SOFTIMAGE®

SOFTIMAGE®|XSI™

Version 1.5

Tutorials

Documentation Team
Judy Bayne, André Demers, Grahame Fuller, François Giard, Edna Kruger, Luc Langevin, Gino Vincelli, and John Woolfrey.

Printed in Canada.

Part No. 0130-04618-02 1200

Contents

Roadmap

About This Guide

Tutorials gives you all the information you need to get up and running with SOFTIMAGE|XSI.

This guide is a collection of twenty-three tutorials, starting with the basics such as selecting, editing, and transforming objects in 3D space. After that come the major XSI toolsets: modeling and deformations, animating, lighting, texturing, scripting, and rendering.

- ***Tutorial 1: Editing Properties***
  Basic methods and tools for creating, selecting, and editing objects.
- ***Tutorial 2: The Interface***
  Create a hierarchy of objects, transform and duplicate them. Set visibility options, customize your layout using toolbars, and organize your scene using layers and groups.
- ***Tutorial 3: Polygonal Modeling—A Boombox for Burt***
  Using a variety of polygon modeling tools, build a boombox starting from a simple cube.
- ***Tutorial 4: NURBS Surface Modeling—Burt's Bumpy Back***
  Create an alien's head and body, then assemble them into a single surface mesh.
- ***Tutorial 5: Deformations and Weight Maps—Burt's Volcano***: Use weight maps to control deformations by painting a deformation onto a grid to create a volcanic landscape.
- ***Tutorial 6: Low-level Animation—Chaos at Brontasaurus and Main***
  Mark and set keyframes for animating a truck: Put a spaceship on a path, then link events in its animation with other events, using linked parameters, custom parameters, and expressions.
- ***Tutorial 7: Particles—Oozing, Oily, Slimy Sludge***
  Create a thick, particle-based liquid oozing from a truck and over the truck's tires.
- ***Tutorial 8: Fluid—Simulating a Liquid***
  Using fluid-type particles, create a liquid pouring from a jug. Plus, two fluid "recipes."
- ***Tutorial 9: Character Setup—Making a Man Out of William***
  Create a skeleton using 2D chains, set up a control rig to assist with animation, and assign and weight an envelope to your skeleton.
- ***Tutorial 10: The Animation Mixer—Jaiqua Poses***
  Part I: Create action sources and clips, then reuse, cycle, trim, and modify the actions.
  Part II: Clean up motion-capture function curves, create actions for various kinds of animation, and then mix these actions.
- ***Tutorial 11: The Animation Mixer—The Commander Attacks***
  Advanced tutorial. Mix and sequence actions; create transitions between the actions; break down the animation into frequencies (equalize); use offsets to solve animation problems; and more.

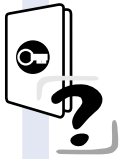
- ***Tutorial 12: Shape Animation—Hi Sailor!***
  Deform a head into different shapes, save shape keys as you go, and apply shapes to the head from a set of predefined models. Mix mix shapes and define custom parameters.
- ***Tutorial 13: Caustic Lighting—Psyched-Out Billiard Ball***
  Create caustics emitted from a sphere with displacement and a specular gradient. Use the render region to interactively preview lighting effects.
- ***Tutorial 14: Global Illumination—The Ball Room***
  Define global illumination to add reflected light effects from a sphere within a closed "room."
- ***Tutorial 15: Final Gathering—Making an Alien Look Real***
  Simulate "real word" lighting without using reflected light or photons by setting up lights for use with final gathering, using a reflector to bounce light, and creating a radiosity map.
- ***Tutorial 16: Basic Lighting—Frank's Bust***
  Create, position, and edit lights to get the most out of your scene or object. Use the Spot Light view to edit your spotlight's cone and point of view.
- ***Tutorial 17: Light Effects—How Many Animators Does It Take to Change a Lightbulb?***
  Apply a lens flare, volumic effect, and glow to a flashlight. Use Inclusive/Exclusive lighting to exclude the table from the light.
- ***Tutorial 18: The Render Tree—Texturing, with a Twist of Lemon***
  Mix materials, apply a texture, and create a displacement map on an object.
- ***Tutorial 19: The Texture Editor—Dressing Up the Boombox***
  Map textures to selected polygons. By superimposing your object's wireframe UV coordinates onto a texture, manipulate any part of your object to fit onto any part of the texture.
- ***Tutorial 20: Creating Render Passes***
  Use a static scene to create a matte pass, a shadow pass, a highlight pass, and a depth pass.
- ***Tutorial 21: Scripting—Repeating Commands***
  Build a simple script to repeat frequently used commands.
- ***Tutorial 22: Scripts for Automating Tasks***
  Use a simple for loop to apply values to selected objects.
- ***Tutorial 23: Building Custom Tools***
  Use subprocedures and variables to build your own custom tool to match object positions.

Comments?

If you have anything you'd like to say or ask about the SOFTIMAGE|XSI documentation, please do not hesitate to e-mail us at **editors@softimage.com**.

Where to Find Information

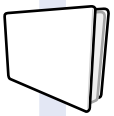
The SOFTIMAGE|XSI package includes a comprehensive set of learning materials. Use this Roadmap to find the information you need to get up and running quickly and effectively.



Start with the ***Setup Guide*** to install and license all components. **Setup Online Help** is also available as you install—just press F1.

Refer to **Limitations & Workarounds** (formerly *Release Notes*) found for this version. Visit the web at **softimage.com > support > SOFTIMAGE|XSI**.

Take the ***New Features Tour***—a set of videoclips giving overviews of the features and tools new to this version. If you installed the Tour with the XSI software, choose **Help > New Features Tour** from the main-menu bar. Also available from the Online Library CD.

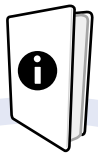


Work through ***Tutorials*** to learn the features in the context of basic productions. This is a full-color set of lessons showing you how to perform typical tasks step-by-step. You can install the scenes from the Media CD that comes with the *Tutorials* guide.

The Softimage Discussion Group

You can join the worldwide network of Softimage users exchanging ideas and techniques by e-mail. To find out more, e-mail **majordomo@softimage.com**. Leave the Subject line empty and type the word "help" in the body of your mail message.

Global Index & Glossary
An index to all user guides and *Tutorials*; a glossary of terms; and a list of books, videos, and web sites related to the 3D animation industry.



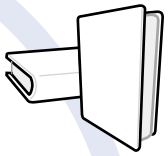
Online Library CD
Contains:

- XSI and mental ray® documentation in both PDF and HTML formats.
- New Features Tour

See the next page for how to use the Online Library CD.

User Guides
Conceptual information and procedures on how to use the XSI toolsets:

- *Fundamentals*
- *Animating*
- *Modeling & Deformations*
- *Shaders, Lights & Cameras*
- *Rendering*



Online Help
On-screen reference information on interface elements, commands, and parameters.

How to access:

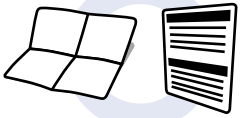
- Click on the **?** button in any property editor or tool view that dislplays this symbol.
- Choose **Help > Contents and Index** from the main-menu bar.

What's New in 1.5
Summarizes all the new features since the last version of SOFTIMAGE|XSI.

HTML Scripting Reference
An HTML-based reference on the syntax for all scripting commands and arguments. It appears in your default HTML browser.

To access, first click on the **!** to open the script editor, then choose **Help > Scripting Reference** or click on **?**

The ***SOFTIMAGE|XSI Interface Layout*** and the ***Quick Reference Card*** will help you become familiar with the interface and keyboard shortcuts.



Using the Online Library

The Online Library contains the SOFTIMAGE|XSI and mental ray documentation in both PDF and HTML electronic formats. You can also browse the New Features Tour from this CD.

To access the Online Library

1. Insert the Online Library CD in your disk drive.
2. Open one of the following documents:
   - **mainmenu.pdf** (PDF format)
   - **mainmenu.htm** (HTML format)

For full-text searching and printing, we recommend PDF format. If you do not have Acrobat Reader installed on your system, you can install it free of charge from the Online Library CD.

Windows NT users can also run Acrobat Reader 4.05 directly from this CD. Path: **/acrobat/win/Run_Reader_from_CD.bat** This batch file runs Acrobat Reader installed on the CD and opens mainmenu.pdf. From here you can access all documents in the Online Library.

Document Conventions

The following are ways that information is displayed in the SOFTIMAGE|XSI documentation.

Typography Conventions

Type style	Usage
Bold	Menu commands, dialog-box and property-editor options, and file and directory names.
Italics	Definitions and emphasized words.
`Courier`	Text that you must type exactly as it appears. For example, if you are asked to type `mkdir style`, you would type these characters and the spacing between words exactly as they are appear in this book.
>	The arrow (>) indicates menu commands (and subcommands) in the order that you choose them: *Menu name* > *Command name*. For example, when you see **File** > **Open**, it means to open the **File** menu and then choose the **Open** command.

Visual Identifiers

These icons help identify certain types of information:

Notes are used for information that is an aside to the text. Notes are reminders or contain important information.

Tips are useful tidbits of information, workarounds, and shortcuts that you might find helpful in a particular situation.

The 3D icon indicates information about differences in workflow or concepts between SOFTIMAGE|3D and SOFTIMAGE|XSI. You will find these very helpful when working with the two products.

Warnings are used when you can lose or damage information, such as deleting data or not being able to easily undo an action. Warnings always appear *before* you are about to do such a task!

Keyboard and Mouse Conventions

SOFTIMAGE|XSI uses a three-button mouse for most operations. These are referred to as the *left*, *middle*, and *right* mouse buttons. In many cases, you will use the different buttons to perform different operations; always use the left mouse button unless otherwise stated.

The two-button mouse is not supported in SOFTIMAGE|XSI.

This table shows the terms relating to the mouse and keyboard.

When this term is used...	...it means this
Click	Quickly press and release the left mouse button. Always use the left mouse button unless otherwise stated.
Middle-click	Quickly press and release the middle mouse button of a three-button mouse.
Right-click	Quickly press and release the right mouse button.
Double-click	Quickly click the left mouse button twice.
Shift+click, Ctrl+click, Alt+click	Hold down the Shift, Ctrl, or Alt key as you click a mouse button.
Drag	Hold down the left mouse button as you move the mouse.
Alt+*key,* Ctrl+*key,* Shift+*key*	Hold down the first key as you press the second key. For example, "Press Alt+Enter" means to hold down the Alt key as you press the Enter key.

Softimage Customer Service

Technical support for SOFTIMAGE|XSI is provided by Softimage resellers working together with Softimage Customer Service. Immediate assistance for any technical issue is available through hotline, e-mail, and web support services.

Licensing Support

You must contact your reseller to request a license for SOFTIMAGE|XSI. This can be done either directly or through the license request form provided at **softimage.com/licensing**

Training Support

If you're interested in SOFTIMAGE|XSI training, you'll find a complete overview of courses, education centers, and training programs at **softimage.com/education**

Hotline Support

If you've purchased a maintenance contract to receive support directly from your Softimage reseller, you'll find assistance for contacting your reseller at **softimage.com/Products/3D/WheretoBuy.htm** In all other cases, contact Softimage Customer Service at the following numbers and during these hours:

World Wide Tel: 1 (514) 845-2199 Fax: 1 (514) 845-8252	Hours 9 AM to 9 PM (EST) 2 PM to 2 AM (GMT) 1400 to 0200 (UTC)
North America Tel: 1 800 387 2559 Fax: 1 (514) 845-8252	Hours 9 AM to 9 PM (EST) 2 PM to 2 AM (GMT) 1400 to 0200 (UTC)
UK and International Tel: + 44 1 753 650 670 Fax: + 44 1 753 658 503	Hours 9 AM to 6 PM (GMT) 0900 to 1800 (UTC) 2 AM to 1 PM (EST)

E-mail Support

You can e-mail Softimage Customer Service at any at **support@softimage.com**

Mailing Lists

If you have an e-mail account, you can join the worldwide network of SOFTIMAGE|XSI users exchanging ideas. To subscribe to the XSI discussion group, send an e-mail to majordomo@softimage.com with subscribe XSI as the body of your message. Use the message lists for information on related groups. Visit **softimage.com/Archives/Help.htm** for more information about our discussion groups and mail server.

The discussion groups are provided for technical exchanges between customers. Although Softimage Customer Service is not provided through these discussions, we do contribute.

Web Support

The Support and Download sections of softimage.com provide quick access to a wide range of resources from the SOFTIMAGE|XSI teams and user community. Downloads—including Presets, Scripts, and Quick Fix Engineering (QFEs)—provide the latest solutions for XSI. Online user guides, tutorials, and knowledge-base articles ensure you get the most out of working with XSI. It's like having a dedicated Softimage Customer Service engineer sitting at your desk!

Softimage Customer Service Addresses

North America

Softimage Customer Service
3510 Saint-Laurent Boulevard
Montreal, Quebec
H2X 2V2 Canada

UK and International

Softimage Customer Service Europe
Pinewood Studios
Pinewood Road
Iver Heath, Buckinghamshire
SL0 0NH United Kingdom

Section 1 Getting Started

Installing the Tutorial Project

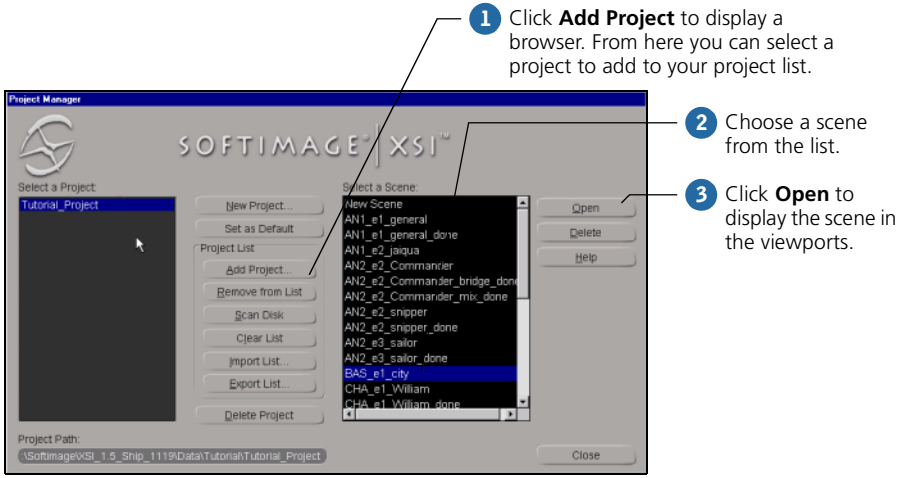
The tutorials in this workbook use scene content that must be installed from the Media CD. By default, the tutorial scenes are installed in your **\Softimage\XSI_1.5\Data\Tutorial\tutorial_project** folder.

The next section shows you how to add the tutorial project to your project list so you can access its scenes.

Adding a Project to the Project Manager

The first time you run SOFTIMAGE|XSI, the Project Manager appears, which gives you access to all projects and scenes. You will add a project to the Project Manager's project list so that you can then select its scenes and use them with the tutorials in this workbook.

1. In the Project Manager, click **Add Project**.
2. Navigate to **XSI_1.5\Data\Tutorial** in your Softimage directory, select **tutorial_project**, and click **Select**. tutorial_project is added to your list of projects.
3. Select a scene from the list of project scenes (on the right side) and click **Open**.
4. To add another project to the Project Manager, choose **File** > **Project Manager** from the main-menu bar at the top of the window.
5. To open another SOFTIMAGE|XSI scene, choose **File** > **Open**.



The Interface

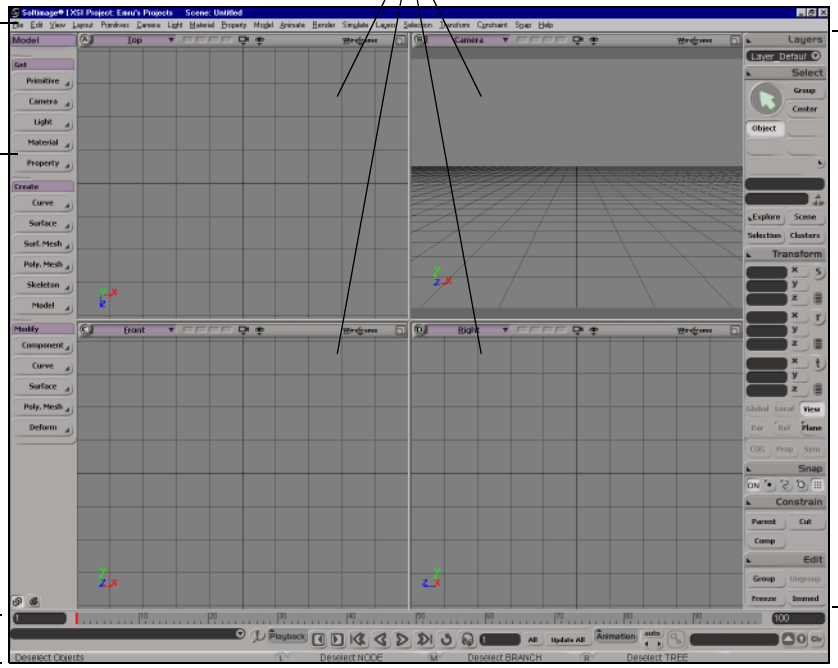
Below is the default layout. As you familiarize yourself with the interface, you can customize it to improve your workflow. For more information about the interface, please see the *INTERFACE LAYOUT*, *Chapter 2: The Interface* in the *Fundamentals* guide, and Online Help, accessible at any time by choosing **Help** > **Contents and Index** from the main-menu bar.

Viewports let you view the contents in your scene in many different ways. You can resize, hide, and mute viewports in any combination.

Press F12 to display the viewport under the pointer at full screen. Press F12 again to revert to a four-viewport display.

The **main-menu bar** provides access to all the primary commands.

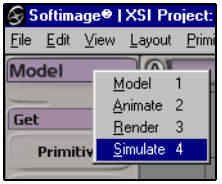
The **toolbar** displays one of four toolsets: Model, Animate, Render, or Simulate. Click the toolbar label or press 1, 2, 3, or 4 on the keyboard to switch between toolbars.



The **lower interface** is where you can create and activate scripts, as well as edit animation and play it back.

The **main command area** is the default location of the Layers, Select, Transform, Snap, and Constrain panels.

Interaction Tools



SOFTIMAGE|XSI provides a variety of tools to simplify the process of interacting with a 3D world on your 2D screen.

Navigate between modules

You can switch between the **Model**, **Animate**, **Render** and **Simulation** toolbars by clicking the top of the toolbar and choosing from the menu that opens, or by pressing the **1**, **2**, **3**, or **4** keys, respectively. Use **Alt+1** and **Alt+2** to switch between the default toolbars and the palette toolbar.

Use shortcut keys to view the scene

Many tools can be activated by means of shortcut keys that replace "point-and-click" menu selection. For example, the **z** key activates the zoom tool. Shortcut keys can be used in temporary or "sticky" mode. If you press and release a key quickly, the tool stays activated ("sticky"). There's no need to keep the **z** key pressed while you're zooming in on an object. The mouse pointer changes to tell you which tool you've activated (in the case of the zoom tool, it becomes a magnifying-glass icon). If, however, you leave your finger on the key while using the tool, the tool deactivates when you release it (temporary or supra mode). You can cancel a tool at any time by pressing the same key again, pressing another key to activate another tool, or pressing the Esc key.

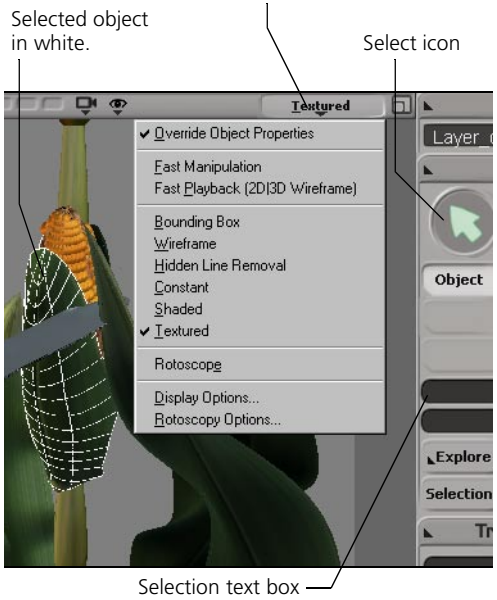
Keyboard shortcut information can be found by choosing **File** > **Keyboard Mapping** from the main-menu bar.

Here are a few of the most common shortcuts you'll use:

- The **f** supra key frames the selected object(s) in the current viewport, while **Shift+f** frames the selected object(s) in all viewports. The **a** supra key frames all objects in the current viewport and **Shift+a** frames all objects in all viewports.
- **Ctrl+z** undoes the last change, and **Ctrl+y** redoes it.

To navigate in a scene, move the mouse pointer into any viewport, then do the following:

- To orbit the scene, press the **o** key and click and drag the mouse. Note that you can only orbit within the Camera, User, or Spot views.
- To pan the scene, press the **z** key and click+drag the mouse.
- To zoom into the scene, press the **z** key and middle-click+drag the mouse. To zoom out, right-click+drag the mouse.
- To dolly in the scene, press the **p** key and click+drag the mouse. Note that you can only dolly within a Camera view.



Section 2 **Basics**

Tutorial 1: Editing Properties

In SOFTIMAGE|XSI it's all about properties. Every object in your scene is distinguished by its own set of properties, which in turn are defined by a series of parameter values. When you edit and manipulate the objects in your scene, you will almost always use property editors to define these values. Using a simple primitive, this tutorial will quickly take you through the basics of editing object properties.

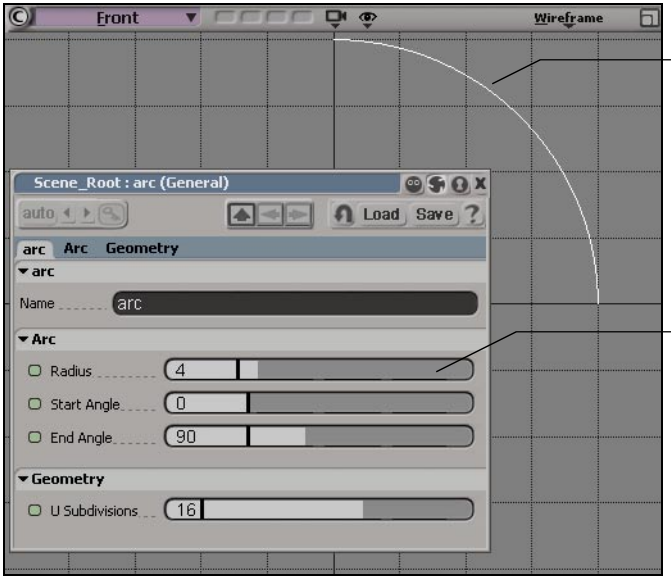
This tutorial shows you how to:

- Edit the primary properties of an object from its property editor.
- View the object's properties from a pop-up explorer list.
- Apply Move and Add Point operators to an object in a viewport.
- Edit an object's operator stack.

Editing from a Property Editor

1. Start this tutorial with a new scene. Press **Ctrl+n**, or choose **File** > **New Scene** from the menu bar at the top of the main window.
2. From the Model toolbar, choose **Get** > **Primitive** > **Curve** > **Arc**.
3. An arc is created and its property editor opens. Here, you can edit the primary properties of the arc. You can, for example, modify the arc's **Start Angle** and **End Angle** parameters by dragging their sliders.

You can a close property editor by clicking on the × in its upper right corner or by pressing **Ctrl+`** (above the **Tab** key on most keyboards).



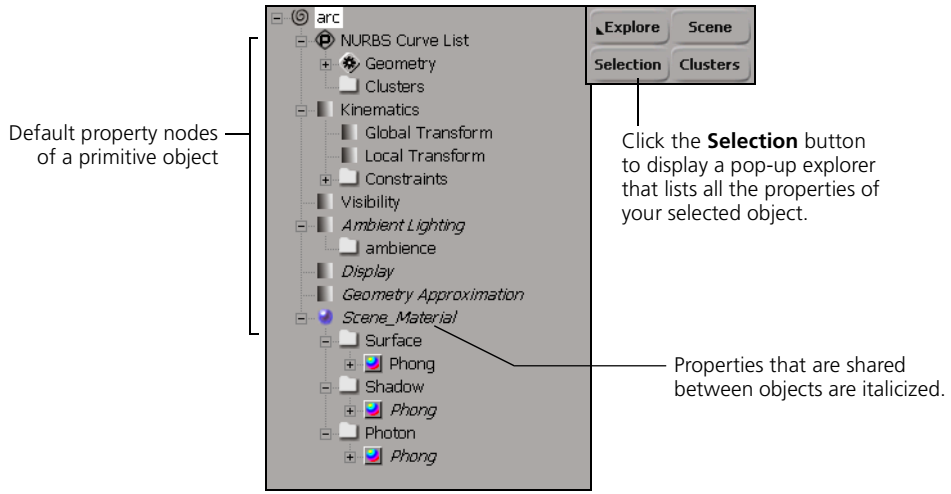
Press **f** to frame the selection in the viewport.

Press **s** in camera view to orbit (left mouse button), dolly (middle mouse button), and track (right mouse button).

Edit the parameters and get immediate visual feedback in the viewports.

Viewing an Object's Property Nodes

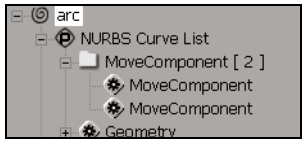
To view the object's properties, select the arc then click the **Selection** button in the Select panel on the main command area.

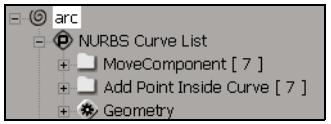


Editing an Object's Operator Stack

The operator stack is fundamental to modeling in SOFTIMAGE|XSI. It is the history of all the operators (such as move point) that have been applied to an object. At any time you can go back and delete the operators.

4. Press and release the **m** key then drag to move a point on the arc from within the Camera view. Since you are in sticky mode, you do not need to hold down the key while you are using the move point tool.
5. Move another point on the arc.
6. Click the **Selection** button again to display the arc's properties. Note that a MoveComponent folder was added under the object's primitive node. The operator stack always appears under the first subnode of an object in the explorer (in this case, the subnode named NURBS Curve List).
7. Expand the MoveComponent folder (by clicking the + sign) to show the two MoveComponent operator nodes.





8. Select one of the MoveComponent nodes and press the Delete key on the keyboard. You will see that the corresponding "move point" operation was deleted from the arc curve.
9. Return to the arc and move a few more points.
10. From the model toolbar, choose **Modify** > **Curve** > **Add Point** to activate the add point tool. Position the cursor between two points on the arc and middle-click to add a point.
11. Press and hold the **m** key as you move a point. When you release the key you will switch back to the previously active add point tool. In this case you are using the move point tool in temporary mode.
12. Click the **Selection** button to view the curve's construction history.

Freezing the Operator Stack

When you are satisfied with an object, you can freeze its operator stack. This removes its construction history—you can no longer go back and change values. However, the object requires less memory and is quicker to update.

13. Select the arc: press the spacebar (if you are still in Add Point mode) to activate the selection tool, then click and drag to select the arc.
14. Choose **Edit** > **Freeze Operator Stack** or click the **Freeze** button on the Edit panel.
15. Click the **Selection** button again and note that the MoveComponent folder has been removed from under the arc's NURBS Curve List.

Conclusion

By editing this simple object, you have been introduced to a few basic but important concepts in XSI: nearly everything in XSI is an object defined by properties that can be edited (and even animated) from a property editor or directly by manipulating it within the 3D views. As you edit an object, a history of the operations you apply is created—the operator stack. You can return and edit or delete these operators as needed. When you are satified with your edits you can freeze the operator stack.

For more information, see *Chapter 5: Working with Scene Elements* in the *Fundamentals* guide.

Tutorial 2: The Interface

Optimization is crucial to the success of a 3D production. Knowing your shortcut keys, working with layers and visibility controls, and ordering your scene elements into manageable hierarchies are all starting points to achieving that goal.

As you familiarize yourself with the interface, you can customize it to improve your workflow.

This tutorial shows you how to:

- Optimize the display of your scene with layer and visibility controls.
- Use selection tools (rectangle, lasso, freeform), filters, and the selection text box for different ways of selecting scene elements.
- Create hierarchical relationships between objects through parenting.
- Create a custom toolbar for some color presets.
- Create groups to share properties.
- Use views to see the scene from different viewpoints.
- Use 3D manipulators to interactively manipulate lights and cameras.
- Use a render region to interactively preview your scene.
- Transform and duplicate objects.
- Edit properties using random and relative values.

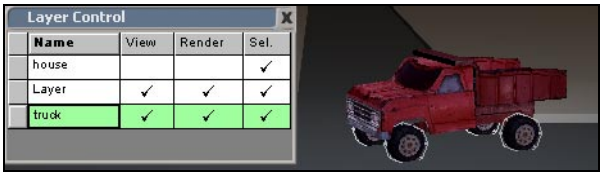
Controlling Layers in a Scene

Layers provide you with a way of dividing and managing your scenes. The layer control options allow you to turn off selectability (**Sel.**) — an easy way to keep a layer visible but not selectable. You'll find this especially useful in crowded scenes. You can also deactivate View and Render visibility. Since XSI does not compute hidden geometry, you really do save in interaction with large scenes.

