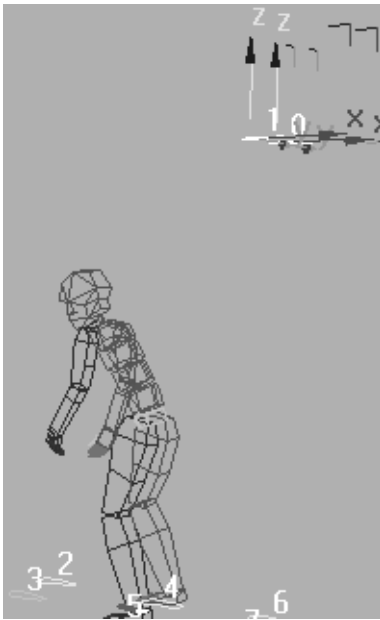
6. Play the animation.  
You'll see the biped do a half twist in the first jump. It will also do a backward half twist in the second jump. You'll fix that next.
7. Click Select and rotate and select footsteps 4, 5, 6, and 7.
8. Drag up on footstep 6 in the Top viewport and rotate the footsteps -180 degrees about the Z axis.
9. Click Move and drag the footsteps so footstep 6 is on top of footstep 0.
10. Play the animation again.  
The biped does a half twist in the first jump. The second jump is a simple flip ending with the biped facing the opposite direction.
11. Load *flip6.bip* to verify that your motion sequence is accurate.

In the following procedure you'll move some of the footsteps to simulate a jump from a height down to a lower level.

You will use a Track View selection technique to select footsteps 0 and 1 because they are covered by footsteps 6 and 7.

### Adding height to the flip

1. Make sure Footstep mode is still on.
2. Right-click and choose Track View Selected. Track View opens. *Bip01Footsteps* are displayed in the track as blue and green blocks.
3. Region-select the blue and green footstep blocks numbered 0 and 1 to select the footsteps.
4. Close the Track View window.
5.  Click Move, and drag footsteps 0 and 1 up 80 units about the Local Z axis.



#### Move the footsteps up.

You can now use regular region selection in the viewports to select and move the remaining footsteps.


6. Select footsteps 2 and 3 and move the footsteps up 40 units about the Local Z axis.

7. Select footsteps 4 and 5 and move the footsteps up 15 units about the Local Z axis. Footsteps 6 and 7 remain at ground level.
8. Turn off Footstep mode and play the animation.

The biped jumps from a height, twists, lands on a lower level, continues the jump, flips, and comes to a stop on the lowest level. The levels make the acrobatics more believable and interesting.



#### Jump and twist from a height

9.  Load *flip7.bip* if you want to verify that your motion sequence is accurate.
- To add to the realism, you could create a complete scene by constructing a series of platforms under each set of footsteps.



Dr X flips out

## Lesson 4: Animating a Pratfall

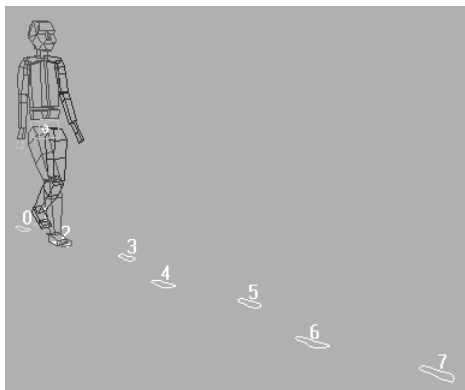
In this tutorial you will animate the biped slipping on a virtual banana peel. It will teach you to change rigid footsteps to sliding footsteps and will also give you an introduction to creating a freeform period in a footstep animation using Track View.

### Setup

1. Load *cs3\_tut03\_slipstart.max*.
  2. Change the display to Wireframe.
  3. Select any part of the biped.
  4. Open the Motion panel.
- The Biped controls are now available.

### Understanding the animation

1. Play the animation by dragging the time slider.

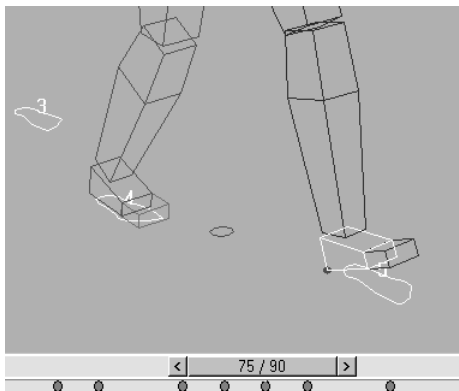


#### The biped walks about 5 steps

You're going to make the biped slide on footstep 5. Imagine a banana peel on that footstep.

2. At frame 75, select the blue *Bip01 L Foot* object.
- Frame 75 is where the heel hits the ground for footstep 5.

3. On the Motion panel, open the IK KeyInfo rollout.
4. Zoom into the viewport so you can see the foot easily.

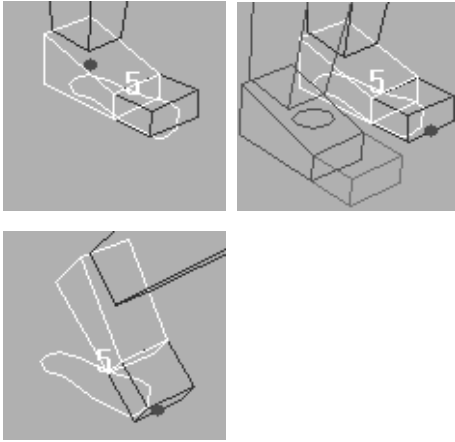


#### Heel touches down

In the viewport, the pivot point is displayed in red, and the track bar displays keys. The key for the foot at frame 75 is a planted key. This means IK Blend is set to 1 and Join to Prev IK is on.

IK Blend at 1 locks the pivot and the foot to the ground plane. Join to Prev IK Key considers the rotation of the foot at the previous key and attempts to match it.

5. Continue to scroll through the animation, watching the pivot points move.





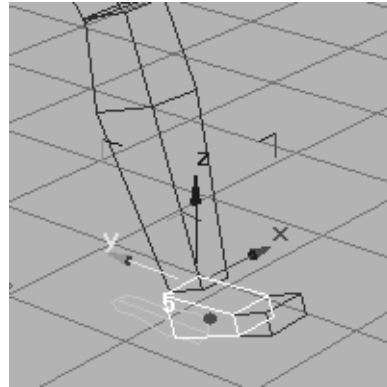
The pivot point animates.

- At frame 78, the pivot shifts to the outside of the heel.
- At frame 84 the pivot moves to the toes.
- At frame 93 the heel is lifting in the air. After this frame the foot is off the footstep and in the air.

**Tip:** Use Key mode to jump from key to key. This makes it easier to understand the animation. Click on the forwards and backwards arrows on the time slider to move to the next or previous keys. You can also use the Next Key and Last Key buttons in the VCR Controls.


### Creating a Sliding Footstep

1.  Turn on Key mode and go to frame 75.
2.  In the IK Key Info rollout, choose Set Sliding Key.
3. Advance to the next key (at frame 78).
4. Rotate the foot so the toes remain up in the air.




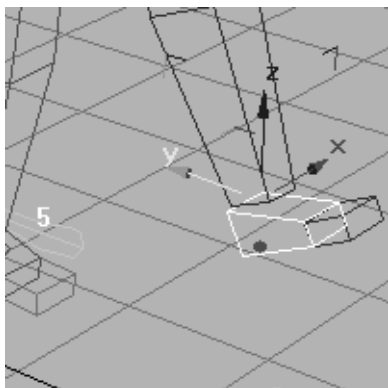
Rotate the foot up.

5. Move the foot forward, so it slides up on the footstep gizmo.


6.  In the IK Key Info rollout, choose Set Sliding Key.

The pivot point jumps to the foot's new position. The footstep is now displayed with a line through it. This indicates it's a sliding footstep: the foot can slide over or off the footstep (and the ground plane).

7. Select the foot again.
8. Advance to the next keyframe (84).
9.  Set a sliding key.
10. Choose Select Pivot and change the pivot from the toe to the heel.
11. Turn off Select Pivot.
12. Rotate the foot, then raise it up in the air, so the foot is starting to kick upward.



The foot at frame 84

13.  Set a sliding key.

Again the pivot point jumps from the footstep to the foot.

If the foot is off the footstep and is in the air instead, use a free key. A sliding key, however, will also let you lift the foot off the footstep.

14. Move the time slider to play the animation.  
15. The biped's left foot slides over and off footstep 5.

Next you'll create a freeform period so you can freely animate the biped's movement in the air.

### Creating a Freeform Period

Using a freeform period in a footstep animation allows you to suspend the automatic systems Bipod uses during footstep animation.

If you set up a freeform period between footsteps 5 and 6, you'll be able to animate the biped slipping, falling and bouncing on the ground plane.

1. Select the biped's center of mass.
2. Turn on Footstep mode.
3. Select a footstep in the viewport.

4. Right-click the footstep and choose Track View Selected.

Track View opens.

5. Expand Bip01 Footsteps.

The sliding footsteps are displayed in orange in Track View.

|   |    |   |    |  |    |   |     |   |     |
|---|----|---|----|--|----|---|-----|---|-----|
| 3 | 63 |   | 75 |  | 93 |   | 105 | 7 | 123 |
|   | 60 |   | 78 |  | 90 |   | 108 |   |     |
|   |    | 4 |    |  |    | 6 |     |   |     |

### Orange sliding footsteps in Track View

6. Double-click footstep 6 in Track View.  
Footstep 6 highlights in white.  
7. In Track View, move footstep 6 so it is directly under footstep 7.

|   |    |   |    |  |    |  |     |   |     |
|---|----|---|----|--|----|--|-----|---|-----|
| 3 | 63 |   | 75 |  | 93 |  | 105 | 7 | 123 |
|   | 60 |   | 78 |  |    |  | 105 |   | 123 |
|   |    | 4 |    |  |    |  | 6   |   |     |

### Create a gap for a freeform period.

The footsteps display the starting and ending frames in Track View. As you move the footstep in Track View, these frame numbers change.

**Tip:** You can also change the duration of the footstep by using the Footstep Edge Selection arrows in the Footstep mode dialog. Select an edge to move that edge. Select the dot in the center to move the footstep in time, without changing its duration.

8. Right-click the Footstep Track.  
9. In the Bip01 Footstep Mode dialog, turn on Edit Free Form (no physics).

In Track View a hollow yellow box appears to the right of footstep 5.

|   |  |   |   |  |  |  |   |  |  |
|---|--|---|---|--|--|--|---|--|--|
| 3 |  |   | 5 |  |  |  | 7 |  |  |
|   |  | 4 |   |  |  |  | 6 |  |  |

### Click the hollow box.

10. Click inside the hollow yellow box.



The box becomes solid yellow. This indicates it is a freeform period.



11. Scrub the time slider and watch the animation.

The biped floats from footsteps 5 to footstep 6.

### Changing the timing of the slip

The slip happens too slowly. You can change the timing of the sliding footstep so the freeform period can start sooner. Also, you need more time for the body to rotate and bounce on the ground.



1. Click Time Configuration in the Time Controls.
2. Change End Time from 123 to 150. Click OK.  
This extends the animation by adding 27 blank frames.
3. Turn on Edit Footsteps in the *Bip01* Footstep mode dialog.
4. In Track View drag a selection rectangle around footsteps 6 and 7.
5. Move footsteps 6 and 7 to the right so the yellow period is extended and the footsteps 6 and 7 don't start until frame 140.  
The last two footsteps extend beyond the end of the animation.
6. Press ALT + R.  
The end of the animation now matches the end of the footsteps. The time slider now shows the animation has been extended to 158 frames  
**Tip:** Be sure the Plug-in Keyboard Shortcut Toggle is on, or the Biped keyboard shortcuts won't work.

7. Click the orange footstep in Track View.
8. In the Footstep mode dialog, in the Footstep Edge Selection group, click the right arrow. A white dot appears at the right side of the orange box.
9. Adjust the right side so the footstep ends at frame 83.




10. Play the animation. The biped slides faster over the footstep.  
Since you're still in Footstep mode, you can also move the footstep's position in the viewport.
11. Select footstep 6 in the Perspective viewport and move it next to footstep 7.
12. Move the time slider to observe the animation.  
Now you have enough time and space for the pratfall.

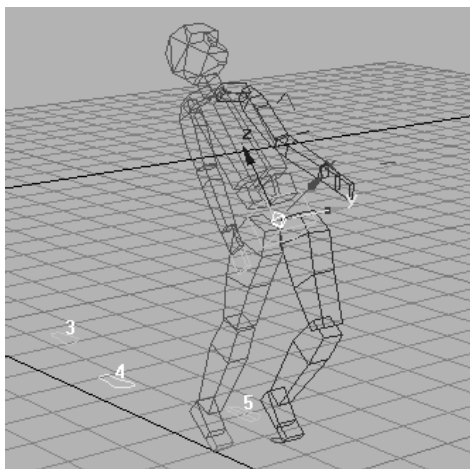
### Keyframing the Fall

To keyframe the fall, you'll rotate the center of mass so the biped moves from vertical to horizontal. You'll also move the center of mass so the biped rises up and then falls to the ground plane. You'll turn off Dynamics Blend, so Biped physics are not used in the animation.

1. Choose Edit Free Form (no physics) in the *Bip01* Footstep Mode dialog,
2. Close the Track View window.
3. Turn Animate on.  
Since the center of mass doesn't use the planted, sliding, or free keys, you can just use the Animate button to set keys automatically.
4. Turn off Footstep mode.

Now you'll be able to see keys in the track bar.

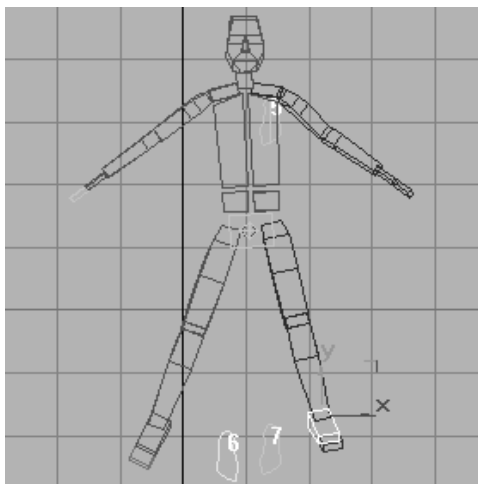
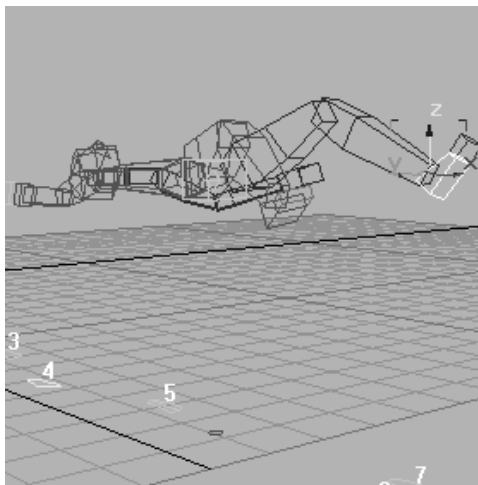
5.  In Track Selection, select the Body Vertical Track.
6. At frame 78, click Set Key on the Key Info rollout.
7. On the Key Info rollout in the Body Dynamics group, change Dynamic Blend to 0.  
This turns off Biped Dynamics and applies Spline Dynamics instead. Biped Dynamics adds physics to the animation. Spline Dynamics doesn't. Dynamics Blend is only available on Body Vertical track keys.
8. Move to frame 84.  
The left foot has a planted key at this frame.
9. Move the center of mass forwards a little.
10. Rotate the center of mass slightly so the biped begins its pratfall.



**The biped at frame 84**

11. Select the blue foot and set a sliding key.
12. At frame 86 rotate the center of mass so the biped is horizontal.

13. Select and move each foot so it's raised up and spread apart.
14. Select and move each arm so it's positioned out from the body.




**The biped at frame 86**

The biped's feet will swing up for a frame or two after the rest of the biped begins to descend.

15. At frame 88 move each arm and foot up some.  
Be sure Animate is on, otherwise set free keys for these movements.
16. Rotate the feet a little so the toes point upwards.  
This adds some secondary motion to the animation.
17. At frame 97 select and move the center of mass so the biped is close to but slightly above the ground.  
This creates the descent of the horizontal biped to the ground.
18. Use the time slider to flip through the biped's slipping motion.

### Adding a bounce

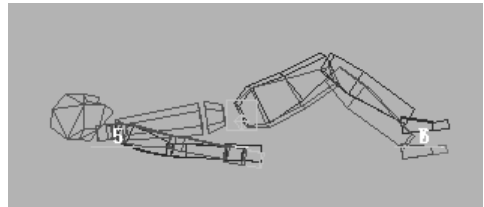
Next you'll add keyframes to make the biped strike the ground and bounce once. There is a Body Vertical key set at frame 115 you need to reset.

1.  Select the Body Vertical Track and move the time slider to frame 115.
2. Make sure Footstep mode is off, then change the Dynamic Blend to 0.

Note: If Dynamic Blend is unavailable, click Set Key first. Dynamic Blend is only available where there are Body Vertical keys.

3. Rotate the biped so it's horizontal.
4. Move the biped to the ground.
5. Keyframe the center of mass so the biped lifts off the ground at frame 102.  
Add secondary movement by keyframing the hands to trail the movement of the center of mass.
6. A few frames after the center of mass rises, raise the hands.

7. A few frames after the center of mass strikes the ground, make the hands strike the ground.  
Next you will keyframe the spine.
8. Rotate the spine so the shoulders touch the ground at frames 95 and 110.  
Next you will keyframe the head.
9. Rotate the head so the chin tucks in at frame 95 and the head rocks back at frame 102.



Biped position at frame 95

10. At frame 95, select the center of mass.
11. In the Key Info rollout change the Tension and Continuity to 0.

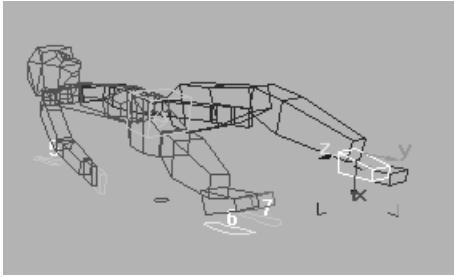
This sharpens the bounce.

**Tip:** If the Tension and Continuity curves are not available, click previous key, then next key to return to this keyframe with the curves displayed.



Tension and Continuity at zero

12. Play the animation.  
The biped bounces more crisply.




**Biped position at frame 115**

13. Save your work to *myslip.bip*. Compare your work to *slip1.bip*.

The timing is still off a bit, and the biped stands back up when the animation finished. Since **character studio** lets you change back and forth between footstep and freeform modes, you can convert between animation modes to correct this.

### Converting from footstep to freeform

You can convert between the footstep and freeform modes without any loss of motion. The motions will remain identical, whether you convert from footstep to freeform or freeform to footstep.

1. Continue in the current file, or load *cs\_tut03\_convert.max*.
2. Turn on Animate.
3. Select the biped's center of mass.
4.  In the General rollout, choose Convert. Do not turn on generate a keyframe per frame. Choose OK.  
The footsteps disappear from the viewport, but the animation is the same as before. Next use Copy Pose to eliminate the biped standing back up.
5. At frame 115 select the entire biped.

6. In the track bar select all the keys after frame 115.
7. Delete all the selected keys.

### Changing the timing

 Animate

1. Be sure Animate is still on.
2. Select the entire biped.
3. In the track bar, drag a box to select all the keys from frame 86 to the end of the animation.
4. Move the time slider to frame 89.
5. Slide all the keys down so the keys from frame 86 are now at frame 89.

**Tip:** To move keyframes to a precise location, first select the keys. Then move the time slider to the desired location, then move the keys.

The pratfall lasts a little longer now.

If you like, you can add some more rotations to the head after frame 115. While the biped is laying on the ground make his head shake with disbelief. Also try keyframing some small movements in the feet, placing the feet flat on the ground and sliding them toward the pelvis, raising the knees.

6. Save your work as *myslip.max*, or load *slip2.bip* to see the results.

---

## Lesson 5: Changing Footsteps using IK Keys




Footstep and Freeform modes both use the same underlying inverse kinematics to animate the biped skeleton. Footstep gizmos are a method for manipulating sequences of IK keys.

In this tutorial you'll learn how making changes to the IK keys affects the footsteps.



**Setup**

- Load *cs3\_tut03\_footsteps\_keys.max*.  
A biped is displayed with four footsteps in the viewport.

**Setting IK Keys to create footsteps**

1. Move the time slider to play the animation.  
The biped hops on his right foot. Notice that there is no footstep for the right foot between footsteps 2 and 3.
2. At frame 45, select *Bip01 R Foot*.
3.  On the Motion panel, in IK Key Info menu, click Set Planted Key.  
The pivot point is displayed in the viewport.
4. Turn on Key mode.
5.  At frame 48, click Set Planted Key.  
The pivot point has been shifted to the toe. Note that the lowest IK pivot is selected by default for cases where IK is applied to new keys.
6.  At frame 54, click Set Planted Key.  
The biped is moved back to the ground. A footstep is displayed beneath the biped's foot.  
A footstep has been created, because there is now an interval of time where IK is applied between the two planted IK keys.

**Changing the duration of footsteps using IK Keys**

1.  At frame 60, click Set Planted Key.
2. Open Track View.
3.  Turn on Footstep mode, so you can see the Biped Footsteps track.

4. Notice that Footstep 3 extends for 15 frames from frames 45 to frame 60.



5. Turn off Footstep mode.

6. Select *Bip01 R Foot*.



7. At frame 63, set another planted key.



8. Turn on Footstep mode.





Track View again displays the Footsteps track.

The duration of the footstep now is 18 frames, from frames 45 to 63.



9. Turn off Footstep mode.

**Removing footsteps using IK Keys**

1.  At frame 45, set a free key.
2.  At frame 48, set a free key.
3.  At frame 54, set a free key.  
Note that a vertical key is modified to bring the biped down over its footstep since the liftoff frame for the jump now begins at frame 54 instead of frame 48.
4.  At frame 60, set a free key.  
The footstep disappears.  
There is only one IK key left. With no IK interval defined, there is no duration and therefore no footstep.



---

## Tutorial 4

# Animating Biped Interacting with Objects

---

## Making Biped Interact with Objects

So far, the focus has been on animating the biped alone or in relation to a ground plane. In this tutorial, you'll make a biped do the following things which involve another object:

- Climb a ladder
- Pick up a briefcase
- Bounce a basketball

The movement of the biped's hands can control the movement of the second object, or the biped's hands can be constrained to or animated by that other object. There are also lessons on creating the illusion of weight, and using In Place mode.

In the following examples, you'll learn more about making bipeds interact with objects.

### Lessons

*Lesson 1: Dribbling a Basketball (see page 504)*

*Lesson 2: Climbing a Ladder Using IK Blend (see page 507)*

*Lesson 3: Picking Up and Carrying Using Link Controller (see page 516)*

*Lesson 4: Creating the Illusion of Weight (see page 519)*

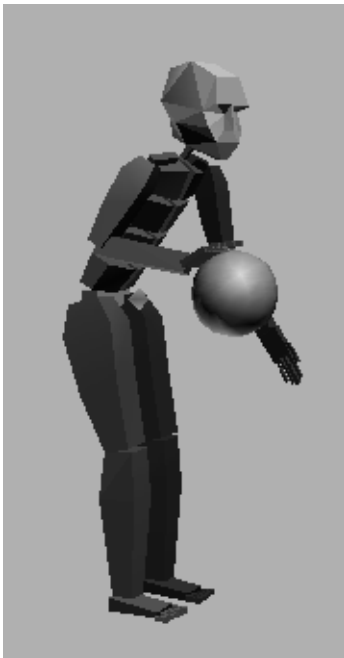
*Lesson 5: Using In Place Mode (see page 522)*

## Lesson 1: Dribbling a Basketball

This example animates the biped's hand and a basketball to simulate the biped bouncing the ball. The ball is animated on its own, bouncing up and down. When the ball is at the height of its bounce, it controls the hand through inverse kinematic blending. When the ball strikes the ground, the ball and the hand are not connected, and the hand is animated independently. You'll animate IK Blend on and off to create this effect.

### Setup


1. Open `cs3_tut04_bouncestart.max`.



2. Play the animation to see what happens. The biped bounces the ball 5 times while walking forward.

### Understanding the Keyframing

Use Key mode to understand the structure of the animation.

1. Select the ball.  
Keys appear in the track bar.
2.  Turn on Key mode.
3. Move to each key using Next Key or clicking the arrows on the time slider.  
The ball strikes the ground at frames 15, 39, 58, 78 and 101.  
The ball is at the top height of its bounce at frames 0, 30, 48, 67, 89, and 111.
4. Select the biped's right hand (*Walk01 R Hand*).  
Keys appear in the track bar.
5. Open the IK Key Info rollout and play the animation.  
The fields in the rollout blink rapidly because IK Blend is animated.
6. Move to each key, looking at the IK Blend field.  
IK Blend fluctuates between 1 and 0.  
Observe the pattern of the keys in the track bar. There are groups of three or more keys followed by a single key.



The groups have IK Blend set to 1, but the single keys have IK Blend set to 0. The groups also have Object Space selected. *Sphere01* (the basketball) has been designated as the Object Space Object.

Note that only one Object Space Object can be defined for the hand. You cannot change from one Object space object to another during the animation.



## Adjusting the Basketball's animation

Animate

1. Turn on Animate.
2. Select the ball.
3. At frame 30, move the ball up and closer to the biped's body.

The hand follows the position of the ball because the hand is locked into the object space of the ball.

**Tip:** If the hand does not move with the ball, select the hand and verify that Object space is turned on in the IK Key Info rollout.

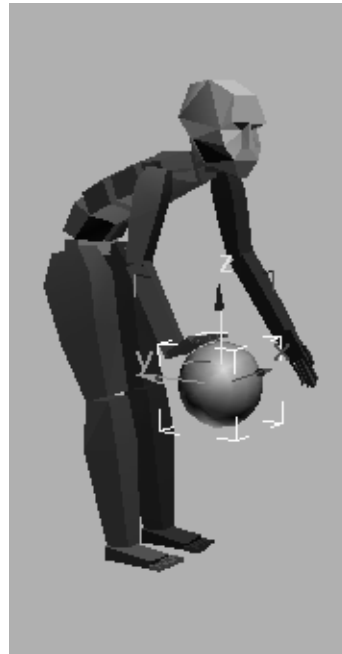


Hand follows ball

4. At frame 0, move the basketball down.  
The hand moves with the ball because since IK Blend is set to 1 and the ball is the designated Object.

Note: If you move the ball too far it leaves the hand behind.

5. Move the basketball up so it is in contact with the hand.
6. Rotate the spine so the biped crouches over.
7. Move the basketball down so the dribble is low.



### Rotate the spine.

Moving the basketball repositions the hand when the IK Blend is set to 1 and the basketball the object.

If you move the basketball while IK Blend is set to 0, the ball will not affect the hand.

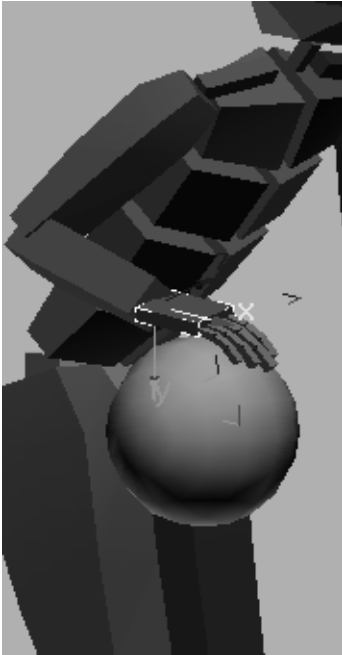
8. At frame 14, set IK blend to 0.
9. At frame 16, set IK blend to 0
10. At frame 15, move the basketball.

Nothing happens to the hand, because IK blend is 0.

### Animating the dribble

The basketball bounces one time more than the biped dribbles it. You'll set up the last dribble to see how this works.

1. With Animate still turned on, go to frame 111.
2. Select the right hand.
3. Move the hand up out of the way.  
You'll be moving it again, so it doesn't matter where you move it.
4. Select the basketball.
5. Move the ball in front of the biped, at about waist height.
6. Zoom in the Perspective, Left and Front viewports so you can see the biped's hand.
7. Rotate and move the biped's hand so it is in contact with the ball. Rotate the fingers so the hand cups the ball.
8. In the IK Key Info rollout, change IK Blend to 1 and turn on Object.  
*Sphere01* is already designated as the Object Space Object, so you don't need to select it.
9. Raise the ball.  
The hand moves with it.  
If you move the time slider back and forth, you'll see there's still more work to do.
10. At frame 109 move the hand down so it is just above the basketball, change IK Blend to 1, and turn on Object.
11. At frame 113 move the hand down so it's just above the basketball, change IK Blend to 1 and turn on Object.
12. Move to frame 117.  
You need to set IK Blend back to 0, or the movement of the ball will continue to influence the hand. But the fields are not available because there is no key set.
13. In the Key Info rollout, click Set Key.  
This sets IK Blend to 0 and turns on Body space.  
Move the time slider to see the movement.



## Lesson 2: Climbing a Ladder Using IK Blend

It takes a lot of steps to create a freeform animation because you're animating the feet and hands without any automation. Once you learn a technique, you repeat the same sequences over and over for each limb to create the keyframes.

In this lesson you'll load a file that is partially animated and you'll create the animation for just one of the arms. Normally, you would then repeat the steps for the other limbs.

### Setup

1. Open *cs3\_tut04\_ladder\_start.max*.
2. Play the animation.

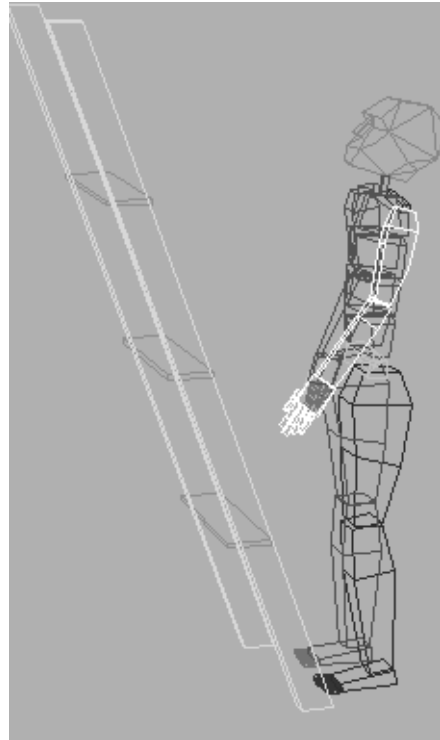
Let the animation loop while you watch the movement of the green hand. It grabs the ladder to support the biped as he steps up on the green foot.

You'll make the left hand grab the rung and then slide up the rail as the biped climbs.

### Creating the first poses

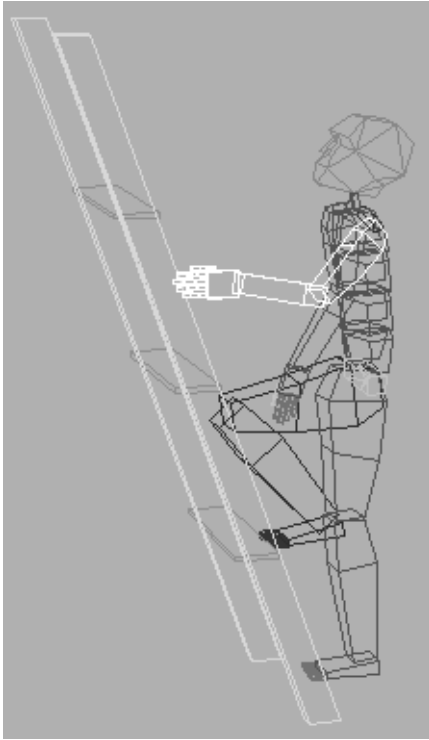


1. Turn on Animate.
2. Select *Bip01 L Hand*.  
At frame 0, the arms are at rest. The first pose is at frame 19.
3. At frame 19 move the hand forward and slightly up, as if the biped's about to shake hands.



The arm at frame 19

4. At frame 30 move the hand so the left fingers just touch the ladder.
5. Rotate the hand about the Y axis so the hand is perpendicular to the ladder rung.



The arm at frame 30

### Grabbing the ladder

You'll position the hand to grab the ladder. The fingers will rotate, the hand will have IK Blend set to 1, and you'll put the hand in the Object space of the ladder rail.

1. At frame 36 select the hand.
2. Move the hand so the palm is flush with the ladder rail, near the third rung.
3. In the Top viewport, move the hand to the right of the ladder rail.
4. Rotate the hand so the palm is flush with the ladder rail.

Watch the Top and Perspective viewports as you move the hand.

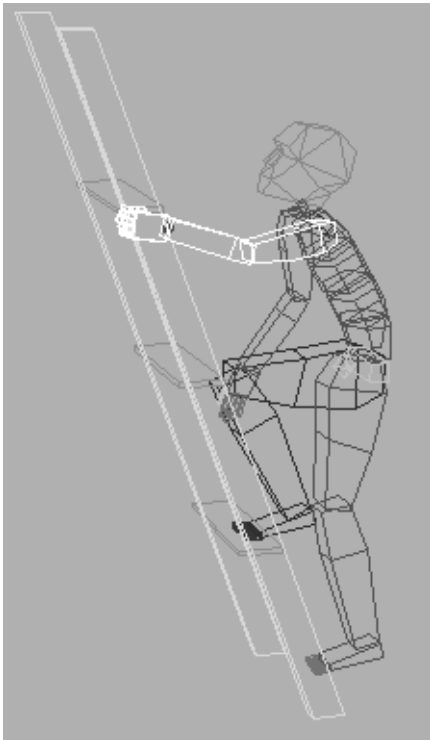
5. To select all the fingers, double-click the hand to select the hand and all the fingers, then hold down the ALT key and click on the hand.

Now all the fingers are selected.

6. In the Top viewport, rotate the fingers about the Z axis so they curl to grab the ladder. You can also adjust each finger individually.
7. Deselect the fingers and select the hand again.
8. In the IK Key Info rollout, change IK Blend to 1.
9. Change from Body to Object Space.

The Object Space Object, Box06 is now active.

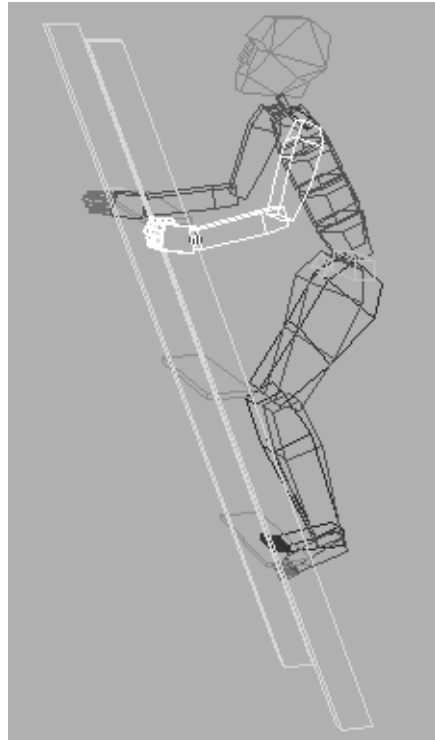
Move the time slider to see the animation. The biped appears to grab the rail.



The arm at frame 36

### Sliding the hand up the ladder

The hand is locked to the ladder rung at frame 36 so it stays as the rest of the biped moves in the frames that follow. To slide the hand up the ladder, you'll use a trick. You'll move the hand in the Reference Coordinate System of the ladder.



The arm at frame 49

1. Choose Select and Move in the Main Toolbar, and make sure *Bip01 L Hand* is selected.
2. In the Reference Coordinate System selector in the Main toolbar, choose Pick, then select *Box06*.

The Transform gizmo changes. You can move the hand so it remains aligned to the ladder rail.

3. At frame 45, move the hand up so it is farther up the rail.

The hand now immediately starts to slide up the rail from frame 36 to frame 45. It should grip the ladder for stability for a few frames before the slide begins.

4. Hold down the SHIFT key and move the key on the trackbar from frame 36 to frame 39.
5. This clones the key from frame 36 to 39.
6. In Key Info, set the Tension and Continuity to 0 at frames 36 and 39.
7. Move the time slider to play the animation. The hand grips the rail starting at frame 36, then begins the slide from frames 39 to 45.
8. At frame 49, move and rotate the hand so the gripping fingers stay wrapped around the ladder.

The movement takes place in the Reference Coordinate System of the ladder rail (*Box06*), but the rotation does not. Each transform can use a different coordinate system. Selecting one system for position doesn't affect the system for rotation.

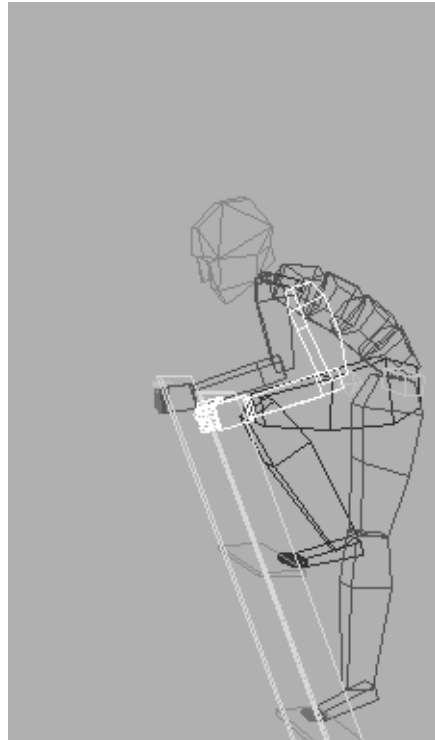
### Opening and Closing the Grip

At frame 61, the green foot is planted on second rung. The hand will release its grip on the rail.

1. At frame 61, select all the fingers.
2. Rotate the fingers so they loosen their grip.
3. Move the hand out so it comes off the rail a little.

At frame 80, the blue toes touch the top rung. The fingers grip again.

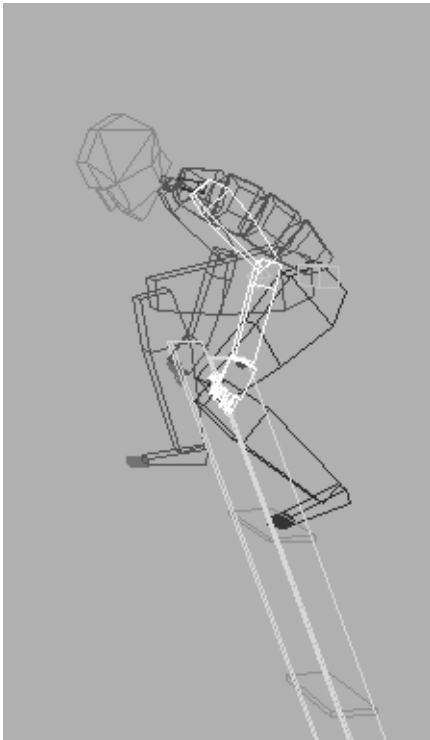
4. At frame 80 select all the fingers.
5. Rotate the fingers back into a grip.
6. Move and rotate the hand so it grips the ladder at the top of the rail.



**The arm at frame 80**

The green foot is planted on top of box at frame 115. This is the last frame in which the hands grip the rungs.

7. At frame 115, rotate the hand about the Y axis, so the hand pivots on the top of the rung.




**The arm at frame 115**

8. Hold down the SHIFT key and move the key on the track bar from frame 115 to frame 122.

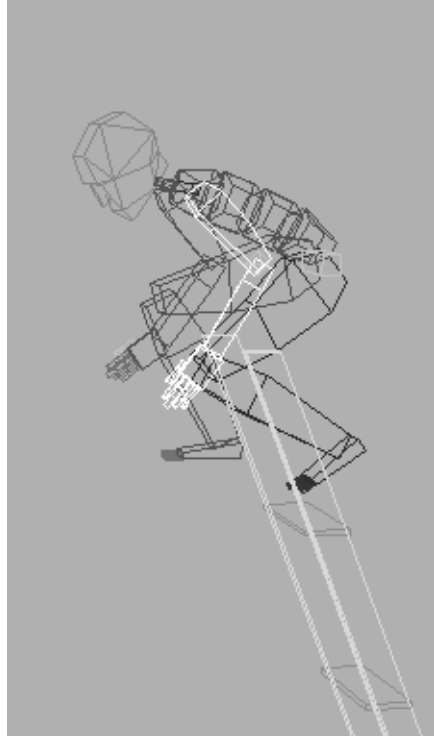
This clones the key from 115 to 122.

9. In the Key Info rollout set Continuity and Tension to 0 for keyframes 115 and 122.  
This locks the hand in place until frame 122.

### Releasing the grip

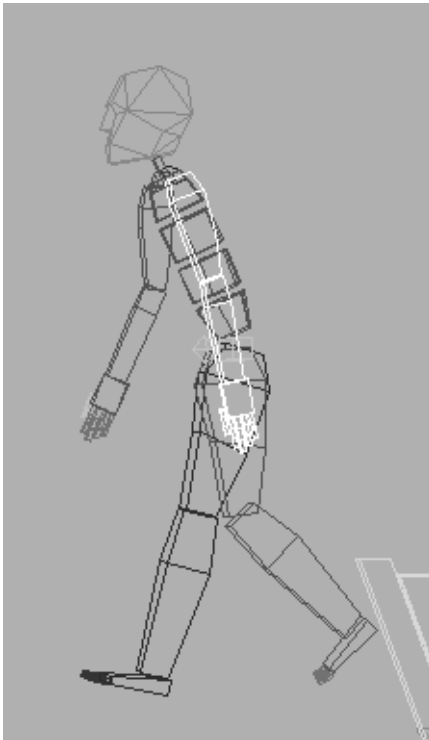
1.  At frame 126, select the hand again and set a free key.  
Setting a free key sets IK Blend to 0 and turns on Body Space (it turns off Object Space).

2. Move the hand off the ladder.  
The hand is not affected by the ladder any more.
3. Select the fingers and rotate them to relax the grip. Rotate the hand about the Y axis so the hand is pointing more towards the ground.



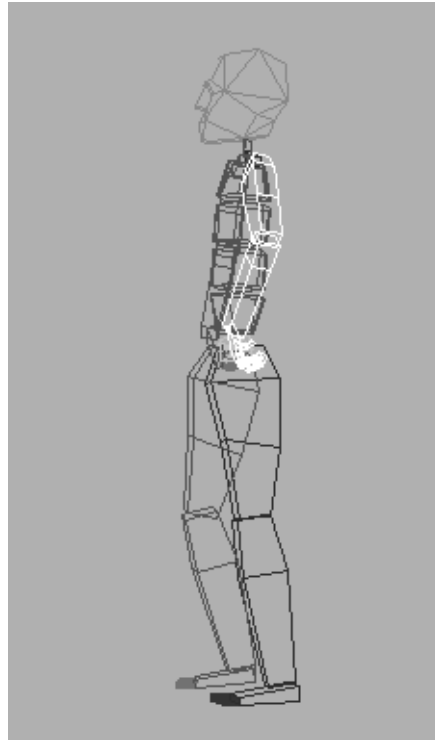
**The arm at frame 126**

4. At frame 154 rotate and move the hand so it is loosely outstretched.  
Biped steps onto the blue foot.



**The arm at frame 154**

5. At frame 169 select the entire right arm. In the Keyframing rollout, choose Copy Posture, then Paste Opposite.
6. The biped stops, rests his hands on hips, and preens.



**Hands on hips, fingers curled back**

Everything you just did was also done on the opposite hand. The feet also were animated by creating IK Blend keys to lock and unlock them to the rungs.

Use *Animating a Walk Cycle with IK Constraints* (see page 457) and *Animating a Leap with Freeform Animation* (see page 586) to animate the feet and hands working with pivot points.

7. Save your work as *myladder1.max*. If you want you can open *cs3\_tut04\_ladder\_end.max* to compare your work to the correct animation.



### Linking the center of mass to an object

When the hand is locked to the ladder, any animation of the ladder affects the hand. If you want the movement of the ladder to affect the entire biped, use the Select and Link tool.

You'll look at a scene from the production "Pool Tools" to see this in action. When the ladder in the scene swings from side to side, the biped will swing with it.

### Setup

1. Open *cs3\_tut04\_ladder\_swing.max*.
2. Play the animation.



The pool guy climbs the ladder.

The animation of the pool guy climbing the ladder has already been done. The ladder has also been animated, but there is no link between the two.




The ladder swings, but the biped stays put.

You will lock the hand down and then link the biped's center of mass to the ladder.

### Making the hand grab the rung




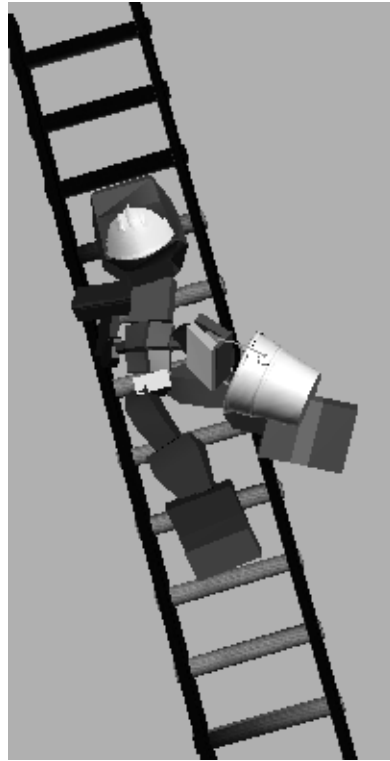
1. At to frame 431, turn on Animate.
2. Select the left hand and move it close to the ladder rung.
3. Select the *Bip01 L Finger0*.
4. Right-click and choose Rotate.

5. Rotate the finger about the Z axis, to grasp the ladder.
6.  In IK Key Info, set a sliding key. This sets IK Blend to 1, which locks the hand in Object Space.

### Making the biped swing with ladder

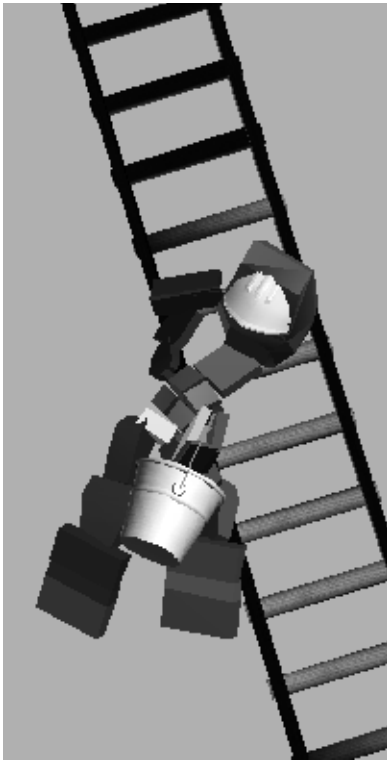
If you play the animation you'll see that the ladder still swings, leaving the biped behind in the air. You'll use the Select and Link tool in the Main toolbar to change this.

1.  Choose Select and Link from the Main Toolbar.
2. Click on the biped's center of mass. When the icon changes, drag to the ladder and link.



Linked center of mass moves biped with ladder

3. At frame 451, select and move the center of mass. The hand is locked, but the rest of the biped can be changed.  
Note: If you move the center of mass far enough, the hand leaves the ladder. If you override the IK limits, the forward kinematics take precedence.
4. Rotate the center of mass about the X axis, so the biped swings a little bit.  
Move and rotate the center of mass until the biped is in position.



**Hand is locked, the rest of the biped moves**

5. Repeat the process at frame 482, moving and rotating the center of mass, so the hand remains on the ladder and the body swings in the opposite direction.



**Biped swings with the ladder**

To finish the animation, you'd have to repeat this process to match all the keys as the ladder continues to wobble back and forth. You can do that for practice if you like.

## Lesson 3: Picking Up and Carrying Using Link Controller

When a biped needs to pick something up, carry something or put something down, you can use the Link Controller to animate the biped, then let the biped's hand animation control the movement of the object.

In this lesson, a character picks up, carries and puts down a briefcase.

### Setup

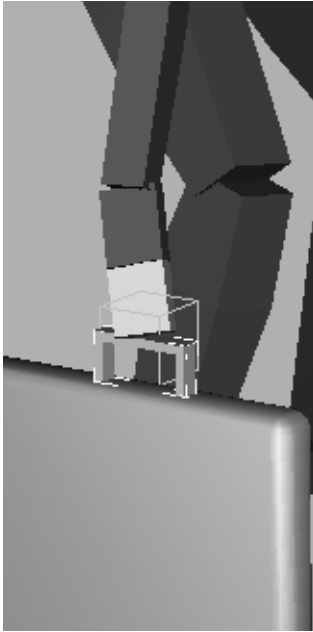
1. Open `cs3_tut04_briefcase.max`.



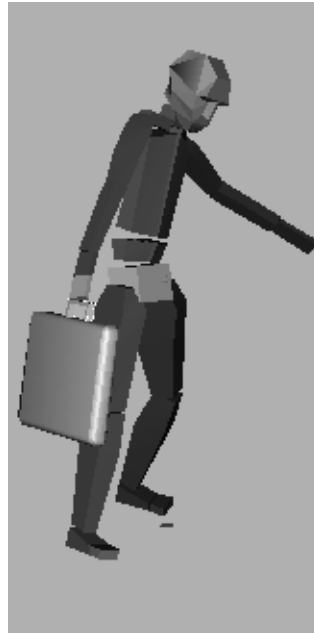
2. Play the animation.  
The biped picks up, carries and sets down a heavy invisible object with a handle.
3. Select *Bip01 R Hand*.
4. Turn on Key mode and move through the keys.  
There is a key at frame 56, where the biped picks up the briefcase. The biped carries the briefcase until frame 228.  
You'll use these frames for the changes of linkage.

### Assigning the Link Controller

1. At frame 56 open the Motion panel.
2. In the Perspective viewport select *handle*.  
**Tip:** Check the name at the top of the command panel to confirm your selection.
3. In the Assign Controller rollout, click Transform: Position/Rotation/Scale.
4. Click Assign Controller.
5. In the Assign Transform Controller dialog choose Link Control and click OK.  
This assigns a Link Controller to the object the biped is going to pick up.
6. Zoom into the viewport to see the handle, the dummy, and the biped's hand.



7. In the Link Parameters rollout, click Add Link.  
You'll link to two different objects. The biped hand will animate the briefcase. The handle will link to a dummy when you want the briefcase to remain stationary.
8. Select the blue dummy object near the briefcase handle.  
The name "*stay put dummy*" appears in the Link Controller list.
9. Change the Start Time to 0.  
Now you'll add *Bip01 R Finger0* as the second link. You don't need to click Add Link again, as it is still active.
10. Select *Bip01 R Finger0* (the large yellow box at the end of the biped's hand).  
*Bip01 R Finger0* appears in the Link Controller list, with a Start time of 56.
11. Select the dummy again and change its Start Time to 228.
12. Play the animation.  
The biped picks up the briefcase, carries it, then sets it down.



We didn't bother rotating the fingers to create a grasp, since this biped has only one giant finger. For bipeds that do have more fingers you would link to the *Bip01 Hand* objects, rather than the finger objects. This would let you rotate the fingers together or individually to create the grasp without affecting the briefcase.

**Tip:** Do not make changes to the biped's hand after creating the links. If you need to correct the animation of the hand, delete the links, and then create them again after the animation is correct.

If the biped carries the object the entire time, you do not need to use the Link Controller. Instead, attach the object to the biped hand using Select and Link.

## Lesson 4: Creating the Illusion of Weight

There are two techniques in **character studio** that give the illusion of weight to an animation. Both affect the biped center of mass.

The first technique uses the Balance Factor, which moves the center of mass. Balance Factor is available the Body Horizontal track. This technique creates the illusion of lifting a heavy object. It lets you keyframe the center of mass moving in and out of the body.


The second technique uses Figure mode to turn on Rubber Band. You move the center of mass in front of or behind the body. You can't keyframe the center of mass with this technique. It creates the illusion of the biped pushing or dragging a heavy object.

### Lifting Heavy Objects

#### Setup

1. Open *cs3\_tut04\_balancefactor.max*.  
Two bipeds have planted keys set on their feet, with the pivot points set at their toes.
2. Select any part of the biped on the left and then open the Motion panel.

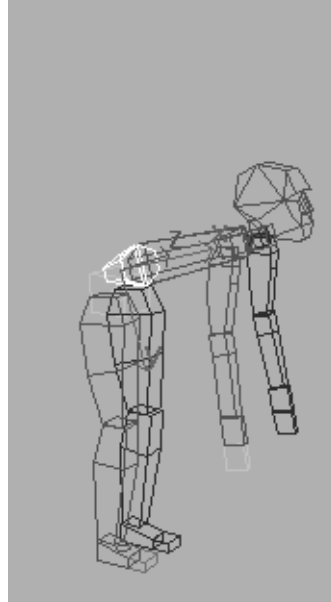
#### Using Balance Factor

1.  Select the Body Horizontal track.
2. In the Key Info rollout, set Balance Factor to 0.  
The Balance Factor is available because this file contains a keyframe on the Body Horizontal track at frame 0.



3. At frame 15, turn on Animate.

4. Select *Bip01 Spine* object (the first spine object).
5. Rotate the spine about the Z axis.

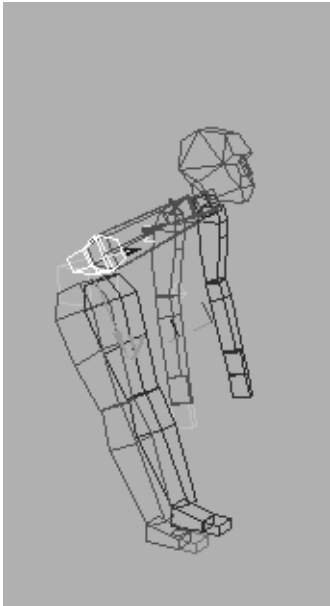


#### Only the upper body rotates

Notice that the upper body rotates, the legs stay firmly planted.

#### Animating the Balance Factor

1. Select the other biped.
2. In the Track Selection rollout, select Body Horizontal.
3. In Key Info rollout, click set key.  
Now Balance Factor is available.
4. Change the Balance Factor to 2.
5. Select *Bip02 Spine*.
6. Rotate the spine again.

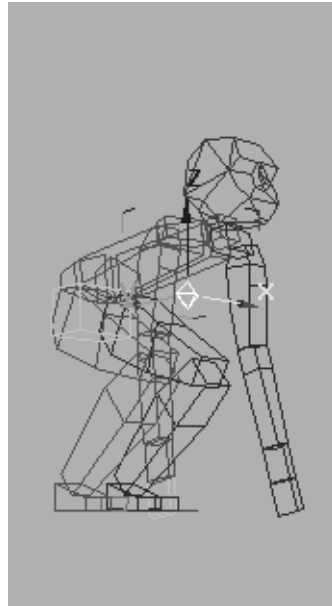


**Hips move back, torso rotates forwards.**

This time the hips move back as the torso rotates forward. If you rotate the torso enough, the feet move off the floor.

Notice also that the center of mass is now in front of the body.

7. Select Body Vertical and move the center of mass down, so the knees are bent.
8. At frame 23, rotate the *Bip02 Spine* more, so the knees touch the chest.
9. Move the center of mass farther away from the body.

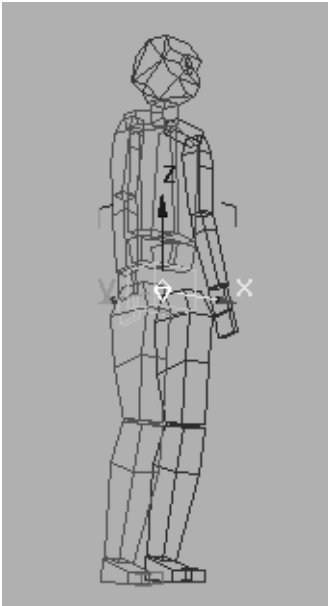


**Center of Mass moves away from the body**

The biped leans over even more.

10. At frame 30, move the center of mass so the biped starts to lift the imaginary object using its legs, rather than its back.
11. At frame 38 in the Track Selection rollout, select the Body Horizontal track..
12. On the Key Info rollout click set key.
13. Change the Balance Factor to 1.  
The center of mass moves back closer to the biped.
14. Select and rotate the spine.
15. At frame 45, rotate the spine more.
16. Move the center of mass so the biped stands up straight. Now the center of mass is back inside the body.





17. Move the time slider back and forth to the animation.

Watch how the center of mass moves outside the biped, then back again.

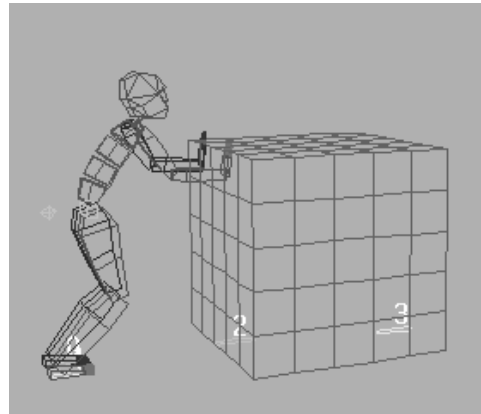
The biped appears to be lifting something heavy because of the positioning of the knees and spine.

## Pushing Heavy Objects

### Setup

1. Open *cs3\_tut04\_pushbox.max*.
2. Play the animation.

The biped is pushing a box along the floor. Notice that the center of mass is behind the biped.



### Adjusting the center of mass in Figure Mode with Rubber Band

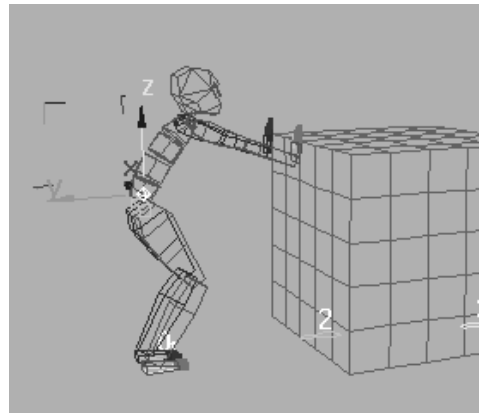
1. Select any part of the biped.
2. In the Motion panel, turn on Figure mode.



3. In Track Selection, choose Body Horizontal.

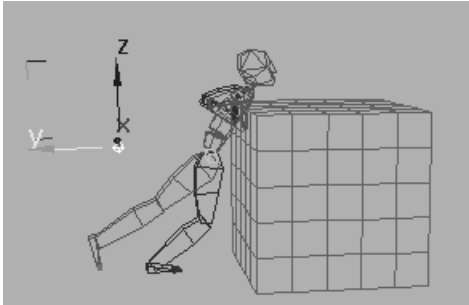


4. Turn on Rubber Band mode and move the biped's center of mass so it is back inside the body.
5. Turn off Figure mode and play the animation.



The upper torso moves back over the feet. The illusion of weight is diminished.

6. Turn on Figure mode and move the center of mass far behind the body.
7. Turn off Figure Mode.
8. The biped leans into the box, as though the box were quite heavy.




## Lesson 5: Using In Place Mode

When you're animating a character who's moving through space it's hard to evaluate the motion when the character moves out of your view. You can use In Place mode to view the moving character.

### Using In Place Mode

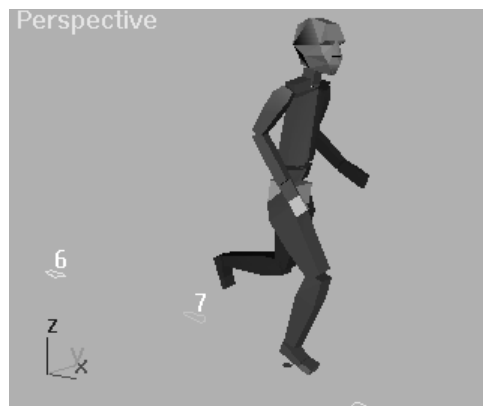
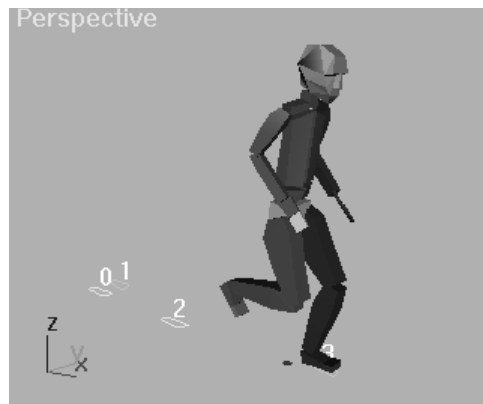
1. Open *cs3\_tut04\_inplace1.max*.
2. Play the animation.  
This is an animation of a biped running. You can see the entire animation but you're too far away to really see what is going on.
3. At to frame 0, in the Left viewport, select the entire biped with a selection rectangle.
4. Choose Zoom Extents Selected.  
The Left viewport zooms to frame the biped and you can see the pose clearly.
5. Maximize the viewport by pressing W.
6. Open the Motion panel.

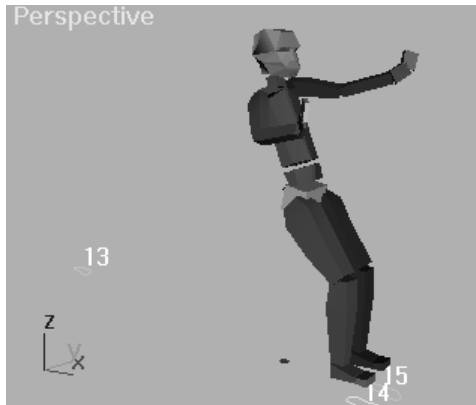
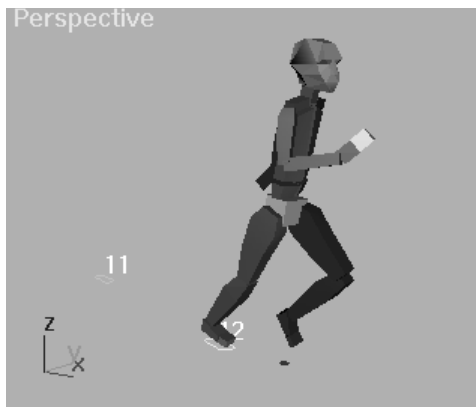
7.  In the General rollout, turn on In Place mode.
8. Play the animation in the Left viewport.
9. While the animation is playing, press CTRL + R + V.

This puts you into Arc Rotate mode. You can now rotate the viewport while the animation plays.

10. Rotate the viewport around, while the biped is running so you can see the animation from all sides.

Since you are using In Place mode, the biped remains in view and never leaves the frame.





In Place Mode keeps the action in view.

**Tip:** There are three navigation actions: Rotate, Pan and Zoom. Use the keyboard and mouse shortcuts to launch these without interrupting other operations. They are:

- **Rotate.** CTRL + R + V to turn on the mode, then left mouse button to rotate.
- **Pan.** Middle mouse button.
- **Zoom.** CTRL + ALT + middle mouse button.



---

## Tutorial 5

# Motion Flow Editing

Motion Flow mode lets you combine *.bip* files to create longer character animations. The *.bip* files, or *clips*, are joined using velocity-interpolated or minimum foot sliding transitions for a seamless transition between the source and destination clip.

In this tutorial, you'll do the following:

- Combine two motion capture *.bip* files and unify the result.
- Create a Shared Motion Flow so you can use one motion flow graph to animate multiple bipeds.
- Use Create Random Motion to create random scripts for multiple bipeds, so you can animate a crowd.
- Use Create Random Motion to create a random script for one biped.

## Lessons

*Lesson 1: Creating Clips in Motion Flow Mode (see page 526)*

*Lesson 2: Creating and Using Motion Flow Scripts (see page 526)*

*Lesson 3: Looping Animation in Motion Flow Mode (see page 529)*

*Lesson 4: Using Shared Motion Flow (see page 530)*

*Lesson 5: Using the Create Random Motion Feature (see page 532)*

## Lesson 1: Creating Clips in Motion Flow Mode

### Setup


Reset 3DS MAX. You must add a line to your *biped.ini* file so that *.bip* files can be found when you load in the example files. *Biped.ini* is in the *\plugcfg* directory under your main 3DS MAX directory. Use a text editor to add the following line to the *biped.ini* file.