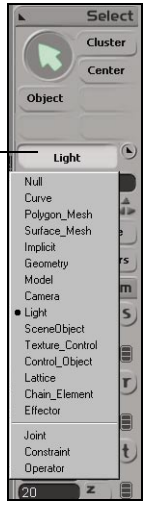
The scene you will load is already organized in layers, but the house layer is hidden.

To unhide the house layer from the Layer Control window

1. Open the **BAS_e1_city** scene from the Tutorials project.
2. Choose **Layers** > **Layer Control** from the Layers panel in the main command area.
3. In the Layer Control box, toggle on both **View** and **Render** for the house layer and close the box.



Setting Visibility Options



Another way of making the scene a little lighter is by choosing which elements are visible in the viewports, regardless of layers.

Choose **View** > **Visibility Options** (**All Cameras**) from the main menu bar or press **Ctrl**+**Shift**+**s** to open the Visibility Options property editor. Deselect **Cameras** and **Nulls** in the Objects tab to hide all cameras and nulls in your scene in all viewports.

If you want to affect only one viewport, click the eye icon on the viewport's menu bar (or press **Shift**+**s**), and choose the options you want to show/hide.

Selecting Objects

As you work, you constantly select and manipulate objects and their components. SOFTIMAGE|XSI provides a number of tools you can use to select objects. This section will describe some of these tools:

- To select an object, simply click and drag the mouse pointer over it. This uses the default rectangle selection tool.
- To deselect all objects, click and drag the pointer on an empty area of the viewport.

- To remove an object from an existing selection, press the Ctrl key (toggle) and select the object.
- To add an object to the selection, press the Shift key while clicking the object.

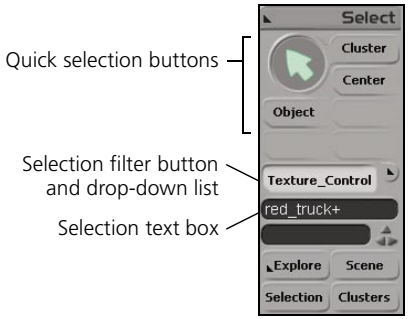
The extended component selection option allows you to remain in the default XSI selection mode while taking advantage of the three mouse button selection method when selecting components (points, polygons, etc.). Use the left mouse button to select components, the middle mouse button to deselect, and the right mouse button to toggle selection.

Selecting Objects with Pre-defined Filters

Selection filters can be extremly useful for narrowing down thea typ of target to use for selection. For example, with the Texture_Control filter, only textures can be selected:

4. From the Select panel, choose the **Texture_Control** filter, then drag a rectangle over the truck object in the scene. Note how only the texture controls are selected.
5. Click the **Object** button to return to object selection mode.

Selecting Objects by Name



The Selection text box lets you select elements using their names. This method lets you use wild cards such as asterisks (*), which are particularly helpful for selecting multiple items. Try to select the elements specified below using only the Selection text box:

6. Select the cabin section of the red truck.

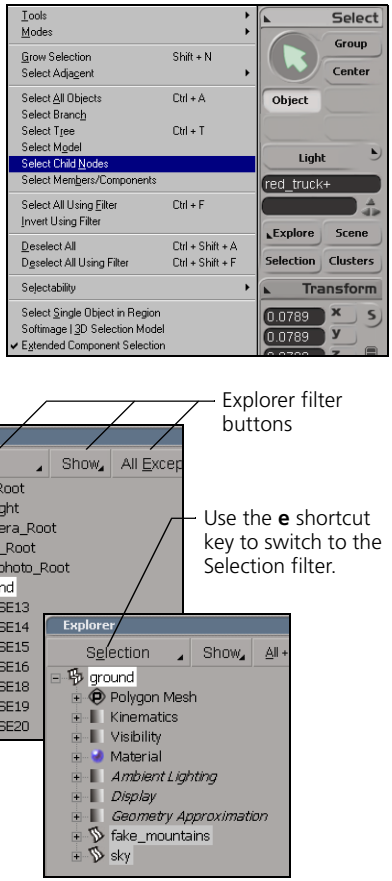
   Type `*cabin` in the Selection text box and press Enter. This selects one specific object in the scene. In this case, you needed to add an asterisk in front of the name because the object is under another model (red_truck) in the hierarchy.

7. Select all elements in the scene that begin with the word window.

   Type `window*` in the Selection text box and press Enter.

8. Select the truck in branch mode.

   Type red_truck+ in the Selection text box and press Enter. A plus sign after the name selects every object under the selected node. In the explorer view, a branch-selected object is highlighted in white and the nodes underneath are highlighted in grey.



Multi-selection

Choose **Select** > **Select Child Nodes** from the Select panel. This changes the nature of your selection: the tree is no longer selected, but rather each item in the tree is now selected individually, which allows multi-selection editing. All elements are now highlighted in white.

Group Selection

Groups are used for organization as well as to override and share properties. You can select the members of a group by growing the selection. For example, in a scene explorer select the first_block_group node or type `first*` in the selection text box (a group is selected in branch by default). Choose **Select** > **Select Members/Components** from the Select panel to select all of a group's members individually.

Selecting and Filtering in the Explorer

The explorer displays the contents of your scene in a hierarchical structure which shows all objects in a scene as well as their properties.

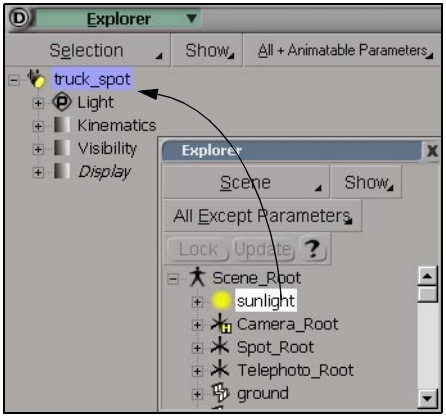
9. Select the ground in branch mode in the explorer by middle-clicking the ground node (or type `*ground+` in the Selection text box).
10. Open a scene explorer and choose **Selection** from the first filter button in the scene explorer to isolate the information specific to your selection (you can also use the **e** shortcut key).
11. You can further filter the information using the other two filter menu buttons. For example, choose **Objects Only** from the All+Animatable Parameters menu button.

Parenting Objects Using the Explorer

Organizing your scene's hierarchy from the explorer is as easy as dragging and dropping one node onto another.

12. From the explorer, choose the **Scene** filter.
13. In the explorer, click the **red_truck** model node to highlight it. Use the right arrow to expand the node, then the down arrow to navigate to the **cabin** node. Expand the cabin node.
14. Note that a wheel has not been parented properly (all wheels should be under the **wheels** null).
15. To correct this, drag-and-drop the **wheel** node onto the **wheels** node to make the correction .

Drag and drop to create a hierarchy



Parenting Lights

In SOFTIMAGE|XSI, you can parent a light to an object:

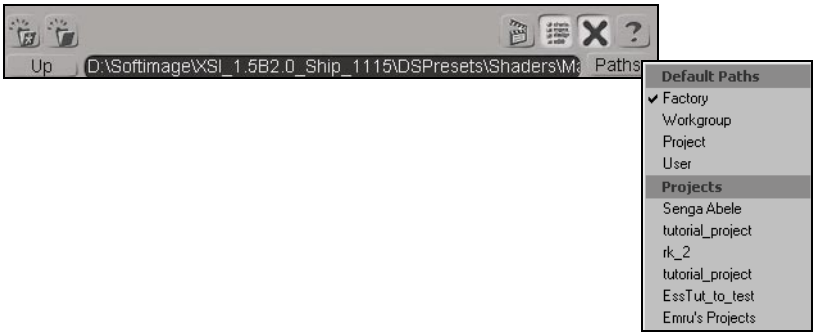
16. In the main command area, enter `sunlight` in the Selection text box. This selects the scene's infinite light.
17. Position your pointer in the explorer, then press the **f** key to focus on ("frame") the infinite light object.
18. Open a second explorer by choosing **View** > **Views** > **Explorer** from the main-menu bar.
19. Enter `truck_spot` in the Selection text box.
20. In the second explorer, filter the view to the selection by clicking the **e** key.
21. Lock this explorer by clicking the **Lock** button. This keeps the selection active even when you select another object.
22. Drag-and-drop the sunlight node from the docked explorer onto the truck_spot node in the locked explorer to parent the infinite "sunlight" to the truck spotlight.

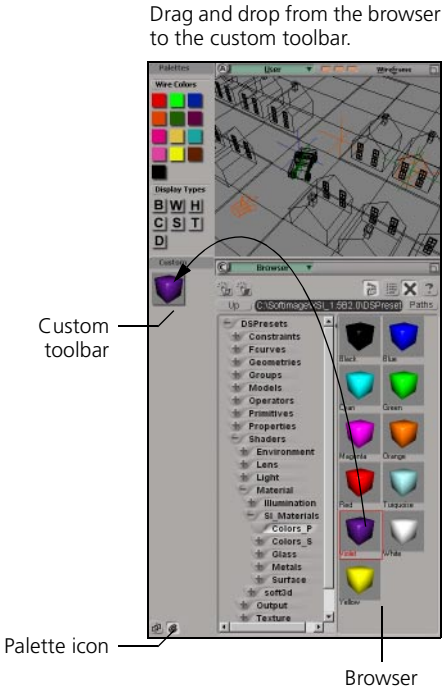
    Et voilà! Your parenting responsibilities are now over.

Adding Toolbars to Your Layout

You will create two new toolbars to house a set of color presets and then edit your layout so that these toolbars are a permanent part of the layout. You can actually place your custom toolbars anywhere you want; placing them within the Palette toolbar is just a convenient location because this toolbar provides a default panel for docking custom toolbars.

23. Choose **View** > **Custom Toolbars**> **New Toolbar** from the main-menu bar. In the dialog enter the name `color presets` and click OK. An empty toolbar appears.
24. Create a second toolbar called `more colors` and click OK.
25. Open a browser by pressing the **5** key. Click the **Paths** button, and choose **Factory**. Navigate to the **DSPresets\Shaders\Material\SI_Materials\Colors_P** directory.



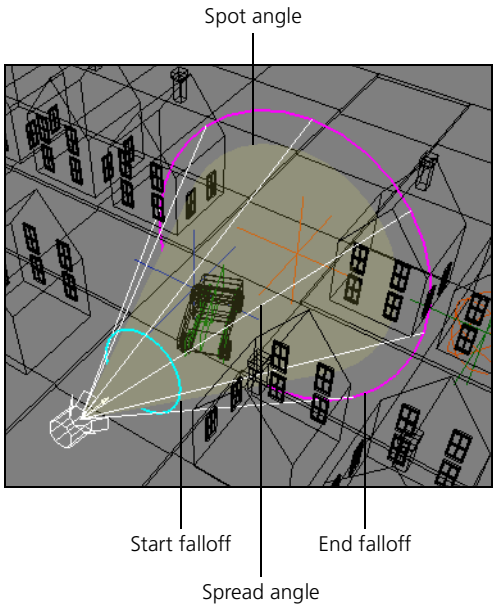


26. Drag and drop the Blue, Green and Turquoise color presets onto the **color presets** toolbar.
27. Navigate to the Colors_S directory and drag and drop the Dark_Brown, Deep_Orange and Dirty_Pink presets on to the **more colors** toolbar.
28. Close the browser and the new toolbars (this is necessary to add the toolbars to the layout).
29. Choose **Layout** > **Edit Layout**. Notice that as you move the cursor, a thin red highlight appears around each area.
30. Click the Palette icon at the bottom of the toolbar to open the Palette toolbar.
31. Add the color presets toolbar to the default custom toolbar already available: right-click the Custom title bar and choose **Set View** > **Custom Toolbars** > **color presets**.
32. To add the second toolbar you will have to create a new panel: right-click in the empty space below the color presets toolbar and choose **Split Horizontally**. Position the split line where you want it by clicking. The area is split into two sections.
33. In the blank space that you have just created, right-click and choose **Set View** > **Custom Toolbars** > **more colors**. Your second toolbar is displayed.
34. Choose **Layout** > **Edit Layout** again to deactivate layout editing mode. Do not save the layout if you do not want it as a permanent part of the interface.

Creating a Group

Groups are a collection of objects that are grouped together, usually for the purpose of sharing some common attribute or operation, such as a texture or material. There is no hierarchy involved in a group as with parent-child structures.

35. Branch-select some of the houses by holding the Shift key while middle-clicking them.
36. Click **Group** in the Edit panel. Give the group a name.
37. Focus on the group in the explorer using the **e** shortcut key.
38. Change the B viewport's display mode to Shaded.
39. Drag-and-drop a color preset on the group node in the explorer.
40. Observe the result in the viewport B. Note that all the houses within the group change color.
41. Adjust the color if you wish, and close the property editor.
42. With the group still selected, press the Delete key; the houses revert to their original color. When you apply a property to a group it takes precedence over the original property but does not remove it.



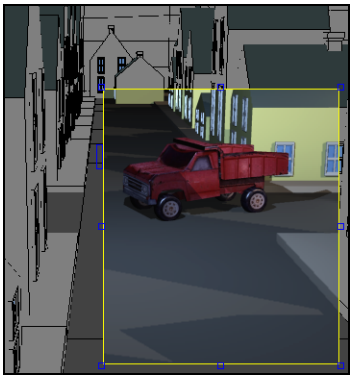
Using a Spotlight's View and Manipulators

You can edit a spotlight directly in a scene using 3D manipulators. You can also view your scene from a spotlight's point of view, which is useful for seeing the illumination effect a spotlight has on a scene.

43. In the Selection text box, type `truck_spot` and press Enter to select the spotlight.
44. Press **Shift+f** to frame the selection in all viewports and use the **s** multi-purpose navigation tool to position the camera as illustrated.
45. Activate the spotlight's 3D manipulators by choosing **Manipulate Tool** from the eye icon in a viewport menu or pressing **b**.
46. Adjust the spotlight by dragging the different parts of the manipulator: the beginning of the falloff area is the turquoise ring, the end of the falloff area is the pink ring, the spread is defined by the white lines, and the light cone is yellow.
47. In another viewport, select the spotlight's view by clicking the viewport's Views menu (its title) and choosing **Spot Lights** > **truck_spot**. This view shows the truck from the perspective of the truck_spot spotlight. Use the orbit (**o**) and dolly (**p**) shortcuts to move the spotlight around. You can also use the multipurpose navigation tool (**s**), which allows orbiting with the left-mouse button , dollying with the middle-mouse button, and panning with the right-mouse button.

Previewing in the Render Region

The render region lets you preview any part of your scene with an interactive renderer.



48. Press **q** and drag a rectangle over the truck area in a camera view. The rectangular area defines a render region which, by default, will be automatically updated whenever the scene changes.
49. In the Render toolbar (press the **3** key), choose **Render** > **Region**. The options here let you view different channels in the render region:
    - **Show RGB** renders the image.
    - **Show Alpha** renders the image's alpha channel, which is made visible.
    - **Show RGB + Alpha** renders an image and its alpha channel.
    - **Show Depth** renders a depth pass (also known as the Z channel).
    - **Show Tag** displays a different color for each object in the scene for easy processing of some parts of the images at compositing. This facilitates integration between 2D and 3D imagery.
50. **Render** > **Region** > **Auto Refresh** refreshes the render region every time you make a change in your scene. Turn this option off to save time, then choose the **Render** > **Region** > **Refresh** command only when you want to update the render region.

When the render region is not set to Auto Refresh, the region's outline is red instead of yellow. You can also access render region properties by right-clicking on the frame of the region box.

51. You can control the antialiasing level of the image by dragging the slider on the right side of the render region (a narrow vertical bar). Dragging the slider down results in faster rendering, but at the expense of an image with more aliasing.
52. To close the render region, press **q** and drag a little beside the region. Press Esc to exit the render region mode.

Duplicating an Object

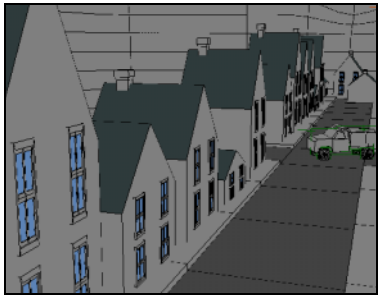
53. Select any house in branch mode, then press **Ctrl+d** to instantly duplicate the hierarchy.
54. Press **v** to enter translation mode, then position the newly duplicated house next to the original.
55. Press Ctrl+d to duplicate again: note how this other duplicate uses a position offset based on the last transformation.

56. With the new house branch-selected, press the **d** supra key then click anywhere in a viewport. A house is duplicated under the mouse pointer.
57. Press **Ctrl-z** (Undo) until all the new duplicated houses disappear.

When using **Edit** > **Duplicate/Instantiate** > **Duplicate or Duplicate Multiple**, the transform offset is taken from any transforms performed immediately between the first duplicate operation and the next. Choose **Edit** > **Duplicate/Instantiate** > **Duplicate/ Instantiate Options** to enter a different offset.

You can set the number of undos available by choosing **File** > **User Preferences** to open the User Preferences dialog box.

Adding Random Values



58. Select all of the houses by entering `house*+` in the Selection text box.
59. Press **x** to enter scaling mode. Make sure that you are in local mode (click the **Local** button on the Transform panel if it isn't selected).
60. Enter `r(0.2, 0.9)` in the scaling transformation Y axis text box. Press Enter. The houses are randomly scaled along the Y axis.

You can also enter linear values across multiple properties. For example, you could set the first house at 0.2, the last one at 0.9, and all the houses in between at regular intervals between the two by entering `l(0.2, 0.9)` in the scaling transformation Y axis text box.

- Entering `r` in a text box generates a random number.
- Entering `r(x)` generates a number between 0 and *x*. The *x* value can also be negative; for example, **r(–100)** will generate a value between –100 and 0.
- Entering `r(x, y)` generates a number between *x* and *y*, which can be in any order. For example, **r(–100, 100)** and **r(100, –100)** will both provide similar results—values between –100 and 100.

Once the random equation is entered, the property editor considers the result only. Therefore only the result, not the equation itself, can be edited.

Adding Relative Values

All the houses are now randomly scaled along the Y axis. You will notice that the random values cannot be displayed in the UV parameters because more than one value is applied. However, you can still increment each of these values.

Enter `0.3+` in the scaling transformation Y axis text box. Each selected house will now have its Y scaling value increased by 0.3.

You can apply any mathematical operation to a text box's original value by entering a number followed by the appropriate mathematical symbol. Use the asterisk (*) for multiplication and the slash (/) for division.

Conclusion

By manipulating the houses in the scene, you have familiarized yourself with the tools that will serve as the basis for all of your work in XSI. You have used selection tools to select scene elements in a variety of ways, depending on your needs, and groups and hierarchies to organize these elements and share properties among them. You have used the transform, select, and edit panels on the main command area to explore different ways to manipulate scene elements, and the viewports, visibility options, and render regions to control the many ways you can view your scene.

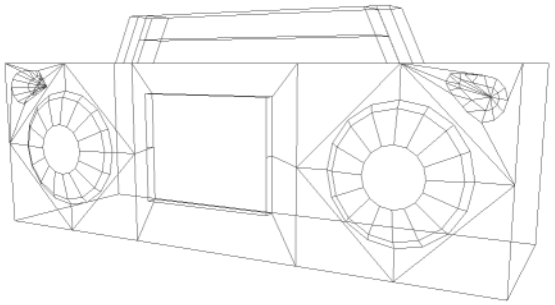
For more information, see the following chapters in the *Fundamentals* guide:

- *Chapter 2: The Interface*
- *Chapter 4: Viewing Your Work, Chapter 5: Working with Scene Elements*
- *Chapter 6: Working in 3D Space*

Section 3 **Modeling & Deformations**

Tutorial 3: Polygonal Modeling—A Boombox for Burt

In this tutorial, you'll build a boombox and familiarize yourself with the polygon modeling tools. And who's Burt? You don't really want to know, but you'll meet him in the next tutorial.

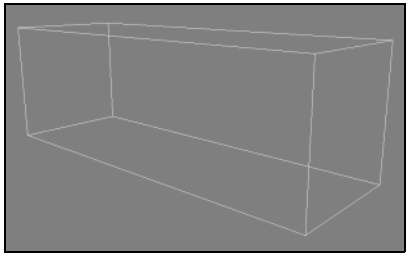


This tutorial shows you how to:

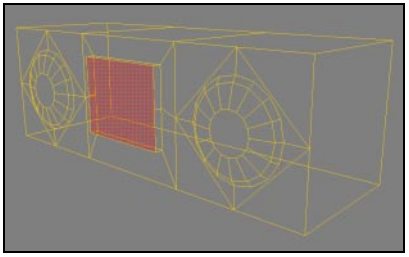
- Select various types of component: polygons, edges, and points.
- Subdivide polygons and edges.
- Duplicate, translate, and scale polygons.
- Snap elements to other elements.
- Modify polygon meshes with Boolean operations.
- Join polygons with the Bridge Polygon command.

Overview

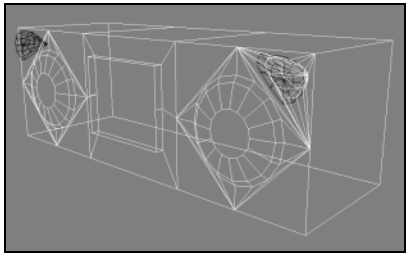
1 **Start with a scaled cube.**



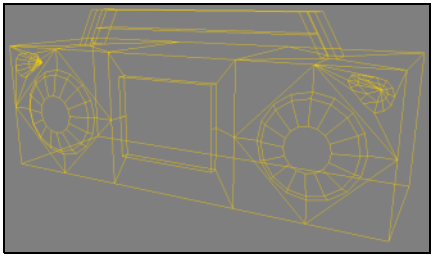
2 **Subdivide, duplicate, scale, and translate polygons to form speakers and a front panel.**



3 **Use Boolean operations to create tweeters.**



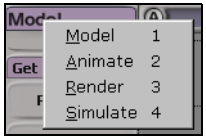
4 **Build a handle using the Bridge Polygons command.**



This tutorial assumes that you are using the SOFTIMAGE|XSI selection model. From the Select menu in the main command area, make sure that both **SOFTIMAGE|3D Selection Model** and **Extended Component Selection** are off.

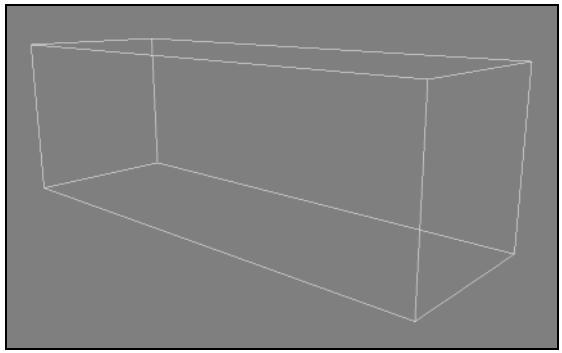
Making a Primitive Box

1. Start this tutorial with a new scene. Press **Ctrl+n**, or choose **File** > **New Scene** from the menu bar at the top of the main window.



2. Make sure the Model toolbar is displayed on the left of the main window. If it isn't, press the **1** key above the letters (not on the numeric keypad) or click on the toolbar's title and choose Model (left).
3. Get a primitive polygon mesh cube by choosing **Get** > **Primitive** > **Polygon Mesh** > **Cube**. A property page opens—leave all the settings at their default values.
4. Scale the cube three times longer in the X direction.

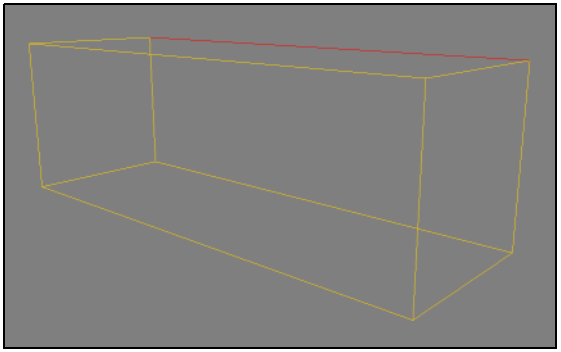
   - One way to do this is to press the **x** key to activate the Scale tool (left), then click and drag with the left mouse button to scale in X.
   - Another way is to is to use the boxes on the Transform panel on the right of the main window.
   - You can also highlight the numeric input in the x scaling text box and middle-click and "scrub" in a circular motion around it. This will increase the value if you scrub clockwise, or decrease it if you scrub counterclockwise. Use the Ctrl modifier key to increment by 10, or the Alt modifier to increment by 0.1.
   - To view your work in a camera view, use the **z** key to pan and zoom, the **o** key to orbit, the **p** key to dolly, or the **s** key to navigate (orbit/dolly/track). As with many tools, you can activate them in sticky mode by pressing and releasing the key, or in supra mode by holding the key while you drag the mouse. When you release the key in supra mode, the last tool that you used is still active.
   - Use the **f** key to frame the selected element; for example, the selected vertices, polygons, or objects. Framing a selection moves the camera interest to the center of the selection so that you can orbit or zoom. Use the **a** key to frame all elements in the scene.



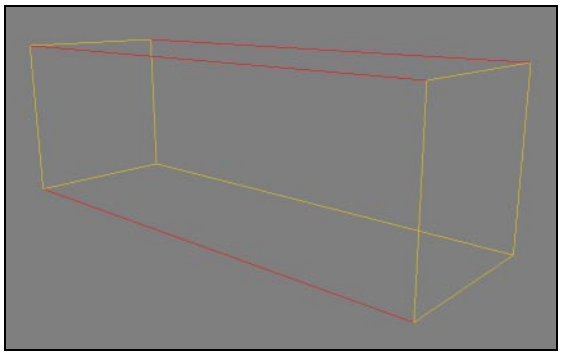
Subdividing the Front and Top

The next steps are to subdivide the front and top polygons. There are several possible ways to accomplish this: Add Polygon, Add Edge, Split Edge, Subdivide Polygon. However, in this case, you'll use the Subdivide Edge operator so that you can subdivide evenly.

5. Switch to edge selection mode.
   - One way to do this is to click the Edge button on the Select panel (left).
   - Another way to do this is simply press and release the **e** key to select edges with the Rectangle tool, or the **i** key to select edges with the Raycast tool.
6. Select the top back edge by clicking on it and dragging slightly; it turns red to indicate that it is selected.

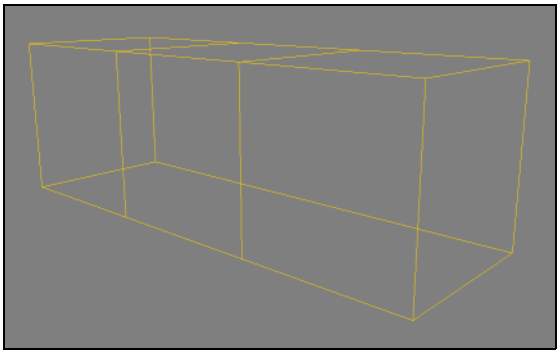


7. Add the top and bottom front edges to the selection by pressing the Shift key while clicking and dragging on them.



8. Subdivide the selected edges by choosing **Modify** > **Poly. Mesh** > **Subdivide Polygons/Edges**. The SubdivideEdge Op property editor opens.

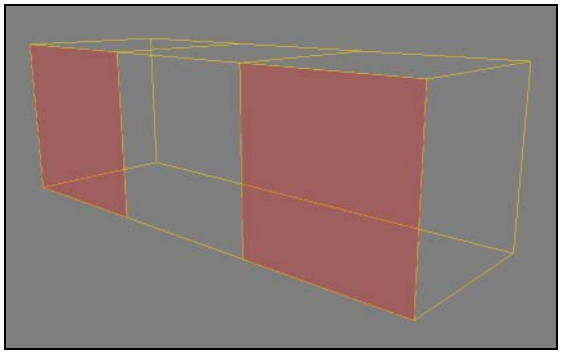
   - Set **Subdivisions** to **3**. The new front polygons will form the basis of the two speakers and the deck in the middle. Later you'll use the top polygons when you create the handle.
   - Leave **Parallel Edge Loop** at its default setting of **off**. When on, this also subdivides all parallel edges—in this case, you only want to subdivide the top and front.
   - Leave **Connect** at its default **on** setting. This adds new edges connecting the new vertices.



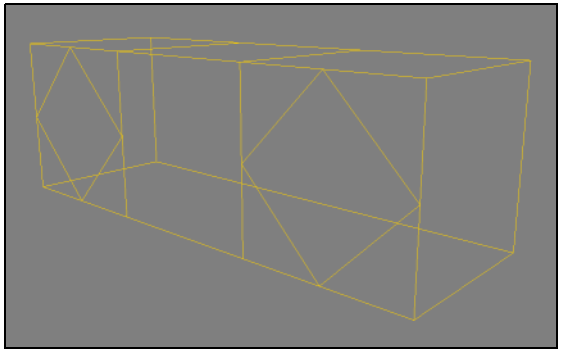
Making the Speakers

Next, create speakers by subdividing the front polygons further.

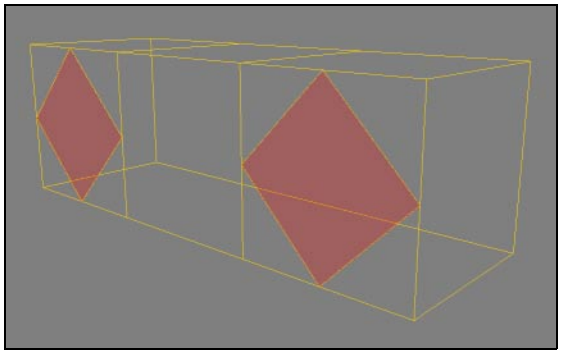
9. Activate raycast polygon selection in sticky mode by pressing and releasing the **u** key quickly.
10. Select the front right polygon by clicking on it; it turns red.
11. Add the front left polygon to the selection by holding the Shift key while clicking on it.



12. Subdivide the selected polygons by choosing **Modify** > **Poly. Mesh** > **Subdivide Polygons/Edges** or by pressing Shift+d. The SubdividePolygon Op property editor opens.
13. Change the **Subdivision Type** to **Diamond**, and leave the other parameters at their default settings.



14. Select the two central polygons.



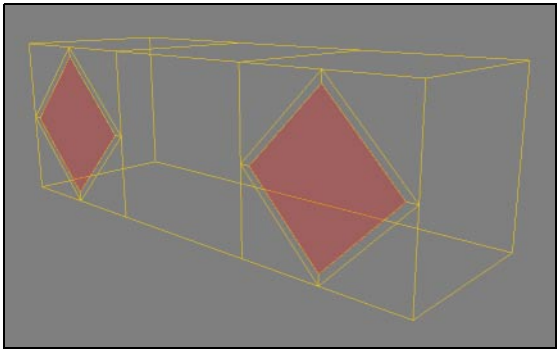
15. Duplicate the selected polygons by pressing Ctrl+d.

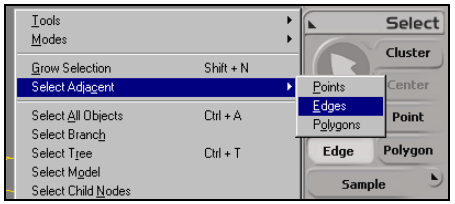
    - Accept the default value and click OK to close the editor.

    - Duplicating polygons is actually a special type of extrusion. You could also have chosen **Edit** > **Duplicate/Instantiate** > **Duplicate Single** or **Modify** > **Poly. Mesh** > **Extrude along Axis** and modified the options in the property editor.

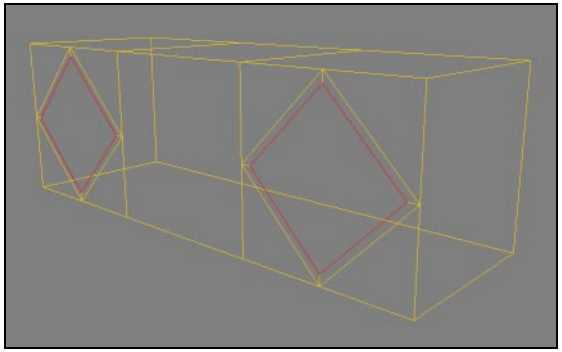
16. Scale the duplicated polygons down slightly.

    - To do this, press **x** to activate scaling, then press and hold the Shift key while dragging the mouse to scale uniformly in all axes.





17. Speakers are usually circular, not diamond-shaped, so you need to add more subdivisions to round them out. With the two polygons still selected, choose **Select** > **Select Adjacent** > **Edges** (left). This selects the eight inner edges (four on each polygon).



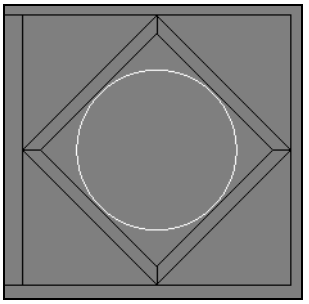
18. Subdivide the selected edges by choosing **Modify** > **Poly. Mesh** > **Subdivide Polygons/Edges** or pressing Shift+d again, and set **Subdivisions** to **4**. In the next section you'll use the extra vertices to snap to a circle.

Snapping

The snapping tools let you use other objects as guides when working. You can snap to all sorts of components: tagged points, points, mid-points, knots, curves, and more. First you'll get two circles to use as reference targets for snapping.

19. Get a primitive circle by choosing **Get** > **Primitive** > **Curve** > **Circle** and accepting the default settings. Note that it has 16 points, just like the inner polygons of the speakers.

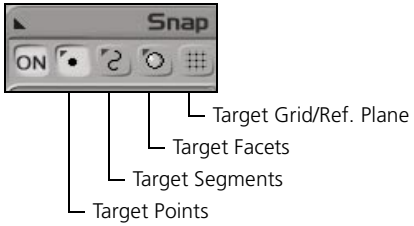
20. Scale and translate the circle so that it is superimposed on one of the speakers in the Front view.



21. Duplicate the circle, then translate it over the other speaker in the Front view.

If the primitive cube was scaled exactly by 3, you can use the Shift modifier key when translating the circle into position. The Shift modifier key translates in increments. The default increments are 1 SOFTIMAGE unit.

22. You'll be using the points of the circle as targets for snapping the points of the speaker, so first make sure that your snap options are properly set on the Snap panel (left):



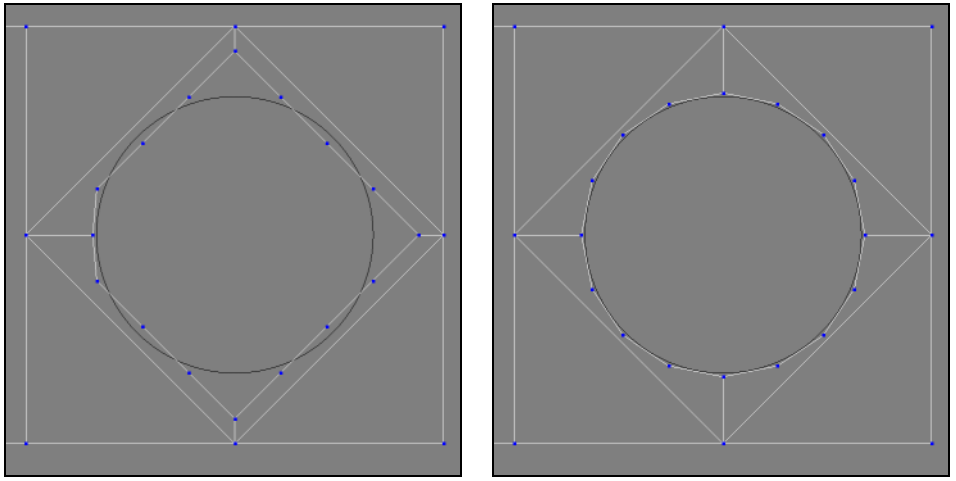
- Turn on snapping by activating the **ON** icon.
- Turn **Target Points** on and turn the other target types off.
- Choose **Snap** > **Target Points** and make sure that only **Points** is on.
- Make sure that either **Snap** > **All Objects** or **Snap** > **NURBS Curves** is on.
- Alternatively, you can choose **Snap** > **Snap Options** and make all these settings in the property editor.

23. Select the boombox.

24. Activate the move-point tool. You can do this either by pressing the **m** key or by choosing **Modify** > **Component** > **Move Point Tool**.

25. Make sure that the **View** transformation mode is active (left).

26. In the Front view, drag one of the speaker's vertices onto one of the circle's points. Repeat for the rest of that speaker's vertices.



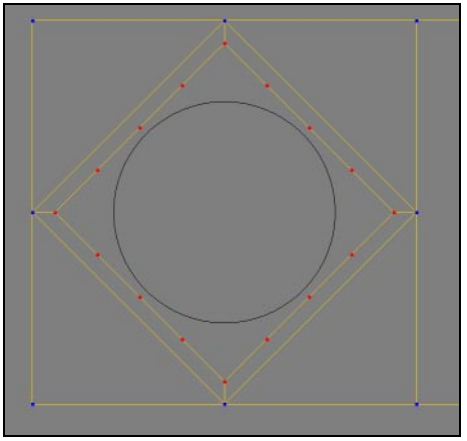
Shrinkwrapping

You could use snapping for the other speaker, but in this tutorial you'll explore an alternative method using the shrinkwrap deformation. Shrinkwrap deforms geometry by projecting it onto a target object.

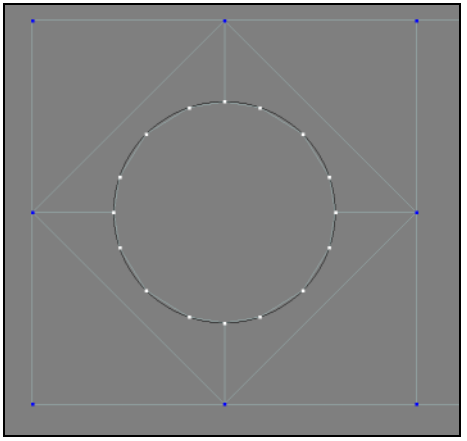
27. Turn snapping off by deactivating the **ON** icon on the Snap panel.

28. Using the Right view as a guide, translate the other circle along the Z axis until it is flush with the front of the boombox.

29. Select the boombox, then press **t** and select all the points on the inside polygon of the other speaker. Make sure that the circle fits completely inside the polygon.



30. Choose **Modify** > **Deform** > **Shrinkwrap**, pick the circle, then click the right-mouse button. Accept the default values in the Shrinkwrap property editor.

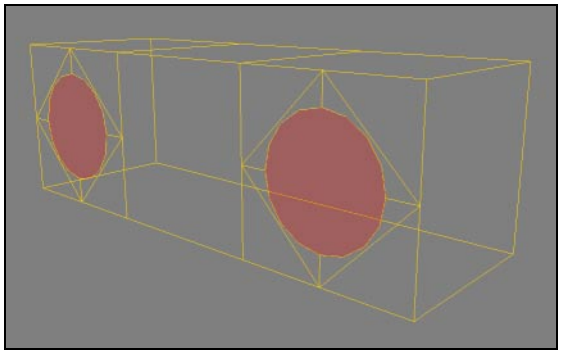


31. If you want, you can now select the two circles, then hide them by pressing the **h** key. If you want to delete them, first make sure to freeze the boombox's operator stack (left), otherwise the shrinkwrap deformation will no longer apply.

Finishing the Speakers

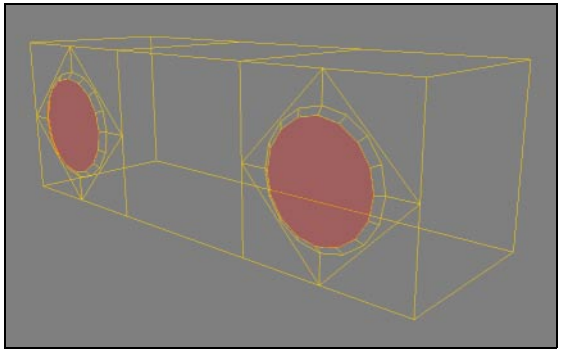
There are just a couple more steps before you've finished with the speakers:

32. Press the **u** key to activate raycast polygon selection again. The two speaker polygons should still be selected—select them again if they're not.

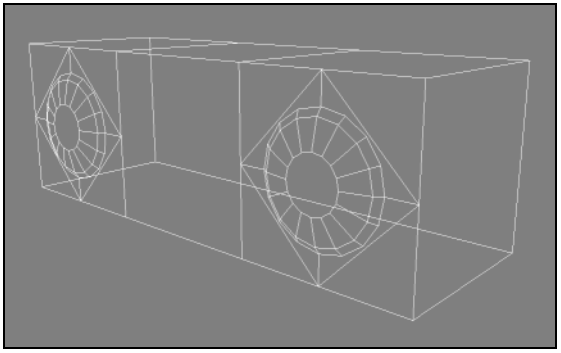


33. Duplicate the selected polygons by pressing Ctrl+d.

34. Scale the duplicated polygons down a bit.
35. Press the **v** key to activate the Translate tool, then activate **Local** mode on the Transform panel (left).
36. Drag the middle mouse button to translate the selected polygons out a bit in their local Y direction.



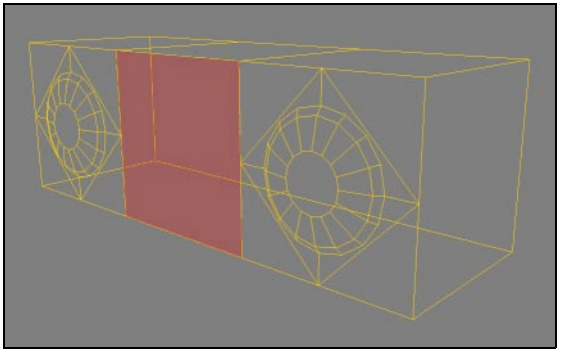
37. Repeat steps 33 to 36 to round out the speakers a little more. Your boombox should be similar to the illustration below.



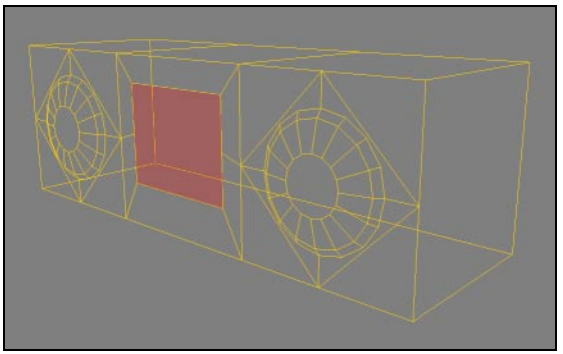
Making the Front Panel

By now you should be familiar with the process of duplicating, scaling, and translating polygons. You'll follow the same process to make the front panel.

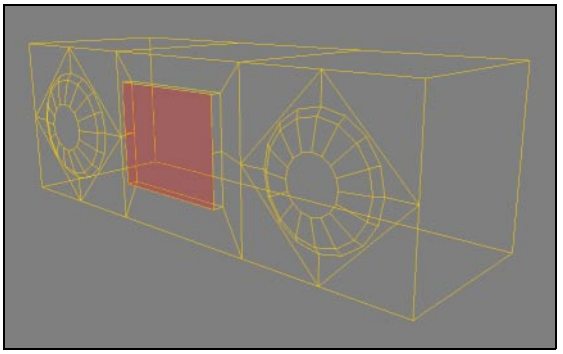
38. Select the middle front polygon.



39. Duplicate it, then scale the duplicate. You probably don't want it to be completely square, so, instead of holding the Shift key to scale uniformly, make sure that **Uni** is off on the Transform panel and use the left and right mouse buttons to scale differently in its local X and Z directions. You can also translate it slightly in its local Z direction to make it a bit higher.



40. The next step is to add some relief to the front panel. This time, instead of duplicating and translating manually, choose **Modify >Poly. Mesh > Extrude along Axis**. The Extrude Op property editor opens.
41. Adjust the **Length** until the front panel sticks out slightly.

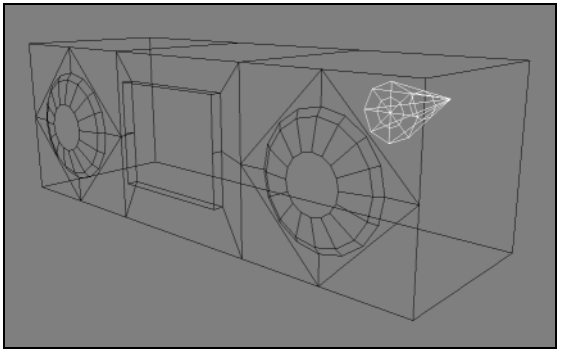


Making a Tweeter Using a Boolean

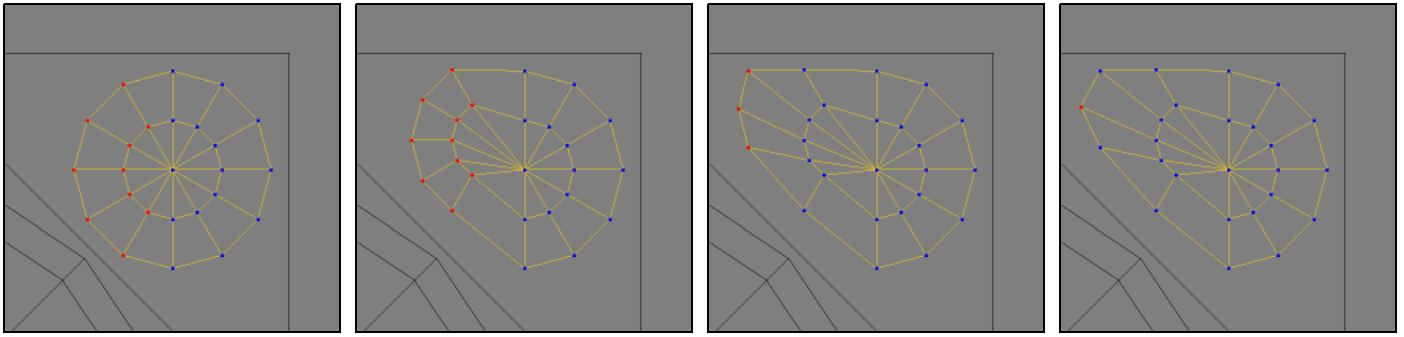
In this section, you'll use a Boolean operator to create a tweeter on your boombox.

42. Get a polygon mesh cone by choosing **Get** > **Primitive** > **Polygon Mesh** > **Cone**, and increase the U **Subdivisions** to **12**.

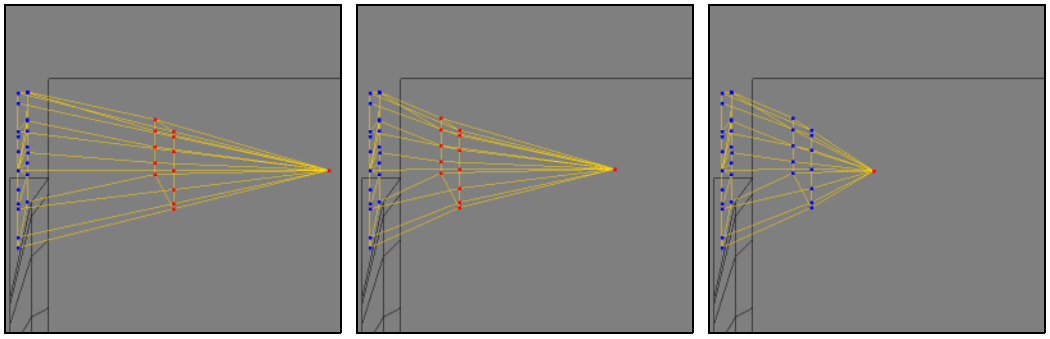
43. Rotate the cone by **–90°** around the X axis, then translate it so that it is barely sticking out of the upper outside corner of one of the speaker panels. You can also scale the cone if it doesn't fit properly. Use the orthogonal views (Top, Front, and Right) as guides.



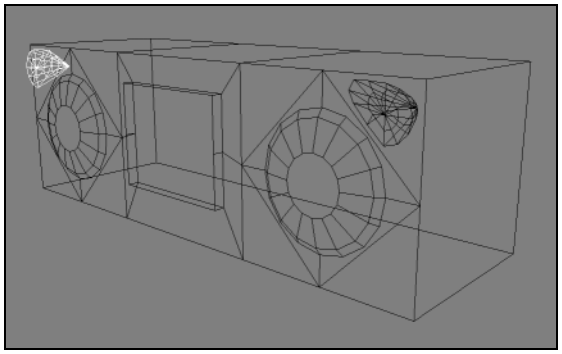
44. To give the tweeter a tear-drop shape, press the **t** key to activate Point mode. In the Front view, select the points on the left side, then scale them down and translate them. Repeat this process a couple of times, each time selecting fewer points.



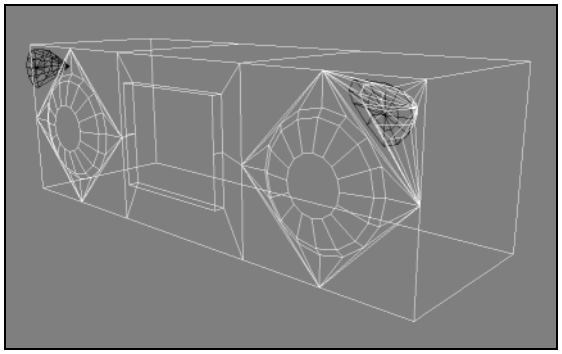
45. Use the same technique to flatten out the cone in the Right view.



46. Now you need to create a symmetric copy of the tweeter for the other speaker panel. First, press the space bar to return to Object mode, then freeze the cone's transformations by choosing **Transform > Freeze All Transforms**.
47. Duplicate the cone.
48. Scale the duplicated cone by –1 in the X direction. The second tweeter should be in place.



49. Freeze the second cone's transformations again.
50. Select the boombox, then choose **Modify > Poly. Mesh > Boolean >Difference**.
51. Pick the first cone. The cone's geometry is subtracted from the boombox.



52. You can now hide the cones. If you prefer to delete them, make sure you freeze the boombox's operator stack first.

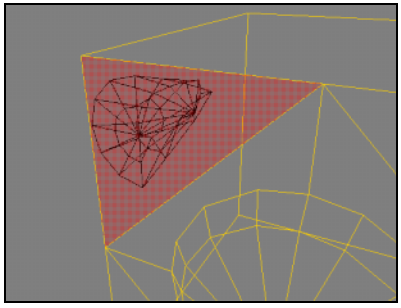
Making Another Tweeter: Using Merge

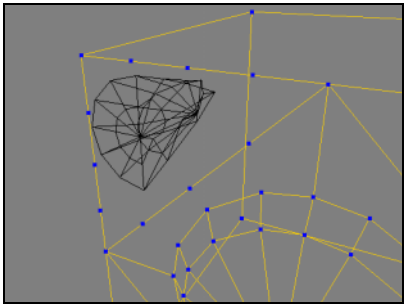
Just as you used both snapping and shrinkwrap for the speakers, you'll use a different technique for the second tweeter.

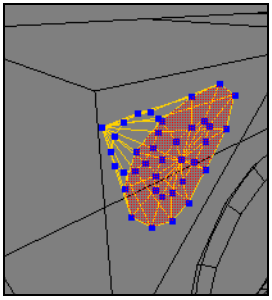
53. Select the triangular polygon on the top corner of the boombox (where the second cone will be attached). Choose **Select** > **Adjacent** > **Edges**.

54. Choose **Modify** > **Poly. Mesh** > **Subdivide Polygons/Edges** and set 4 as the number of **Subdivisions**. Note that if Points are not visible, you can press **m** (move-point tool) to temporarily display them and see the endpoints of the new edges.

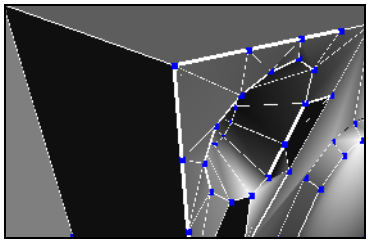
55. Press **u** to return to raycast-polygon mode. The corner triangle should still be selected—press the Delete key to remove it.



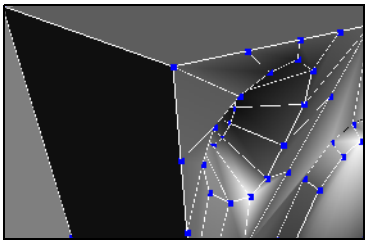




56. Select the cone, then select all the polygons at the base and delete them too.
57. Press the space bar to select the cone in Object mode, then choose **Modify** > **Poly. Mesh** > **Invert Normals**.
58. Select both the boombox and the cone, then choose **Create** > **Poly. Mesh** > **Merge**.
59. Lock the Merge property editor by clicking the lock icon in the top right corner.
60. Press 8 to open a scene explorer.
61. Select the cones and the cube (the original boombox), then press **h** to hide them. This leaves the result of the merge operation (polymsh) still visible.
62. Select the merged geometry again.
63. Notice that when the cone and cube were merged, individual points were also merged and moved to their average position. Activate the **Blend** option in the Merge property editor—now the points stay in their original locations and polygons are added to fill in gaps under the specified **Tolerance**.
64. Adjust the **Tolerance** until polygons fill in all the holes properly. Make sure no polygons are overlapping by looking for bold edges (see the following illustration)



Bold edges indicate overlapping polygons.



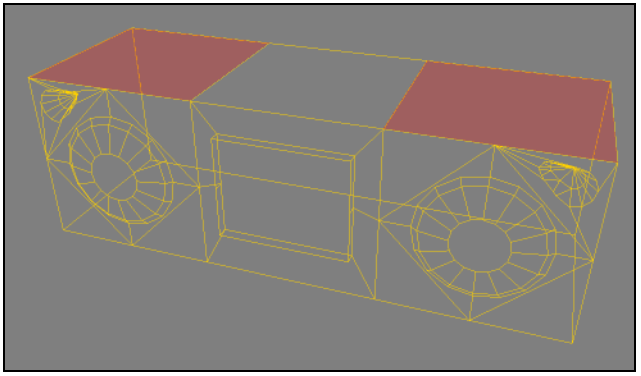
Correct tolerance value fills all holes.

65. You now have a boombox with two tweeters. If desired, you can now freeze it, then use an explorer view to select and delete the cones, the original boombox, and intermediate boombox accumulated since you started this model.

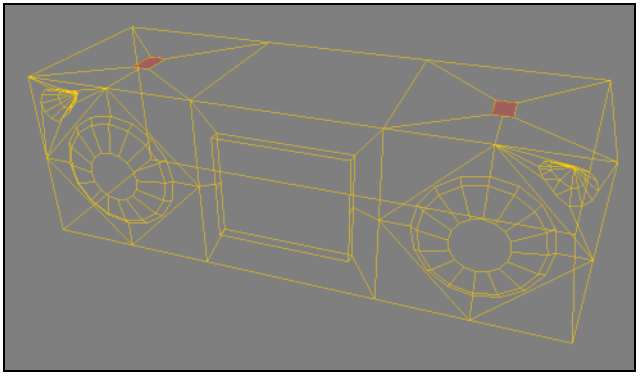
Building a Handle

You are going to create a handle for the boombox.

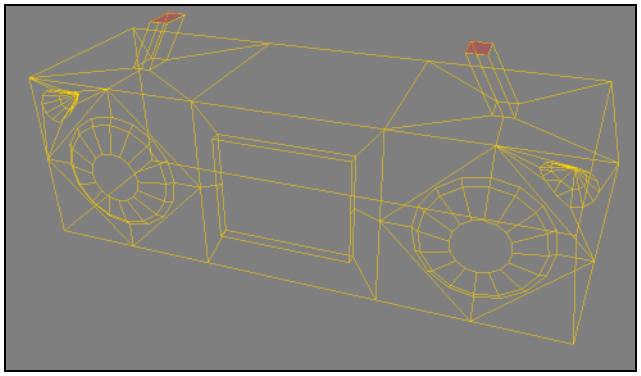
66. Select the two outer polygons on top of the boombox.



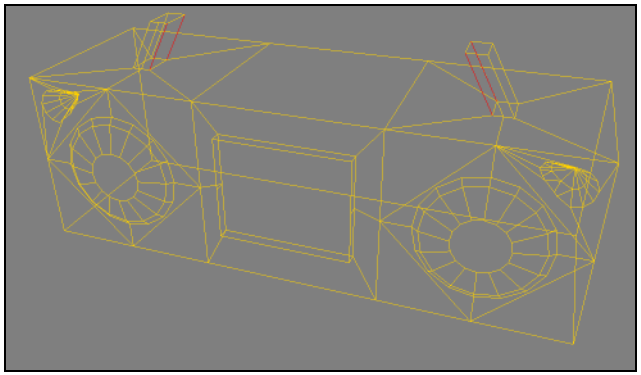
67. Duplicate and scale them as shown.



68. Duplicate them again, then translate them in Y and X in Local transformation mode.

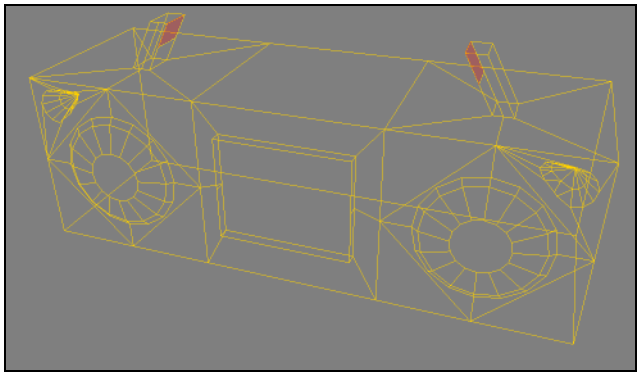


69. Select the four inner vertical edges as shown.

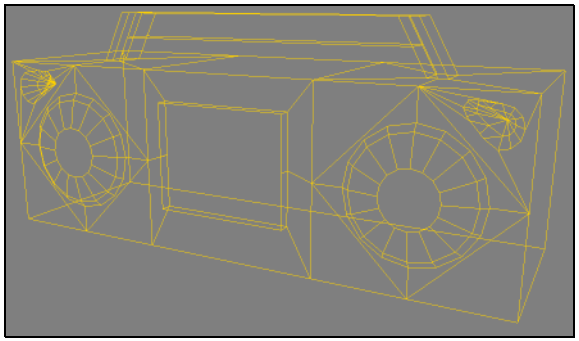


70. Choose **Modify** > **Poly. Mesh** > **Subdivide Polygons/Edges**. Accept the default **Subdivisions** of 2 and other default values.

71. Select the two upper inside polygons as shown.



72. Choose **Modify** > **Poly. Mesh** > **Bridge Polygons**. The two selected polygons are removed, and the holes joined by a "bridge."