MoFlowDir=<maxdir>\tutorials\tutorial\_5



Replace <maxdir> with the drive letter and directory of your copy of 3DS MAX. For example if your copy of 3DS MAX is in your d:\3DSMAX directory the line would be:


MoFlowDir=d:\3DSMAX\tutorials\tutorial\_5

### Creating a biped

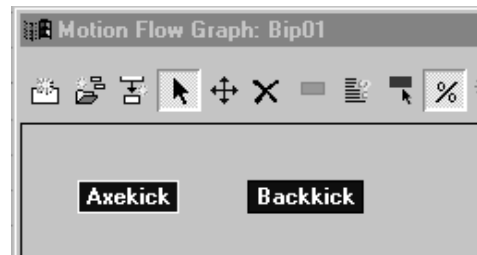
1.  On the Create Panel, click Systems.
2. On the Object Type rollout, click Biped.  
The Biped button turns green.
3. In the Perspective Viewport, click and drag, starting at approximately 0,0,0.  
A biped appears in the viewports.

### Adding clips in Motion Flow Mode

1. Select any part of the biped and open the Motion Panel.  
A biped must be selected for Biped controls to display on the Motion Panel.
2.  On the General rollout, turn on Motion Flow mode.  
Two rollouts display: Motion Flow and Motion Flow Script.
3.  On the Motion Flow rollout, click Show Graph.

4.  On the Motion Flow Graph toolbar, click Create Multiple Files.
5. Click Browse and load *cstudio\tutorials\tutorial\_5 axekick.bip*.
6. Hold down the CTRL key and click *cstudio\tutorials\tutorial\_5 backkick.bip*.
7. Click OK.

The two clips appear in the Motion Flow Graph window.




## Lesson 2: Creating and Using Motion Flow Scripts

The Motion Flow Graph stores clips. To create a script, you select clips from the Motion Flow Graph.

### Setup

Continue from the previous lesson or load *MoFlow\_tut01.max*. Files are in the *cstudio\tutorials\tutorial\_5* directory.

### Creating a script

1.  On the Motion Flow Script rollout, click Define Script.
2. On the Motion Flow Graph, click the Axekick icon first, and then the Backkick icon.  
The clip names appear in the list on the Motion Flow Script rollout. The name

Script1 displays in the text field above the list. In the Motion Flow Graph, a red transition line joins the two clips, showing that they are included in the current script. By default, Biped uses “Minimum Motion Loss” to find a likely starting frame in the source and destination clips. You have created a script called *Script1*.



3. Close the Motion Flow Graph window and click Play.

Use Arc Rotate Selected to rotate the Perspective view for a better look at the motion.

As the animation plays, a three-star icon next to the clip names (in the Motion Flow Script list) changes to show which clip is playing. The numbers on the right are the frame numbers where the animation starts.

The default Minimum Motion Loss search method did not choose ideal starting frames in this case: the biped spins unnaturally. You'll first edit the transition manually then you'll try an optimized transition.

Note: Optimized transitions use minimum foot sliding.


### Using The Transition Editor

When you edit transitions manually, you can discard unwanted motion and set the duration of the transition. You can also change the direction of the destination clip in the Transition Editor.

1. Click Axekick in the Motion Flow Script list. Controls on the Motion Flow Script rollout are now active.

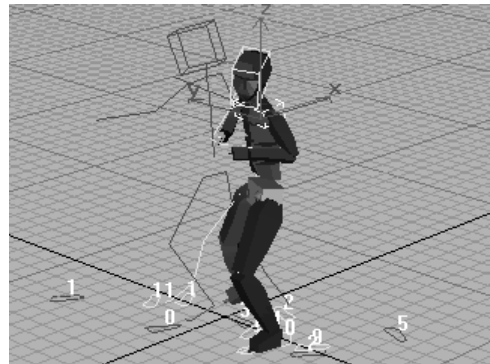
2.  Click Edit Transition.

The Transition Editor dialog displays; the source and destination clip names appear in the title of the dialog. Position the dialog so the Perspective Viewport is visible.

3.  Click Go to Start Frame near the upper right of the dialog.

The time slider is positioned at the first frame of the transition. By default, the duration of the transition is 25 frames. This is displayed in the Length field on the upper left of the Transition Editor.

Red and yellow stick figures (Ghosts) in the viewports help to establish a starting frame for both clips. Both stick figures display during the period of transition. Red represents the destination clip motion. The biped interpolates between the yellow and red stick figures over the course of the transition.



4. In the dialog, set Angle to -50.  
The destination clip reorients.
5. Set the Destination Clip Start Frame to 0.


You can choose the destination starting frame by viewing all of the clips motion.

6. Scrub the time slider back and forth over the transition.

You'll look for areas in the source and destination clips where foot and body movement will work. Frame 80 looks like a reasonable starting frame for the source clip, the character's right foot is in the air. Body weight is shifting to the right foot.

7. Set the Source Clip Start Frame to 80.
8. Move the Frame spinner in the Destination Clip Ghost area and watch the red stick figure representing the destination clip. At frame 20, the character is starting to push backward with his right foot. The character's weight will be on the right foot in both clips if this frame is used as the Destination Start Frame.



9.  Click Set Start Frame in the Destination Ghost area. The Frame value (20) is copied to the Destination Set Start Frame field; BipEd recalculates and positions the destination clip.
10. Set Length to 6 frames.
11. Play the animation.

The weight shifts to the right foot in both clips during the transition. The transition looks natural. Next you'll try an optimized transition.

**Tip:** Turn on the Plug in Keyboard Shortcut toggle next to the 3DS MAX prompt line and Press ALT+R to change the Active Time to the animation length.


12. On the transition editor dialog click Optimize Transition in the upper right hand corner.
13. On the Transition Optimization dialog turn on Search Near Existing Transition. Click OK.

Play the animation. The automatic optimized transition looks good also. Optimized transitions are a time saver if there are many clips and transitions that need to be processed.

**Tip:** You can select multiple transitions in the Motion Flow Graph window and use Optimize Selected Transitions in the Motion Flow Graph toolbar to optimize all the transitions at once. Optimized transitions can take time to compute but are high quality.

### Making the animation available outside Motion Flow mode

To make the animation available in your scene outside Motion Flow mode, use the Create Unified Motion command.

1.  On the Motion Flow Script rollout click Create Unified Motion.
  2. Exit Motion Flow Mode.
- The motion is available as a freeform animation in your scene.







Load *MoFlow\_tut02.max* to see the animation. Note that a unified motion does not contain footsteps.

## Lesson 3: Looping Animation in Motion Flow Mode

By creating a script that calls the same motion clip repeatedly you can create a loop cycle that lengthens the motion. This is a good way to lengthen a walk or run cycle. You will use layers to change the looped animation.


### Turn on Motion Flow Mode

1. Reset 3D Studio MAX.
2. Create a biped
3.  Turn on Motion Flow mode.
4.  On the Motion Flow rollout, click Show Graph.  
The Motion Flow Graph displays.
5.  On the Motion Flow Graph toolbar, click Create Multiple Clips.
6. Click Browse and load *cstudio\tutorials\tutorial\_5\Walk2Loop.bip*.  
The motion clip appears in the Motion Flow Graph dialog.
7.  On the Motion Flow Graph toolbar, click Synthesize Motion Flow Graph.  
A loop arrow appears on the clip.




8.  On the Motion Flow Graph toolbar turn on Select Clip/Transition.

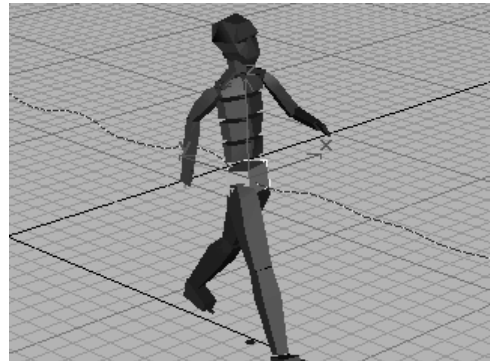
9. Select the transition arrow in the Motion Flow Graph window.

10.  On the Motion Flow Graph toolbar click Optimize Selected Transitions.  
The transition is optimized.



### Creating a script

1.  On the Motion Flow Script rollout, turn on Define Script.
2. On the Motion Flow Graph dialog, click the *Walk2Loop* clip 5 times.  
You've created a script that calls for the clip to transition to itself four times.
3. Click Play.


The clip loops, extending the walk cycle.

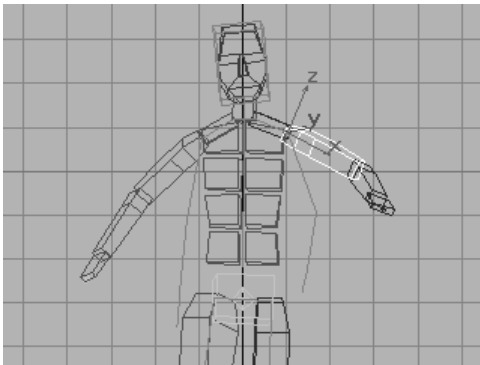



### Making the motion available in your scene outside of Motion Flow Mode

1.  On the Motion Flow Script rollout, click Create Unified Motion.
2.  On the General rollout, turn off Motion Flow mode.  
The walk cycle is available for editing outside of Motion Flow mode.

### Adding a layer and changing the walk cycle

1.  On the Layers rollout, click Create Layer.  
A new layer is created.
2. Turn on Animate.
3. At frame 0, in the viewports, rotate both upper arms about the Y axis to move the arms away from the body.
4. Click Play.  
The walk loop now has the character swinging his arms far from his body. The red stick figure represents the original motion.



5.  On the Layers rollout, click Collapse.  
The layer showing the arms away from the body is collapsed back to the base animation. Click Play to see the animation. Load *MoFlow\_tut03.max* to see the finished animation.

Looping animation and layers lets you lengthen and edit animations. To reduce keys and create footsteps, save the *.bip* file and use Load Motion Capture File in the Motion Capture rollout to reload the file. Options in the Motion Capture Conversion Parameters dialog let you reduce keys and extract footsteps. The Convert command on the General rollout will not

extract footsteps because the feet have no IK Constraints applied.

---

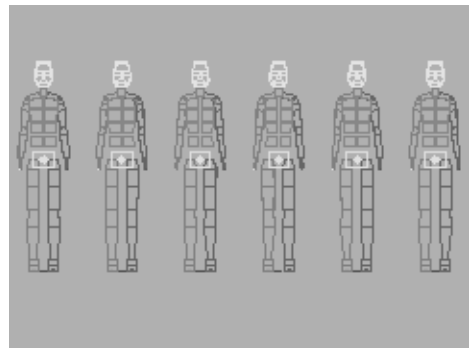
## Lesson 4: Using a Shared Motion Flow

By including multiple bipeds in a shared motion flow you can animate multiple bipeds using only one motion flow graph. The Create Random Motion command randomly picks clips to create a unique script for each biped. Use this to create a crowd that is loitering or cheering, for example.






### Creating multiple bipeds

1. Reset 3D Studio MAX.
2. Create one biped at the far left of the Front Viewport.
3. In the Front Viewport, select the center of mass.
4. In the Front Viewport, press SHIFT and drag to the right to create a copy of the biped. In the Clone Options dialog, type 5 for the number of copies.



Six bipeds appear standing shoulder to shoulder.

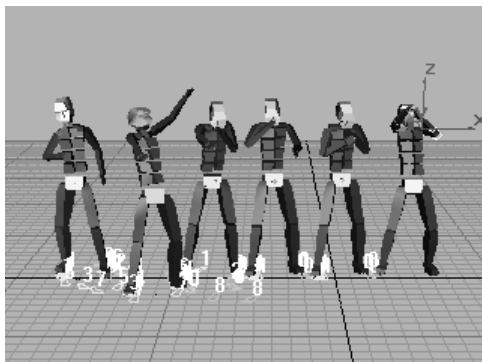


### Creating a shared Motion Flow

1.  Select the biped at the far right of the Front Viewport and turn on Motion Flow mode.  
Select one one object on the biped to enable biped controls in the Motion panel.
2.  On the Motion Flow rollout, click Shared Motion Flow.  
The Shared Motion Flows dialog is displayed.
3. On the Shared Motion Flows dialog, click New.  
A new shared motion flow is created. A default name appears in the list.
4. On the Shared Motion Flows dialog, click Add.  
The Select dialog is displayed.
5. Select all the bipeds in the Select dialog and click Select.  
On the Shared Motion Flows dialog the biped names are displayed in the window.
6. On the Shared Motion Flows dialog, click OK.  
The Shared Motion Flow icon on the Motion Flow rollout now has a white circle around it. This means that the selected biped is part of a shared motion flow.
7.  On the Motion Flow rollout, click Show Graph.  
The Motion Flow Graph dialog is displayed. The word “shared” appears in the dialog name. This Motion Flow Graph will be shared by the six bipeds.
8.  On the Motion Flow Graph toolbar, click Create Multiple Clips.
9. From *cstudio\tutorials\tutorial\_5* add *hatsoff.bip*, *laugh.bip*, *shocked.bip*, *sneeze.bip*, and *wave.bip*. Hold down the CTRL key and click each file name to add it to the selection. Click OK.  
The selected clips are added to the Motion Flow Graph.
10.  On the Motion Flow Graph toolbar, click Synthesize Motion Flow Graph.  
Transitions appear between all the clips.



11.  On the Motion Flow Graph toolbar, click Select Random Start Clip.
12. Select all the clips in the Motion Flow Graph window by dragging a window around them.  
Each clip has an even chance of being the first clip in a script.
13.  On the Motion Flow Script rollout, click Create Random Motion.  
The Create Random Motion dialog is displayed.
14. On the Create Random Motion dialog set Random Start Range from 0 to 30.  
If two bipeds share a clip the clips start frame is picked randomly. This prevents identical motion.
15. On the Create Random Motion dialog, click “Create motion for all bipeds sharing this motion flow,” and then click Create.



An alert asks if you want to put all the bipeds into Motion Flow mode. Click Yes. Each biped has a random script based on the clips in the Motion Flow Graph. Click Play to see the animation.


**Tip:** Use the “Put bipeds into motion flow mode” command on the Shared Motion Flows dialog to put multiple bipeds into Motion Flow mode.

## Lesson 5: Using the Create Random Motion Feature

This lesson illustrates how to use the Create Random Motion feature with clips in the Motion Flow Graph to create interesting scenarios quickly. It uses an available set of *.bip* motion files.


You'll add four skating motions to the Motion Flow Graph, create a network of transitions, and generate a random script to animate the biped.

### Creating a biped


1. Reset 3D Studio MAX.
2. Create a biped.
3.  On the Motion panel, on the General rollout, turn on Motion Flow mode.

4.  On the Motion Flow rollout, click Show Graph.


The Motion Flow Graph displays.

5.  On the Motion Flow Graph toolbar, click Create Multiple Clips.
6. From the *cstudio\tutorials\tutorial\_5* directory, select the following four motion files: *skateup.bip*, *skatestart.bip*, *skateloop.bip* and *skatespin.bip*. Click OK.

The motion files load into the Motion Flow Graph.

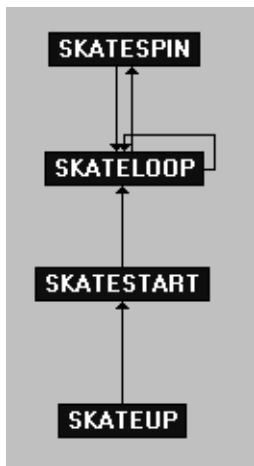
7.  On the Motion Flow Graph toolbar, turn on Move Clip.
8. Arrange the clips in a column with *skateup* at the bottom, then *skatestart*, *skateloop*, and *skatespin*.




9.  On the Motion Flow Graph toolbar, turn on Create Transition.
10. In the Motion Flow Graph window, click and drag from one clip to the next to create transitions between the clips. Start with


*skateup* and drag to *skatestart*, then *skateloop* and finally *skatespin*.

Use the illustration as a guide to create the transitions. To create a loop, click a clip twice.



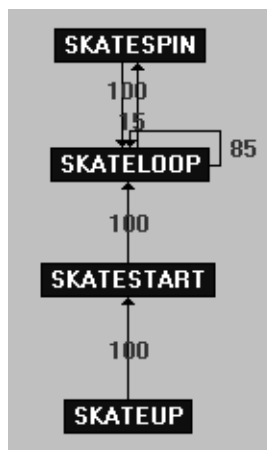
11.  Turn on Select Clip/Transition and region select all the transitions, and then click Optimize Selected Transitions on the Motion Flow Graph toolbar.



Optimize creates high quality transitions.

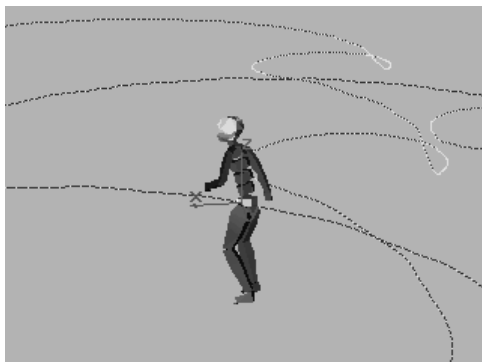
12.  On the Motion Flow Graph toolbar, turn on Show Random Percentages. A percentage number displays next to each transition. You'll alter the percentage for the loop transition and the transition between *skateloop* and *skatespin*.

13. In the Motion Flow Graph window, right-click the loop transition on the *skateloop* clip.
14. On the upper left of the Transition Editor dialog enter 85 in the Probability field. Click OK.

15. In the Motion Flow Graph window, right-click the transition from *skateloop* to *skatespin*. On the Transition Editor dialog, enter 15 in the Probability field. Click OK. When a random script is generated *skateloop* will be chosen over *skatespin* 85 percent of the time. The percentage numbers in the Motion Flow Graph are normalized, so if two transitions from a clip are set at 100, each one has an even chance of being selected.

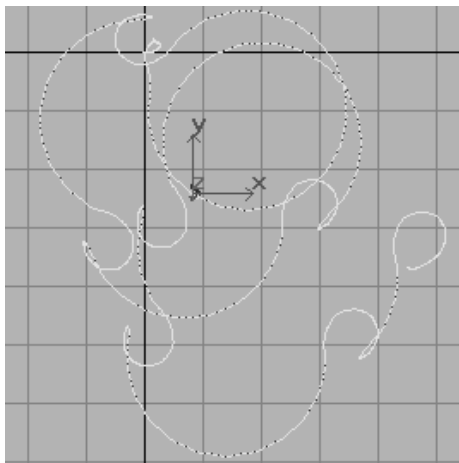


16.  On the Motion Flow Graph toolbar, turn on Select Random Start Clips, and click *skateup*. The *skateup* clip will always be the first clip to play when a random script is created.
17. Close the Motion Flow Graph window.
18.  On the Motion Flow Script rollout, click Create Random Motion.
19. In the Minimum Animation Length field, enter 2000 and click Create.



A random script is generated. Click Play to view the results. The skater gets up off the ice and begins skating. Once the script reaches the *skateloop* clip it loops back to *skateloop* 85 percent of the time and *skatespin* 15 percent of the time.

Select the biped Center of Mass and turn on Trajectories on the Display rollout, to get a birds eye view of the entire motion.



---

## Tutorial 6

# Working with Motion Capture Data

Three types of motion capture data are included with **character studio 3**:

- In the *cstudio\motions\mocap* directory, raw motion capture files have keys at every frame and do not contain footsteps.
- In the *cstudio\motions\footstep* directory, the raw data has been converted: it has footsteps and has been filtered for key reduction.
- In the *cstudio\motions\freeform* directory, the raw data has been filtered for key reduction but it contains no footsteps. It is freeform animation.

Use Load Motion Capture File on the Motion Capture rollout to try your own filter settings on the raw motion capture data supplied with the product. Use it to filter and import motion capture data that you purchase or acquire and to loop any *.bip* file. Filter settings allow you to convert a footstep animation into a freeform animation and back again.

The supplied raw motion capture data is very similar to a BVH format: these files contain limb rotation information. You can also import motion capture marker (*.csm*) files, which contain marker position data. The extracts limb rotation data is extracted from the marker positions to create keys for the biped.

In this tutorial you'll do the following:

- Import motion capture data.
- Filter the data.
- Edit the data.

## Setup

Reset 3DS MAX. Files for this tutorial are *cstudio\tutorials\tutorial\_6* directory in your 3DS MAX path.

## Lessons

*Lesson 1: Importing Motion Capture Data (see page 536)*

*Lesson 2: Comparing Trajectories (see page 538)*

*Lesson 3: High Frequency Data and Looping (see page 539)*

*Lesson 4: Editing with Layers (see page 540)*

## Lesson 1: Importing Motion Capture Data

You can import a file either in its raw state with keys at every frame, or with key reduction filtering. You can edit raw motion capture data by moving footsteps in Footstep mode and adding layers to keyframe the upper body. In this lesson you will import a raw motion capture file and edit footsteps.

## Setup

### Editing a raw motion capture file

1. Create a biped.



2. On the Motion panel, on the Motion Capture rollout, click Load Motion Capture File.

3. Load *fashion1.csm*.

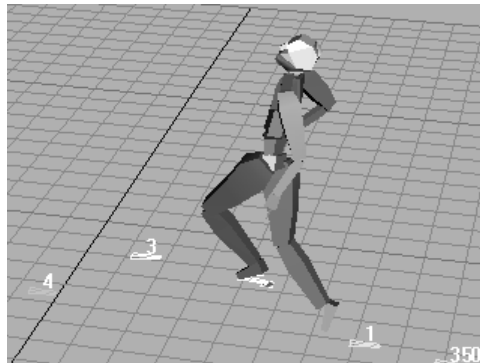
If this file does not appear, be sure the Files of Type drop down is set to *.csm*.

4. On the Motion Capture Conversion Parameters dialog choose On for Footstep Extraction and choose No Key Reduction in the Conversion field. Click OK.

The file loads with keys at every frame.

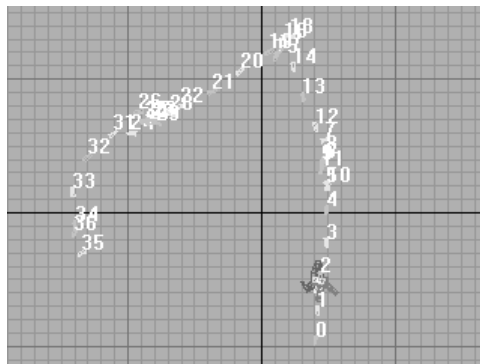


5. On the General rollout, turn on Footstep Mode.
6. Select footsteps 2 and 3 and move them up in the Front or Left viewport.






7. Click Play.
8. In the Top viewport, select footsteps 7 through 21.
9. On the Footstep Operations rollout, set Bend to a negative value.


The footstep path bends out. The biped adapts naturally to the footsteps.








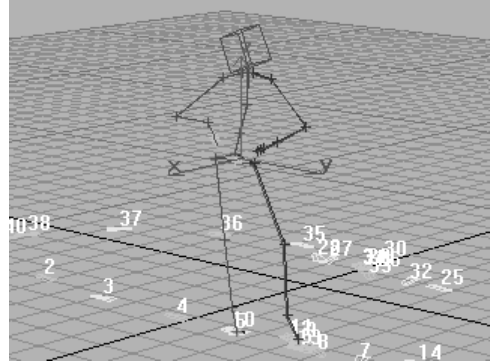
10.  Turn off Footstep mode.
11. Drag the time slider to frame 250.
12.  On the General rollout, turn on Bend Links mode.
13. Turn on the Animate button.
14.  On the Layers rollout, click Create Layer.
15. Rotate the spine so the character stands up straight rather than leaning back during the walk.
16. Click Play to view the changes, the character walks more upright.
17. On the Layers rollout click Collapse. This will collapse the spine editing down to the original animation. This will take some time.

### Filtering the data for key reduction

1.  On the Motion Capture rollout, click Convert from Buffer.
2. On the Motion Capture Conversion Parameters dialog, turn on Footstep Extraction and choose Use Key Reduction in the Conversion field. Click OK.  
The raw data in the motion capture buffer is filtered with key reduction. The motion capture buffer contains the raw data buffered when the file was first read in. This data is used to either try new filter settings quickly or to paste a pose from the raw information in the buffer onto a biped with filtered data. If you plan on major editing to the motion capture data then key reduction is the way to go.

### Comparing the raw and filtered data

1.  On the Motion Capture rollout, click Show Buffer.
2.   On the Display rollout, turn off Objects and turn on Bones.






The raw and filtered data match closely.

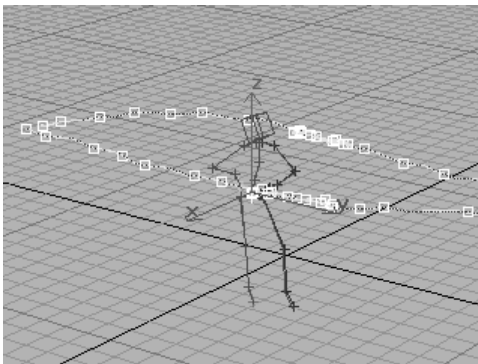
3. Click Play.  
The raw data represented by the red stick figure closely matches the filtered data represented by the other stick figure. This motion does not contain any high frequency data (fast motion) so it is not necessary to correct the filtered motion. Later in the tutorial you'll filter a file with high frequency information and add correction.
4. Save your work.

## Lesson 2: Comparing Trajectories

This lesson continues from the end of *Lesson 1* (see page 536).

### Setup

1.  On the Motion Capture rollout, click Show Buffer Trajectories.
2.  On the Display rollout, click Trajectories.
3.  On the Display rollout, click Display Preferences. In the Trajectories group, select Bone Base.



4. Close the Display Preferences dialog.
5. Select various parts of the biped to compare filtered and raw trajectories for each biped body part.
6. On the Display rollout, click Display Preferences. In the dialog's Trajectories area, turn off Show Entire Trajectory. Now the trajectories display only 50 frames before and after the current frame.

The trajectories between the buffered and filtered data are very close. Create your own

directory for data that you will filter. Save this animation as a *.bip* file, if you like.

The default filter settings work well in many cases, but information may be lost in motion capture files that contain a lot of high-frequency motion. There are a number of ways to work with such files. One is to filter the file normally, as you did in this tutorial, and paste the missing information using Paste from Buffer on the Motion Capture rollout.

You can also change Tolerance and Key Spacing parameters for less key reduction, or import a motion capture file with no key reduction and add a layer on the Layers rollout. This is an easy way to make changes to a file that contains keys at every frame.

### Using Calibration Controls

Motion Capture calibration controls (the bottom row of buttons on the Motion Capture rollout) are active only if a raw motion capture file is imported with no key reduction. They are used primarily to calibrate marker (*.csm*) files, although they work with all raw motion capture data.

Marker files represent raw marker position data directly from a motion capture device, and typically require more work than the *.bvh* or *.bip* motion capture files that ship with **character studio**.

### Using Marker Files




You can manage biped size and limb position relative to the markers with the two main calibration functions on the Motion Capture rollout: Talent Figure Mode and Adjust Talent Pose. The size and limb offsets are saved as *.fig* and *.cal* files, respectively. These files, as well as a marker name (*.mmm*) file, can be loaded prior to importing a motion capture file to apply the

size and limb offsets to marker files requiring the same adjustment.

## Lesson 3: Using High-Frequency Data and Looping

With high-frequency motion, such as sprinting, you can lose the nuances of the motion if you use the default filter settings. In this lesson, you will minimize key reduction on the legs of a sprinter so the legs follow the raw motion data more closely. You will also learn to loop a motion file in the Motion Capture Parameters dialog.

### Loading a file

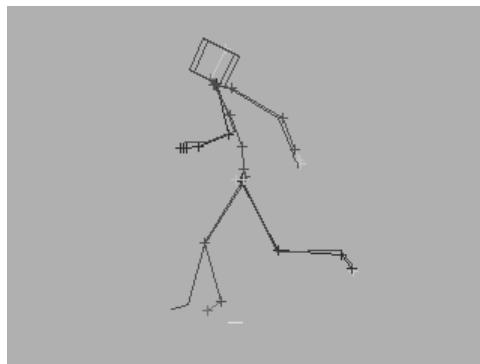
1. Reset 3DS MAX.
2. In the Front viewport, create a biped.
3.  On the Motion panel, on the Motion Capture rollout, click Load Motion Capture File. Choose *.csm* as the file type.
4. Open *jez.sprint.csm*.  
The Motion Capture Parameters dialog is displayed.
5. Choose Footstep Extraction and Use Key Reduction.
6. Click OK.  
The clip loads but has no footsteps.
7.  On the Display rollout, turn on Bones and turn off Objects.
8.  On the Motion Capture rollout, turn on Show Buffer.  
The red stick figure represents the raw motion capture data. The other stick figure represents the filtered data. Note that you can't use In Place mode with Show Buffer.
9. Maximize the Left viewport.

### Comparing the data


1. Click Play.

The filtered data lags behind the raw data in the feet, making the sprint look more like a fast shuffle.

**Tip:** Right-click the Play button and turn on Real Time and 1 / 4x Speed in the Time Configuration dialog to slow down the motion. Key bracketing can confuse the view. You can turn off key bracketing by choosing Customize > Preferences > Animation > Key Bracket Display group > None.

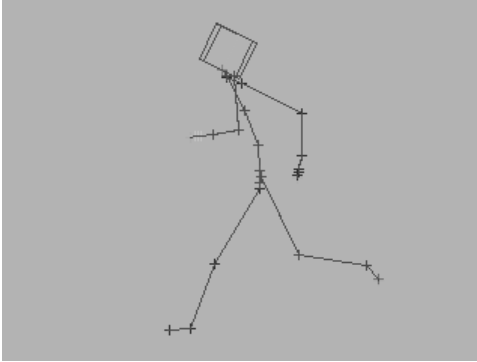


The filtered data lags behind the unfiltered data.

2.  On the Motion Capture rollout, click Convert from Buffer.  
The Motion Capture Buffer contains raw motion capture data. You'll enter parameters to correct the lagging and to extract footsteps.  
Footstep Extraction and Use Key Reduction should both be chosen.
3. In the Footstep Extraction area set Extraction Tolerance to 1.23.


4. In the Key Reduction Settings group change the Minimum Key Spacing for both legs to 1. Click OK.

The filtered legs match the raw legs during the sprint. The legs have a key at almost every frame, but the rest of the biped has key reduction.



The filtered data is identical to the unfiltered data.


### Looping

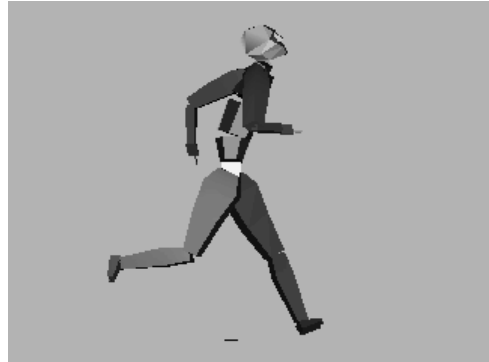
1.  On the Motion Capture rollout, click Load Motion Capture File. Choose *.csm* as the file type.
2. Open *jez.sprint.csm*.  
The Motion Capture Parameters dialog is displayed. All the settings on the Motion Capture Parameters dialog should be the ones you entered previously.
3. In the Load Frames group, turn on Loop and enter 1 in the Loop field. Click OK.  
The sprinting motion is looped, creating a longer sprinting period. You can also loop a motion in the Motion Flow Graph with similar results.

## Lesson 4: Editing with Layers

You'll import a file and use layers to add changes. The motion file is a sprinting motion. You'll add a layer and rotate the center of mass so the character is running on a wall.


### Loading in the motion

1. Create a biped.
2.  On the Motion Capture rollout click Load Motion Capture File.
3. Load *jez.sprint.csm*.
4. On the Motion Capture Conversion Parameters dialog, click OK.  
No Footstep Extraction and No Key Reduction are both chosen. The motion file loads with keys at every frame.
5. Click Play to see the running motion.




### Adding a layer


The following steps illustrate the use of Snap Set key to alter the runner's original position.

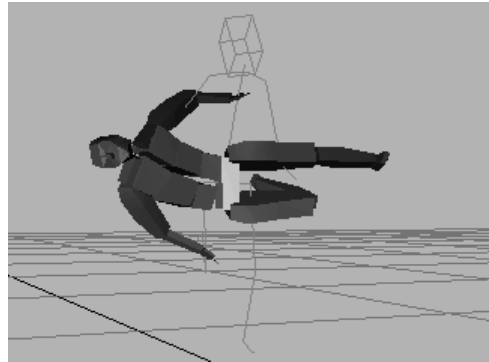
1.  On the Layers rollout, click Create Layer.  
A layer is created.

At frame 8, the character's left foot is just touching down.

2. Turn on Animate.
3. On the Track Selection rollout, click Body Rotation.  
This selects the center of mass rotation track.
4. In a viewport, click and drag to rotate the center of mass **-90** degrees about the X axis.
5. Click Play.  
The character is running on an invisible wall.
6. Move the time slider to frame 0.
7.  On the Layers rollout click Snap Set Key.  
The character snaps to his original position.

8. Move the time slider to frame 16.  
The right foot is just touching down.

9.  On the Layers rollout, click Snap Set Key.  
The character snaps to his original position.
10. Click Play to view the animation.  
The character runs on the ground, then on the wall, and back to the ground again.



To improve the effect you can rotate the spine objects and stretch the legs to make it appear that the character is jumping on the wall.



---

## Tutorial 7

# Skinning and Linking with Physique

---

## Skinning with Physique



A skinned biped is a biped figure that has been aligned to a mesh model with a Physique modifier applied to it. Once the modifier is applied to the mesh and attached to the biped, the mesh follows the movement of the underlying biped skeleton.

In this tutorial, you'll do the following:

- Attach a mesh to the biped
- Adjust envelopes to correct mesh distortion
- Create a bulge in Sub-Object Bulge
- Scale a character
- Use linking rather than Physique to attach a character to the biped

## Lessons

*Lesson 1: Aligning a Biped to the Mesh Model (see page 544)*

*Lesson 2: Applying and Adjusting the Physique Modifier (see page 548)*

*Lesson 3: Adjusting Envelopes and Weighted Vertices (see page 549)*

*Lesson 4: Possible Problem Areas: Shoulders and Pelvis (see page 551)*

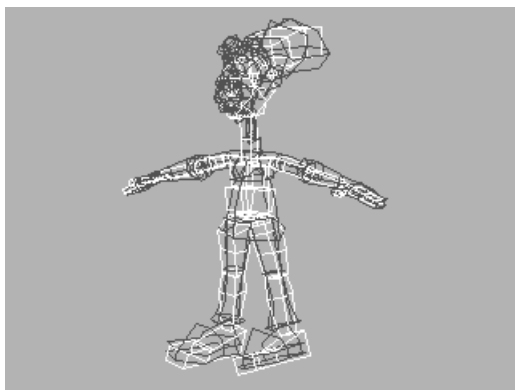
*Lesson 5: Animating Muscles with the Bulge Editor (see page 552)*

*Lesson 6: Scaling Characters with Physique (see page 553)*

*Lesson 7: Linking to the Biped (see page 554)*

---

## Lesson 1: Aligning a Biped to the Mesh Model



The character you'll work with is a patch character. Patch modeling is a popular alternative to mesh and NURBS modeling. Before aligning the biped to the mesh (patch) you must setup the structure of the biped.

## Setup

Files for this tutorial are in the `cstudio\tutorials\tutorial_7` directory under your 3DS MAX directory.

### Loading a character

1. Load the file `Tut_DrX_01.max`.
2. Select the mesh in the viewports.
3. On the Display panel, on the Freeze rollout, click Freeze Selected.

This freezes the mesh so you can see it but you can't alter it by accident.

### Creating the Biped

1. On the Create > Systems panel, click Biped.
2. In the Front Viewport, center the cursor between the feet of the mesh and drag up until the biped's bounding box is roughly the same height as the mesh.

When you release the cursor, the program creates a default biped.

3. On the Motion panel, on the General rollout, turn on Figure mode.

In Figure mode you can change the biped structure and align it to the mesh.

4. On the Structure rollout, set fingers to 2 and finger links to 2.

All you need is a thumb and one large finger to bend the glove.

5. Set toes to 1 and toe links to 1.

The character is wearing shoes and needs only one toe to bend the shoe when he walks.

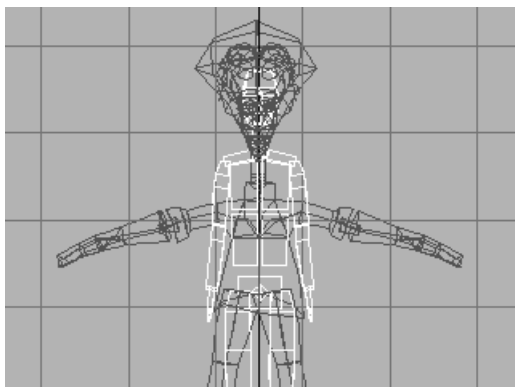
6. Set Ponytail 1 links to 3.

The character has a large hat that you can animate using the ponytail links.

7. Set spine links to 2 and neck links to 2.



## Aligning the biped to the mesh

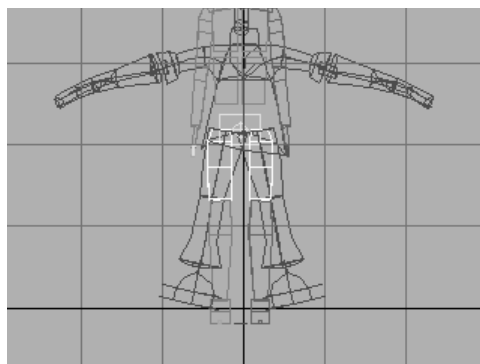


**Center of mass lined up to the mesh pelvis.**

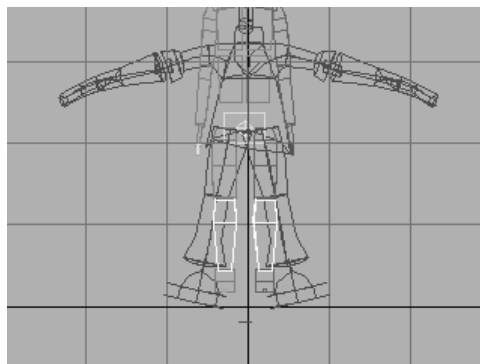
8. With Figure mode on, select the biped's center of mass and use Select and Move on the 3D Studio MAX toolbar to align it to the pelvis on the character.
9. Scale the biped's pelvis so the biped's leg links are centered in each leg of the mesh.

### Fitting the biped legs to the mesh

1. Select one of the thigh objects on the biped.
2. On the Motion panel, Track Selection rollout, click Symmetrical Tracks.  
This selects the opposite thigh link as well, ensuring that transforms done to one thigh will simultaneously apply to the other.
3. Turn on Select and Non-Uniform Scale on the 3D Studio MAX toolbar and scale the thigh links about the X axis so they end at the knees of the mesh.

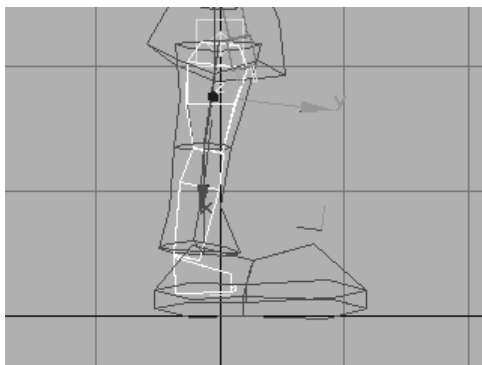


4. Press PAGE DOWN to select Bip01 L Calf and Bip01 R Calf.  
Using PAGE UP or PAGE DOWN highlights symmetrical objects in a biped when two symmetrical objects are selected. PAGE DOWN also highlights all child objects when you move down from a branching node. For example the pelvis is parent to both thighs and the lower spine.
5. With the two lower legs selected, use Non-Uniform Scale to scale the lower legs about the X axis until the lower ends of the links stop at the ankles.



6. In the Left Viewport, rotate the thighs and lower legs to better center their links in the legs of the mesh.

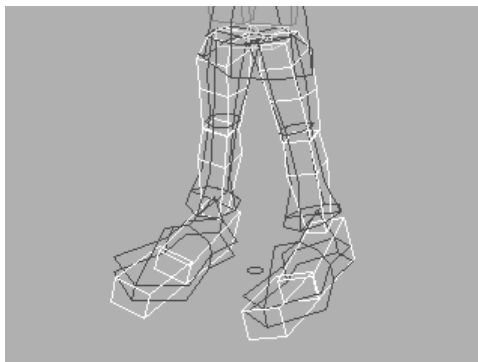
Use Symmetrical Tracks to select both thighs. Press PAGE DOWN to select the calves.



7. In the Front Viewport, rotate the thigh objects about the Y axis to spread the legs to fit the mesh.
8. Select both of the biped's feet in the viewports.
9. In the Left Viewport, non-uniformly scale the biped's feet so their profile roughly matches the profile of the mesh feet.
10. Select the toe of either foot and click Symmetrical Tracks.
11. Non-uniformly scale the toes so they are a bit longer and the same width as the mesh toes.

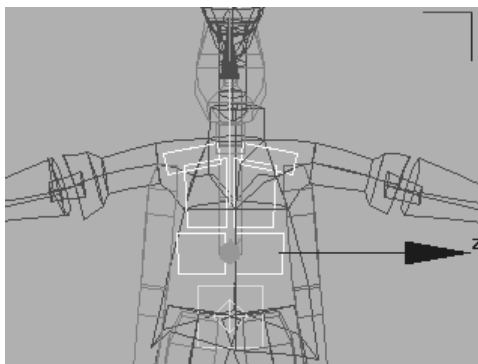
Physique can use biped limb scale to size the envelopes. Scaling the length and width of the toe will size the toe envelopes to approximate the biped toe objects.

12. In the Front Viewport, rotate both feet about the Y axis to line them up with the mesh feet.



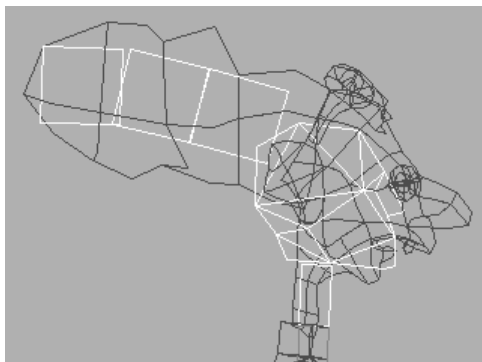
### Fitting the spine links to the mesh

- In the Front Viewport, non-uniformly scale the two spine objects to fit the mesh. The collar bones should be inside the mesh.



### Fitting the neck and head to the mesh

1. In the Left Viewport, select the neck links and scale the neck: place the chin of the biped head just above the chin of the mesh.
2. Scale the biped head (*Bip01 Head*) to fit the mesh.
3. Move, rotate, and scale the pony tail links to fit inside the hat.

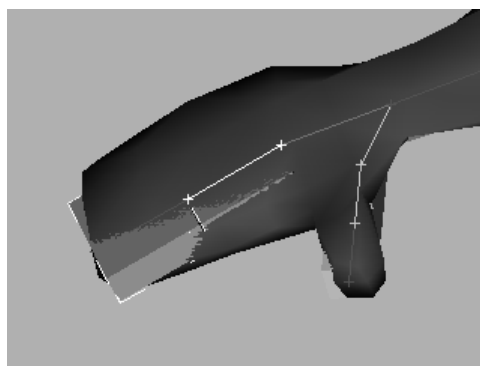
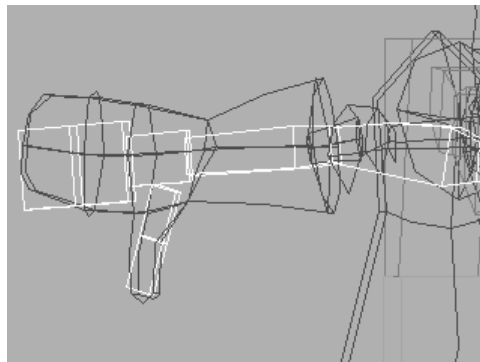


### Fitting biped arms and hands to the mesh

1. Select Bip01 L UpperArm.  
You can either select Symmetrical Tracks before you edit the arm positions, or pose one arm and then use Copy Posture and Paste Posture Opposite to make the other arm match the first one.
2. On the Track Selection rollout, click Symmetrical Tracks.
3. In the Front Viewport, rotate the upper arms about the Y axis so that they match the mesh.
4. In the Top Viewport, rotate the arms about the Z axis to fit the arms to the mesh.
5. Use Non-Uniform Scale to scale the arm objects to fit at the elbow and wrist on the mesh.
6. Scale and move the palm and fingers of the right hand to match the mesh. You can move the fingers without affecting the palm. If you move the palm however, the biped forearm moves also.

Use a shaded view is to position the palm and fingers. When fingers are positioned properly, they may not line up with the biped palm object. This is normal. Turn on Bones on the Display rollout to display the links in the hand. These links show you

where Physique will create envelopes. It is more important to position each finger accurately than to worry about their alignment with the biped palm.



### Copying the right hand posture to the left hand

At this point, the biped right hand should be fitted to the mesh.

1. In the Top Viewport, region-select the biped palm and fingers of the right hand.
2. On the Keyframing rollout, click Copy Posture.
3. On the Keyframing rollout, click Paste Posture Opposite.

All the work you did on the right hand is pasted to the left hand. The biped left hand now fits inside the mesh.

Note: Character models are often symmetrical. For unsymmetrical characters, use the Copy and Paste procedure described above to get a general fit for the hand; you may need to fine tune finger and palm position and scale.

4. Save your work as a *.max* file.
5. With Figure mode active, use Save on the General rollout to save a figure (*.fig*) file as well. Name it *MyDrX.fig*

Use a figure file if you move the biped in Figure mode by accident, or if you need to fit another character with similar proportions.

Note: On the General rollout, the Save option saves *.fig* files only if Figure mode is active.

In the next lesson, Physique is applied to the mesh object. You'll use Attach to Node to attach the mesh objects to the biped.

---

## Lesson 2: Applying and Adjusting the Physique Modifier

The next step is to apply the Physique modifier. Leave Figure mode turned on when you apply the Physique modifier.

**Tip:** It is best to decide on structural details before using the Physique modifier to attach a mesh to the biped. To change the scale or structure of the biped after Physique is applied you must reinitialize in Physique to account for changes to the biped scale and structure.

### Setup

- Continue where you left off with *Tut\_DrX\_01.max*, or open *Tut\_DrX\_02.max*. Files for this tutorial are in the *cstudio\tutorials\tutorial\_7* directory under your 3DS MAX directory.

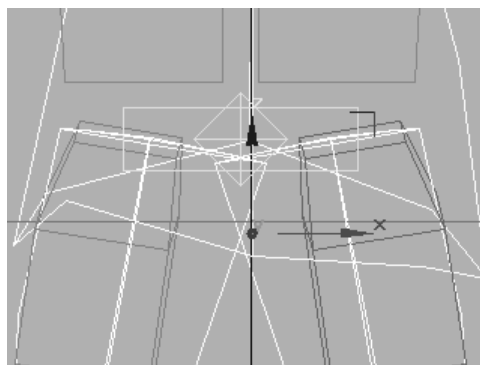
### Applying Physique

The biped should already be in Figure mode.

1. Select the character mesh in the viewports.
2. On the Modify panel, on the Modifiers rollout, click More.

The Modifiers dialog displays.

3. Choose Physique from the list of modifiers and click OK.
4. Maximize the Front Viewport.
5. Zoom into the pelvis area.



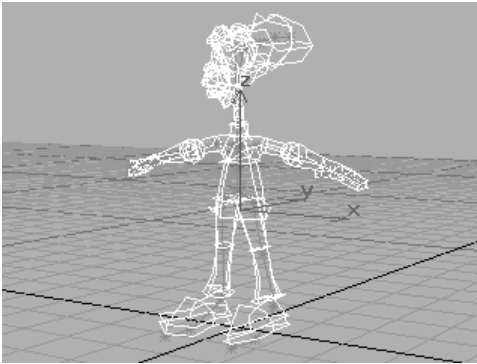
6. On the Modify panel, on the Physique rollout, click Attach to Node.
7. When the cursor is positioned over the pelvis, a white cross appears. Click the pelvis object.

The Physique Initialization dialog is displayed.

Note: The biped's pelvis object is displayed as a pale orange horizontal rectangle. If you accidentally pick the center of mass the software will automatically pick the pelvis.

8. On the Vertex-Link Assignment rollout in the Initialization dialog, check that the following default options are selected:
  - Deformable (specifies deformable envelopes for every link)

- N Links (specifies that all overlapping envelopes affect a vertex)
  - Create Envelopes (creates envelopes for the links)
  - Object Bounding Box (uses biped limb scale to size the envelopes)
9. Click Initialize at the bottom of the dialog. After a pause, a series of orange Physique links appear. These links make up the Physique Deformation Spline. Each link has an associated envelope that influences the vertices on the mesh to closely follow the biped skeleton. The mesh is now attached to the biped.



10. Save your work as a *.max* file.

In the next lesson, you will work on fine-tuning envelopes so the vertices deform in a predictable manner.

## Lesson 3: Adjusting Envelopes and Weighted Vertices

The bulk of the work in Physique is adjusting the size and overlap of all the envelopes to fine-tune mesh behavior as the character moves. Each area of the mesh requires some thought.

- The head should have little deformation, so assign it a rigid envelope.

- The three pelvis envelopes must be adjusted for proper deformation.
- The arm and shoulder envelopes must cover the armpits.

As you adjust the envelopes, scrub the time slider and rotate a User viewport. Moving a character reveals different flaws in the mesh as the biped posture changes in the animation.

**Tip:** While adjusting envelopes, you can use In Place mode on the Motion panel General rollout. In Place mode keeps the character in a viewport at any frame of the animation by limiting center of mass motion about the XY axes.

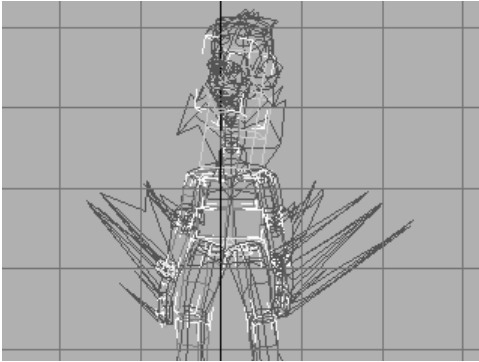
After attaching the mesh to the biped with Physique and loading the animation, parts of the mesh seem to stretch off the body. Vertices that are not within envelopes remain in their original positions and create stretched polygons. Roughly adjust the radial scale of the envelopes to pull in these vertices and then begin the process of fine-tuning the envelopes. For example, vertices on the mesh fingers may not be covered by envelopes creating the stretched polygons when animation is loaded.

### Setup

Continue where you left off with *Tut\_DrX\_02.max*, or load *Tut\_DrX\_03.max*. Files for this tutorial are in the *cstudio\tutorials\tutorial\_7* under your 3DS MAX directory.

### Loading a motion file

1. Select the biped.
2. On the Motion panel, on the General rollout, turn off Figure mode.
3. The character adopts an animated position. The envelopes need work.

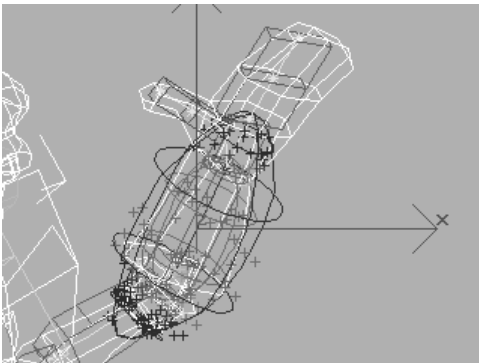


### Adjusting the envelopes

1. Select the mesh.
2. On the Modify panel, on the Physique Level of Detail rollout, turn on Hide Attached Nodes.

The biped disappears and you have a better view of the mesh.

3. Scrub the Time Slider to frame 27, the left hand is up in the air.
4. At frame 27 turn on Sub-Object Envelopes on the Physique modifier.
5. Enlarge the Front Viewport and zoom in on the left hand.



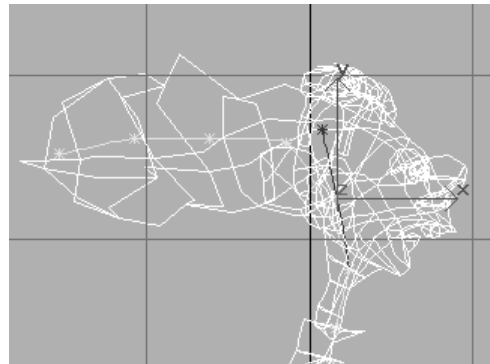
6. Select the forearm, palm, finger and thumb links in turn. Change the Radial Scale and

Parent\Child overlap until the distortions of the mesh disappear.

7. Scrub to frame 51, the right hand is in the air.
8. At frame 51 on the right arm, select the links of the forearm, hand, and fingers in turn. Change Radial Scale and Parent\Child overlap until the distortions of the mesh disappear.

Because of the ponytail link there are two links in the head. You must turn off the link that does not continue to the ponytails.

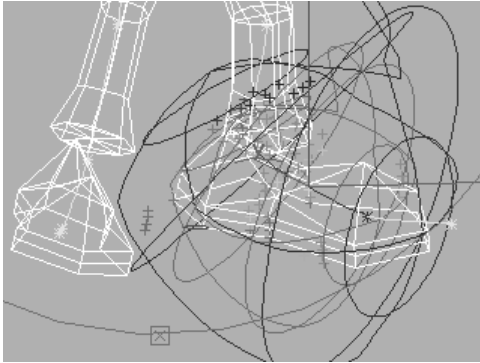
9. Select the link that does not continue to the pony tails, on the Blending Envelopes rollout in the Active Blending area, turn off Deformable.



10. Select the head link that continues to the ponytail and change it to a rigid envelope. Turn off Deformable and turn on Rigid.
11. Scale up the rigid envelope until the character's face and nose are not deformed. The face should not twist when the character rotates his head. This is the reason for using a rigid envelope in the head.
12. Scale and change the parent\child overlap of the ponytail links to smooth out this area.

13. At frame 51 adjust the envelopes of the feet so the shoes are not deformed.

If you must scale the feet envelopes to the point that one foot's envelope deforms the opposite foot, turn on Control Point in the Selection Level area and move the envelope control points to correct deformation.



**Tip:** Try turning on Initial Skeleton Pose in the Blending Envelopes rollout in Sub-Object Envelopes for a clear view of the envelopes. To see how much of the mesh is encompassed by an envelope, see if Initial Skeleton Pose is turned on.

14. Save your work as a *.max* file.

## Lesson 4: Fixing Possible Problem Areas: Shoulders and Pelvis

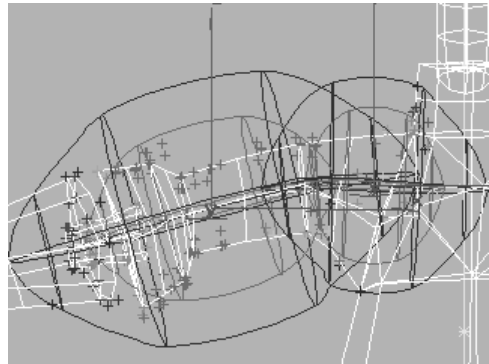
You may find problems in the mesh deformation of the shoulders, armpits, and pelvis. Slim characters are easier to set up than characters with large arms and thighs. Dr. X has slim arms and thighs, so the envelope setup is straightforward.

### Setup

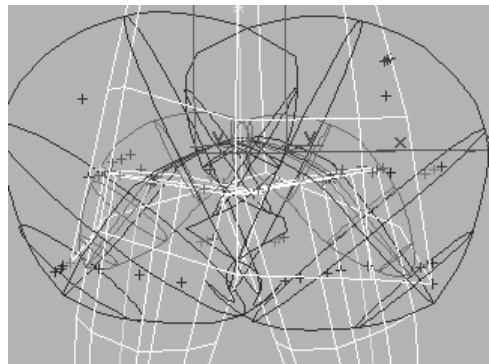
Load *Tut\_DrX\_04*. The character has animation. Use the time slider to see the mesh at different frames. Files for this tutorial are in the *cstudio\tutorials\tutorial\_7* under your 3DS MAX directory.

### Correcting envelope overlap

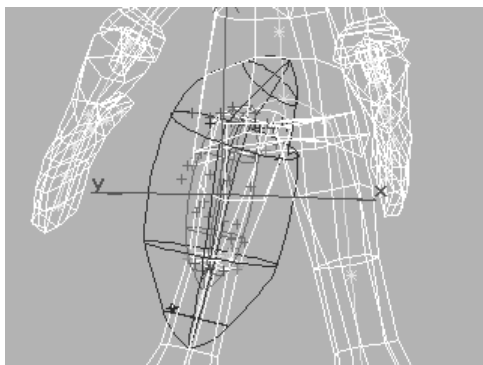
1. Make sure that the shoulder envelope encompasses the armpit area. The upperarm envelope should create a large overlap area with the shoulder envelope.



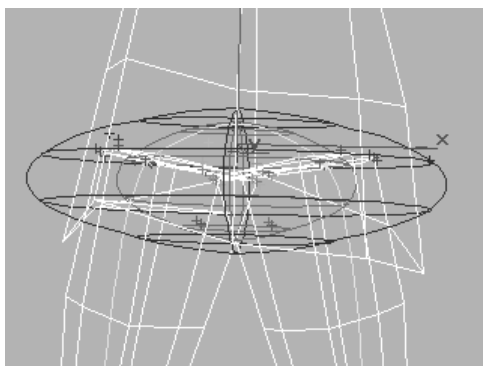
2. Make the links from the spine to the thighs large enough to cover the buttocks. Make sure the links have a significant overlap with the thigh envelopes.



3. Make the thigh envelopes large enough to encompass the buttocks area.



4. Turn off the third link in the pelvis if you like. If you find the crotch area is not deforming correctly, try enlarging this envelope.



## Lesson 5: Animating Muscles with the Bulge Editor

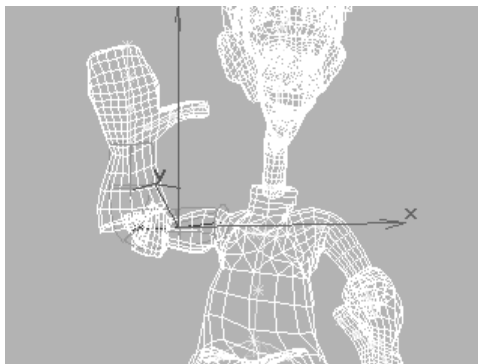
By adding bulge angles, you can make an arm bulge according to the angle of a joint. As an arm bends to a ninety-degree angle, biceps can be made to bulge for example. For this tutorial, you'll use a mesh version of the Dr. X character, rather than a patch version.

Files for this tutorial are in the *cstudio\tutorials\tutorial\_7* directory under your 3DS MAX directory.

### Creating a bulge angle

You'll be creating a biceps bulge in the character's right arm.

1. Load *Tut\_DrX\_Mesh.max*.
2. Scrub the time slider to frame 50.  
The right arm is at a 90-degree angle.
3. Select the mesh.
4. On the Modify panel, turn on Sub-Object Bulge.
5. Select the link on the right biceps.

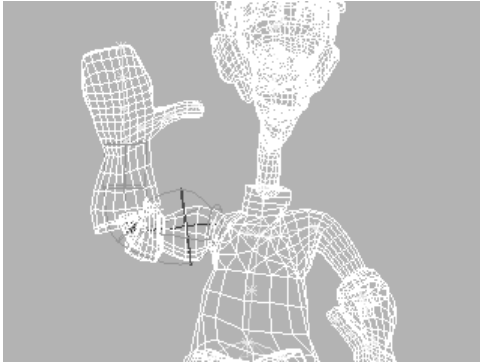


6. On the Bulge rollout, in the Bulge Angle Parameters area, click Insert Bulge Angle and then click Set Bulge Angle. In the Current Bulge Angle list, the bulge angle name increments. You've just created a new bulge angle at 90 degrees.  
Now you'll move the biceps cross-section to the middle of the arm.
7. On the Bulge rollout, in the Selection Level area, click Cross Section.
8. In the viewport, move the biceps-cross section to the middle of the biceps area.



Turn on Select and Move on the 3DS MAX toolbar and then click and drag to move the cross-section in the viewport.

9. Turn on Select and Uniform Scale on the 3DS MAX toolbar and scale the biceps cross-section in the viewport.



The mesh around the biceps area should bulge. If the mesh scales very little or not at all, increase Weight in the Bulge Angle Parameters area.

10. Scrub the time slider and examine the arm at different angles. When the arm is straight, there is no biceps bulge. As the arm bends to 90 degrees the biceps bulge.

There are many controls that allow you to fine-tune the bulge. You can use Insert to add more cross-sections and limit the bulge to a certain area. You can use Power to determine when the start of the bulge occurs. A high Power value will only start the bulge very close to the ninety-degree angle.

## Lesson 6: Scaling Characters with Physique

Files for this tutorial are in the *cstudio\tutorials\tutorial\_7* directory under your 3DS MAX directory.

1. Load *Tut\_DrX\_05.max*.

The biped is already in Figure mode. Use Figure mode when you scale a character.

2. Select the biped.
3. On the Motion panel, on the Structure rollout, change the height spinner to scale the character.

The biped and mesh scale together.

### To reinitialize the scaled mesh

This step is optional. Use these steps if you have to reinitialize for any reason.

1. Disable physique or drop down in the modifier stack below Physique.
2. Scale and move the mesh until it matches the biped or bones.
3. Re-enable or go back to Physique. The mesh will get large because it is doubly scaled.
4. Reinitialize with Initial Skeleton Pose. The mesh will shrink to the size you set during scaling.

## Lesson 7: Linking to the Biped

In certain cases, you want to link objects to the biped rather than using Physique to deform a mesh. The Link tool on the 3DS MAX toolbar is used to link objects directly to the biped. Objects that make up mechanical characters, a character in a suit of armor, or a space suit, and regular jointed characters are attached to the biped using the 3DS MAX Link tool.

Characters for sale often come in two varieties, jointed and jointless. A jointed character has separate objects with ball joint geometry for the limbs and lends itself to the linking technique described in this tutorial. Jointless characters have a seamless mesh at limb joints. Jointless characters should be attached to the biped using Physique.

The procedure to link the objects representing your character to the biped is straightforward. Load your character, create a biped, turn on Figure mode, align the biped to the character and use the Link tool on the 3DS MAX toolbar to link each object on your character to the corresponding limb on the biped. The thigh object on your character is linked to the biped thigh for example. Any animation applied to the biped will now animate your character.

In this tutorial, you learn the linking process using a simple human figure with a low resolution mesh for the body and limbs.

### Setup

Reset 3DS MAX. *.max* files for this tutorial are installed in the *cstudio\tutorial\tutorial\_7* directory under your 3DS MAX directory.

### Linking objects to the biped

Every mesh object except the arms are already linked.

1. Load *Joe2.max*.  
The biped is already in Figure mode.
2. Turn on Select and Link on the 3DS MAX toolbar.
3. In the Front Viewport, select one upperarm and click-drag to the biped arm. On mouse up the link is performed.
4. Select each arm object in turn and click+drag to the corresponding body part on the biped.
5. Select the biped, on the General rollout, turn off Figure mode.
6. Minimize the Front Viewport.
7. Click Play.

The character runs. All of the mesh objects follow the biped objects.

---

## Tutorial 8

# Working with Crowd Animation



This tutorial covers a range of techniques used with **character studio's** crowd animation features for creating and animating crowds. Crowd animation starts out with behaviors applied to delegates, which are stand-ins for crowd members. Next, that motion is applied to mesh objects or bipeds. These objects might carry their own animation, which can be combined with the delegate animation in a variety of ways.