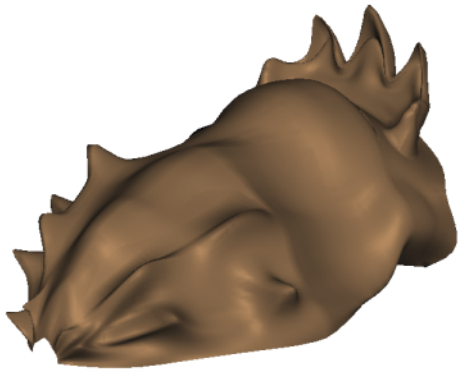
Conclusion

This tutorial provided a quick overview of some of the polygon modeling tools available. There are many more—experiment on your own. For example, try beveling the edges of your boombox.

For more information, see the *Modeling & Deformations* guide.

Tutorial 4: NURBS Surface Modeling—Burt's Bumpy Back

Burt is an alien creature. As disgusting as he is, you'll build his outer shell using the various tools for creating and modifying NURBS surfaces.

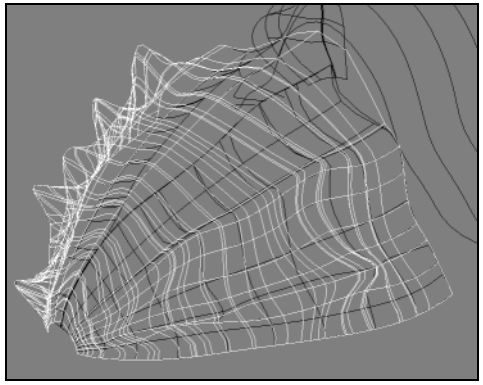


This tutorial shows you how to:

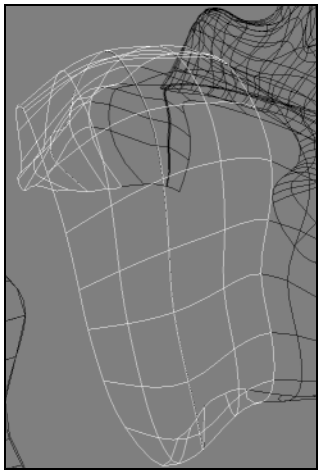
- Create surfaces using Curve Net, Loft, Merge, and Fillet Intersection.
- Modify surfaces using Clean and Extend to Curve.
- Use layers to organize your scene.
- Create seamless NURBS surface meshes by snapping boundaries and assembling.

Overview

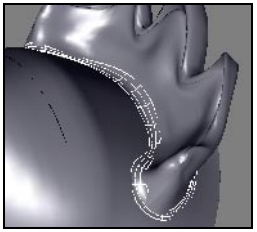
1. **Create Burt's head using Curve Net, Clean, and Merge.**



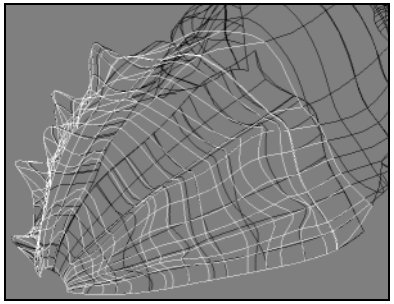
2. **Create Burt's middle using Loft and Extend to Curve.**



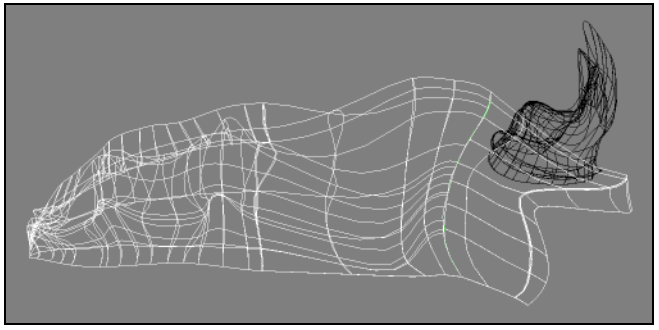
3. **Create a fillet between Burt's back and spines.**



4. **Snap Burt's head to his middle.**



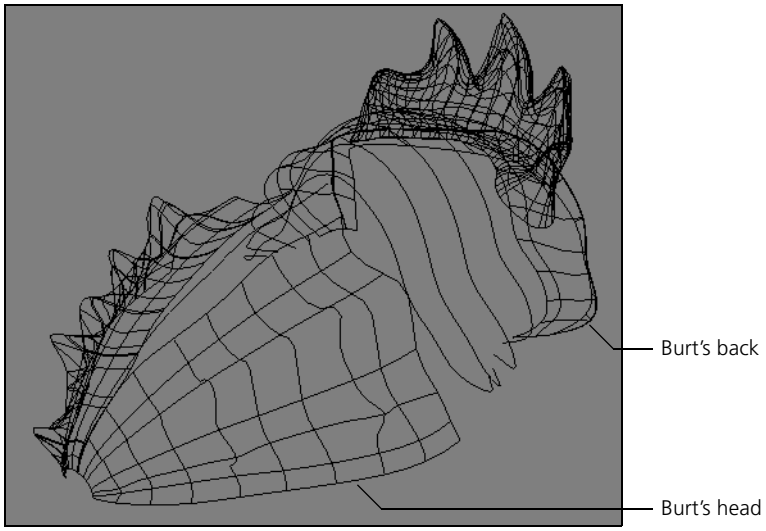
5. **Assemble a single surface out of Burt's head, middle, and back.**



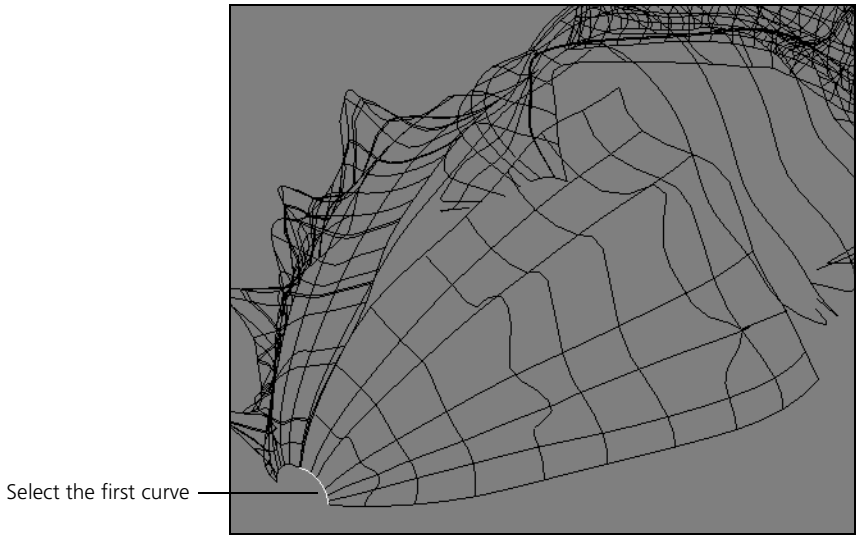
Creating a Surface with Curve Net

You'll use the Curve Net tool to create a surface section of Burt's head. Curve Net creates a surface from two sets of curves: a set in U and a set in V.

1. Open **MOD_e2_BURT** from the **Scenes** folder of the **Tutorial_Project**. The Burt model (shown below) comprises the creature's head and back.



Pressing the **g** key hides or displays the grid for the viewport that currently has focus. This is the one under the mouse pointer if **Give Windows Focus When the Mouse Enters Them** is on in the **General** page of **File > User Preferences**. (Shift+g to hide the grid in all views)



2. Select the first curve at the tip of the head (called *first_curve*).

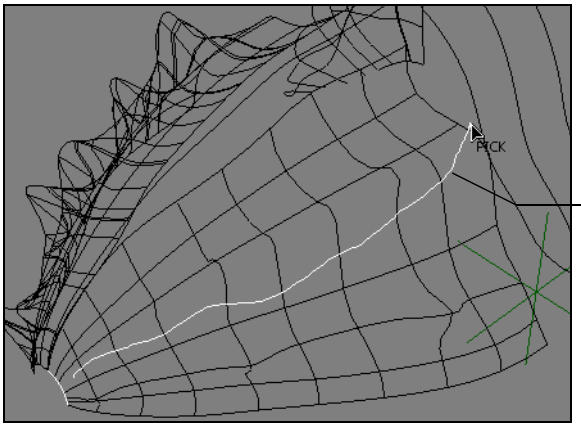
   To make this easier, change your view so that the area is well centered. Move the mouse pointer over the camera view and press F12 to toggle to full screen mode (press F12 to toggle back if needed). Select a few curves in the front section area and press **f** to frame them and center the camera interest. You may need to orbit, pan or dolly a little:

   - The **s** key (Navigation tool) combines orbit (left mouse button), dolly (middle mouse button), and tracking (right mouse button). Take some time to enjoy this tool.
   - To track from side to side or up and down, hold down the **z** key while dragging the left mouse button.
   - To zoom in, hold down the **z** key while middle-clicking.
   - To zoom out, hold down the **z** key while right-clicking.
   - To dolly in or out, hold down the **p** key while dragging left or right. Use the left, middle, and right mouse buttons to dolly at a slow, medium, or fast speed, respectively.
   - To orbit in the perspective view, hold down the **o** key while dragging. Use the left mouse button to orbit freely, the middle mouse button to orbit horizontally, and the right mouse button to orbit vertically.

If you pressed and released the **s**, **z**, **p**, or **o** key too quickly, you may have activated the tool in sticky mode instead of supra mode. If this is the case, press and release the space bar to return to the selection tool before selecting the first curve.

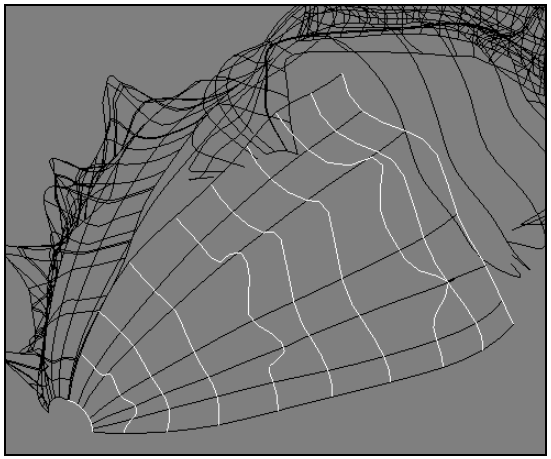
3. Choose **Select** > **Tools** > **Freeform** from the Select panel in the main command area, or press F9. The Freeform selection tool lets you select scene elements by drawing a line across them.

4. From the Model toolbar, choose **Create** > **Surface** > **Curve Net**. The status bar at the bottom of the screen prompts you to pick curves.

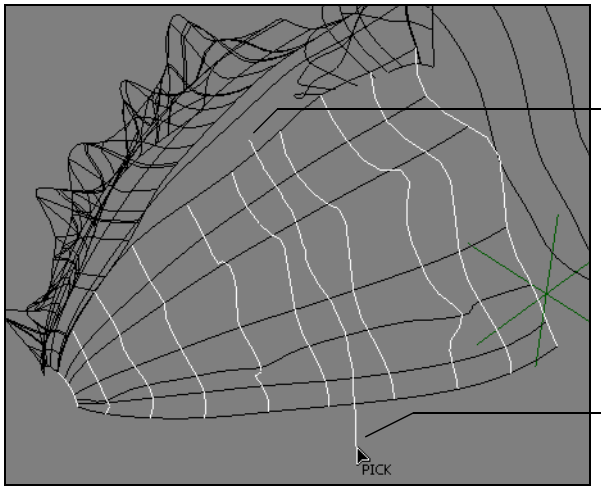


The white line shows the freeform path used to select the curves.

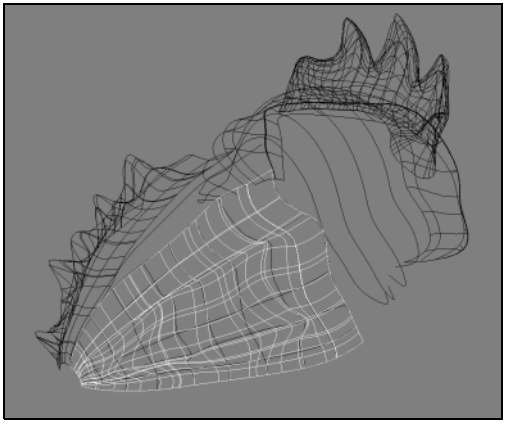
If you make a mistake, Ctrl+click to “unpick” the last curve. Repeat to “unpick” successive curves.



5. Right-click when you have finished picking the first set of curves. Now you can select the next set of curves.
6. With the left mouse button, draw a line across the head section to select all the curves in the other direction, beginning with the curve near the center of the head (don't pick the curve in the center just yet).



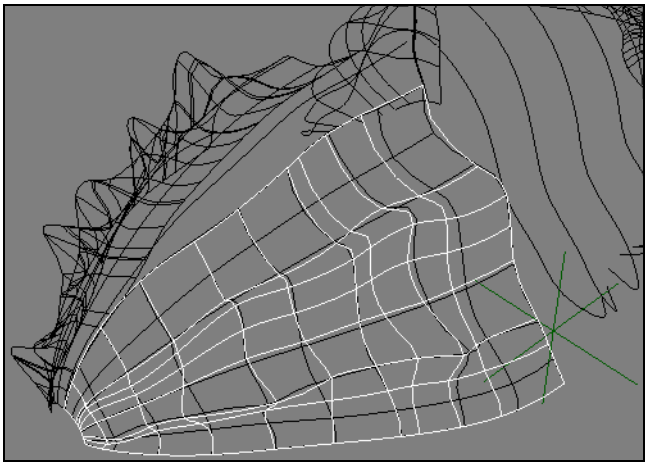
7. Right-click to finish the surface. A new surface is created with the default name **surfmsh**, and the Curve Net property editor opens. Leave the values at their defaults and close the property editor.



Cleaning a Surface

Because the resulting surface may have too many points, you can use the Clean tool to control the number of points on your surface.

8. With the half-head surface (**surfmsh**) still selected, choose **Modify** > **Surface** > **Clean** from the Model toolbar.
9. Make sure that both **Clean in U** and **V** are on, and modify the **Tolerance** values until the balance between the amount of detail and the number of subdivisions (the "heaviness" of the geometry) is to your liking.

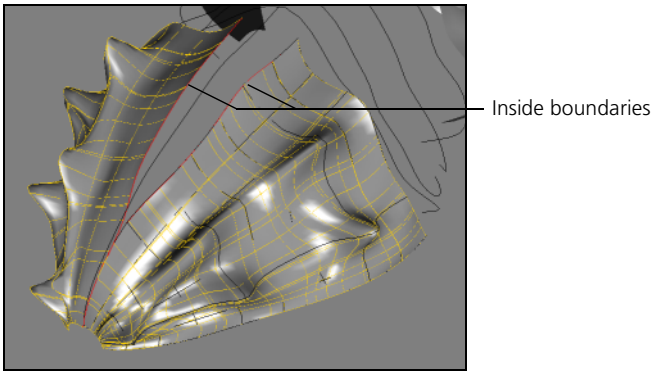


Merging Surfaces

The next task is to merge the new surface to the other half of the head, using a curve for the transition. The Merge Surfaces command creates a new surface that spans the two original ones.

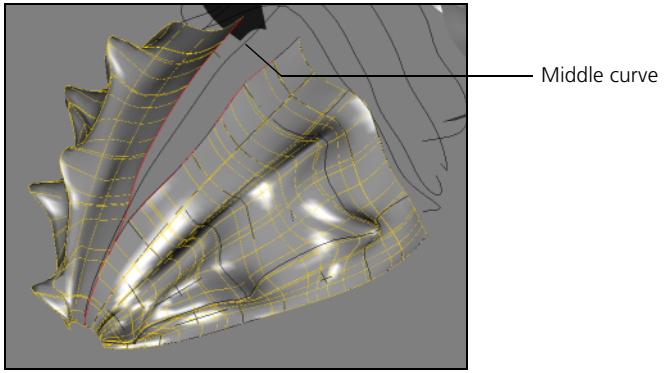
10. With the cleaned surface (**surfmsh**), add the other half of the front section (**front_section**) to the selection by clicking on it while pressing the Shift key.
11. Click the **Boundary** button (left) on the Select panel to activate the boundary selection filter. Both surfaces are highlighted in yellow. When this filter is active, you can only select boundaries of NURBS objects.

12. Shift+select the inside boundary of each surface.



13. Choose **Create** > **Surface** > **Merge** from the Model toolbar.

14. Pick the middle curve to act as an intermediate between the two surfaces (and right-click to end the picking session). A new surface is created with the default name **surfmsh1**, and the Merge Surfaces property editor opens.

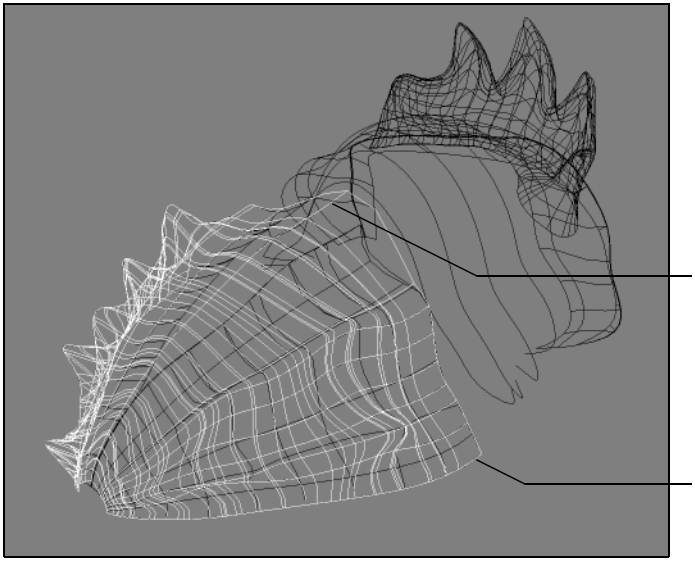


You can cancel any mode or tool you are using by pressing the Esc key.

15. The **Boundary Curve Adaptor** properties determine which boundaries of the two input surfaces are used for the merge. By default, they are set to the boundaries you picked, but you could change them if you wish.

16. On the **Shape** page of the **Merge Surfaces** property, set **Seam** to **Curve**. This specifies that the seam between the two input surfaces is the curve you picked earlier.

All of the viewport functions (Pan, Orbit, Shaded mode, etc.) are available while a property editor is open and being modified.



17. Close the property editor.

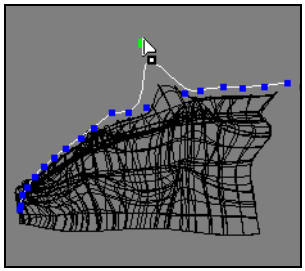
18. Select the half-head section (**surfmsh**)that you first created and press the **h** key to hide it. The other half will be hidden using the layer feature later on.

Relational Modeling

Relational modeling lets you modify a surface using the curves from which the surface was built. For example, you can create a glass by revolving a profile, then edit the glass by moving points on the original profile curve. Relational modeling is SOFTIMAGE|XSI's default behavior.

19. Enter **middle_curve** in the Selection text box (left) in the Select panel (main command area) and press Enter. This selects the middle of the head.

20. Move the mouse pointer over the perspective view, and press **F12** to enlarge it to full screen. Press **f** to frame the selected curve.



21. Use the **m** key to interactively move points on the curve, or tag points with the **t** key and translate them. Notice how the merged head changes shape in response to the new shape of the curve. Notice also that you can select the merged head and move points on it—these changes are preserved when you reselect the curve and move points on it.
22. Press Ctrl+z until the curve get's its original shape back.

Freezing the Operator Stack

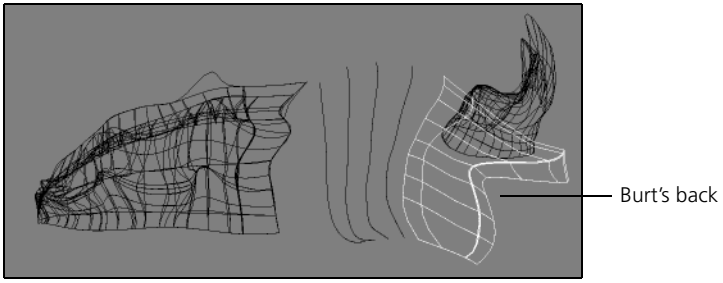
Freezing the operator stack collapses the surface's construction history and breaks the modeling relation with the input curves. It is as if the surface was imported "as is" directly into SOFTIMAGE|XSI.

23. Select the merged head surface that you just created (**surfmsh1**).
24. In the Edit panel of the main command area, click the **Freeze** button (left). This removes the operator history of the surface, including the modeling relations.
25. Try selecting the curve in the middle of the head and moving points again. Notice how the merged head surface is no longer affected by changes to the curve.

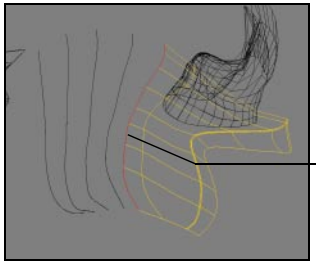
Lofting

Lofting creates a surface out of a series of curves, similar to the Skin feature of SOFTIMAGE|3D. However, with the Loft command you don't need the same number of points on each curve. You can also loft using any combination of curve, objects, isolines, boundaries, or knot curves.

26. With the mouse pointer over the viewport, press F12 again to return to the four-port view.
27. Select the back of the body (**back_section**).



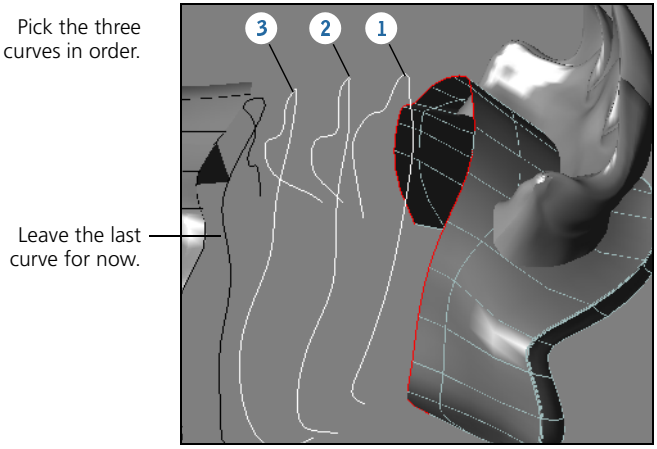
28. Click the **Boundary** button (left) on the Select panel to activate the boundary selection filter, and select the boundary at the front of the surface.



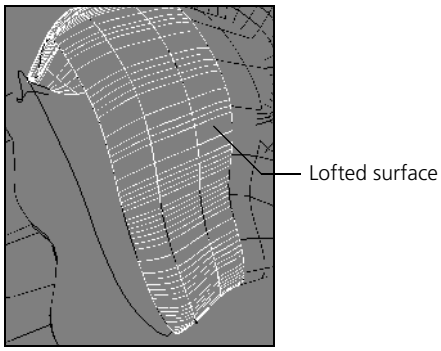
29. Press F9 to activate freeform selection mode.

30. From the Model toolbar, choose **Create** > **Surface** > **Loft**.

31. Pick the first three of the free-floating curves in order, as in the illustration below. Don't pick the last curve—you'll add it in a later step.



32. Right-click to finish picking. A new surface is created (by default, named **surfmsh2**) and the Loft property editor opens. Leave the values at their default and close the property editor.



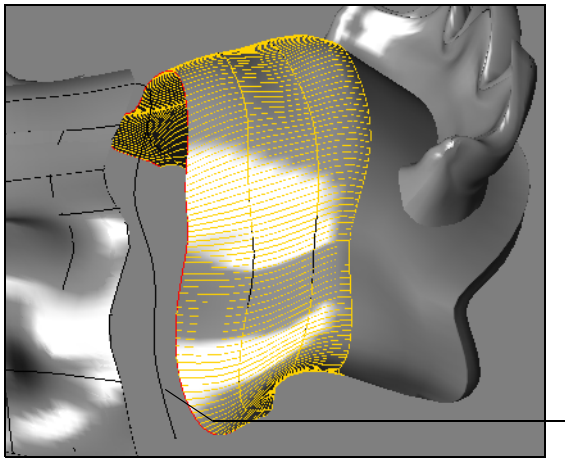
Using the Extend-to-Curve Command

You can now attach the last curve to Burt's back. The extend-to-curve tool extends the selected surface to include a chosen curve.

33. Make sure that only the lofted section (**surfmsh2**) is selected.

- You can check the Selection text box to validate your selection—if you have more than one selection, it displays MULTI (nn).
- If more than one object is selected, you can use the Selection button in the Select panel (main command area) to refine the selection. Click the Selection button, and a pop-up explorer appears listing all selected elements. Click on an individual element to select only it.

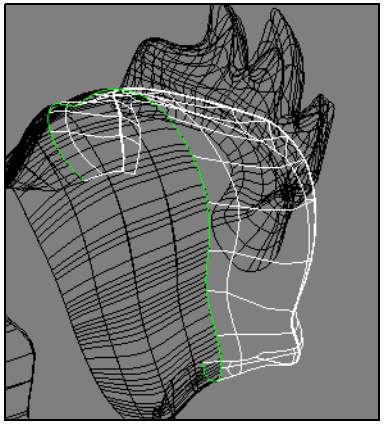
34. Pick the boundary of surfmsh2 that is closest to the remaining curve.
35. Choose **Modify** > **Surface** > **Extend to Curve** from the Model toolbar.
36. Pick the last remaining curve to join it to the alien's back surface. The Extend to Curve property editor opens; simply close it.



Filleting Intersections

Use the fillet-intersection tool to create a seamless surface between two intersecting objects; in this case, the back and the spikes.

37. In Object selection mode, select the back section (**back_section**) once again.

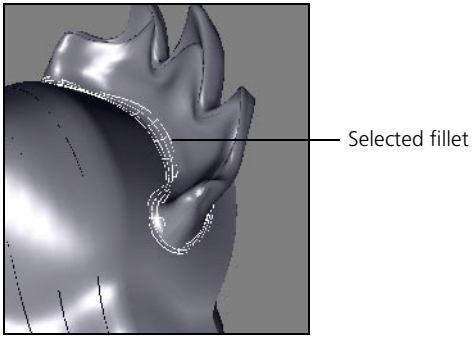


38. Choose **Create** > **Surface** > **Fillet Intersection** from the Model toolbar.

39. Pick the spiked surface on the model's back with a left click. The fillet is created (default name **surfmsh3**), and the Fillet Intersection property editor opens.

To tweak your fillet parameters, you may find it helpful to change your viewport display to Shaded view.

40. Adjust the parameters in the Fillet Intersection property editor. You can set the number of U and V subdivisions as well as the radius of the fillet. If you set the radius too high for the intersecting objects, it becomes impossible to calculate the fillet, and the result is a degenerate surface.



Preparing to Snap Boundaries

The final goal is to assemble a seamless surface mesh composed of Burt's front, middle, and back. To assist in the assemble operation, you need to snap boundaries together. However, the Snap Boundaries command requires that both surfaces have the same number of points at their junctions. To accomplish this, use a combination of fitting and cleaning surfaces.

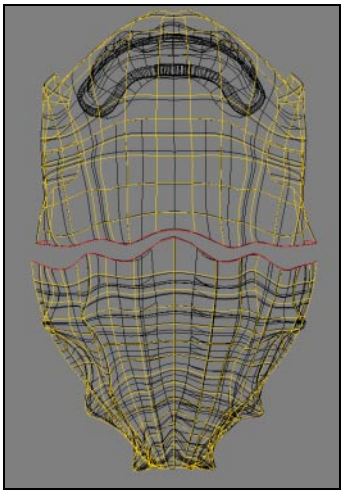
41. Multi-select the merged front, middle, and back.

42. Choose **Create** > **Surface** > **Fit**. Three new surfaces are created.

43. Notice how a great deal of detail is lost from the front section with the default values. To correct this, increase both **U** and **V Subdivision** to **15** or more in the Fit Surface property editor. This creates unnecessary detail in the back and middle, which you'll remove in the next steps.

44. With the three new surfaces still selected, choose **Modify** > **Surface** > **Clean** again.

45. In the Clean Surface property editor, turn off **Clean in U**. This removes unnecessary points in V while ensuring that all three surfaces have the same number of points in U.



Snapping Boundaries

You are now ready to snap boundaries together. Notice how the back and middle sections are already joined and their knot curves line up exactly. This means that you only need to snap the boundaries of the front and middle sections.

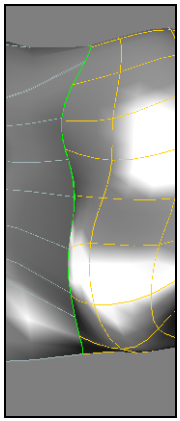
46. With the three cleaned surfaces still selected, choose the Boundary filter again and select both boundaries between the front and middle.
47. Choose **Create** > **Surf. Mesh** > **Snap Boundary**. The two boundaries snap together and the Snap Boundary property editor opens.

Creating a Working Layer

Use layers to ease the management of all the original curves and surfaces in this scene. You'll create a new layer to work in and include only the objects you need for the final model. This means you can hide the original curves and geometry, used as reference when building these final surfaces, by turning off the view and render visibility of the original layer Layer_Default.

48. The spike_section and the fillet (**surfmsh3**) are done, so select them both now in Object mode.
49. Create a "working" layer that includes the selected objects by choosing **Layers** > **New Layer** (left). Name it as you like. Notice that the new layer automatically becomes the active one in the Layers list.