In this tutorial, you'll learn how to do the following:

- Create and animate crowds.
- Apply behaviors to delegates.
- Apply crowd simulation to mesh objects and bipeds.

## Lessons

*Lesson1: Getting Started with Behaviors (see page 556)*

*Lesson 2: Using Multiple Delegates and Behaviors (see page 559)*

*Lesson 3: Applying Avoidance and Animating Behavior Assignments (see page 562)*

*Lesson 4: Applying Logic to Crowd Behavior (see page 564)*

*Lesson 5: Using Crowd with Animated Non-Biped Objects (see page 567)*


*Lesson 6: Creating a Crowd of Swimming Bipeds (see page 573)*

*Lesson 7: Advanced Crowd/Bipeds (see page 577)*

## Lesson 1: Getting Started with Behaviors

The Crowd object lets you assign behaviors to delegates. This lesson teaches you how to add these two objects and make them work together. You'll add a behavior and assign it to the smallest possible crowd: a single delegate.

### Creating a delegate

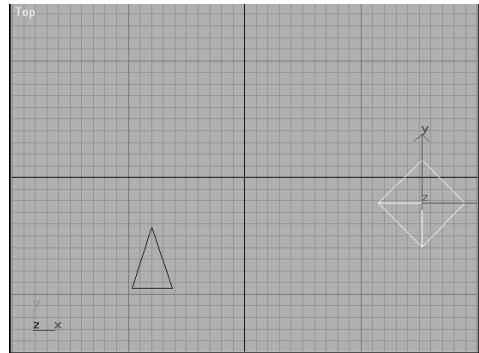
1.  On the Create panel, click Helpers.
2. In the Object Type rollout, click Delegate.
3. In the Top viewport, drag to add a Delegate helper object.

By default, the tip of the pyramid-shaped delegate object is its front; it helps to think of it as an arrowhead pointing in the direction it wants to go. Adding a delegate in the Top viewport points it forwards, in terms of the default world orientation.

### Creating a crowd object

The Crowd helper object is available from the same rollout.

1. In any viewport, add a crowd object.




**Delegate and Crowd helper objects in Top viewport**

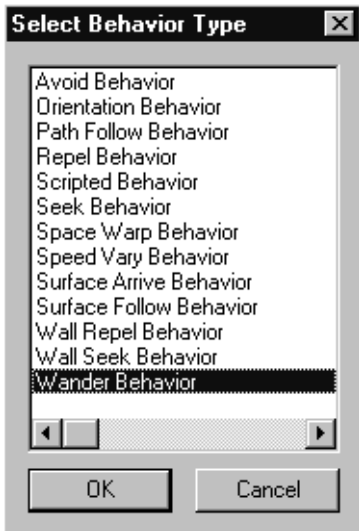
The size, position, and orientation of the crowd object are not important. Be sure to make it easily accessible, because you'll be selecting it frequently as you work with crowd simulation.

**Tip:** Create a selection set containing the crowd object to be able to select it quickly without using the viewports.

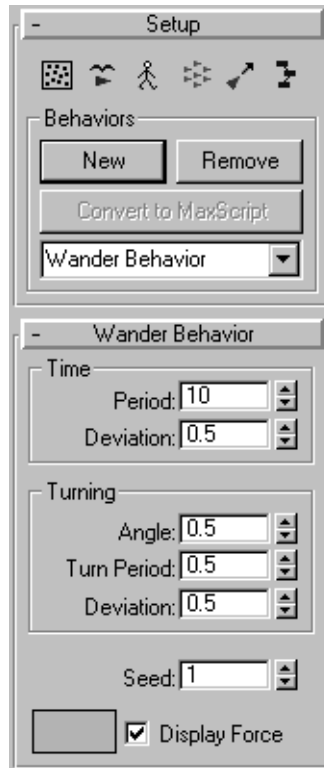
### Adding a behavior

You'll add a behavior and apply it to the delegate. The easiest behavior to use is the Wander behavior.

1.  Go to the Modify panel, which shows rollouts for the crowd object.
2. In the Setup rollout, click the New button.



3. In the Select Behavior Type dialog, choose the Wander behavior and click OK.



The default name of the behavior, Wander, appears in the list at the bottom of the Setup rollout, and a new Wander Behavior rollout appears in the command panel under the Setup rollout. You can find a full explanation of its parameters in the *Wander Behavior* (see page 416).

When you have multiple behaviors in a scene, this rollout shows the type and parameters for the current behavior as indicated in the list.

4. Change the name of the Wander behavior by clicking in the list box and typing a new name, such as **My Wander Behavior**.


**Tip:** When a scene contains multiple behaviors of the same type, give them

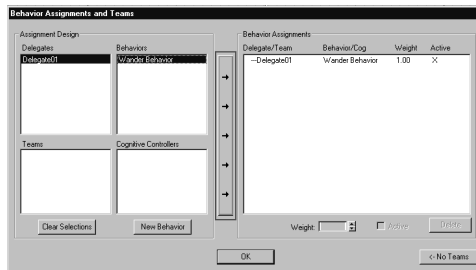
distinguishing names so you don't confuse them.

The name doesn't change in the behavior's rollout title. The rollout title always displays the name of the behavior type.

### Assigning the behavior

Next, you'll apply the behavior to the delegate.

1.  In the Setup rollout, click the Behavior Assignments button.  
This dialog lets you group delegates into teams and assign behaviors and cognitive controllers to delegates and teams. We'll cover the more advanced functionality in further lessons; meanwhile, you can find reference material on the dialog in *Behavior Assignments and Teams Dialog* (see page 375).
2. In the Behavior Assignments and Teams dialog > Assignment Design group, click the two list entries: Delegate01 and My Wander Behavior.



### The New Assignment button, outlined in red

When you select both a delegate and a behavior, the New Assignment button becomes available. This vertical button, situated between the Assignment Design and Behavior Assignments group, displays a column of right-pointing arrows.

3. Click the New Assignment button.

The assignment is added to the list in the Behavior Assignments group. By default, it has the highest available Weight value of 1.0 and is active.

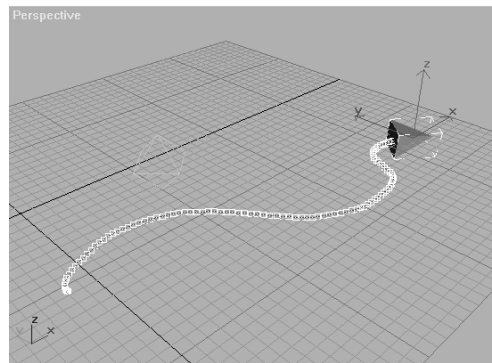
4. Close the dialog by clicking OK.

### Solving the simulation

You're now ready to solve the simulation.


1. In the Crowd command panel > Solve rollout, click the Solve button.

The software solves the simulation, while the bar graph under the Solve button shows its progress. As a result, the delegate wanders randomly around on the world grid.



### The wandering delegate's trajectory

**Tip:** You can see the delegate's trajectory by going to the Motion panel and clicking Trajectories.

2.  Play the animation.

You might need to change viewport settings to see the delegate throughout the animation.

3. Experiment with the behavior and delegate settings and continue to solve. Remember that you must have the crowd object selected to see the various crowd parameters.

Here are some things to try:

- Change the Period setting to adjust how long the delegate travels before it changes direction.
- Change Seed to vary the random path.
- Select the delegate, and in the Motion Parameters rollout, turn off Constrain to XY Plane. This lets the delegate wander in 3D space! Note that, to solve after you do this, you'll have to select the crowd object again.



**Tip:** To save time, turn on the Plug-in Keyboard Shortcut Toggle and then press the S key to run a solution when the Modify panel is active.

Now that you've learned the basic features of the crowd system, you can work with real crowds that contain more than a single delegate.

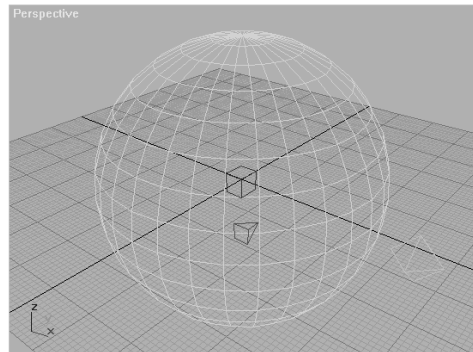
## Lesson 2: Using Multiple Delegates and Behaviors

**character studio** has a number of special facilities for working with multiple delegates and crowds, as you'll see in this lesson. You'll also explore several new behaviors.

This lesson builds on the knowledge gained in the previous one. If you haven't completed the *Getting Started with Crowds* (see page 556) lesson, please complete it before undertaking this one.

### Setup


1. Run 3DS MAX or reset the program, and then add a delegate and crowd object as in the first lesson.
2. Add a small box near the center of the workspace, then add a large sphere surrounding it. The two should be roughly concentric.



You can load a .max file that already has these objects. Open *cs\_tut08\_lesson2.max* from the *cstudio\tutorials\tutorial\_8* directory in your 3DS MAX path.

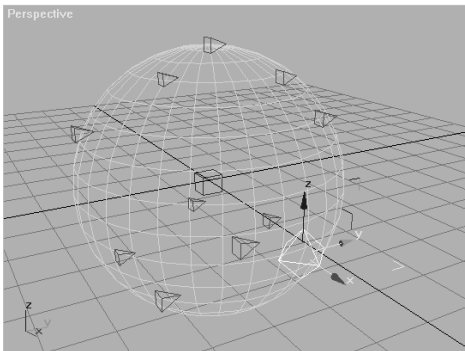
### Cloning delegates

You'll use the Scatter Objects tools to create several clones of the delegate, position them about the large sphere's surface, and aim them at the box.

1. Select the crowd object and go to the Modify panel.
2.  In the Setup rollout, click the Scatter button to open the Scatter Object dialog. This dialog is fully documented in *Scatter Objects Dialog* (see page 362).
3. In the Scatter Objects dialog > Clone tab, under Object to Clone, click the None button and select Delegate01. Click Select to close the dialog.
4. Change the How Many setting to 9.
5. Click Generate Clones.

This creates nine copies of the delegate (10 delegates in all). You can't see them yet because they're all coincident with the original delegate.

6. In the Scatter Objects dialog > Position tab > Placement Relative to Object group, choose On Surface.
7. In the same group, click the None button, and select Sphere01.  
Even though you're using a sphere to distribute the delegate clones, you're not choosing Inside Sphere. That's because you're going to position the clones on the surface of the sphere, rather than throughout its volume.
8. Click the Generate Locations button at the bottom of the Position panel.

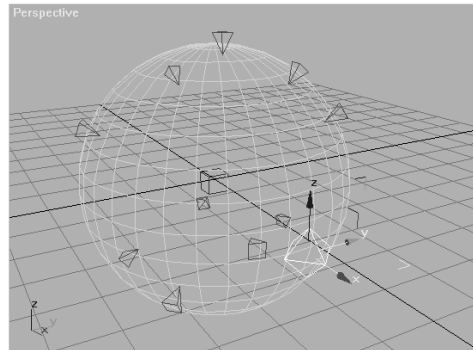


**Delegates are scattered about the sphere surface.**

### Reorienting delegates

The delegate clones are positioned at random locations across the large sphere's surface. They're all pointing in the same direction. In the next steps, you'll reorient them so they're all pointed at the small sphere.

1. In the Scatter Objects dialog > Rotation tab > Look At Target group, choose Selected Object.  
This makes the None button available.
2. In the same group, click the None button and select Box01.
3. Click the Generate Orientations button.




**Delegates are pointed at the box.**

The delegate clones are all now pointing at the box.

4. Click OK to close the Scatter Objects dialog.

### Adding behaviors

1. Select the large sphere and press the DELETE key to remove it from the scene.  
You'll use this setup in the fourth lesson.
2. Save the file as *cs\_tut08\_lesson4a.max*.  
Next you'll use two behaviors, Seek and Repel, to make half the delegates move toward the small box and the other half move away from it.
3. Select the crowd object and, in the Setup Rollout, add a Seek behavior.
4. In the Seek Behavior rollout, click the None button, and select the box.  
The box's name appears on the button, indicating that it is now the object that will be sought by any delegate to which the behavior is applied.
5.  In the Setup rollout, click Behavior Assignments.  
You could assign the Seek behavior one delegate at a time, but that's not necessary.



Crowd lets you group delegates into teams and assign a behavior to the entire team.

6. In the Behavior Assignments and Teams dialog > Teams group, click New Team. This adds a team with the default name Team0. You can rename it by clicking its name at the top of the group box.
7. Rename the team **Seek Team**. Next, you'll specify which delegates are to be members of Seek Team.
8. Click the Add Members button. In the Select Delegates dialog, click Delegate01, and then hold down SHIFT and click Delegate05 to select the first five delegates. Click OK to exit. The delegate names appear on the Seek Team roster.
9. In the Assignment Design group > Teams list, click Seek Team, and in the Behaviors list, click Seek Behavior.
10. Click the New Assignment button to create the assignment.

### Creating a behavior in the Behavior Assignments and Teams dialog

You'll create a Repel behavior in the Behavior Assignments and Teams dialog.

1. In the Behavior Assignments and Teams dialog > Assignment Design group, click the New Behavior button, and choose Repel Behavior. The behavior is added to the Behaviors list. You still need to use the command panel to modify the behavior. The dialog stays open as you do this.
2. In the Setup rollout, choose Repel Behavior from the list. The Repel Behavior rollout replaces the Seek Behavior rollout.

3. In the Repel Behavior rollout, specify Box01 as the repel object, using the same method as with the Seek behavior.

4. In the same rollout, in the Radius group, turn off Use Radii.


This applies the Repel behavior at any distance, not just within a specific radius.

5. Create a new team containing Delegate06 through Delegate10 and rename it **Repel Team**.
6. Assign the Repel behavior to Repel Team and close the dialog.
7. Solve the simulation.

Half the delegates move toward the box and the other half move away, but they all stay in their own plane. You can make them move in three dimensions.

If you tried the third suggestion at the end of the previous lesson, you might have an idea of how to fix this: Turn off the delegates' Constrain to XY Plane switches. But there's no need to do so one delegate at a time.

### Moving delegates in three dimensions


1.  In the Setup rollout, click Multiple Delegate Editing. The Edit Multiple Delegates dialog appears. You can use this dialog to apply changes to several delegates simultaneously.
2. In the Edit Multiple Delegates dialog > Delegates to Edit group, click the Add button.
3. In the Select dialog, click All and then Select. This adds all delegates in the scene to the Delegates to Edit list. You cannot work with teams in the Edit Multiple Delegates dialog. However, the

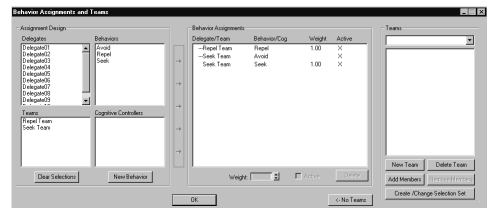
Behavior Assignments dialog lets you create named selection sets from teams, and the Select dialog lets you specify selection sets.

4. In the General group, turn off Constrain to XY Plane.
5. Turn on SET, the adjacent check box.  
This ensures that the changed setting takes effect when you exit the dialog.
6. Click the Apply Edit button.  
This closes the dialog and applies your change to all specified delegates.
7. Solve the simulation again.  
This time, the delegates move in three dimensions. Half of them move directly towards the box, and the other half turn and move directly away. The seeking delegates continue to seek the box after they reach it, continually circling it.
8. Save this file as **cs\_tut08\_lesson3a.max** for use in the next lesson.

2. Select the crowd object.
3. Add an Avoid behavior.

Like Seek and Repel, the *Avoid behavior* (see page 392) has a button for specifying a single object to avoid. But in this case, you want to specify several Avoid objects, so you use the Multiple Selection button.

4.  In the Avoid Behavior rollout, click Multiple Selection.
5. Highlight all objects (the delegates and the box) and click Select.
6. Use the Behavior Assignment dialog to apply the Avoid behavior to the Seek team.



When you click New Assignment, the Avoid behavior appears in the assignments list above the Seek behavior. That's because the list is sorted alphabetically, starting with the leftmost column. All behaviors are applied simultaneously, unless you animate their Active status, so when you solve, both the Seek and Avoid behaviors are applied to the Seek team at each frame.

7. Solve the simulation.

For the most part, the delegates now stay far away from each other and the target box.

## Lesson 3: Applying Avoidance and Animating Behavior Assignments

In the previous lesson, you might have noticed that the delegates penetrated each other when they were close to the box. In this lesson, you'll learn to use the Avoid behavior to prevent delegates from colliding. You'll also learn to apply a second "desired direction" behavior to the Seek team and animate the behaviors' influences.

### Adding the Avoid behavior

1. Load the file you saved at the end of the previous lesson. Or, if you like, load the file **cs\_tut08\_lesson3.max** from the **cstudio\tutorials\tutorial\_8** directory, in your 3DS MAX path.

### Using visual cues

You can get a better idea of how Avoid works by turning on its various feedback options. These give visual cues for distance-related settings and activities.

1. In the Avoid Behavior rollout, turn on Display Hard Radius and Display Repel Radius.  
Wireframe spheres appear around the delegates showing the respective radii.
2. Turn off Display Hard Radius and Display Repel Radius to avoid an overly complex display.
3. In the Avoid Behavior rollout > Display During Solve group, turn on Potential Collisions, Repel Activity, and Look Ahead Radius.
4. Solve again.  
During the solution, white lines appear indicating Potential Collisions detection, colored lines indicate Repel Activity calculations, and spheres indicate Look Ahead radii. The latter are available only during solution because the distance must be computed from the Look Ahead setting and the delegate's current velocity.

### Adjusting the behavior

The delegates give each other a wide berth because the Look Ahead setting, which tells delegates how many frames to look ahead for potential collisions before turning to avoid each other, is set to the fairly large default value of 30. The delegates will get closer to each other if you reduce this amount.

1. Set Look Ahead to 3.
2. Solve the simulation.  
This time the Seek Team delegates don't stay quite so far apart. In fact, you might see some interpenetration, because the small Look Ahead setting doesn't give them enough time to start turning to avoid each other. The best settings for Avoid, as with most other behaviors and Crowd

settings, depend on the situation, and finding them usually takes trial and error.

### Changing the behavior

Next, you'll make the delegates stop seeking the box and start moving away when they reach it. You can do this by assigning the existing Repel behavior to the Seek team and activating it at the proper frame while simultaneously deactivating the Seek behavior.

First, examine the animation to determine when to change the behaviors.

1. Move the time slider back and forth to find frame where the delegates come closest to the box.  
Choose frame 70, or a frame near frame 70.
2. Move the time slider back to frame 0 and turn on the Animate button.
3. Open the Behavior Assignments dialog and assign the Repel behavior to the Seek team.
4. In the Behavior Assignments list, click the Seek Team Repel Behavior item.

The corresponding items in the Assignment Design group are highlighted, and the controls at the bottom of the Behavior Assignments group become available.

5. At the bottom of the Behavior Assignments group, turn off Active.  
The X next to the assignment in the list disappears, indicating that the assignment is inactive. This setting is animatable, so you can directly control which assignments are active on a frame-by-frame basis throughout the animation.
6. Drag the time slider to frame 70.
7. Turn on Active for the Seek Team Repel Behavior assignment, and then choose the Seek Team Seek Behavior assignment, and turn off Active.

8. Drag the time slider back and forth to either side of frame 70 and watch the Behavior Assignments list.

At the same time that the Repel behavior turns on, the Seek behavior turns off.

9. Close the dialog and turn off Animate.
10. Solve the simulation.

At frame 70, the delegates stop seeking the box, turn abruptly, and begin to move away.

For a more gradual change, you can animate the Weight setting. This controls the relative influence of the behavior over the delegates, and ranges from 0 to 1.0.

---

## Lesson 4: Applying Logic to Crowd Behavior

In the previous lesson, you learned how to switch a team from one behavior to another by animating the Active status. You did this by setting up and solving the first part of the simulation, and then using that feedback to set up the second part. This isn't a very flexible method, though. If you change the target position, or the delegates' speed, the frame the behaviors should switch would change, and you'd have to move at least two keys. With more complex simulations, such as those with multiple or moving targets, or delegates that travel different distances, modifications can be difficult.

For complex situations, you can assign *cognitive controllers* to delegates and teams instead of behaviors. The cognitive controller has several *states*. Each state can contain one or more behaviors, each with a different weight. You set the states up as a diagram and link them with transitions. The transitions use MAXScript scripts to determine when one state relinquishes control to another.


In this lesson you'll use a cognitive controller to change a team's behavior from Seek to Repel when the reaches the box.

### Setting up the behaviors

1. Open the file you saved in the middle of Lesson 2 (*cs\_tut08\_lesson4a.max*) or, if you like, open *cs\_tut08\_lesson4.max* from the *cstudio\tutorials\tutorial\_8* directory in your 3DS MAX path.

This scene contains 10 delegates scattered in a spherical pattern about a small, centrally located box.

2. Select the crowd object and open the Modify panel.


3.  Use the Edit Multiple Delegates dialog to turn off Constrain to XY plane for all the delegates.

The method for doing this is covered in the previous lesson, *Applying Avoidance and Animating Behavior Assignments* (see page 562).

4. Add a Seek behavior and a Repel behavior and set the box as the target for both behaviors.
5. For the Repel behavior, turn off Use Radii.

### Using the cognitive controller

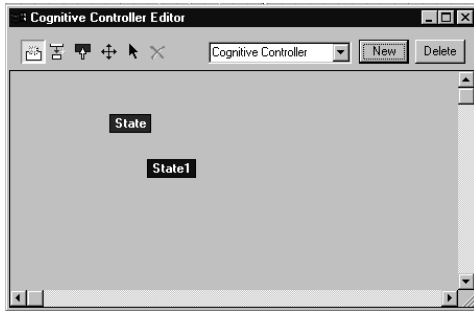
Instead of assigning these behaviors to the delegates, you'll use them to build a cognitive controller and assign the controller to the delegates.

1.  In the Setup rollout, click Cognitive Controller.
2. In the Cognitive Controller Editor window, click the New button on the right side of the toolbar.

This creates a new cognitive controller with the name Cognitive Controller and activates the Create State tool.

3. Click twice in the editor window, in two different locations.

This adds two state boxes to the controller. Each state can contain any number of behaviors.



The cognitive controller editor with two states added

The first state you added is dark red, which means it's the Start State. It is executed first when the cognitive controller begins working. You can set any state to be the Start state.

To make the delegates seek the box first, you'll assign the Seek behavior to the first state. You'll also make the states' names more descriptive, and assign a behavior to each state.

4. Right-click in the window to exit Create State mode, and then right-click the red state box.

The dialog name reflects the name of the state at the time you open it. You can change the name, and the new name is displayed the next time the screen refreshes.

5. In the State dialog, click the name in the upper, one-line edit window and change it to **Seek**.
6. Click the Add button. In the Select Behaviors dialog, click Seek Behavior and click OK.

The Seek behavior is highlighted in the list. You can change the weight, so if there are several behaviors in a state, some can predominate over others.

When you opened the Select Behaviors dialog, the screen refreshed and the State dialog name changed to Seek.

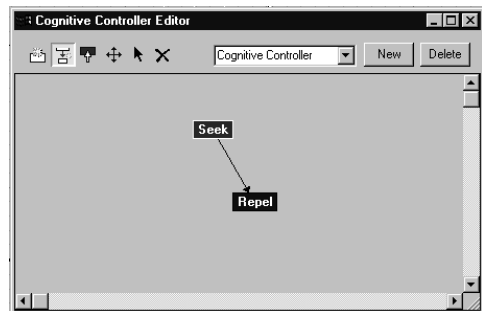
7. Close the Seek dialog by clicking the Close box (X) in its upper-right corner.
8. Change the second state's name to Repel and assign it the Repel behavior.

### Creating a transition

You'll add a transition between the two states and create a short script to tell the cognitive controller when to switch between them.



1. In the editor toolbar, click Create Transition.
2. Drag a line from the Seek state to the Repel state.



An arrow points from the first state to the second. This is the transition.

3. Right-click the transition arrow.

4. In the Transition ("Seek -> Repel") dialog, change the Duration setting to 10.  
This makes the transition to take place over 10 frames instead of 25. A shorter transition causes faster reactions.  
Note the other settings. If you change the priority and a state has two or more outbound transition conditions, all of which are true, the one with the highest priority takes precedence. You can also change the rates at which a transition eases out of one state and into the next.
5. Click the MAXScript button.
6. In the MAXScript window, enter the following:

```
fn TestDist del t = (
  get_dist=distance $box01.pos del.sim-
  pos
  if get_dist < 120 then 1
  else 0
)
```

Alternatively, you can load the MAXScript file *test\_dist.ms* from the *cstudio\tutorials\tutorial\_8* directory in your 3DS MAX path.

This script tests, at each frame, the distance between each delegate and the box. If it is less than 120 units, the delegate moves into the next state, which uses the Repel behavior.

The name of the function, which is the second word of the script (after "fn"), must appear in the Transition dialog as well. The script that starts with "get\_dist" calculates the distance between the position ([name].pos) of a specific object (\$box01) and the position of a generic delegate (del.simpos). "Simpos" is a special position property used to determine a delegate position during a simulation solution.

The transition script is executed once per frame for each delegate that uses this cognitive controller. Each time the script is executed, the cognitive controller substitutes a subsequent delegate for "del".

The first time the script is executed at a given frame, it calculates the position the distance between the objects box01 and delegate01. The second time, it calculates the position the distance between the objects box01 and delegate02, and so on. To see each delegate's name as its position is tested, insert this line in the script anywhere between the first line and the last line:

```
print del.name
```

Be sure to have a Listener window open (press F11) before you run the solution.

To learn more about how scripts work in transitions, read the Procedures sections in the topics *Cognitive Controller Editor* (see page 382) and *State Transition Dialog* (see page 386), where you'll find more sample scripts with descriptions of their components and functions.

7. Double-click the name of the function ("TestDist") to highlight it, and then press CTRL+C to copy it into the Windows clipboard.
8. Close the MAXScript dialog.
9. In the Transition dialog, click in the text box in the Transition Condition group to highlight the text, and then press CTRL+V to paste the name of the function into the box.
10. Close the Transition dialog, and then the Cognitive Controller Editor.  
Two more short steps remains before you can test the transition.

11. Open the Behavior Assignments dialog, and create a team from all the delegates.
12. Assign the cognitive controller (CogControl) to the team, and close the dialog.
13. Solve the simulation.  
The delegates move towards the box until they're fairly close, at which point they turn and move directly away.
14. Try these variations:
  - Add a second transition from the Repel state to the Seek state that turns the delegates back around after they get a certain distance away. **Hint:** You need to change only three items in the first script.
  - Animate the box so the delegates have to chase it before they can get close enough for the transition to begin. **Tip:** You might have to increase the End Solve setting.
  - Use Edit Multiple Delegates to set random speeds for the delegates so they don't all reach the box and turn around at the same time.

---

## Lesson 5: Using Crowd with Animated Non-Biped Objects

**character studio's** Crowd animation system has a number of special features designed for applying behaviors to non-biped objects. In this lesson, you'll learn how to create a flock of birds that automatically flap their wings when they fly horizontally and upwards, and glide without flapping when they fly downwards.

You'll animate several delegates and link them with clones of an animated "global" object. Next, you'll use clip controllers to define clips from the global object's animation. You'll define

"states" to specify which animation clip the clones should use, depending on their parent delegates' speed, acceleration, pitch, and so on. Crowd will blend the clips and synthesize an animation based on the clip and state information.

### Assigning behaviors

1. Load the file *cs\_tut08\_lesson5.max* in the *cstudio\tutorials\tutorial\_8* directory in your 3DS MAX path.

This file contains the basic setup for the birds scene. It contains a crowd object and a "flock" of five delegates in V formation, as well as a vector field surrounding a C-shaped "obstacle" object.


You'll set up a Space Warp behavior so the delegates fly around within the vector field and an Avoid behavior so they don't collide.

2. Select the crowd object and in the Modify panel> Setup rollout add a Space Warp behavior.
3. In the Space Warp Behavior rollout, click the None button and select the vector field object.

The vector field object is the gizmo shaped like two double-headed arrows in an X formation.

Now, any delegates using the Space Warp behavior will use the vector field to guide their motion.

4. Add an Avoid behavior.

5.  In the Avoid Behavior rollout, click Multiple Selection to the right of the None button.

6. In the Select dialog, select all of the delegates and click Select.

This sets the delegates to avoid each other.


7. Open the Behavior Assignments dialog.

The delegates have already been grouped into a team (Team0).

8. Assign both behaviors to the team and close the dialog.
9. In the Solve rollout, click the Solve button. You can stop the simulation at any time by pressing ESC. As Crowd solves the simulation, the delegates move around the obstacle object, roughly following its contours while remaining at their starting height. Some might fly away, which you'll solve with the next step.

To make the birds to fly up and down, you must allow the delegates to move out of the current XY plane.

### Modifying the delegates

1.  In the Setup rollout, click Multiple Delegate Editing.  
The delegates have already been added to the Delegates to Edit list. You don't need to highlight them.
2. In the General group, turn off Constrain to XY Plane.
3. Turn on SET for Constrain to XY Plane. You can also vary the delegates' speeds.
4. In the Speed group, to the right of Average Speed, turn on both Random and SET. Turning on Random gives you access to both Average Speed settings.
5. Change the first Average Speed setting to **3.0**.  
Crowd will use random values between 3 and 5 units per frame for the delegates' speeds.


6. Click the Apply Edit button to close the dialog and apply your changes to all specified delegates.
7. Solve the simulation again. This time, be sure to let it finish.

The delegates move in three dimensions, soaring and then descending.

### Revealing the eagle

You'll assign an animated bird object to each delegate.




1.  Open the Display panel. On the Hide rollout, click Unhide All.  
Only the Eagle object was hidden. If you haven't changed the Perspective view, you'll see it in that viewport. If you have, and the Eagle object is not visible, use Shift+Z to undo viewport changes.
2. Select the Eagle object and drag the time slider to see its animation.  
The bird flaps its wings from frames 0 to 186 and then remains still. The last key frame is 288.

The bird is animated in place, so you can combine the delegates' motion with the bird's animation to create a convincing animation.

There are quite a few interim keyframes, even while the bird doesn't move, because Motion Clips uses interim frames to blend animation clips when it synthesizes an object's animation.

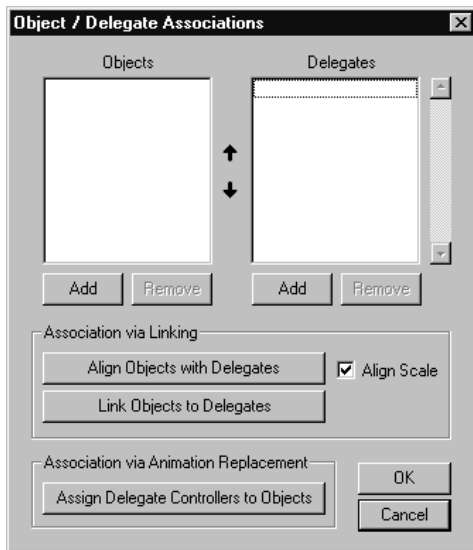
### Assigning animated objects to delegates

1.  Select the crowd object and click Scatter to open the Scatter Object dialog.
2. Specify Eagle as the object to clone  
How Many is already set to 5.



3. Turn off Clone Controllers so the eagle's animation won't be replicated in the clones. If you don't do this, the existing animation mixes with the synthesized new keyframes, which can create confusing and erroneous results.
4. Click Generate Clones and then click OK to exit the dialog.
5. Drag the time slider to see the non-animated clones superimposed over the original, animated eagle.  
Next, you'll link the delegates with the eagle clones, repositioning the clones at the same time.

6.  Click Objects/Delegate Associations to open its dialog.



7. Click the Add button under the Objects list and select the five eagle clones: Eagle01 through Eagle05.
8. Click the Add button under the Delegates list and select the five delegates.

The two lists appear in alphabetical order, indicating the order in which each delegate and eagle mesh are to be paired. You can change the order by highlighting list items and clicking the up and down arrows in between the two lists. In this case, it's not necessary.

9. Click Align Objects with Delegates.

In the viewports, the eagle clones line up with the delegates. But they're not linked yet.

10. Click the Link Objects to Delegates button and click OK to close the dialog.

This creates a parent-child hierarchy for each pair, with the delegate as the parent in each pair.

11. Drag the time slider to observe the eagle clones moving with the delegates.

The eagle clones remain in their initial, wings-raised animation position as they move.

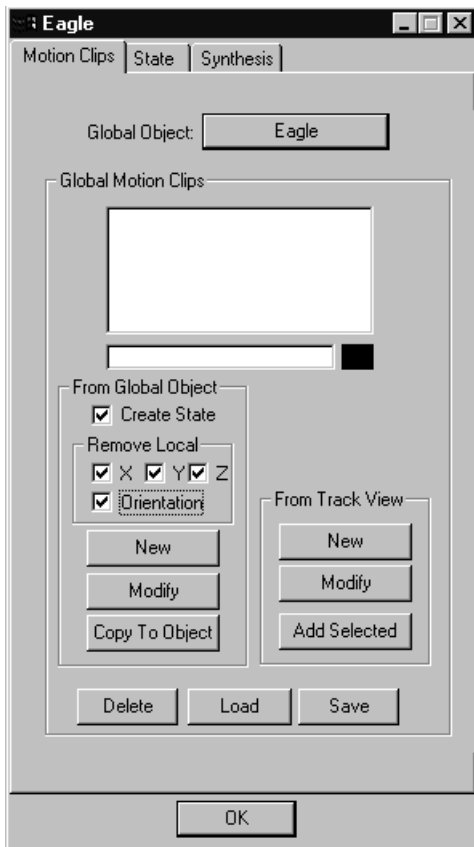
### Using Clip Controllers

You'll use Clip Controllers to apply the correct animation clips from the original eagle object to the clones. There are three clip controller types:

- **Global Clip Controllers.** the objects from which the animation clips are derived, in this case, the Eagle object.
- **Master Clip Controllers.** the block controllers that are created in Track View. There is one controller for each object to which the synthesized animation is applied. For more information, see the Block Controller topic in the 3D Studio MAX Online Reference.
- **Slave Clip Controllers.** objects containing all animation tracks in each Master Clip controller. For more information, see the

Slave Parameters Dialog topic in the 3D Studio MAX Online Reference.

1. In the Crowd Modify panel, scroll down to the Global Clip Controllers rollout at the bottom.
2. Click New and select the Eagle object.
3. In the Global Clip Controllers list, click the Eagle entry and click the Edit button to open the *Synthesis Dialog* (see page 270).

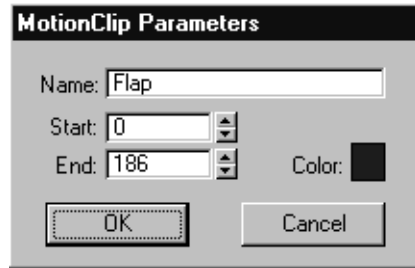


The dialog's three tabbed panels let you set up the clip controllers in a logical workflow.

4. In the Motion Clips panel > From Global Object group, turn off all five check boxes.

These are useful only when you've animated the global object's position and rotation and want to incorporate this animation in the resulting states.

5. In the same group, click New to add a new clip with the default name MotionClip.



The MotionClip Parameters dialog defines animation clips from the global object. The current end frame is automatically set to the last key in the global object's animation.

6. Change Name to **Flap**, and set End to **186**, the last frame of the flapping motion. Click OK.
7. Create a second clip called **Glide**, and set it to use frames **187** through **288**. Click OK.

The clips use different colors so you can distinguish them in Track View.

### Defining states

You'll define two states to specify how a delegate moves at any given time. You can specify values or value ranges for one or more of these motion categories: speed, acceleration, pitch, pitch velocity, and heading velocity. You can also use MAXScript to define a state. As you define states, you assign clips to them. For example, if your animated object uses different motions for turning sideways or slowing down, you create a state for each one and assign it the appropriate motion clip.

Create one state for upward motion that uses the Flap clip, and a second state for downward motion that uses the Glide clip.

1. Click the State tab and click New State. Name the state **Flap**.
2. Click Add Clip and choose the Flap clip.
3. Click Edit Properties.  
You'll use the States dialog to apply the Flap state when the delegate moves upwards.
4. Click the Pitch tab and turn on Use Pitch. The Range settings become available. Use these to specify the range of pitches to activate this state.
5. Set Min to 1.  
The bird will flap its wings whenever its pitch is between 1 through 180 degrees (when it's flying upwards).
6. Click Exit to close the States dialog.
7. Add a second state and call it **Glide**.
8. Assign it the Glide motion clip and set the Glide state to use a pitch range of **-180 to 0**.  
The bird will stop flapping its wings whenever it's flying downward or level. You can also set different flapping rates for different pitches by creating more clips when you set up the global object.
3. In the Synthesis Blend Parameters group, click Auto Blend All.  
Auto Blend creates natural-looking transitions between the clips. To see an example of the results of Auto Blend, set From Clip to Glide and To Clip to Flap. The blend from gliding to flapping begins on the 80th frame of the Glide clip.
4. Click Synthesize All.  
This generates the animations for the clones. You don't have to solve the crowd simulation unless you change delegate or behavior settings.
5. Turn off real-time animation playback in the Time Configuration dialog and play back the animation. Maximize the Perspective viewport and zoom in.  
Although the birds should flap their wings only when they fly upward, and then stop flapping to glide downward, it isn't quite working that way. The problem is that the clips are too long for their matched states. For instance, a bird might fly upward for only 50 frames, but the animation clip for the Flap state is 187 frames long. You'll need to modify the clips.

### Specifying master motion clips


This is the final part of the motion clip setup procedure. Here you specify the master motion clips, that is, the objects to which the synthesis is to be applied. You also blend the clips and synthesize the animation.

1. Click the Synthesis tab.
2. Click the New Master Motion Clip button and select the five eagle clones.  
The global object doesn't appear in the Select dialog because you designated it as the object from which the clips are derived.

### Understanding Block and Slave controllers

Before you modify the clips, take a look at the "behind the scenes" results of the synthesis process; that is, the creation of block and slave controllers.






1.  Open Track View.
2. In the hierarchy pane on the left side, expand Global Tracks, expand Block Control, and expand Global Motion Clip::Eagle.

3. Scroll down or expand the Track View window vertically so you can see a few of the Master Motion Clip items.  
 In the track hierarchy, the global motion clip from the Eagle object's animation is the parent to the master motion clips for the eagle clones. Each master motion clip is a different sequence of blocks, created during the synthesis process. The Flap and Glide blocks are each a different color.
4. Expand a master motion clips to see the Flap and Glide slave tracks. Expand a slave track to see its keys.
5. Close Track View.

### Modifying the clips

Examine the global object's animation to see how to fix the synthesis.

1. Click the Eagle object and drag the time slider slowly.  
 The wing-flapping animation repeats four times. The clip needs only part of one cycle, though, because the clip controller can blend between the last frame of one cycle and the first of the next. The flap cycle should start with the wings all the way up and end with the wings slightly up.
2.  Move the time slider back to frame 0 and turn on Key mode.
3.  Click Next Key repeatedly while watching the eagle object. Stop at frame 48. The wing-flapping animation returns to the initial wings-up position at frame 48.
4.  Click Previous Key once.  
 The last key before the end of the flap cycle is 42. The Flap clip should go from frame 0 to frame 42.

The glide cycle is different. Because there's no actual animation, you must make the clip long enough to avoid generating too many keyframes (as the result of repeating a short clip many times to fill the state), while keeping it short enough to fit within the state length.

For this tutorial, you'll set a glide-cycle length of 10 frames. In your own animations, experiment to find a good length for different clips. Even with animation clips, such as the wing-flap cycle, you may need to scale the animation so the cycle doesn't exceed the shortest length of a state.

Clip controllers perform spline interpolation over the intersection of each pair of blocks using keys within the blocks. To avoid this interpolation altering the motion within the clip, insert additional keyframes near the ends of the motion. This makes the interpolation occur at the extremes of the clip, rather than over its entire length.

So, for the glide clip, you'll move some keys immediately inside each endpoint. These keys are of Bend modifiers applied to the wings, and have already been set to use linear interpolation.

5. With the master eagle still selected, find the key at frame 205 of the track bar (the mini-track view below the viewports).
6. Drag the key from frame 205 to frame 211. It helps to watch the status bar text while you do this.
7. Drag the next key from frame 228 to frame 212.

The key icon will overlap the icon of the key at frame 211.

8. Drag the next key from frame 253 to frame 219.
9. Drag the next key from frame 270 to frame 220.



**Enlarged view of track bar, showing keys at frames 211, 212, 219, and 220.**

In the above illustration, the track bar has been “zoomed in” for illustrative purposes by shortening the active time segment. You’ll see more overlap on your screen.

Now you have keys at frames 211 and 220 that define the clip ends, and keys immediately inside the ends that can be used for interpolation.

To change the clip lengths used for synthesis, edit the global clip controller.



10. In the Crowd Global Clip Controllers rollout, click the Eagle list item, and then click the Edit button.
11. In the Global Motion Clips group, choose the Flap list item, and then, in the From Global Object group, click Modify.
12. Set the End parameter to 42 and click OK.
13. Modify the Glide clip to extend from frame 211 to frame 220.

### Redoing the synthesis

1. On the Synthesis tab, in the Synthesis Blend Parameters group, click Auto Blend All. From Clip and To Clip now reflect the changed clip lengths. Also, after the Auto Blend, the Blend Start items change.
2. Click Synthesize All to apply the new clip lengths to the states.
3. Close the dialog and play the animation.

### Hiding the global object and the delegates

At this point you can hide the global eagle object and the delegates. This lets you see the eagles without the delegates.

1.  Use Select by Name to select all the delegates and the Eagle object.
2.  On the Display panel, click Hide Selected to hide the delegates and global object.

In this tutorial you worked with a global object that was animated in place. All *transformational* animation was supplied by the crowd system. You can also derive delegate motion on the global object’s transformational animation, if it exists.

The global object’s animation wasn’t ideal for the crowd simulation. You learned how to adapt an animation so it works better in a simulation.

## Lesson 6: Creating a Crowd of Swimming Bipeds

In this lesson, you’ll use a Surface Follow behavior to make a crowd of swimming bipeds stay afloat as they swim on an animated ocean.

### Setup

- Open `cs_tut08_lesson6.max` in `cstudio\tutorials\tutorial_8`.  
This file has a plane with animated noise to simulate ocean swells. It also has a crowd, a delegate, and a biped ready for you to select. The first task will be to clone the bipeds.

### Cloning the bipeds

You can clone a gang of bipeds from a single one with Shift-Move.


1. In the Main toolbar, select *biped1* from the Named Selection Sets.
2. Hold down the SHIFT key and move the biped with the mouse, using the transform gizmo.  
It doesn't matter where you move the biped. The Clone Options dialog appears.
3. Set the number of copies to 5 and click OK. You don't have to change the Name field. Five bipeds appear next to the original one.  
**Tip:** You can also clone bipeds one at a time using Edit menu > Clone.

### Setting up the behaviors

1. Select the Crowd object.
2. On the Modify Panel > Setup rollout > Behaviors group, click New and choose the Surface Follow behavior.
3. In the Surface Follow Behavior rollout, click the None button and select the Plane01 object in the viewport.  
This defines Plane01 as the surface to follow.
4. Add a Seek behavior.
5. In the Seek Behavior rollout, click None, and then select the green pyramid (Buoy) in the viewport.

### Scattering the delegates

In this section, you'll use Crowd's Scatter facility to clone the delegate and distribute the clones.

1.  In the Setup rollout, choose Scatter.  
Some of the Scatter parameters are already set up for you.
2. On the Clone tab, confirm that Object to Clone is set to Delegate01.
3. Change the How Many setting to 5.


4. Verify that, in the Position tab > Placement Relative to Object group, "Plane01" appears on the button.
5. In the Rotation tab > Look At Target group, choose Selected Object, then click the None button and pick Buoy from the list.  
You have not yet generated clones or locations, so there is only one delegate in the scene. You can use the All Ops panel to perform all scatter-related operations simultaneously.
6. On the All Ops tab, in the Compute column, turn on Clones, Positions and Rotations, and then click the Scatter button.

The six delegates are now scattered on the surface. Next, you'll redistribute them using random seeds.

7. Turn on Inc Seed for Positions and click Scatter again.  
This increments the random seed value. As a result, the six delegates move to different locations.
8. Keep clicking Scatter until you find a random placement you like. Then click the OK button to exit the dialog.


**Tip:** You can use this dialog to select and scatter other objects in the scene. Click the Select Objects to Transform button and select the other objects.

### Assigning Behaviors


1.  In the Setup rollout, click the Behavior Assignments button.
2. Group the delegates into a team.  
This method is covered in Lesson 1, *Getting Started with Behaviors* (see page 556).
3. Assign the Seek behavior to the team.  
Assignments are displayed in the window on the right.


4. Close the dialog.
5. In the Solve rollout, click the Solve button.  
The delegates seek the buoy for 100 frames and then stop.  
**Tip:** To Solve, you can click Solve in the Solve rollout, or press S with Plug-in Keyboard Shortcut Toggle turned on.
6. In the Setup rollout, click Behavior Assignments.
7. Assign the Surface Follow behavior to the team.
8. Close the dialog and solve again.  
Now the delegates seek the buoy, but also follow the surface.
9. In the Solve rollout, change the End Solve frame to 160.

### Associating the bipeds and delegates

1.  In the Setup rollout, click Biped/Delegates Associations.
2. Below the Bipeds list, click the Add button.
3. In the Select dialog, click the All button and then click Select.
4. Below the Delegates list, click the Add button.
5. In the Select dialog, click the All button and then click Select.
6. Click the Associate button to connect the bipeds and the delegates.  
Nothing changes in the viewport yet. There is more to set up before the bipeds move to the delegate positions.

### Creating a Motion Flow file

1. Select any part of any biped and open the Motion panel.
2.  Turn on Motion Flow Mode.




3.  In the Motion Flow rollout, open the Motion Flow Graph by clicking the Show Graph button.
4. Create a clip in the motion flow graph.
5. Right-click to stop creating clips, and then right-click the clip.
6. In the clip1 dialog, click the Browse button, and then choose *rtswimforward.bip* from the tutorial 8 subdirectory.
7. In this file, frames 0 through 80 comprise a cycle.
8. Change the end frame to 80.
9. Close the dialog.
10. This is a file based on the swimmer you created in *Tutorial 2: Freeform Biped Animation* (see page 447).

The *rtswimforward.bip* file has been modified so there is forward motion for the swimmer. The delegate will take its forward speed from the movement of the center of mass in the *.bip* file.

**Tip:** You can use Layers to add forward motion to a *.bip* file. Create a new layer, animate the Body Horizontal track, and then click Collapse Layers.

11. In the Motion Flow Graph dialog, choose Create Transition.
12. Click the clip to loop the animation.  
This animation can now be repeated to fill the 160 frames of the crowd simulation.
13. In the Motion Flow rollout, choose Save File.
14. Name the file *myswimloop.mfe*.  
You've just saved a motion flow editor file that loops *rtswimforward.bip*.

## Creating a Shared Motion Flow

1.  In the Motion Flow rollout, choose Shared Motion Flow.  
The Shared Motion Flows dialog appears.
2. Click New to create a new Shared Motion Flow.
3. In the Parameters group, click the Load .mfe button and select the *myswimloop.mfe* file you saved earlier.
4. Click the Add button, click All, and then click Select.  
All of the bipeds appear in the Bipeds Sharing this Motion Flow group.
5.  Click the Put Multiple Bipeds in Motion Flow button, and then click OK.  
You must designate the random start clip so the crowd simulation can solve.
6.  In the Shared Motion Flow Graph dialog, click Select Random Start Clips, and then click the *rtswimforward* clip in the graph.  
The clip is now displayed in purple with a probability value of 100.  
Since there is only one clip in this graph, it is 100 percent probable that this clip will be used to start.
7. Select the Crowd Object in the viewport.
8. In the Modify Panel, click Solve.  
After a pause, the bipeds snap into position over the delegate objects.
9. When the solution is finished, play the animation.  
The bipeds are all swimming above the surface of the water. This is because Crowd always uses the default distance between the delegate and the biped's center of mass as an

offset between the two, even if the biped's orientation has changed from its original standing position. You can correct this, allowing the bipeds to swim on the water surface, by changing the Surface Follow behavior's Offset value.

## Adjusting the biped positions

1. Select the crowd object and go to the Surface Follow Behavior rollout.
2. Change the Position On Surface group > Offset setting to -50.
3. Solve again.

The delegates are correctly offset 50 units below the surface. But during the solution, the bipeds are still above the water. This is because, for optimal efficiency, Crowd doesn't apply the offset to bipeds associated with delegates until after it solves the simulation.


Also, when the bipeds reach the buoy, they don't turn around to try to seek it again, but just keep swimming in the same direction. This is because the motion flow network, with its single motion clip, doesn't provide any means for the bipeds to change direction. This situation is covered in the next lesson: *Advanced Crowds/Bipeds*.

4. Play the animation.

The bipeds are swimming in the water.

## Varying the animation

If you want the bipeds not to swim in unison, you need to vary the frame of the animation at which each biped starts. You can do this by letting the software choose a different, random start frame from the motion clip for each biped.

1.  On the Setup rollout, click the Multiple Delegate Editing button.



2. In the Delegates to Edit group box, add all the delegates.
3. In the Biped group box, to the right of the Value 1 and Value 2 columns, turn on Random and SET.

When you turn on Random, the Value 2 parameter becomes available.

4. Change the Value 2 setting to a different frame number, such as 8.  
This means the software will start the `rtswimforward` motion clip at a different, random frame for each biped. For example, one biped might start swimming at frame 0, while another might wait until frame 4 to start moving.
5. Click the Apply Edit button to make the changes and exit the dialog.
6. Solve again, and then play the animation.  
The bipeds no longer swim in unison.

In the final lesson, *Advanced Crowd/Bipeds* (see page 577), you'll learn how to apply **character studio's** motion synthesis engine to crowd-influenced bipeds to create motion flow scripts that avoid collisions with other bipeds.

---

## Lesson 7: Advanced Crowd/Bipeds

To create sophisticated crowd behavior using bipeds, you need to use a motion flow graph with multiple motion clips. In this lesson, you'll set up a graph that allows branching behavior, with most of the clips able to transition to several other clips. As it solves the simulation, Crowd will use *motion synthesis* to apply to each biped the clips that make the most sense in the context of the scene setup.

Here's the premise for this lesson: Two rival mini-gangs, the Tough Cookies and the

Milquetoasts, meet in a face-to-face showdown. Each wants to get to a goal behind its adversaries. The Tough Cookies, whose members brook no interference, charge straight ahead to get to their goal. But the Milquetoasts, being a timid, fearful bunch, will have to make way for Tough Cookies before they can reach their own goal.

As preparation for doing this lesson, please read *Attaching Bipeds to Crowd Delegates* (see page 116). The topic contains important background information for understanding the methods Crowd uses to when solving simulations with bipeds.


### Initial Setup

1. Load the file `cs_tut08_lesson7.max` in the `cstudio\tutorials\tutorial_8` directory, in your 3DS MAX path.

This file contains the delegates and target objects for both gangs, plus the Crowd helper object with team and behavior assignments already made. In the Top viewport, the Tough Cookies stand shoulder to shoulder (figuratively speaking) at the bottom, while the Milquetoasts are more loosely arrayed at the top.

Take a look at the basic Crowd setup.

2. Select the Crowd object and open the Modify panel.


3.  In the Setup rollout, click the Behavior Assignments button.

The ToughCookies team contains delegates 1-3 and is assigned the Seek Green behavior, while the Milquetoasts team, with delegates 4-6, is assigned the Seek Red behavior. A third team, containing all six delegates, is assigned an Avoid behavior whose targets are all six delegates and the spheres.

4. Close the Behavior Assignments and Teams dialog.
5. Solve the simulation. When the delegates come near their goals, press Esc to halt the solution.

The two teams head toward their respective targets while the delegates avoid each other, much as you'd expect.

### **Adding bipeds and associating them with the delegates**

1. In the Perspective viewport, add a Biped about 80 units in height.
2. Use Shift-Move to clone the biped. When prompted for Number of Copies, enter 5. You end up with six bipeds.
3. Select the Crowd object.
4.  Click the Biped/Delegate Associations button in the Setup rollout.
5. Beneath the Bipeds list, click the Add button.
6. In the Select dialog, click the All button, and then click Select.
7. Beneath the Delegates list, click the Add button.
8. In the Select dialog, click the All button, and then click Select.
9. Click the Associate button to connect the bipeds and the delegates.

Note: When using delegates with bipeds, none of the settings in the delegates' Motion Parameters rollout, except those in the Biped group, have any effect in the simulation.

### **Setting up a shared motion flow**

You'll set up a motion flow graph that they'll all share, also known as a shared motion flow. When solving a simulation with delegate-


controlled bipeds, Crowd uses this graph to synthesize motion flow scripts for the bipeds. The motion is shared, but Crowd typically generates a unique script for each biped, depending on the behaviors influencing its delegate. This process is *motion synthesis*.

1. Select any part of one of the bipeds, and go to the Motion panel.

2.  Click the Motion Flow Mode button.

3.  Click the Show Graph button.


As a first step in creating the motion flow graph, you'll add all the necessary clips.

4.  On the Motion Flow Graph dialog toolbar, click the Create Multiple Clips button.
5. Navigate to the `cstudio\tutorials\tutorial_8` directory in your 3DS MAX path, and load the following files (you can select multiple files in the Open dialog by using SHIFT+click and CTRL+click):

- walk\_start.bip
- walk\_stop.bip
- walk.bip
- walk\_L90.bip
- walk\_R90.bip
- walk\_180.bip
- loiter.bip


The software creates clips for each file in the motion flow graph.

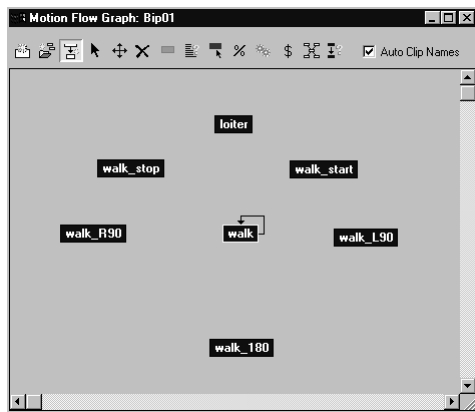
Each clip's file name describes its motion. For instance, *walk\_L90* has the biped make a right-angle turn to the left while walking, and *walk\_180* has the biped turn around and walk in the opposite direction.

6.  If you like, use the Move Clip tool to rearrange the clips in the graph.
- Tip:** The arrangement is arbitrary, but in this case a roughly circular arrangement with *walk* in the center seems to work well with this type of graph.

### Creating a motion flow network

You'll create a motion flow network by adding transitions between pairs of clips. Think about which clips might logically lead to other clips. For instance, it makes sense to create a transition from *walk* to *walk\_L90*, but not from *walk\_stop* to itself.

1.  On the Motion Flow Graph dialog toolbar, click the Create Transition button.
2. Start by clicking the *walk* clip so that it loops back to itself.

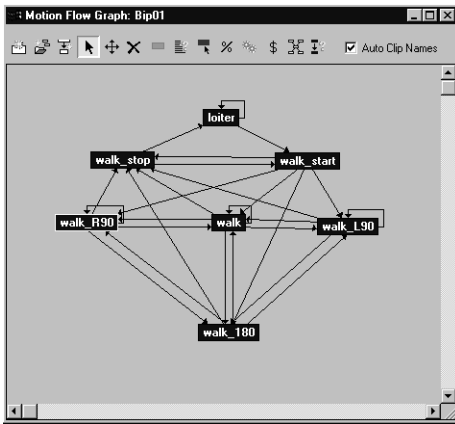


The motion flow network after looping the *walk* clip

This lets the software use the clip repeatedly if necessary.

Typically, you'd want to proceed directly from *walk\_start* to a walking motion.

3. Add another transition from *walk\_start* to *walk* by dragging from the first to the second. The arrow direction should indicate the proper order of the transition.
4. This step gives recommendations for all of the transitions, which, if followed, will produce the expected results. In your own animations, the motion flow network structure is entirely up to you. The transitions you specify are used as strict guidelines by the software when synthesizing the crowd/biped simulation.
  - from *walk\_start* to all other clips except itself and *loiter*
  - to *walk\_start* from *walk\_stop* and *loiter*
  - from *walk* to all other clips, including itself, except *walk\_start* and *loiter*
  - to *walk\_stop* from all other clips except itself and *loiter*
  - from *walk\_L90* to *walk\_180* and vice-versa
  - from *walk\_R90* to *walk\_180* and vice-versa
  - from *walk\_stop* to *loiter*
  - to *walk* from *walk\_R90*, *walk\_180*, and *walk\_L90*
  - from *walk\_R90*, *walk\_L90*, and *loiter* to themselves





### The motion flow network

The final result should resemble the illustration above.

**Tip:** For best results, optimize the transitions. See *Transition Optimization Dialog* (see page 245).

To share the motion flow network among all the bipeds, you need to save it and then reload it with the Shared Motion Flows dialog.

5.  In the Motion Flow rollout, click the Save File button. Save the motion flow network as *my\_Lesson7.mfe*.
6.  In the Motion Flow rollout, click the Shared Motion Flow button.
7. In the Shared Motion Flow dialog, click the New button to add a shared motion flow.
8. In the Parameters, click the Load .mfe button, and load the *my\_Lesson7.mfe* file you saved previously.


If you like, you can instead load the *Lesson7.mfe* file included in the *cstudio\tutorials\tutorial\_8* folder. But if you do, you might first need to edit the *biped.ini*

file in your *plugcfg* folder. Add this line to the end of the file:

```
MoFlowDir=X:\3dsmax\cstudio\tutorials\tutorial_8
```


Replace X with the letter of the drive containing the *3dsmax* folder. This is so the software can find the BIP files referenced by the motion flow network. If your installation differs from the default, change the sample line accordingly.

9. In the Shared Motion Flows dialog, click the Add button, and use the Select dialog to add all of the bipeds to the shared motion flow. All of the bipeds appear in the Bipeds Sharing this Motion Flow list.

10.  Click the Put Multiple bipeds in Motion Flow button, and then click OK.

This turns on Motion Flow mode for all of the bipeds.

You must designate a random start clip so the crowd simulation can solve.


11.  In the toolbar at the top of the Shared Motion Flow Graph dialog, click the Select Random Start Clips button, and then click *walk\_start* in the graph.

The *walk\_start* clip is now displayed in purple with a probability value (100).

This means the software will always use this clip to begin the motion flow script that Crowd synthesizes for each biped.

### Set the delegates to use random start clips

You can specify that biped/delegates should use the first start clip from the current script or from the Random Start Clip setting. When you first solve this simulation, none of the bipeds will have scripts, so you must choose the second option.

1. Select the Crowd Object in the viewport.
2.  In the Setup rollout, click Multiple Delegate Editing.
3. In the Edit Multiple Delegates dialog > Delegates to Edit group, click the Add button.
4. In the Select dialog, click All and then Select.
5. In the Biped group, choose Random Start Clip and turn on its SET check box.
6. Click the Apply Edit button.

### Solving the simulation

- On the Modify Panel > Solve rollout, click the Solve button.

**Tip:** In general, you can speed up the solution significantly by increasing the Solve rollout > Display During Solve group > Frequency setting to **100** or so. But for this lesson, leave it at 1 so you can see exactly what happens during the solution. Or, if it proceeds too slowly on your computer, set Frequency to **2** or **3**.

As before, all six delegates, now teamed with bipeds, solve simultaneously. However, they don't seem to be avoiding each other successfully; in fact, they pass through each other. This happens because, when using bipeds in crowd simulations, directional changes indicated by the delegate behaviors can be overridden by the motion clips guiding the biped animation. For correct character animation, the biped motion must take precedence.

To resolve this, use Crowd's ability to set priorities and solve one biped/delegate pair at a time. Subsequent bipeds can accurately predict the location of solved bipeds and take corrective measures to avoid collisions.

Using *backtracking*, Crowd avoids collisions by backing up in the motion flow graph and trying other paths through the network.

### Setting priorities

1. At the bottom of the Priority rollout, turn on Display Priorities.

In the viewport, each delegate displays the number 0. By default, the Priority value 0 is assigned to all delegates. You can change Priority assignments manually, picking delegates in the appropriate order.

Note: Delegates with lower Priority values take precedence over those with higher values. Thus, for example, a delegate with Priority 0 goes before a delegate with Priority 1, and so on.

2. In the Priority rollout, click the Assign by Picking group > Pick/Assign button.  
For this lesson, assume that each gang's "captain" is in its center. The Tough Cookies, at the bottom of the Top viewport, go first.
3. Use the Top viewport to change priority assignments. First pick the center delegate in the bottom group, near the red sphere. Because, by default, you assign a Priority value of 0 to the first delegate you pick, the number doesn't change.
4. Pick the delegate to its left or right.  
The Priority value 1 appears next to the delegate.
5. Pick the remaining delegate in the bottom group, and then pick the three in the top group, starting with the center delegate.  
The Tough Cookies now have priorities 0, 1, and 2, while the Milquetoasts have 3, 4, and 5.
6. Right-click in the viewport to turn off picking.

In order for Crowd to be able to use priorities, you must turn the feature on.

7. In the Solve panel > Biped group, turn on Biped/Delegates Only. Also turn on Use Priorities and Backtracking.
8. Lastly, still in the Solve panel, turn on Delete Keys Before Solve.

This disables any existing animation keys, so that it's easier to see what's happening during the solve.

### Solving the simulation

- On the Modify Panel > Solve rollout, click the Solve button.

This time, the solution proceeds one biped/delegate pair at a time because Use Priorities is on. First, the captain of the Tough Cookies heads toward the goal, and then his two lieutenants join in, one at a time. During each subsequent sub-solution, you can see the previously solved delegates moving, along with the current biped/delegate pair. Next, the captain of the Milquetoasts goes. This is where things start to get interesting. At about frame 250, the time slider jumps back to frame 220 or so, where the solution begins again (your results may vary). Each time it backtracks, Crowd is returning to the start of the current motion clip and trying a different path through the motion flow network, based on the transitions you supplied. If it can't avoid a collision by jumping back one clip, it returns to the start of the previous clip, and so on. Crowd should backtrack two or more clips at least once in this lesson; watch the last biped closely during the solution.

You might notice that some of the turns taken by the Milquetoast gang members seem too extreme. That's because the bipeds are limited to the motion clips available to

them in the motion flow network. For best results in your biped-based crowd simulations, provide as many different appropriate motion clips as possible. For instance, the results of this lesson would probably look more realistic if you had motion clips for turning 45 and 135 degrees in addition to the existing 90- and 180-degree turns. Of course, that would necessitate a larger and more complex motion flow graph as well.

Other techniques for successful simulations with bipeds include:

- Keep motion clips as short as possible, so Crowd doesn't have to back up too far in the animation when backtracking.
- If you can't shorten clips, change starting positions to allow more time for transitions. For example, in this lesson, you might move the two gangs farther apart.
- Adjust the transitions between clips. See *Transition Editor* (see page 237) for more information.

### Checking the results

To see the results of the motion synthesis, check each biped's motion flow script.

1. Select one of the Tough Cookies bipeds, and then go to the Motion panel.
2. In the Motion Flow Script rollout, scroll through the script. Check the other two Tough Cookies members as well.

Their scripts are pretty straightforward: after the initial walk\_start clip, they just repeat the walk clip until they have to turn right to reach the goal. By the way, the directional adjustment is necessary because of a slight discrepancy between the direction of walking in the walk motion clip and the direction indicated by the Seek behavior. You could correct this by incorporating into the motion flow two additional versions of the walk clip, with the footsteps bent slightly to the left and right.

3. Next, check the Milquetoasts' scripts.

Each uses a different sequence of walk and turning clips to reach their goal without getting in the Tough Cookies' way.

Note that some of the clips, such as Loiter, weren't used. Loiter would be more likely to be used if you restricted the use of turning to avoid. Try reducing the Avoid behavior's Steer To Avoid > Detour Angle setting to see if you can get Crowd to use the Loiter clip. Also try different combinations of other Avoid settings and Seek settings to vary the simulation.





---

## Tutorial 9

# Animating a Multilegged Creature

In this tutorial, you'll use a biped to animate a quadruped. You'll make a freeform animation of a beetle. You'll use IK Blend on the hands and feet with animated pivot points.





Beetle bucking like a bronco (with motion blur)

The shot is from a short video called “Lunchtime” on the World Creating Toolkit CD that ships with 3D Studio MAX. To play the entire animation choose File > View File, then double-click *Wctk\_max\_r3/tutorials/beetle/bugsnfrij.avi*.

In this animation, one beetle hops on the back of the other, and rides him like a bucking bronco, (frames 550 to frames 730 of the movie.)

In this tutorial you’ll how to do the following:

- Animate a leaping quadruped with freeform animation
- Use IK Blends and pivot points on the hands and feet
- Use Snapshot to create extra limbs

## Lessons

*Lesson 1: Animating a Quadruped with Freeform Animation (see page 586)*

*Lesson 2: Adding Extra Limbs (see page 598)*

---

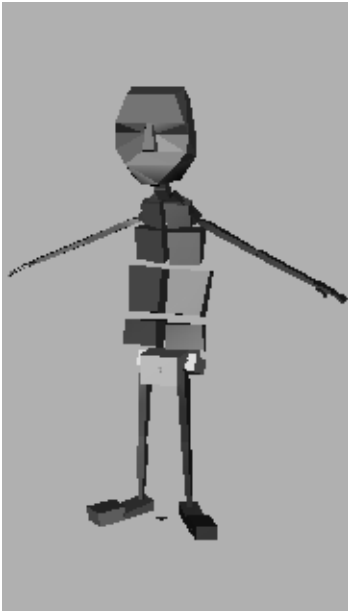
## Lesson 1: Animating a Quadruped with Freeform Animation

You’ll use the same techniques you used in *Animating a Freeform Walk Cycle (see page 457)*, constraining both feet and the hands, and animating the pivot points. You’ll lock the hands and feet at certain keys so you can adjust the biped’s center of mass and spine to create your basic poses.

You’ll learn to animate a character that leaps on all fours, but also can walk upright on two legs.

### Setup


- Open *cs3\_tut09\_quad1.max*.  
This is the biped used for the beetle.




The beetle and the biped



He walks upright through most of the animation, but in this shot he'll move on all fours. Because he is a two-legged character pretending to move like a horse, you will not change him in figure mode to a horse pose. Instead you will put him into his initial riding pose at frame 0 of a freeform animation.

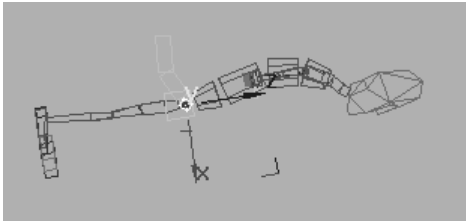
#### Saving a *.fig* file

1. Select the biped, then open the Motion panel.
2.  Turn on Figure mode.
3. Save a *.fig* file of the biped as *mybeetle1.fig*. This preserves the beetle's original pose.

4.  Turn off Figure mode.  
The pose does not change, since the beetle isn't animated in this file.

#### Putting the beetle on all fours

1.  Turn on Animate.  
You'll use the Animate button instead of the Set Key buttons for the first pose.
2.  On the Track Selection rollout choose Body Rotation.  
This selects the center of mass and activates rotation.
3. Rotate the center of mass approximately 80 degrees, so the biped is nearly horizontal.

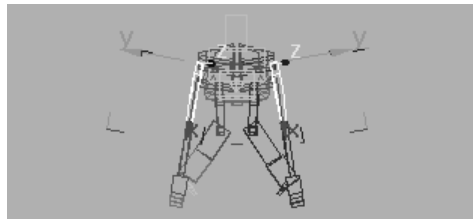
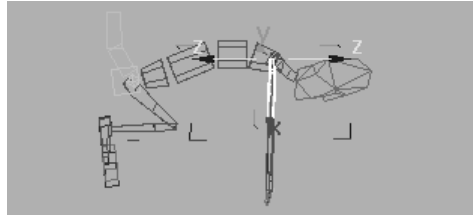


#### Rotate the root (Bip01)

The next steps rotate the arms and legs into position.

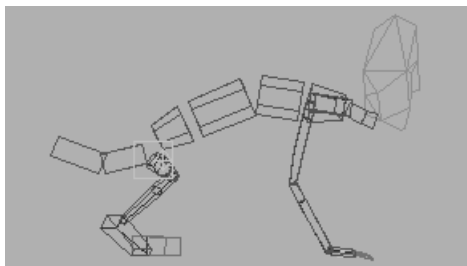
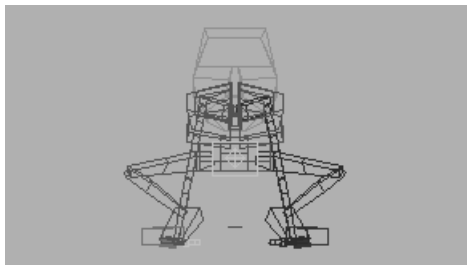
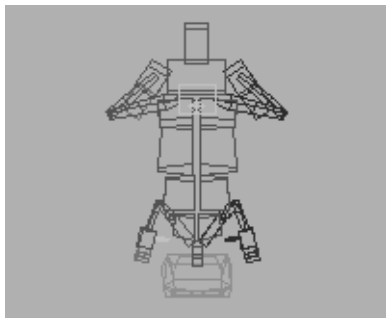
4. In the Left viewport, draw a small selection box through the middle of the thighs. This selects both thighs in a single move.
5. Right-click and choose Rotate.
6. Rotate the thighs approximately 120 degrees about the Z axis.
7. Press PAGE DOWN to move down the chains.  
This selects the calves.
8. Rotate the calves so they are nearly horizontal and the feet are behind the biped.  
You'll add a couple of rotations to the arms now.
9. In the Top viewport, select *Bip01 R Clavicle*. Hold down the CTRL key and click the *Bip01 L Clavicle*.  
Both clavicles are selected.
10. In the Top viewport rotate the clavicles about the Y axis, approximately -40 degrees.
11. Press PAGE DOWN to select both upper arms.
12. Rotate the arms approximately -90 degrees about the X axis, so the palms are parallel to the ground.
13. Rotate the arms approximately -90 degrees about the Z axis, so the arms stretch out in front of the biped.

14. Rotate the arms about the Y axis, until they hang down.
15. Rotate the arms about the Z axis so they move away from the body a bit more.



#### Rotate the arms down.

16. Press PAGE DOWN twice to select the hands.
17. Rotate the hands so they are parallel to the ground.
18. **Tip:** Zoom in so you don't select the fingers by accident.
19. Rotate the head up, so the biped is looking straight ahead instead of downwards.
20. Rotate the toes so they are flat on the ground. If necessary, rotate the foot, then the toes, so the feet and toes point forwards.  
Watch the Perspective viewport as you rotate.  
Make any additional adjustments to the legs, arms, spine, and so on until your model resembles the one illustrated here. Look at the beetle from several views, or rotate around it in the Perspective viewport.



The beetle down on all fours

If the beetle's posture is incorrect, open *cs3\_tut09\_quad2.max* to compare.

### Changing Dynamic Blend and Balance Factor

You'll need to turn off Balance Factor and Biped Dynamics before you begin this freeform animation. These are settings that add secondary motion to footstep animation, which assumes the biped is walking upright.

For the beetle traveling on all fours, you'll turn off Balance Factor and Dynamics Blend.

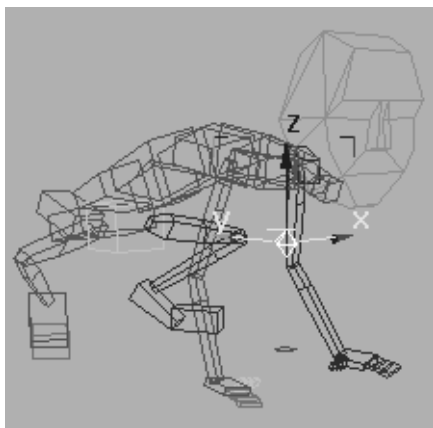
1. Open *cs3\_tut09\_quad2.max*.

Animate

2. Be sure Animate is still on.
3. Select the Body Horizontal Track.  
This selects the center of mass and turns on Move.
4. On the Key Info rollout in the Body Dynamics group, change the Balance Factor from 0 to 1.

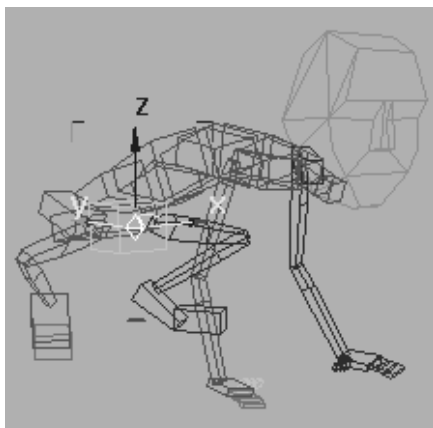
In the Perspective viewport, the center of mass jumps from the pelvis to the area under the chest.

5. Change the Balance Factor to 2.  
The center of mass jumps to the area under the throat.



#### Balance Factor 2

6. Change the Balance Factor back to 0. This will ensure that the hips and spine balance around the pelvis area.



#### Balance Factor 0

In this file, the Balance Factor was already set to 0.

7. In the Track Selection rollout, choose Body Vertical.
8. Set Dynamics Blend to 0.

**Tip:** You can also access Balance Factor and Dynamic Blend from track bar keys for Body Horizontal and Body Vertical. Just Right-click on the key and choose the either from the list. You'll find Balance Factor at the bottom of the Body Horizontal key dialog, Dynamics Blend at the bottom of the Body Vertical dialog.

For another tutorial that teaches about Balance Factor, see Tutorial 4, *Lesson 4: Creating the Illusion of Weight* (see page 519).

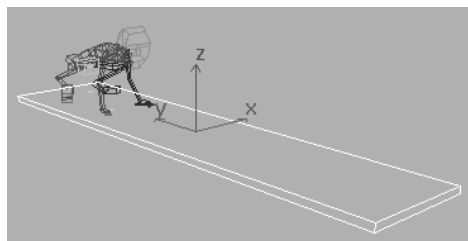
#### Adding a ground plane

In the tutorial *Animating a Freeform Walk Cycle* (see page 457), you set keys that locked the feet to the ground. These keys locked the feet to the world rather than to an object.

In this tutorial, you'll create a box and lock the beetle's feet to the object instead of locking them to the world.

1. Zoom out of the Top viewport so the biped is smaller.
2. In the Create panel, choose *Box01*.
3. Create a box beneath the beetle. In the Top viewport, start at the top left and drag down to the lower right. After you click, continue to drag down to create a negative height for the box.

Make the box 500 units in length, 100 units in width and any negative height.



**Biped and box**

4. In the Perspective viewport, select any part of the biped.



5. In the Motion panel, choose Body Vertical in the Track selection rollout.

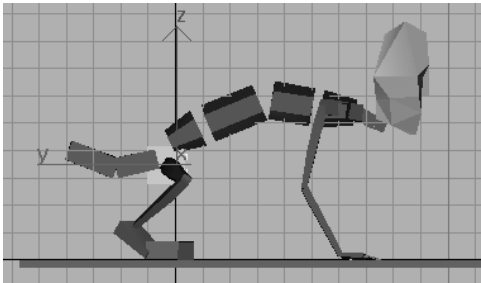
6. Move the biped up and down in the viewport.

The biped passes through the box.

7. Right-click on the Left viewport. Press **W** to maximize it.

8. Move the biped down so its hands and feet are in contact with the box.

Zoom in to see the effect. Adjust the bipeds hands and feet if they are uneven.



Biped positioned on box

### Locking down the hands and feet

1. Open *cs3\_tut09\_quad\_and\_box.max*.

This file is like the one you've been working on, but the feet have been rotated so the heels are flat on the ground.



2. Turn Animate on.
3. Change the Perspective viewport to Wireframe and zoom in so you can see the hands and feet.
4. Select the green *Bip01 R Foot*.



5. In the IK Key Info rollout, choose Set Key.

The pivot point is on the ball of the foot. You can leave it there.



6. At the bottom of the IK Key Info rollout, click the Select Object Space Object arrow. Click the box in the viewport.

The name *Box01* is displayed in the rollout.



7. In the IK Key Info rollout, choose Set Sliding Key.

8. Select *Bip01 L Foot* and set another key.

9. Select *Box01* again, and set a sliding key.

The feet are now locked to the box.

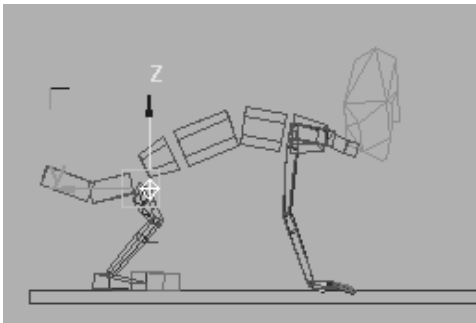
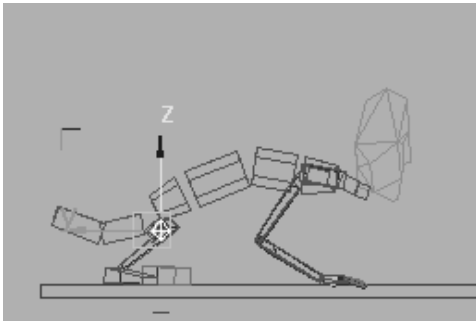
**Warning:** You must set the Object Space Object before you set an IK Key. Otherwise the IK key parameters will be reset to IK Blend of zero and Body Space on. Also note that you can define only one object space object. You can not keyframe from one object space object to another.

10. Select the Body Vertical track and move the center of mass up and down.

The knees bend with the movement. The feet don't go through the box.

11. Repeat these steps again for the left and right hands.

The hands and feet act the same. Both are locked and do not pass through the box.



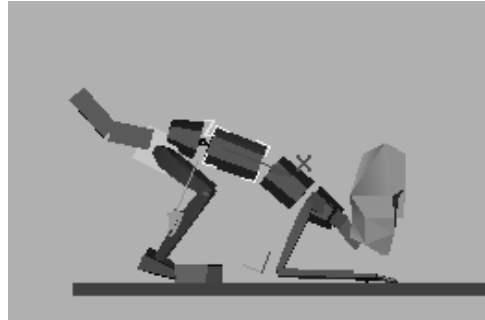
Knees and elbows bend when the center of mass moves.

### Adding Anticipation

Before a character jumps, you must prepare for the jump. To show anticipation a character moves in the opposite direction from the jump, and holds the pose, as if to tell the audience that something is about to happen.

1. With Animate still on, move to frame 10.
2. In the Track Selection rollout, choose Body Rotation.
3. In the Left viewport, rotate the center of mass so the chin is almost touching the ground.
4. Move the center of mass back so the elbows lower almost to the ground.
5. Rotate the head so it's looking up.

6. Add extra rotations to the Spine objects so the spine arches.
7. Raise the center of mass if the elbows pass through the box.



Pose at Frame 10

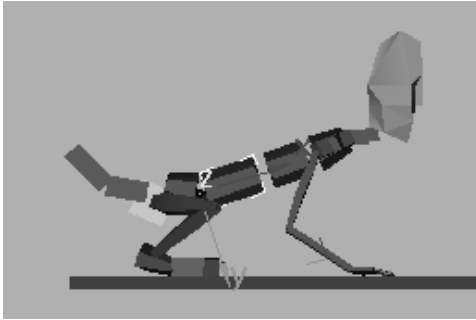
8. Move the time slider back and forth between frames 0 to 10 and observe the motion.
9. Save your work as *myquad1.max*.

### Continuing the motion

You'll continue the motion at frame 20. You'll adjust the timing after you get all the poses in place.

1. Move the time slider to frame 20.
  2. In the Track Selection rollout, choose Body Vertical.
  3. Lower the center of mass so the beetle crouches down.
  4. Select the biped spine elements (*Bip01 Spine* through *Bip01 Spine03*).
  5. Rotate the spine links.
  6. Rotate the upper spine links.
- Make your biped match the illustration.

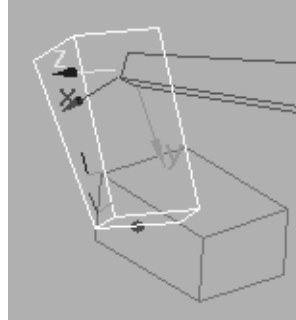


**Pose at Frame 20**

Next, you'll animate the pivot points and rotate the feet so they roll from the heel to the toe. You'll rotate the hands so they rock from wrist to palm to fingertips.

### Rotating the feet

1. With Animate still on, move to frame 23.
2. Select *Bip01 R Foot*.  
The pivot is displayed.
3. Set a sliding key.  
Now you can select a pivot.
4. Select the pivot at the ball of the foot and set a planted key.  
The pivot is locked at that frame.
5. Turn off Select Pivot and right-click on the foot, then choose Rotate.
6. Rotate the foot, so the heel is raised.  
If you look closely you will see the foot pop back. The reason this is happening is that the IK solution is giving the knee precedence over the ankle. To stop this pop, you'll need to change the Ankle Tension.
7. Go to frame 0.
8. In the IK Key Info, set the Ankle Tension to 1, and set a Planted Key.
9. Now move the time slider back and forth and you will see the correct rotation.



10. Go to frame 27.

11. In the IK Key Info rollout, click set planted key.

12. Click Select Pivot and select the pivot at the front and center of the toes.

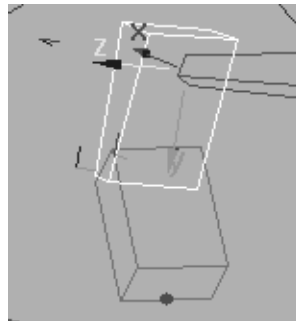
You may have to zoom in to see it.

Selecting the pivot automatically sets a key for it. You don't have to choose set planted again.

13. Turn off Select Pivot.

14. Move the time slider to frame 30 and rotate the foot.

The foot now rotates off the toe.



### Rotating the hands

1. With Animate still on, move the time slider to frame 23.
2. Select the green *Bip01 R Hand* and set a planted key.

The pivot is displayed and locked at frame 23.

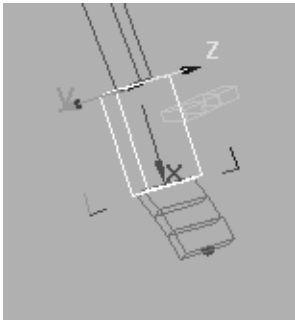
3. Rotate the hand, so the heel of the palm is raised.

The fingers rotate downwards unlike the toes. You'll have to rotate the fingers to correct this.

4. Select the fingers and rotate them upwards, so they don't go through the surface of the box.

Since Animate is still on, you don't have to set keys for the fingers after you rotate them.

5. Select the hand again, move the time slider to frame 27, and set a sliding key.
6. Choose Select Pivot and select the pivot in the front and center of the fingers.
7. Turn off Select Pivot.
8. Rotate the hand.

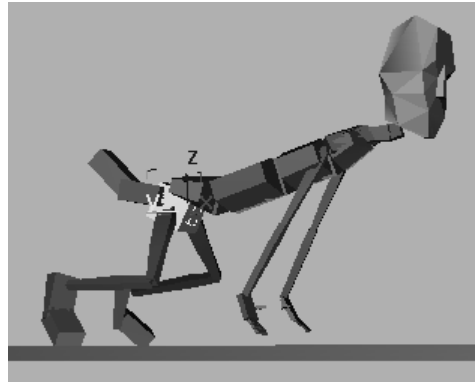


The hand rolls up, with only the fingertips touching the ground.

**Tip:** If the hand doesn't rotate around the new pivot, reassign the Box01 as Object Space Object and rotate again. This time it will.

9. Repeat the same procedures for the opposite set of limbs.

10. At frame 30 move the center of mass up and forward so the beetle is just leaving the ground.



**Pushing off pose**

11. Save your work to *myquad2.max*.

### Flying in the Air

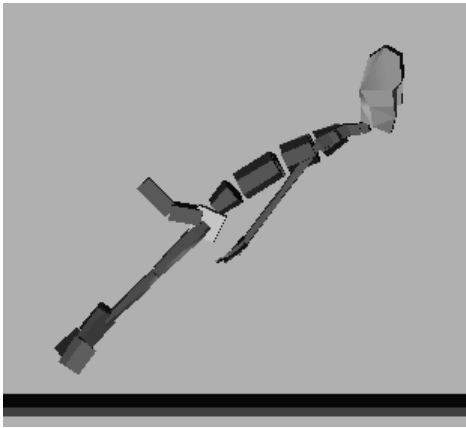
When the beetle sails through the air, you'll use free keys on the hands and feet. No IK blend is needed.

1. Turn on Animate, and move to frame 40.
2. Set free keys for each foot and hand. With Move or Rotate active in the Main toolbar, click a hand or foot and, in the IK Key Info rollout, click Set Free Key.

This sets IK Blend to 0 at frame 40 for the extremities.

3. Select the center of mass and raise the biped up and to the right.  
The hands and feet are now off the box and the biped is in the air.
4. Select the *Bip01 R Foot*, and move it so the legs extends fully.
5. Extend the left leg the same way.  
The knees are no longer bent.
6. Select both feet, and rotate them downwards and backwards.

7. Press PAGE DOWN to select the toes and rotate them as well.
8. Select both the hands and rotate them downwards and backwards.
9. Select and rotate *Bip01 Spine* (the first spine object) so the biped straightens up.
10. Rotate each spine objects individually. Go up through the spine, rotating each object, then rotate the first object again.  
This will remove the swayback in the spine.
11. Rotate the head so the beetle looks ahead.

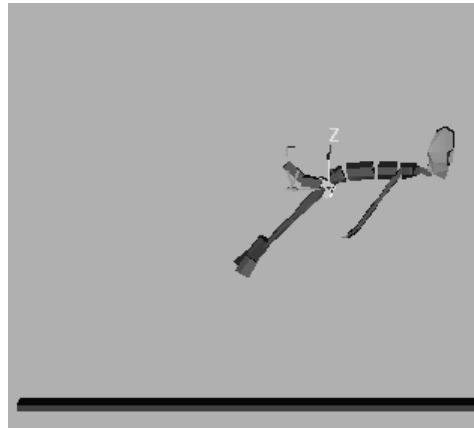


Flying-into-air pose

### Creating the top of the jump pose

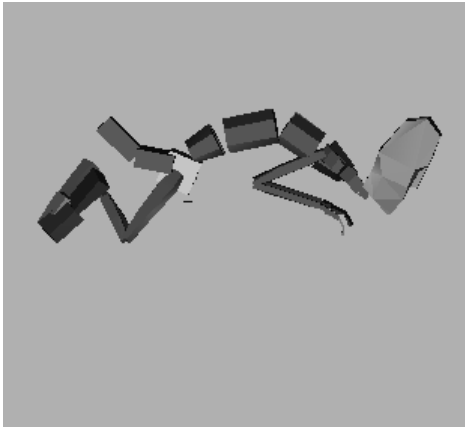
The character is almost floating, reaching the height of his jump. You'll rotate the spine to hunch him over, tuck his elbows back, and bend his knees.

1. At frame 56, on the Track Selection rollout, click the Body Vertical track.
2. Move the biped up and over to the height of his jump.



Move center of mass to height of jump.

3. Select *Bip01 Spine* and rotate it so the biped bends forward.
4. Press PAGE DOWN to select *Bip01 Spine01*.
5. Deselect the thighs, then rotate the selected spine link.
6. Continue moving and rotating each biped spine object.
7. Rotate the head so the pose is more natural.
8. Select both biped upper arms, and rotate them so the elbows are closer to the chest.
9. Press PAGE DOWN to select both biped lowerarms, and rotate them so the fingers are near the biped's chin.
10. Select both thighs and rotate them so the feet kick back.
11. If you like, you can select both feet and then move them to tuck under the tail a little.



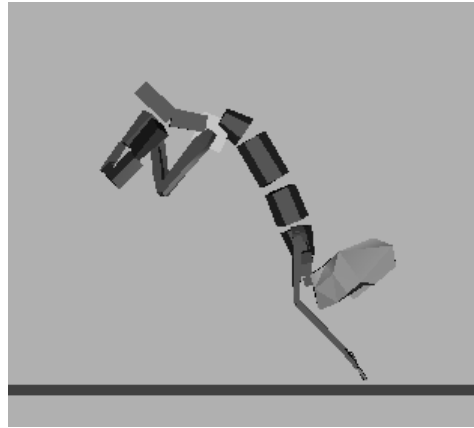
Top of jump pose

### Nearing the Ground Pose

You'll continue the biped's descent to the ground.

1. At frame 72, with Animate still on, move the beetle downwards and to the right.
2. Select the first three *Bip01 Spine* objects and rotate them downwards until the biped's shoulders are pointing at the ground.
3. Select only *Bip01 Spine 2* and rotate about the Z axis until the biped's back straightens.
4. Select both upper legs and rotate them.
5. Rotate the lower legs.
6. Select both hands and move them so they are outstretched.

You can position the limbs using forward rotations (as you just did with the legs), or with inverse kinematics moves (as you just did with the hands).



Arms extended to touch the ground

### Creating the Landing Pose

The beetle will land at frame 85. You'll lock down the hands and adjust the center of mass.

1. Move the time slider to frame 85.
2. Select and move the center of mass so the fingers touch the box.
3. Select both feet and move them outwards and up so the legs extend behind and above the biped.
4. Rotate the feet so the toes point skywards.
5. Move the time slider back and forth and watch the motion.

You've created a little kick with the legs.

### Rotating the hands

Rotate the hands up so the beetle doesn't break his fingers when he lands.

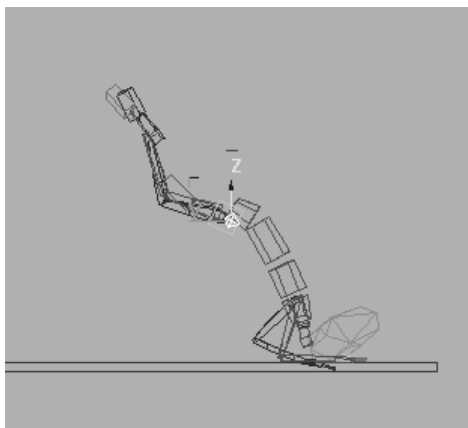
1. At frame 93, select and rotate one hand so it is parallel to the ground.
2. Rotate the other hand in the same way.
3. Move the center of mass so the biped's hands are touching the box.

Next you will lock down the pivot points.

4. Select one hand and set a sliding key.

Now you can set a key to lock the pivot.

5. Choose Select Pivot, click on the pivot at the wrist and set a sliding key.
6. Move the center of mass down so the elbows are close to the ground.  
The hands should lock to the box and the elbows should bend in response to the movement.
7. Move one hand in front of the other.  
This makes it look like the hands are hitting the ground independently.



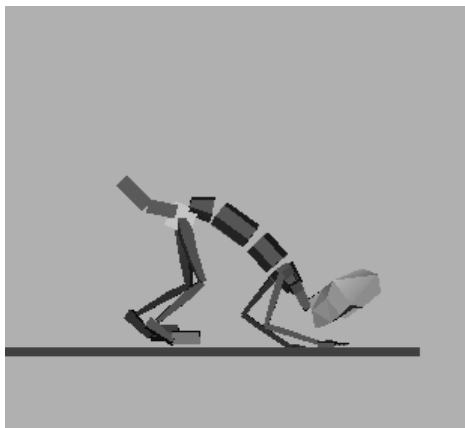
The hands don't land together

#### Adjusting the legs

1. Select both thighs and rotate them so the legs are under the beetle.
2. Press PAGE DOWN twice to select both feet, then rotate them so the feet prepare to contact the ground.
3. Press PAGE DOWN and rotate both toes until they are nearly horizontal.
4. Select and move the center of mass so the toes are in contact with the ground.
5. Move one foot backwards and upwards slightly.

When an animal lands, its feet don't hit the ground at the same time.

6. Rotate *Bip01 Spine01* to raise the elbows and upper body slightly.



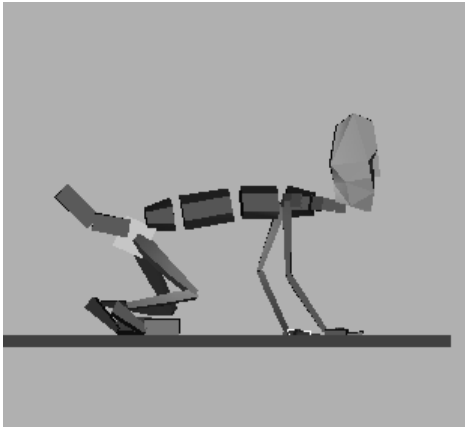
Hitting the ground pose

7. Select both feet and move the time slider to frame 91.
8. Move the feet so the beetle compresses as he lands.

The feet should be tucked under the tail.

#### Creating the Last Pose

1. At frame 94 select one hand and set a planted key.
2. Select the pivot on the wrist.
3. Select the other hand and set a planted key.
4. Select the pivot on the wrist.
5. Select one foot and set a planted key.
6. Select the pivot at the ball of the foot.
7. Select the other foot and set a planted key.
8. Select the pivot at the ball of the foot.
9. Rotate the spine so the arms straighten out and the shoulders raise up a bit.



Last Pose

10. Save your work to *myquad2.max*.

### Adjusting the timing

The take off and landings are too slow. Use the keys in trackbar to improve the timing.

1. Select the entire biped.  
The keys displayed in the trackbar are the keys for the entire animation.  
There is too much time between frames 30 and 40.
2. Select the keys from 72 through 100 by dragging a selection rectangle around them in the trackbar.
3. Slide the selected keys so the key that was at frame 72 is now around frame 60.
4. Select the keys from frame 40 through the end of the animation.

5. Slide these keys so the key that was 40 is now around frame 33.
6. Move the time slider back and forth.  
The take off and landing is much quicker.
7. Save your work as *myquad3.max*.  
To compare your work to the finished animation, load *cs3\_tut10\_quad\_finished.max*.

---

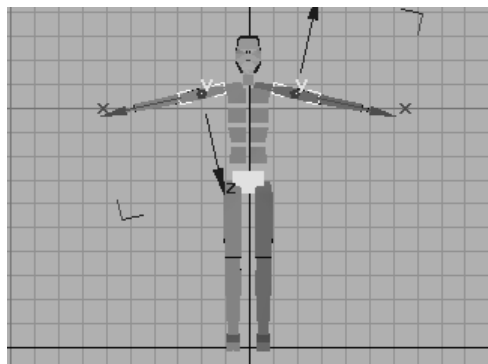
## Lesson 2: Adding Extra Limbs

In this lesson you will add two arms and link them to the biped to make a six-legged creature. You'll use the Snapshot command is used to create the two extra arms which are repositioned and linked to the spine.

Note that the extra limbs become 3D Studio MAX objects, and must be animated with rotations. They will not respond to Biped figure mode or inverse kinematics.

### Using the Snapshot command

1. Create a biped.
2. Select the upper arms and rotate them about the local Y axis to move them away from the body.

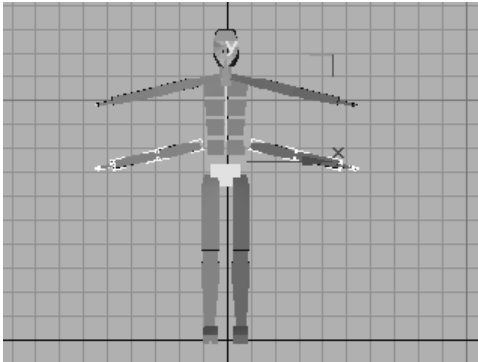


3. Use region select to select both the biped arms.

4. On the 3D Studio MAX Main toolbar click Snapshot.  
Snapshot is one of the options in the Array flyout.
5. Make sure the Single and Mesh options are on in the Snapshot dialog. Click OK.

### Selecting and moving the arms down

1. On the 3D Studio MAX toolbar, click Select by Name.  
The copied arms are coincident with the original arms so you'll use Select by Name to select the copied arms.
2. In the Select by Name dialog turn on Display Subtree and Select Subtree.
3. Click *Bip01 L UpperArm01*. Hold down the CTRL key and click *Bip01 R Upperarm01* and click Select.  
Both of the copied arms, hands and fingers are selected.
4. In the viewports, move the copied arms down to where they should attach to the body.



### Linking the arms to the biped spine

1. On the 3D Studio MAX toolbar, click Select and Link.
2. Link both of the copied upper arms to the closest biped spine object.
3. Animate the copied limbs using rotations.

Note that the copied arms are now 3DS MAX objects. Figure mode cannot be used on the copied arms.

Use frame 0 as the fit frame to a mesh. Start animating the arms at frame 1.

You can use the same procedure with 3D Studio MAX bones to create jaws, wings, fins and so on. Create the additional bones, link them to the biped at the nearest location, and then animate them using 3DS MAX techniques.





---

# Biped MAXScript Extensions

---

## Biped Load and Save Methods

---

This topic will cover the load and save methods for motion, figure, score, keys, motion capture, and talent files.

### -- Motion File Access Methods

`biped.LoadBipFile <biped_ctrl> <filename>`

Load a biped motion file. Motion files include, footsteps, keyframe settings, the biped scale, and the active gravity value (GravAccel). IK Blend values for the keys and Object Space Object information are also loaded.

`biped.SaveBipFile <biped_ctrl> <filename>`

Save a biped motion file. A *.bip* file includes footsteps and keyframe data. Biped files store the complete movement and allow you to create libraries of motion. Create your own *.bip* library by animating the biped and saving a *.bip* file.

### Notes

Load/Save a Biped (.BIP) file

### -- Figure File Access Methods

`biped.LoadFigFile <biped_ctrl> <filename>`

Load a Figure file. Figure mode must be active to load a Figure file. Figure files allow you to apply the structure of one biped to another. Reload a Figure file if you accidentally lose your biped Figure mode pose; this pose is the biped fitted to a mesh.

`biped.SaveFigFile <biped_ctrl> <filename>`

Save the structure and position of a biped in Figure mode. After fitting the biped to a mesh in Figure mode, save a figure file. If the biped is accidentally moved in Figure mode, reload this file.

### Notes

Load/Save a Figure (\*.FIG) file

### -- Score File Access Methods

`biped.LoadScoreFile <biped_ctrl> <filename>`

`biped.SaveScoreFile <biped_ctrl> <filename>`

Score files are Step files which save footsteps, but don't save body keyframes. They are an ASCII file format that enables developers to write programs that generate step files for biped motion. The online document *stp.rtf*, provided with character studio in the `\cstudio\docs` directory, describes the *.stp* format.

### Notes

Load/Save a Step (\*.STP) file

### -- Motion Capture File Access Methods

```
biped.loadMocapFile <biped_ctrl> <file_name> [#prompt]
```

Load a motion capture (.BIP, .BVH, .CSM) file. If #prompt is specified, the file is not automatically loaded, rather the Motion Capture Conversion Parameters dialog is displayed with the specified file as the motion capture file.

#### -- Talent File Access Methods

```
biped.adjustTalentPose <biped_ctrl>
```

After loading a marker file, use Adjust Talent Pose to correct the biped position relative to the markers. Align the biped limbs to the markers then call **biped.adjustTalentPose** to compute this offset for all the loaded marker data.

#### Note

Calibration controls are only enabled when a marker or *.bvh* file is imported in its raw form. Do not use key reduction or extract footsteps when you import a marker file for the first time.

```
biped.saveTalentFigFile <biped_ctrl> <file_name>
```

After changing the biped scale in Talent Figure mode, save the changes into a *.fig* file. Use this file in the Motion Capture Conversion Parameters dialog to adjust marker files created by the same actor.

```
biped.saveTalentPoseFile <biped_ctrl> <file_name>
```

Save a Talent Pose adjustment as a *.cal* file.

Save a *.cal* file after adjusting the biped relative to the markers. A *.cal* file is used for processing marker files that require the same adjustment. A *.cal* file can be loaded in the Motion Capture Conversion Parameters dialog during marker file importation.

#### -- Layer related methods

```
biped.collapseAtLayer <biped_ctrl> <index>
```

Collapses all layers till the specified layer. All underneath should be active for this function to work. Return true if successful, otherwise false.

```
biped.createLayer <biped_ctrl> <index> <name>
```

Creates a layer at the specified position, maximum index value can be NumLayers + 1

```
biped.deleteLayer <biped_ctrl> <index>
```

Deletes the specified layer

```
biped.getCurrentLayer <biped_ctrl>
```

Returns the position of the currently active layer in the UI

```
biped.getLayerActive <biped_ctrl> <index>
```

Returns true if the specified layer is active.

```
biped.getLayerName <biped_ctrl> <index>
```

Returns the specified layer name

biped.**numLayers** <biped\_ctrl>

Returns the number of layers

biped.**setCurrentLayer** <biped\_ctrl> <index>

Sets the active layer in the UI to the specified layer

biped.**setLayerActive** <biped\_ctrl> <index> <bool\_val>

Sets the specified layer to active/inactive based upon the bool\_val passed

biped.**setLayerName** <biped\_ctrl> <index> <name>

Sets the specified layer name to the value passed

The following example will create a new Biped, access it's Vertical\_Horizontal\_Turn(Body) controller and load a specific \*.Bip file:

#### Script Example

```
-- create a new Biped
bipObj = biped.createNew 100 100 [0,0,0]
-- select bipObj
bip = bipObj.transform.controller
max motion mode
-- File I/O
biped.LoadBipFile bip (CSPATH + "scripts\\Limloop.bip")
```

#### See Also

*Biped Creation (see page 603)*

*Biped Controllers (see page 610)*

*Biped Node Hierarchy (see page 606)*

---

## Biped Creation

The following properties and methods are applicable to any created Biped.

#### Constructor

biped.**createNew** <height\_float> <angle\_float> <wpos\_point3> . . .

<height\_float> the height of the Biped to be created

<angle\_float> angle in degrees

0( will make the Biped face towards the right.

<wpos\_point3> world position of the Biped. This is the position of the COM shadow.

#### Creation Properties

<biped>.arms Boolean Default: True

Specifies whether or not arms will be generated for the current biped.

<biped>.neckLinks Integer Default: 1

Specifies the number of links in the biped neck.

<biped>.spineLinks Integer Default: 4

Specifies the number of links in the biped spine.

<biped>.legLinks Integer Default: 3

Specifies the number of links in the biped legs.

<biped>.tailLinks Integer Default: 0

Specifies the number of links in the biped tail. A value of 0 specifies no tail.

<biped>.ponytail1Links Integer Default: 0

Specifies the number of Ponytail links.

<biped>.ponytail2Links Integer Default: 0

Specifies the number of Ponytail links.

<biped>.fingers Integer Default: 1

Specifies the number of biped fingers.

<biped>.fingerLinks Integer Default: 3

Sets the number of links per finger.

<biped>.toes Integer Default: 5

Specifies the number of biped toes.

<biped>.toeLinks Integer Default: 3

Specifies the number of links per toe.

<biped>.ankleAttach Float Default: 0.2

Specifies the right and left ankles' point of attachment along the corresponding foot block. The ankles may be placed anywhere along the center line of the foot block from the heel to the toe.

A value of 0 places the ankle attachment point at the heel. A value of 1 places the ankle attachment point at the toes.

<biped>.trianglePelvis Boolean Default: True

Select this control to create links from the upper legs to the lowest biped spine object when Physique is applied. Normally, the legs are linked to the biped pelvis object.

The pelvis area can be a problem when the mesh is deformed with Physique. Triangle Pelvis creates a more natural spline for mesh deformation.

## Notes

Creates a new Biped with the given keyword parameters. All UI limit constraints apply on keyword parameters, example: **fingers** and **fingerLinks** parameters are not respected when **arms** is set to false.

The following example will create a Biped with specific non default values and facing the front.

## Script Example

```
bipObj = biped.createNew 100 -90 [0,0,0] arms:true neckLinks:2 \
spineLinks:4 legLinks:4 tailLinks:1 ponyTail1Links:1 \
ponyTail2Links:1 fingers:5 fingerLinks:2 toes:4 \
toesLinks:2 ankleAttach:0.3 trianglePelvis:True
```

## Biped Display Preferences Access

Displays the Biped Display Preferences dialog.

### Method

```
biped.displayPrefsDlg <biped_ctrl>
```

<biped\_ctrl>      **Controller** which is attached to the transform track of the Biped root objects.

A *Biped Vertical\_Horizontal\_Turn(Body):Matrix3 Controller* (see page 610)

The following example will display the Biped Display Preferences dialog:

### Script Example

```
bipObj = biped.createNew 100 100 [0,0,0]
select bipObj
bip = bipObj.transform.controller
-- Display properties
biped.displayPrefsDlg bip
```

### See Also

*Biped Creation* (see page 603)

*Biped Controllers* (see page 610)

## Biped Sample Scripts

Here are the sample scripts that ship with **character studio 3**.

The custom user interface, **CS3Tools.cui**, **character studio 3** one-touch preset custom keys for rapid animation. It uses the MAXScripts: CS3Customkeys.ms, CS3convertBips.ms and CS3Bip2BonesFloater.ms and CS3CustomKeysPresetFloater.ms.

Please see the *CS3Tools.cui Tutorial* (see page 705).

An updated ../Cstudio/Docs/**Readcsm.ms**.

```
-- Fixed frame-off-by-1 bug by making keyframes start at 0 instead of 1, to match biped
-- Fixed frame rate interpretation - it was dropping mocap data correctly, but not skipping
max frames
```

../Cstudio/Docs/**csmxport.ms**

```
-- A script to output a .csm file
-- This script assumes that you have set up a .max file containing a biped or bone structure
with a set of markers linked to the skeleton.
-- The markers you want to record should be contained in a Selection Set titled "markers"
-- The script will output the data at every frame for the entire animation range.
```

---

## biped\_object : GeometryClass

This class represents the individual Biped body part and footstep nodes' baseobject.

### Properties

`<biped_object>.rootNode`                      Node                      Default: Varies                      -- Read-only

The controller for the `biped_object`. For the COM, the controller type is `Vertical_Horizontal_Turn`. For footsteps, the controller type is `FootSteps`. For the remaining biped body parts, the controller type is `BipSlave_Control`.

### See Also

*Vertical\_Horizontal\_Turn Matrix3 Controller (see page 610)*

*FootSteps : Matrix3 Controller (see page 619)*

*BipSlave\_Control Controllers (see page 620)*

---

## Biped Node Hierarchy

`biped.getNode <biped | biped_ctrl> <name | index> [link:<int_link>]`

Returns the specified limbnode where the second argument can be a named limb like `#larm`, `#rarm`, `#lfingers`, `#rfingers`, `#lleg`, `#rleg`, `#ltoes`, `#rtoes`, `#spine`, `#tail`, `#head`, `#pelvis`, `#footprints`, `#neck`, `#pony1`, `#pony2` or an integer index. . If you don't specify the link argument the top (first) node is returned. The only exception to this is for the biped COM, which does not contain any linked nodes (including itself). If the specified node does not exist, a value of undefined is returned.

The top level and their link nodes in **character studio 3** are:

| Index | Top Level  | Link Nodes in Link Index Order |            |             |            |
|-------|------------|--------------------------------|------------|-------------|------------|
| 1     | L Clavicle | L Clavicle                     | L UpperArm | L Forearm   | L Hand     |
| 2     | R Clavicle | R Clavicle                     | R UpperArm | R Forearm   | R Hand     |
| 3     | L Finger0  | L Finger0                      | L Finger01 | L Finger02  | L Finger1  |
|       |            | L Finger11                     | L Finger12 | L Finger2   | L Finger21 |
|       |            | L Finger22                     | L Finger3  | L Finger31  | L Finger32 |
|       |            | L Finger4                      | L Finger41 | L Finger42  |            |
| 4     | R Finger0  | R Finger0                      | R Finger01 | R Finger02  | R Finger1  |
|       |            | R Finger11                     | R Finger12 | R Finger2   | R Finger21 |
|       |            | R Finger22                     | R Finger3  | R Finger31  | R Finger32 |
|       |            | R Finger4                      | R Finger41 | R Finger42  |            |
| 5     | L Thigh    | L Thigh                        | L Calf     | L HorseLink | L Foot     |
| 6     | R Thigh    | R Thigh                        | R Calf     | R HorseLink | R Foot     |
| 7     | L Toe0     | L Toe0                         | L Toe01    | L Toe02     | L Toe1     |
|       |            | L Toe11                        | L Toe12    | L Toe2      | L Toe21    |
|       |            | L Toe22                        | L Toe3     | L Toe31     | L Toe32    |
|       |            | L Toe4                         | L Toe41    | L Toe42     |            |
| 8     | R Toe0     | R Toe0                         | R Toe01    | R Toe02     | R Toe1     |
|       |            | R Toe11                        | R Toe12    | R Toe2      | R Toe21    |
|       |            | R Toe22                        | R Toe3     | R Toe31     | R Toe32    |
|       |            | R Toe4                         | R Toe41    | R Toe42     |            |
| 9     | Spine      | Spine                          | Spine1     | Spine2      | Spine3     |
|       |            | Spine4                         |            |             |            |
| 10    | Tail       | Tail                           | Tail1      | Tail2       | Tail3      |
|       |            | Tail4                          |            |             |            |
| 11    | Head       | Head                           |            |             |            |
| 12    | Pelvis     | Pelvis                         |            |             |            |
| 13    | Biped COM  |                                |            |             |            |
| 14    | Biped COM  |                                |            |             |            |
| 15    | Biped COM  |                                |            |             |            |
| 16    | Footsteps  | Footsteps                      |            |             |            |
| 17    | Neck       | Neck                           | Neck1      | Neck2       | Neck3      |
|       |            | Neck4                          |            |             |            |
| 18    | Ponytail1  | Ponytail1                      | Ponytail11 | Ponytail12  | Ponytail13 |
|       |            | Ponytail14                     |            |             |            |
| 19    | Ponytail2  | Ponytail2                      | Ponytail21 | Ponytail22  | Ponytail23 |
|       |            | Ponytail24                     |            |             |            |

**Example**

```

bipObj = biped.createNew 100 100 [0,0,0] -- create a biped
nn = biped.maxNumNodes bipObj
nl = biped.maxNumLinks bipObj
for i = 1 to nn do
( anode = biped.getNode bipObj i
  if anode != undefined do
    ( format "% : \t%\n" i anode.name
      for j = 1 to nl do
        ( alink = biped.getNode bipObj i link:j
          if alink != undefined do
            format "% : % \t%\n" i j alink.name
          )
        )
      )
    )
  )
)

```

**Biped Node Hierarchy related methods**

`biped.maxNumNodes` <biped | biped\_ctrl>

Maximum nodes supported by Biped. . In **character studio 3**, this value is 19. In future versions of character studio, additional top level nodes may be present. See the description of `biped.getNode()` for a list of the top level nodes.

`biped.maxNumLinks` <biped | biped\_ctrl>

Maximum link nodes supported by Biped. In **character studio 3** this value is 16. In future versions of character studio, additional link nodes may be present. See the description of `biped.getNode()` for a list of the link nodes.

There are two levels of nodes in biped hierarchy, the top level ones are

- L Clavicle
- R Clavicle
- L Thigh
- R Thigh
- Head
- Spine
- Pelvis
- Footsteps
- Neck
- The rest of the nodes are links just like what you see in the structure rollout.
- UpperArm
- L Hand
- Finger2
- Calf
- Foot
- ...etc

You can get to left hand as follows: `biped.getNode $ #lArm link:4`



If the link argument is not specified then the top(first) node is returned.

---

## Biped Layers

### Layer Related Methods

`biped.collapseAtLayer <biped_ctrl> <index_integer>`

Collapses all layers till the specified layer. All underneath **must** be active for this function to work. Return true if successful, otherwise false.

`biped.createLayer <biped_ctrl> <index_integer> <name_string>`

Creates a layer at the specified position, maximum index value can be NumLayers + 1.

`biped.deleteLayer <biped_ctrl> <index_integer>`

Deletes the specified layer

`biped.getCurrentLayer <biped_ctrl>`

Returns the position of the currently active layer in the UI.

`biped.getLayerActive <biped_ctrl> <index_integer>`

Returns true if the specified layer is active.

`biped.getLayerName <biped_ctrl> <index_integer>`

Returns the specified layer name.

`biped.numLayers <biped_ctrl>`

Returns the number of layers.

`biped.setCurrentLayer <biped_ctrl> <index_integer>`

Sets the active layer in the UI to the specified layer.

`biped.setLayerActive <biped_ctrl> <index_integer> <boolean_val>`

Sets the specified layer to active/inactive based upon the boolean\_val passed.

`biped.setLayerName <biped_ctrl> <index_integer> <name_string>`

Sets the specified layer name to the value passed.

## Biped Controllers

The Biped controller, referred to as “<biped\_ctrl>” in this documentation, is called the *Biped Vertical\_Horizontal\_Turn(Body):Matrix3 Controller* (see page 610). This is sometimes also called the body controller because it is equivalent to the Biped’s.transform.controller.

The Biped footsteps have a controller. It is referred to as “<footsteps\_ctrl>” in this documentation. See *Biped Footsteps* (see page 621) for details regarding the FootSteps:Matrix3 Controller.

Individual Biped body part objects and the Vertical, Horizontal, and Turning tracks of the **Biped\_Vertical\_Horizontal\_Turn\_Body\_Matrix3\_Controller** use the **Biped\_Slave\_Controller**. This type of controller is attached to the transform track.

---

## Biped Vertical\_Horizontal\_Turn(Body):Matrix3 Controller

This controller is attached to the transform track of the Biped root objects.

### Constructor

```
biped_ctrl=bipobj.transform.controller
```

or:

```
<biped_ctrl>=$bip01.controller
```

### Properties

All the following properties are get/set properties unless specified.

#### -- Structure properties

```
<biped_ctrl>.rootName                                String                Default: BipXX,
```

where XX is a number

The name of the biped center of mass object. The center of mass (COM) is the root of the biped hierarchy, and is visible as a diamond shaped object in the biped pelvis area. The Root Name is appended to all the links of the biped hierarchy.

```
<biped_ctrl>.rootNode                                Node                    Default: Varies      -- Read-only
```

The root node (COM) of the Biped system this biped\_ctrl belongs to.

```
<biped_ctrl>.arms                                     Boolean                 Default: True
```

Specifies whether or not arms will be generated for the current biped.

```
<biped_ctrl>.neckLinks                               Integer                 Default: 1
```

range: 1-5

Specifies the number of links in the biped neck.

```
<biped_ctrl>.spineLinks                              Integer                 Default: 4
```

range: 1-5

Specifies the number of links in the biped spine.

```
<biped_ctrl>.legLinks          Integer      Default: 3
                                range: 3-4
```

Specifies the number of links in the biped legs.

```
<biped_ctrl>.tailLinks         Integer      Default: 0
                                range: 0-5
```

Specifies the number of links in the biped tail. A value of 0 specifies no tail.

```
<biped_ctrl>.ponytail1Links    Integer      Default: 0
                                range: 0-5
```

Specifies the number of Ponytail links.

```
<biped_ctrl>.ponytail2Links    Integer      Default: 0
                                range: 0-5
```

Specifies the number of Ponytail links.

```
<biped_ctrl>.fingers           Integer      Default: 1
                                range: 0-5
```

Specifies the number of biped fingers.

```
<biped_ctrl>.fingerLinks       Integer      Default: 3
                                range: 1-3
```

Sets the number of links per finger.

Note: If the number of fingers is 0, fingerLinks is always equal to 1.

```
<biped_ctrl>.toes              Integer      Default: 5
                                range: 1-5
```

Specifies the number of biped toes.

```
<biped_ctrl>.toeLinks           Integer      Default: 3
                                range: 1-3
```

Specifies the number of links per toe.

```
<biped_ctrl>.ankleAttach       Float        Default: 0.2
                                range: 0-1
```

Specifies the right and left ankles' point of attachment along the corresponding foot block. The ankles may be placed anywhere along the center line of the foot block from the heel to the toe.

A value of 0 places the ankle attachment point at the heel. A value of 1 places the ankle attachment point at the toes.

`<biped_ctrl>.height` Float Default: Defined at creation

Sets the height of the current biped.

`<biped_ctrl>.trianglePelvis` Boolean Default: True

Select this control to create links from the upper legs to the lowest biped spine object when Physique is applied. Normally, the legs are linked to the biped pelvis object.

The pelvis area can be a problem when the mesh is deformed with Physique. Triangle Pelvis creates a more natural spline for mesh deformation.

## -- Animation properties

`<biped_ctrl>.gravAccel` Float Default: 5.63855\*Body height

Sets the strength of the gravitational acceleration used to calculate the biped's motion.

By default, this parameter is set to accurately simulate Newtonian gravity as found on the Earth's surface.

`<biped_ctrl>.dynamicsType` Integer Default: 0

range: 0-1

### 0 – Biped Dynamics

Keys for the center of mass Balance Factor and Dynamics Blend parameters are set to a value of 1. Biped calculates airborne trajectories and biped balance.

### 1 – Spline Dynamics

Create new center of mass keys using full spline interpolation.

## Adapt Locks Group

Lock specified tracks to prevent automatic adjustments being made to those tracks when footsteps are moved in space or edited in time. All the locks except for Time work for footstep editing in space. Time, locks upper body keys when footsteps are edited in time (Track View). Adapt Locks only applies to a Footstep animation not a freeform animation. When you move a footstep in space or adjust footstep timing, Biped automatically adapts existing keyframes to match the new footsteps. Adapt locks allows you to preserve the exact position of already created keys for a selected track.

Adapt Locks does not need to be on all the time. For example, if you want to raise all the footsteps along the world Z-axis, without changing the upper body position, turn on Adapt Locks Body Vertical Keys, turn on Footstep mode, select all the footsteps and move them up along the world Z-axis. The footsteps are repositioned, the legs are adapted, but

the upper body retains the same motion rather than being raised with the footsteps. Now turn off Adapt Locks Body Vertical Keys, the upper body still retains its original motion.

`<biped_ctrl>.adaptLockFreeform` Boolean Default: False

Turn on to prevent adaptation of a freeform period in a footstep animation. The biped's position during a freeform period will not move if footsteps after the freeform period are moved further away.

`<biped_ctrl>.adaptLockHorz` Boolean Default: False

Turn on to prevent adaptation of body horizontal keys when footsteps are edited in space.

`<biped_ctrl>.adaptLockTurn` Boolean Default: False

Turn on to prevent adaptation of body turning keys when footsteps are edited in space.

`<biped_ctrl>.adaptLockVert` Boolean Default: False

Turn on to prevent adaptation of body vertical keys when footsteps are edited in space.

`<biped_ctrl>.adaptLockLLeg` Boolean Default: False

Turn on to prevent adaptation of left leg move keys (a leg move key, is a leg key between footsteps) when footsteps are edited in space.

`<biped_ctrl>.adaptLockRLeg` Boolean Default: False

Turn on to prevent adaptation of right leg move keys (a leg move key, is a leg key between footsteps) when footsteps are edited in space.

`<biped_ctrl>.adaptLockTime` Boolean Default: False

Use Adapt Locks Time to retain upper body motion while editing footstep duration in Track View. When the duration of a footstep is changed, the biped leg will adapt by re-timing the touch, plant and lift keys. The biped upper body keys will retain their exact motion.

## Separate Tracks Group

By default character studio stores a finger, hand, forearm, and upper-arm key in the clavicle track. The toe, foot and calf keys are stored in the thigh track. This optimized approach to key storage works well in most cases. If you need extra tracks, turn them on for a specific biped body part.

For example, turn on Arms if you plan on extensive finger-hand animation; if an arm key is deleted, it will not affect the finger-hand keys. Notice that in Track View a transform track is now available for the first link of the thumb (stores all finger keys), hand, forearm, and upper-arm.

`<biped_ctrl>.sepArmsTracks` Boolean Default: False

Turn on to create separate transform tracks for the finger, hand, forearm and upper-arm.

There is one finger track per hand. All finger keys are stored in the “Finger0” transform track, the first link of the biped thumb.

```
<biped_ctrl>.sepLegsTracks           Boolean      Default: False
```

Turn on to create separate toe, foot, and calf transform tracks.

```
<biped_ctrl>.sepPonytail1Tracks      Boolean      Default: False
```

Turn on to create separate ponytail 1 transform tracks.

```
<biped_ctrl>.sepPonytail2Tracks      Boolean      Default: False
```

Turn on to create separate ponytail 2 transform tracks.

#### Note:

If the number of pony tail links is 0, you cannot set Separate Ponytail Tracks to true.

```
<biped_ctrl>.sepNeckTracks           Boolean      Default: False
```

Turn on to create separate transform tracks for the neck links.

```
<biped_ctrl>.sepTailTracks           Boolean      Default: True
```

Turn on to create separate transform tracks for each tail link.

```
<biped_ctrl>.sepSpineTracks          Boolean      Default: False
```

Turn on to create separate spine transform tracks.

#### -- Modes

```
<biped_ctrl>.figureMode              Boolean      Default: False
```

-- cannot be in Buffer mode to enter figureMode

Use Figure mode to fit a biped to the mesh or mesh objects representing your character. Leave Figure Mode on when you attach the mesh to the biped with Physique. Figure mode is also used to scale a biped with a mesh attached, to make biped “fit” adjustments after Physique is applied, and to correct posture in motion files that need a global posture change.

```
<biped_ctrl>.footstepMode            Boolean      Default: False
```

Create and edit footsteps; generate a walk, run, or jump footstep pattern; edit selected footsteps in space; and append footsteps using parameters available in Footstep mode.

```
<biped_ctrl>.motionMode              Boolean      Default: False
```

-- cannot be in Buffer mode to enter motionMode

Create scripts and use editable transitions to combine *.bip* files together (to create character animation) in Motion Flow mode. After creating a script and editing transitions, use Save Segment on the General rollout to store a script as one long *.bip* file. Save a *.mfe* file, this enables you to continue Motion Flow work in progress.

```
<biped_ctrl>.bufferMode              Boolean      Default: False
```

Footsteps are required in the buffer to enter bufferMode.

Edit segments of an animation in Buffer mode. Copy footsteps and associated biped keys into the buffer using Copy Footsteps on the Footstep Operation rollout first, then turn on Buffer mode to view and edit the copied segment of your animation.

#### Note

Paste buffered motion back to the original animation repeatedly to create looping motions.

Edit footsteps and biped animation that has been copied into the buffer using Copy Footsteps on the Footsteps Operation rollout. The changes can be pasted back by turning off Buffer Mode, turning on Paste Footsteps on the Footstep Operation rollout and overlapping the buffered footsteps with the original footsteps. The buffered motion is spliced into the original animation.

`<biped_ctrl>.bendLinksMode` Boolean Default: False

Bend all the biped spine objects naturally by rotating a biped spine object. Bend Links also works for the biped tail and ponytail links.

`<biped_ctrl>.rubberBandMode` Boolean Default: False

Use this to reposition the biped elbows and knees without moving the biped hands or feet in Figure mode. Reposition the biped center of mass to simulate the physics of wind or weight pushing against the biped. Figure mode must be turned on to enable Rubber Band Mode.

#### Note

Rubber Band mode behaves differently than Non-Uniform Scale. If you “Rubber Band” the biped thigh, for example, the thigh and biped calf objects scale proportionally to keep the biped foot stationary. Using Non-Uniform Scale, the calf retains its scale and the foot moves.

`<biped_ctrl>.scaleStrideMode` Boolean Default: True

Footstep stride length and width are scaled to match the stride length and width of the biped figure. scaling occurs automatically when you load a *.bip*, *.stp*, or *.fig* file. When you paste footsteps; or when you scale the biped’s legs or pelvis.

`<biped_ctrl>.inPlaceMode` Boolean Default: False

-- cannot be in Figure mode to enter inPlaceMode

Keep the biped visible in the viewports while the animation plays. Use this for biped key editing or adjusting envelopes with Physique. Prevents XY movement of the biped center of mass during animation playback; motion along the Z-axis is preserved. In Place mode is stored with the 3DS MAX file.

`<biped_ctrl>.inPlaceXMode` Boolean Default: False

-- cannot be in Figure mode to enter inPlaceXMode

Lock center of mass X-axis motion. Use this for game export where the character stays in place but the swinging motion of the hips and upper body along the Y-axis is preserved.

`<biped_ctrl>.inPlaceYMode` Boolean Default: False

-- cannot be in Figure mode to enter inPlaceYMode

Lock center of mass Y-axis motion. Use this for game export where the character stays in place but the swinging motion of the hips and upper body along the X-axis is preserved. Biped limbs, footsteps, and center of mass keys can be adjusted using In Place mode (when the center of mass is moved on the XY-axes in this mode, the footsteps move). View biped playback without requiring a follow camera. In this viewing mode, visible footsteps “slide” under the biped.

For export to games, this feature is valuable since many game engines intelligently move the character’s center of mass laterally according to game play. In Place mode makes it easy to view, tune, and export animation in a manner that is complimentary to game engine playback.

## Note

Trajectories do not display when In Place mode is active.

## -- Track Selection

`<biped_ctrl>.trackSelection` Integer Default: 0

0 - No track selection

While a **trackSelection** value of 0 is a valid return value, setting trackSelection to 0 is meaningless and will not change the current track selection.

1 - Body Horizontal

Turn on to prevent adaptation of body horizontal keys when footsteps are edited in space.

2 - Body Vertical

Turn on to prevent adaptation of body vertical keys when footsteps are edited in space.

3 - Body Rotation

Turn on to prevent adaptation of body turning keys when footsteps are edited in space.

## -- Display properties

`<biped_ctrl>.displayBones` Boolean Default: False

Displays biped bones. Bones are represented as yellow lines, which do not render. Selecting Bones is useful for seeing exactly where the joints fall in relation to the biped objects.

`<biped_ctrl>.displayObjects` Boolean Default: True

Displays biped body objects (objects).

`<biped_ctrl>.displayFootsteps` Boolean Default: True

Displays biped footsteps in the viewport. Footsteps are represented as green and blue foot-shaped outlines by default; these are also visible in preview renderings. Turning off the Footsteps button also turns off the footstep numbers and the center of mass shadow.

`<biped_ctrl>.displayFSNumbers` Boolean Default: True



Displays biped footstep numbers. Footstep numbers specify the order in which the biped will move along the path created by the footsteps. Footstep numbers are displayed in white and do not render, but do appear in preview renderings.

`<biped_ctrl>.displayTrajectories` Boolean Default: False

Displays trajectories for selected biped limbs.

#### -- Layers

`<biped_ctrl>.visibleBefore` Integer Default: 1

Set the number of preceding layers too display as stick figures.

`<biped_ctrl>.visibleAfter` Integer Default: 0

Set the number of succeeding layers too display as stick figures.

`<biped_ctrl>.keyHighlight` Boolean Default: False

Display keys by highlighting the stick figures.

#### -- Footstep related properties

`<biped_ctrl>.fsAppendState` Boolean Default: False

Each new footstep is appended to the end of the biped's footstep sequence.

`<biped_ctrl>.fsCreateState` Boolean Default: False

Set to true to create new states. The first state you add is, by default, the first state in the controller that executes when the simulation is run.

`<biped_ctrl>.fsGroundDuration` Time Default: 18f

`<biped_ctrl>.fsAirDuration` Time Default: 3f

Specifies the number of frames when the body will be in the air during a run or a jump.

#### Note

If **fsGaitMode** is #walk, the above 2 parameters are the Walk Footstep and Double Support durations, respectively. If **fsGaitMode** is #run, the above 2 parameters are the Run Footstep and Airborne durations, respectively. If **fsGaitMode** is #jump, the above 2 parameters are the 2-Foot down and Airborne durations, respectively.

`<biped_ctrl>.fsGaitMode` Name Default: #walk

`fsGaitMode` can be set to any of the three gaits: #walk, #run, #jump

#### -- Motion Flow properties

`<biped_ctrl>.motionFlow` MoFlow

Returns an instance of **MoFlow**. See *Biped Motion Flow (see page 628)* section for more details.