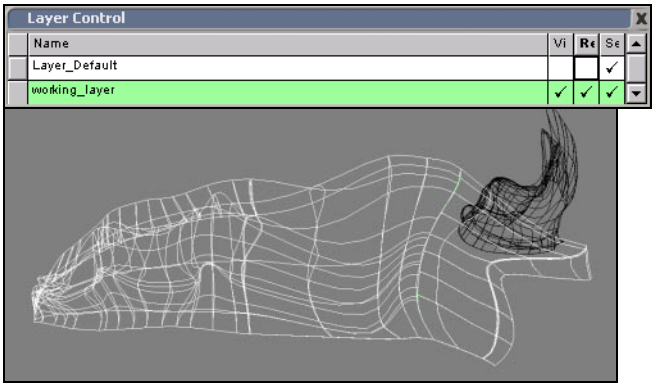
Any additional objects that you create will automatically be inserted into the active layer.



Assembling Seamless Surfaces

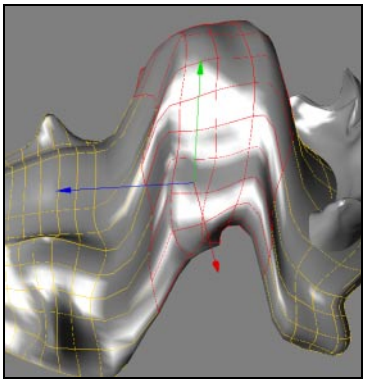
The Assemble tool creates a new surface mesh based on the selected surfaces. A surface mesh is a single 3D object composed of multiple subsurfaces. Although the result may look like a single surface, you can still pick the individual component subsurfaces. So let's assemble the parts!

50. In Object selection mode, multi-select the cleaned and snapped front, middle, and back sections. These should be named **surfmsh4**, **surfmsh5**, and **surfmsh6**.
51. Choose **Create** > **Surf. Mesh** > **Assemble** from the Model toolbar. The Assemble NurbsMesh dialog box opens. Accept the default values and click OK.
52. The Surface Mesh property editor opens. Give the new surface mesh a name if desired, then close the property editor.
53. As discussed earlier, you'll now hide all the other objects used to create your surfaces: choose **Layers** > **Layer Control** from the Layers panel and deselect View and Render visibility for Layer_Default. Close the Layer Control window.



54. From the Selection filter list, choose **Subsurface** (**left**).

55. Select one of the three subsurfaces and translate it a few units away. Notice how the continuity between subsurfaces is maintained.



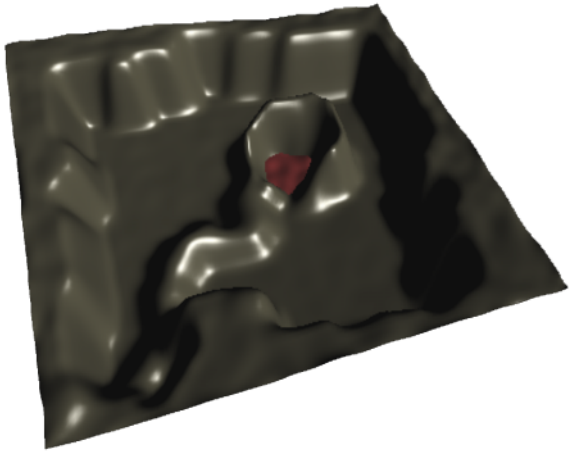
Conclusion

This tutorial introduced you to a few of the NURBS modeling tools available in SOFTIMAGE|XSI. There are many other tools for creating and modifying surfaces, including Revolution, Extrusion, Birail, and so on.

For more information, see the *Modeling & Deformations* guide.

Tutorial 5: Deformations and Weight Maps—Burt's Volcano

Deformations are operators that change the shape of geometric objects.

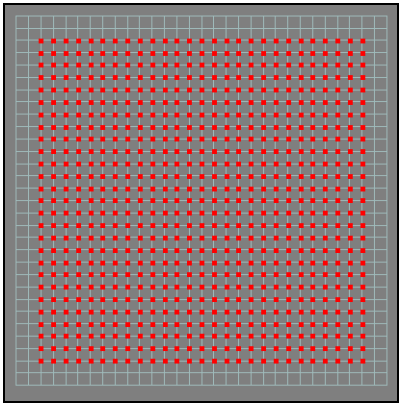


This tutorial shows you how to:

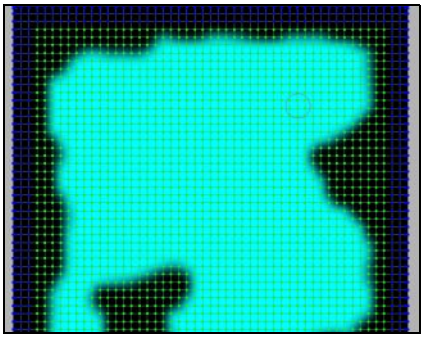
- Apply various deformations.
- Use weight maps to modulate the amplitude of deformations.
- Control deformations using other objects: curves (deform by spine) and lattices.
- Use proportional modeling to adjust a point's neighbors automatically as you move it.

Overview

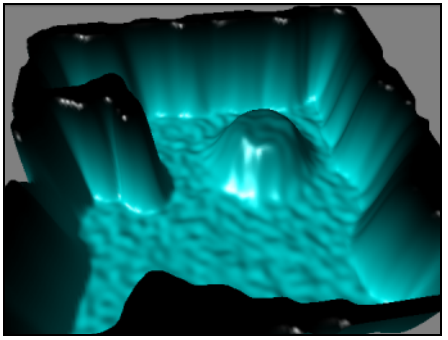
1 **Start with a grid.**



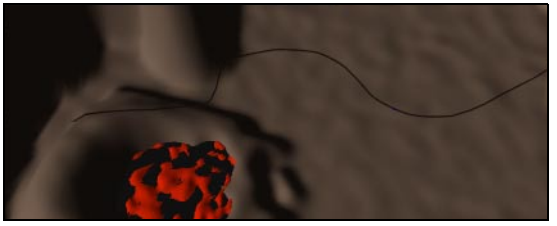
2 **Paint a weight map on it.**



3 **Create a volcanic landscape with Push and Randomize deformations.**



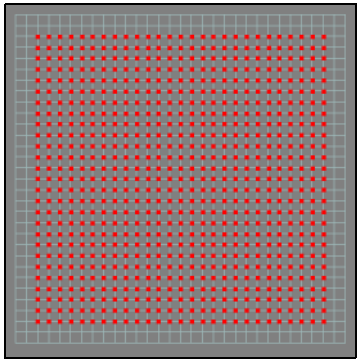
4 **Add more deformations:**
Shrinkwrap,
Deform by Spine,
Deform by Lattice.



Burt Lives in a Volcano

Burt needs a new landscape architect, and you've just landed the job. You'll create a grid with a weight map and apply many types of deformations (Push, Randomize, Shrinkwrap, Deform by Spine, Lattice, Twist, Wave) to various parts of that grid to achieve a suitable environment.

1. Start this tutorial with a new scene. Press **Ctrl+n**, or choose **File** > **New Scene** from the menu bar at the top of the main window.
2. Choose **Get** > **Primitive** > **Surface** > **Grid**. Set **U** and **V Length** to 50, **U** and **V Subdivisions** to 30, and name the grid **landscape**.



Deforming the Landscape

By creating a weight map, you can define by how much each area of points on the model's surface will be affected by a deformation. The weighting is defined by preset maps or by using the Paint tool.

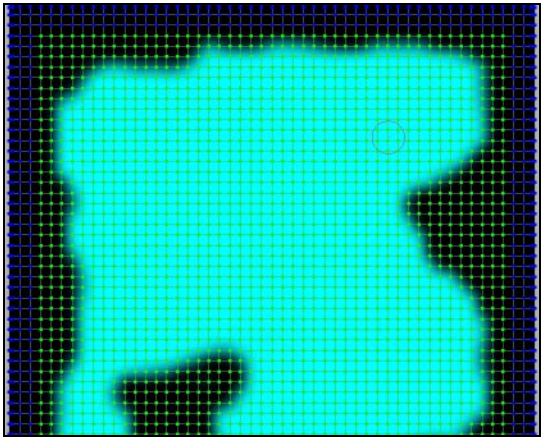
3. In the Top view, tag some points on the grid (press **t**), leaving a border of about two rows.
4. Choose **Get** > **Property** > **Weight Map** to create a weight map for the grid. Name the weight map **wm_bottom** and close the property editor.
5. Press **w** to activate the Paint tool, which you'll use to apply different weights. Note that when the Paint tool is active, moving the pointer over a 3D view activates the display of weight maps in that viewport.

To display weight maps when the Paint tool is not active, set the display type to **Constant** and turn on **Weight Maps** in a viewport.

The default weight map is a constant value of 1.00 everywhere. This is represented by the turquoise color.

6. Press Ctrl+w to open the Brush Properties editor and play with the brush properties to see the effect of setting the **Radius**, **Hardness** (the core of the brush), **Softness** (the area surrounding the core), and **Opacity** (strength of the weight).

7. Right-click and drag the brush around the edges of the turquoise area to decrease the weight and create more organic-looking edges. You can add weight back again with the left mouse button.



8. With the weight map still selected, create a sunken hole in the landscape by choosing **Modify** > **Deform** > **Push** from the Model toolbar. Set the **Amplitude** to –10.

Now that you can see the new lay of the land, use the Paint tool to modify the weight map more. The map's turquoise area is affected 100% by the deformations, the black area is affected 0% by deformation, and the gradation area between the two is a value between 0 and 100.

9. Add random deformations to the landscape by choosing **Modify** > **Deform** > **Randomize**. Set **X** to 0 and **Y** to 0.2 for some small vertical bumps. Note that because the weight map is still selected, it is used to control the Randomize deformation as well as the Push.

In the explorer, you can expand the landscape/NURBS Surface mesh node to see the deformation operations you have done (Push and Randomize). These operators can be deleted at any time, if need be.

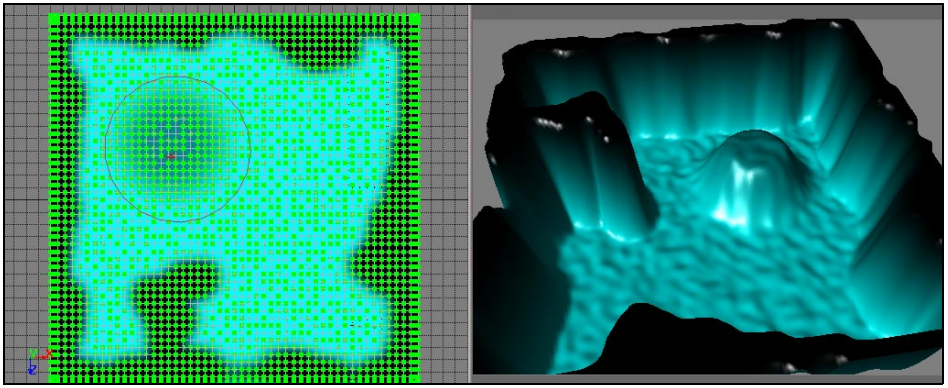
10. In the Front or Right view, tag the undeformed points at the top of the pit.

11. Add a Randomize deformation of about 0.75 in Y.

Creating a Volcano

And now.... a volcano! Burt loves nothing more than chowing down on red-hot gobs of lava.

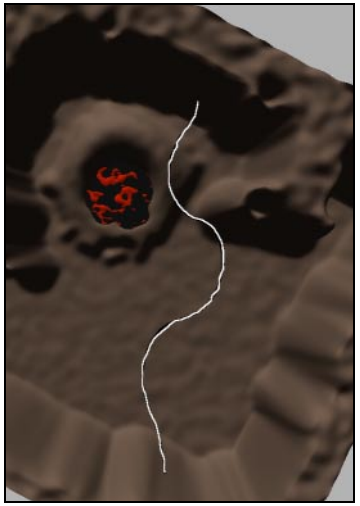
12. Reactivate the Paint tool if it is no longer active by pressing **w** again. Note that even though the weight map is no longer selected, you can still paint on it. This is because the last selected weight map is considered to be active.
13. Change the Paint brush's properties (Ctrl+w) so that the **Radius** is 6, **Hardness** is 40, **Softness** is 80, and **Opacity** is 60.
14. Paint a mound for the volcano in a corner of the weight map.



15. Make the brush's diameter a bit smaller and decrease the **Opacity** to 45. Paint a crater, offset a little on top of the volcano.

Drag with the middle mouse button to quickly change the brush radius.

16. Create a small grid to act as the lava base and name it **lava**.
17. Position the grid under the crater and scale it to cover the bottom of the crater. Add a Randomize deformation to it for a little texture.
18. With the lava still selected, click the **Parent** button in the Constraint panel (or press the / key) and middle-click the landscape to make it the parent of the lava. Right-click to terminate the Parent tool.



Shrinkwrapping and Deforming a Path Formation

Now you'll use the Shrinkwrap and Deform by Spine deformations. When using these two in combination, you need to freeze the object's operation stack (history) before using them; otherwise a "cycle" is created because one deformation depends on the result of the other deformation. It's **very important** to understand that when you freeze a history stack, you cannot modify any of the operations that were previously applied to your object. Freezing is final!

First, you want to create a path formation that cuts across the volcano and the rest of the landscape base:

19. In the Top view, create a curve (**Create** > **Curve** > **Draw CV NURBS**) winding along the bottom and across the side of the volcano.

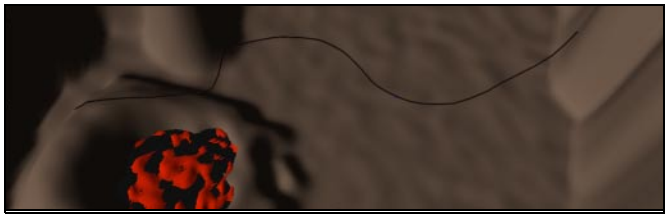
You need quite a few points on the curve to give enough definition for the deformations that you'll do shortly.

20. With the curve still selected, choose **Create** > **Curve** > **Fit on Curve.** Increase the number of points to about 28 and name the new resampled curve **spine**.

    Using **Fit on Curve** is a quicker way of resampling a curve than adding points to the original curve.

21. Select the curve and choose **Modify** > **Deform** > **Shrinkwrap**. Pick the landscape and right-click to end.

22. In the Shrinkwrap property editor, select the **Reverse** option, select **Parallel to Axes** as the **Projection** type, and select **Along Y** in the Parallel to Axes section.

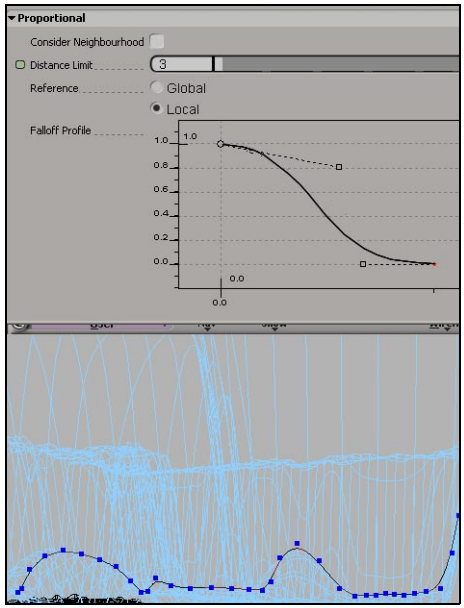


23. With the curve still selected, click **Freeze** again because you're now going to deform the landscape by the spine and the spine is already shrinkwrapped to the landscape.

24. Select the landscape and choose **Modify** > **Deform** > **by Spine**. Pick the curve and right-click to end. Notice how the blue points show the area of deformation influence of the curve (make sure **Clusters** are visible in the Top view).

25. In the Deform by Spine property editor, click the Radius tab and notice that the curve on the **Longitudinal** graph is uniform along the spine. If you move a point on the curve, you'll see the area of influence increase or decrease significantly (the setting is fairly sensitive so don't change it too much). **Amplitude** is the falloff of the effect as you get further from the curve.
26. Translate the curve up a little in Y to see how the topology of the landscape changes. Close the property editor when the path formation looks as you want it.

Moving Points for Random Deformation



To make the curve a little more random, you'll move some points on it.

27. In the model toolbar choose **Modify**> **Component** > **Proportional**. Proportional modeling lets you specify the overall influence a point has on its neighbors.
28. Choose **Modify** > **Component** > **Proportional Setup** and Change the **Distance Limit** to a value that you like.
29. Adjust the **Falloff Profile** curve to change the amount of influence each point has on its neighbors.
30. Move point on the curve until your are happy with the landscape's topology.

Deforming the Landscape with Lattice and Twist

Change the overall shape of the landscape by using a lattice.

31. Branch-select the landscape (use the middle mouse button) and choose **Get** > **Primitive** > **Lattice**. Change the **Interpolation** for the XYZ axes to **Curve** instead of Linear for a curvier deformation.
32. Tag some points on the lattice and translate to change the landscape as you like.
33. Now add a little twist to the landscape!
34. Tag all points on the lattice except for the top row and choose **Modify** > **Deform** > **Twist**. Set the **Angle** to a value that works well.

Conclusion

NURBS are extremely useful for modeling smooth organic shapes without requiring a huge number of points. Although NURBS surfaces must be more or less rectangular, you can assemble multiple surface patches together to create seamless objects.

Deformations are a powerful and flexible way to control the shape of objects, and weight maps give you even more control. You can paint weights onto maps to create exactly the shape you need.

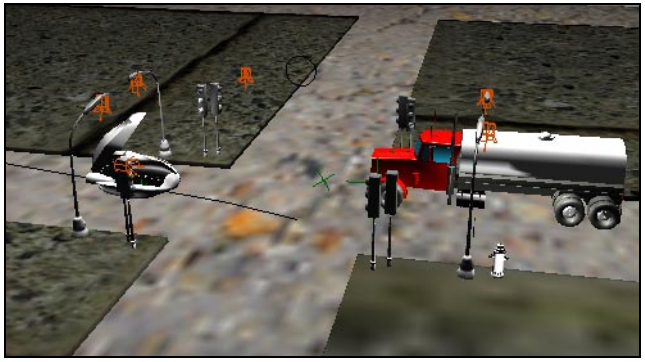
For more information, see the *Modeling & Deformations* guide.

Section 4 **Animation Basics**

Tutorial 6: Low-level Animation—Chaos at Brontasaurus and Main

In busy downtown Megasauratropolis, a spaceship swoops down for a landing. A truck comes to a screeching halt. Street lights sway and twist, and traffic lights flash crazily. As the alien ship lights gently in the middle of an intersection, the cover slowly opens.

This is an exercise in animating several events at the same time, with some events linked to others.

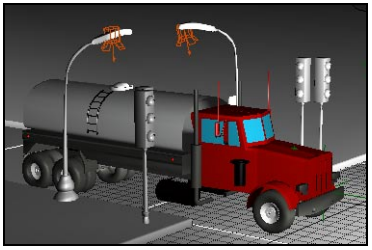


This tutorial shows you how to:

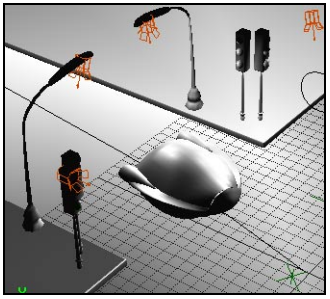
- Mark parameters, set keyframes, and use autokey to animate the truck.
- Create a path to animate the spaceship.
- Link the twist motion of the street lamps to the space ship's position along its path using linked parameters.
- Defne custom parameters for controlling the traffic lights.
- Use expressions to open the cover of the spaceship.

Overview

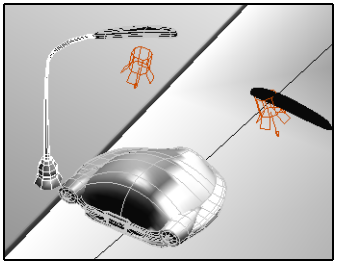
1 **Animate the truck by keyframing its translation.**



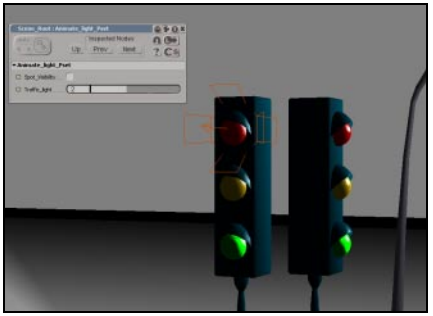
2 **Animate the spaceship on a path.**



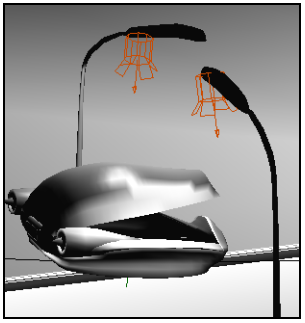
3 **Link lamp and spaceship parameters.**



4 **Create custom parameters to animate traffic lights.**



5 **Use expressions to open the spaceship cover.**



Animating the Truck along Its Local X Axis

One of the most basic ways to animate is through keyframing. With SOFTIMAGE|XSI, you can "mark" the parameters you want to animate beforehand, then move the timeline pointer and press the **k** shortcut key to save a keyframe (or use autokey).

Get the scene and isolate the truck

1. Open the **AN1_e1_General** scene from the Tutorials project.
2. Select the truck by typing `milk*+` in the Selection text box (the + sign is used to pick the hierarchy).
3. Use the **f** key to locate the truck in the explorer (or use the **e** key to isolate the selection).

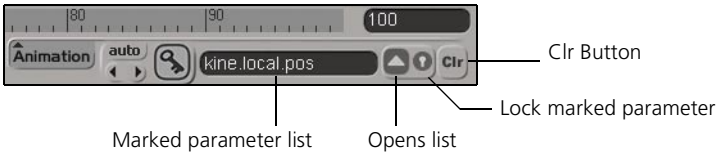
Mark the parameters

Now mark some parameters before setting keyframes. You can mark a parameter by simply clicking its name in the marked parameter list, a property editor, or an explorer. However, you will first mark a transformation, which is done a little differently.

4. Click the **x** Translation button to mark it.

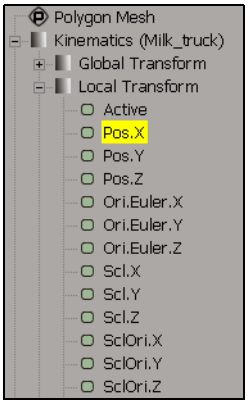
Activating any of the transformation tools (scale, rotate, or translate) automatically marks the appropriate parameters.

Notice the information entered in the marked parameter list box.

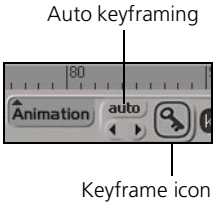


- Activating a transformation tool also removes any existing markings from other parameters, including other types of transformations. To ensure that your marked parameters are unaffected by using the transformation tools, click the Lock Marked Parameter icon.
- The **Clr** button clears the marked parameters for a given object.

5. To verify which parameters are marked, click on the arrow beside the marked parameter box. Marked parameters appear in yellow.



Keyframe with the k key



Once the parameters that need to be keyframed are marked, you must go to a specific frame (time), modify that parameter, and keyframe it. Let's do just that, as well as key the translation of the milk truck to the street intersection.

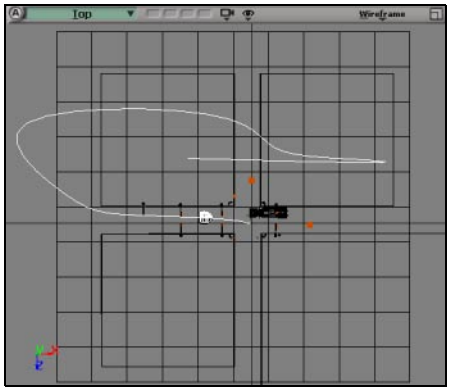
6. At this point, you should have the **x** translation. Making sure you are on frame 1, press the **k** key or click the keyframe icon to keyframe the initial position.

Use the auto-keyframing option

7. Now try using autokeying: Click the **auto** button, which automatically saves any subsequent marked transformations.
8. Go to frame 30 and move the truck again until you reach the intersection.
9. The second keyframe was recorded automatically, indicated by the keyframe icon turning red.
10. Deactivate autokeying by clicking the **auto** button again. Drag the playback cursor to see the truck move.
11. Beside the marked parameter in the **Local Transform** property page or the explorer (under **Kinematics** > **Local Transform** > **Pos**), an fcurve icon is now visible.

Whether you are in Global, Local, View, or Par mode (Transform panel), your keyframes are always saved in **Local Transform**. To save them in **Global Transform**, you must explicitly mark the Global Transform parameters in the marked parameter list in the Animation panel.

Animating the Spaceship along a Path



Now make the spaceship zip into town on a path.

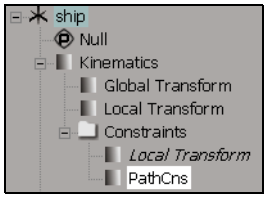
12. Enter `Ship+` in the Selection text box, then frame the ship in the explorer and one of the viewports.
13. In the Animate toolbar, choose **Create** > **Path** > **Set Path** and select the **Linear** and **Tangency** options. This creates a linear path translation fcurve (that is, with no ease in or out). Tangency ensures that the space ship's orientation will follow the path's curves. Keep the default **Start** and **End Frame** values and click OK.
14. Pick the path curve in the scene.
15. In the Path property editor, click the **Tangency** tab. Align the ship to negative **Z**: Enter `-1` in the text box. Enter `0` for **X** and **Y**.
16. Click the **Up Vector** tab and activate the up vector. Try animating the **Roll** parameter by moving the playback cursor to different frames and clicking this parameter's animation icon (green box) to set a keyframe where you like.

Linking Lamp and Spaceship Parameters

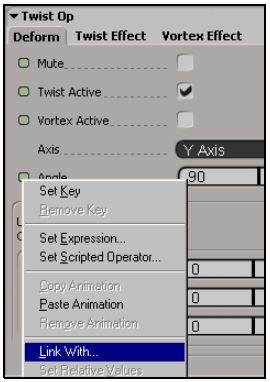
When one object's parameter is linked to another, SOFTIMAGE|XSI creates an expression establishing the dependency relationship. Then you must set relative values between the objects to trigger any kind of animation. Once you do this, a function curve representing the relative animation between the two objects is created and you can adjust it as you like.

You will link specific lamp and spaceship parameters so that the street lamps twist as the spaceship flies past them.

17. Type `lamp_6*+` in the Selection text box and frame it in both a viewport and the explorer.
18. Choose **Deform** > **Deform** > **Twist** to apply a twist to the lamp. Select to twist on the **Y axis**, then lock the property editor (keyhole icon) to keep it open.



19. Branch-select the spaceship and click the **Selection** button. Open the Path property editor of the spaceship in the explorer by expanding the **Kinematics** > **Constraints** > **Path Cns** and clicking the PathCns icon. Lock this property editor.



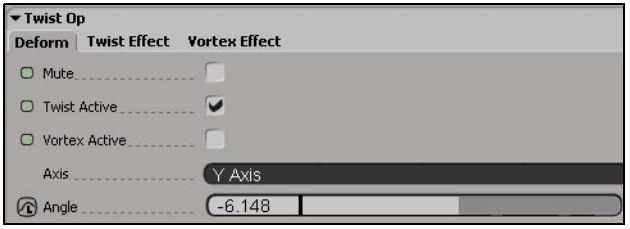
20. In the **Twist** property editor, right-click the Angle's animation icon and choose **Link With**.
21. In the explorer that is displayed, expand the ship model, then **Kinematics** > **Constraints** > **Path Cns**, and click **Path Percentage**.

    On the **Twist** property editor, the animation icon changes to include a little "L" to reflect the link.

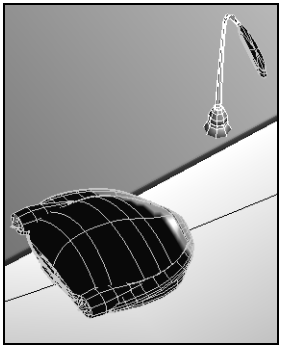
Adjust the lamp's twist in relation to the location of the spaceship

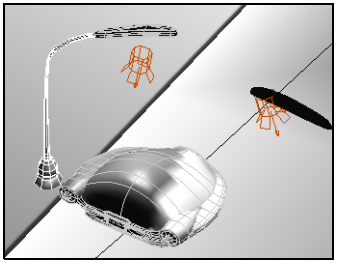
In the next step, you are going to adjust a twist in the lamp post as the spaceship flies by. You will need to find the appropriate frame where the action will take place, and then set a relative value with the correct twist effect.

22. Move the spaceship on its path by dragging the playback cursor. When it comes close to the traffic lamp, adjust the lamp's angle and save that state by right-clicking on the Angle icon and selecting **Set Relative Value**.



Repeat the operation as many times as you need (maybe for three or four frames).





Modify the ship's path percentage

23. Change the ship's path percentage so that the ship backs up along its flight path. You can do this in the **Path Constraint** property editor or in the animation editor.

    Notice that the twisting of the lamp is automatically affected by the change in the ship's motion because of the linked parameters you just created. This saves you from creating keyframes for this motion.