#### -- Motion Capture properties

`<biped_ctrl>.displayBuffer` Boolean Default: False

A red stick figure appears, representing the raw motion capture data.

```
<biped_ctrl>.displayBufferTraj      Boolean      Default: False
```

Display a trajectory based on the buffered raw motion capture data for any biped body part. Use this in combination with Show/Hide Trajectories on the Display rollout to see how closely the raw and filtered data match.

```
<biped_ctrl>.talentFigMode          Boolean      Default: False
```

Turn on Talent Figure mode to scale the biped relative to the markers. Calibration for the entire marker file takes place when you exit Talent Figure mode.

Keyframe adaptation takes place in order to accommodate the new biped scale; because of this, you should adjust the biped scale before adjusting the biped position relative to the markers.

Use Rubber Band Mode on the General rollout and Non-Uniform Scale to size the biped in Talent Figure mode.

Ideally, you will not need to use this feature. When loading a motion capture file, Biped attempts to extract the appropriate figure scale from the given data. Use Talent Figure mode only if the extracted scale of the biped doesn't match the scale of the original talent. Minor differences in scale will alter the motion.

#### Note

Calibration controls are only enabled when a marker or *.bvh* file is imported in its raw form. Do not use key reduction or extract footsteps when you import a marker file for the first time.

#### -- Object Space Object

```
<biped_ctrl>.osObject                Node          Default: Undefined
```

The Object Space Object for the currently selected body part. The **osObject** can be specified for the Clavicle or any of its decendents, and the Thigh or any of its decendents. The **osObject** is undefined for all other body parts. The **osObject** is normally specified in the IK Key Info rollout.

#### -- SubAnim Properties (as seen in Track View)

```
<biped_ctrl>.vertical                Matrix3      -- Animatable
```

The vertical track of controller

```
<biped_ctrl>.horizontal              Matrix3      -- Animatable
```

The horizontal track of controller

```
<biped_ctrl>.turning                 Matrix3      -- Animatable
```

The turning track of controller

```
<biped_ctrl>.Ease_Curve              Point3       -- Animatable
```

Animated over range 0 to 1

**Script Example**

```

bipObj = biped.createNew 100 100 [0,0,0]
select bipObj
bip = bipObj.transform.controller
-- Display properties
bip.displayBones = true
bip.displayObjects = false
bip.displayFootsteps = false
bip.displayFSNumbers = false
-- Animations properties
bip.gravAccel = 700
bip.dynamicsType = 1

```

**See Also**

*Biped Creation (see page 603)*

---

## FootSteps : Matrix3 Controller

**Constructor**

```
<footsteps_ctrl> = '$Bip01 Footsteps'.transform.controller
```

or:

```
<footsteps_ctrl>=$'Bip01 Footsteps'.controller
```

**Notes**

This controller is attached to the transform track of the named Biped footstep object.

**Properties**

|                                    |         |                |
|------------------------------------|---------|----------------|
| <footsteps_ctrl>.freeFormMode      | Boolean | Default: False |
| false - Edit Footsteps             |         |                |
| true - Edit Free Form (no physics) |         |                |
| <footsteps_ctrl>.dispNumType       | Integer | Default: 0     |
| 0 - Start and End Frame            |         |                |
| 1 - Start Frame                    |         |                |
| 2 - Duration                       |         |                |
| 3 - Double Support                 |         |                |
| <footsteps_ctrl>.dispAirDur        | Boolean | Default: False |

Displays the foot air duration. This is the number of frames each foot is in the air in the lower part of each footstep's portion of the track.

|                                |         |                |
|--------------------------------|---------|----------------|
| <footsteps_ctrl>.dispNoSupport | Boolean | Default: False |
|--------------------------------|---------|----------------|

Displays the number of frames when both feet are off the ground, directly above the areas without footsteps.

|                           |        |                 |
|---------------------------|--------|-----------------|
| <footsteps_ctrl>.rootName | String | Default: Varies |
|---------------------------|--------|-----------------|

Contains the root name of the Biped using the Footsteps Controller. Changing the value of this property also changes the root name of the Biped.

`<footsteps_ctrl>.rootNode`                      Node                      Default: Varies                      -- Read-only

The root node (COM) of the Biped system this footstep\_ctrl belongs to.

The following example will access the footstep controller, set several values and then display all of its properties.

#### Script Example

```
fsCont = '$Bip01 Footsteps'.transform.controller
fsCont.freeFormMode = true
fsCont.dispNumType = 3
fsCont.dispAirDur = false
fsCont.dispNoSupport = false
showProperties fsCont
```

#### See Also

*Biped Footprints (see page 621)*

*Biped Keys (see page 636)*

---

## Biped Slave Controller

Individual Biped body part objects and the Vertical, Horizontal, and Turning tracks of the *Biped\_Vertical\_Horizontal\_Turn\_Body\_Matrix3\_Controller* (see page 610) use the Biped\_Slave ControllerBiped\_Slave\_Controller. This type of controller is attached to the transform track.

`<bipslave_ctrl>=$'Bip01 Pelvis'.controller`

It is not always possible to get the controller via `<biped_object>.transform.controller`. The transform track is not exposed for all nodes but it is possible to get the controller via: `<biped_object>.controller`.

#### Properties

`<bipslave_ctrl>.rootName`                      String                      Default: Varies

The name of the root node (COM) of the Biped system this bipslave\_ctrl belongs to. Note: changing the value of this property also changes the root name of the biped.

`<bipslave_ctrl>.rootNode`                      Node                      Default: Varies                      -- Read-only

The root node (COM) of the Biped system this bipslave\_ctrl belongs to.

## Biped Footsteps and Footprints

This topic will cover the Biped's footsteps controller, individual footprints, the multiple footprint creation dialog, as well as obtaining and setting the parameters for multiple footsteps.

*Biped Footsteps (see page 619)*

*Biped Footprints (see page 621)*

## Biped Footprints

### Methods

`biped.addFootprint <biped_ctrl> <matrix3> [append:<boolean>]`

Specifies the position and rotation of the footprint. The scale portion of the matrix3 should always be the identity matrix ([1,1,1]).

Creates a single footprint for the biped, where matrix3 specifies the position and rotation of the footprint. The scale portion of the matrix3 should always by unity ([1,1,1]).

### Notes

Creates a single footprint for the Biped.

### Global Properties

`biped.fsAddSide` Integer Default:0

0 - Start Right

1 - Start Left

### Notes

For multiple footprint creation, the starting foot needs to be identified. There is a UI radio button in the Multiple Footstep Creation dialog "Start Right/Start Left". The biped global property, **fsAddSide**, exposes access to this UI element.

This property also effects the start foot when manually creating footsteps in the viewports and when using `biped.addFootprint`. If the `biped.fsAddSide` is 1, a left footprint is created.

If 0, a right footprint is created. When you create a footprint in the viewports or by using `biped.addFootprint`, the value of `biped.fsAddSide` is flipped, resulting in alternating footsteps being created by default.

### Methods

`biped.multipleFSDlg <biped_ctrl>`

Displays the Multiple Footstep Creation dialog

`biped.getMultipleFSParams <gait_type_name>`

Returns an instance of `MultFprintParams`, `gait_type_name` can be **#walk**, **#run** or **#jump**. For information on `MultFprintParams` and `gait_type_name` see `Biped MultFprintParams ClassBiped_MultFprintParams_Class`.

The following example will obtain the MultipleFSParams for a walk cycle, set several of its values and then show all of the properties.

### Script Example

```
walk = biped.getMultipleFSParams #walk
walk.numFootsteps = 5
walk.actualStrideWidth = 2.5
walk.paramStrideWidth = 2.5
walk.actualStrideLength1 = .6
walk.paramStrideLength1 = .6
walk.actualStrideHeight1 = .6
walk.cycle1 = 6
walk.actualStrideLength2 = .7
walk.paramStrideLength2 = .7
walk.actualStrideHeight2 = .7
walk.cycle2 = 7
walk.autoTiming = true
walk.interpTiming = true
walk.alternate = true
walk.multiInsertInTime = true
showProperties walk
```

### Method

```
biped.addMultipleFootprints <biped_ctrl> <MultFprintParams>
```

Create footsteps for a Biped based on the MultFprintParams value

```
biped.newFootprintKeys <biped_ctrl>
```

Activates all inactive footsteps. Activation creates default keys for any footsteps that do not have them. If a footstep does not have keys, it is displayed as bright green (right foot) or bright blue (left foot). After keys are created for the footsteps, the footsteps change color to pastel green and pastel blue.

Create Biped keys for inactive footsteps on the Biped

The following example will create a biped, create 10 footsteps and then have Biped create default keys for the footsteps.

### Script Example

```
-- create a Biped
bipObj = biped.createNew 100 100 [0,0,0]
-- get the transform controller for the Biped
bip = bipObj.transform.controller
-- get the multiple footstep parameters interface
mfsf=biped.getMultipleFSParams #walk
-- set the number of footsteps to 10
mfsf.numFootsteps=10
-- create the inactive footsteps
biped.addMultipleFootprints bip mfsf
-- create the Biped keys for inactive footsteps
```

```
biped.newFootprintKeys bip
```

## See Also

*Biped Controllers (see page 610)*

*Biped MultFprintParams Class (see page 623)*

*Biped Keys (see page 636)*

---

## Biped Class : MultFprintParams

This class represents multiple footstep creation parameters of a Biped. An instance of this class is returned by the `biped.getMultipleFSParams <gait_type_name>` method, where `gait_type_name` can be `#walk`, `#run` or `#jump`.

### Properties

|                                                                                                                                                                                                                                                     |         |          | #walk | #run  | #jump |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----------|-------|-------|-------|
| <MultFprintParams>. <b>alternate</b>                                                                                                                                                                                                                | Boolean | Default: | true  | true  | true  |
| Footsteps will alternate between right and left. Alternate is always selected when the Walk gait is selected. You can only clear Alternate when Run or Jump gaits are selected.                                                                     |         |          |       |       |       |
| <MultFprintParams>. <b>numFootsteps</b>                                                                                                                                                                                                             | Integer | Default: | 4     | 4     | 4     |
| Determines the number of new footsteps to be created.                                                                                                                                                                                               |         |          |       |       |       |
| <MultFprintParams>. <b>paramStrideWidth</b>                                                                                                                                                                                                         | Float   | Default: | 1.0   | 1.0   | 1.0   |
| Sets the stride width as a percentage of the pelvis width. A value of 1.0 produces a stride width equal to the pelvis width. A value of 3.0 produces a wide, waddling stride. Changes to this setting automatically change the Actual Stride Width. |         |          |       |       |       |
| Parametric describes the parameter in terms of biped anatomy, and Actual describes the value in 3DS MAX units.                                                                                                                                      |         |          |       |       |       |
| <MultFprintParams>. <b>actualStrideWidth</b>                                                                                                                                                                                                        | Float   | Default: | 0.0   | 0.0   | 0.0   |
| Sets the stride width in modeling units. Changes to this setting automatically change the Parametric Stride Width.                                                                                                                                  |         |          |       |       |       |
| <MultFprintParams>. <b>autoTiming</b>                                                                                                                                                                                                               | Boolean | Default: | true  | true  | true  |
| Sets timing parameters automatically. Auto Timing affects the following timing parameters for the Walk gait:                                                                                                                                        |         |          |       |       |       |
| Walk footstep, Double Support                                                                                                                                                                                                                       |         |          |       |       |       |
| When Auto Timing is selected, these parameters are automatically adjusted to reasonable values. Control the footstep sequence by adjusting the Stride Length and Time to Next Footstep parameters.                                                  |         |          |       |       |       |
| When Auto Timing is cleared, you can control the footstep sequence by adjusting the gait timing parameters, but you can't change the Time to Next Footstep parameter.                                                                               |         |          |       |       |       |
| <MultFprintParams>. <b>interpTiming</b>                                                                                                                                                                                                             | Boolean | Default: | false | false | false |
| Control acceleration or deceleration of the series of footsteps.                                                                                                                                                                                    |         |          |       |       |       |
| <MultFprintParams>. <b>multiInsertInTime</b>                                                                                                                                                                                                        | Boolean | Default: | false | false | false |

**false - Start after last footstep**

Appends the newly created footsteps to the end of the existing footstep sequence.

**true - Start at current frame**

Inserts the newly created footsteps at the current frame after the existing footstep sequence allowing you to make a gap in time before the footsteps start again.

```
<MultFprintParams>.actualStrideLength1 Float Default: 0.0 0.0 0.0
```

Sets the stride length for new footsteps in 3DS MAX units.

```
<MultFprintParams>.paramStrideLength1 Float Default: 0.75 1.5 2.4
```

Sets the stride length for the new footsteps as a percentage of the length of the biped's leg. The default value of 0.75 gives an average stride of normal proportions.

A value of 1.0 will produce a stride length equal to the leg length, which makes the biped stretch slightly to reach the next step. A value of 0.0 will make the biped walk in place. A negative stride length will make the biped walk backwards.

When a biped walks backwards, it does not simply reverse the forward movement but maintains the correct foot-state sequence with the toe touching the ground first, followed by the heel.

Adjusting Parametric Stride Length automatically changes the value for Actual Stride Length.

```
<MultFprintParams>.actualStrideHeight1 Float Default: 0.0 0.0 0.0
```

Sets the rise or fall between footsteps. You can use this parameter to create a set of footsteps going up or down a slope or a stairway.

The value for Actual Stride Height is the difference in height in units between each of the new footsteps. Positive values step up and negative values step down.

```
<MultFprintParams>.cycle1 Time Default: 15f 15f 15f
```

```
<MultFprintParams>.actualStrideLength2 Float Default: 0.0 0.0 0.0
```

Sets the stride length for new footsteps in 3DS MAX units. The same rules apply as for Parametric Stride Length. Adjusting Actual Stride Length automatically changes the value for Parametric Stride Length.

```
<MultFprintParams>.paramStrideLength2 Float Default: 0.75 1.5 2.4
```

Sets the stride length for the new footsteps as a percentage of the length of the biped's leg. The default value of 0.75 gives an average stride of normal proportions.

A value of 1.0 will produce a stride length equal to the leg length, which makes the biped stretch slightly to reach the next step. A value of 0.0 will make the biped walk in place. A negative stride length will make the biped walk backwards. When a biped walks backwards, it does not simply reverse the forward movement but maintains the correct foot-state sequence with the toe touching the ground first, followed by the heel.

Adjusting Parametric Stride Length automatically changes the value for Actual Stride Length.

```
<MultFprintParams>.actualStrideHeight2 Float Default: 0.0 0.0 0.0
```



Sets the rise or fall between footsteps. You can use this parameter to create a set of footsteps going up or down a slope or a stairway.

The value for Actual Stride Height is the difference in height in units between each of the new footsteps. Positive values step up and negative values step down.

```
<MultFprintParams>.cycle2           Time           Default:   15f      15f      15f
```

See the `biped.addMultipleFootprints` method for creating footsteps based on a `MultFprintParams`, and `biped.newFootprintKeys` creating the biped keys for the inactive footsteps.

Since `MultFprintParams` is biped independent, the actual stride lengths and heights are not accessible in this class. To convert from a `paramStrideWidth` value to an `actualStrideWidth` value, multiply the `paramStrideWidth` value by the width of the pelvis. Given a `biped_ctrl`, the pelvis width can be calculated by:

```
thePelvis = biped.getNode biped_ctrl #pelvis
```

```
pelvisWidth = in coordsys thePelvis (thePelvis.max- thePelvis.min).z
```

To convert from a `paramStrideLength` value to an `actualStrideLength` value, multiply the `paramStrideLength` value by the length of the right leg. Given a `biped_ctrl`, the length of the right leg can be calculated by:

```
legLength = distance (biped.getNode biped_ctrl #rleg link:1) (biped.getNode  
biped_ctrl #rleg link:2) -- thigh
```

```
legLength += distance (biped.getNode biped_ctrl #rleg link:2) (biped.getNode  
biped_ctrl #rleg link:3) -- calf
```

```
if ((biped_ctrl.rootnode).controller.leglinks) > 3 do
```

```
    legLength += distance (biped.getNode biped_ctrl #rleg link:3) (biped.getNode  
biped_ctrl #rleg link:4) - horsellink, if present
```

## See Also

*Biped Vertical\_Horizontal\_Turn(Body):Matrix3 Controller (see page 610)*

*Biped Slave Controller (see page 620)*

*FootSteps : Matrix3 Controller (see page 619)*

*Biped Footsteps (see page 621)*

## FootSteps : Matrix3 Controller

### Constructor

```
<footsteps_ctrl> = $('Bip01 Footsteps').transform.controller
```

or:

```
<footsteps_ctrl>=$'Bip01 Footsteps'.controller
```

### Notes

This controller is attached to the transform track of the named Biped footprint object.

### Properties

|                                                  |         |                |
|--------------------------------------------------|---------|----------------|
| <code>&lt;footsteps_ctrl&gt;.freeFormMode</code> | Boolean | Default: False |
| false - Edit Footsteps                           |         |                |
| true - Edit Free Form (no physics)               |         |                |

|                                                 |         |            |
|-------------------------------------------------|---------|------------|
| <code>&lt;footsteps_ctrl&gt;.dispNumType</code> | Integer | Default: 0 |
| 0 - Start and End Frame                         |         |            |
| 1 - Start Frame                                 |         |            |
| 2 - Duration                                    |         |            |
| 3 - Double Support                              |         |            |

|                                                |         |                |
|------------------------------------------------|---------|----------------|
| <code>&lt;footsteps_ctrl&gt;.dispAirDur</code> | Boolean | Default: False |
|------------------------------------------------|---------|----------------|

Displays the foot air duration. This is the number of frames each foot is in the air in the lower part of each footprint's portion of the track.

|                                                   |         |                |
|---------------------------------------------------|---------|----------------|
| <code>&lt;footsteps_ctrl&gt;.dispNoSupport</code> | Boolean | Default: False |
|---------------------------------------------------|---------|----------------|

Displays the number of frames when both feet are off the ground, directly above the areas without footprints.

|                                              |        |                 |
|----------------------------------------------|--------|-----------------|
| <code>&lt;footsteps_ctrl&gt;.rootName</code> | String | Default: Varies |
|----------------------------------------------|--------|-----------------|

Contains the root name of the Biped using the Footsteps Controller. Changing the value of this property also changes the root name of the Biped.

|                                              |      |                 |              |
|----------------------------------------------|------|-----------------|--------------|
| <code>&lt;footsteps_ctrl&gt;.rootNode</code> | Node | Default: Varies | -- Read-only |
|----------------------------------------------|------|-----------------|--------------|

The root node (COM) of the Biped system this footprint\_ctrl belongs to.

The following example will access the footprint controller, set several values and then display all of its properties.

### Script Example

```
fsCont = $('Bip01 Footsteps').transform.controller
fsCont.freeFormMode = true
fsCont.dispNumType = 3
fsCont.dispAirDur = false
fsCont.dispNoSupport = false
showProperties fsCont
```

See Also

*Biped Footprints (see page 621)*

*Biped Keys (see page 636)*

---

BipedFSKey : MAXObject

This class represents a Biped footstep in the viewports and trackview.

Properties

|                                                                                                       |         |                          |    |
|-------------------------------------------------------------------------------------------------------|---------|--------------------------|----|
| <fskey>.time                                                                                          | Time    | Default: Varies          |    |
| A value indicating when in time the key occurs.                                                       |         |                          |    |
| <fskey>.duration                                                                                      | Time    | Default: Varies          |    |
| The number of frames in each footstep.                                                                |         |                          |    |
| <fskey>.selected                                                                                      | Boolean | Default: False           |    |
| <fskey>.edgeSel                                                                                       | Integer | Default: 0               |    |
| 0 - no edges selected<br>1 - left edge Selected<br>2 - right edge selected<br>3 - both edges Selected |         |                          |    |
| <fskey>.active                                                                                        | Boolean | Default: True            |    |
| Activate or make inactive the footstep.                                                               |         |                          |    |
| <fskey>.transform                                                                                     | Matrix3 | Default: Varies          |    |
| <fskey>.side                                                                                          | Name    | Default: #left or #right | -- |
| Read-only                                                                                             |         |                          |    |

See Also

*Biped Controllers (see page 610)*

*Biped Footsteps (see page 621)*

*Biped Vertical\_Horizontal\_Turn(Body):Matrix3 Controller (see page 610)*

*Biped Keys (see page 636)*

## Biped Motion Flow

*MoFlow* : *MaxWrapper* (see page 628)

*MoFlowScript* : *MaxWrapper* (see page 630)

*MoFlowSnippet* : *MaxWrapper* (see page 631)

*MoFlowTranInfo* : *MaxWrapper* (see page 633)

*MoFlowTransition* : *MaxWrapper* (see page 635)

## MoFlow : MaxWrapper

This class can be used to access features in the Biped Motion Flow panel. An instance of this class is returned by the `.motionFlow` property of a *Biped Body Controller* (see page 610).

### Properties

`<moflow>.scripts` Array Default: #()

-- Read-only

An array of motion flow scripts (MoFlowScript values).

A Script is a list of clips (*.bip* files) that are executed sequentially to animate a character.

`<moflow>.activeScript` MoFlowScript Default: undefined

Get/set which MoFlowScript in scripts array is the active script. If the specified MoFlowScript is not in the scripts array, no action occurs.

`<moflow>.snippets` Array Default: #()

-- Read-only

An array of motion flow snippets (MoFlowSnippet values).

`<moflow>.selSnippets` BitArray Default: #{}

A bitarray specifying the motion flow snippets that are selected in the Motion Flow Graph.

`<moflow>.startFrame` Integer Default: 0

-- Read-only

The start frame of currently active motion flow script.

`<moflow>.endFrame` Integer Default: 0

-- Read-only

The last frame of correctly active motion flow script.

### Methods

`loadMoFlowFile` `<moflow>` `<file_name>` [ `quiet:<boolean>` ]

If **quiet:true**, which is the default, any warning message dialogs are suppressed.

Load a Motion Flow Editor file (.mfe). Motion Flow Editor files include:

**Clips:** references to biped animation files.

**Transitions:** The names, attributes, and connections between clips.

**Scripts:** different paths through a set of connected clips and transitions.

**saveMoFlowFile** <moflow> <file\_name>

Save a Motion Flow Editor file (.mfe).

Load/Save a motion flow (.MFE) file.

## Note

The location of the referenced *.bip* files is saved in the *.mfe* file. If the *.bip* file cannot be found, the program looks to the motion flow directory specified in *\plugcfg\biped.ini*. By default, this setting is *MoFlowDir=<maxdir>\cstudio\scripts*

If a referenced *.bip* file cannot be found in its current location, you will need to move it to the specified Motion Flow directory. You can change the location of this directory at any time by editing your *biped.ini* file with a text editor. The new directory will be used the next time you restart 3D Studio MAX. You can also add multiple search paths to your *biped.ini* file by repeating the *MoFloDir=* line multiple times. The program will search the directories in the order they appear and will use the first instance of the file that it finds.

**loadSnippetFiles** <moflow>

Loads the snippet files whose file names are assigned. This function should be called whenever new snippets are added.

Clips represent all or part of a *.bip* file.

**addScript** <moflow> <name>

Scripts represent different paths through the clips in the Motion Flow Graph.

Creates a new motion flow script with the given name and adds it to the motion flow.

Returns the new MoFlowScript.

**deleteScript** <moflow> <MoFlowScript>

**deleteScript** <moflow> <index\_integer>

Deletes the specified script. If the second argument is an integer, the script deleted is the indexed script in the motion flow's *.scripts* array.

**getScriptIndex** <moflow> <MoFlowScript>

Given the script, returns its index in the script's combobox.

**getSnippetIndex** <moflow> <MoFlowSnippet>

Given the snippet, returns its index in the in the motion flow's *.snippets* array.

**computeAnimation** <moflow> [**redraw:<true>**] [**incGlobals:<false>**]

Computes the global flow network. This function has to be called to update any changes made to the motion flow network. **redraw:true** will redraw the viewports. **incGlobals:true** will also include the global motion flow network.

### Related Methods

**newSnippet** <filename> <point2\_pos> <redraw:true> <load:true>

Adds a new MoFlowSnippet to the motion flow network, from the given .bip file. <point2\_pos> is the position in windows coordinates where the origin is the top left of the snippet in the motion flow graph. **Redraw:true** will redraw the graph window. **Load:true** will immediately load the snippet

### See also

*Biped Controllers (see page 610)*

*MoFlowScript : MaxWrapper (see page 630)*

*MoFlowSnippet : MaxWrapper (see page 631)*

---

## MoFlowScript : MaxWrapper

### Constructor

`addScript <moflow> <script_name>`

Create a new motion flow script with the given name to the motion flow. Returns the new MoFlowScript.

### Properties

|                                                                 |         |                  |              |
|-----------------------------------------------------------------|---------|------------------|--------------|
| <MoFlowScript>.startFrame                                       | Integer | Default: 0f      |              |
| <MoFlowScript>.snippets                                         | Array   | Default: #()     | -- Read-only |
| Array of MoFlowSnippet values defined in the motion flow script |         |                  |              |
| <MoFlowScript>.name                                             | String  | Default: Varies  |              |
| <MoFlowScript>.startPos                                         | Point3  | Default: [0,0,0] |              |
| <MoFlowScript>.startRot                                         | Float   | Default: 0       |              |

### Methods

**addSnippet** < MoFlowScript > < MoFlowSnippet >

Adds the specified MoFlowSnippet to a motion flow script. Returns the MoFlowSnippet value.

**insertSnippet** < MoFlowScript > < MoFlowSnippet > <index\_integer>

Inserts the snippet at the location specified and returns the new script item. Returns the MoFlowSnippet value.

**deleteSnippet** < MoFlowScript > <index\_integer>

Deletes the indexed MoFlowSnippet from the motion flow script.

### Related Methods

**deleteScript** <moflow> <index\_integer>

Deletes the indexed MoFlowScript from the motion flow.

**computeAnimation** <moflow> [redraw:<true>] [incGlobals:<false>]

Computes the global flow network. This function has to be called to update any changes made to the motion flow network. redraw:true will redraw the viewports. incGlobals:true will also include the global motion flow network

### NotesC

Changes to the MoFlowScript property values do not cause an immediate update of the biped. ComputeAnimation must be called on the MoFlow value to recompute the biped motion.

### Constructor

**newSnippet** <moflow> <filename> <point2\_pos> <redraw:true> <load:true>

Adds a new MoFlowSnippet to the motion flow network, from the given .bip file.

<point2\_pos> is the position in windows coordinates where the origin is the top left of the snippet in the motion flow graph. **Redraw:true** will redraw the graph window. **Load:true** will immediately load the snippet. Returns the new MoFlowSnippet.

### Properties

|                       |         |            |
|-----------------------|---------|------------|
| <MoFlowSnippet>.start | Integer | Default: 0 |
|-----------------------|---------|------------|

The start frame in the snippet file.

|                     |         |                 |
|---------------------|---------|-----------------|
| <MoFlowSnippet>.end | Integer | Default: Varies |
|---------------------|---------|-----------------|

The end frame in the snippet file.

|                          |        |                 |
|--------------------------|--------|-----------------|
| <MoFlowSnippet>.clipName | String | Default: Varies |
| <MoFlowSnippet>.fileName | String | Default: Varies |
| <MoFlowSnippet>.pos      | Point2 | Default: Varies |

The coordinates of the snippet in the Motion Flow Graph.

|                             |         |                                                  |
|-----------------------------|---------|--------------------------------------------------|
| <MoFlowSnippet>.active      | Boolean | Default: True                                    |
| <MoFlowSnippet>.transitions | Array   | Default: #(MoFlowTransition : [X,X]) - read only |

An array of the transitions defined for the snippet (MoFlowTransition values). The MoFlowTransition values printed show the from and to MoFlowSnippet names.

|                                        |         |              |
|----------------------------------------|---------|--------------|
| <MoFlowSnippet>.randomStartProbability | Integer | Default: 100 |
|----------------------------------------|---------|--------------|

### Methods

**addTransition** <from\_MoFlowSnippet > <to\_MoFlowSnippet > <bool\_optimize>

Adds a new MoFlowTransition to a motion flow Snippet. The optimize parameter acts as the “Optimize Selected Transition” in the Motion Flow Editor.

**deleteTransition** < MoFlowSnippet > <index\_integer>

Deletes the indexed MoFlowTransition emanating from the MoFlowSnippet.

**deleteTransitionTo** <from\_MoFlowSnippet > <to\_MoFlowSnippet >

Deletes the transition that’s emanating to the specified snippet.

### Related Methods

**addSnippet** < MoFlowScript > < MoFlowSnippet >

Adds the specified MoFlowSnippet to a motion flow script. Returns the MoFlowSnippet value.

**insertSnippet** < MoFlowScript > < MoFlowSnippet > <index\_integer>

Inserts a snippet at the location specified and returns the new script item. Returns the MoFlowSnippet value.

**deleteSnippet** < MoFlowScript > <index\_integer>

Deletes the indexed MoFlowSnippet from the motion flow script.

**loadSnippetFiles** <moflow>

Loads all of the snippet files whose file names are assigned. This function should be called whenever new snippets are added.

**getSnippetIndex** <moflow> <MoFlowSnippet>

Given the snippet, returns its index in the motion flow’s **.snippets** array.

**computeAnimation** <moflow> [redraw:<true>] [incGlobals:<false>]

Computes the global flow network. This function has to be called to update any changes made to the motion flow network. redraw:true will redraw the viewports. incGlobals:true will also include the global motion flow network

### Notes

Changes to the MoFlowSnippet property values do not cause an immediate update of the biped. ComputeAnimation must be called on the MoFlow value to recompute the biped motion.



## MoFlowTranInfo : MaxWrapper

### Constructor

**addTranInfo** < MoFlowTransition >

Adds a new MoFlowTranInfo to the MoFlowTransition and returns the newly created MoFlowTranInfo.

### Properties

|                                                 |         |                 |
|-------------------------------------------------|---------|-----------------|
| <code>&lt;MoFlowTranInfo&gt;.length</code>      | Integer | Default: 25     |
| The transition length in frames.                |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.angle</code>       | Float   | Default: 0.0    |
| The direction of the destination clip           |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.easeFrom</code>    | Float   | Default: 0.5    |
| <code>&lt;MoFlowTranInfo&gt;.easeTo</code>      | Float   | Default: 0.5    |
| <code>&lt;MoFlowTranInfo&gt;.sourceStart</code> | Integer | Default: Varies |
| The start frame for the source clip             |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.destStart</code>   | Integer | Default: Varies |
| The start frame for the destination clip        |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.sourceState</code> | Boolean | Default: True   |
| false - Fixed<br>true - Rolling                 |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.destState</code>   | Boolean | Default: True   |
| false – Fixed<br>true – Rolling                 |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.length</code>      | Integer | Default: 25     |
| The transition length in frames.                |         |                 |
| <code>&lt;MoFlowTranInfo&gt;.note</code>        | String  | Default: ""     |
| <code>&lt;MoFlowTranInfo&gt;.probability</code> | Integer | Default: 100    |

### Related Methods

**deleteTranInfo** < MoFlowTransition > <index\_integer>

Deletes the indexed MoFlowTranInfo from the MoFlowTransition. . If the MoFlowTranInfo is the only MoFlowTranInfo in MoFlowTransition, it is not deleted.

computeAnimation <moflow> [redraw:<true>] [incGlobals:<false>]

Computes the global flow network. This function has to be called to update any changes made to the motion flow network. `redraw:true` will redraw the viewports. `incGlobals:true` will also include the global motion flow network

### Notes

Changes to the `MoFlowTranInfo` property values do not cause an immediate update of the biped. `ComputeAnimation` must be called on the `MoFlow` value to recompute the biped motion.

The following example will find the transition information from the first clip (snippet) to the next in the first script defined for the Biped motion flow.

### Script Example

```
CSPATH = "f:\\3dsmax31_86\\cstudio\\"
bipObj = biped.createNew 100 100 [0,0,0]
select bipObj
max motion mode
bip = bipObj.transform.controller
-- get the MoFlow value from the biped controller:
mf=bip.motionFlow
-- go into motion flow mode and load a motion flow file
bip.motionmode=true
loadMoFlowFile mf (CSPATH + "scripts\\4flololoop.mfe")
--
-- get the script of interest from the MoFlow:
mfs=mf.scripts[1]
-- the script items from the script are in mfs.scriptItems.
-- get the snippet (snippet_from) from the first script item:
snippet_from=mfs.scriptItems[1].snippet
-- get the snippet (snippet_to) from the second script item:
snippet_to=mfs.scriptItems[2].snippet
-- search the transitions in snippet_from to find
-- the one whose toSnippet property == snippet_to:
theTrans=undefined
for trans in snippet_from.transitions where (trans.toSnippet == snippet_to) do
(
  theTrans=trans
  break
)
--get the transition information for the from script item:
theTransInfo=theTrans.tranInfos[mfs.scriptItems[1].tranindex+1]
```

## MoFlowTransition : MaxWrapper

### Constructor

**addTransition** <from\_MoFlowSnippet > <to\_MoFlowSnippet > <bool\_optimize>

Adds a new MoFlowTransition from the **from\_MoFlowSnippet** to the **to\_MoFlowSnippet** if one doesn't already exist. The optimize parameter acts as the "Optimize Selected Transition" in the Motion Flow Editor.

### Properties

<MoFlowTransition>. **fromSnippet** MoFlowSnippet Default: Varies

Specifies the MoFlowSnippet transitioning from.

<MoFlowTransition>. **toSnippet** MoFlowSnippet Default: Varies

Specifies the MoFlowSnippet transitioning to.

<MoFlowTransition>. **active** Boolean Default: True -- Read-only

Specifies whether the MoFlowTransition is active.

<MoFlowTransition>. **selected** Boolean Default: False

Specifies whether the MoFlowTransition line is selected in the Motion Flow Graph.

<MoFlowTransition>. **tranInfos** Array Default: #(MoFlowTranInfo : [X,X])

-- Read-only

Array of motion flow transition info blocks (MoFlowTranInfo values). The element of this array used is determined by the <mfscriptItem>.tranIndex property.

### Methods

**addTranInfo** < MoFlowTransition >

Adds a new MoFlowTranInfo to the MoFlowTransition and returns the newly created MoFlowTranInfo.

**deleteTranInfo** < MoFlowTransition > <index\_integer>

Deletes the indexed MoFlowTranInfo from MoFlowTransition. If the MoFlowTranInfo is the only MoFlowTranInfo in MoFlowTransition, it is not deleted.

### Related Methods

**deleteTransition** < MoFlowSnippet > <index\_integer>

Deletes the indexed MoFlowTransition emanating from the MoFlowSnippet.

**deleteTransitionTo** <from\_MoFlowSnippet > <to\_MoFlowSnippet >

Deletes the transition that's emanating to the specified snippet.

**computeAnimation** <moflow> [redraw:<true>] [incGlobals:<false>]

Computes the global flow network. This function has to be called to update any changes made to the motion flow network. `redraw:true` will redraw the viewports. `incGlobals:true` will also include the global motion flow network

### Notes

Changes to the `MoFlowTransition` property values do not cause an immediate update of the biped. `ComputeAnimation` must be called on the `MoFlow` value to recompute the biped motion.

---

## Biped Keys

All common MAXScript key functions like `deleteKey`, `selectKeys`, `moveKeys` etc. can be used with Biped keys. The exceptions `addNewKey` and `deleteKeys` are substituted with the following methods.

### Method

```
biped.addNewKey <biped_controller> <time> [ #select ]
<time>           Adds a new key to the controller track at the time specified.
[ #select ]      The new key is also selected if the #select optional argument is
specified.
```

The value for the new key is the interpolated controller value at the specified time. The value for the new key is the interpolated controller value at that time. The new key is also selected if the **#select** optional argument is specified.

**addNewKey()** will not add a key if a key already exists at the specified time. The return value is the key located at the specified time.

### Method

```
biped.deleteKeys <biped_controller> [ #allKeys ] [ #selection ]
[ #allKeys ]
[ #selection ]
```

Deletes either all keys or all selected keys from the controller. If neither **#allKeys** or **#selection** is specified, all keys are deleted.

## Accessing a Biped controller key by index

Accessing a Biped controller key by indexing into the **.keys** property of the controller returns a type of key that MAXScript does not recognize. To get a Biped controller key by index use the following method:

### Method

```
biped.getKey ( <biped_controller> | <footstep_ctrl> ) <index>
<biped_controller>
<footstep_ctrl>
<index>
```

## Notes

Returns an instance of BipedKey for Biped body controllers and BipedFSKey for footstep controllers. BipedKey and BipedFSKey are defined below.

Not all of the Biped elements contain a transform controller. Rather, a controller on an object typically higher in the hierarchy may store the transform key for an element. For example, the transforms for all the fingers on a hand are stored in either the Finger0 or Clavicle transform controller. This depends on whether “Separate Tracks for Arms” is set to true or false, respectively.

The following example will get the first key for each of the subcontrollers of the \$Bip01 Vertical\_Horizontail\_Turn transform controller and show their properties. Additionally, the first key of the \$Bip01 Footstep transform controller will have its properties shown.

### Script Example

```
bip = $Bip01.transform.controller
-- Obtain the subcontrollers
vertCont = bip.vertical.controller
horzCont = bip.horizontal.controller
turnCont = bip.turning.controller
-- Get the first key for each subcontroller
vk = biped.getKey vertCont 1
hk = biped.getKey horzCont 1
tk = biped.getKey turnCont 1
-- Show the properties for the individual subcontroller key types
showProperties vk
showProperties hk
showProperties tk
-- Obtain the Biped's Footstep controller
fsCont = '$Bip01 Footsteps'.transform.controller
-- Show the Footstep controller properties
showProperties fsCont
-- Get the first Footstep controller key
fk = biped.getKey fsCont 1
showProperties fk
```

---

## BipedKey : MAXObject

All Biped vertical, horizontal, turning, and body keys are represented by this class.

### Properties

|                                                       |         |                 |
|-------------------------------------------------------|---------|-----------------|
| <bipedkey>.time                                       | Time    | Default: Varies |
| Enter a value to specify when in time the key occurs. |         |                 |
| <bipedkey>.selected                                   | Boolean | Default: False  |
| <bipedkey>.tension                                    | Float   | Default: 25.0   |

Controls the amount of curvature in the animation curve.

High Tension produces a linear curve. It also has a slight Ease To and Ease From effect.

Low Tension produces a very wide, rounded, curve. It also has a slight negative Ease To and Ease From effect.

|                                          |       |               |
|------------------------------------------|-------|---------------|
| <code>&lt;bipedkey&gt;.continuity</code> | Float | Default: 25.0 |
|------------------------------------------|-------|---------------|

Controls the tangential property of the curve at the key. The default setting is the only value that produces a smooth animation curve through the key. All other values produce a discontinuity in the animation curve causing an abrupt change in the animation.

|                                    |       |               |
|------------------------------------|-------|---------------|
| <code>&lt;bipedkey&gt;.bias</code> | Float | Default: 25.0 |
|------------------------------------|-------|---------------|

Controls where the animation curve occurs with respect to the key.

|                                      |       |              |
|--------------------------------------|-------|--------------|
| <code>&lt;bipedkey&gt;.easeTo</code> | Float | Default: 0.0 |
|--------------------------------------|-------|--------------|

Slows the velocity of the animation curve as it approaches the key.

High Ease To causes the animation to decelerate as it approaches the key.

The default setting causes no extra deceleration.

|                                        |       |              |
|----------------------------------------|-------|--------------|
| <code>&lt;bipedkey&gt;.easeFrom</code> | Float | Default: 0.0 |
|----------------------------------------|-------|--------------|

Slows the velocity of the animation curve as it leaves the key.

High Ease From causes the animation to start slow and accelerate as it leaves the key.

The default setting causes no change of the animation curve.

| <code>&lt;bipedkey&gt;.type</code> | Name | Default: Varies |
|------------------------------------|------|-----------------|
|------------------------------------|------|-----------------|

```
-- Read-only
```

Contains **#vertical**, **#horizontal**, **#turning**, or **#body** based on the key type

There are additional properties **based on the type** of the key:

```
-- #vertical
```

|                                |       |                 |
|--------------------------------|-------|-----------------|
| <code>&lt;vertkey&gt;.z</code> | Float | Default: Varies |
|--------------------------------|-------|-----------------|

Reposition the selected biped part in z.

|                                            |       |              |
|--------------------------------------------|-------|--------------|
| <code>&lt;vertkey&gt;.dynamicsBlend</code> | Float | Default: 1.0 |
|--------------------------------------------|-------|--------------|

Control the amount of gravity in an airborne period, as in a running or jumping motion.

This parameter has no effect on a walking motion where footsteps overlap.

|                                               |       |              |
|-----------------------------------------------|-------|--------------|
| <code>&lt;vertkey&gt;.ballisticTension</code> | Float | Default: 0.5 |
|-----------------------------------------------|-------|--------------|

Control the amount of spring or tension when the biped lands or takes off from a jump or run step. The change is subtle.

A walk cycle will not activate this value. The biped has to be airborne, *then* the Lift and

Touch vertical keys will display a Ballistic Tension value.

```
-- #horizontal
```

|                                |       |                 |
|--------------------------------|-------|-----------------|
| <code>&lt;horzkey&gt;.x</code> | Float | Default: Varies |
|--------------------------------|-------|-----------------|

Reposition the selected biped part in x.

|                                |       |                 |
|--------------------------------|-------|-----------------|
| <code>&lt;horzkey&gt;.y</code> | Float | Default: Varies |
|--------------------------------|-------|-----------------|

Reposition the selected biped part in y.

<horzkey>.balanceFactor                      Float              Default: 1.0

Position the biped's weight anywhere along a line that extends from the center of mass to the biped's head. A value of 0 places the biped's weight at the center of mass. A value of 1 places the biped's weight above the center of mass. A value of 2 places the biped's weight in the head.

-- #body  
<bodykey>.ikBlend                              Float              Default: 0.0

Determines how forward kinematics and inverse kinematics are blended to interpolate an intermediate position.

<bodykey>.ikSpace                              Integer              Default: 0

0 - Body

The biped limb is in biped coordinate space.

1 - Object

The biped limb is either in World coordinate space or the coordinate space of the selected object. Coordinate space can be blended between keys.

<bodykey>.ikAnkleTension                      Float              Default: 0.0

Adjusts the precedence of the ankle joint over the knee joint. When set to 0, the knee takes precedence. When set to 1, the ankle takes precedence.

<bodykey>.ikJoinedPivot                      Boolean              Default: True

Put the biped foot in the coordinate space of the previous key.

<bodykey>.ikPivotIndex                      Integer              Default: 1

Index into <bodykey>.ikPivots array of active (selected) pivot

<bodykey>.ikNumPivots                      Integer              Default: Varies - Read only

Number of pivot points in <bodykey>.ikPivots array

<bodykey>.ikPivots                              Array              Default: Varies - Array of Point3 values

Array of positions of all the pivot points for the body part

Here is an example

### Script Example

```
k = biped.getKey $'bip01 L clavicle'.transform.controller 2
showProperties k
k.ikBlend = .5
k.ikSpace = 0
k.ikAnkleTension = .8
k.ikJoinedPivot = false
```

```
k.ikPivotIndex = 8
k.ikNumPivots = 2
k.ikPivots
-- to add a new one
k = biped.addNewKey '$bip01 L clavicle'.transform.controller 10f
```

## BipedFSKey : MAXObject

This class represents a Biped footstep in the viewports and trackview.

### Properties

|                                                                                                       |         |                          |
|-------------------------------------------------------------------------------------------------------|---------|--------------------------|
| <fskey>.time                                                                                          | Time    | Default: Varies          |
| A value indicating when in time the key occurs.                                                       |         |                          |
| <fskey>.duration                                                                                      | Time    | Default: Varies          |
| The number of frames in each footstep.                                                                |         |                          |
| <fskey>.selected                                                                                      | Boolean | Default: False           |
| <fskey>.edgeSel                                                                                       | Integer | Default: 0               |
| 0 - no edges selected<br>1 - left edge Selected<br>2 - right edge selected<br>3 - both edges Selected |         |                          |
| <fskey>.active                                                                                        | Boolean | Default: True            |
| Activate or make inactive the footstep.                                                               |         |                          |
| <fskey>.transform                                                                                     | Matrix3 | Default: Varies          |
| <fskey>.side                                                                                          | Name    | Default: #left or #right |
| Read-only                                                                                             |         |                          |

### See Also

- Biped Controllers (see page 610)
- Biped Footsteps (see page 621)
- Biped Vertical\_Horizontal\_Turn(Body):Matrix3 Controller (see page 610)
- Biped Keys (see page 636)



## Biped Motion Capture

All motion capture properties in Biped are global parameters and cannot be varied from Biped to Biped. Hence they are exposed as system globals residing in the **mocap** struct.

### Properties

**mocap.fsExtractionMode** Integer Default: 0

0 - None: Freeform

No footsteps are extracted.

1 - On

Extract footsteps.

2 - Fit to existing

For motion data that has both footstep motion and flying, swimming, falling, or tumbling motions.

**mocap.fsConversionMode** Integer Default: 0

0 - No Key Reduction

Do not reduce keys. Use this on files that are already key reduced or if you want to work with all the data in a raw motion capture file.

1 - Use Key Reduction

Reduce keys for simpler key editing.

2 - Load Buffer Only

Do not apply the data to the biped, load the data to the motion capture buffer only. Use this to either compare your edited version with the original or to paste postures from the motion capture buffer to the biped in the scene.

**mocap.upVector** Integer Default: 2

Set the vertical axis used in the motion capture data.

0 - X

1 - Y

2 - Z

**mocap.scaleFactor** Float Default: 1.0

Multiply the stored talent size by this value and size the biped accordingly.

-- range: 1+

### -- Footstep Extraction

**mocap.fsExtractionTol** Float Default: 0.05

Set the sensitivity of footstep extraction. Biped determines if the footstep is there by checking that the foot does not move beyond the distance determined by the Extraction Tolerance value. Smaller numbers are more sensitive and extract more footsteps. Value is a percentage of foot length.

-- range: 0+

`mocap.fsSlidingDist` Float Default: 0.0

Create a sliding footstep when positional tolerance is reached. This value is a percentage of foot length. By default the foot must slide its own distance (100), before a sliding footstep is created.

-- range: 0+

`mocap.fsSlidingAngle` Float Default: 0.0

Create a sliding footstep when rotational tolerance is reached. This value is in degrees. If this is set high (360 degrees), the foot must make a complete turn before a sliding footstep is created.

-- range: 0-360

`mocap.fsUseVerticalTol` Boolean Default: False

Turn on Z-axis Tolerance. The control filters out footsteps that do not fall within a given range of the ground plane. Use this when filtering motions, such as hopping or pitching a baseball, in which a foot may come off the ground and remain stationary, but its position is not intended as a footstep.

`mocap.fsVerticalTol` Float Default: 0.5

Value is a percentage of leg length

-- range: 0+

`mocap.fsZLevel` Float Default: 0.0

Set a Z value (ground).

`mocap.fsUseFlatten` Boolean Default: False

Extracted footsteps are moved to Z=0. Use this to flatten out minor differences in the height of the extracted footsteps.

## -- Load Frames

`mocap.startFrame` Integer Default: 0

Start importing at this frame. Default is frame 0, the first frame.

`mocap.endFrame` Integer Default: 0

Stop importing at this frame. Default is the last frame of the clip.

`mocap.loop` Boolean Default: False

Loop the data by the value set here.

This is relative. Succeeding loops start where the previous loop left off. The clips are not blended and may require editing unless the original clip was designed to loop.

Use this for clips designed to loop.

## Note

This often works best if Footstep Extraction is tuned off.

`mocap.loopFrameCount` Integer Default: 0

-- range: 0+

## -- Key Reduction Settings

**Tolerance:** Set the maximum angular or positional deviation for a track.

Values are in degrees for rotation tracks. Values are in units of translation for position tracks.

**Minimum Key Spacing:** Set the minimum number of frames between keys.

Tolerance is computed first, then Minimum Key Spacing computes further key reduction. A Minimum Key Spacing value of 10 for the head track ensures that no two keys are closer than 10 frames for this track.

**Set All:** Force all tracks to the values set in these fields.

Higher values here can determine how much key reduction is possible while preserving the original motion.

**Filter:** Clear to prevent filtering of the motion capture data into a track. If this is cleared, there will be no key reduction for the track.

|                                |         |                |
|--------------------------------|---------|----------------|
| <code>mocap.allTol</code>      | Float   | Default: 0.0   |
| -- range: 0+                   |         |                |
| <code>mocap.allSpacing</code>  | Integer | Default: 3     |
| -- range: 0+                   |         |                |
| <code>mocap.allFilter</code>   | Boolean | Default: False |
| <code>mocap.horzTol</code>     | Float   | Default: 1.0   |
| -- range: 0+                   |         |                |
| <code>mocap.horzSpacing</code> | Integer | Default: 3     |
| -- range: 0+                   |         |                |
| <code>mocap.horzFilter</code>  | Boolean | Default: True  |
| <code>mocap.rotTol</code>      | Float   | Default: 1.0   |
| -- range: 0+                   |         |                |
| <code>mocap.rotSpacing</code>  | Integer | Default: 3     |
| -- range: 0+                   |         |                |
| <code>mocap.rotFilter</code>   | Boolean | Default: True  |
| <code>mocap.vertTol</code>     | Float   | Default: 1.0   |
| -- range: 0+                   |         |                |
| <code>mocap.vertSpacing</code> | Integer | Default: 4     |
| -- range: 0+                   |         |                |
| <code>mocap.vertFilter</code>  | Boolean | Default: True  |
| <code>mocap.pelvisTol</code>   | Float   | Default: 6.0   |
| -- range: 0+                   |         |                |

|                                    |         |               |
|------------------------------------|---------|---------------|
| <code>mocap.pelvisSpacing</code>   | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.pelvisFilter</code>    | Boolean | Default: True |
| <code>mocap.spineTol</code>        | Float   | Default: 6.0  |
| -- range: 0+                       |         |               |
| <code>mocap.spineSpacing</code>    | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.spineFilter</code>     | Boolean | Default: True |
| <code>mocap.neckTol</code>         | Float   | Default: 6.0  |
| -- range: 0+                       |         |               |
| <code>mocap.neckSpacing</code>     | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.neckFilter</code>      | Boolean | Default: True |
| <code>mocap.leftArmTol</code>      | Float   | Default: 6.0  |
| -- range: 0+                       |         |               |
| <code>mocap.leftArmSpacing</code>  | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.leftArmFilter</code>   | Boolean | Default: True |
| <code>mocap.rightArmTol</code>     | Float   | Default: 6.0  |
| -- range: 0+                       |         |               |
| <code>mocap.rightArmSpacing</code> | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.rightArmFilter</code>  | Boolean | Default: True |
| <code>mocap.leftLegTol</code>      | Float   | Default: 6.0  |
| -- range: 0+                       |         |               |
| <code>mocap.leftLegSpacing</code>  | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.leftLegFilter</code>   | Boolean | Default: True |
| <code>mocap.rightLegTol</code>     | Float   | Default: 6.0  |

|                                    |         |               |
|------------------------------------|---------|---------------|
| -- range: 0+                       |         |               |
| <code>mocap.rightLegSpacing</code> | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.rightLegFilter</code>  | Boolean | Default: True |
| <code>mocap.tailTol</code>         | Float   | Default: 6.0  |
| -- range: 0+                       |         |               |
| <code>mocap.tailSpacing</code>     | Integer | Default: 3    |
| -- range: 0+                       |         |               |
| <code>mocap.tailFilter</code>      | Boolean | Default: True |

## -- Limb orientation

**Angle:** Move the knee or Elbow *position* to create the biped joint key.

**Point:** *Rotate* the shoulder-elbow-wrist or hip-knee-ankle to create the biped joint key.

**Auto:** Auto reads exact hand and foot positions from the motion capture data, character studio then places the knees and elbows in a natural position. For marker files involving running and walking, this option can clean up the data nearly instantly, regardless of how many markers were used (and where they were placed).

|                               |         |            |
|-------------------------------|---------|------------|
| <code>mocap.kneeOrient</code> | Integer | Default: 0 |
|-------------------------------|---------|------------|

0 - angle

1 - point

2 - auto

The biped knee hinge joints are perpendicular to the triangles formed by the hip-knee-ankle. Resolve errors in the motion capture data that break this rule by using either the angle or point method.

|                                |         |            |
|--------------------------------|---------|------------|
| <code>mocap.elbowOrient</code> | Integer | Default: 0 |
|--------------------------------|---------|------------|

0 - angle

1 - point

2 - auto

The biped elbow hinge joints are perpendicular to the triangles formed by the shoulder-elbow-wrist. Resolve errors in the motion capture data that break this rule by using either the angle or point method.

|                               |         |            |
|-------------------------------|---------|------------|
| <code>mocap.footOrient</code> | Integer | Default: 0 |
|-------------------------------|---------|------------|

0 - angle

1 - auto

The biped foot hinge joints are perpendicular to the triangles formed by the hip-knee-ankle respectively. Resolve errors in the motion capture data that break this rule by using either the angle or point method.

`mocap.handOrient` Integer Default: 0

0 - angle

1 - auto

The biped hand hinge joints are perpendicular to the triangles formed by the shoulder-elbow-wrist. Resolve errors in the motion capture data that break this rule by using either the angle or point method.

#### -- Talent Definition

`mocap.talentFigStrucFile` String Default: "" -- figure  
structure file (.fig)

A *.fig* file containing changes made to the biped scale while in Talent Figure mode.

`mocap.useTalentFigStrucFile` Boolean Default: False

Turn on `useTalentFigStrucFile` to use the file whose string is defined in `mocap.talentFigStrucFile`.

`mocap.talentPoseAdjFile` String Default: "" -- pose  
adjustment file (.cal)

A *.cal* file containing changes made to the biped while in Adjust Talent Pose mode.

`mocap.useTalentPoseAdjFile` Boolean Default: False

Turn on `useTalentPoseAdjFile` to use the file whose string is defined in `mocap.talentPoseAdjFile`.

#### -- Marker Name Files

`mocap.markerNameFile` String Default: "" -- csm  
marker name file (.mnm)

A Marker Name file to map incoming marker names in motion capture files (*.bvh* or *.csm*) to the character studio marker naming convention.

`mocap.useMarkerNameFile` Boolean Default: False

When set to true, use the file defined in `mocap.markerNameFile`.

`mocap.jointNameFile` String Default: "" -- bvh  
marker name file (.mnm)

A Marker Name file to map incoming marker names in motion capture files (*.bvh* or *.csm*) to the character studio marker naming convention.

`mocap.useJointNameFile` Boolean Default: False

When set to true, use the file defined in `mocap.jointNameFile`.

#### -- Marker Display Options

Marker and marker names are displayed around the biped. Discrepancies like the biped elbow position relative to the elbow marker can be spotted and adjusted.

`mocap.dispKnownMarkers` Boolean Default: False

|                                         |         |                |
|-----------------------------------------|---------|----------------|
| <code>mocap.dispKnownMarkersType</code> | Boolean | Default: False |
| true – on all props                     |         |                |
| false – on selected props               |         |                |
| <code>mocap.dispPropMarkers</code>      | Boolean | Default: False |
| <code>mocap.dispUnKnownMarkers</code>   | Boolean | Default: False |

-- Load Save Methods

```
mocap.loadParameters <file_name>
mocap.saveParameters <file_name>
```

Load/Save a motion capture parameter (.MOC) file





# Crowd MAXScript Extensions

## Crowd : helper

### Constructor

crowd ...

### Properties

<crowd>.simStart Integer Default: 0

The first frame of the simulation.

<crowd>.solveStart Integer Default: 0

The frame at which you begin solving.

<crowd>.solveEnd Integer Default: 100

Specifies the last frame considered for the solution.

<Crowd>.deleteKeys Boolean Default: False

When true, Crowd deletes the keys of active delegates in the range over which the solution takes place.

<Crowd>.saveNthPos Integer Default: 1

Saves every nth position frame after the solution.

<Crowd>.saveNthRot Integer Default: 1

Saves every nth rotation frame after the solution.

<Crowd>.flashing Boolean Default: True Alias:  
Hilite\_Delegates\_During\_Assignments

<crowd>.vectorScale Float Default: 1.0

Globally scales all force and velocity vectors that are displayed during the simulation.

Scaling vectors up helps to see them better when they are very small. It does not effect the simulation.

<Crowd>.useScript Boolean Default: False

When true, Crowd executes a script at each frame of the solution.

<Crowd>.functionName String Default: "PerFrameFn" Alias:  
Script\_Context\_Name

The name of the function to be executed. This name must also be specified in the script.

<Crowd>.script String Default: Undefined

`<crowd>.update` Boolean Default: True Alias:  
Update\_Display\_During\_Solve

When true, motion produced during solution of a crowd simulation appears in the viewports.

`<Crowd>.updateFrequency` Integer Default: 1

How often the display is updated during the solution. If 1, the update occurs every frame. If 2, the update occurs every other frame, and so on.

`<Crowd>.solveForBipeds` Boolean Default: False

When on, only biped/delegates are included in the computation. Also, the options to use priorities and backtracking become available. These options are available only for biped-only computations.

`<Crowd>.usePriorities` Boolean Default: False

When on, biped/delegates are computed one delegate at a time, in order of their Priority values, from lowest to highest. Also, backtracking becomes available and Step Solve becomes unavailable.

`<Crowd>.backtracking` Boolean Default: False

Turns on backtracking functionality when solving a crowd simulation that uses bipeds. When Backtracking is on during the solution, in the case of an impending collision between bipeds, the Crowd system will back up the simulation to the beginning of the current clip, and then try a different traversal of the lower-priority delegate/biped's motion flow graph. If necessary, the system will back up two or more clips.

`<crowd>.showCollisions` Boolean Default: False Alias:  
ClearColl

When true the delegates that collide are highlighted in the collision color.

`<crowd>.whenToShowCollisions` Integer Default: 0

0 – only during collisions - Colliding delegates are highlighted only in frames in which they actually collide.

1 – always - Colliding delegates are highlighted in frames in which they collide and all subsequent frames.

`<crowd>.collisionColor` Color Default: (color 255 0 0)

The color swatch indicates the color used to highlight colliding delegates.

`<crowd>.iconSize` Float Default: 0.0

`<crowd>.behaviors` ArrayParameter Default: #()

Array of Behavior objects

`<crowd>.teams` ArrayParameter Default: #()

Array of Team objects

<crowd>.assignments                      ArrayParameter Default: #()

<crowd>.cogcontrols                      ArrayParameter Default: #()    Alias:  
Cognitive\_Controllers

*CogControl : MAXObject (see page 670)*

<crowd>.scatter                      MAXObject    Default: Scatter\_Parameters

*CrowdScatter: (see page 657)*

<crowd>.objAssoc                      MAXRefTarg    Default: ObjAssoc    Alias:  
Object\_Delegate\_Associations\_Parameters

Display the dialog to link any number of delegate-object pairs.

<crowd>.smooth                      MAXRefTarg    Default: Smooth    Alias:  
Smoothing\_Parameters

Display the Smoothing dialog to smooth existing animation keys on a solved simulation to create more natural-looking animation

<Crowd>.priority                      MAXRefTarg    Default: Priority -- Read-only;  
Alias: Priority\_Parameters  
See Priority Properties**Crowd\_Priority\_Properties.**

## Delegate : Helper

The Delegate helper object is a special pyramidal object used in crowd animation. By default, the point of the pyramid indicates the forward direction.

The Crowd object controls the delegate or delegates, whose motion can then be imparted to a biped or other object.

### Constructor

Delegate ...  
CrowdDelegate ...

### Properties

<Delegate>.width                      Float              Default: 0.0    -- world units

The width of the Delegate object.

<Delegate>.depth                      Float              Default: 0.0    -- world units

The depth of the Delegate object.

<Delegates>.height                      Float              Default: 0.0    -- world units

The height of the Delegate object.

<Delegate>.velocityColor              Color              Default: (color 0 0 0)

When **showVelocity** is true, uses the specified color to draw a vector in the delegate's center during the simulation solution. The vector length indicates the delegate's relative speed.

```
<Delegate>.active Boolean Default: True
```

The delegate object is subject to control by a Crowd object.

```
<Delegate>.showForces Boolean Default: True
```

The forces being applied to a delegate by any applicable behaviors are drawn as vectors whose length indicate the extent of the forces. For example, if the delegate is affected by a Space Warp behavior and a Wander behavior, the vectors (using default colors) are yellow and blue-green, respectively. These vectors are visible only during solution of the crowd simulation.

```
<Delegate>.showVelocity Boolean Default: False
```

Uses the Velocity Color (see above) to draw a vector whose length depicts the delegate's relative speed. This vector is visible only during solution of the crowd simulation.

```
<Delegate>.showCogStates Boolean Default: True
```

During a solve, a text label appears next to the delegate showing the name of the cognitive controller state or transition that currently directs its behavior, if any.

```
<Delegate>.xyConstrain Boolean Default: True
```

The delegate remains at its initial height (position on the world XY axis) throughout the simulation. When off, the delegate's height can change during the simulation, for example when seeking an object at a different height.

```
<Delegate>.useHierBbox Boolean Default: True
Alias Use_Hierarchy_in_Bounding_Box_Computation
```

When true, the Avoid behavior uses the bounding box of the delegate and all of its children to perform its behavior.

This bounding box is computed by the Crowd object, is used by the collision detector, and can be accessed by other behaviors.

```
<Delegate>.averageSpeed Float Default: 5.0 -- animatable;
world units
```

Specifies the delegate's baseline velocity in 3DS MAX units (or the current unit type) per second. This can be modified during the simulation by a variety of factors, such as a linked biped's built-in speed and Deviation settings in a behavior.

```
<Delegate>.maxAccel Float Default: 0.1 -- animatable;
world units
```

Multiplied times Average Speed to determine the maximum acceleration.

```
<Delegate>.turnDecel Float Default: 0.3 -- animatable;
Alias: Turn_Deceleration_Weight
```

Specifies how much a delegate should slow down when turning. The higher this setting, the more the delegate slows down when it reaches the turn angle (see following parameter). A value of 0 specifies no slowdown; a value of 1 tells the delegate to stop.

The algorithm computes a value,  $d$ , which goes linearly from 0 to  $(1 - \text{Decel Weight})$  as the turn angle of the delegate goes from 0 to the Turn Angle specified by the user. The speed of the delegate is then multiplied by  $d$ . For example, when the delegate turns at the Turn Angle or greater, its speed will be multiplied by  $1 - \text{Decel Weight}$ , slowing it down as much as possible based on this parameter. When the delegate is not turning at all, its speed is not affected by the Decel Weight. When the delegate is turning at half the specified Turn Angle,  $d = \text{Decel Weight} / 2$ , so its speed will be multiplied by  $(1 - \text{Decel Weight} / 2)$ . As a practical example, take a delegate traveling at 10 units/sec, Decel Weight is set to 0.4, and At Turn Angle is set to 30. When the delegate has turned 15 degrees (half the At Turn Angle), the effective deceleration weight is 0.2. Subtract that quantity from 1 to get 0.8, and then multiply that times the delegate's speed to get 8 units per second halfway into the turn. At the full turn (30 degrees), the delegate travels at 6 units per second.

```
<Delegate>.turnDecelAngle          Float          Default: 10.0  -- animatable;
Alias: Turn_Deceleration_Angle
```

Specifies the turn angle at which Decel Weight's full slowdown effect is applied. If the current turn angle is less than At Turn Angle, the algorithm divides the latter by the former, and then divides the Decel Weight setting by the result to derive the effective decel weight.

```
<Delegate>.inclineDecel            Float          Default: 0.1   -- animatable;
Alias: Incline_Deceleration_Weight
```

Specifies how much the delegate should slow down when moving at an upward slant.

```
<Delegate>.inclineDecelAngle       Float          Default: 90.0  -- animatable;
Alias: Incline_Deceleration_Angle
```

Specifies the upward slant angle at which Decel Weight's full slowdown effect is applied.

```
<Delegate>.declineAccel            Float          Default: 0.1   -- animatable;
Alias: Decline_Acceleration_Weight
```

Specifies how much the delegate should speed up when moving at a downward slant.

See Decel Weight for a full explanation, taking into account that Accel Weight produces a speedup effect rather than a slowdown. Thus, the effective acceleration weight is added to 1, not subtracted from it.

```
<Delegate>.declineAccelAngle       Float          Default: 90.0  -- animatable;
Alias: Decline_Acceleration_Angle
```

Specifies the downward slant angle at which Accel Weight's full speedup effect is applied.

```
<Delegate>.maxTurnVel              Float          Default: 30.0  -- animatable;
Alias: Max_Turn_Velocity
```

Specifies the maximum number of degrees a delegate can turn.

```
<Delegate>.maxTurnAccel            Float          Default: 3.0   -- animatable;
Alias: Max_Turn_Acceleration
```

Specifies how much the delegate's turn angle can change per frame. If this is not small, the delegate's direction might jerk around. It would be allowed to turn suddenly, rather than smoothly.

```
<Delegate>.maxIncline          Float          Default: 90.0  -- animatable;
Alias: Max_Incline_Velocity
```

Specifies the maximum number of degrees a delegate can turn upward per frame.

```
<Delegate>.maxDecline          Float          Default: 90.0  -- animatable;
Alias: Max_Decline_Velocity
```

Specifies the maximum number of degrees a delegate can turn downward per frame.

```
<Delegate>.maxBank             Float          Default: 30.0  -- animatable
```

Specifies the maximum number of degrees a delegate can bank.

```
<Delegate>.maxBankVel          Float          Default: 3.0   -- animatable;
Alias: Max_Bank_Velocity
```

Specifies how much the delegate's bank angle can change per frame.

```
<Delegate>.bankPerTurn         Float          Default: 1.0   -- animatable
```

The number of degrees the delegate will bank as a function of the turn angle at the current frame. For example, if Bank per Turn=1, the delegate will bank one degree for every degree it is turning at a given frame.

```
<Delegate>.useBiped            Boolean        Default: False
```

When true, the delegate is associated with a biped.

```
<Delegate>.biped              Node            Default: Undefined
```

The biped with which the delegate is associated.

```
<Delegate>.startFrame          Integer        Default: 0
```

The frame at which the associated biped's first clip will begin to play.

```
<Delegate>.priority            Integer        Default: 0
```

Sets the delegate priority, which determines the order of solution in biped/delegate simulations.

```
<Delegate>.startClip           Integer        Default: 0
```

0 – First clip of current script

1 – Random start clip

```
<Delegate>.duration            Integer        Default: 0
```

The number of frames the delegate has been in the current state.

```
<Delegate>.behaviors           ArrayParameter Default: #()
```

The behaviors property contains valid values only during a solve. Contains the behaviors being used during the solve.

```
<Delegate>.weights             ArrayParameter Default: #()
```

The weights property contains valid values only during a solve. Each element contains the weight of the corresponding behavior in the behaviors property value.

`<Delegate>.simpos` Point3      Default: [0,0,0]

The simpos property contains a valid value only during a solve. Contains the current position of the delegate during the simulation.

`<Delegate>.randID` Integer      Default: 0

A possibly non-unique identifier whose purpose is to be used by behaviors as part of it's seed calculation. If more than one delegate has the the same randId then they should have the same random behavior. By default, crowd creates unique randIds.

`<Delegate>.index` Integer      Default: -1    -- Read-only

Index used by various behaviors as indices into internal arrays. Before a solve, Crowd makes sure that the delegates who are participating in the solve are given a contiguous set of indices, so that behaviors can set up arrays which use these indices.

### Related Delegates Methods

`delegates.isComputing` `<delegate_node>`

Returns a '1' if the delegate is currently active in a crowd solve simulation, it returns a '0' otherwise.

`delegates.velocity` `<delegate_node>`

Returns the normalized velocity of the delegate at the current simulation frame as a Point3 value.

`delegates.prevVelocity` `<delegate_node>`

Returns the normalized velocity of the delegate at the previous simulation frame as a Point3 value.

`delegates.startVelocity` `<delegate_node>` `<time>`

Returns the normalized velocity of the delegate at the start of the crowd simulation as a Point3 value. Thus the `<time>` parameter is a time value that should be equal to `crowd.simStart`.

`delegates.speed` `<delegate_node>`

Returns the speed of the delegate at the current simulation frame as a float.

`delegates.isAssignmentActive` `<delegate_node>` `<assignmentIndex>` `<time>`

Returns a '1' if the delegate is active for a particular crowd assignment at the time specified, if not it returns a '0'. The `<assignmentIndex>` directly corresponds to the index of the assignment in the `crowds.assignment` array.

**Note**

The functions **lineDisplay**, **sphereDisplay**, **bboxDisplay**, and **textDisplay** are all functions which can be used to draw a graphic primitive for a particular delegate when the crowd simulation is being solved. All of the positional values are in world space. Usually these functions will be used within a scripted behavior to visually demonstrate the behavior. For example, **lineDisplay** could be used to draw a line that represents the force that the behavior is exerting on that delegate.

```
delegates.lineDisplay <delegate_node> <startPosition> <endPosition> <color>
<vectorScale>
```

This function will draw a line from the start position to the endPosition with the color specified .

<startPosition> is a point3 value.

<endPositions> is a point3 value.

<color> is a color value.

<vectorScale> boolean value. Whether or not to scale the line display by the Crowd.vectorScale value.

```
delegates.sphereDisplay <delegate_node> <position> <radius> <color>
```

This function will draw a sphere at the position <position>, with a radius of size <radius>, and a color of <color>. <position> is a point3 value, radius is a float and <color> is a color value.

```
delegates.bboxDisplay <delegate_node> <minPosition> <maxPosition> <color>
```

This function will draw a bounding box specified by the two extrema points, <minPosition> and <maxPosition>, with the color specified. <minPosition> and <maxPosition> are point3 values and <color> is a color value.

```
delegates.textDisplay <delegate_node> <position> <color> <string>
```

This function will draw the text found in the <string> parameter at the position specified at <position>, with the color found in <color>. <position> is a point3 value, <color> is a color value, and <string> is a string value.

```
delegates.simTransform <delegate> <time>
```

Returns the delegate's transformation matrix at the specified time as a Matrix3 value. This method returns valid values during a solve.

**Note**

During a Solve, the simulation doesn't set the delegate nodes' position and rotation keys until the simulation stops. As such, accessing the delegate nodes' position and rotation during the solve will return incorrect results. The position of a delegate during a solve can be accessed via the **.simpos** property.

Delegates cannot be rendered, so the size of the Delegate object is primarily for use in scene setup and for determining bounding box extents.



## See Also

*Node Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## CrowdScatter:

These properties correspond to the Scatter Objects dialog displayed by clicking the Scatter Objects icon.

### Clone Tab Properties

`<crowd.scatter>.cloneObject`      Node      Default: Undefined

Object in the scene to be cloned.

`<crowd.scatter>.numClones`      Integer      Default: 10

The number of clones to be generated.

`<crowd.scatter>.cloneType`      Integer      Default: 0    Alias  
(0\_copy\_\_1\_reference\_\_2\_instance)

0 - copy

1 - Instance

2 - Reference

Specify how the object is cloned. It can be cloned as a copy, an instance, or a reference.

`<crowd.scatter>.cloneHierarchy`      Boolean      Default: True

When true, all objects linked to the selected object are cloned as well, with the hierarchical structure retained intact for each clone.

`<crowd.scatter>.cloneControllers`      Boolean      Default: True

Set true to clone an object when calling Scatter All. The object is cloned and then any specified transforms are applied to the clones.

### Position Tab Properties

`<crowd.scatter>.positionSpace`      Integer      Default: 0    Alias  
0\_Grid\_\_1\_Box\_\_2\_Sphere\_\_3\_Surface

0 - On Grid

1 - Inside Box

2 - Inside Sphere

3 - On Surface

4 - In Radial Area

Choose position object before selecting the reference object. On Grid distributes the clones over the surface of a grid object. Inside Box and Inside Sphere distribute the clones within the volume of a primitive box or sphere object, respectively.

<crowd.scatter>. **positionObject**                      Node                      Default: Undefined    Alias:  
Grid\_Box\_Sphere\_Surface

An object in the scene to be used as a reference object.

#### Note

You can use only a primitive sphere, a primitive box, or a grid helper object as a reference object. A primitive sphere or box that has been converted to a editable mesh object can't be used as a reference object.

<Crowd.scatter>. **surfaceOffset**                      Float                      Default: 0.0

On Surface specifies a consistent distance above the surface using surface normals for distribution. Available only when **.positionSpace** is set to On Surface.

<Crowd.scatter>. **centerX**                      Float                      Default: 0.0

Specifies the **X** value for the center of the distribution in world coordinates.

<Crowd.scatter>. **centerY**                      Float                      Default: 0.0

Specifies the **Y** value for the center of the distribution in world coordinates.

<Crowd.scatter>. **centerZ**                      Float                      Default: 0.0

Specifies the **Z** value for the center of the distribution in world coordinates.

<Crowd.scatter>. **radius**                      Float                      Default: 10.0

Specifies the maximum distance from the center within which clones are to be positioned.

<Crowd.scatter>. **XYPlane**                      Boolean                      Default: False

Specifies that clones are to be distributed on the world XY plane only, resulting in a disc-like array.

<crowd.scatter>. **childBbox**                      Boolean                      Default: True                      Alias:  
Include\_childrens\_\_bounding\_boxes\_in\_spacing\_calculations

When true, all of a hierarchical object's sub-objects are considered when determining spacing. When false, only the selected object is considered.

<crowd.scatter>. **spacing**                      Float                      Default: 1.0                      Alias:  
Bounding\_Box\_Multiplier\_for\_Position\_Spacing

Specifies the minimum distance between cloned objects. The Spacing setting is multiplied by the size of the object's bounding sphere to determine how close objects can get. If Spacing is left at 1.0, the default, objects normally cannot be positioned within each others' bounding spheres. If Spacing is set to 2.0, objects are separated by a distance equal to or greater than the size of the bounding sphere.

<crowd.scatter>. **positionSeed**                      Integer                      Default: 0

Specifies a seed value for randomizing the clones' locations. If a scene has more than one crowd, each should use a different seed to avoid having identical configurations.

## Rotation Tab Properties

`<crowd.scatter>.forwardAxisSign` Boolean Default: True

If true, the forward axis is in the positive direction. If false the forward axis is in the negative direction.

`<crowd.scatter>.forwardAxis` Integer Default: 1

0 - X

1 - Y

2 - Z

Specifies which axis of the cloned objects points forward, for use with the Look At Target option.

`<crowd.scatter>.UpAxisSign` Boolean Default: True

If true the up axis is in the positive direction. If false the up axis is in the negative direction.

`<crowd.scatter>.UpAxis` Integer Default: 2

0 - X

1 - Y

2 - Z

Axis of the cloned objects points upward; this axis is aligned with the world Z axis.

### Note

You cannot specify the same axis as Local Forward and Local Up simultaneously. If you choose an axis for one that's already chosen for the other, the software switches the other to a different axis.

`<crowd.scatter>.lookFrom` Integer Default: 0

0 - Look From Self

1 - Look From Selected Object

Determines the direction from which the clones look. By default, each clone looks from its own position (Self), so that when several clones are looking at a single target, each is oriented differently. To orient each clone so that it's parallel to an imaginary line between two objects (the "from" object and the "to" object), choose Selected Object and specify the object with the (None) button.

`<crowd.scatter>.lookFromObject` Node Default: Undefined

An object from which the clones are to look if **lookFrom** = 1.

`<crowd.scatter>.lookAtTarget` Node Default: Undefined

An object toward which the clones are to look.

`<crowd.scatter>.sideDeviation` Float Default: 0.0

Sets a maximum deviation angle in degrees for the clones' sideways orientation. If clones should look in an object's general direction but may look at a spot to either side of the target, use Sideways Deviation to set the maximum amount by which they can deviate

from the calculated angle. The actual deviation amount for each clone is calculated at random, based on the Deviation settings and the Rand Seed setting. Range=0.0 to 180.0.

`<crowd.scatter>.upDownDeviation` Float Default: 0.0

Sets a maximum deviation angle in degrees for the clones' up/down orientation. If clones should look in an object's general direction but may look at a spot above or below the target, use Up/Down Deviation to set the maximum amount by which they can deviate from the calculated angle. The actual deviation amount for each clone is calculated at random, based on the Deviation settings and the Rand Seed setting. Range=0.0 to 180.0.

`<crowd.scatter>.rotationSeed` Integer Default: 0

Specifies a seed value for randomizing the clones' orientations, based on the Deviation settings. If a scene has more than one crowd, each should use a different seed to avoid having identical configurations.

### Scale Tab Properties

Contains options for scaling object clones. For each scaling axis you can specify alternative forward and up axes, plus a target object toward which the clones will point. In addition, you can specify a source object; when using both source and target objects, the clones are rotated so they're parallel to the line between the two.

`<crowd.scatter>.xScale` Float Default: 1.0

Sets scaling on the X axis as a multiplier.

`<crowd.scatter>.xScaleDeviation` Float Default: 0.0

Sets the maximum factor for randomization of scaling. For each clone, Deviation is multiplied by a random number between 0.0 and 1.0, and then added to the Scale multiplier.

`<crowd.scatter>.matchXtoYscale` Boolean Default: False

Lets you use the same scaling as on the Y axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

`<crowd.scatter>.matchXtoZscale` Boolean Default: False

Lets you use the same scaling as on the Z axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

`<crowd.scatter>.yScale` Float Default: 1.0

Sets scaling on the Y axis as a multiplier.

`<crowd.scatter>.yScaleDeviation` Float Default: 0.0

Sets the maximum factor for randomization of scaling. For each clone, Deviation is multiplied by a random number between 0.0 and 1.0, and then added to the Scale multiplier.

`<crowd.scatter>.matchYtoXscale` Boolean Default: False

Lets you use the same scaling as on the X axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable

<crowd.scatter>. **matchYtoZscale** Boolean Default: False

Lets you use the same scaling as on the Z axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable

<crowd.scatter>. **zScale** Float Default: 1.0

Sets scaling on the Z axis as a multiplier.

<crowd.scatter>. **zScaleDeviation** Float Default: 0.0

Sets the maximum factor for randomization of scaling. For each clone, Deviation is multiplied by a random number between 0.0 and 1.0, and then added to the Scale multiplier.

<crowd.scatter>. **matchZtoXScale** Boolean Default: False

Lets you use the same scaling as on the X axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

<crowd.scatter>. **matchZtoYScale** Boolean Default: False

Lets you use the same scaling as on the Y axis, whether explicit or randomized. When you specify an axis, the parameters group for that axis becomes unavailable.

<crowd.scatter>. **scaleSeed** Integer Default: 0

Specifies a seed value for randomizing the clones' scales, based on the Deviation settings.

### All Ops Tab Properties

<crowd.scatter>. **ComputeClones** Boolean Default: False

Set true to clone an object when calling Scatter All. The object is cloned, then any specified transforms are applied to the clones.

Turning on Clones makes the Select Objects to Transform button unavailable. The object to clone and cloning parameters must be specified on the with **.ComputeClone**.

<crowd.scatter>. **ComputePositions** Boolean Default: False

When true and crowds.**scatterall** is called, the transforms are applied according to the settings in the Position panel.

<crowd.scatter>. **ComputeRotations** Boolean Default: False

When true and crowds.**scatterall** is called, the transforms are applied according to the settings in the Rotation panel.

<crowd.scatter>. **ComputeScales** Boolean Default: False

When true and crowds.**scatterall** is called, the transforms are applied according to the settings in the Scale panel.

<Crowd.scatter>. **IncPosSeed** Boolean Default: False

When true, add 1 to the **.positionSeed** value, and redistributes the objects using the new random seed.

```
<Crowd.scatter>.IncRotSeed           Boolean      Default: False
```

When true, add 1 to the **.rotationSeed** value, and redistributes the objects using the new random seed.

```
<Crowd.scatter>.IncScSeed           Boolean      Default: False
```

When true, add 1 to the **.scaleSeed** value, and redistributes the objects using the new random seed.

```
<crowd.scatter>.ObjectsToScatter    ArrayParameter Default: #()
```

Objects to be affected by calling crowds.**scatterall**

### ObjAssoc Properties

These properties correspond to the Object / Delegate dialog displayed by clicking the Associate Objects with Delegates icon.

```
<ObjAssoc>.objects                  ArrayParameter Default: #()  -- node array
```

Array of linked Objects.

```
<ObjAssoc>.delegates                ArrayParameter Default: #()  -- node array
```

Array of linked delegates.

```
<ObjAssoc>.alignScale               Boolean      Default: True
```

When true, Align Objects with Delegates sets each object's absolute scaling factor to that of its corresponding delegates. This is useful if, for example, you've randomized delegates' sizes with the Scatter Objects Scale panel, and want the associated objects to match.

### Smooth Properties

These properties correspond to the Smoothing dialog displayed by clicking the Smooth Paths button in the Solve rollout.

```
<Crowd.smooth>.objects              ArrayParameter Default: #()  -- node array;
Alias: Objects_to_Smooth
```

Specify which objects' positions and/or rotations to smooth.

```
<Crowd.smooth>.filterDelegates      Boolean      Default: True   Alias:
Filter_Delegate_Selection
```

When true, the Select dialog opened by the Select Objects to Smooth button shows only delegates. When off, it shows all scene objects.

```
<Crowd.smooth>.wholeAnim            Integer      Default: 0
```

0 - Whole Animation: Smooths all animation frames.

1 - Animation Segment: Smooths only the frame ranges specified in the From and To fields.

`<Crowd.smooth>.from` Integer Default: 0 Alias:  
`smooth_from_frame`

When Animation Segment is chosen, specifies the first animation frame for smoothing.

`<Crowd.smooth>.to` Integer Default: 0 Alias:  
`smooth_to_frame`

When Animation Segment is chosen, specifies the last animation frame for smoothing.

`<Crowd.smooth>.positions` Boolean Default: True Alias:  
`Smooth_positions`

When true, selected objects' animation paths generated via the simulation are smoothed after the simulation has finished.

`<Crowd.smooth>.rotations` Boolean Default: True Alias:  
`Smooth_rotations`

When true, selected objects' rotations generated via the simulation are smoothed after the simulation has finished.

---

## CrowdAssignment : MAXObject

### Constructor

`CrowdAssignment ...`

### Properties

`<crowdassignment>.team` CrowdTeam Default: Undefined

A named group of delegates.

`<crowdassignment>.delegate` Node Default: Undefined

A delegate object. See the *CrowdDelegate : Helper* (see page 651) topic.

`<crowdassignment>.behavior` MAXObject Default: Undefined

One of the *Crowd Behavior* (see page 671) classes.

`<crowdassignment>.cogcontrol` CogControl Default: Undefined Alias:  
`CognitiveController`

A cognitive controller. See *CogControl:MAXObject* (see page 670) for more details.

`<crowdassignment>.weight` Float Default: 1.0 --  
`animatable`

The relative effect of the assigned behavior or cognitive controller. The higher an assignment's **weight** setting is than others', the more effect it will have. This setting is animatable. Range=0.0 to 1.0.

`<crowdassignment>.active` : Boolean

When true, the assignment is currently in effect. When false, the assignment has no effect.  
Default=true.

### Notes

Normally either the team or the delegate property will contain a value of undefined. If both properties contain a value other than undefined, the assignment is applied to the team.

Normally either the cogcontrol or the behavior property will contain a value of undefined. If both properties contain a value other than undefined, the behavior will be used.

Do not assign a value other than undefined or a CrowdTeam value to the team property.

Do not assign a value other than undefined or a Behavior value to the behavior property.

Do not assign a value other than undefined or a CogControl value to the cogcontrol property.

Do not assign a node type other than a Delegate node to the delegate property.

### See also

*CrowdTeam : MAXObject (see page 664)*

*Crowd Behaviors (see page 671)*

*CogControl : MAXObject (see page 670)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## CrowdTeam : ReferenceTarget

### Constructor

```
CrowdTeam ...
CrowdGroup ...
```

### Properties

```
<crowdteam>.name           String           Default: "CrowdTeam"
    The default team name is Team followed by number one more than the last added team.
<crowdteam>.members        ArrayParameter Default: #()           -- array
of Delegate nodes
    Participants of the current team.
```

### Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.



```
<CrowdTeam>.members      -- array of Delegate nodes
```

### Notes

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a Members ArrayParameter element to undefined.

Assigning a non-Delegate node to the Members ArrayParameter.

### See also

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## CrowdState: ReferenceTarget

### Constructor

```
CrowdState ...
```

### Properties

```
<crowdstate>.name          String          Default: "State"
```

The name of the state.

```
<crowdstate>.behaviors      ArrayParameter Default: #() -- array of
behaviors
```

See Notes below.

The names of all behaviors associated with the state.

```
<crowdstate>.weights        ArrayParameter Default: #() -- array of floats
```

See Notes below.

Specifies the selected behavior's weight. The higher the weight in relation to other behaviors' weights, the more evident the results of the behavior in the state. Default=1.0. Range=0.0 to 1.0.

```
<crowdstate>.transitions    ArrayParameter Default: #() -- array of
CrowdTransition objects
```

See Notes below.

### Notes

You can perform the following MAXScript operations:

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

```
<CrowdState>.behaviors      -- array of behaviors
<CrowdState>.weights        -- array of floats
```

`<CrowdState>.transitions`    -- array of CrowdTransition objects

If you add or delete elements to the Behaviors ArrayParameter, the corresponding element in the Weights ArrayParameter **must** be added or deleted.

The following MAXScript operations will cause Crowd to fail, either right away or later:  
**NEVER** set Behavior ArrayParameter element to undefined.  
**NEVER** set a Behavior ArrayParameter element to anything other than an object of the appropriate type.

See Also

- Crowd Behaviors (see page 671)*
- CrowdTransition : MAXObject (see page 666)*
- MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*
- Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

# CrowdTransition : MAXObject

Constructor

`CrowdTransition ...`  
`Transition ...`

Properties

|                                                                                           |            |                      |        |
|-------------------------------------------------------------------------------------------|------------|----------------------|--------|
| <code>&lt;crowdtransition&gt;.priority</code>                                             | Integer    | Default: 0           |        |
| Sets the transition's priority.                                                           |            |                      |        |
| <code>&lt;crowdtransition&gt;.duration</code>                                             | Integer    | Default: 25          |        |
| The number of frames the software takes to affect the transition between states.          |            |                      |        |
| <code>&lt;crowdtransition&gt;.easeIn</code>                                               | Float      | Default: 0.5         |        |
| Ease in value for the source clip.                                                        |            |                      |        |
| <code>&lt;crowdtransition&gt;.easeOut</code>                                              | Float      | Default: 0.5         |        |
| Ease out value for the destination clip.                                                  |            |                      |        |
| <code>&lt;crowdtransition&gt;.functionName</code><br><code>Script_Context_Name</code>     | String     | Default: "transFunc" | Alias: |
| The name of the MAXScript conditional that specifies when/how the transition is to occur. |            |                      |        |
| <code>&lt;crowdtransition&gt;.script</code>                                               | String     | Default: Undefined   |        |
| <code>&lt;crowdtransition&gt;.from</code><br><code>FromState</code>                       | CrowdState | Default: Undefined   | Alias: |

The state that the transition originated from.

`<crowdtransition>.to`      CrowdState    Default: Undefined    Alias: ToState

The state that the transition is destined for.

## Notes

The `<CrowdTransition>.functionName` value must match the name of the function defined in `<CrowdTransition>.script`.

The script function is compiled with the function name in global scope. The function name cannot be the same as a MAXScript-defined global variable name.

The `<CrowdTransition>.script` is of the form:

```
fn transFunc del trans t =
( if t < 50 then 0 else 1
)
```

where **del** is the delegate node, **trans** is the CrowdTransition value, and **t** is the time value of the frame being evaluated. The MAX time context will also be frame being evaluated. The return value must be an integer value. If the return value is 0, the transition is not taken. For any other return value, the transition is taken.

The `<CrowdTransition>.script` is not evaluated for a delegate while the delegate is in transition between states.

The following MAXScript operations will cause Crowd to fail, either right away or later: NEVER set the **from** or **to** properties to undefined.

NEVER set the **from** or **to** properties to anything other than a CrowdState object.

## See also

*CrowdState : MAXObject (see page 665)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Crowds: Methods

`crowds.solve` `<crowd_node>`

Equivalent to clicking Solve in the crowd's Solve rollout. **.solve** starts the solving, based on all other current crowd parameter settings.

`crowds.genclones` `<crowd_node>`

Equivalent to clicking Generate Clones in the Scatter Objects dialog, Clone tab, displayed by clicking the Scatter Objects icon. Generates clones based on all other current crowd parameter settings.

`crowds.genlocations` `<crowd_node>`

Equivalent to clicking Generate Locations in the Scatter Objects dialog, Position tab, displayed by clicking the Scatter Objects icon. Generates locations based on all other current crowd parameter settings.

`crowds.genrotations <crowd_node>`

Equivalent to clicking Generate Orientations in the Scatter Objects dialog, Rotation tab, displayed by clicking the Scatter Objects icon. Generates rotations based on all other current crowd parameter settings.

`crowds.genscales <crowd_node>`

Equivalent to clicking Generate Scales in the Scatter Objects dialog, Scale tab, displayed by clicking the Scatter Objects icon. Generates scales based on all other current crowd parameter settings.

`crowds.scatterall <crowd_node>`

Equivalent to clicking Scatter button in the Scatter Objects dialog, All Ops tab. Scatters objects based on all other current crowd parameter settings.

`crowds.alignObjects <crowd_node>`

Equivalent to clicking Align Objects with Delegates in the Object / Delegates dialog displayed by clicking the Associate Objects with Delegates icon. Aligns objects based on all other current crowd parameter settings.

`crowds.linkObjects <crowd_node>`

Equivalent to clicking Link Objects to Delegates in the Object / Delegates dialog displayed by clicking the Associate Objects with Delegates icon. Links objects based on all other current crowd parameter settings.

`crowds.assignControllers <crowd_node>`

Equivalent to clicking Assign Delegate Controllers to Objects in the Object / Delegates dialog displayed by clicking the Associate Objects with Delegates icon. Assigns controllers based on all other current crowd parameter settings.

`crowds.smooth <crowd_node>`

Equivalent to clicking OK in the Smoothing dialog displayed by clicking the Smooth Paths button in the Solve rollout. Smooths objects motions based on all other current crowd parameter settings.

`crowds.assignGridProximityPriorities <crowd_node>`

Equivalent to clicking Proximity to a Grid / Assign in the Priority rollout. Assigns priorities to the delegates specified in `<Crowd.priority>.delegates` based on their distance from the grid specified in `<Crowd.priority>.grid`.

`crowds.assignObjectProximityPriorities <crowd_node>`

Equivalent to clicking Proximity to an Object / Assign in the Priority rollout. Assigns priorities to the delegates specified in <Crowd.priority>.delegates based on their distance from the object specified in <Crowd.priority>.object.

```
crowds.assignRandomPriorities <crowd_node>
```

Equivalent to clicking Assign Random Priorities in the Priority rollout. Assigns random priorities to the delegates specified in <Crowd.priority>.delegates.

```
crowds.assignUniquePriorities <crowd_node>
```

Equivalent to clicking Make Priorities Unique in the Priority rollout. Ensures that the priorities of the delegates specified in <Crowd.priority>.delegates are unique. If two delegates share the same priority, one of them will be given a new priority.

```
crowds.incrementPriorities <crowd_node>
```

Equivalent to clicking Increment Priorities in the Priority rollout. Increments the priorities of the delegates specified in <Crowd.priority>.delegates by the value in <Crowd.priority>.increment.

## Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

|                                          |                                         |
|------------------------------------------|-----------------------------------------|
| <Crowd>. <b>behaviors</b>                | -- array of Behavior objects            |
| <Crowd>. <b>teams</b>                    | -- array of CrowdTeam objects           |
| <Crowd>. <b>assignments</b>              | -- array of Assignment objects          |
| <Crowd>. <b>cogcontrols</b>              | -- array of CognitiveController objects |
| <Crowd.scatter>. <b>ObjectsToScatter</b> | -- node array                           |
| <Crowd.objAssoc>. <b>objects</b>         | -- node array                           |
| <Crowd.objAssoc>. <b>delegates</b>       | -- node array                           |
| <Crowd.smooth>. <b>objects</b>           | -- node array                           |

If you delete a team or a cognitive controller via MAXScript **deleteitem \$crowd01.teams[1] 1** or **deleteitem \$crowd01.cogcontrols[1] 1**, it may still be referenced by some assignments. Nothing will go wrong, but it's a strange thing to do and one probably should not do it.

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a Crowd ArrayParameter element to undefined.

**NEVER** set a Crowd ArrayParameter element to to anything other than an object of the appropriate type.

## CogControl : MAXObject

### Constructor

`CogControl ...`

### Properties

`<cogcontrol>.name` String Default: " CognitiveControl"

The name of the state diagram.

`<cogcontrol>.startState` CrowdState Default: Undefined

Set the starting *CrowdState* (see page 665). By default the state that executes first in a cognitive controller is the one that was added first.

`<cogcontrol>.states` ArrayParameter Default: #() -- array of CrowdState objects

All of the states used by this cognitive controller.

### Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

`<CogControl>.states` -- array of CrowdState objects

If you delete a CrowdState (`deleteitem $crowd01.cogcontrols[1].states 1`), it may still be referenced by either the startState or by a CrowdTransition stored in another CrowdState.

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a States ArrayParameter element or the startState to undefined.

**NEVER** set a States ArrayParameter element or the startState to anything other than a CrowdState.

### See also

*CrowdState:MAXObject* (see page 665)

*CrowdAssignment : MAXObject* (see page 663)

*CrowdTeam : MAXObject* (see page 664)

*CogControl : MAXObject* (see page 670)

*Node Common Properties, Operators, and Methods* (see the MAXScript Online Reference)

*MAXWrapper Common Properties, Operators, and Methods* (see the MAXScript Online Reference)

*Value Common Properties, Operators, and Methods* (see the MAXScript Online Reference)

*Avoid\_Behavior* : MAXObject (see page 671)

*Orientation\_Behavior* : MAXObject (see page 673)

*Path\_Follow\_Behavior* : MAXObject (see page 675)

*Repel\_Behavior* : MAXObject (see page 677)

*Scripted\_Behavior* : MAXObject (see page 678)

*Seek\_Behavior* : MAXObject (see page 686)

*Space\_Warp\_Behavior* : MAXObject (see page 687)

*Speed\_Vary\_Behavior* : MAXObject (see page 688)

*Surface\_Arrive\_Behavior* : MAXObject (see page 689)

*Surface\_Follow\_Behavior* : MAXObject (see page 693)

*Wall\_Repel\_Behavior* : MAXObject (see page 694)

*Wall\_Seek\_Behavior* : MAXObject (see page 696)

*Wander\_Behavior* : MAXObject (see page 697)

## Constructor

## Properties

The number of frames in advance of the current frame that the software looks for potential collisions.

Enables display of a wireframe sphere that depicts the extent of the **.hardRadius** setting.

|                   |                    |       |                |               |
|-------------------|--------------------|-------|----------------|---------------|
| <Avoid Behavior>. | <b>detourAngle</b> | Float | Default: 360.0 | -- animatable |
|-------------------|--------------------|-------|----------------|---------------|

Maximum necessary turning angle relative to the direction of delegate's goal that the delegate will steer to avoid rather than slow down and wait.

```
<Avoid_Behavior>.brakePressure      Float      Default: 2.0      -- animatable
```

Determines how strongly the brakes are applied. Higher values induce more gradual and slower stops, but may cause brakes to “pump” in order to slow down.

```
<Avoid_behavior>.repelStrength      Float      Default: 0.2 -- animatable
```

Determines the strength of the repelling force; higher values result in greater repulsion force.

```
<Avoid_behavior>.repelRadius        Float      Default: 3.0
```

Distance from hard radius of delegate where “repel” avoidance is sensed and carried out.

```
<Avoid_behavior>.repelFalloff        Float      Default: 3.0 -- animatable
```

Higher values cause the repel to fall off to zero more rapidly with distance, thus focusing its effect closer to the delegate's hard radius.

```
<Avoid_behavior>.vfieldStrength      Float      Default: 1.0 Alias
Vector_Field_Avoid_Strength -- animatable
```

Higher values result in more severe influence. Delegates will be directed to move perpendicular to the field.

```
<Avoid_behavior>.vfieldFalloff        Float      Default: 8.0 Alias
Vector_Field_Falloff -- animatable
```

Higher values cause vector field influence to fall off to zero more rapidly with distance, thus focusing its effect closer to the delegate's hard radius.

```
<Avoid_Behavior>.showRepelRadius      Boolean      Default: False
```

When true, displays a wireframe sphere that depicts the extent of the **.repelRadius** setting.

```
<Avoid_Behavior>.showPotentialCols      Boolean      Default: False
```

When true, displays a green line from the delegate to the location of a potential collision.

```
<Avoid_Behavior>.showRepelActivity      Boolean      Default: False
```

When true, displays a white line between the delegate and target when the repel force is in effect.

```
<Avoid_Behavior>.showLookAhead          Boolean      Default: False
```

When true, displays a sphere that shows the current distance used to check for potential collisions.

```
<Avoid_behavior>.forceColor            Color      Default: (color 255 0 0)
```

Sets the color used to draw the Avoid force vector during the solution.

```
<Avoid_behavior>.displayForce          Boolean      Default: True
```



When true, force exerted on the delegate(s) by the Avoid behavior is drawn in the viewports as a vector during the simulation solution.

### Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

<Avoid\_Behavior>.obstacles

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a Crowd/Behavior ArrayParameter element to undefined.

### See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Orientation\_Behavior : MAXObject

### Constructor

Orientation\_Behavior ...

OrientationBehavior ...

### Properties

|                             |        |                        |
|-----------------------------|--------|------------------------|
| <orientation_behavior>.name | String | Default: "Orientation" |
|-----------------------------|--------|------------------------|

|                                        |         |                |
|----------------------------------------|---------|----------------|
| <Orientation_Behavior>.headingRelative | Boolean | Default: False |
|----------------------------------------|---------|----------------|

|                                   |       |                        |
|-----------------------------------|-------|------------------------|
| <orientation_behavior>.minheading | Float | Default: -180.0 Alias: |
| HeadingMin - animatable           |       |                        |

The minimum permissible heading. This number should be lower than the Min Heading value. Range=-180 to 180.

|                                   |       |                                         |
|-----------------------------------|-------|-----------------------------------------|
| <orientation_behavior>.maxheading | Float | Default: 180.0 -HeadingMax - animatable |
|-----------------------------------|-------|-----------------------------------------|

The maximum permissible heading. This number should be higher than the Min Heading value. Range=-180 to 180.

|                                      |       |                       |
|--------------------------------------|-------|-----------------------|
| <orientation_behavior>.maxHeadingVel | Float | Default: 180.0 Alias: |
| maxHeadingVelocity - animatable      |       |                       |

Specifies how much the delegate's heading can change per frame. This controls angular acceleration and deceleration.

`<orientation_behavior>.headingresponse` Float Default: 1.0 - animatable

Determines how quickly the heading follows the direction the object is moving in. A value of 1.0 indicates maximum responsiveness, and will point in the direction the delegate is moving while a lower value means that it is less responsive. Range=0 to 1.

`<orientation_behavior>.minpitch` Float Default: -180.0 Alias: PitchMin - animatable

The minimum number of degrees a delegate can incline or decline. This number should be lower than the Max Pitch value. Range=-180 to 180.

`<orientation_behavior>.maxpitch` Float Default: 180.0 Alias: PitchMax - animatable

The maximum number of degrees a delegate can incline or decline. This number should be higher than the Min Pitch value. Range=-180 to 180.

`<orientation_behavior>.maxpitchVel` Float Default: 1.0 Alias: maxPitchVelocity - animatable

Specifies how much the delegate's pitch can change per frame. This controls angular acceleration and deceleration.

`<orientation_behavior>.pitchresponse` Float Default: 1.0 - animatable

Determines how quickly the pitch follows the direction the object is moving in. A value of 1.0 indicates maximum responsiveness while a lower value means that it is less responsive. Range=0 to 1.

`<orientation_behavior>.maxbank` Float Default: 30.0 - animatable

The maximum number of degrees the delegate can bank.

`<orientation_behavior>.maxbankAccel` Float Default: 3.0 Alias: MaxBank\_Change - animatable

The maximum number of degrees the delegate's bank angle can change per frame. This controls angular acceleration and deceleration.

`<orientation_behavior>.bankPerTurn` Float Default: 1.0 - animatable

The number of degrees the delegate will bank as a function of the turn angle at the current frame. For example, if Bank per Turn=1, the delegate will bank one degree for every degree it is turning at a given frame.

## Notes

If this behavior is active, it controls the orientation of the delegate by allowing the animator to specify limits and responsiveness on the pitch and heading of the object. The animator can limit the min and max values, the rate, and can also set a 'response' value which controls how quickly the pitch or heading follows the direction the object is

moving in. A value of 1.0 means it's responsive, and will point in the direction the delegate is moving (within the limits), while a lower value means that it is less responsive. It also calculates the roll, which is done using the same parameters that the default behavior does.

## See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Path\_Follow\_Behavior : MAXObject

### Constructor

`Path_Follow_Behavior ...`

`PathFollowBehavior ...`

### Properties

`<path_follow_behavior>.name` String Default: "Path Follow"

`<path_follow_behavior>.path` Node Default: Undefined

A path object. Suitable path objects include splines and NURBS curves. If a path object contains more than one spline or curve, character studio uses the lowest-numbered element (usually the earliest created one).

`<path_follow_behavior>.radius` Float Default: 20.0 Alias: Radius\_about\_Path -- animatable

The radial distance from the path within which the delegate stays while traversing the path. Range=0.0 to 99,999.0.

`<path_follow_behavior>.awareness` Float Default: 0.5 -- animatable

Specifies how "intelligent" the delegate is while traversing this path. A high Awareness setting means that it takes into account the curve of the path while moving and will try to anticipate changes. A low value for Awareness, on the other hand, means that the delegate notices the path only when leaving it. Range=0.0 to 1.0.

Note: You can randomize awareness behavior with the Deviation and Seed settings.

`<path_follow_behavior>.awareDeviation` Float Default: 0.0 Alias: AwarenessDeviation -- animatable

Specifies the maximum amount by which Awareness should vary. character studio takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Awareness setting, and adds the result to Awareness. Range=0.0 to 1.0.

Note: You can vary behaviors among different Path Follow behaviors that use the same Awareness and Deviation settings by changing the Seed value.

`<path_follow_behavior>.start` Integer Default: 0 -- animatable

0 - Path Beginning

1 - Path End

This setting determines where on the path the delegate begins to follow the path.

`<path_follow_behavior>.direction` Integer Default: 0 -- animatable

0 – Forwards: The delegate moves along path vertices in ascending order.

1 – Backwards: The delegate moves along path vertices in descending order.

`<path_follow_behavior>.endAction` Integer Default: 0 -- animatable

0 – Loop: The delegate loops around the path, even if it isn't closed. If Beginning of Path or End of Path is chosen, it returns to the path's start or end point each time it finishes traversing the path. If Nearest Point is chosen, it returns to an arbitrary point determined by its position and the path shape.

1 – Reverse: The delegate reverses direction at the end of the path. Use this choice to simulate a back-and-forth "patrol" behavior.

2 – Continue: The delegate continues moving in the same direction it faced at the end of the path until the simulation ends or it's acted upon by another force or behavior.

`<path_follow_behavior>.seed` Integer Default: 1 -- animatable

Specifies a seed value for randomizing Awareness.

`<path_follow_behavior>.forceColor` Color Default: (color 0 0 255)

The color used to draw the Path Follow force vector during the solution.

`<path_follow_behavior>.displayForce` Boolean Default: True

When true, force exerted on the delegate(s) by the Path Follow behavior is drawn in the viewports as a vector during the simulation solution.

`<path_follow_behavior>.targetColor` Color Default: (color 0 0 127.5)

Sets the color used to draw the target icon.

`<path_follow_behavior>.displayTarget` Boolean Default: True

Enables display of the target icon, which appears during the solution when a new interim goal is calculated for the delegate.

`<path_follow_behavior>.targetScale` Float Default: 5.0 -- animatable

The overall size of the target icon.

See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

## Repel\_Behavior : MAXObject

### Constructor

```
Repel_Behavior ...
RepelBehavior ...
```

### Properties

```
<Repel_Behavior>.name                String          Default: "Repel"

<Repel_Behavior>.repulsionSources    ArrayParameter Default: #() -- node array
    See Notes below

<Repel_Behavior>.targetComp          Integer          Default: 0
```

0-Closest Source : Each delegate is repelled by the closest of the assigned sources. Use this to have delegates assigned a single Repel behavior move away from sources in different directions.

1-Average of Sources : All delegates head to a common point determined by averaging all sources' locations.

```
<Repel_Behavior>.repelMethod          Integer          Default: 0 -- animatable
```

0-Angle : Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly away from the source. If the behavior's Weight is set to 1, the delegate will be given a force that points directly away from the source. If Weight is set to .5, the force will bisect the angle between its current direction and the direction opposite that of the source.

1-Force : Always applies a force directly away from the source, but at different magnitudes. If the behavior's Weight is 1, the magnitude of the force will be the average speed. If the Weight is .5, the magnitude of the force will be half the average speed.

```
<Repel_Behavior>.useRadii             Boolean          Default: True
```

When true, the behavior applies only to delegates closer to the target than the Outer Distance value.

```
<Repel_Behavior>.innerRadius          Float            Default: 0.0 -- animatable
```

The distance from the target at which the force is applied at full strength.

```
<Repel_Behavior>.outerRadius          Float            Default: 10.0 - animatable
```

The distance from the target at which the force begins to be applied.

```
<Repel_Behavior>.falloff              Float            Default: 2.0 -- animatable
```

```
<Repel_Behavior>.forceColor           Color             Default: (color 255 0 255)
```

The color used to draw the Repel force vector.

`<Repel_Behavior>.showRadii` Boolean Default: True

When true, the radii are displayed when the force is active.

`<Repel_Behavior>.displayForce` Boolean Default: True

When true, force exerted on the delegate(s) by the Repel behavior is drawn in the viewports as a vector during the simulation solution. If Use Radii is turned on, the radii are also displayed when the force is active.

## Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

`<Repel_Behavior>.repulsionSources`

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a Crowd/Behavior ArrayParameter element to undefined.

## See Also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Scripted\_Behavior : MAXObject

### Introduction to the Crowd System

The crowd system works by assembling all of the force *vectors* and speeds for a delegate from its active force behaviors. Then based upon the delegate's motion parameters, these force *vectors* and speeds are averaged and integrated to create a velocity for the delegate. This velocity may then be modified if the delegate has an active constraint behavior or if it has an active default **Avoidance** behavior.

After the velocity is set, if there is an active orientation behavior, it is used to set the orientation of the delegate, otherwise the orientation is calculated from the default delegate parameters. Finally, the velocity is integrated and a new position is calculated.

The new position of the delegate isn't set until after all of the forces are calculated for all of the behaviors.

For a **biped crowd**, bipeds work off of goals and not forces when determining which direction they should go in. So, in order to create a force behavior that will work with a Biped, a goal

position must be specified. No forces need to be integrated into velocities because the movement of the Biped isn't based upon calculating velocities but rather by selecting clips found in the motion flow graph.

### Constructor

```
Scripted_Behavior ...
ScriptedBehavior ...
```

### Properties

```
<scripted_behavior>.name                String    Default: "Scripted"

<Scripted_Behavior>.scriptContextName    String    Default: "apply"        Alias:
Script_Context_Name
```

The script functions declared in the Edit MaxScript window are declared in a hidden MaxScript context. This textbox specifies the name of the context. Giving a unique name here avoids conflicts between different scripted behaviors.

```
<Scripted_Behavior>.type                Integer    Default: 0        Alias:
Behavior_Type
```

One of three behavior types: Force, Constraint, or Orientation.

**Force:** Behaviors of this type force the delegates to move in a particular direction, for example, Seek and Repel. Force behaviors work by returning a force *vector* in the direction that the behavior wants the delegate to go in, and in some cases the speed it should be traveling at and the goal at which it is trying to reach.

**Constraint:** Behaviors of this type restrict the position and velocity of a delegate, for example, SurfaceFollow. Constraint behaviors set the velocity and sometimes may even change the delegate's position, in order to meet the constraint. You may only have one active constraint behavior per delegate per frame.

**Orientation:** Behaviors of this type affect only the orientation of the delegates, for example Orientation. These behaviors don't work with forces but instead return an orientation that the delegate should be at, represented by a quaternion. Any active orientation behavior will override the default orientation of the delegate. The velocity determines the default orientation. Like a constraint behavior, you may only have one active orientation behavior per delegate per frame.

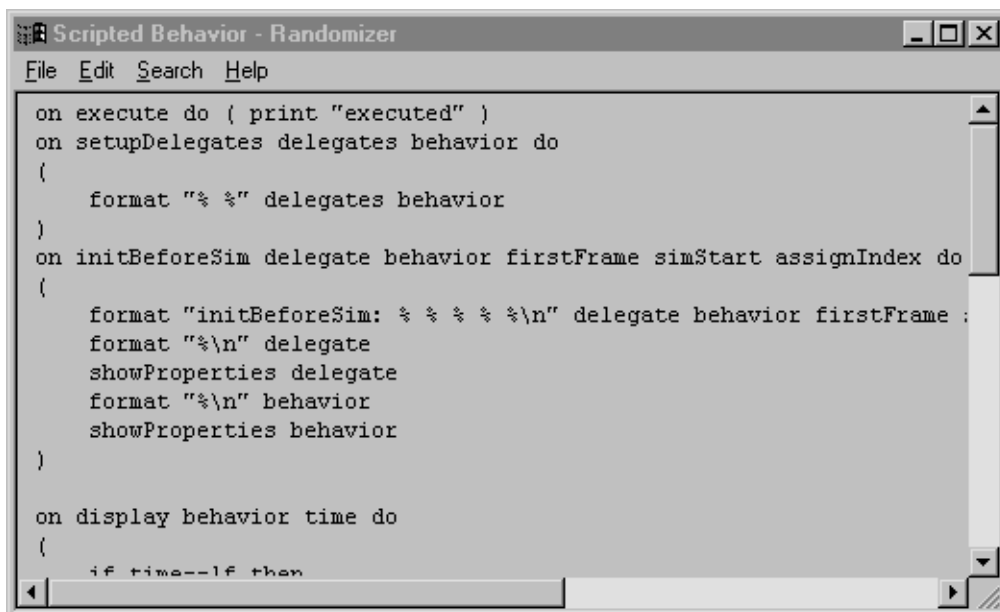
### Note

Though there are no avoid behavior types, a force or constraint behavior may be used to create an avoidance behavior. We **do recommend** though that the default **Avoidance** behavior be used due to the fact that it is well integrated into the crowd solution pipeline.

```
<scripted_behavior>.script : string
```

### Edit MaxScript

This button “**Edit MaxScript**”, in the Scripted Behavior Rollout, opens a MAXScript editor window that can be used to enter a scripted behavior script. The editor window is similar to other MaxScript editor window but is slightly customized for scripted behaviors.



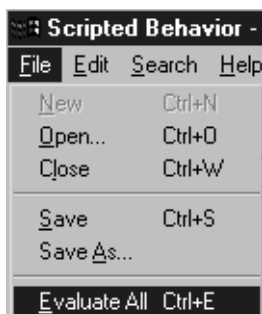
The following elements in the **File Menu** work differently

**New:** disabled since only one editor per scripted behavior can be open at any time

**Open:** opens a file in the existing editor window. **Any existing text will be lost.**

**Evaluate:** evaluates the script and saves it into the behavior’s paramblock

**Close:** saves into the behavior’s paramblock and closes the editor if no errors found



Scripted Behavior Syntax



The scripted behavior script is implemented by defining a set of **event handlers** that get called by the crowd system when solving the simulation. The event handlers are automatically enclosed into a hidden MAXScript context. The name of the context is taken from the dialog's **Script Context Name** text box or its equivalent **.scriptContextName**.

### Event Handlers for Scripted Behaviors

**on execute** do <expr>

Called whenever the script is evaluated. All initializing should be done here.

**on setupDelegates** <delegates> <behavior> **do** <expr>

<delegates> an array of all the delegates after they are ordered participating in the simulation. All delegates listed in the Behavior Assignments pane of the Behavior and Team Assignment dialog are found in this list.

<behavior> the current behavior

Called before the simulation starts.

### Note

All of the delegates are passed in, regardless if they have this behavior assigned, so the behavior knows the max number of delegates that may be used and therefore set up whatever data structures required. Some behaviors need to keep track of parameters on a per delegate basis. This function allows the behavior to set up data structures to get that info. Also that is one of the reasons why the **delegate.id** is unique, so it can be used to hash or index these data structures. Due to cognitive controllers, you never really know exactly all of the delegates which will be active for a particular behavior.

**on display** <behavior> <time> **do** <expr>

<behavior> the current behavior

<time> current time of the simulation

Called every time the crowd system is redrawn. You can use the low level drawing functions from the **gw** struct, *Viewport Drawing Methods* (see the *MAXScript Online Reference*), to have a custom display for the behaviors.

### Note

This handler gets called a number of times, so the system will slow down if implemented.

**on behaviorStarted** <delegate> <behavior> <time> **do** <expr>

<delegate> the delegate node the behavior is assigned to

<behavior> the current behavior

<time> current time of the simulation

Called whenever a behavior becomes active for a particular delegate. This can occur at the beginning of the simulation for an assigned behavior or when a cognitive controller activates the behavior.

**on initAtThisTime** <behavior> <time>

This function is called for each behavior at the beginning of the frame.

```
on applyForce <delegate> <behavior> <time> <numSubSamples> <displayHelpers>
<weight> <speed> do <expr>
```

If the behavior type is **Force** or **Orientation**, this handler is called every frame to move the delegate.

**<delegate>** the delegate node the behavior is assigned to  
**<behavior>** the current behavior  
**<time>** current time of the simulation  
**<numSubSamples>** is the number of sub samples required per frame.  
**<displayHelpers>** this value is true/false depending on the <Crowd>.update and <Crowd>.updateFrequency, as well as the current frame. If it is true, you should display any helper imagery you want using the display functions available from the delegates structure. For instance, most behaviors display their force if this is true, and pathfollow also displays its target. The *delegates.lineDisplay* (see page 651), *.sphereDisplay*, *.bboxDisplay*, and *.textDisplay* are all functions which can be used to draw a graphic primitive for a particular delegate when the crowd simulation is being solved. Please see Viewport Drawing Methods for additional viewport drawing methods.

#### Note

The graphic window functions gw.\* described in Viewport Drawing Methods will not display correctly or at all while in Step Solve.

**<weight>** this behavior's weight on the delegate

The event handler should return an array of the following type:

```
#(<int_flags>, <point3_force>, <point3_goal>, <float_speed>)
```

**<int\_flags>** sum of any of the following values:  
 0 - the return value **is not** used by the crowd system  
 1 - **only force** is affected  
 2 - **only goal** is affected  
 4 - **only speed** is affected

For example, a value of 5, (1+4), indicates that the force and speed are affected.

**<point3\_force>** the force vector. It should be unit normalized and then multiplied by the delegate's average speed. Modify this value, greater or less, to make the force stronger or weaker.  
**<point3\_goal>** the goal position in world space.  
**<float\_speed>** the net speed of the delegate. This value should be normalized to the average speed of the delegate. A value of 1.0 means go at the average speed.

```
on constraint <delegate> <behavior> <time> <numSubSamples> <displayHelpers>
<finalSet> <velocity> <speed> <pos> do <expr>
```

If the behavior type, <Scripted\_Behavior>.type, is **Constraint**, this handler is called at least once every frame to constrain the delegate. The constraint behavior will override any

other force behavior, the Avoidance Behavior, plus it will override any acceleration limits, speed limits etc. So it should be used with caution in a simulation.

The constraint behavior constrains the position that the delegate is moving towards by changing the current position that the delegate is already at, it's current velocity, and it's current speed. These values are combined to get the next position, **Next\_position = Current\_Position + Normalized(vel) \* Speed**. Changing the current position doesn't actually change where it is at though, it just changes the position that is used to calculate it's new position. This design allows the constraint to make the objects velocity change abruptly and keep its speed, thus conserving the energy of the delegate.

<delegate> the delegate node the behavior is assigned to

<behavior> the current behavior

<time> current time of the simulation

<numSubSamples> is the number of sub samples required per frame.

<displayHelpers> this value is true/false depending on the <Crowd>.update and <Crowd>.updateFrequency, as well as the current frame. If it is true, you should display any helper imagery you want using the display functions available from the delegate. For instance, most behaviors display their force if this is true, and pathfollow also displays its target. The *delegates* (see page 651) .lineDisplay, .sphereDisplay, .bboxDisplay, and .textDisplay are all functions which can be used to draw a graphic primitive for a particular delegate when the crowd simulation is being solved. Please see Viewport Drawing Methods for additional viewport drawing methods.

<finalSet> whether or not this is the final time this behavior will be called during the current frame.

## Note

In order to handle constraints working correctly within the avoidance system the constraint behavior may be called more than once per frame. When the passed in parameter, **finalSet**, is **true**, it will be the last time the constraint function is called for that frame.

Currently the constraint behavior's "on constraint" event handler is called twice per frame. The first call occurs before an existing Avoid\_Behavior with the passed in parameter **finalSet** having a value of **false**. The second call occurs, after the delegate's limits and Avoid\_Behavior are applied with the passed in parameter **finalSet** having a value of **true**. This evaluation order is done so that the delegate's limits and the Avoid\_Behavior can affect the changes that the constraint performs. The second call is made to make sure that the constraint is still being met.

One reason to check the state of **finalSet** is if internally something was cached and you didn't want it to change. For example, the **Surface\_Follow** constraint behavior keeps track of which triangle it's on and the barycentric coordinates of where it's at. The simulation will not change these values until finalSet is true.

<velocity> the current velocity of the delegate at this frame.

<speed> the current speed of the delegate at this frame.

<pos> the current position of the delegate at this frame.

**Note**

After the last delegate's "**on constraint**" has executed for the last frame of the simulation, "on setupDelegates delegates behavior do" will be called with an empty array "#() ReferenceTarget:Scripted\_Behavior". This is guaranteed and can be used by written constraint behaviors to clear out any internal caches.

The event handler should return an array of the following type:

```
#(<int_flags>, <point3_velocity>, <point3_pos>, <point3_goal>,
  <float_speed>)
```

**<int\_flags>** sum of any of the following values:

- 0 - the return value **is not** used by the crowd system
- 1 - the return value **is** used by the crowd system

**<point3\_velocity>** the velocity.

**<point3\_pos>** the modified current position that is used to calculate it's new position. In order to meet a constraint this behavior is allowed to modify the current position of a delegate.

**<point3\_goal>** the goal position in world space.

**<float\_speed>** the net speed of the delegate. This value should be normalized to the average speed of the delegate.

```
on orient <delegate> <behavior> <time> <velocity> do <expr>
```

If the behavior type, **<Scripted\_Behavior>.type**, is Orientation, this handler is called every frame. This handler is always called in conjunction with the applyForce handler.

**<delegate>** the delegate node the behavior is assigned to

**<behavior>** the current behavior

**<time>** current time of the simulation

**<velocity>** the current velocity of the delegate at this frame.

The event handler should return an array of the following type:

```
#(<int_flags>, quat_orientation)
```

**<int\_flags>** sum of any of the following values:

- 0 - the return value **is not** used by the crowd system
- 1 - the return value **is** used by the crowd system

**<quat\_orientation>** the orientation of the delegate in quaternions.

**Notes**

The purpose of passing the crowd delegate and behavior to all of the event handlers is that you can call persisted global, saved with scene, MAXScript functions that work for more than one crowd simulation.

**Example**

The following is a pair of scripted behaviors called "FormationConstraint" and "FormationOrientation".

The “FormationOrientation” will constrain the orientation to a fixed location, in this case based on the current delegate’s orientation found at its first rotation keyframe. The entire scripted orientation behavior can be implemented with one event handler:

```
on orient delegate behavior time vel do
(
    #(1,delegate.rotation.keys[1].value as quat )
)
```

The “FormationConstraint” keeps a team of delegates in a given positional formation while maintaining the pace and direction of a designated leader named “FormationLeader”. This is similar to a drum major and a marching band or like football blockers running in front of a running back who may dart in any direction. This person dictates how and where the formation will move. This “FormationLeader” is not using the scripted behavior.

This behavior will affect both the position and orientation of the delegate. The orientation is being calculated by the delegate’s velocity, and the velocity is being set correctly.

```
on execute do ( print " executed" )
on constraint delegate behavior time numsubsamples displayHelpers finalSet vel
speed pos do
(
    leader_velocity = delegates.velocity $TheFormationLeader
    Normalized_Leader_Velocity = normalize (delegates.velocity $TheFormationLeader)
    magVel = speed* Normalized_Leader_Velocity
    temp_dis = (pos + leader_velocity) - magVel
    the_goal = temp_dis + magVel

    if displayHelpers == true then
    (
        delegates.textDisplay delegate the_goal delegate.wirecolor delegate.name
        delegates.lineDisplay delegate pos (pos+leader_velocity) delegate.wirecolor
    true
        delegates.sphereDisplay delegate (pos+ (leader_velocity *
        $Crowd01.vectorScale)) 4 delegate.wirecolor
    )
    #(1, Normalized_Leader_Velocity, temp_dis, the_goal, speed)
)
```

## See Also

*Delegate:Helper (see page 651)*

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

## Seek\_Behavior : MAXObject

### Constructor

```
Seek_Behavior ...
SeekBehavior ...
```

### Properties

```
<seek_behavior>.name                String        Default: "Seek"

<seek_behavior>.targets              ArrayParameter Default: #()      -- node array
```

See Notes below.

Specify the object or objects as a stationary or moving target for delegates. Delegates move toward the target during the crowd simulation while turning as necessary.

```
<seek_behavior>.targetComp           Integer        Default: 0
```

0 – Closest Source Only: Each delegate seeks the closest of the assigned targets. Use this to have delegates assigned a single Seek behavior move in different directions.

1 – Average of Sources: All delegates head to a common point determined by averaging all targets' locations.

```
<seek_behavior>.seekMethod           Integer        Default: 0      -- animatable
```

0 – Angle: Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly towards the goal. If the Weight of the Seek behavior is 1, the delegate will be given a force that points directly towards the goal. If it's .5 it will bisect the angle between its current direction and the direction of the goal.

1 – Force: Applies a force in the direction of the goal always, but at different magnitudes. If the Weight of the Seek behavior is 1, the magnitude of the force will be the average speed. If the Weight of the Seek behavior is .5, the magnitude of the force will be half the average speed.

```
<seek_behavior>.forceColor           Color          Default: (color 0 255 0)
```

The color used to draw the Seek force vector during the solution.

```
<seek_behavior>.displayForce         Boolean         Default: True
```

When true, force exerted on the delegate(s) by the Seek behavior is drawn in the viewports as a vector during the simulation solution.

```
<Seek_Behavior>.useRadii             Boolean         Default: False
```

When true, the Seek behavior applies only to delegates less than the Outer Radius distance from the target.

```
<Seek_Behavior>.showRadii            Boolean         Default: True
```

When true, the radii are displayed when the force is active.

|                             |       |              |               |
|-----------------------------|-------|--------------|---------------|
| <Seek_Behavior>.innerRadius | Float | Default: 0.0 | -- animatable |
|-----------------------------|-------|--------------|---------------|

The distance from the target at which Seek is applied at full strength.

|                             |       |               |               |
|-----------------------------|-------|---------------|---------------|
| <Seek_Behavior>.outerRadius | Float | Default: 10.0 | -- animatable |
|-----------------------------|-------|---------------|---------------|

The distance from the target at which Seek begins to be applied.

|                         |       |              |               |
|-------------------------|-------|--------------|---------------|
| <Seek_Behavior>.falloff | Float | Default: 2.0 | -- animatable |
|-------------------------|-------|--------------|---------------|

## Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

|                         |
|-------------------------|
| <Seek_Behavior>.targets |
|-------------------------|

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a Crowd/Behavior ArrayParameter element to undefined.

## See Also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Space\_Warp\_Behavior: MAXObject

### Constructor

```
Space_Warp_Behavior ...
WSMBehavior ...
```

### Properties

|                                  |        |                            |
|----------------------------------|--------|----------------------------|
| <space_warp_behavior>.name       | String | Default: "Space Warp"      |
| <space_warp_behavior>.spaceWarp  | Node   | Default: Undefined         |
| <space_warp_behavior>.forceColor | Color  | Default: (color 255 255 0) |

The color used to draw the Space Warp force vector during the solution.

|                                    |         |               |
|------------------------------------|---------|---------------|
| <space_warp_behavior>.displayForce | Boolean | Default: True |
|------------------------------------|---------|---------------|

When true, force exerted on the delegate(s) by the Space Warp behavior is drawn in the viewports as a vector during the simulation solution.

See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Speed\_Vary\_Behavior : MAXObject

### Constructor

```
Speed_Vary_Behavior ...
SpeedVaryBehavior ...
```

### Properties

```
<speed_vary_behavior>.name           String           Default: "Speed Vary"

<speed_vary_behavior>.period          Integer          Default: 10  -- animatable
```

Specifies how many frames should elapse before a new speed is chosen.

```
<speed_vary_behavior>.period_deviation Float           Default: 0.5  -- animatable;
Alias: Period_Deviation
```

Specifies the maximum amount by which Period should vary. Each time a period ends, character studio takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Period setting, and adds the result to Period.

Range=0.0 to 1.0.

```
<speed_vary_behavior>.center_speed    Float           Default: 1.0  -- animatable;
Alias: Center_Speed
```

Specifies the speed the delegate should change to. Center is a multiplier: A value of 0.0 means to stop, a value of 1.0 means to move at its average speed, and a value greater than 1.0 means to move faster than its average speed. Range=0.0 to 99,999.0.

```
<speed_vary_behavior>.speed_deviation Float           Default: 0.25 -- animatable;
Alias: Speed_Deviation
```

Specifies the maximum amount by which the delegate's calculated speed (Average Speed\*Center) should vary. Each time a period ends, character studio takes a random number between the negative and positive values of the Deviation setting, multiplies it by the calculated speed, and adds the result to the calculated speed. Range=0.0 to 99,999.0.

```
<speed_vary_behavior>.seed            Integer          Default: 1    -- animatable;
Alias: Speed_Vary_Seed
```

Specifies a seed value for randomizing the Wander behavior.

```
<speed_vary_behavior>.accelPeriod     Float           Default: 0.5  -- animatable;
Alias: Acceleration_Period
```



Specifies the rate at which the delegate's speed should change. An acceleration of 1.0 means that the transition to the new speed will proceed as quickly as possible to its new speed, while a value of 0.0 means the transition will take the entire period. Range=0.0 to 1.0.

```
<speed_vary_behavior>.accelDeviation    Float          Default: 0.5  --  animatable;
Alias: Acceleration_Deviation
```

Specifies the maximum amount by which the delegate's calculated speed (Average Speed\*Center) should vary. Each time a period ends, character studio takes a random number between the negative and positive values of the Deviation setting, multiplies it by the calculated speed, and adds the result to the calculated speed. Range=0.0 to 99,999.0.

## See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Surface\_Arrive\_Behavior : MAXObject

### Constructor

```
Surface_Arrive_Behavior ...
SurfaceArriveBehavior ...
```

### Properties

```
<Surface_Arrive_Behavior>.name                String  Default: "Surface Arrive"
```

```
<Surface_Arrive_Behavior>.surfaces            ArrayParameter Default: #()      -- node
array
```

See Notes below

```
<Surface_Arrive_Behavior>.disableAfterArriving Boolean Default: True
```

When true, turns off the Surface Arrive behavior after the delegate arrives at the surface.

```
<Surface_Arrive_Behavior>.rate                Float    Default: 0.5    --
animatable
```

A multiple of the delegate's Max Accel setting that specifies the acceleration with which it will try to arrive. A value of 1.0 uses the full acceleration of the delegate.

```
<Surface_Arrive_Behavior>.rateDeviation        Float    Default: 0.0    -- animatable
```

Adds a variation to the to the rate setting.

```
<Surface_Arrive_Behavior>.speed                Float    Default: 0.0    --
animatable
```

The speed at which to arrive and relative to the speed of the target.

```
<Surface_Arrive_Behavior>.speedDeviation      Float    Default: 0.0    --
animatable
```

Adds a variation to the to the speed setting.

```
<Surface_Arrive_Behavior>.distance            Float    Default: 1e+008  --
animatable
```

The maximum radial distance from the target within which the behavior will be active.