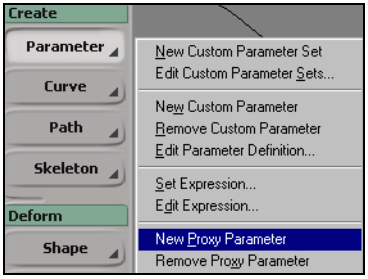
Something else to try: Right-click the Twist Angle's **Link** icon and select the **Animation Editor** to display the Link's function curve. Try different adjustments. You'll find that changing the function curve affects how quickly the street lamp twists in response to the passage of the ship.

Using Custom Parameters and Expressions

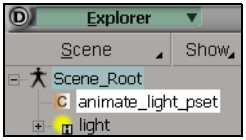
Now you'll create a custom parameter to animate the traffic light's diffuse material so that the traffic lights go berserk when the spaceship lands in the middle of the intersection.

A custom parameter set creates a property page that you customize by adding custom parameters. For example, you create a custom parameter that is a slider that goes from 0 to 1. This slider is stored in a custom parameter set and any parameters in the scene can be hooked by expressions or by linked parameters to this slider. A typical example is a face model with a custom property page that has phoneme sliders linked to all the model's mouth shapes (e, o, i, u, etc.).

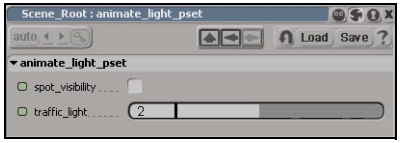
As opposed to the custom parameter, you can also use proxy parameters, which are clones of real parameters: when you change a proxy parameter, it changes the real thing. For example, you can gather any parameters (including custom parameters) you need to animate a scene to create a custom property page.



Animate the traffic lights



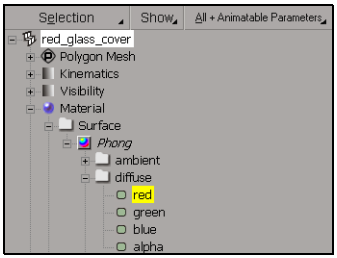
24. Select the scene root from the explorer.
25. From the Animate toolbar, choose **Create** > **Parameter** > **New Custom Parameter Set**. Name the custom parameter set `animate_light_pset` and click OK. This creates an empty custom parameter set at the scene root level.
26. With this new parameter set selected, create a parameter to go in it by choosing **Create** > **Parameter** > **New Custom Parameter**.
27. Name the parameter `spot visibility` and select **Boolean** as the **Value Type**. Click OK. This parameter will control the on/off state of the spotlight.
28. Create another custom parameter and call it `traffic light`. Specify a **Value Range** of 0 to 5 with a **Default** of 2 (leave the defaults for the rest). You will use this slider to control how dark or light the traffic light color will be.



29. In the explorer, choose **Show** > **Animatable Parameters**, and click on the animate_light_pset's icon (the square on the left side) to open its property editor. Lock this property editor. Notice that the parameters are there but have no effect yet.
30. In a viewport, frame the traffic light's red glass object (type `red_glass_cover*` in the Selection text box and press **f**). Make sure that **Shaded** mode in selected in this viewport.

Affect the traffic light's Diffuse parameter

Next you'll link the spotlight's **Visibility** and the traffic light's **Diffuse** parameters on the new property page to the appropriate parameter of the objects you want to affect. You will do this by using an expression.

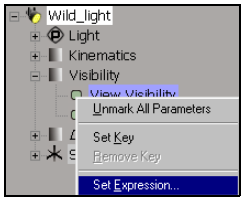


31. Follow this path in the explorer: **red_glass_cover** > **material** > **Surface** > **Phong** > **diffuse** > **red** (make sure **All** + **Animatable Parameters** is selected in the explorer).
32. Right-click on "red" and choose **Set Expression**.
33. In the expression editor, clear any existing values in the lower pane.
34. Click the **Object** button, then select the **traffic light** custom parameter in the **animate_light_pset** customer parameter set. Click **Apply** and close the expression editor.

    The expression should look like this:

    ```
    animate_light_pset.traffic_light
    ```

Affect the spotlight's Visibility parameter



35. Select the corresponding spotlight in the viewport (called Wild_light).
36. In the explorer, expand the spotlight's node. Right-click the **View Visibility** parameter of the **Visibility** property and choose **Set Expression**.
37. Clear any existing values in the expression editor, then click the **Object** button. Select the **Spot visibility** custom parameter of the **animate_light_pset** customer parameter set, then click Apply and close the property editor.

    The expression should look like this

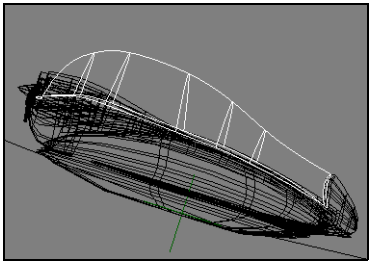
    ```
    animate_light_pset.spot_visibility
    ```

38. Drag the custom parameter sliders to see the effect in the viewport.

For a better result on the "red_glass_cover," apply the same traffic light custom parameter on its **Ambient** parameter.

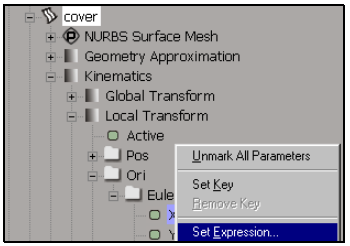
Opening the Spaceship's Cover Using Expressions

Now you need to create an expression that makes the ship's cover open gradually as it comes in for a landing.



The expression compares the vertical distance between the ship and the reference object (here called **cover_trigger**). The cover will open only if the ship moves below the reference object. The value of the opening angle of the cover is proportional to the distance between the ship and the reference object, so that, as the ship lands, the distance between the two increases and the cover opens more.

39. In a viewport, select the spaceship's cover.
40. In the explorer, expand the spaceship cover's hierarchy as follows: **Kinematics** > **Local Transform** > **Ori** > **Euler**.



41. Right-click on **X** and choose **Set Expression**.

Use the **Function** tab to help you write the expression. After writing the first component of an element (including the trailing period), press F12 to help you complete it. For example, type `ship.` and press F12.

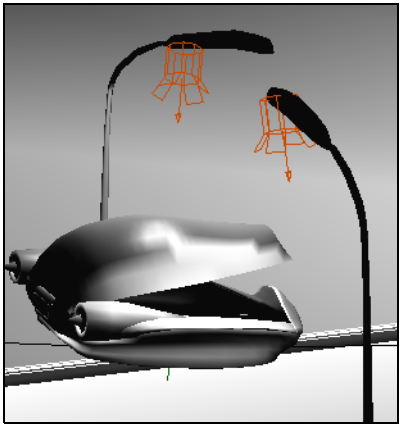
You can also use Ctrl+c and Ctrl+v (copy/paste) to complete the expression more quickly.

42. Enter the following expression:

```
cond( ship.cover.kine.global.posy < Cover_trigger.kine.global.posy,
-( Cover_trigger.kine.global.posy
- ship.cover.kine.global.posy )  * 4, 0 )
```

43. Click **Apply** and close the expression editor.

Move the playback cursor and see how the cover opens as the spaceship comes in for a landing.



Conclusion

You've been introduced to a number of basic animation tools in SOFTIMAGE|XSI, including keyframing, marking, and path animation. You have also seen how easy it is to link the animation of one parameter to another, and the power that expressions offer via custom parameters. Try using proxy parameters instead of custom to use a clone of a real parameter.

For more information, see the following chapters in the *Animating* guide:

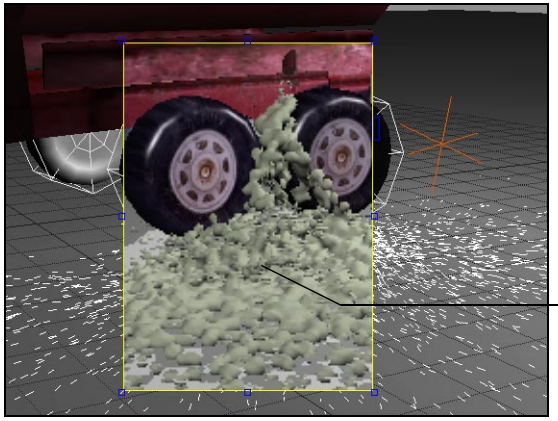
- *Chapter 2: Animating with Keys*
- *Chapter 5: Animating along Paths and Trajectories*
- *Chapter 7: Animating with Expressions*
- *Chapter 9: Linked Parameters*
- *Chapter 10: Custom and Proxy Parameters*

Section 5 Simulation

Tutorial 7: Particles—Oozing, Oily, Slimy Sludge

Use particle effects to create a stream of oil sludge leaking from the truck's tank. The sludge is flowing from the valve onto the wheel, and then finds its way to the ground.

Particles are fully integrated in SOFTIMAGE|XSI, which means that they can be affected by light, color, shadows, reflections, and refractions. Furthermore, you do not have to create "dummy" obstacles when you define them: you can pick the actual objects in the scene.

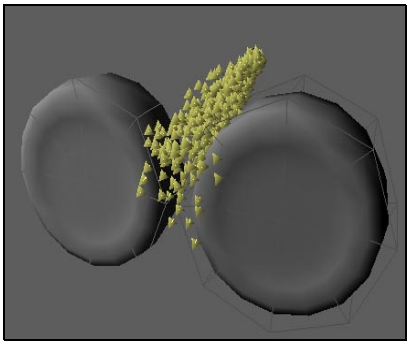


This tutorial shows you how to:

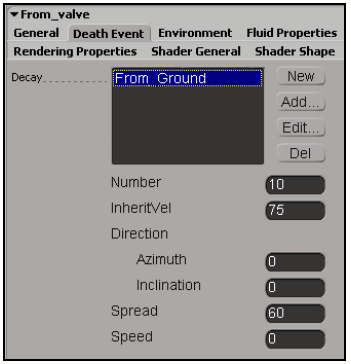
- Create a particle system, including setting up an emission and a particle type.
- Create a death event to have one particle type die off as another one begins.
- Set up obstacles for the particles to flow over.
- Apply the blob shader to make the particles look like thick liquid.

Overview

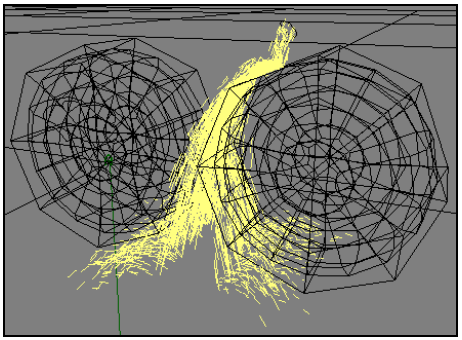
1 **Create the particle system.**



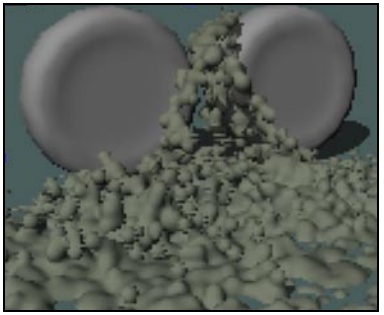
2 **Create another particle type to be emitted.**



3 **Set up obstacles for the particle flow.**



4 **Apply the Blob shader to the particles.**

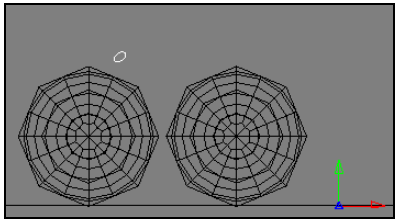


Creating the Particle System

1. Open the **SIM_e1_Leak** scene from the Tutorials project.
2. You can hide extraneous elements in the scene by choosing **Layers** > **Layer Control**. Deselect **View** for the **other_tires_n_stuff** layer.

Create the particle cloud

First, you must create the particle object itself (called a particle cloud) to define the basic particle system.



3. Select the small disc (the leaky valve) above the tires in the C viewport.
4. From the Simulate toolbar (press 4), choose **Create** > **Particles** > **From Selection**.
5. In the ParticlesOp property editor that opens, select **Interactive** as the Execution State if you want to have the particle animation update interactively as the particle parameters are changed. If not, you will have to move the playback cursor to see the effect of a parameter change.
6. Leave the default values for **Start Frame, Duration**, and **Particle Percentage** as is. Lock the property editor to keep it open (click the lock icon in its upper-right corner).

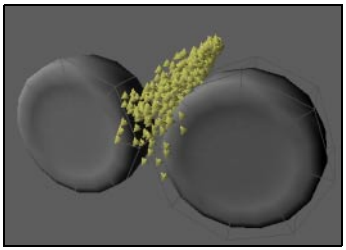
The Particle Percentage controls the number of particles that originate from emitters belonging to a particle cloud.

Define the particle type

Now rename and define the main particle type, which are the particles originating from the tank's valve:

7. On the Emission page, beside the **ParType** text box, click **Edit**, then type `From_valve` in the **Name** text box. Close that property editor.
8. Back on the main Particle_Type > General property page, enter these values for the following parameters:
   - **Maximum Life**: 4 (seconds)
   - **Maximum Life Variance**: 1.2
   - **Size:** 0.1

Switch to Shaded display mode in a viewport to see the size of the particles.



9. On the **Emission** page, leave the default values for **Generation** and **Emission Direction** as is: **Generation** sets the point of origin of the emitted particles, and the **Emission Direction** sets their direction.
10. The other parameters control the density, speed, and spread angle of the emitted particles. Enter the following values:

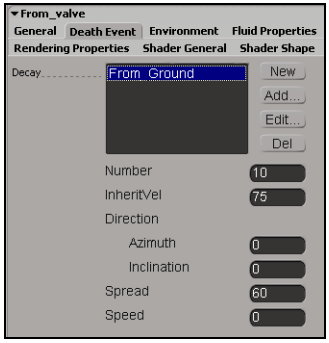
    - **Rate**: 350 and its **Variance**: 50
    - **Spread**: 15 and its **Variance**: 7.5
    - **Speed**: 1.4 and its **Variance**: 0.5

    Leave the rest of the parameters at their default values (**Inherit Vel** considers the source's velocity, if it is in motion).
11. Move the playback cursor to frame 40 to see what the particles look like.

Setting Up a Second Particle Type for the Death Event

When the From_Valve particles hit the ground and die, a second particle type is emitted at the ground. You need to define this second type:

12. On the Particle Type > Death Event page, click **New** to create the decay particle.
13. Select **Particle_Type** in the box and click **Edit**.
14. In the Particle Type property page that opens, enter the name `From_Ground` for the decay particle.
15. Set the **Particle Size** to 0.08, leave the other parameters as they are, and close the property page.



Now you have to define how the From_Ground particles will behave after the From_Valve particles die off.

16. Complete the **Death Event** information:

    - Set **Number** to 10. This means that for every dead From_Valve particle, 10 From_Ground particles are generated.
    - Set **Inherit Vel** to 75 so that every new From_Ground particle inherits 75% of the velocity from the From_Valve particles.
    - Set **Spread** to 60.
    - Leave the default values for the other parameters and close the property editor.

Applying Gravity to the Particles

Apply a natural gravitational force that will affect the particles.

17. Choose **Get** > **Force** > **Gravity** from the Simulate toolbar and accept the default **Amplitude** value of 9.8 (earth's gravity).

If the particle cloud is selected when you create a force, it is applied automatically to that cloud—you don't need to apply it with the **Apply Force** command.

18. To see the gravity icon in the viewport, click the eye icon in the title bar and choose **Control Objects**. Remember that you can keyframe any kind of **S\R\T** transformations on the gravity icon.
19. To apply the force, select the cloud icon and choose **Modify** > **Environment** > **Apply Force** from the Simulate toolbar. Pick the gravity icon to apply it to the particle cloud.

Setting Up Obstacles

Next, specify the obstacles that the sludge can slide down the tires and the ground.

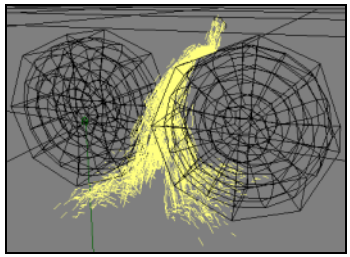
20. Select the cloud and choose **Modify** > **Environment** > **Set Obstacle** from the Simulate toolbar. Pick both tires, then the ground, and right-click to finish.

At this point, if you enter values in the **Obstacle** property editor, they will be applied to all selected obstacles. Instead, to give the obstacles unique properties, select them individually in the viewport while keeping the **Obstacle** property editor open (click its Lock icon), then modify their values:

21. Select the ground and change its **Obstacle Type** parameter to **B-Plane**.
22. Enter these other values for the ground:
    - **Friction**: 0.5
    - **Elasticity**: 0.35

Select **Double-face Collision** for when the particles hit both sides of an object's face.

23. On the **ParObstExtSparks page**, select From Valve in the box and change its **Mode** to **DECAY**.
24. Shift+click to multi-select the tires and use **Actual Shape** for the **Obstacle** parameter. This setting creates a more precise simulation by considering the actual geometry of the obstacles.



25. Enter these other values for the tires:
    - **Friction**: 0.75
    - **Elasticity**: 0.25
26. Close the Obstacle property editor.

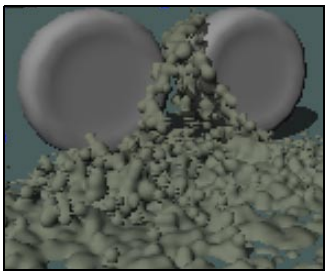
Play the simulation to see the particles going around the tires and bouncing on the ground.

Applying the Blob Shader

27. Select the cloud, then choose **Get** > **Shader** > **Blob** to apply a shader that makes particles look somewhat like metaclay objects. To open the blob shader's property editor, select the cloud icon and choose **Modify** > **Shader**.
28. Draw a render region (press **q**) around the particles and note how they are casting shadows.

    Notice that the particles are too big coming out of the valve. You can animate them to grow from a small size to the normal size. To do this:

29. In the explorer, expand the cloud node so that you see Particle Cloud, and click on its icon to open the ParticleOp property editor.
30. Click on the **From_valve** tab and animate the size of the **From_Valve** particle type:
    - Move the playback cursor to frame 1, set the **Size** is 0.07, and click the animation icon beside this parameter to set a keyframe.
    - At frame 30, set the **Size** 0.1, and set another keyframe.
    - Select the **Age%** mode beside the **Size** parameter: this means that the variation of the size of the particle is proportional to its age—as it ages, it increases in size.



Conclusion

You have learned how to use the particle system in a basic way, as well as using natural forces and setting obstacles. You have also seen how the particles' appearance changes when using the blob shader. Try using the Particle shader for a different effect.

You can change the From_Valve and From_Ground particles' colors on the Rendering Properties page in each of their Particle Type property editors. In the blob shader's property editor, you can adjust the color, transparency, etc. (if you want to control the color with the latter method, you must deselect the **Diffuse From Particle File** and **Ambient from Particle File** options.

For more information, see *Chapter 17: Particles* in the *Animating* guide.

Tutorial 8: Fluid—Simulating a Liquid

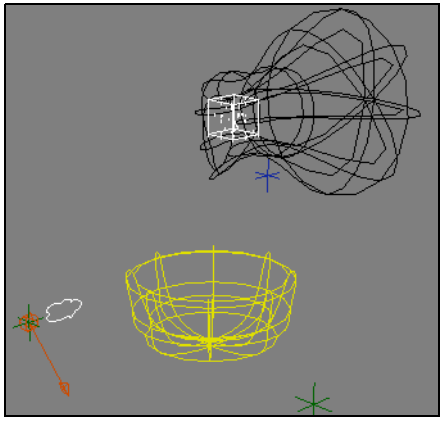
You will use “fluid” to create a liquid effect for a thick liquid as it is poured from a jug into a bowl.

This tutorial shows you how to:

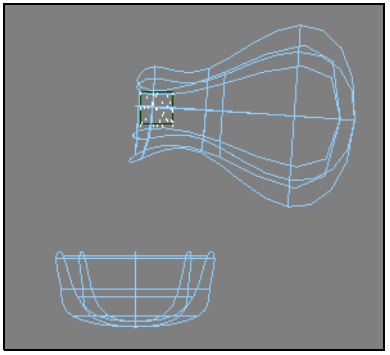
- Use the fluid particle simulator.
- Set up an obstacle.
- Adjust the fluid parameters to define its appearance.

Overview

1 **Create fluid pouring from the jug.**



2 **Set the jug and bowl as obstacles.**



3 **Adjust the fluid's appearance.**

Creating the Fluid and Setting an Obstacle

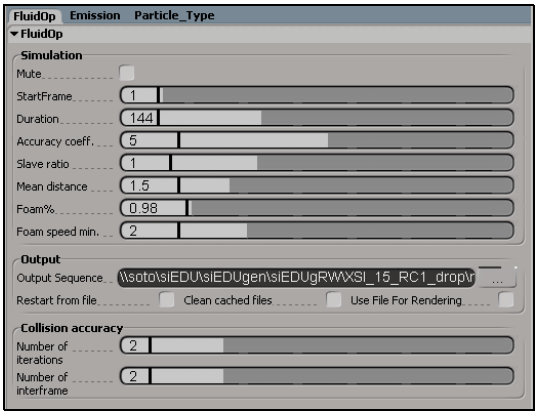
With fluid, you create a specific space for its particle cloud using an implicit object, such as a cube, disk, or square.

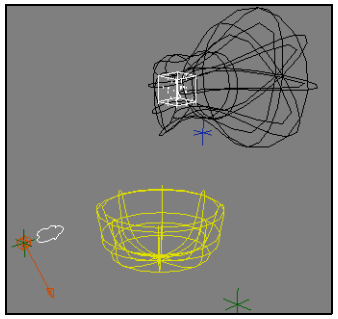
1. Open the **SIM_e3_Bowl** scene from the Tutorials project.
2. Choose **Create** > **Fluid** > **From Cube** from the Simulate toolbar.

   The FluidOp property editor opens—lock it (click the Lock icon) to keep it open while you're working.
3. Close the Emission page (click the arrow beside the Emission name) in the property editor, as these settings have no influence on fluid (except for the Speed, which you don't need to set here).

Set the lifetime and number of particles

4. On the Particle Type > General page, set the **Maximum Life** to 12 (seconds).
5. Click the "Go to last frame" button on the timeline to see the results. This calculates the simulation without having to play every frame. You can do this every time you make a change and want to see the resulting simulation.
6. Set the **Mean distance** to 1.5 for previewing—this takes less time to calculate. This parameter controls the number of particles by determining the distance between the main *(master)* particles in a set space. The lower the value for this (the smaller the space between particles), the greater the number of particles.





Set up the obstacles

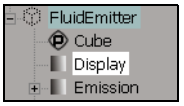
7. At frame 1, position the emitter cube (not including the cloud icon) to be in the middle of the jug neck.
8. Select the cloud icon and choose **Modify** > **Environment** > **Set Obstacles** from the Simulate toolbar. Pick the jug and bowl as obstacles (they turn blue) and right-click to end.
9. In the Obstacle property editor, set **Obstacle Type** to Actual Shape so that the collision is more accurately simulated (this sets the obstacle type for the jug and the bowl). Then close the Obstacle property editor.

Set the display

When you play back the simulation, you see that the particles are barely distinguishable.

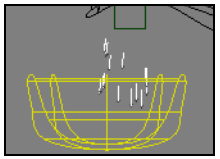
10. In the C viewport, select **Shaded** mode and deselect **Override Object Properties** in the display type menu.

The display properties of the jug and bowl have already been changed to show them in wireframe, but you can change the display properties for the emitter cube.



11. Select the emitter cube. In the main command area, click **Selection** from the Select panel and expand the tree so that you can see the Display node. Click the Display icon (the shaded box) to open the Display property editor.
12. Change all parameter settings that are Automatic to Wireframe.
13. Solo the C viewport by clicking its letter so that it turns green.

Applying Gravity to the Liquid



When you play the simulation, you'll see the particles floating about and not pouring into the bowl. That's because there's no gravity—time to create some!

14. Select the particle cloud and choose **Get** > **Force** > **Gravity** from the Simulate toolbar to create it a force and apply it to the cloud.
15. Change the Amplitude value to `50` for a strong gravitational pull, and close the property editor.

Changing the Viscosity

The particles don't stick together. so you need to change the viscosity (thickness).

16. On the **Particle Type** > **Fluid Properties** page, change the **Viscosity** value to `0.7`.

Setting Up the Liquid's Look

Take a look from another perspective:

17. Solo the B viewport, and in it select **Shade** mode and deselect **Override Object Properties**.

Now that the motion looks okay, concentrate on the look of the fluid.

18. Create a render region (press **q** and drag) at frame 85 to see what the liquid currently looks like. Not very much there yet, is there?

19. Change the **Particle size** to `0.5` and the **Mean Distance** to `0.3`. This makes smaller particles and more of them.

You can also increase the number of particles by increasing the size of the cube. This is because the **Mean Distance** is the space between the particles, not the actual number of particles. If you change the volume in which the particles exist, they increase to fill up that space.

20. On the **Rendering Properties** page, give the particles any color you like.

When you're satisfied with the results, play the simulation.

Conclusion

You have learned how to use the fluid particle simulator for creating a basic fluid. To complete the effect, apply the fluid shader by selecting the particle cloud and choosing **Get** > **Shader** > **Fluid**. To open the fluid shader property editor (or any shader currently applied to an object), select the particle cloud and choose **Modify** > **Shader** from the Simulate toolbar.

For more information, see *Chapter 17: Particles* in the *Animating* guide.

See the next page for some fluid recipes.

Fluid Recipes

If you want to have a more watery effect, here are some recipes. These settings are found in either the FluidOp property editor or the fluid shader property editor.

The **Slave Ratio** value lets you control the number of instances of particles to a “master” particle. It augments the effect without adding to the number of particles requiring calculation, but it adds a “foamy” effect to the particles.

Recipe 1: Rolling Water

Create an emitter: disk with radius = 1
Mean distance = 0.2
Slave Ratio = 2
Emission speed = 5
Particle size = 0.1
Attenuation = 0
Blend = 5
Density correction = 0
Voxel size = 0.1
Anisotropy = 1

Recipe 2: Splashing Water

Create an emitter: disk with radius = 0.5
Mean distance = 3.1
Slave Ratio = 300
Emission speed = 20
Particle size = 0.04
Attenuation = 0
Blend = 10
Density correction = 0.2
Voxel size = 0.06
Anisotropy = 1

Section 6 Character Animation

Tutorial 9: Character Setup—Making a Man Out of William

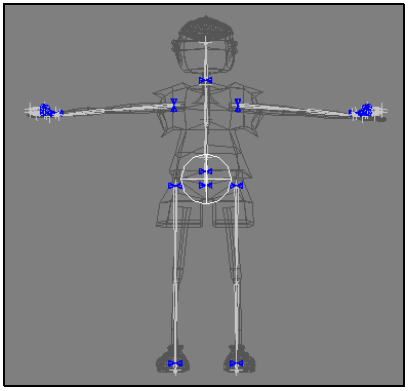
Though not a *super* model, William is a model nevertheless. But he doesn't have a skeleton. Here's your chance to build one, and an animation rig as well.

This tutorial shows you how to:

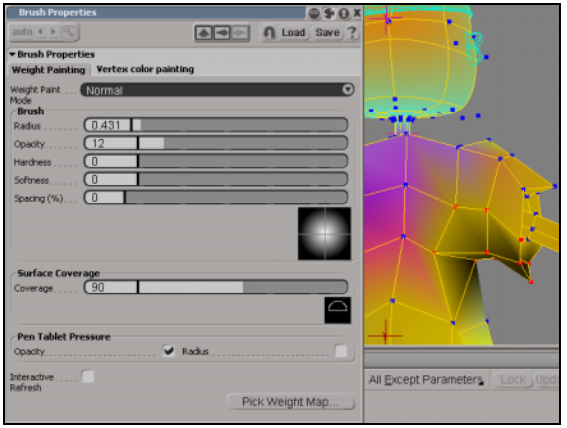
- Create chains and modify them, including using duplicate symmetry.
- Create and assign up vectors to arms and legs to help prevent flipping when using IK.
- Set up a simple rig with position constraints.
- Set rotation limits for proper limb movement when using IK.
- Use compensation to create offsets in constraints.
- Assign an envelope and edit the envelope weight map.
- Use the Paint tool to change the envelope weighting.
- Reassign certain envelope points locally to share the weighting with more than one bone.
- Use mesh subdivision to create a high-resolution model from a low-resolution one.

Overview

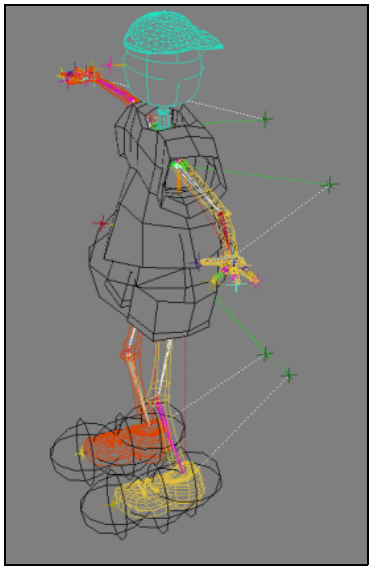
1 Create a skeleton and set up the hierarchy.