Until the delegate is within this radius, the behavior will have no influence.

```
<Surface_Arrive_Behavior>.distanceDeviation    Float    Default: 0.0    -- animatable
```

Adds a variation to the to the distance setting.

```
<Surface_Arrive_Behavior>.offset              Float    Default: 0.0    --
animatable
```

Specifies a consistent distance from the calculated arrival point, based on the surface normal, for the delegate to use.

```
<Surface_Arrive_Behavior>.facing              Boolean   Default: False   --
animatable
```

When true, the delegate will try to arrive only at points on triangles on the surface that are facing it.

```
<Surface_Arrive_Behavior>.random_closest      Integer   Default: 0
```

0 – Random: The software chooses a random point on the target surface as the arrival point.

1 – Closest: The software chooses the closest point on the target surface as the arrival point.

```
<Surface_Arrive_Behavior>.everyFrame          Boolean   Default: False   --
animatable
```

When true, the software chooses arrival points for delegates at every frame. Available only when Closest is chosen.

```
<Surface_Arrive_Behavior>.displayOffset       Boolean   Default: False
```

When true, shows the Offset distance as lines emanating from each vertex in the surface object, perpendicular to the surface.

```
<Surface_Arrive_Behavior>.height              Float    Default: 0.0    --
animatable
```

Specifies a distance from the arrival point along its face normal. This is the point that the delegate will go to first before descending to the arrival point.

```
<Surface_Arrive_Behavior>.heightDeviation     Float    Default: 0.0    -- animatable
```

Adds random variation to the to the Height setting. See introduction for the formula used to calculate the deviation.

```
<Surface_Arrive_Behavior>.descentStart          Float    Default: 0.0    --
animatable
```

Specifies the distance between the delegate and the arrival point at which the descent should start.

#### Note

Be careful that Descent Start is set high enough that the delegate won't overshoot when descending because its speed is too high and deceleration too low, compared to when it should start descending.

```
<Surface_Arrive_Behavior>.descentstartDeviation Float    Default: 0.0    -- animatable
```

Adds a variation to the to the Descent Start setting.

```
<Surface_Arrive_Behavior>.useNormal              Boolean   Default: False    Alias:
Off_This_Normal
```

When true, use vector specified in the **.xNormal**, **.yNormal**, and **.zNormal** to specify the angle at which the final approach occurs.

```
<Surface_Arrive_Behavior>..xNormal              Float    Default: 0.0    --
animatable; Alias: X_Normal
```

Specify the final approach X value for the vector in world coordinates.

```
<Surface_Arrive_Behavior>..yNormal              Float    Default: 0.0    --
animatable; Alias: Y_Normal
```

Specify the final approach Y value for the vector in world coordinates.

```
<Surface_Arrive_Behavior>..zNormal              Float    Default: 1.0    --
animatable; Alias: Z_Normal
```

Specify the final approach Z value for the vector in world coordinates.

```
<Surface_Arrive_Behavior>..seed                 Integer   Default: 1        --
animatable
```

Affects the random numbers used to calculate the Deviation settings.

```
<Surface_Arrive_Behavior>.targetColor           Color     Default: (color 0 0 127.5)
```

The color used to draw the target icon.

```
<Surface_Arrive_Behavior>.displayTarget         Boolean   Default: True
```

When true, displays the target icon which appears during the solution when a new interim goal is calculated for the delegate.

```
<Surface_Arrive_Behavior>.targetScale           Float    Default: 5.0    --
animatable
```

Specifies the overall size of the target icon.

## Methods

**SurfaceArriveBhvr .Getstate** <Surface\_Arrive\_Behavior> <delegate\_node>

Returns an integer value giving the state the delegate is in with respect to the Surface\_Arrive\_Behavior. A return value of -1 means that the delegate isn't active for that state, 0 means that the delegate is outside the distance radius and so isn't arriving, 1 means that the delegate is in the process of arriving, and 2 means that the delegate has arrived. If 'Disable After Arriving' is TRUE, the delegates will never reach state 2.

The main purpose of this function is so that the 'State' of a delegate as it arrives to a surface can be determined from MAXScript. It can be used for both the cognitive controller transitions as well as the script definable in MotionClips. See the **ClipState Dialog** and the **Cognitive Controller** topics in the **character studio 3** online reference for more details.

Track View > Global Tracks > Block Control > GlobalClip Properties > Synthesis dialog > State Tab > New State > Edit Properties > ClipState dialog

Select a Crowd helper. > Modify panel > Global Clip Controllers rollout > New > Choose GlobalClip object. > Select object in list. > Edit > Synthesis dialog > State Tab > New State > Edit Properties > Clip State dialog

Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > Cognitive Controllers

Select a Crowd object. > Modify panel > Setup rollout > Cognitive Controllers

**SurfaceArriveBhvr .getPos** <Surface\_Arrive\_Behavior> <delegate\_node>

Returns a Point3 value of the position that the delegate is arriving to at the time the function is called.

## Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

<Surface\_Arrive\_Behavior>.**surfaces**

The following MAXScript operations will cause Crowd to fail, either right away or later:  
**NEVER** set a Crowd/Behavior ArrayParameter element to undefined.

## See Also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

## Surface\_Follow\_Behavior : MAXObject

### Constructor

```
Surface_Follow_Behavior ...
SurfaceFollowBehavior ...
```

### Properties

```
<Surface_Follow_Behavior>.name          String          Default: "Surface Follow"
```

```
<Surface_Follow_Behavior>.surfaces      ArrayParameter Default: #()    -- node array
```

See Notes below.

```
<Surface_Follow_Behavior>.useProjection Boolean          Default: False
```

When true, Surface Follow calculates delegate direction from the specified vector, rather than using the default.

```
<Surface_Follow_Behavior>.xVector       Float           Default: 0.0    -- animatable
```

Specifies the x component of a vector using world coordinates. . Range=-1.0 to 1.0.

```
<Surface_Follow_Behavior>.yVector       Float           Default: 0.0    -- animatable
```

Specifies the y component of a vector using world coordinates. . Range=-1.0 to 1.0.

```
<Surface_Follow_Behavior>.zVector       Float           Default: 1.0    -- animatable
```

Specifies the z component of a vector using world coordinates. . Range=-1.0 to 1.0.

```
<space_warp_behavior>.targetColor       Color           Default: (color 0 0 127.5)
```

Sets the color used to draw the Surface Follow during the solution.

```
<space_warp_behavior>.displayTarget     Boolean          Default: True
```

When true, the interim goal for each delegate influenced by the Surface Follow behavior is drawn in the viewports as a wireframe sphere during the simulation solution.

### Notes

If the delegate starts out away from the surface to be followed then the target is most visible before the delegate reaches the surface where the target is positioned along the surface edge.

While the delegate is actually following the surface, the target is usually coincident with the delegate because Surface follow sets a new destination only a frame or two ahead.

```
<space_warp_behavior>.targetScale       Float           Default: 5.0
```

Specifies the overall size of the target icon.

```
<Surface_Follow_Behavior>.offset       Float           Default: 0.0    -- animatable
```

Specifies the delegate's distance above the surface, using the surface normal.

<Surface\_Follow\_Behavior>.displayOffset Boolean Default: False

When true, shows the .offset distance as lines emanating from each vertex in the surface object, perpendicular to the surface.

### Notes

You can perform the following MAXScript operations

```
deleteitem <array> <itemnumber>
<array> = #(item,item...)
<array> = append <array> <item>
```

on all of the properties containing an ArrayParamater of objects listed below. You can also undo/redo these operations.

<Surface\_Follow\_Behavior>.surfaces

The following MAXScript operations will cause Crowd to fail, either right away or later:

**NEVER** set a Crowd/Behavior ArrayParameter element to undefined.

### See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Wall\_Repel\_Behavior : MAXObject

### Constructor

```
Wall_Repel_Behavior ...
WallRepelBehavior ...
```

### Properties

```
<Wall_Repel_Behavior>.name String Default: "Wall Repel"
<Wall_Repel_Behavior>.repelGrids ArrayParameter Default: #() -- node array
```

Set the repelling grid.

```
<Wall_Repel_Behavior>.repelMethod Integer Default: 0 -- animatable
```

Determines whether delegate direction as influenced by the behavior is calculated by an angular method or a force method.

0 – Angle: Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly away from the source. If the behavior's Weight is set to 1 the delegate will be given a force that points directly away from the source. If Weight is set to .5, the force will bisect the angle between its current direction and the direction opposite that of the source.

1 – Force: : Always applies a force directly away from the source, but at different magnitudes. If the behavior's Weight is 1, the magnitude of the force will be the average speed. If the Weight is .5, the magnitude of the force will be half the average speed.

<Wall\_Repel\_Behavior>.**repelDirection** Integer Default: 0 -- animatable  
Determines whether the grid repels from its positive-axis side, its negative-axis side, or both.

0 – Positive Axis: The grid repels only from the positive-axis side.

1 – Negative Axis: The grid repels only from the negative-axis side.

2 – Both Axes: The grid repels from both sides.

<Wall\_Repel\_Behavior>.**useDistance** Boolean Default: True

When true, the behavior applies only to delegates closer to the target than the **outerRadius** value.

<Wall\_Repel\_Behavior>.**innerDistance** Float Default: 0.0 -- animatable

The distance from the target at which the force is applied at full strength.

<Wall\_Repel\_Behavior>.**outerDistance** Float Default: 10.0 -- animatable

The distance from the target at which the force begins to be applied.

<Wall\_Repel\_Behavior>.**fallOff** Float Default: 2.0 -- animatable

<Wall\_Repel\_Behavior>.**showDistance** Boolean Default: True

Shows the inner and outer distance settings as grids offset from the target grid in the viewports. The **.innerDistance** grid is light blue, while **.outerDistance** grid is blue-white.

<Wall\_Repel\_Behavior>.**gridSpacing** Integer Default: 500

Specifies the spacing of grid lines used to draw the Inner/Outer Distance grids. The lower the value, the closer the spacing.

<Wall\_Repel\_Behavior>.**gridEnd** Boolean Default: True Alias:  
End\_force\_at\_grid\_edges

When true, the force emanates only from the grid object. When off, the force emanates from an imaginary infinite grid created by extending the grid plane in all directions.

<Wall\_Repel\_Behavior>.**forceColor** Color Default: (color 255 0 255)

Sets the color used to draw the Wall Repel force during the solution.

<Wall\_Repel\_Behavior>.**displayForce** Boolean Default: True

The force, when activated, is drawn in the viewports as a wireframe rectangle during the simulation solution.

## See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

## Wall\_Seek\_Behavior : MAXObject

### Constructor

```
Wall_Seek_Behavior ...
WallSeekBehavior ...
```

### Properties

```
<Wall_Seek_Behavior>.name                String      Default: "Wall Seek"

<Wall_Seek_Behavior>.seekGrids            ArrayParameter  Default: #()  -- node array
```

Set the repelling grid.

```
<Wall_Seek_Behavior>.seekMethod          Integer      Default: 0      -- animatable
```

Determines whether delegate direction as influenced by the behavior is calculated by an angular method or a force method.

0 – Angle: Applies a force to the delegate based on the angle between the delegate's current direction and the direction it would need to take in order to be moving directly toward the target. If the behavior's Weight is set to 1, the delegate will be given a force that points directly away from the target. If Weight is set to .5, the force will bisect the angle between its current direction and the direction opposite that of the target.

1 – Force: Always applies a force directly toward the target. If the behavior's Weight is 1, the magnitude of the force will be the average speed. If the Weight is .5, the magnitude of the force will be half the average speed.

```
<Wall_Seek_Behavior>.seekDirection        Integer      Default: 0      -- animatable
```

Determines whether the grid attracts from its positive-axis side, its negative-axis side, or both.

0 – Positive Axis: The grid attracts only from the positive-axis side.

1 – Negative Axis: The grid attracts only from the negative-axis side.

2 – Both Axes: The grid attracts from both sides.

```
<Wall_Seek_Behavior>.useDistance          Boolean      Default: True
```

When true, the behavior applies only to delegates closer to the target than the Outer Distance value.

```
<Wall_Seek_Behavior>.innerDistance        Float        Default: 0.0  -- animatable
```

The distance from the target at which the force is applied at full strength.

```
<Wall_Seek_Behavior>.outerDistance        Float        Default: 10.0 -- animatable
```

The distance from the target at which the force begins to be applied.

```
<Wall_Seek_Behavior>.falloff              Float        Default: 2.0  -- animatable
```



|                                                         |         |               |        |
|---------------------------------------------------------|---------|---------------|--------|
| <Wall_Seek_Behavior>.showDistance                       | Boolean | Default: True |        |
| <Wall_Seek_Behavior>.gridSpacing                        | Integer | Default: 500  |        |
| <Wall_Seek_Behavior>.gridEnd<br>End_force_at_grid_edges | Boolean | Default: True | Alias: |

When true, the force emanates only from the grid object. When off, the force emanates from an imaginary infinite grid created by extending the grid plane in all directions.

|                                 |       |                            |  |
|---------------------------------|-------|----------------------------|--|
| <Wall_Seek_Behavior>.forceColor | Color | Default: (color 255 0 255) |  |
|---------------------------------|-------|----------------------------|--|

Sets the color used to draw the Seek force vector during the solution.

|                                   |         |               |  |
|-----------------------------------|---------|---------------|--|
| <Wall_Seek_Behavior>.displayForce | Boolean | Default: True |  |
|-----------------------------------|---------|---------------|--|

When true, force exerted on the delegate(s) by the Seek behavior is drawn in the viewports as a vector during the simulation solution. If **useRadii** is turned on, the radii are also displayed when the force is active.

## See also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## Wander\_Behavior : MAXObject

### Constructor

```
Wander_Behavior ...
WanderBehavior ...
```

### Properties

|                                                  |         |                   |                |
|--------------------------------------------------|---------|-------------------|----------------|
| <wander_behavior>.name                           | String  | Default: "Wander" |                |
| <wander_behavior>.period<br>Alias: Wander_Period | Integer | Default: 10       | -- animatable; |

Specifies how many frames should elapse before a new direction is chosen.

|                                                                     |       |              |                |
|---------------------------------------------------------------------|-------|--------------|----------------|
| <wander_behavior>.periodDeviation<br>Alias: Wander_Period_Deviation | Float | Default: 0.5 | -- animatable; |
|---------------------------------------------------------------------|-------|--------------|----------------|

Specifies the maximum amount by which Period should vary. Each time a period ends, character studio takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Period setting, and adds the result to Period.

Range=0.0 to 1.0.

|                             |       |              |               |
|-----------------------------|-------|--------------|---------------|
| <wander_behavior>.turnAngle | Float | Default: 0.5 | -- animatable |
|-----------------------------|-------|--------------|---------------|

Specifies how far to turn when changing direction. A small value means to change direction only by a small amount, while as the value approaches 1.0 it will randomly turn in any direction. Range=0.5 to 1.0.

```
<wander_behavior>.turnPeriod          Float          Default: 0.5  --  animatable
```

Specifies how long over the current period it takes to turn. A value of 0.0 means that it will rotate as quickly as possible to face a direction and then travel in that a direction, while a value of 1.0 means it will take the entire period to rotate in that direction. Range=0.5 to 1.0.

```
<wander_behavior>.turnPeriodDeviation  Float          Default: 0.5  --  animatable
```

Specifies the maximum amount by which Angle should vary. Each time a period ends, character studio takes a random number between the negative and positive values of the Deviation setting, multiplies it by the Angle setting, and adds the result to Angle. Range=0.0 to 1.0.

```
<wander_behavior>.seed                  Integer        Default: 1    --  animatable
```

Specifies a seed value for randomizing the Wander behavior.

```
<wander_behavior>.forceColor           Color          Default: (color 0 255 255)
```

The color used to draw the Wander force vector during the solution.

```
<wander_behavior>.displayForce         Boolean         Default: True
```

When true, force exerted on the delegate(s) by the Wander behavior is drawn in the viewports as a vector during the simulation solution.

## See Also

*Crowd : Helper (see page 649)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

---

## MotionClips and GlobalMotionClip

```
globalMotionClipOps          const          StructDef Struct:
globalMotionClipOps (
```

```
globalMotionClipOps.loadfile          Primitive  loadfile()
```

'loadfile' expects a GlobalMotionClip, then a filename for a "\*.ant" GlobalMotionClip file to load.

```
globalMotionClipOps.savefile          Primitive  savefile()
```

'savefile' expects a GlobalMotionClip, then a filename for the "\*.ant" file to save.

```
globalMotionClipOps.synthesize        Primitive  synthesize()
```

‘synthesize’ expects a GlobalMotionClip and will synthesize it.

Note that the **GlobalMotionClip** objects are found under the **GlobalTracks.Block\_Control** list controller.

So to call savefile, one would call:

```
GlobalMotionClipOps.savefile GlobalTracks.Block_Control.globalMotionClip__globalObject
“globalObject.ant”
```

The following classes exist but are not supported.

```
MotionClipSlaveRotation : RotationController {4306d06,97c6025}
MotionClip : FloatController {e937ded,5a002ad2}
MotionClipSlavePos : PositionController {18c22740,522802b9}
ClipAssignerReferenceTarget : ReferenceTarget {2cb93ce2,33825f4}
MotionClipSlaveFloat : FloatController {31822cdf,34c146b}
Global_MotionClip : MasterBlockController {37d25755,278c6957}
ClipState : ReferenceTarget {455d5a81,3dcb1dc2}
ClipAssigner : ReferenceTarget {49f14ffa,33801afd}
MasterMotionClip : MasterBlockController {532a2038,6acd39bf}
MotionClipSlave_Point3 : Point3Controller {6c95514a,801b4f}
MotionClipSlaveScale : ScaleController {71031345,770b4347}
MotionClipFloatController : FloatController {7f016988,1f9f0342}
```

---

## SpaceWarps

---

### Vector\_Field: SpacewarpObject

#### Constructor

```
Vector_Field ...
VectorField ...
```

#### Properties

|                       |       |              |          |
|-----------------------|-------|--------------|----------|
| <Vector_Field>.Length | Float | Default: 0.0 | -- world |
| units                 |       |              |          |

Specify the length dimension of the vector field lattice. The lattice should be larger than the vector field object.

|                      |       |              |          |
|----------------------|-------|--------------|----------|
| <Vector_Field>.Width | Float | Default: 0.0 | -- world |
| units                |       |              |          |

Specify the width dimension of the vector field lattice. The lattice should be larger than the vector field object.

|                       |       |              |          |
|-----------------------|-------|--------------|----------|
| <Vector_Field>.Height | Float | Default: 0.0 | -- world |
| units                 |       |              |          |

Specify the height dimension of the vector field lattice. The lattice should be larger than the vector field object.

<Vector\_Field>. **LenSegs** Integer Default: 1 Alias:  
Length\_Segments

Specify the resolution of the vector field lattice's length segments. The greater the resolution, the higher the accuracy of the simulation.

<Vector\_Field>. **WidSegs** Integer Default: 1 Alias:  
Width\_Segments

Specify the resolution of the vector field lattice's width segments. The greater the resolution, the higher the accuracy of the simulation.

<Vector\_Field>. **HgtSegs** Integer Default: 1 Alias:  
Height\_Segments

Specify the resolution of the vector field lattice's height segments. The greater the resolution, the higher the accuracy of the simulation.

<Vector\_Field>. **showLattice** Boolean Default: True

Set on to display the vector field lattice as a yellow wireframe box. The vectors are generated at lattice intersections inside the vector field range.

<Vector\_Field>. **showRange** Boolean Default: True

Set on to display the volume about the obstacle object within which vectors are generated as an olive-colored wireframe.

<Vector\_Field>. **showVectors** Boolean Default: False

Set on to display vectors, which appear as blue lines emanating outward from the lattice intersections within the range volume.

<Vector\_Field>. **showSurfSamps** Boolean Default: False

Set on to display short green lines emanating from sample points on the surface of the obstacle object.

<Vector\_Field>. **vecScale** Float Default: 1.0 -- world units

Scales the vectors so they're more visible or less obtrusive. This setting does not affect the strength of the vectors only their visibility.

<Vector\_Field>. **iconSize** Float Default: 0.0 -- world units

Adjusts the size of the Vector Field space warp icon. The icon is a pair of crossed double-headed arrows. Increase the size for easier viewport selection.

<Vector\_Field>. **strength** Float Default: 1.0 -- animatable;  
world units

Sets the degree of effect the vectors have on the movement of an object entering the vector field. Changing Strength does not require that you recalculate the vector field.

<Vector\_Field>. **falloff** Float Default: 2.0 -- animatable;  
world units

Determines the rate at which the strength of the vectors falls off with distance from the surface of the object. A value of 0 will make all the vectors the same size. A value greater than 0 will make them get smaller as they get further away. A value less than 0 will make them get larger as they get further away.

```
<Vector_Field>.direction Integer Default: 1 -- animatable
```

Sets whether the force generated by the vectors works parallel or perpendicular to the vector field. Because the vectors are perpendicular to the object surface, and you typically would want delegates to travel parallel to the surface, you would normally use a perpendicular force.

0 – Parallel

1 - Perpendicular

```
<Vector_Field>.pull Float Default: 0.0 -- animatable; world units
```

Adjusts objects' position relative to the field. Available only when Perpendicular is chosen. Range=-1.0 to 1.0. Objects moving perpendicular to a vector field sometimes tend to drift away from it, due to lack of subsampling. The Pull parameter helps to pull objects back. Pull values greater than 0 create a pulling force towards the source of the vector field vector. Values less than 0 pull the force towards the direction in which the vector field's vector is pointing. A value of 0.0 produces a force perfectly perpendicular to the vector field's vector.

```
<Vector_Field>.object Node Default: Undefined Alias: Vector_Field_Object
```

The obstacle object around which the vector field is to be generated. Only primitives and unmodified editable mesh objects can be used as obstacles. The object should be fully enclosed in the Vector Field lattice.

```
<Vector_Field>.range Float Default: 1.0 -- world units
```

Determines the volume within which vectors are generated. The Range is represented in viewports as an olive-colored wireframe that starts out the same size and shape as the obstacle object. Increasing the Range setting moves the wireframe away from the obstacle object in the direction of its surface normals.

## Notes

In crowd simulations, the Range outline is where the delegates start to “see” the obstacle object, and begin to turn to avoid it. If your crowd members are penetrating the obstacle, or even just coming too close to it before turning, increase the Range setting. Also try increasing the Vector Field lattice resolution and/or the Sample Res setting.

```
<Vector_Field>.resolution Integer Default: 1
```

Acts as a multiplier of the effective sampling rate used on the obstacle object's surface to calculate vector directions in the field. The basic sampling rate is determined by the program from the size of the lattice and the size of each polygon.

<Vector\_Field>. **flipFaces** Boolean Default: False

Setting to true causes flipped normals to be used during the computation of the vector field. By default, vectors are generated in the same direction as the obstacle object's face normals, so that assuming its face normals point outward, objects move around its exterior in a crowd simulation. However, if you want objects to remain within an object's interior, turn on **flipFaces**.

<Vector\_Field>. **blendStart** Float Default: 0.0 -- world units

The distance from the object at which blending the vectors starts.

<Vector\_Field>. **blendFalloff** Float Default: 2.0 -- world units

The falloff of the blend of the surrounding vectors.

<Vector\_Field>. **blendWidSegs** Integer Default: 1

The number of adjacent lattice points to blend on the X axis.

<Vector\_Field>. **blendLenSegs** Integer Default: 1

The number of adjacent lattice points to blend on the Y axis.

<Vector\_Field>. **blendHgtSegs** Integer Default: 1

The number of adjacent lattice points to blend on the Z axis.

## Methods

vfields.**computeVectors** <Vector\_Field>

Calculates the vector field using the current Vector Field parameters. Always recalculate the vector field after changing any of the non-display related parameters.

vfields.**BlendVectors** <Vector\_Field>

Blends the vectors for reducing abrupt changes in angles of neighboring vectors.

## Associated Methods

**bindSpaceWarp** <node> <Vector\_Field\_node>

## Associated Binding Modifier

Vector\_Field\_Mod

This modifier is automatically created by the *bindSpaceWarp()* (see the MAXScript Online Reference) method, and is not otherwise creatable by MAXScript. There are no properties associated with this binding modifier.

## See Also

*Node Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*MAXWrapper Common Properties, Operators, and Methods (see the MAXScript Online Reference)*

*Value Common Properties, Operators, and Methods (see the MAXScript Online Reference)*





---

# CS3Tools.cui Tutorial

The **CS3Tools Custom User Interface** implements several very useful tools. This tutorial will concentrate on the Custom Keys Utility, a custom keys mechanism that gives one-touch access to 6 key frame settings that you use most when animating a Biped. The Custom Keys Utility uses 13 icon buttons. The first 6 are “Set custom Biped Keys”. The next 6 are “Change Preset to current key”. The following one is “Launch Preset Floater”.

Once you have stored your favorite custom keys, animation workflow is greatly improved, since you spend little or no time tuning default key frame attributes that don’t fit your current character’s situation or style. Simply pose your character and click on the custom key icon that fits your current situation.

---

## Overview

Custom Keys are implemented via a new dedicated 3DS MAX Tool Bar, called Character Studio Tool. This tool bar is the home base for a growing number of MAXScript-enabled character studio commands. All operations associated with Tool Bars are available. For example, individual icons can be relocated, deleted, and renamed. Tool tips can also be customized, as implied above, to let you annotate each custom key type for rapid recall. Custom Keys are displayed on the Character Studio Tool Bar as color-coded “set key” icons, similar to the standard Biped red set key icon in the Biped Motion panel. Each color indicates a specific custom key type. There are two icons for each key type – one for setting a key, and second for storing the preset value, based on the currently selected track and key. The second icon – used for storing the preset values – displays a small black arrow pointing to the center of the icon – indicating that a value will be stored. Once you have set your keys as you like them, you may choose to delete the storage icons entirely, to save space. You can also float or re-dock the Custom Key icons to customize access to suit your style and needs. Additionally, you can launch a floating rollout of the same set of custom keys.

To use Custom Keys, simply set a Biped key as you normally would using the Biped Motion Panel and set its Key Info settings as you prefer. Next store that key for the currently selected track at the current frame into one of the six preset custom key icons using the set key icons with the black arrows.

---

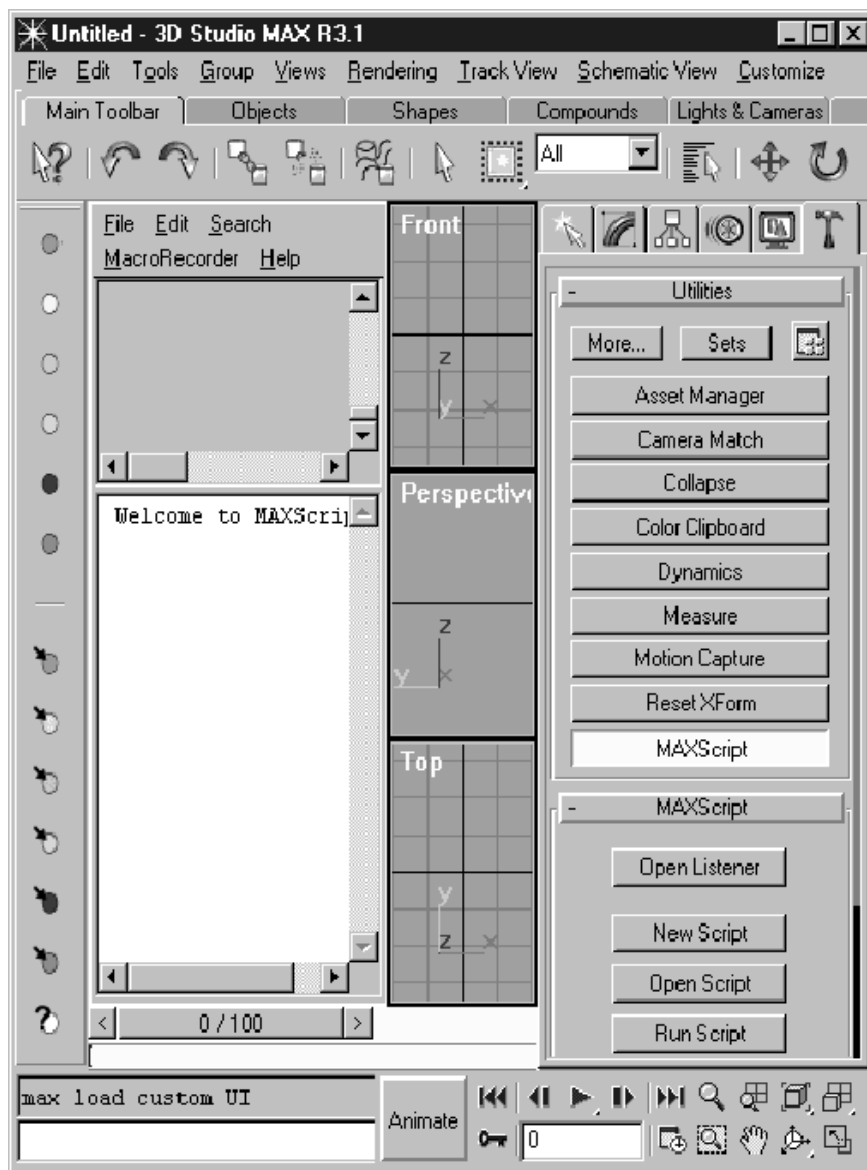
## Launching the CS3CustomKeys User Interface

The file named CS3Tools.cui is found in the 3DS MAX UI directory.

Choose Customize -> Load Custom UI...> CS3Tools.cui

The user interface is displayed on the left side of the screen made with bitmapped buttons icons. You can also build a Tab Panel from scratch and manually add these icons by selecting from the Character Studio Tool category of the Customize UI Dialog.

See help page *Macro Scripts* (see the *MAXScript Online Reference*) in the “Interacting with the 3DS MAX User Interface” topic and *Defining Macro Scripts* (see the *MAXScript Online Reference*) in the “Creating MAXScript Tools” topic for additional details regarding Macro Scripts.



The icon buttons of CS3Tools.cui are displayed in the above image.

The bitmaps for the custom ui are installed into the 3DS MAX UI directory. See the *Creating Icon Bitmap Files* topic in the *MAXScript Online Reference* for more information on creating and customizing icon bitmaps.

For access to Custom User Interface (CUI) files via MAXScript, see the *MAXScript Online Reference* help page *Custom User Interface Files* in the “File Access”. Note that the \*.cui file format specification is not currently provided.

---

## File Details

**character studio 3** ships with the following Character Studio Tool files:

```
..\ui\CS3Tools.cui
..\Scripts\Startup\CS3Customkeys.ms
..\Scripts\Startup\CS3CustomKeys-presets.ini
..\Scripts\Startup\CS3convertBips.ms
..\Scripts\Startup\CS3Bip2BonesFloater.ms
..\Stdplugins\stdscripts\CS3CustomKeysPresetFloater.ms
```

Tutorial\_Filename\_CS3CustomKeysPresetFloater\_ms bitmap files:

```
\3ds3.1\Cstudio\ui\Cstudio_16a.bmp, Cstudio_16i.bmp, Cstudio_24a.bmp, and
Cstudio_24i.bmp.
```

---

## Design Details

Custom Keys operate similar to channel buttons on a car radio. Anytime you “dial in” a set of parameters for a particular Biped key using the standard Biped “Key Info” rollout, you can store those values in one of the 6 custom key buttons available, by clicking on the associated tool bar icon.

All of the standard Biped Key Info parameters are supported, including: ease to, ease from, tension, continuity, bias, IK Blend value, IK body/space setting, and pivot point information. This can be seen by reviewing the **struct Biped\_key** defined in `..\Scripts\Startup\CS3Customkeys.ms`.

Custom Keys do not store transformation information (like a pose). Only Key Info attributes that affect the interpolation of a Biped’s motion are stored. This allows stylistic settings to be easily transferred from frame to frame and even from Biped to Biped.

Each Custom Key is stored in a *structure* with the name **Biped\_key**. Here is the definition of **Biped\_key**:

```
struct Biped_key (type, ikblend, ikspace, easefrom, easeto, tension,
continuity, bias, ikAnkleTension, balanceFactor, dynamicsBlend,
ballisticTension)
```

Here is the initial Custom Key definition for the buffer “Preset Key A”. Note that it is *global* and therefore visible in all running MAXScript code and will hold its value until 3D Studio MAX is exited.

```
Global keya_buffer = Biped_key type:#body ikblend:0.1 ikspace:1
easefrom:0.0 easeto:0.0 tension:25.0 continuity:0.0 bias:25.0
ikAnkleTension:.8 balanceFactor:1 dynamicsBlend:1 ballisticTension:0.5
```

The **CS3Tools.cui** has 13 icon buttons. The first 6 are “Set custom Biped Key \*”. The next 6 are “Change Preset \* to current key”. The last one is “Launch Preset Floater”.

Pressing the 7th icon, “Change Preset A to current key”, will change the value of the specified Preset A Key’s **keya\_buffer** to the value of the currently selected key. This will trigger the execution of the code:

```
macros.run "Character Studio Tool" "Change_Preset_Key_A"
```

The line above will be displayed in the *MacroRecorder* if the MacroRecorder is enabled.

This macro, “**Change\_Preset\_Key\_A**”, is defined in the file \3ds3.1\ui\macroscripts\**Character Studio Tool-Change\_Preset\_Key\_A.mcr** and contains:

```
macroScript Change_Preset_Key_A
category:"Character Studio Tool"
toolTip:"Change Preset A to current key"
Icon:#"cstudio",7)
(
    keya_buffer = CS3Change_Key_FN keya_buffer
    CS3Save_presets()
)
```

See the *MAXScript Online Reference* help page *Macro Scripts* in the “Interacting with the 3DS MAX User Interface” topic and *Defining Macro Scripts* in the “Creating MAXScript Tools” topic for additional details on creating and using Macro Scripts.

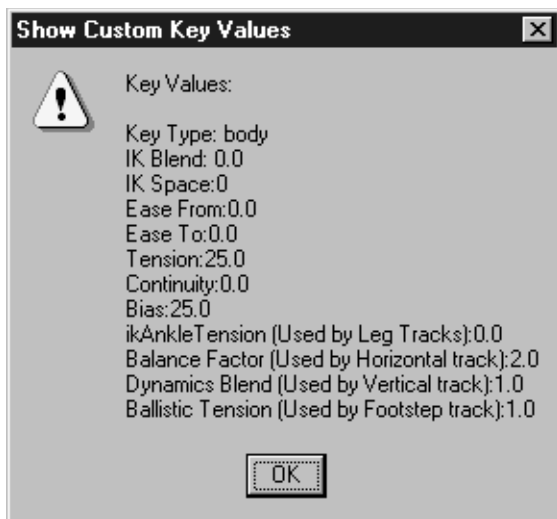
Select the last icon, “Launch Preset Floater”, in the **CS3Tools.cui**. The Preset Floater dialog is launched and displayed. There are 6 “Change \*” buttons and 6 “Show \*” buttons. The 6 “Change \*” buttons on the floating rollout perform the same actions as the 6 icon buttons on the CS Custom Key toolbar.



Pressing the “Change A” button executes the following code from the file  
..\Scripts\Startup\CS3CustomKeys.ms

```
on ChangeA pressed do
(
    keya_buffer = CS3Change_Key_FN keya_buffer
    CS3Save_presets()
)
```

Pressing the “Show A” button will display the current contents of the **keya\_buffer** as seen here:



Note that in both cases of Changing Preset A, the same function “**CS3Change\_Key\_FN**” was called. Additionally, notice that **CS3Save\_presets()** is called anytime that a preset is changed. **CS3Save\_presets()** will save all key values to a file called **CS3CustomKeys-presets.ini** that can be included in next launch of **CS3Customkeys.ms**. This is accomplished by **Save\_presets** formatting the presets as valid MAXScript code and using the *include statement* to include the code in **CS3Customkeys.ms**. The included code redefines the global `key_buffers` already defined. The file `\3ds3.1\Scripts\Startup\CS3Customkeys.ms Tutorial_File_name_CS3Customkeys_ms` contains the following functions:

```
fn Set_One_Key_FN
fn Set_Key_FN
fn CS3Save_Presets
fn CS3Show_key_FN
fn CS3Change_Key_FN
fn ControllerOf
fn RootOf
```

The functions `Change_Key_FN`, `Save_Presets` and `Set_Key_FN` will be reviewed in more detail below.

As seen in the above examples of “ChangeA”, the `Change_Key_FN` function is called with a single parameter, the particular `local_key_buffer`, and in the above examples with the **keya\_buffer**. The `local_key_buffer` will either be returned unmodified or it will be returned modified.

The values for `easeto`, `easeFrom`, `tension`, `continuity`, `bias`, and `ikAnkleTension` are stored for all valid keytypes while `ikblend` and `ikspace` is stored for only **#body** keys. For **#vertical** keys, `dynamicsblend` and `ballistictension` are also stored. For **#horizontal**, `balancefactor` is also stored.

See *BipedKey:MAXObject* (see page 637) for more details.

## The Change\_Key\_FN Code

The pseudo code for the **Change\_Key\_FN** is as follows:

```
function Change_Key_FN local_key_buffer
  if nothing selected return local_key_buffer
  if more than one selected return local_key_buffer
  if not of class Biped_Object return local_key_buffer
  get root node of the biped and controller for this body part
  If body part is COM and trackselection not vertical, horizontal or turning
  then
    print skipped and return local_key_buffer
  else
    save controller selection
  if in figuremode return local_key_buffer
  if in motion mode return local_key_buffer
  if in footstepmode return local_key_buffer
  if save controller selection not undefined then
    set my_index to getkeyindex my_controller slidertime
    if a key was found then
      set my_key to biped.getKey my_controller my_index
      if (my_key.type equals #body) then
        (
          set local_key_buffer.ikblend to my_key.ikblend
          set local_key_buffer.ikspace to my_key.ikspace
        set local_key_buffer.ikAnkleTensio to my_key.ikAnkleTension
        )
      if (my_key.type equals #vertical) then
        (
          set local_key_buffer.dynamicsblend to my_key.dynamicsblend
          set local_key_buffer.ballistictension to my_key.ballistictension
        )
      if (my_key.type equals #horizontal) then
        (
          set local_key_buffer.balancefactor to my_key.balancefactor
        )
      set local_key_buffer.easefrom to my_key.easeto
      set local_key_buffer.easeto to my_key.easefrom
      set local_key_buffer.tension to my_key.tension
      set local_key_buffer.continuity to my_key.continuity
      set local_key_buffer.bias to my_key.bias
      set return modified local_key_buffer
    else
      return local_key_buffer
  else
    return local_key_buffer
```

The actual function can be seen here:

```
FN CS3Change_Key_FN local_key_buffer =
(
```

```

    if ($selection.count == 0 ) then
    (
        messagebox "No object selected.\nPlease select a Biped Object first.\n"
        return local_key_buffer
    )

    if ($selection.count > 1) then
    (
        messagebox "There are multiple objects selected.\nPlease select a single
Biped Object first."
        return local_key_buffer
    )
    .....          if ((classof $) != Biped_Object) then
    (
        messagebox "The selected object is not a Biped Object.\n Please select a
Biped Object first\n."
        return local_key_buffer
    )

    my_root = $.controller.rootnode
    my_controller = $.controller

    if (my_root == $) then -- user is working on COM
    (
        if (my_controller.trackselection (see page 610) == 2) then
        (
            my_controller = my_controller.vertical.controller (see page 610)
        )

        else if (my_controller.trackselection (see page 610) ==1) then
        (
            my_controller = my_controller.horizontal.controller (see page 610)
        )

        else if (my_controller.trackselection (see page 610) == 3) then
        (
            my_controller = my_controller.turning.controller (see page 610)
        )

        else
        (
            msgtext = "" + $.name + "" skipped.\n"
            messagebox msgtext
            return local_key_buffer
        )
    )

    if (my_root.transform.controller.figuremode (see page 610) == true) then
    (
        messagebox "Change Preset Key is not supported in Figure Mode.\n Please
exit Figure Mode first.\n"

```



```

        return local_key_buffer
    )

    if (my_root.transform.controller.motionmode (see page 610) == true) then
    (
        messagebox "Change Preset Key is not supported in Motion Flow Mode.\n
Please exit Motion Flow Mode first.\n"
        return local_key_buffer
    )

    if (my_root.transform.controller.footstepmode (see page 610) == true) then
    (
        messagebox "Change Preset Key is not supported in Footstep Mode.\n Please
exit Footstep Mode first.\n"
        return local_key_buffer
    )

    if (my_controller != undefined) then
    (
--        format "Controller: %\n" my_controller

        my_index = getkeyindex my_controller slidertime
--        format "Key Index = %\n" my_index
        if (my_index>0) then
        (
            my_key = biped.getKey (see page 637) my_controller my_index

--            format "key type is % \n" my_key.type

            if (my_key.type == #body (see page 637)) then
            (
                local_key_buffer.ikblend = my_key.ikblend (see page 637)
                local_key_buffer.ikspace = my_key.ikspace (see page 637)
                local_key_buffer.ikAnkleTension = my_key.ikAnkleTension (see page 637)
            )
            if (my_key.type == #vertical (see page 637)) then
            (
                local_key_buffer.dynamicsblend = my_key.dynamicsblend (see page 637)
                local_key_buffer.ballistictension = my_key.ballistictension (see page 637)
            )

            if (my_key.type == #horizontal (see page 637)) then
            (
                local_key_buffer.balancefactor = my_key.balancefactor (see page 637)
            )
            local_key_buffer.easefrom = my_key.easeto (see page 637)
            local_key_buffer.easeto = my_key.easefrom (see page 637)
            local_key_buffer.tension = my_key.tension (see page 637)
            local_key_buffer.continuity = my_key.continuity (see page 637)
            local_key_buffer.bias = my_key.bias (see page 637)

```

```

        return (local_key_buffer)
    )

    else
    (
        messagebox "Cannot change this preset because there is no key for
the selected track at the current frame.\n Please advance the time slider to a
frame where a key exists.\nOr select a track and frame where a key exists.\nOr set
a key first.\n"
        return (local_key_buffer)
    )
)
else
(
    messagebox "There is no controller for the selected track.\nYou may need
to enable separate tracks.\n"
    return (local_key_buffer)
)
)

```

---

## CS3Save\_Presets Code

```

-- *****
-- CS3Save_Presets - save all key values to a .ms file that can be re-read at next
launch
-- *****
FN CS3Save_Presets =
(
    my_path = scriptspath + "startup\CS3CustomKeys-presets.ini"
    outfile = createfile my_path
    format "-- This file was created by the CS3 Custom Key Macro
\"Save_Key_Presets\" to store user-customized presets for Biped custom key macro
tools\n" to:outfile
    format "-- These values can be edited by hand for existing fields\n\n"
to:outfile

    k = keya_buffer
    format "Global keya_buffer = Biped_key type:% ikblend:% ikspace:% easefrom:%
easeto:% tension:% continuity:% bias:%\n" k.type k.ikblend k.ikspace k.easefrom
k.easeto k.tension k.continuity k.bias to:outfile
    k = keyb_buffer
    format "Global keyb_buffer = Biped_key type:% ikblend:% ikspace:% easefrom:%
easeto:% tension:% continuity:% bias:%\n" k.type k.ikblend k.ikspace k.easefrom
k.easeto k.tension k.continuity k.bias to:outfile
    k = keyc_buffer
    format "Global keyc_buffer = Biped_key type:% ikblend:% ikspace:% easefrom:%
easeto:% tension:% continuity:% bias:%\n" k.type k.ikblend k.ikspace k.easefrom
k.easeto k.tension k.continuity k.bias to:outfile

```

```

    k = keyd_buffer
    format "Global keyd_buffer = Biped_key type:% ikblend:% ikspace:% easefrom:%
easeto:% tension:% continuity:% bias:%\n" k.type k.ikblend k.ikspace k.easefrom
k.easeto k.tension k.continuity k.bias to:outfile
    k = keye_buffer
    format "Global keye_buffer = Biped_key type:% ikblend:% ikspace:% easefrom:%
easeto:% tension:% continuity:% bias:%\n" k.type k.ikblend k.ikspace k.easefrom
k.easeto k.tension k.continuity k.bias to:outfile
    k = keyf_buffer
    format "Global keyf_buffer = Biped_key type:% ikblend:% ikspace:% easefrom:%
easeto:% tension:% continuity:% bias:%\n" k.type k.ikblend k.ikspace k.easefrom
k.easeto k.tension k.continuity k.bias to:outfile

    close outfile
)

```

---

## Set\_Key\_FN Code

The pseudo code for the **Set\_Key\_FN** is as follows:

function **Set\_Key\_FN** local\_key\_buffer

```

    store state of shift key
    if nothing selected then return false
    if one object selected and not a Biped Object then return false
    if local_key_buffer undefined then return false
    -- process multiple selections independently so that we can
    -- support key setting of multiple bipeds
    else for each of the items in the selection
        ( if current item not a Biped Object then get next item
          set my_root to the root of the hierarchy
          if my_root is in figuremode then get the next item
          if my_root is in motionmode then get the next item
          if my_root is in footstepmode then get the next item
          set my_controller to the item's controller
          if hierarchy root is the current item then
              set my_controller to the current track ( horizontal / vertical /
turning)
          else get the next item

        if shift key was pressed then
            (
                for each key in my_controller
                    if key is selected set values for key
            )
        else
            (
                add a new key to my_controller at the current time and select the new key
                get the index of the newly created key
                if the index is greater than 0 then

```

```

get the newly created key
    if (my_key.type == #body) then
    (
        set my_key.ikblend to local_key_buffer.ikblend
        set my_key.ikspace to local_key_buffer.ikspace
    )
    set my_key.easeto to local_key_buffer.easefrom
    set my_key.easefrom to local_key_buffer.easeto
    set my_key.tension to local_key_buffer.tension
    set my_key.continuity to local_key_buffer.continuity
    set my_key.bias to local_key_buffer.bias
)

```

The actual function can be seen here:

```

FN Set_Key_FN local_key_buffer =
(
    if (keyboard.shiftpressed) then
    keyboardshiftpressed_atstart = true
    else
    keyboardshiftpressed_atstart = false
        if ($selection .count == 0 ) then
        (
            messagebox "There are no objects selected.\nPlease select a Biped Object
first.\n"
            return false
        )

        -- Showclass "Biped_" reports Biped_Object:GeometryClass{9125,0}
        if (($selection .count == 1) and ((classof $) != Biped_Object)) then
        (
            messagebox "The selected object is not a Biped Object.\nPlease select a
Biped Object first\n."
            return false
        )

        if (local_key_buffer == undefined) then
        (
            messagebox "This preset key has not been initialized.\nPlease select a
key first, and set its preset\n."
            return false
        )

        --      process multiple selections independently so that we can support key setting
of multiple bipeds

        else for i=1 to $selection .count do
        (
            if ((classof $selection[i]) != Biped_Object) then
            (
                msgtext = "The selected object `` + $selection[i].name + `` is not
a Biped Object.\nObject Skipped.\n"

```

```

        messagebox msgtext
        continue
    )

    my_root = $selection[i].controller.rootnode
    if (my_root.transform.controller.figuremode (see page 610) == true) then
    (
        msgtext = "" + $selection[i].name + "` is in Figure Mode.\nSet Key
is not supported in Figure Mode.\nPlease exit Figure Mode for this object
first.\nObject Skipped.\n"
        messagebox msgtext
        continue
    )

    if (my_root.transform.controller.motionmode (see page 610) == true) then
    (
        msgtext = "" + $selection[i].name + "` is in Motion Flow Mode.\nSet
Key is not supported in Motion Flow Mode.\nPlease exit Motion Flow Mode for this
object first.\nObject Skipped.\n"
        messagebox msgtext
        continue
    )

    if (my_root.transform.controller.footstepmode (see page 610) == true) then
    (
        msgtext = "" + $selection[i].name + "` is in Footstep Mode.\nSet
Key is not supported in Footstep Mode.\nPlease exit Footstep Mode for this object
first.\nObject Skipped.\n"
        messagebox msgtext
        continue
    )

    -- find the root controller of this object
    -- my_controller = (controllerof ($selection[i]))

    my_controller = $selection[i].controller
    if (my_root == selection[i]) then -- user is working on COM
    (
        if (my_controller.trackselection (see page 610) == 1) then
        (
            my_controller = my_controller.horizontal.controller (see page 610)
        )

        else if (my_controller.trackselection (see page 610) == 2) then
        (
            my_controller = my_controller.vertical.controller (see page 610)
        )

        else if (my_controller.trackselection (see page 610) == 3) then
        (
            my_controller = my_controller.turning.controller (see page 610)

```

```

    )

    else
    (
        msgtext = "" + $selection[i].name + "` skipped.\n"
        messagebox msgtext
        continue
    )
)

-- loop through all keys and call only if a key is selected

if (keyboardshiftpressed_atstart == true) then -- user wants to process
all selected keys in this track
(
    found_selected_keys = 0

    for temp_index=1 to (numkeys my_controller) do
    (
        if (iskeyselected my_controller temp_index) then
        (
            set_one_key_FN my_controller temp_index local_key_buffer -- set
this key to the local key buffer
            found_selected_keys += 1
        )
    )

    if (found_selected_keys == 0 ) then
    (
        msgtext = "Warning: No keys selected.\nNo changes made to " +
$selection[i].name + "."
        messagebox (msgtext)
    )
    else
    (
        msgtext = "Changed " + (found_selected_keys as string) + " keys
in " + $selection[i].name + "."
        messagebox (msgtext)
    )
)

    else -- (default) user wants to process just the current key
    (
        temp_index = getkeyindex my_controller slidertime
        -- format "index of key at this frame (temp_index) is %\n" temp_index

        if (temp_index == 0) then -- no key at this frame yet, let's create
one...
        (
            biped.addNewKey my_controller slidertime #select

```

```

        temp_index = getkeyindex my_controller slidertime
    )

    -- temp_index should now indicate the ordinal index of the key for
this track, 1st, 2nd, 3rd, etc.
    -- if it is still zero, then the user has selected a track or mode
that prevents adding a key, like footsteps

    if (temp_index >0) then -- key was created ok
    (
        set_one_key_FN my_controller temp_index local_key_buffer
    )
    else -- addnewkey failed, probably because the footstep track is
selected
    (
        msgtext = "The selected track '" + ($selection[i]).name + "' does
not support keys.\n Object Skipped.\n"
        messagebox msgtext
    )
    )
)

return true
)

```

---

## Naming the Buttons

You can name the key button accordingly to help you remember the purpose for that key type: “heel down,” “finger point,” “hand grab,” etc.

To change the tooltip for the buttons on the CS3Tools.cui toolbar, right click on the button, choose “Edit Button Appearance,” modify the text in the “Tooltip:” text box and choose the OK button.

---

## Extending the Number of Presets

The number of different settings that CS3Tools.cui supports can be extended. Two new macro files (\*.mcr) for each new additional custom key, one for “Set” and one for “Change Preset,” need to be added to the ..\ui\macroscripts directory and identical macros need to be added to ..\Scripts\Startup\CS3Customkeys.ms as well. Global key\*\_buffer’s that matched the number of pairs of new macros added to the CS3Tools.cui need to be inserted into the file ..\Scripts\Startup\CS3Customkeys.ms Tutorial\_File\_name\_CS3Customkeys.ms. Finally, the new macro files (\*.mcr) have to be incorporated into the CS3Tools.cui. See the *MAXScript Online Reference* help page *Macro Scripts* in the “Interacting with the 3DS MAX User Interface” topic and *Defining Macro Scripts* in the “Creating MAXScript Tools” topic for additional details.





---

# Glossary

**Active/Inactive Footsteps.** When footsteps are first created in Footstep mode they are inactive. You must activate these footsteps using the Create Keys for Inactive Footsteps control. Active footsteps have keys to animate the biped. Inactive footsteps have not been given keys by **character studio**.

**Adapt Locks.** To avoid automatic adaptation when making Footstep Space edits, you can use Adapt Locks to lock specified tracks so that Biped doesn't adapt them when you edit footsteps. Adapt Locks only applies to a Footstep animation not a freeform animation.

**Adaptation.** One of the most powerful features in Biped is the ability to adapt keyframes to edits you might want to make in your footstep pattern. By analogy, the footsteps become a kind of "gizmo" for manipulating the keyframes of your character's animation. In most cases, edits you make to footsteps will act upon your keys in an intuitive fashion.

**Adjust Talent Pose.** After loading a marker file, use Adjust Talent Pose to correct the biped position relative to the motion capture markers. Align the biped limbs to the markers then click Adjust Talent Pose to compute this offset for all the loaded marker data.

**Airborne Period.** A "ballistic gait" is defined to be any footstep pattern in which there are periods with no feet on the ground, causing the biped to become airborne, or ballistic. For example, running, hopping and jumping are ballistic gaits with airborne periods.

**Animation Layers.** Layers allow you to add layers of animation above the original biped

animation. This is a powerful way of making global changes to your character animation. For example, simply add a layer and rotate the spine forward at any frame, and a run cycle becomes a crouched run. The original biped motion is kept intact and can be viewed by switching back to the original layer.

Layers can be viewed individually or as a composite of all the animation in all the layers. Layers behave like a freeform animation; the biped can adopt any position.

Layers will allow you to easily adjust raw motion capture data, which contains keys at every frame. Simply add a layer and keyframe the biped.

**Apply Increment.** Apply Increment is a powerful tool: it allows you to increment a set of keys in a biped track. For instance, if you have an animation of a running biped and you want the biped's legs to kick higher, you can select all of the biped's moving leg keys and increment them slightly so that all the kicks are higher.

**Attachments (IK).** The biped hands and feet can be linked to the world, another object in the scene, or to the bipeds body. This linking is also called IK attachments. Attachments can be blended, this lets you start with one attachment and end with another. If the biped is catching a ball you can start with the hand in body space and end with the hand in the coordinate space of the ball.

**Auto Grid.** AutoGrid is an option on the Create panel that allows you to create an object (biped) on the surface of another object.

**Avoid Behavior.** The Avoid behavior lets you specify any object or objects that delegates must keep away from. As delegates approach designated objects during the crowd simulation,

they steer clear of them while turning and/or braking as necessary. This behavior uses three different methods to let delegates avoid each other and other objects: Steer To Avoid (the preferred method), Repel, and Vector Field.

**Balance Factor.** Balance Factor positions the biped's weight anywhere along a line that extends from the center of mass to the biped's head.

**Ballistic Gait.** A "ballistic gait" is defined to be any footstep pattern in which there are periods with no feet on the ground.

**Ballistic Tension.** Controls the amount of spring or tension when the biped lands or takes off from a jump or run step.

**Behaviors.** Simulate a range of crowd activities. Seek, avoid, path follow, surface follow, repel, orientation, scripted, space warp, surface arrive, wall repel, wall seek, and wander are all behaviors available in a crowd simulation.

Behaviors let you assign procedural activity types to delegates and objects linked to delegates. You can associate any number of behaviors with each Crowd object, and then link delegates and teams of delegates to each behavior. A specific behavior assigned to a Crowd object belongs only to that crowd; it cannot be assigned to any other crowds.

**Bend Links.** Bend Links mode in the General rollout lets you bend all the biped spine objects naturally by rotating a biped spine object. Bend Links also works for the biped tail and ponytail links.

**Biovision.** Biovision files contain the "actor's" skeletal and limb/joint rotation data. These files have a *.bvh* extension.

**BIP Files.** The native motion file format of **character studio**. BIP files contain skeletal size and limb rotation data.

**Biped.** The armature used to pose a character.

**Biped Dynamics.** Biped Dynamics calculates biped airborne trajectory, knee bend on landing and positions the biped to maintain balance when the spine is rotated. When parameters change, the biped adapts.

**Biped Playback.** Biped Playback in the General rollout plays the animation for all bipeds unless they are excluded on the Display Preferences dialog. This playback mode usually gives real-time playback, which you may not get if you use Play on the 3DS MAX animation toolbar.

In Biped Playback mode, the biped is displayed as bones only, with no other scene objects visible.

**Body Space.** A biped limb can be put into the coordinate space of the world or an object in the scene as well as body space. Body space moves the biped limbs when the biped moves—if you rotate the bipeds hips the feet, in body space, move also.

**Bulge.** Physique allows you to “bulge” a mesh based on the orientation of a limb. Bulging the mesh is used to simulate muscle contraction.

**Bulge Angle.** A bulge angle is a control in Physique that sets the limb angle where the bulge will occur. Generally you orient the limb first and then set the bulge angle. After setting the bulge angle you then deform the mesh to bulge.

**BVH Files.** *See Biovision (see page 722).*

**Center of Mass (COM).** The root node of the biped. Transforming this moves the entire biped.

This node is used as the center of mass and can move outside of the biped body. The center of mass uses three animation tracks to animate the biped. Two of these tracks, Body Vertical and Body Horizontal, contain Biped Dynamics parameters.

**character studio Marker Files.** The CSM file format is an ASCII file format that uses positional markers rather than limb rotation data. When a raw marker file is imported, only marker position data is buffered in the motion capture buffer. The software uses the marker data to extract limb rotation data to position the biped.

**Clip Controller.** The GlobalMotionClip and MasterMotionClip controllers are used to create animation for multiple objects. Birds, butterflies, schools of fish and bugs can be animated using these tools. Clip controllers can be created either as block controllers in Track View or more directly in the Crowd helper controls on the Global Clip Controllers rollout. Clip controllers should be used to animate non-biped creatures.

**Cognitive Controller.** The Cognitive Controller editor lets you sequence different behaviors using state diagrams, where conditionals written in MAXScript impose changes in behavior. For example, you can specify that a character or object is to wander aimlessly until it comes within a certain distance of another object, whereupon it heads straight for that object. Or you can specify that one character is to avoid another only when the second character is avoiding the first.

**Control Point.** A vertex used to control cross sections in envelopes, bulges and tendons.

**Controller.** Software that controls animation. Controllers handle the following functions.

- Storing animation key values
- Storing procedural animation settings
- Interpolating between animation key values

**Coordinate Space.** In **character studio**, the three most used coordinate spaces are world, object and body space. These are often used to control the biped hands and feet. Another coordinate system are the footstep gizmos themselves—a foot on a footstep is in the coordinate space of the footstep. If the footstep is moved the foot moves also. Bear in mind that a sliding footstep is a footstep that moves relative to the coordinate system of the corresponding footstep gizmo.

**Cross Section.** Envelope cross sections can be moved and scaled to encompass more or less of an object (character). In Sub Object Bulge, cross sections handle the amount of mesh bulge.

**Crowd Helper Object.** The Crowd helper object, available in Create panel > Helpers, serves as the command center for setting up and solving crowd simulations. The Crowd helper object also lets you add behaviors to the scene choose the current behavior from a list, and provides a rollout for modifying that behavior.

**Crowd System.** The crowd system comprises the Crowd helper, Delegate helper, the Vector Field space warp, and Motion Flow mode. These are used in combination to animate characters or other objects.

**CSM Files.** See *character studio Marker Files* (see page 723).

**Deformable Envelope.** Deformable envelopes follow the Physique deformation spline that runs through the joints in the hierarchy and can

be deformed using bulge angles, tendons, and link parameters.

Typically you use deformable envelopes; however, game developers with game engine restrictions may want to use rigid envelopes exclusively. Both rigid and deformable envelopes can be turned on for the same link.

**Deformation.** The effect caused by Physique on a mesh. Envelopes, bulges, link parameters and tendons all affect how a mesh deforms.

**Delegate Helper Objects.** The Delegate helper object is a special object used in crowd animation. It serves as an agent for motion created by a Crowd object and its behaviors. The Crowd object controls a delegate or delegates, whose motion can then be imparted to a biped or other object. Delegates cannot be rendered. The delegate object takes the shape of a pyramid. By default, the point of the pyramid indicates the forward direction.

**Delegates.** See *Delegate Helper Objects*. (see page 724)

**Double Support Period.** A period where both the biped feet are on the ground.

**Dynamics.** See *Biped Dynamics* (see page 723).

**Dynamics Blend.** Blends between biped and spline dynamics. Select the Body Vertical track (center of mass vertical track) and control the amount of gravity in an airborne period, as in a running or jumping motion. Dynamics Blend has no effect on a walking motion where footsteps overlap.

**Editable Meshes.** Physique will work on any point-based objects including geometric primitives, editable meshes, patch-based objects, NURBS, and even FFD space warps. For NURBS

and FFDs, physique deforms the control points, which, in turn deform the model.

**Envelopes.** Envelopes are Physique's primary tool to control skin deformation. Envelopes define an area of influence about a single link in the hierarchy and can be set to overlap adjacent links. Vertices that fall in the overlap area of the envelopes are weighted to produce smooth blending at joint intersections. Each envelope comprises a pair of inner and outer bounds, each with four cross sections.

**FFD.** FFD stands for Free Form Deformation. Its effect is used in computer animation for things like dancing cars and gas tanks. You can use it as well for modeling rounded shapes such as chairs and sculpture.

The FFD modifier surrounds the selected geometry with a lattice box. By adjusting the control points of the lattice, you deform the enclosed geometry. For NURBS and FFDs, physique deforms the control points, which, in turn deform the model.

**Figure Mode.** Use Figure mode to fit a biped to the mesh or mesh objects representing your character. Leave Figure Mode on when you attach the mesh to the biped with Physique. Figure mode is also used to scale a biped with a mesh attached, to make biped "fit" adjustments after Physique is applied, and to correct posture in motion files that need a global posture change.

When Figure Mode is turned on, the biped jumps from its animated position to its Figure Mode pose. Animation is preserved when you exit Figure mode.

All of the parameters on the Structure rollout are active in Figure mode.

**Filtering.** Motion capture and marker data typically have keys at every frame. Filtering motion capture data reduces keys, making the job of altering or personalizing the motion data simpler. Other filtering options include footstep extraction, applying the skeletal structure stored in the motion capture file to the biped, looping the data, importing a portion of the motion capture file and selecting tracks to load.

**Foot States.** The biped feet can be in one of four states: touch, plant, lift, and move. The general rollout displays foot states.

- **Plant.** The biped foot state in full contact with the footstep.
- **Lift.** The biped foot state just before leaving a footstep.
- **Move.** the biped foot state between footsteps; an airborne period.
- **Touch.** The biped foot state at which a biped foot first contacts a footstep.

**Footstep Animation.** Biped's patented footstep-driven keyframe animation feature allows animators to use footsteps to create broad, global brush strokes for character movement. Once footsteps are in place, key frames are generated automatically to produce an initial sketch of the 3D character's motion. Throughout edits and revisions, the original nuances of the character are preserved; Biped remembers everything about how a character moves, and it makes all of the appropriate adjustments if the footsteps are changed.

**Footsteps Method.** In the viewports, footsteps represent support periods in space for the biped feet. Moving or rotating footsteps in space is done in the viewports. In Track View, each footstep appears as a block that represents a support period in time for each of the biped's feet. Moving footsteps in time is done in Track

**View.** The footstep position and orientation in the viewport controls where the biped will step. There are three ways to create footsteps for the biped. The first way is to place footsteps individually, one at a time. The second way is to invoke Biped's multiple footstep creation tools to create a walk, run, or jump animation. The third way is to extract footsteps from raw motion capture data.

A key advantage of the footstep method is the natural adaptation of the biped that occurs when the footsteps are edited in time and space. Editing footsteps in the viewports will allow you to reposition all of the footsteps to move the entire animation.

**Forward Kinematics.** Using an arm to move a hand is an example of forward kinematics. Using the hand to move the arm is an example of inverse kinematics. The IK Blend parameter in the IK Key Info rollout determines how forward kinematics and inverse kinematics are blended to interpolate an intermediate position.

**Freeform Animation.** **character studio** gives you the option to animate character poses with and without the aid of footsteps. Freeform animation does not use footsteps. You set all the keys in a freeform animation.

**Freeform Method.** In Freeform mode (without footsteps), you can pose every joint of your character exactly as you like using traditional keyframe methods. You can even blend dynamically between forward kinematics and inverse kinematics to introduce higher-level control in just the cases you need it to motivate your character's motion.

**Gait Pattern.** The pattern created by a gait: walk, run or jump. When you create new footsteps, the timing for the footsteps is

determined by the gait you have chosen--walk, run, or jump--and the parameters for that gait. Gait parameters are in the Footstep Creation rollout in the Motion panel.

**Gait Type.** There are three types of gaits; walk, run or jump.

In a walk, at least one foot is always in contact with the ground. The periods where one or both feet are in contact with the ground are known as support periods.

In running, there is a period between each support period in which the body is airborne.

Jumping is a special case of running. Both feet are in contact with the ground at the same time, or airborne at the same time.

**Geometric Primitives.** Simple primitive objects such as, spheres, boxes, cylinders and so on.

**Gizmo.** A footstep is actually a gizmo to edit the bipeds feet. Gizmos are used by Physique to visually identify bulge angles. The transformation gizmo appears in the viewports to move or rotate objects.

**Global Motion Clip Controller.** A controller that contains the animation necessary to animate a non-bipedal crowd of objects. It consists of a list of motion clips and the logic needed to instance and blend these motion clips for a crowd animation. The Global Motion Clip Controller is accessed via the Crowd Helper.

**GravAccel.** GravAccel (for Gravitational Acceleration) Lets you scale the height of the airborne periods. The greater this value, the greater the height. If the biped appears to be going too high, reduce this value; if the biped goes too low, increase it. Each biped has its own Gravitational Acceleration value. Default=Based on the height of the biped.

If feet are the active 3D Studio MAX units and the biped is about 5 feet 10 inches tall, then Gravitational Acceleration equals about 32, for 32 ft. per second per second. For other biped heights, Biped scales this value to naturalistically fit the scene; the Gravitational Acceleration value also changes to agree with other unit systems, such as metric.

**Gravity.** See *GravAccel* (see page 726).

**Helper Object.** Helpers are used to help you set up an animation but do not render. In **character studio** you have two helper objects; the crowd and delegate helpers.

**IK Blend.** Determines how forward kinematics and inverse kinematics are blended to interpolate an intermediate position. Using an arm to move a hand is an example of forward kinematics. Using the hand to move the arm is an example of inverse kinematics.

**In Place Mode.** Use In Place Mode to keep the biped visible in the viewports while the animation plays. Use this for biped key editing or adjusting envelopes with Physique. Prevents XY movement of the biped center of mass during animation playback; motion along the Z-axis is preserved. This is a three-button fly-out. In Place mode is stored with the 3DS MAX file.

**Initial Pose.** The original position of the mesh relative to the biped just before applying physique. In Physique the Initial Skeleton Pose puts the mesh into its original pose as when Physique was first applied.

**Initialize.** In Physique you initialize the mesh to attach it to the biped. This creates envelopes around the limbs to control the mesh.

**Instance.** An instance is a completely interchangeable clone of the original. Modifying

an instanced object is the same as modifying the original.

Instances are not only alike in geometry, but also share modifiers and materials. When you change one instance by applying a modifier, for example, all the other instances change with it.

Each instance has its own set of transforms, object properties and space warp bindings—these are not shared among instances.

Within the program, instances derive from the same master object. What you're doing is applying a single modifier to a single master object. In the viewport, what you see as multiple objects are multiple instances of the same definition.

If you wanted to create a school of swimming fish, you might begin by making many instanced copies of a single fish. You could then animate the swimming motion by applying a ripple modifier to any fish in the school. The whole school would swim with exactly the same motions.

**Interpolation.** Interpolation is the calculation of intermediate values.

**Inverse Kinematics.** Moving a hand to position the arm is an example of inverse kinematics. Biped's inverse kinematics has three parameters that you can vary during the limb's motion by setting them at each key of the arm and leg keyframe tracks. As the limb moves through each key:

- **IK Blend** sets the motion interpolation to be a blend of forward and inverse kinematics. This will allow you to blend swinging motions with hand/foot directed motions. Using an arm to move a hand is an example of forward kinematics. Using the hand to move the arm is an example of inverse

kinematics. The default is 0.0, or full forward kinematics.

- **Body/Object** determines the reference coordinate space of the IK path. This will allow you to move the IK path with your character's body or temporarily attach the hands/feet to follow other objects or be attached to world space. The default is Body.
- **Join to Previous IK Key** determines if the key should be part of the previous key (same reference position as previous key).

**Keyframe Animation.** Keyframes record the beginning and end of each transformation of an object or element in the scene. The values at these keyframes are called keys.

For example, if you have a box that has not been animated, no keyframes (or keys) exist for it. If you turn on the Animate button, move to frame 20, and rotate the box 90 degrees, Rotate keys are created at frames 0 and 20.

The key at frame 0 represents the orientation of the box before it was rotated, while the key at frame 20 represents the orientation of the box after it was rotated 90 degrees. When you play the animation, the box rotates from 0 to 90 degrees over 20 frames.

**Layers.** Layers allow you to add layers of animation above the original biped animation. This is a powerful way of making global changes to your character animation. For example, simply add a layer and rotate the spine forward at any frame, and a run cycle becomes a crouched run. The original biped motion is kept intact and can be viewed by switching back to the original layer. Layers can be viewed individually or as a composite of all the animation in all the layers. Layers behave like a freeform animation; the biped can adopt any position.

Layers will allow you to easily adjust raw motion capture data, which contains keys at every frame. Simply add a layer and keyframe the biped.

**Lift.** The state of a foot at the frame it is about to lift from a footstep.

**Links.** Links form the hierarchical chain that makes up the biped and the segments in the Physique deformation spline. Link parameters in Physique allow you to bend, twist, change sliding behavior and radially scale the mesh on the selected link.

**Marker Data.** Data from a motion capture device. Rather than limb rotational data, marker data uses marker positions to specify limb position.

**Marker Files.** A file from a motion capture device. CSM is the native marker file format of **character studio**.

**Markers.** Reflective objects placed on talent in a motion capture session.

**Master Motion Clip Controller.** A controller similar to the Block Controller that consists of a list of motion clips. When instanced these motion clips may blend from one animation to the other. The Master Motion Clip Controller is accessed via the Crowd helper.

**Mirroring.** The Mirror control in the Keyframing rollout allows you to mirror the entire biped animation.

**Motion Blending.** Transitions are used to blend clips together in Motion Flow mode. By default, minimum motion loss is the algorithm used to calculate a transition. If optimized transitions are used then a sophisticated algorithm that



minimizes foot sliding is used—this algorithm is computationally expensive.

**Motion Capture.** Digitizing a character's (talent) movements. This requires a motion capture device.

**Motion Clip.** A clip of animation specifying a specific motion. One animation can yield all the motion clips necessary to animate a crowd. A bird animation may have a flap, glide and land motion clips for example. Motion clips are used by the Global Motion Clip Controller and the Master Motion Clip Controller to animate non bipedal objects.

**Motion Files.** *character studio* can load a variety of motion files, *.csm*, *.bip*, *.bvh*.

**Motion Flow.** See *Motion Flow mode*. (see page 729)

**Motion Flow Editor.** In Motion Flow mode the Motion Flow Editor allows you to manually create a transition between two clips. You set the start frame and transition duration for both clips and the orientation of the destination clip.

**Motion Flow Mode.** In Motion Flow mode, you combine *.bip* files, using either velocity-interpolated transitions or an algorithm to minimize sliding feet to create longer character animation. You also use Motion Flow mode along with the crowd system to automatically generate crowd behavior.

**Motion Flow Scripts.** Scripts are created either manually or automatically. A script is a sequence of motion files that are played to create a character's motion.

**Motion Synthesis.** Motion Synthesis is the process of combining motions (clips) automatically, using the Crowd system, for

animating a biped(s). Clips are added to the Motion Flow Graph, transitions are created between appropriate clips. In the crowd system delegates are animated. During synthesis (solving the motion) the delegates speed and direction are analyzed by the software. Based on the analysis clips are selected to animate the biped(s).

**N Links.** In *Physique* any number of overlapping envelopes (N Links) can influence vertices. Normally, N Links is preferred. For special purposes such as game, for example, you can limit the number of links (envelopes) that can affect a vertex.

**Object Instance.** An instance is a completely interchangeable clone of the original. Modifying an instanced object is the same as modifying the original.

Instances are not only alike in geometry, but also share modifiers, materials and maps, and animation controllers. When you change one instance by applying a modifier, for example, all the other instances change with it.

Each instance has its own set of transforms, object properties and space warp bindings—these are not shared among instances.

Within the program, instances derive from the same master object. What you're doing is applying a single modifier to a single master object. In the viewport, what you see as multiple objects are multiple instances of the same definition.

**Object Space.** Place a biped limb into the space of another object or the world space. If the bipeds hands are in the space of a ball, then wherever the ball moves the hands moves with it. If the biped feet are in world space then, when you move the center of mass, the feet stay planted in the same location.

**Obstacle-Avoidance Behavior.** An important part of crowd behavior is avoidance of obstacles. Think of an obstacle as anything that impedes a crowd member's progress. Examples of obstacles include walls, telephone poles, and fences, as well as other crowd members. Encountering such objects can cause avoidance behavior, which consists of any combination of slowing down, turning, and stopping.

**Optical Markers.** Reflective markers used by certain motion capture equipment.

**Orientation Behavior.** The Orientation behavior lets you control whether and how delegates rotate, independent of their direction of motion. Normally, a delegate always faces in the direction it's moving. You can use the Orientation behavior to specify limits to the delegate's rotational activity without affecting its path, which is generated by other behaviors. Use these settings, for example, to keep delegates facing in one direction while moving in another. Note: These settings do not affect the path a delegate takes, which is produced by other behaviors such as Seek and Avoid. They influence only the direction it faces as it traverses the path.

**Patch-Based Objects.** Objects made from patches. Physique can work with meshes, patches, NURBS, splines and FFDs.

**Path Follow Behavior.** The Path Follow behavior lets you direct delegates to traverse a specified path during a crowd simulation. Delegates can move forward or backward along paths, and when they reach the end, they can loop back to the start or reverse direction, or even continue in the same general direction. If the delegate's start position isn't on the path at the start of the simulation, it moves to the path

before following the path. During the solution, **character studio** intermittently displays an optional *target* icon to show the delegate's immediate goal; this changes as the simulation proceeds.

**Period.** A freeform period is a period between footsteps where you can animate the biped any way you want—biped dynamics are suspended.

**Phases of Leg Motion.** A leg's motion has four phases, beginning with the foot on the ground. Then the foot lifts, moves through the air, and returns to the ground again. Biped divides this motion into four phases as follows:

- Touch occurs at the leg keyframe where the leg's foot first touches the ground and always corresponds with the start frame of a footstep in Track View.
- Plant occurs after touching, and before lifting. It is always in between the start and end frames of a footstep in Track View.
- Lift occurs at the keyframe where the leg's foot lifts off the ground and always corresponds with the end frame of each footstep in Track View.
- Move occurs while the foot is in the air and is always in the intervals in between steps in Track View. In walking, while one foot moves the body is supported by the other leg. In running or jumping, while a foot moves there is a period where the body is not supported, and moves in midair.

**PHY Files.** You can save Physique data to a Physique (PHY) file to save data common to all objects sharing a given Physique modifier. Later, you can reload the data file, either to restore the data that belongs to a particular skin or portion of skin, or to transfer the Physique of one skin (or portion) to a different one.

**Physique.** Physique is a modifier that when applied to a mesh, allows the movements of an underlying skeleton to seamlessly move the mesh like bones and muscle under a human skin. Physique will work on any point-based objects including geometric primitives, editable meshes, patch-based objects, NURBS, and even FFD space warps. For NURBS and FFDs, physique deforms the control points, which, in turn deform the model. It will attach to any skeleton structure including a biped, 3DS MAX bones, splines, or any 3DS MAX hierarchy.

**Plant.** The state of the biped foot when it is flat on a footstep.

**Poses.** The stance of a biped, you can cut and paste poses.

**Positional Markers.** Reflective objects placed on talent in a motion capture session.

**Prop Bone.** The CSM marker file format supports a prop bone in either or both hands. There are six additional markers for the top, bottom, and middle of the two props. If these tracks are detected, a 3DS MAX dummy object is created.

The length of the prop is the average distance between the top and bottom prop marker during animation. The prop will be oriented in the plane of the three prop markers, and its origin will be at the bottom prop marker.

**Reference.** References are like “one-way” instances. Referenced objects are based on the original object, as are instances, and can also have their own unique modifiers. Any modification made to the original object is passed on to its references, but any modification made to a reference is not passed back to the original.

The one-way effect is useful, since you can maintain an original that will affect all its references, while the references themselves can take on individual characteristics.

If you are modeling heads, for example, you might want to keep a family resemblance in your characters. You could model basic features on the original, then model specifics on each reference.

In the modifier stack, a line of dashes separates the reference from its parent object, so you can see that the effect of modifiers on the reference will not affect the parent object or other references to it.

**Reinitialize.** When you need to reset vertex, envelope and other skin parameters to a Physique mesh, click Reinitialize to display the Physique Initialization dialog. Using controls in this dialog, you select which category to update, and apply the new global settings you specify.

For example, if you’ve added a new bone to the hierarchy and want it included and influenced by the Physique modifier, you’d use the reinitialization mechanism to effect its inclusion. Or maybe you’ve repositioned the biped structure relative to the mesh, or scaled both; you’d need to reinitialize Physique settings to recognize those changes.

**Repel Behavior.** The Repel behavior lets you specify any object or objects (sources) that will force delegates to move away from them. This is basically the opposite of the Seek behavior. If you want delegates to back away from an object, as opposed to turning to face the direction they’re moving, use Repel in conjunction with the Orientation behavior.

**Rubber Band Mode.** Rubber Band mode provides a way to proportion the arm and leg links simultaneously, by moving the link with

the Move transform, instead of using scale. Rubber Band mode scales both the link and it's child in a single step.

This is particularly useful when fitting a biped to a skin prior to applying the Physique modifier. For example, rubber-banding the upper arm rescales the upper and lower arm objects and moves the elbow link without affecting the position of the shoulder or the wrist. If you've spent a lot of time getting the fingers in the right place you can reposition the elbow by rubber-banding, without affecting the hands.

**Scale Stride.** In the Footstep Operations rollout Scale Stride changes the width or length for the selected footsteps. The selected footsteps are scaled around the first footstep in the selection.

**Scripted Behavior.** A behavior defined by MAXScript.

**Scripts.** A Script is a list of clips (BIP files) that are executed sequentially to animate a character. Scripts are created either manually or automatically using the Crowd system. Scripts are created in Motion Flow mode.

**Seek Behavior.** The Seek behavior lets you specify any object or objects as a stationary or moving target for delegates. Delegates move toward the target during the crowd simulation while turning as necessary.

**Sliding Footstep.** Changing biped foot key parameters enables the biped feet to move or slide during a footstep period. This feature is also available for motion capture file import to allow the biped feet to slide or pivot. Sliding footsteps display with a gold color in the viewports and in Track View.

**Space Warp Behavior.** The Space Warp behavior lets you assign a space warp, such as wind or

gravity, to a delegate or delegates. Any space warp that works with dynamics can be used with the Space Warp behavior.

In particular, use the Space Warp behavior to tie delegates to a Vector Field Space Warp, so that they avoid an object while following its contours.

**Speed Vary Behavior.** The Speed Vary behavior is useful for objects whose velocity changes as they move, such as strolling tourists who might occasionally slow down to do some sightseeing.

**Spline Dynamics.** Turning on Spline Dynamics will create center of mass keys, without gravity and balance calculation, for newly created footsteps.

**Support Period.** The period where one or both of the biped feet are on the ground.



**Surface Arrive Behavior.** Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Surface Arrive Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Surface Arrive Behavior

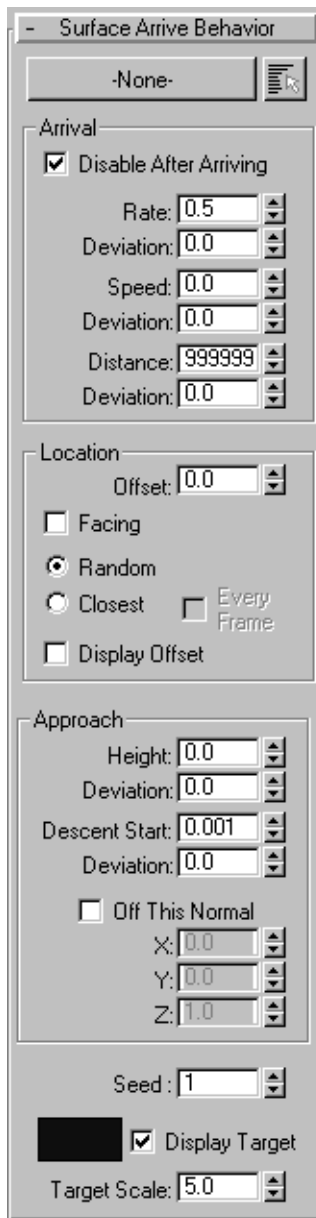
The Surface Arrive behavior is similar to the *Seek behavior* (see page 403) in that it lets you specify an object or objects as a stationary or moving target for delegates. The principal difference is that you can use the Approach settings to specify an intermediate target. After reaching this location, the delegates will then make their final approach to the ultimate target surface. An example would be birds flying over a row of telephone poles, and then each one dropping to land on top of a different pole.

## Procedure

### To use the Surface Arrive behavior

1. Add a Surface Arrive behavior to the Crowd object.
2. Add an object or objects to serve as the target surface to the scene.  
Note: If you use multiple objects, delegates will arrive at the surface of the closest object.
3. In the Surface Arrive Behavior rollout, use the None button or the Multiple Selection button to designate the one or more target objects.
4. Change the default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



## Surface Arrive Behavior rollout

**None (label).** Specifies a single target. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you've selected multiple targets using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are designated as targets, click the Multiple Selection button.

**Multiple Selection.** Opens the Select dialog to let you designate multiple targets. When you have more than one target, you can set delegates to move toward the closest target in the group, or to a computed average of the target positions.

### Arrival group

Specifies three aspects of the Surface Arrive behavior: Rate, Speed, and Distance.

**Disable After Arriving.** When on, turns off the Surface Arrive behavior after the delegate arrives at the surface. Default=on.

**Rate.** A multiple of the delegate's *Max Accel* (see page 343) setting that specifies the acceleration with which it will try to arrive. A value of 1.0 means to use the full acceleration of the delegate. Default=0.5.

**Deviation.** Adds random variation to the to the Rate setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Rate setting. Default=0.0.

**Speed.** The speed at which to arrive, relative to the speed of the target. Default=0.0.

**Deviation.** Adds random variation to the Speed setting. The actual deviation is calculated by multiplying the Deviation setting by a random

number between -1 and 1, and then multiplying the result by the Speed setting. Default=0.0.

**Distance.** The maximum radial distance from the target within which the behavior will be active. Until the delegate is within this radius, the behavior has no influence. Default=9999999.0.

**Deviation.** Adds random variation to the to the Distance setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Distance setting. Default=0.0.

### Location group

**Offset.** Specifies a consistent distance from the calculated arrival point, based on the surface normal, for the delegate to use. Default=0.0.

**Facing.** When on, the delegate will try to arrive only at points on triangles on the surface that are facing it. Default=off.

**Random.** The software chooses a random point on the target surface as the arrival point. When using the Random option, the software chooses arrival points for delegates once, at the beginning of the simulation. This is the default choice

**Closest.** The software chooses the closest point on the target surface as the arrival point. If Closest is chosen, but Every Frame is off, the software chooses arrival points for delegates once, at the beginning of the simulation.

**Every Frame.** When on, the software chooses arrival points for delegates at every frame. Available only when Closest is chosen. Default=off.

Every Frame is useful when the target object is rotating during the animation, but requires more time for calculation.

**Display Offset.** When on, shows the Offset distance as lines emanating from each vertex in the surface object, perpendicular to the surface.

### Approach group

The Height and Descent settings together specify the path the delegate will take for its arrival. They allow for a wide range of behavior, from soft, gradual landings to direct helicopter-type descents.

In both cases, the actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and multiplying the result by the relevant value.

**Height.** Specifies a distance from the arrival point along its face normal. This is the point that the delegate will go to first before descending to the arrival point.

**Deviation.** Adds random variation to the to the Height setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Height setting.

**Descent Start.** Specifies the distance between the delegate and the arrival point at which the descent should start.

Note: Be careful that Descent Start is set high enough that the delegate won't overshoot when descending because its speed is too high and deceleration too low, compared to when it should start descending.

**Deviation.** Adds random variation to the to the Descent Start setting. The actual deviation is calculated by multiplying the Deviation setting by a random number between -1 and 1, and then multiplying the result by the Descent Start setting.

**Off This Normal.** When on, lets you set an approach vector to specify the angle at which the final approach occurs. Default=off.

**X/Y/Z.** Use these settings to specify the final approach vector in world coordinates. For example, the vector specified by the default settings of X=0, Y=0, Z=1 means that the delegates will approach the target along the vertical world axis.

**Seed.** Affects the random numbers used to calculate the Deviation settings. For similar randomization among different Surface Arrive behaviors, use the same Seed value.

**Color Swatch.** Shows the color used to draw the target icon. Default=dark blue.

**Display Target.** Enables display of the target icon, which appears during the solution when a new interim goal is calculated for the delegate. Interim goals are created when using the Approach group settings. Default=on.

**Target Scale.** Specifies the overall size of the target icon. Default=5.0.



**Surface Follow Behavior** Create panel > Helpers > Object Type rollout > Crowd > Setup rollout > New button > Surface Follow Behavior

Select a Crowd object. > Modify panel > Setup rollout > New button > Surface Follow Behavior

The Surface Follow behavior moves delegates with respect to object surfaces, which can be still or animated. For example, you can apply an animated Noise modifier to a patch grid to simulate a choppy water surface, and objects guided by Surface Follow will stay on top.

Note: By default, a delegate influenced by Surface Follow picks a direction to move in at any given frame based on its current facing and the plane of the face it's currently over. Thus objects moving up a hill, while seeking a point

at the bottom of the other side of the hill, tend to turn left or right to skirt the hill, rather than following the upward slope. You can override this with the Projection Vector option.

## Procedure

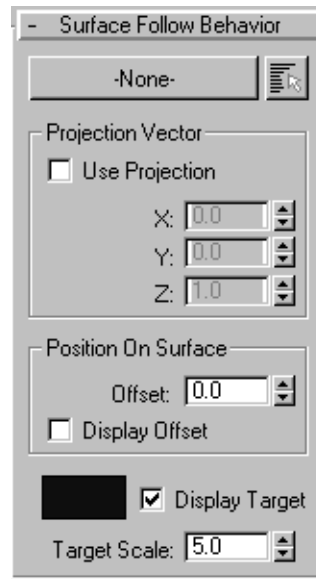
### To use the Surface Follow behavior

1. Add a Surface Follow behavior to the Crowd object.
2. Add an object or objects to serve as the follow surface to the scene.

Note: If you use multiple objects, they must intersect to form a contiguous surface. Each delegate will move to the closest surface, follow it to the next closest that it encounters, and then start following that one, and so on.

3. In the Surface Follow Behavior rollout, use the None button or the Multiple Selection button to designate the object or objects whose surface(s) the assignees are to follow.
4. Change the default settings as desired.
5. Use *Behavior Assignments* (see page 375) to assign the behavior to a delegate or team.

## Interface



### Surface Follow Behavior rollout

**None (label).** Specifies a single “target” object to use as a surface. Click this button, and then click the target object in the viewport. The target name then appears on the button.

If you’ve selected multiple targets using Multiple Selection (see next item), the word Multiple appears on the button. To see which objects are designated as targets, click the Multiple Selection button. The Select dialog appears with designated targets highlighted.

**Multiple Selection.** Opens the Select dialog to let you designate multiple targets. When you have more than one target, delegates initially move toward the closest target in the group, and then move over its surface until they encounter another target, at which point they switch to its surface, and so on.



### Projection Vector group

These controls let you override the default direction calculated by the Surface Follow behavior by describing a virtual plane along which the delegate is to move. You do this by specifying a vector, in world coordinates, that's perpendicular to the desired virtual plane.

For example, if you want the delegate, when it encounters a hill, to keep moving forward, straight up and over the hill, instead of skirting it, you would use the default Projection Vector settings:  $X=Y=0, Z=1$ . This vector is aligned with the world Z (vertical) axis, so it specifies a plane parallel to the world XY plane. Thus, the delegate always moves straight ahead while following the surface.

**Use Projection.** When on, Surface Follow calculates delegate direction from the specified vector, rather than using the default.

**X/Y/Z.** Specifies a vector using world coordinates. Default= $X=Y=0, Z=1$ . Range=-1.0 to 1.0.

If only one of these settings is not 0, then the projection vector is aligned with the non-zero axis. Combine non-zero settings to create angled planes for Surface Follow. For example, to create a virtual plane that's rotated 45 degrees clockwise about world Y axis, set  $X=Z=1$  and  $Y=0$ . Also, while you can set all three axes to 0, that specifies no vector, and so effectively turns off Use Projection.

### Position on Surface group

**Offset.** Specifies the delegate's distance above the surface, using the surface normal. Recalculated at each frame.

**Display Offset.** When on, shows the Offset distance as lines emanating from each vertex in the surface object, perpendicular to the surface.

**Color Swatch.** Shows the color used to draw the Surface Follow target (see Display Target, next) during the solution. Click the box to choose a different color.

**Display Target.** When on, the interim goal for each delegate influenced by the Surface Follow behavior is drawn in the viewports as a wireframe sphere during the simulation solution.

If the delegate starts out away from the surface to be followed, the target is most visible before the delegate reaches the surface; the target is then positioned along the surface edge. While the delegate is actually following the surface, the target is usually coincident with the delegate, because Surface follow sets a new destination only a frame or two ahead.

**Synthesis, Synthesize** The process of computing (solving) motions for crowd simulations.

**Talent Figure Mode.** After loading a raw marker file, turn on Talent Figure mode to scale the biped relative to the markers. Calibration for the entire marker file takes place when you exit Talent Figure mode.

**TCB.** Tension, Continuity and Bias are used to change the trajectory of a limb through keys.

**Tendons.** After adjusting envelope parameters for good mesh deformation, use tendons to control the amount of skin stretching across multiple links. While envelopes provide smooth skin deformations, tendons provide additional stretching in much the same way that actual human tendons might create pulling in the wrist (several joints away) when the fingers are moved.

**Tension, Continuity, Bias.** See *TCB* (see page 737).

**Touch.** Touch is the state of the biped foot on the first frame of a footstep.

**Trajectory.** Whenever an object moves through world space, you can view its trajectory. A trajectory is the visible path the object makes because of its movement. You can think of a trajectory as a three-dimensional function curve for the Position track of an object.

**Transform Gizmo.** This gizmo displays in the viewports and allows you to transform or rotate the biped objects.

**Vector Field.** A vector field is a special type of space warp that crowd members can use to move around irregular objects such as a curved, concave surface. The vector field object, a box-shaped lattice, is placed and sized so that it surrounds the object to be avoided. The vectors are generated from the lattice intersections. These vectors are, by default, perpendicular to the surface of the object to which the field is applied; if necessary, you can smooth them out with a blending function. The crowd members move around the object by traveling perpendicular to the vectors.

**Vector Field Space Warp.** The Vector Field space warp, when used as a Space Warp Behavior in a crowd simulation, allows crowd members to move automatically around *obstacle objects* of any shape, following the object contours. It also lets crowd members move within the confines of an enclosed space, such as a room, while avoiding the walls. And you can use vector field space warps to control particle motion.

The Vector Field space warp works by generating a number of vectors that surround an object and are perpendicular to its surface. The software then uses these vectors to guide delegates

around the object by moving them perpendicular to the vectors.

**Velocity Interpolation.** By default, in a transition between two motion clips, velocity is interpolated to blend smoothly between clips. If transitions are optimized then a sophisticated algorithm is used that minimizes sliding feet.

**Walking Gait.** An example of a walking gait has the feet moving from a left foot support, then both feet support and then right foot support.

**Wall Repel Behavior.** The Wall Repel behavior uses a grid object to repel delegates. When influenced by the Wall Repel force, delegates turn until they're heading away from the grid. It's useful for keeping objects inside an enclosed, straight-sided enclosure, such as a room in a building.

**Wall Seek Behavior.** The Wall Seek behavior uses a grid object to attract delegates. When influenced by the Wall Seek force, delegates turn until they're heading toward the grid. It's useful for moving objects toward a rectangular area, such as a doorway.

You can set the grid to attract from either side or both sides, and optionally specify a maximum distance for attraction. You can also set the behavior to act as though the grid extends infinitely along its plane.

**Wander Behavior.** The Wander behavior imparts a random motion to delegates, letting you simulate meandering activity in which delegates move and turn in a haphazard manner. It works by randomly picking a new direction, and then turning and moving in that direction. You can specify how often to pick a new direction, how far to turn, and how fast or slow to turn while moving.

**Workflow.** A series of steps to perform a task.

**World Space.** Place a biped limb in the world coordinate space to make it react. Placing the feet in world coordinate space allows you move the center of mass and have the feet stay planted in space for example.



---

# Index

## Numerics

- 2 3 4 links 331, 335
- 2 feet down 248, 255
- 3D Studio MAX
  - bones 11, 136, 294
  - knowledge of 2

## A

- about
  - buttons 143, 296
  - footstep animation 51
  - freeform animations 30
- absolute 331
- acceleration 291
- activate options 178
- activating footsteps 59
- active 307
- active time segment 241
- active/inactive footsteps 721
- actual stride height 250, 253
- actual stride length 250, 253, 255
- actual stride width 250, 253, 255
- adapt locks 66, 181, 721
- adaptation 721
- adapting
  - keyframes to edits 66
  - keys to footstep edits 66
- add change option 298
- adding
  - extra limbs 598
  - footsteps 57
  - poses 103
- adjust talent pose option 186
- adjusting
  - biped arms/legs/torso 93
  - default envelope shape 97
  - keys in Track View 65
  - link parameters 100
  - multiple keys 38
  - physique 548
  - talent pose 186, 721
  - vertical dynamics 63
- advanced biped features 72
- advanced inverse kinematics 2
- after trajectory 177
- airborne option 248, 253, 255
- airborne periods 64, 721
- aligning biped to mesh model 544
- all bipeds 177
- all links 331
- alternate 250, 253, 255
- anatomy of biped 22
- anchors 173
- angle 192, 237
- animatable IK attachments to 3DS MAX objects 79
- animating
  - a leap 586
  - a pratfall 494
  - behavior assignments 562
  - biped with footsteps 469
  - carrying and dropping 516
  - climbing a ladder 507
  - crowds 555
  - muscles with the bulge editor 552
  - using freeform techniques 29
  - walk cycle with IK constraints 457
- animating,a biped with footsteps 51
- animation 290
  - animation properties rollout 181
  - combining animations 85
  - expanding tracks 30
  - layers 721

- previewing animations after attaching
    - Physique 96
  - sample animations in this release 24
  - selecting and moving tracks 30
- ankle attach option 24, 184
- ankle tension 167
- append footsteps 57, 147
- append to end of script 234
- apply increment 38, 173, 269, 722
- applying
  - avoidance 562
  - logic to crowd behavior 564
  - physique 95, 548
- arms
  - arm link 184
  - fitting to skin 93
  - resizing 47
  - turning on 24
- assign to link 104, 331
- associate bipeds with delegates 374
- attach points/tendon 326
- attach to node 296
- attached links 326
- attaching
  - mesh to a biped using Physique 296
  - object instance to crowd system 116
  - tendon to another link 103
- attachments (IK) 722
- authorization xxvi, 143, 146
- auto 192
- auto clip names 230
- auto timing 250, 253, 255
- autogrid 722
  - creating biped 21
- automatic transitions 234, 237
- Avoid\_Behavior ReferenceTarget 671
- avoidance behavior 115, 392, 562, 722

**B**

- backward knees (creating characters with) 33
- balance factor 161, 722
- ballistic gaits 722
  - dynamics of 64
- Ballistic Tension 64
- ballistic tension 22, 160, 161, 291, 481, 722
- base
  - 161

- base index
  - 161
- batch file conversion 186
- batch file conversion (motion capture) 197
- before trajectory 177
- behavior assignments and teams dialog 375
- behavior rollout 351, 392
- behaviors 113, 556, 722
  - avoid 392, 722
  - obstacle-avoidance 115, 730
  - orientation 396, 730
  - patch-based 730
  - path follow 398, 730
  - repel 400, 731
  - scripted 402, 732
  - seek 403, 732
  - space warp 404, 732
  - speed vary 405, 732
  - surface arrive 406, 732
  - surface follow 409, 735
  - wall repel 411, 738
  - wall seek 414, 738
  - wander 416, 738
- bend
  - 258, 307
- bend links 722
- bend links mode 36, 147
- bend parameters (links) 100
- bending
  - center of mass track 82
  - footstep path 58
- bias 100, 161, 307
- Biovision motion capture data files 186
- BIP files 24, 186, 284, 723
- Biped 723
- biped 5, 143
  - aligning to mesh 544
  - and physique 11
  - and physique (Tutorial 1) 429
  - body parameters 20, 24
  - center of mass 5
  - creating 20
  - display options 177
  - dummies 139
  - dynamics 5, 181
  - dynamics parameters 161
  - editing keys in Track View 65

- features 2
  - figure files (.fig) 46
  - load motion file 147
  - naming 22
  - playback 147, 723
  - root object 22
  - select keys based on foot states 269
  - setting keys 37
  - tracks in Track View 30
  - user interface 133
  - with physique 438
  - Biped Classes 623
  - Biped Controllers 610
  - Biped Creation 603
  - Biped Display Preferences 605
  - Biped Footprints 621
  - Biped Footsteps 621
  - Biped Keys 636
  - Biped Layers 609
  - Biped Load and Save 601
  - Biped Motion Flow 628
  - Biped MultFprintParams Class 623
  - biped multiple keys dialog 38
  - biped playback 147
  - Biped Slave Controller 620
  - Biped Vertical\_Horizontal\_Turn(Body)
    - Matrix3 Controller 610
  - biped.ini file xix
  - biped\_object GeometryClass 606
  - BipedFSKey MAXObject 627, 640
  - BipedKey MAXObject 637
  - bipeds
    - adjusting to fit skin 93
    - deleting 24
    - linking objects to 50
    - moving objects 33
    - rotating objects 33
    - scaling after physique is applied 106
    - visible in playback 177
  - bipeds,posing posing a biped 45
  - blend from/to 307
  - blending
    - between links 88, 331, 335
    - envelopes 97
    - forward and inverse kinematics 503
  - block controllers (track view) 119
  - blue vertices 331
  - body 161, 167
    - horizontal tracks 22, 160
    - parameters (biped) 24
    - space 723
    - tracks 22
    - turning track 160
    - vertical tracks 22, 134, 160
  - bones 136, 176, 294
    - bone base 177
    - bone tip 177
    - bones rollout 295
    - used with physique 91
  - both 300
  - both feet support 237
  - boundary conditions (and tendons) 103
  - bounding box (and envelope creation) 94, 96
  - bounds (inner/outer) 88
  - box generator utility 94, 294
  - breathe option (links) 100, 307
  - browse 192
  - buffer mode 68, 147
  - bulge angles 313, 318, 723
    - adding 102
    - changing 102
    - choosing for editing 102
    - color 313
    - deleting 102
    - parameters 318
    - setting 102
  - bulge editor 103, 296, 307, 313, 318
    - and animating muscles 552
  - bulge sub-object 313
  - bulges 335, 338, 723
    - creating 101
    - fine-tuning 103
    - overview 90
    - shaping 102
    - workflow 101
  - BVH files 83, 186, 199, 284, 723
- ## C
- CAL files 192
  - calibrating marker files 186
  - carrying and dropping motions 516
  - center of mass 5, 134, 292, 723
    - object 22
    - selecting tracks 32

- shadow 22
- tracks in Track View 160
- changed feature in v3 130
- changing
  - biped body parameters 20
  - biped name 22
  - initial biped anatomy 24
  - initial biped structure 43
- character studio
  - authorizing xxvi
  - marker files (.csm) 198
  - supported marker names 198
  - version 3 xix
- child overlap 300
- clear 300
- climbing motion 507
- clip controllers 119, 723
- clip mode 230, 234
- Clip Properties Dialog 246
- clip properties dialog 230
- clips
  - combining 85
  - creating in motion flow mode 526
  - looping with motion capture filtering 83
  - name of 230
- cloning characters 139, 442
- CogControl ReferenceTarget 670
- cognitive controller 723
- cognitive controller editor 382
- collapse layers 178
- collapsing animation tracks 30
- collision detection 33
- collisions rollout 358
- colors 65
  - biped keys in Track View 65
  - footsteps 53, 60
  - in Track View 177
  - vertex type 104
- COM 134
- combining animations 85
- comparing trajectories 538
- composing footsteps 53
- connect
  - to child link 326
  - to parent link 326
- continuity 100, 161, 307
- control points 300, 313, 318, 326, 723
  - and bulges 102
  - and envelopes 97
  - rotating 97
- controller 724
- convert 147
  - between footstep and freeform animations 70
  - data in motion capture buffer 186
  - from buffer 186
  - to freeform 30
  - to freeform/footsteps dialogs 158
- coordinate space 724
- copy 234, 300, 313, 326
  - footsteps 258
  - motion 530
  - pose 173
  - posture 173
  - postures 40
  - selected cross section 318
- copy tracks 41
- copyrights xxv
- crease
  - at parent 307
- create
  - a biped 143, 429
  - clip 230
  - clips from files xix
  - envelopes 335
  - footsteps 248
  - keys for inactive footsteps 59, 258
  - layer 178
  - new script 234
  - separate tracks for biped arms 181
  - transition 237
- create biped rollout 20
- create method rollout 419
- create multiple footsteps 248, 250
  - jump 248, 255
  - run 248, 253
  - walk 248, 250
- create panel 143, 146
- create random motion dialog 241
- creating
  - a script 229
  - biped character 20
  - biped skin 93
  - biped with AutoGrid 21
  - bulges 101



- clips in motion flow mode 526
  - crowd of swimming bipeds 573
  - crowd system 111
  - expressive walk 470
  - footstep animations 53
  - footsteps 57
  - freeform animations 29, 30
  - illusion of weight 519
  - individual footsteps 57
  - motion flow scripts 526
  - multilegged creature 585
  - multiple footsteps 57
  - physique links and envelopes 335
  - physique modifier 95
  - simple freeform animation 447
  - skin 92
  - creation parameters 20
  - cross section editor viewport options 313
  - cross sections 298, 300, 313, 318, 326, 724
    - and bulges 102
    - and envelopes 97
    - and tendons 103
    - parameters 318
    - view (bulge editor) 103
  - crowd
    - advanced crowd/bipeds 577
    - attaching object instances to 116
    - behaviors 113, 115
    - creating crowd systems 111
    - crowd object 110
    - delegates 111, 116
    - with animated non-biped objects 567
  - crowd animation 555
    - user interface 341
    - using 109
  - Crowd Behaviors 671
  - crowd behaviors 112
  - Crowd helper 649
  - crowd helper object 346, 724
  - crowd system 724
  - CrowdAssignment ReferenceTarget 663
  - Crowds Methods 667
  - CrowdScatter 657
  - CrowdState ReferenceTarget 665
  - CrowdTeam ReferenceTarget 664
  - CrowdTransition ReferenceTarget 666
  - CS amplitude option (links) 100
  - CSCustomKeys.cui Tutorial 705
  - CSM files 83, 186, 724
    - file structure 198
    - sample 284
    - specification 213
  - current bulge angle 313, 318
  - currently assigned links only 331
  - custom keys 141
  - customizing biped characters in figure mode 42
  - cut 234
  - cycle time 292
- ## D
- deactivate footsteps 59, 258
  - default color 313
  - define script 234
  - deformable
    - envelopes 11, 88, 99, 298, 300, 335, 724
  - deformation 95, 724
  - deformation spline 88, 96, 100
  - degree of freedom and rotating links 33
  - Delegate Helper 651
  - delegates 111, 724
    - helper objects 342, 724
    - using bipeds with 116
  - delete 300, 313, 326
    - biped 24
    - bulge angle 313, 318
    - bulge angles 102
    - bulge cross sections 102
    - clip/transition 230
    - cross section slice 318
    - ctrl pts 318
    - footsteps 58, 258
    - key 37, 161, 173
    - layers 178
    - script 234
    - transition 237
  - destination directory 197
  - detach 326
  - digit index
    - 161
  - disabling playback 96
  - display
    - footsteps 176
    - marker 186

- options/
  - preferences 22, 27, 28, 97, 176, 177, 300, 313
- trajectories 27
- vertex weight values 105
- display rollout 176
- display subtree 300
- divisions 313, 318
- double support 248, 250, 724
- draw ctrl pts 318
- draw in profile view 318
- dribbling a basketball 504
- dummies 139
- dynamics 723, 724
  - and footsteps 262
  - dynamics blend 134, 160, 161, 292, 724
  - of ballistic gaits 64
  - options 22

## E

- ease options
  - key info rollout 161
  - transition editor 237
- edit
  - clip 234
  - footsteps 147, 267
  - freeform 267
  - time 262
  - transition 234
- edit commands 300
  - and envelopes 97
- edit keys 262
- edit multiple delegates dialog 372
- editable meshes 724
- editing
  - active footsteps in time 66
  - footsteps 248
  - footsteps in place 58
  - footsteps in space 58
  - footsteps in time 60
  - motion flow 525
  - transitions manually (ghosts) 237
  - with layers 540
- editing, footsteps in Track View 60
- end effector 136
- entire link 313, 318
- envelope parameters 300

- envelope sub-object 300
- envelopes 725
  - adjusting shapes 97
  - and control points 97
  - and cross sections 97
  - and edit commands 97
  - and weighted vertices 11, 549
  - blending types 97
  - choosing default fit 96
  - choosing default types 96
  - copying 97
  - copying to mirrored link 97
  - display options 97
  - exclude for selected links 300
  - excluding influence 97
  - overview 88
  - scaling size 97
  - selecting 97
  - types of 88
  - updating display manually 97
  - using transforms with 97
  - working with 99
  - working with both envelope types 104
  - working with rigid 103
- exclude option 97, 300
- expanding animation tracks 30, 32
- extraction tolerance 192

## F

- facial animation 131
- fall - animating 494
- falloff 300, 307, 335
- FAQs and procedures (biped) 128
- features
  - Biped 2
  - Crowd 15
- FFDs 11, 725
  - and physique 340
- FIG (figure) files 147, 192
  - saving and loading 46
- figure mode 11, 43, 147, 184, 433, 725
- figure structure 192
- file specifications 199, 213
- file types
  - BIP 24, 723
  - BVH 199, 723
  - CAL 192

- CSM 198, 213, 723, 724, 728
  - FIG 147, 192
  - MFE 229
  - MNM 83, 198
  - MOC 186, 192
  - PHY 108, 296, 730
  - STP 70
  - filtering 725
    - motion capture and marker data 192
  - fine-tuning envelopes 97
  - fingers option 24
  - fit 300, 326
  - fit to existing 192
  - fitting
    - arms to skin 93
    - bipeds to skin 93
    - spine to skin torso 93
  - fixed 237
  - flatten footsteps 192
  - flexibility (neck and spine) 24
  - flips 481
  - floating bones 95, 295
  - foot states 5, 147, 269, 725
  - footstep
    - adaptation 247
    - animation (glossary) 725
    - animation workflow 51
    - converting to freeform 70
    - creating footstep animation 53
    - creation 57
    - display 176
    - edge selection 267
    - editing 247
    - editing footstep buffer 68
    - numbers 176
    - timing (gait parameters) 56
  - footstep creation 5, 147, 248
    - create multiple footsteps (jump) 255
    - create multiple footsteps (run) 253
    - footstep operations 258
  - footstep extraction 192
    - using motion capture filtering 83
  - footstep method 5
  - footstep mode 147, 247
  - footstep operations rollout 147, 258
  - FootSteps
    - Matrix3 Controller 621
  - footsteps 158, 469
    - activating 59
    - alternating 57
    - and IK keys 500
    - appending 57
    - bending path 58
    - composing pattern 53
    - convert to 158
    - converting to freeform 70
    - creating 57
    - creating multiple 57
    - deleting 58
    - editing in space 58
    - editing in time 60
    - footstep mode dialog 267
    - moving 58
    - planning for creating 53
    - rotating 58
    - specifying number when creating 57
    - splicing motion segments 68
    - starting side 57
    - timing 60
    - timing gait parameters 56
    - troubleshooting 70
    - workflow 53
  - FootSteps Matrix3 Controller 619, 626
  - footsteps method 725
  - forward kinematics 161, 726
    - blending with IK 503
  - frame 237
  - freeform 158
    - animations 29, 74
    - convert to 158
    - converting to footsteps 70
    - inserting period between footsteps 68
    - method 5
    - setting period in footstep animations 68
    - walking animation using IK constraints 30
  - freeform animation 30, 262, 586, 726
  - freeform method 726
  - freeform, converting to footsteps 70
  - from z level 192
- ## G
- gait pattern 726
  - gait type 726
  - general rollout 147

- generate colors 177
- geometric primitives 726
- geometry parameters rollout 342
- geometry rollout 359
- ghosts 237
- gizmo 726
- global clip 270
- global clip controller 359, 726
- GlobalClip 119
- globalmotionclip 119
- go to frame 234
- go to start frame 237
- GravAccel (gravitational acceleration) 22, 64, 134, 181, 726
- gravity 291, 727
- green
  - line 318
  - rigid vertices 331
- ground plane (and collision detection) 33
- groups
  - and attaching physique 95
  - animating 555
- gymnastic motion flips 481

## H

- height option 24, 184
- helper object 727
- hide 331
- hide attached nodes 298
- hide/show all 177
- hierarchy of biped objects (Track View) 30
- high frequency data 539
- hopping (dynamics of) 64
- hot keys 130
- how many 250, 253, 255
- how tos 123

## I

- IK 2
- IK Blend 503
- IK blend 62, 161, 167, 727
- IK constraints 30, 72, 74
  - animating
    - walk cycle with 457
- IK key info rollout 167
- importing

- marker file 83
- motion capture data 536
- motion capture file 83
- in place mode 147, 522, 727
  - and trajectory display 27
  - options 27
  - using to adjust keyframes 39
- include new bones 335
- incrementing keys 38
- influence 313, 318
  - areas of and envelopes 88
- influenced vertices 326
- initial pose 300, 313, 326, 331, 335, 338, 727
- initializing physique 96, 727
- inner 300
- inner/outer bounds 88
- insert 300, 313, 326
  - above selected clip item 234
  - below selected clip item 234
  - bulge angle 313, 318
  - cross section slice 318
  - ctrl pts 318
- inside 307
- installing the software xxvi
- instance 657, 727
- interactive redraw 313, 326
- interpolation 250, 253, 255, 727
  - stride 57
- introduction 1
- inverse kinematics 2, 161, 727

## J

- join to previous key option 167
- joint intersections 298
  - parameters 101, 105
- joint intersections rollout 307
- joint rotation data (in BVH files) 83
- jump 147, 248
- jumping
  - and running 477
  - dynamics of 64
  - parameters 56

## K

- key info rollout 72, 161
- key interpolation 5

- key reduction
  - settings 192
  - using motion capture filtering 83
- keyboard shortcuts 130
- keyframe animation 728
- keyframes
  - adapting to edits 66
  - adjusting with in place mode 39
- keyframing 173
  - IK Blend 516
  - the biped 5
- keys (setting) 37

## L

- ladder
  - animating climbing 507
- landing dynamics 64
- lattice parameters rollout 419
- layers 178, 529, 728
- left arrow 300
- left foot support 237
- leg link 184
- leg motion phases 63
- length 237, 258
- lift 147, 269, 728
- lift phase (leg motion) 63
- link 161, 300, 313, 326, 554
  - biped hand to an object 161
  - blending 298
  - envelopes list (left side) 300
  - index
    - 161
  - length 335
  - length as basis for envelope creation 94, 96
  - linking objects to biped 50
  - name 331
  - scale 100, 307
  - sub-object 307
  - to root attach node 296
- links 728
  - adjusting parameters 100
  - and joint settings 335, 338
  - blending between 88, 96
  - moving 33
  - parameters 100
  - radial scale parameters 100
  - rotating 33

- scaling 46
- selecting and rotating 36
- setting parameters 100
- sliding parameters 100
- transforming 32
- twist parameters 100
- load 192
  - buffer only 186, 192
  - file option 147
  - marker name file 186
  - motion capture file 186
  - parameters 192
  - specification 293, 296
- loading
  - biped figure files 46
  - biped step files 70
  - motion files 24
- locate vertical center of mass keys 160
- lock assignments 104, 331
- logic and crowd behavior 564
- looping 83, 192, 539
- lower bound 326

## M

- manual update (envelopes) 300
- mapping biped motion 69
- marker data 83, 728
- marker display dialog 197
- marker files 186, 723, 728
  - importing 83
  - name files 83
- marker name file dialog 83
- marker names 198
- master motion clip 728
- masterclip 119
- mastermotionclip 119
- maximum angular/positional deviation for a
  - track 192
- merging characters 139, 442
- mesh object (as Physique skin) 92
- mesh size (reducing) 106
- MFE files 229
- minimum key spacing 192
- minimum motion loss 234
- mirror 81, 173, 300, 313, 326
- mirror selected cs 318
- mirrored link

- copying envelope settings to 97
  - mirroring 728
  - mirroring motion 81
  - MNM files 83, 186, 198
  - MOC files 186, 192
  - modes 147, 227, 247
    - bend links 36
    - buffer 68
    - in place 26, 522
    - motion flow 526
    - rubber band 48
    - talent figure 83
  - modifying
    - biped structure in figure mode 433
    - footsteps 477
  - MoFlowScript MaxWrapper 630
  - MoFlowSnippet MaxWrapper 631
  - MoFlowTranInfo MaxWrapper 633
  - MoFlowTransition MaxWrapper 635
  - motion
    - duplicate 530
    - mapping 69
    - mirroring 81
    - sequence 68
    - splicing 68
  - motion blending 728
  - Motion Capture 641
  - motion capture 62, 729
    - batch file conversion dialog 197
    - buffer 83, 186, 192
    - conversion parameters dialog 192
    - converting data from buffer 186
    - file 192
    - importing data 536
    - importing files 83
    - rollout 83, 186
    - working with 535
  - motion capture files 284
  - motion clip 270, 729
  - motion files 729
    - information saved in 24
    - loading motion files 24
    - samples 24
  - Motion Flow 628
  - motion flow 526, 729
    - BIP file location 24
    - creating and using scripts 526
    - creating clips 526
    - editing 525
    - editor files 229
    - graph 229, 230
    - mode 147, 227, 229, 234
    - rollout 229
  - motion flow editor 729
  - motion flow mode 147, 729
  - motion flow script rollout 234
  - motion flow scripts 729
  - motion interdependencies 68
  - motion library 284
  - motion panel 146
  - motion parameters rollout 342
  - motion synthesis 729
  - motion transfer 530
  - motionclip parameters dialog 283
  - MotionClips and GlobalMotionClip 698
  - move 147, 269
  - move clip 230
  - move phase (leg motion) 63
  - moved pivot
    - 161
  - moving
    - center of mass object 48
    - links 33
  - multilegged characters 585
  - multiple delegates and behaviors 559
  - multiple keys
    - selecting 38
  - multiple links 36
    - selecting and rotating 36
- ## N
- N links 729
  - n links 331, 335
  - neck link 184
  - new feature in
    - v3 xix, 341, 342, 346, 351, 352, 354, 357, 358, 359, 360, 362, 370, 372, 374, 375, 380, 381, 382, 385, 386, 392, 396, 398, 400, 402, 403, 404, 405, 406, 409, 411, 414, 416, 417, 419, 420, 732, 735
  - next key-previous key 161, 167
  - next transition 237
  - next/previous 300, 313

- no blending 331, 335
- no footsteps 192
- no key reduction 192
- no support 237
- non-biped skeletons 94
- normalized 331
- numbers
  - of links that can affect a vertex 11
  - show/hide all 177
- NURBS 11

## O

- object bounding box 335
- object instance 729
- object space 729
- object space object 161, 167
- object/delegate associations dialog 370
- objects 161, 167, 176
- obstacle parameters rollout 420
- obstacle-avoidance behavior 115, 730
- on selected objects/on all objects 197
- only extract footsteps within tolerance 192
- open physique file 296
- open physique file button 108
- operating requirements xxv
- opposite tracks 160
- optical markers 730
- orientation bar 313, 318
- orientation behavior 396, 730
- Orientation\_Behavior ReferenceTarget 673
- outer 300
- outside 307
- overlap 335
- overview
  - biped 2
  - crowd 13, 112
  - physique 11
  - workflow 15

## P

- parametric stride length 250, 253, 255
- parametric stride width 250, 253, 255
- parent overlap 300
- partial blending 300, 306
- paste 234, 300, 313, 326
- paste footsteps 258

- paste from buffer 186
- paste posture 173
- paste posture opposite 173
- paste selected cs 318
- pasting 40
- patch-based objects 730
- path follow behavior 398, 730
- Path\_Follow\_Behavior ReferenceTarget 675
- perform footstep extraction 192
- performance
  - and biped 25
  - optimizing with physique 106
- period 730
- phases of leg motion 63, 730
- PHY files 108, 296, 730
- physique 11, 731
  - and biped 438
  - and changing geometry 108
  - and FFDs 340
  - and groups 95
  - and other modifiers 107
  - applying 95
  - applying and adjusting 548
  - deformation spline 11
  - getting started with 87
  - initializing 96
  - overview 11
  - possible problem areas 551
  - reinitializing settings 105
  - saving settings 108
  - scaling a character 553
  - skinning and linking with 543
  - storing settings in PHY files 108
  - user interface 293
- physique initialization dialog 96, 335
- physique level of detail rollout 298
- physique load specification dialog 293, 296
- physique rollout 296
- pinch 326
- pinch bias 326
- pivot points 161
- pivots (IK extensions) 77
- plant 147, 269, 731
- plant phase (leg motion) 63
- playback 27, 147
- point 192
- ponytails 184

- pose adjustment 192
- poses 731
  - adding 103
  - copying between bipeds 40
  - reference 93
- position data (in CSM files) 83
- position the entire animation 234
- positional markers 731
- postures 40
- power 313, 318
- pratfall 494
- preferences (display) 28
- preset keys 141
- previewing
  - biped motion 25
  - motion 96
- previous link/next link 318, 326
- previous transition 237
- priority rollout 354
- problem areas (physique) 551
- procedures 123
- product
  - authorizing xxvi
  - system requirements xxv
- productivity 5
- profile 313
- profile view 318
- prop bone 198, 731
- properties (clips) 230
- pull 326
- pull bias 326
- pull/pinch/stretch options (tendons) 103

## Q

- quadruped 586

## R

- radial scale 94, 300, 307
  - parameters (links) 100
- radius 326
- random motion 241
- random placement difficulty dialog 370
- real-time playback 96
- reassign globally 298
- reassigning vertices 104
- re-attachment 338

## red

- deformable vertices 331
- line 318
- redefine script 234
- reducing mesh size 106
- reference 731
- reference pose 93
- reinitialize 731
- reinitialize (physique) 105, 296, 338
- reinitialize selected links 100, 307
- relative scale 331
- remap locally 298
- remove from link 104, 331
- repel behavior 400, 731
- Repel\_Behavior 677
- requirements (system) xxv
- resizing arms 47
- resolution 313, 318, 326
- restructure biped to match file 147, 186
- reverse knees (creating characters with) 33
- right arrow 300
- right foot support 237
- rigid envelopes 11, 88, 103, 298, 300, 335
- rolling 237
- root name 22, 143, 184
- rotating 36
  - elbows and knees 33
  - links 33
  - spine 33
- rubber band mode 47, 48, 147, 731
- rubber-banding
  - arms and legs 47
  - center of mass 48
- run 147, 248
- run footstep 248, 253
- running 64
  - and jumping 477
  - dynamics of 64
  - in place 26
  - parameters 56

## S

- Sample Scripts 605
- samples 290
  - animations 24, 290, 291, 292
  - motion files 24
- save file 147



- save parameters 192
- save physique file 296
- save script to a BIP file 241
- save segment 147
- save segment dialog 241
- save segment in motion flow mode 241
- save talent figure structure 186
- save talent pose adjustment 186
- saving
  - BIP files 24
  - biped figure files 46
  - biped step files 70
  - physique data 108
- scale
  - 258, 313
  - a biped 338
  - factor 192
  - stride 147, 732
  - tail keys 269
- scaling
  - a biped 184
  - a node 100
  - and system units 106
  - arm 47
  - characters 339, 442, 553
  - height 64
  - links 46
  - tail keys 38
- scatter objects dialog 362
- scripted behavior 402, 732
- Scripted\_Behavior ReferenceTarget 678
- scripts 234, 732
- section view 318
- sections 313, 326
- seek behavior 403, 732
- Seek\_Behavior ReferenceTarget 686
- segments 318
- select 331
  - and rotate ctrl pts 318
  - and scale ctrl pts 318
  - and translate cs 318
  - by link 104, 331
  - clip/transition 230
  - multiple keys 38
  - multiple links 36
  - nearest bulge angle 102, 313, 318
- select behavior type dialog 380
- select button 104
- select delegates dialog 381
- select scale rotate control points 318
- selected deformable/rigid envelope areas 300
- separate tracks options 32, 181, 262
- set all 192
- set bulge angle 313, 318
- set free key 167
- set key 37, 161, 167, 173
- set lowest starting foot height to Z=0 147
- set multiple keys 173, 262, 269
- set parents 161, 173
- set planted key 167
- set sliding key 167
- set start frame 237
- Setting Biped Transforms 606
- setting keys 37
- setup rollout 360
- shaded display of vertex weight values 105
- shadow (center of mass) 22, 134
- shared motion flow 243, 530, 532
- shortcuts 130
- shoulders and crotches (problem areas in physique) 551
- show buffer 186
  - to show original motion 83
  - trajectory 186
- show entire trajectory 177
- show graph 229
- show markers 186, 197
- show prop markers 197
- show recognized markers 197
- show script dependencies 230
- show time 177
- show unrecognized markers 197
- show/hide all 177
- sides 300, 313
- skeletons 5
  - and physique 91
  - structure 11
- skin
  - attaching to skeleton 87
  - creating 92, 93
  - defined 87
  - deformable 87
  - fine-tuning 99
  - mesh 106

- optimizing 106
  - rigid 87
  - sliding 298
  - sliding parameters 100
  - valid types 92
- skin deformation xix
- skinning and linking with physique 543
- sliding 307
  - angle 192
  - distance 192
  - footsteps 62, 732
  - parameters (links) 100
- sliding angle 192
- sliding distance 192
- smooth 335
- smoothing rollout 357
- snap set key 178
- solve rollout 352
- source file selection 197
- space warp (vector field) 417
- space warp behavior 404, 732
- Space\_Warp\_Behavior 687
- specify conversion parameters once option 197
- specify parameters for each file option 197
- speed 250, 253, 255
- speed vary behavior 405, 732
- Speed\_Vary\_Behavior ReferenceTarget 688
- spine
  - fitting to skin torso 93
  - flexibility 24
- spine link 184
- splicing biped motion 68
- spline dynamics 5, 51, 134, 181, 732
- spline-based deformation in physique 95
- springing dynamics 64
- stack updates 107, 298
- start after last footstep 250, 253, 255
- start at current frame 250, 253, 255
- start frame 234, 237
- start left 250, 253, 255
- start position x 234
- start position y 234
- start position z 234
- start right 250, 253, 255
- start rotation 234
- state dialog 275, 385
- state filters 269
- state transition dialog 386
- step files 70, 147
- stick 192
- stick figures (in transition editor) 237
- STP files 70
- strength 300
- stretch 100, 103, 307, 326
- stretch bias 326
- stride
  - length 291
  - parametric and actual 57
  - scaling length 58
- structure rollout 184
- support period 732
- surface arrive behavior 406, 732
- surface follow behavior 409, 735
- Surface\_Arrive\_Behavior MAXObject 689
- Surface\_Follow\_Behavior ReferenceTarget 693
- symmetrical tracks 160
- synthesis control dialog 119
- synthesis dialog 119, 270
- synthesis/synthesize 737
- system requirements xxv
- system units and scaling 106
- systems (bones and biped compared) 91

## T

- tails
  - adding 24
  - scaling 38
- talent definition area 192
- talent figure mode 83, 186, 737
- TCB 161, 737
- tendon attach points 326
- tendon attached links 326
- tendon cross sections 326
- tendons 87, 298, 326, 335, 338, 737
  - adding 103
  - adjusting 103
  - and fixed attach points 103
  - attach points 103
  - attaching to another link 103
  - boundary conditions 103
  - deleting 103
  - inserting 103
  - overview 90
  - using 103

- workflow 103
- tension/continuity/bias 161, 737
- time 161, 167
- time in the air 64
- time to next footstep 250, 253, 255
- timing parameters 248
- toes option 24
- tolerance 192
- touch 147, 269, 738
- touch phase (leg motion) 63
- track selection 160
  - using motion capture filtering 83
- track selection rollout 32
- Track View 262, 267
  - editing biped keys 65
  - editing footsteps 60
  - hierarchy of biped objects 30
  - opening 30
  - pick dialog 283
- trademarks xxv
- trajectories 161, 176, 738
  - display 27
  - displaying 27
- trajectory key editing 82
- transfer motion 530
- transform gizmo 738
- transforms
  - and envelopes 97
  - and links 32
- transition dialog 245
- transition editor 237
- transitions 237
  - between clips 234
  - state transition dialog 386
  - using 529
- triangle pelvis 184
  - and physique 24
- troubleshooting footstep keys 70
- twist 307
- twist parameters (links) 100
- type-in weights 331

## U

- understanding crowd behaviors 113
- unhide all 331
- unlock assignments 331
- up vector 192

- update 300
- upper bound 326
- use key reduction 192
- using 526
  - bend links mode 36
  - bipeds with crowd delegates 116
  - crowd animation 109
  - freeform animation 447
  - IK keyframe parameters 72
  - in place mode to adjust keyframes 39
  - layers 81, 529
  - transitions 529
  - tutorials 423
- using in place mode 522

## V

- vector field 738
- vector field space warp 417, 738
- Vector\_Field SpacewarpObject 699
- velocity interpolation 237, 738
- vertex 104
  - operations 331
  - settings 335, 338
  - vertex sub-object 331
- vertex-link assignments 335, 338
- vertical dynamics (adjusting) 63
- vertices
  - changing type 104
  - checking assignments 104
  - choosing type 104
  - making rigid 104
  - manually assigning deformable blended 104
  - manually overriding assignments 104
  - reassigning manually 104
  - removing deformable 104
  - working with 104
- viewing sample biped animations 24
- visible after/before 178

## W

- walk 147, 248
  - creating 470
  - footstep 248, 250
- walking gait 738
  - defined 63
- walking parameters 56

- wall behaviors
  - repel 411, 738
  - seek 414, 738
- Wall\_Repel\_Behavior MAXObject 694
- Wall\_Seek\_Behavior MAXObject 696
- wander behavior 416, 738
- Wander\_Behavior ReferenceTarget 697
- weight 313, 318, 331
  - assignments (vertex) 306
- weighted vertices 11, 300
  - and envelopes 549
- what you should know to use character studio 2
- width 258
- Workflow
  - tendons 103
- workflow 227, 739
  - applying physique 95
  - creating bulges 101
  - footstep animation 51
  - in character studio 15
  - motion capture 83
- world space 161, 739

## X

- xyz position 161

## Z

- zoom selected object option (Track View) 30