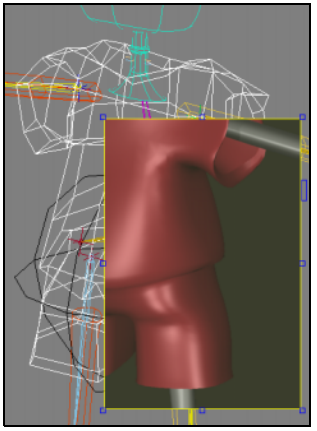
2 Assign an envelope to the skeleton and weight parts of it.



3 Set up nulls and rigs for controlling parts of the character.



4 Make a high-resolution envelope model from a low-resolution one.



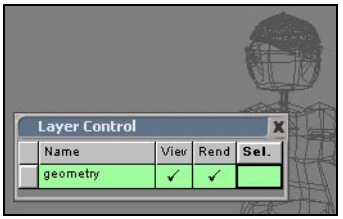
Creating a Skeleton for William

While there are probably dozens of different ways to set up characters, this is one basic setup that you can use for most typical biped animations.

Setting up a character must come only after determining what kind of movement is required. William is not going to be doing any backflips or the like, so he doesn't require a complicated setup. However, you will need him to walk well and do most basic movements.

1. Open the **CHA_e1_william** scene from the Tutorials project.

   Right now, William is a spineless shell of a boy. He needs a backbone, as well as a host of other bones to give him some structure.

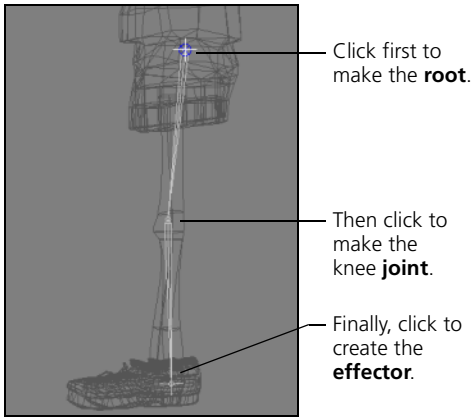


Make geometry layer unselectable

To make it easier to work with only the skeleton, make the model layer (envelope) unselectable.

2. Choose **Layers** > **Layer Control** in the main command area and deselect the **Sel.** column for "geometry." When unselectable, the wireframe objects are displayed in gray.
3. Create a new layer to work in. With nothing selected, choose **Layers** > **New Layer** and give the layer a name such as `skeleton`. By default, this layer is the default active layer. You will continue to work in this new layer and all new created objects will be in this layer.

You can select a layer from the the Layer menu in the main command area to activate it.



Create William's legs and feet

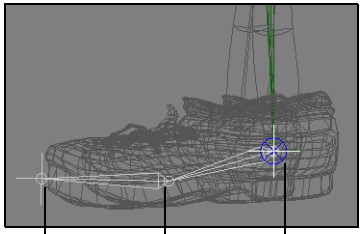
Start with a leg:

Note that William is not anatomically correct! The size, position, and number of bones that you will be using in this lesson are approximated to get you up and running quickly.

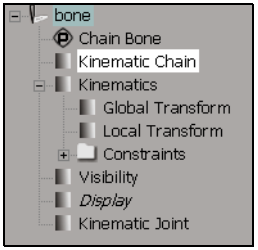
4. You should always use an orthographic view when drawing chains. You will draw the first chain in the Right view. Make it fullscreen (press F12 or click in the viewport's resizing icon in its right corner) and zoom and pan to frame the leg in the viewport (remember that the model is not selectable).
5. From the Animate toolbar, choose **Create** > **Skeleton** > **Draw 2D Chain**. Start the chain at the hip, click to make a joint at the knee, and click again at the ankle. Middle-click to end this chain but keep the **Draw 2D Chain** command active.

The chains don't need to be exactly inside the model when you draw them—you'll position and resize them later.

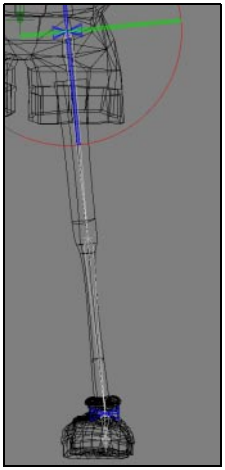
- When drawing a chain, XSI does not place the point until you release the mouse button. If you hold the mouse button down while moving the point, you can see exactly how the chain will look before finishing it.
- Make sure that the knee angle is clearly defined in the correct direction—an incorrect angle will make the leg bend the wrong way.

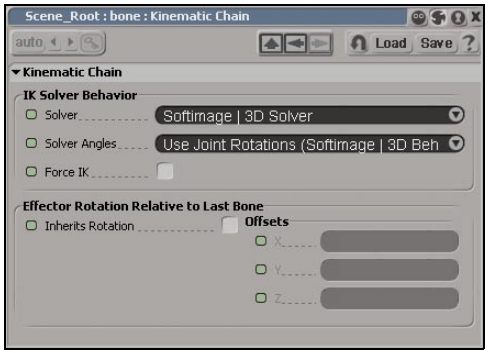


6. Use the illustration on the left as a reference to create the foot chain. Still in the Right view, left-click to start drawing a new chain: create a 2-bone foot chain in the appropriate place, starting at the ankle. Right-click to end chain-drawing mode.
7. Select the first bone of the leg chain and click the **Selection** button in Select panel of the main command area (this shows the properties for only the selected element). To open the Kinematic Chain property editor, click on the icon (little shaded box) beside its name.

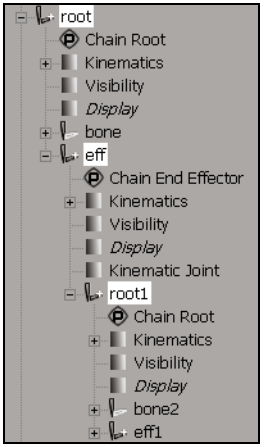


8. In the property editor, deselect **Inherits Rotation** so that the leg's effector can rotate independently of the last bone in the leg chain. Later you'll do the same thing on the arm effector so that it can rotate freely (as a normal wrist would do) without the default constraint to the last bone.



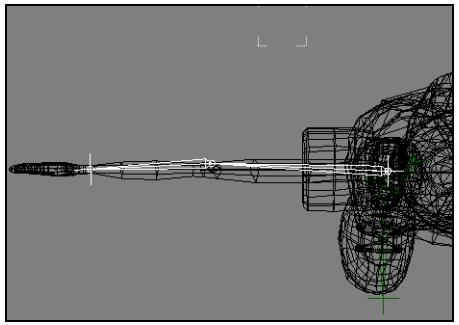


9. In a Front view, translate and rotate the leg and foot chain so that it fits inside the character's model.
10. In an explorer, expand the leg chain to see its effector. Drag+drop the foot chain on the leg's effector node to make it a child of the leg's effector. The resulting hierarchy should look like something like [below], with **root** being the leg chain and **root1** being the foot chain under the leg's effector (**eff**).



In the explorer, you can rename any of the chain elements (such as roots) for easier identification. Simply right-click on the name of the chain element and choose **Rename**.

11. Branch-select the leg and duplicate the hierarchy by choosing **Create** > **Skeleton** > **Duplicate Symmetry**.



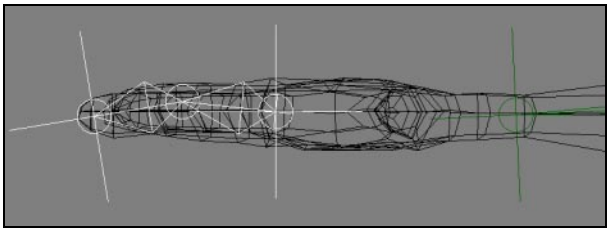
12. Select the **YZ** plane in the Symmetry dialog box, then move the new leg into position inside the model. You will need to rotate the leg and feet bones independently until they fit properly.

Create William's arms and hands

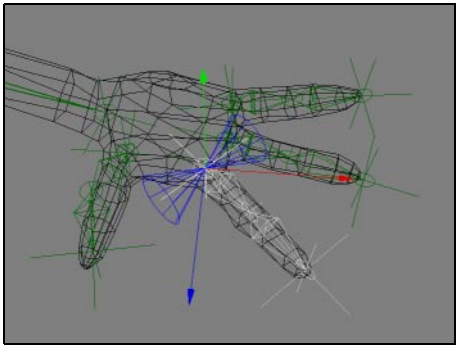
13. In the Top view, make a 2-bone chain for an arm, starting at the shoulder.
14. Make the arm effector free from the last bone rotation constrain, so that it can rotate freely on its own. Select the arm first bone and press Enter to open the Kinematic Chain property editor. Deselect **Inherits Rotation** as you did for the leg effector.

You can adjust the bone length if you need to. Select the bone and press Enter (or click the **Selection** button on the main command area). In the Chain Bone property editor, drag the slider for **Length**.

15. To create fingers, draw a 2-bone middle finger chain in the Front view and position it.



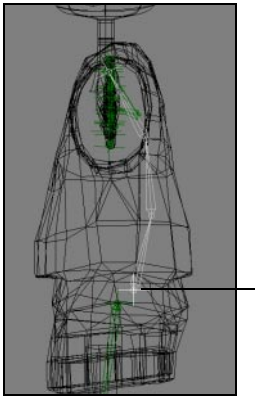
16. Duplicate this chain three times (press **d**), then rotate and position each bone in each finger appropriately.



You'll need to rotate the thumb locally (click the **Local** button in the Transformation panel) so that it bends naturally in the opposite direction that the fingers do.

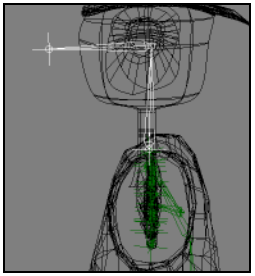
17. Using the explorer again, make all fingers the children of the arm effector.
18. Branch-select the arm chain and fingers and duplicate them all with **Create** > **Skeleton** > **Duplicate Symmetry** (YZ plane).

    Duplicating a chain ensures that they are using the same preferred angle and will have the up vector in the same place (in this case, both up vectors will be behind the arms instead of one in front and one behind).



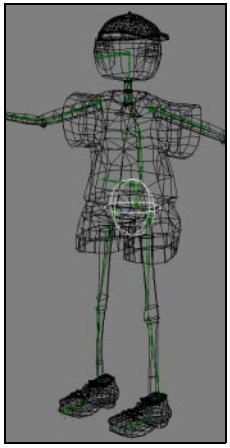
Give William some backbone ... and a head

19. In the Right view, create a 3-bone chain for the backbone, as shown on the left, starting at the base of the spine.
20. Starting at the base of the neck, create a 2-bone chain for the head, with one bone in front of the face to make it easier to manipulate the head.

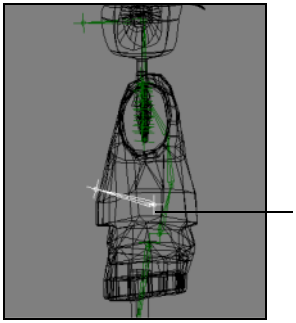


Create a control object for the body's trunk

21. Get an implicit sphere and position it in the hip area, as shown on the left. This is the control object to help animate the body's trunk. Rename this new implicit sphere to `COG` (center of gravity).
22. Using the explorer, make this COG control object the parent of the spine and the two leg roots.
23. Now make the spine's effector the parent of the head and two arm roots.



24. Create a single-chain bone in the stomach for manipulating and deforming the shirt in that area and make it a child of the first (bottom) spine bone.



Create a stomach bone, then make it a child of the first spine bone.

25. Branch-select the COG control object to verify that the hierarchy is complete and correct (all the chain elements should be selected).

Assigning the Envelope to the Skeleton

Now that you have given William some structure, you can attach the model (envelope) to his skeleton.

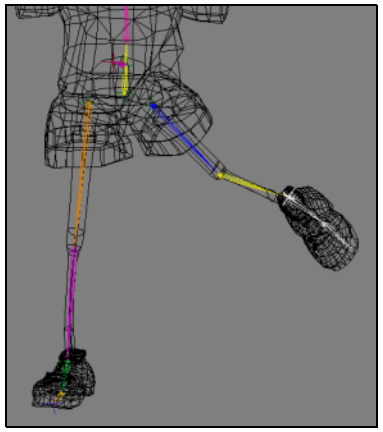
26. Make the geometry layer selectable again by choosing **Layers** > **Layer Control** and clicking in the **Sel**. column for the geometry layer.

27. Branch-select the model (envelope) and choose **Deform** > **Envelope** > **Set Envelope**.

28. Expand the COG node in the explorer, and middle-click each root object to include them within the envelope. Right-click to end the picking.

29. In the Automatic Envelope Assign property editor that appears, leave the settings as they are and close the property page. The envelope is now attached to the skeleton—notice the bones are now colored: This shows the influence that each bone has on its corresponding points on the envelope.

In the property editor, the **Normal based** Assignment Method can be effective when skeletons are inside the geometry. In this example, some geometry (like the shorts) have double geometry (facing in and out) that prevents weighting using the normal based option. Otherwise, this can be a great timesaver for subjects such as a hand with fingers.



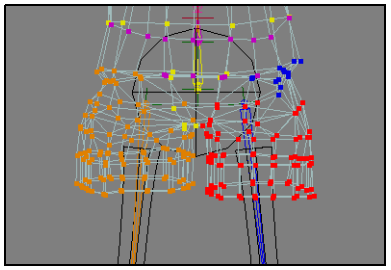
Weighting the Envelope

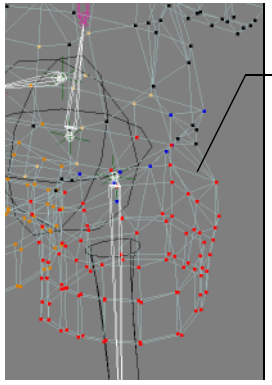
30. Branch-select and drag the leg effector (as in the illustration at left) to see how the movement affects the shorts. As you can see on the left, some weighting adjustment is required around the inner leg of William's shorts.

Make sure you middle-click the leg effector so that your selection includes the leg's effector *and* the foot chain.

31. In the Front view, select the T-shirt model (shirt and shorts are one model).

32. Press **t** and tag the points on one leg of the shorts as shown by the red points in the following illustration. Drag a rectangle around the area, then add points to or delete points from the selection.





Tag points in the area to modify.

Then choose **Reassign Locally** and pick the bones that will influence the weighting in that area.

33. Before you do the local re-assignment, choose **Chain Element** in the Selection filter on the main command area—this way, you can only select the skeleton's elements.

34. Choose **Deform** > **Envelope** > **Reassign Locally** and pick the bones for which you want to influence the weighting in this area: the stomach bone, the first (bottom) spine bone, and the two thigh bones. Right-click to end the picking.

35. When you move the leg, you can see that the other leg has the same problem: repeat the envelope reassignment steps for this leg (select the points that are influenced by a wrong bone and then reassign locally to the right bones only).

William's head and neck also need a little adjustment.

36. Branch-select the head and tag all points down to the "Adam's apple" and reassign the envelope locally as you did for the legs. Pick the two head bones.

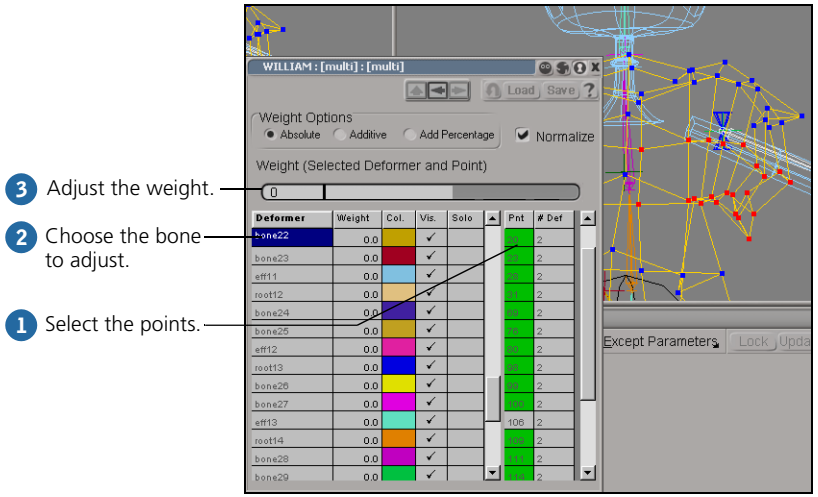
Use a weight map on the shirt sleeves

When you move an arm around, you can see that problems occur in the shirt, especially the armpit area. You can fix this using a weight map to redistribute the influence that the bones have on the envelope's points.

37. Move the arm around until it's in a position that is between extremes. Select the upper arm bone and take note of its name in Selection text box. You'll need to know exactly which bone to select in the Envelope Weight Map property editor (a skeleton has a lot of bones with no distinctive names!).

38. Select the T-shirt model and choose **Deform** > **Envelope** > **Edit Weights** or press Ctrl+e to open the Envelope Weight Map property editor.

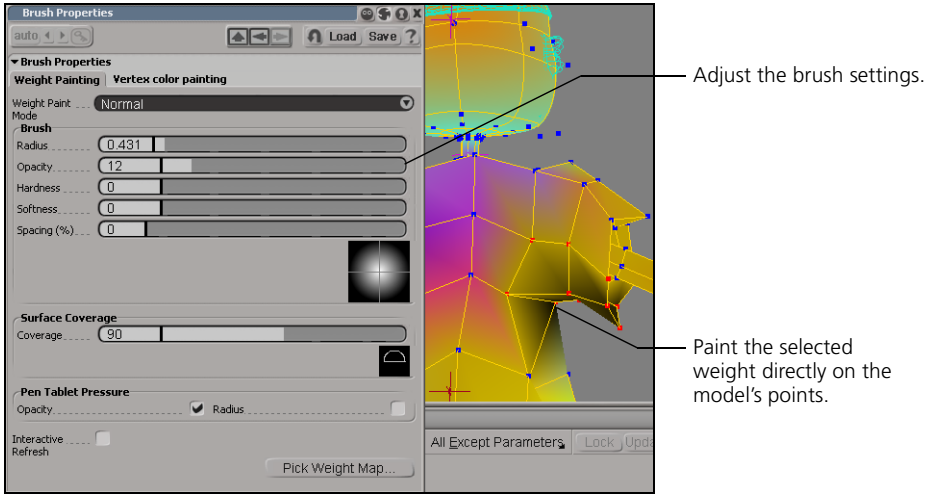
39. Find the upper arm bone you had selected and select it in Deformers list (bones, in this case)— notice the unique color coding for each bone.



40. To do the major reweighting of the envelope, tag the points of the problematic area (the armpit area, in this case). In the Envelope Weight property editor, Shift-select all the points in the Pnt column, select **Add Percentage**, and modify the weight percentage on the selected tagged points.

41. To do minor weighting adjustments, use the Paint tool on the weight map— press **w** to activate the Paint tool. Notice how the points' colors on the envelope correspond to the bones: the spread of each color shows the influence of each bone on the points.

42. Press Ctrl+w to open the Brush Properties editor. Make the brush **Size** smaller and set the **Opacity** (strength) to 12 for a lighter weight.

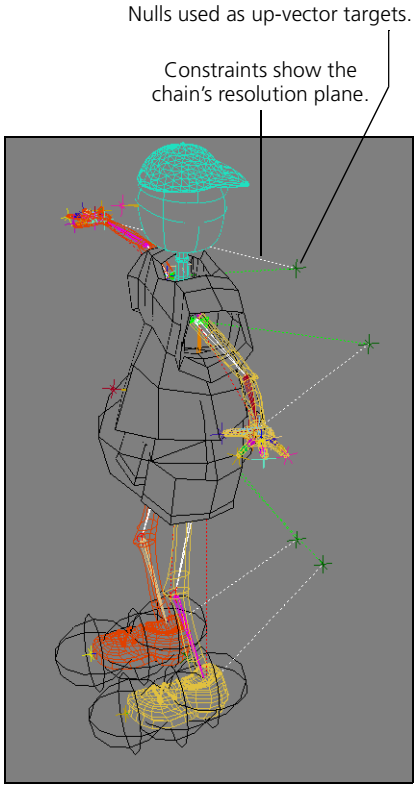


43. You want to decrease the influence that the upper arm bone has on the shirt. To do this, right-click and drag in small strokes around the arm bone. Orbit in a camera view to see the armpit from beneath.

44. Fine-tune the weighting around the shoulder and chest areas—test this by moving the arm around.

45. Repeat the same steps for other arm. Use the following illustration as a guideline for the weighting repartition between bones.

Point Reassignment Overview



Creating Up Vectors for the Arms and Legs

Up vectors help to constrain the orientation of a chain to prevent flipping when invoking inverse kinematics. This constraint provides control of the orientation of your chain's resolution plane. You will use a few nulls as up-vector constraints for the arms and legs.

46. Get a primitive null, duplicate it, and position each null behind each arm, as shown on the left.
47. Select the upper arm bone, choose **Create** > **Skeleton** > **Chain Up Vector**, and pick the null. A plane of movement is displayed showing how the arm chain can move.
48. Repeat this process for the other arm.
49. Duplicate the nulls twice and position each null behind each knee, as shown.
50. Select the thigh bone, choose **Create** > **Skeleton** > **Chain Up Vector**, and pick the null. Repeat for the other leg.
51. Select one of the nulls and translate to see how the chain reacts. You can animate the nulls too!

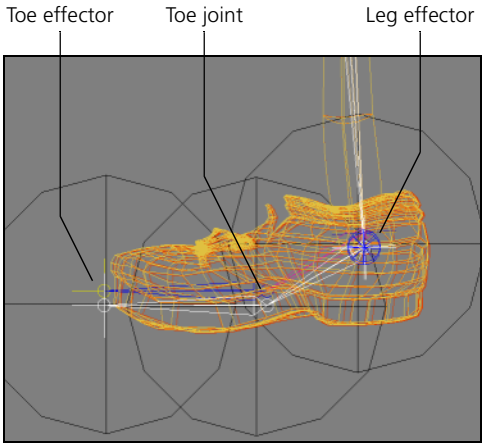
Making a Control Rig for the Feet

Control rigs are useful tools that help you easily control various parts of a skeleton. They are usually placed just outside of the skeleton and envelope so they are easier to select than effectors and bones. In the next steps, you will use simple implicit spheres to manipulate the foot by constraining various parts of the foot chain to these spheres.

52. Create a control object by getting an implicit sphere with a radius of 4 (easier to select), and move it on top of the ankle, over the leg's effector.
53. In the Right view, duplicate the sphere twice: move one over the toe bone, and one over the toe effector.

Be as precise as possible when positioning the control objects, because you will later constrain the chain elements to them. If the distance between the chains and the control objects is too great, you may get unpredictable position jumps.

Position each sphere over these parts:



54. Making all these spheres part of a single hierarchy gives you a quick and easy selection of the rig. Create a cascading foot hierarchy:

    - Make the toe sphere a child of the middle foot sphere.
    - Make the middle sphere a child of the ankle sphere.

Cascading the elements into each other like this means you can branch-select the ankle sphere to select the whole foot rig and branch-select the middle sphere to select the lower part of the rig.

55. Branch-select the ankle sphere (top parent).
56. Duplicate this control setup hierarchy (press Ctrl+d), and position it on the other foot.
57. Select the leg's effector and constrain its position to the ankle sphere: choose **Constrain** > **Position** in the main command area and pick the corresponding sphere. Make sure it's the effector you have selected and not the root of the foot (check the Selection text box).

58. In the same manner, constrain the toe bone to the sphere in the middle of the foot, and constrain the foot's effector to the sphere at the toe.

Branch-select the leg's effector to adjust its rotation and reposition the foot's control objects after setting the constraints.

Branch-select and move the foot control setup around to see how the foot moves.

59. Set up the same constraints for the other foot.

60. Branch-select the model hierarchy (envelope) and choose **Create** > **Model** > **New** from the Model toolbar. Name it `Willam_Model` and use the default **Internal** storage.

61. In the explorer, select the implicit sphere objects (including the COG sphere) and drag them under the William_Model node.

Set limits to the leg's joint rotation

Unless you're modeling a contortionist from the Cirque du Soleil, legs don't usually bend in the unnatural ways that IK chains allow them to. For more normal movement, you need to set limits on the legs' rotation.

62. In the Right view, multi-branch-select the control objects for both feet.

63. Straighten the legs (translate on Y) to the furthest extension that you want the leg to reach without making them snap completely.

64. Select the shin bones and choose **Create** > **Skeleton** > **Set Minimum Rotation Limit**. You'll see a little icon appear around the knees.

65. Multi-select the control objects of both feet again, and bend the legs to the maximum bent position you want.

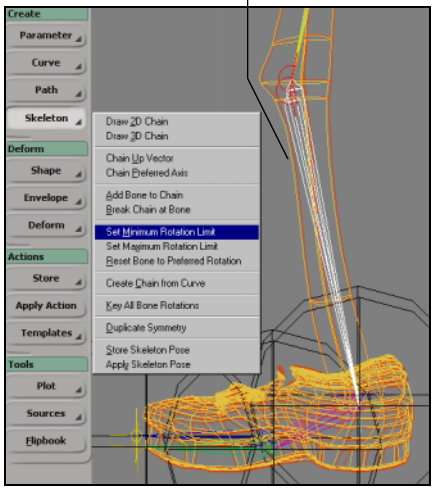
66. Select the shin bones again and choose **Create** > **Skeleton** > **Set Maximum Rotation Limit**.

Now when you move the legs, you see that you can't rotate them past these limits.

67. Set rotations limits for the foot and toe bones as well.

Extend and bend both legs at the same time to find the min/max knee angle.

Then select both shin bones to set their min/max rotation limits.

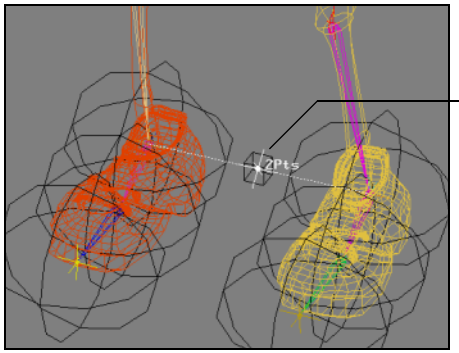


Hips relation

Make the hips follow a position between the feet: instead of animating the feet and the hip area separately, you can change the reference position of the feet and the hips will follow.

68. Get a null and position it between the feet. Constrain it to the foot control spheres using a 2-point constraint (**Constrain > 2 Points**).

69. Get an implicit cube and position it on the null. Make it child of the null. This cube is used as a "spacer" between the position of the constrained and the constraining objects.



Null constrained between feet rigs. Spacer cube makes null control easier to see.

To position the spacer precisely and quickly, choose **Constrain > Position**, pick the null, then remove the constraint by choosing **Constrain > Remove Constraint**.

70. Select the cube in the explorer, right-click on it, and choose **Rename**—call it `spacer`.

You will constrain the position of the COG object to the spacer's position, but first you'll use the compensation mode. Compensation lets you interactively offset a constrained object from another and animate the objects independently while maintaining the constraint.

71. Click the **Comp** (compensation) button on the Constrain panel to automatically create a compensation offset in the position constraint to prevent the COG object from jumping over the position constraint.

72. Now constrain the position of the COG object to the spacer. Select the COG sphere, choose **Constrain** > **Position**, and pick the spacer cube.

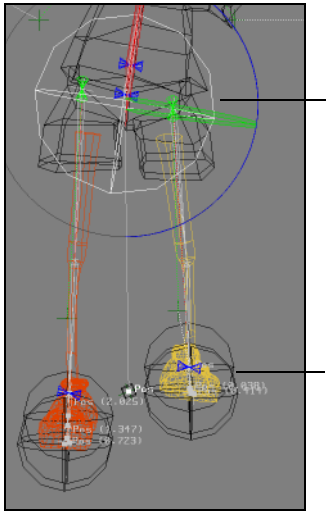
73. Deselect the **Comp** button and see how the COG stays in place, keeping a position constraint with an offset.

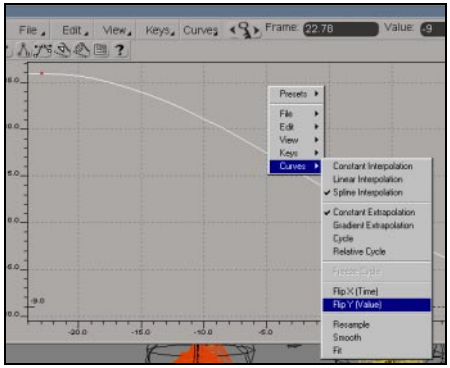
Make sure you do not leave the **Comp** button on!

74. Drag the spacer cube under the William_Model node to see the effect. Undo the translation to position it back on top of the null. You can locally animate the spacer to make the character sit down, for example. Note that if you animated the spacer, an offset from the reference null is created and will influence the reaction of the COG.

75. Branch-select one of the foot rigs and translate it to see the result of the COG constraint: as you move the rig, the COG is automatically adjusted to stay between the two feet.

Now William can easily pose—try raising one of his feet to see how he reacts.





Making the COG Reference the Spacer Cube's Rotation on X

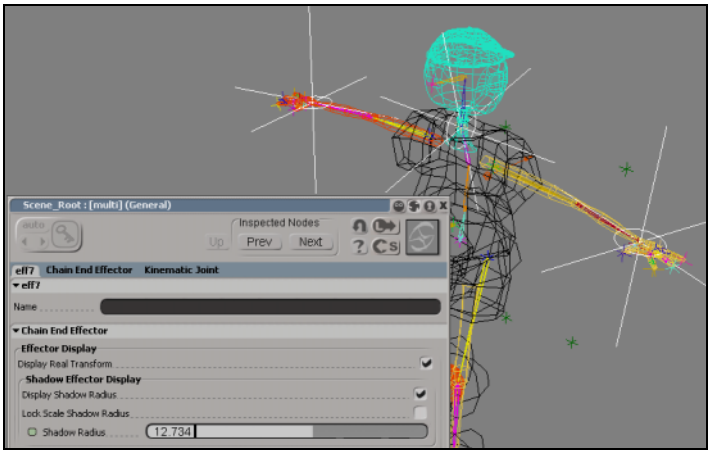
In the next step, you are going to use linked parameters to link the Z rotation of the COG to the spacer cube's global Z rotation.

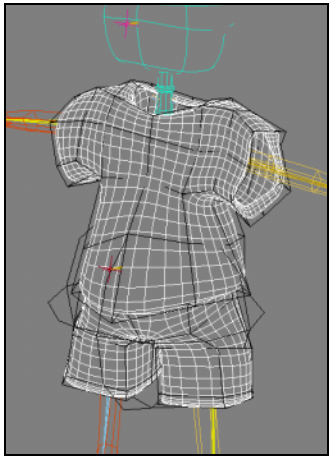
76. Drag one foot up and rotate the COG (branch-selected) in Z in the same direction as the raised foot.
77. Select the COG object and press Ctrl+k to open the Local Transform property editor. Lock the editor.
78. Right-click the animation icon (green box) beside the **Rotation Z** parameter and choose **Link With**.
79. Expand the **null** * > **spacer** > **Kinematics** > **Global Transform** > **Ori** > **Euler** > **Z** node (this is the null that has a 2-point position constraint between the feet).
80. Move that foot back down, move the other foot up, and rotate the COG (branch-selected) in Z in the same direction as the raised foot.
81. Back in the COG Local Transform property editor, right-click the Rotation Z animation icon again and choose **Set Relative Value**.
82. Translate the feet in Y to see how the rotation now reacts to the Y position of the feet.

    Notice that this is the wrong way—try raising a foot and balancing the body toward that foot without falling! To correct it, you will have to inverse the link's function curve.
83. Back in the Local Transform page of the COG, right-click the Rotation Z animation icon again and choose **Animation Editor**.
84. Select the function curve in the animation editor. Right-click in the graph area and choose **Curves** > **Flip Y Value**. Now the relation between the rotation curves is inverted.

Making the Hands and Spine Easier to Select

85. Select both hand's effectors (use the Effector filter in the Selection text box first), and press Enter to open their property editors.
86. Select **Display Shadow Radius** and increase the **Shadow Radius** value to a size that makes the effects easy for you to select.
87. Repeat these steps for the spine effector so that you can easily animate the upper body.





Making a High-Resolution Model from a Low-Resolution One

Not enough resolution in the envelope? Using mesh subdivisions, you can create a high-resolution model while keeping the weighting on the low-resolution model.

88. Select the envelope (T-shirt and shorts model) and choose **Create** > **Poly. Mesh** > **Subdivision** from the Model toolbar.
89. Select Doo-Sabin as the **Subdivision Rule** and enter a **Subdivision Level** of 2.

    Select one of the hand effectors and drag it around to see the high-resolution geometry update.

Make this high-resolution geometry only visible at render time; this will keep a high interaction rate in OGL display mode for animating.

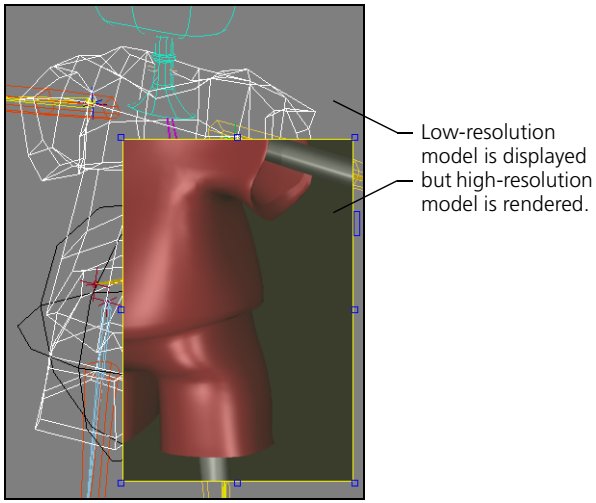
90. Select the high-resolution model (it's called **polymsh** in the explorer).

91. Click the **Selection** button in the main command area, and click the **Visibility** icon in the explorer that is displayed.

92. In the Visibility property editor, deselect **View Visibility**, and leave the property editor open.

93. Select the original envelope model (low-resolution), and the property editor updates with the current selection. Deselect **Render Visibility** for the original model.

Now only the high-resolution model is renderable, but only the low-resolution model is visible in the viewport. Draw a render region (press **q**) to see what it looks like rendered.



Conclusion

You have created a basic skeleton hierarchy and rig that can be used with other biped models. Depending on how your character is animated, you can adjust the rig or add more control objects for specific controls. If you're feeling adventurous, make a walk cycle for William!

For more information, see the following chapters in the *Animating* guide:

- *Chapter 11: Skeletons & Kinematics*
- *Chapter 12: Envelopes.*

Section 7 Advanced Animation

Tutorial 10: The Animation Mixer—Jaiqua Poses

The animation mixer works with the same logic that you would find in a non-linear video editing system. It's a tool that gives you high-level control over animation by letting you position, cycle, scale, warp, bounce, and mix and weight actions.

The animation mixer uses actions, which are different types of animation segments stored in a "package." You define (store) an action once and apply as many times as you like. You can create an entire library of actions, like walk cycles or poses, and copy them from one model to another.

After you create an action, you can load it as a clip onto a track in the animation mixer. In the mixer, you can sequence the actions, mix actions together, or create compound actions that contain other actions.

Part I of this tutorial shows you how to:

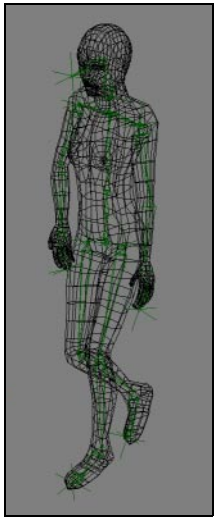
- Create action sources.
- Instantiate action clips.
- Cycle and crop action clips.
- Modify the underlying function curves of the original action source.

Part II of this tutorial shows you how to:

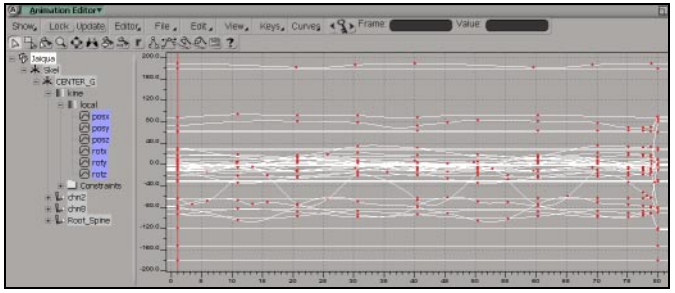
- Import and clean motion-capture function curves.
- Create actions for various kinds of animation.
- Mix between animated actions and static poses.
- Create markers on animation tracks.

Overview - Part I

1 **Start with motion-capture animation.**

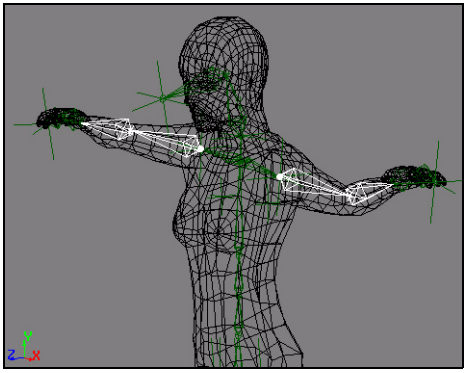


2 **Clean up the function curves.**



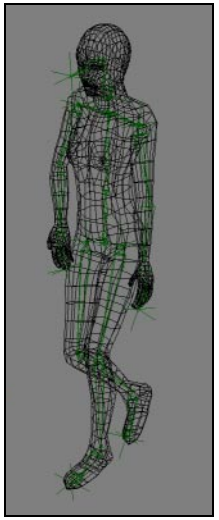
3 **Create an action.**

4 **Store additional poses.**

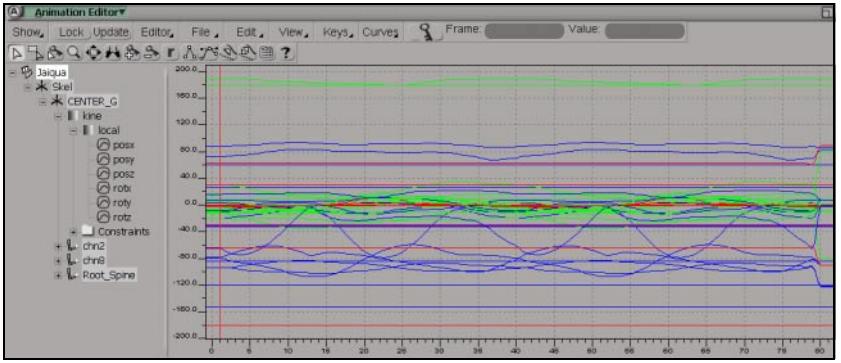


5 **Mix actions by setting weights.**

Processing Motion Capture Animation



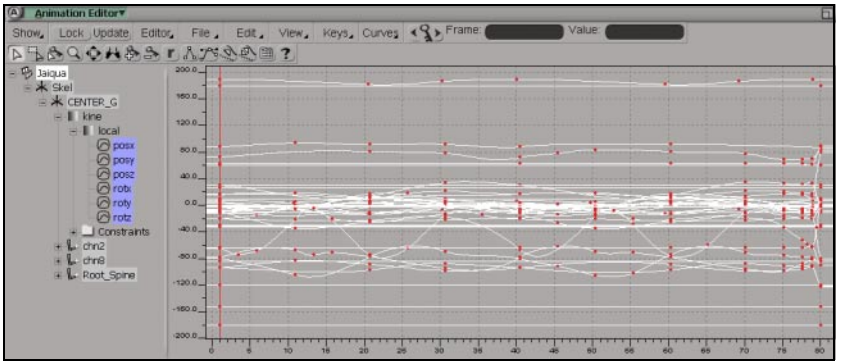
1. Open the **AN2_e1_jaiqua** scene from the Tutorials project.
2. Play back the animation. Notice that a full walk cycle is 39 frames long.
3. Select the whole model hierarchy by right-clicking on it.
4. Choose **Animation** > **Animation Editor** from the Animation panel (bottom of the main window) or change a viewport to **Animation Editor**.



5. In the View menu of the animation editor, make sure that **Animated Parameters** is on.
6. Choose **Edit** > **Select All Curves** in the animation editor.

7. Click the Preferences icon on the command bar.
8. In the Animation Editor Preferences property editor, click the **Curve Processing** tab. This page gives you three options for processing curves:
   - You can **smooth** curves. This works in a similar way to blur on pixels. Smoothing tries to decrease the noise often seen on mocap (motion capture) files.
   - You can **resample** the curve to add keys at regular time steps while retaining (or not) the existing keys.
   - You can **fit** a curve onto the raw values. This reduces the key density of a curve while keeping the same overall curve shape.
9. In this example, you fit a curve to raw values. Enter `3` as the **Fitting Tolerance** value.

10. Click the **Fit** button and wait for the process to finish.



11. Close the Preferences property editor and the animation editor.

Storing Transformations]

You can think of actions as encapsulated animations. You can create actions for the transformations or the marked parameters. For either of these options, you can store three types of action:

- The current "static" values (**Current**)
- Those parameters that are animated by keys (**fcurves**)
- Those parameters that are animated in any way, including function curves, expressions, constraints, and so on (**All Sources**)

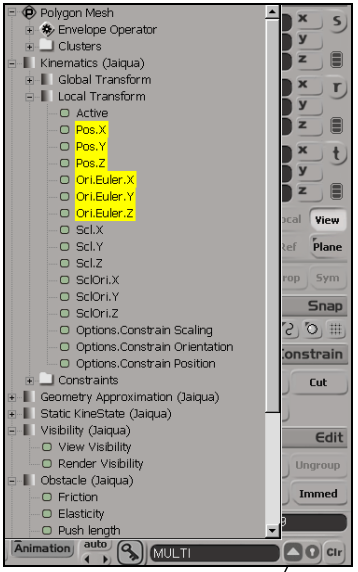
Mark parameters

Creating actions based on marked parameters can increase performance because the actions don't contain unnecessary parameters. The list of parameters included in the action is also easier to manage when working with templates.

- You should always mark local transformations when storing actions. Actions that contain local transformations are easier to work with. Remember that *Global always overrides Local!*
- Before you store an action, always check that the marked parameter box shows **foo.local.bar** and *not* **foo.global.bar**.

You can mark parameters quickly by selecting an SRT tool from the Transform panel. For example, if you click the **S** button to select the scale tool, **kine.local.scl** is automatically marked and appears in the marked parameter list in the Animation panel (lower-right of the main window).

If you click a specific axis on the Transform panel and want *only* that axis marked, you must be in Parent mode for scaling and translation, or Add mode in rotation. In other modes, all three axes will be marked even if only one axis is selected.



Opens the marked parameter list.

You can also use the marked parameter list in the Animation panel. Click the triangle to display a list of animatable parameters of the current selection, then click on any parameter to mark it—it turns yellow to show it's marked. To add or subtract marked parameters, use the Shift+ or Ctrl+key modifiers. You can also use the Lock and Clr (Clear) icons just beside the triangle button.

12. In an explorer, expand the Jaiqua node, then the Skel node, and then middle-click CENTER_G to branch-select it.

13. In the explorer command bar, select the All + Animatable Parameters filter. Then expand the CENTER_G > Kinematics > Local Transform node. Shift+click the **Pos** > **X**, **Y**, and **X** nodes and the **Ori** > **Euler** > **X**, **Y**, and **Z** nodes to mark them.

    The marked parameter list displays "MULTI" because more than one parameter is marked.

Store an animation in an action source

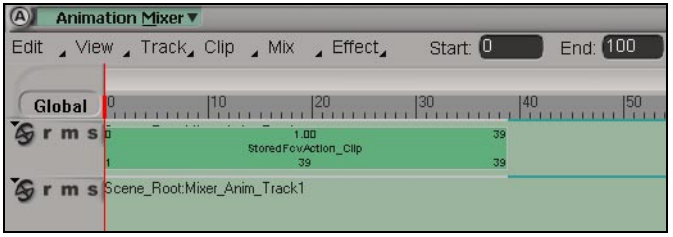
14. Choose **Actions** > **Store** > **Marked Parameters - Fcurves** from the Animate toolbar. In the Stored Action dialog box, set **Default Out** to 39 (the length of the walk cycle), then click OK.

You are leaving **Remove Animation** on because you want to deal only with actions in the animation mixer: this option removes all the function curves from the character, storing all the animation only in the action source, making it easy to continue animating the character and storing different actions.

The animation is disconnected from the model and becomes a source in the action library under the model (Mixer\Sources\Animation).

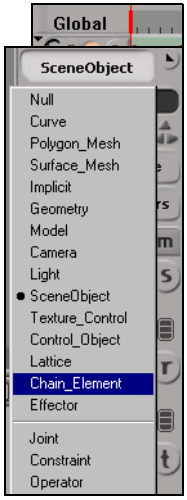
Creating Action Clips in the Animation Mixer

15. Open the animation mixer (in any viewport or from the Views menu) and click **Update**.
16. Position the mouse pointer near the start of the first track, then right-click and choose **Load Source**.
17. Pick the only source available from the menu (**StoredFcvAction**). The action clip is placed on the track and is represented by a green rectangle.

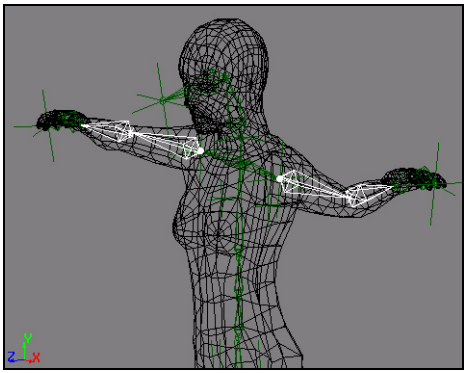


You can change the timespan shown in the animation mixer independently from the first and last frame of the animation. Simply enter new values in the Start and End boxes on the animation mixer command bar. However, remember that frames outside of the start and end frames of the scene will not play back.

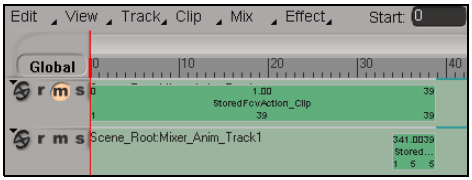
Storing Poses



18. Now you will store some static poses. First click the **m** button on the left of the animation track to temporarily mute the track. This prevents the action from updating the skeleton's transformations, such as if you accidentally jog the timeline.
19. Choose **Chain_Element** from the Filter menu in the Selection panel on the main command area. (Click on the small arrow beside the text box.)
20. To create some static positions, first select one of Jaiqua's upper arms and rotate it locally in Y into a neutral "T"pose. Repeat for the other upper arm.
21. Ctrl+click to select all four arm bones as shown. Notice how easy it is to select only the bones with the **Chain_Element** selection filter on.



22. With these four bones selected, press the **c** key to mark the rotations.
23. Save the current rotation values by choosing **Actions** > **Store** > **Marked Parameters - Current**. Click OK to accept the default values.

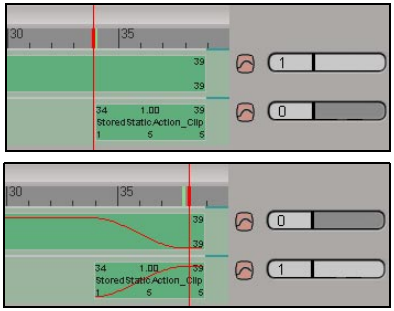


24. Now load this new pose in the animation mixer: right-click in the second track and choose **Load Source** > **StoredStaticAction**. Drag the new 5-frame clip so that it ends at the same time as the walk cycle.
25. Click the **m** icon again to "unmute" the first track, then play back the animation.

    Notice that the arms jump halfway to the neutral pose at the beginning of the second clip. You'll fix that by keyframing the mix weights to blend the two clips.

Mixing Clip Weights

26. Activate **View** > **Weight Mixer Panel** (if necessary) to show the weight sliders at the right end of the tracks.



27. Animate the weighting so that the first track fades over the second track as shown in the illustrations:

    - Move the playback cursor to the beginning of the second clip. Click the animation icon (green box) for the first (top) track to set a keyframe on the first action's mix weight at its current value of 1. Set the second action's mix weight to 0 and click its animation icon.
    - Move the playback cursor to the last frame of the clip. Set the first action's mix weight to 0 and save a key. Set the second action's weight to 1 and save a key.

      If you don't see the weight curves in the mixer, make sure that **View** > **Weight Curves** is on.

Creating Markers

You can add some markers to help organize the layout or to set in/out points for loop playback:

28. On the empty area of the second track, press Ctrl+drag to select an area on a track that you want to loop (make sure to not select a clip; otherwise, the markers will be on the clip, not the track). The selected area turns gray.
29. Right-click on the gray region and choose **Add Marker**. A rust-colored box appears around the area you selected.
30. Right-click again on the marker (make sure you don't click on the track) and choose **Properties**. Rename the marker `loop`.
31. Right-click on the marker and choose **Set In-Out Loop**. Yellow markers appear on the mixer's timeline showing the delimited area.

    Play back the scene and watch the marked frames play in a loop. To play the whole scene again, click the Loop button in the Playback controls below the timeline.

Conclusion

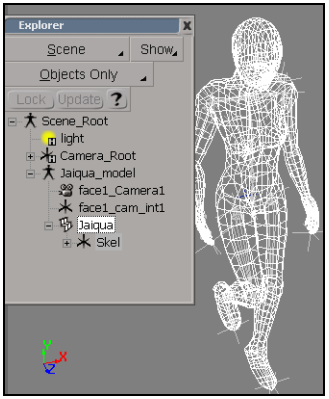
You have learned what actions are and how to use them in the animation mixer, as well as how to change and key the weights of overlapping clips. Part II shows you how to create a cycle and add an offset to the existing animation.

For more information, see the following chapters in the *Animating* guide:

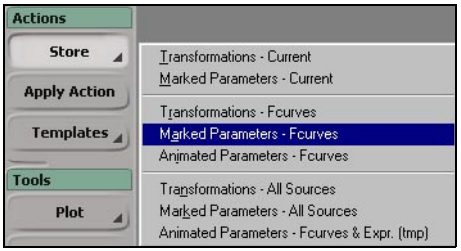
- *Chapter 3: Editing Function Curves in the Animation Editor*
- *Chapter 13: The Animation Mixer*
- *Chapter 14: Actions*

Overview - Part II

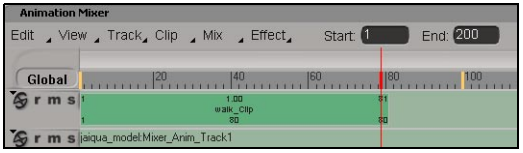
1 **Start with an animated scene.**



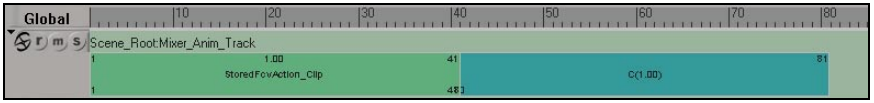
2 **Store an action source.**



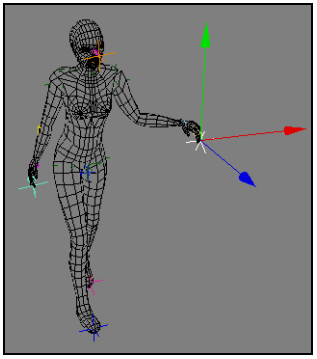
3 **Create an action clip from the action source.**



4 **Create and adjust a cycle.**



5 **Create an offset “on top of” the motion-capture source function curves.**

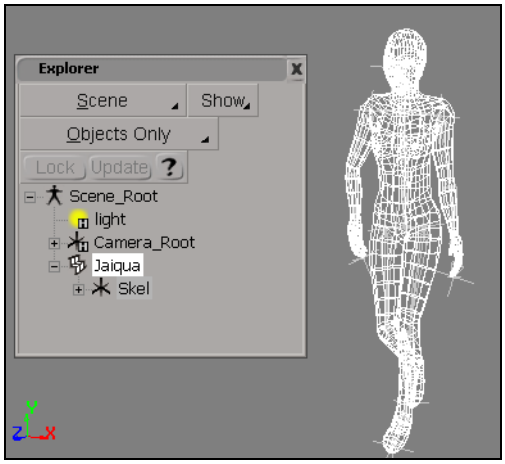


Storing Actions

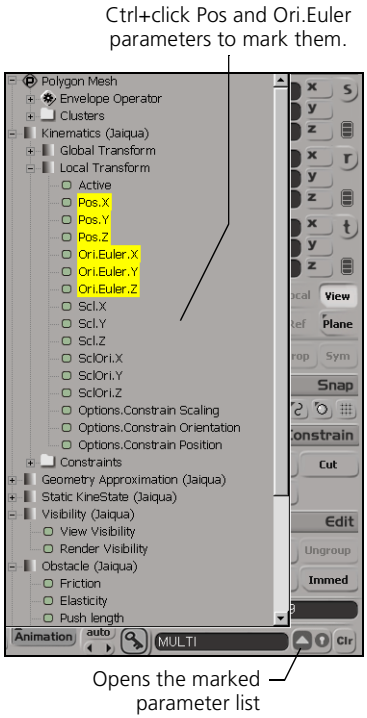
Once you have any kind of animation, you can store this animation in an action source. When storing actions, you can choose to store all transformations or just the marked parameters, thus creating a smaller action file. In this example, you will mark the position and orientation parameters before storing them in an action.

1. Open the **AN2_e1_Jaiqua** scene. This is the same scene as was used in Part I of this tutorial. If you saved this scene after completing the previous lesson, load the original scene from the Backup folder.

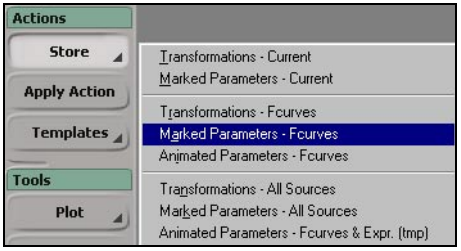
2. Right-click on part of the skeleton to select Jaiqua's entire skeleton hierarchy and envelope.



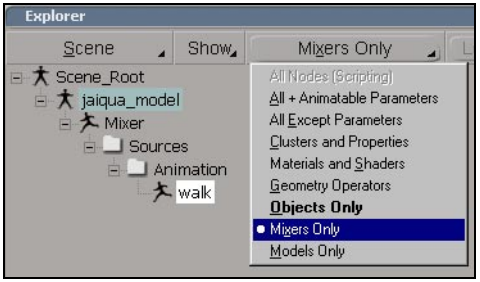
3. Open the marked parameter list in the Animation panel by clicking the triangle button on the bottom right. Expand the Kinematics > Local Transform nodes, then Ctrl+click to mark all six **Pos** and **Ori.Euler** (position and Euler rotation) parameters.



4. From the Animate toolbar, choose **Actions** > **Store** > **Marked Parameters - Fcurves**. Enter `walk` as the action name and click OK. The animation is now stored in an action source and is no longer linked to the object.



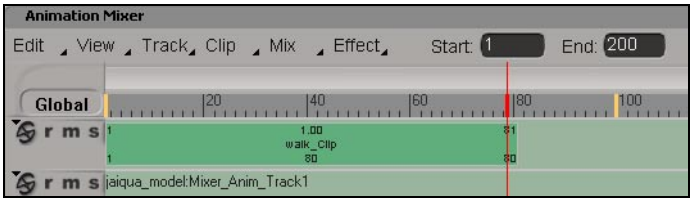
If you ever need to restore the original animation after creating an action, select the action source and choose **Actions** > **Apply Action**. To access the action source, select the model geometry and choose the **Mixers Only** filter in the explorer. Expand the Mixer node as shown.



Once the animation is back on the character, the action source is still available, even if you choose **Animation** > **Remove All Animation**. The only way to delete the action source is to select it in the explorer and delete it manually.

Creating Action Clips

5. Open the animation mixer in a viewport or floating window.
6. With Jaiqua still selected, click **Update** on the animation mixer's command bar.
7. Right-click on a track and choose **Load Source** > **walk** (or StoredFcvAction if you didn't change the default name). The action is loaded as a clip on the track.

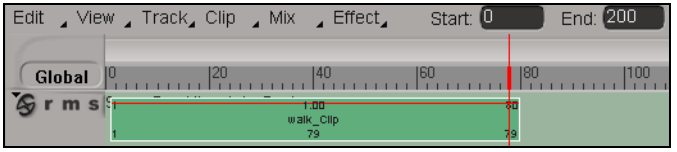


8. Play back the resulting animation.

Cropping the Action Clip

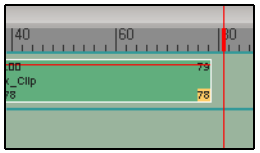
When playing the animation, you can see that Jaiqua snaps back to her default "T" position on the last frame of the clip. You will trim the extra frame.

9. Drag the playback cursor to frame 79 of the clip.
10. Select the clip and choose **Clip** > **Trim After** from the mixer's command bar, or press the] key. This trims the frame after the position of the cursor.
11. Drag the action clip so that it starts at frame 1 and change the animation mixer length by setting its End frame to 200.



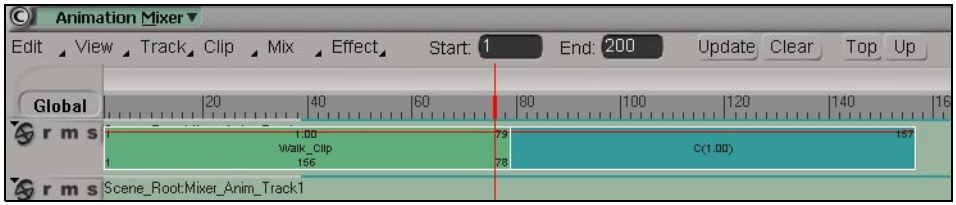
12. Change the main timeline End frame to 200 as well.

Adding a Cycle to the Action Clip



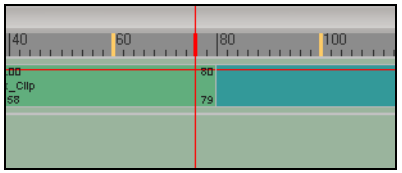
13. Click on the lower-right corner of the action clip so that a yellow square appears. Drag the clip so that it ends just before frame 160.

    The animation did not cycle all the way to frame 160 because dragging creates only cycles that are integral multiples of complete actions. You can create fractions of cycles by setting it in the Time Control property editor of the action clip, which you will display in the next step.



Looping the Animation

When working in the animation mixer, you may find it useful to loop certain frames of animations.



14. When you click the Loop button in the playback controls below the timeline, the start and end frames in the mixer are defined by the timeline's length.

15. In the mixer's timeline, click and drag the yellow markers to adjust the looping area between frames 60 and 100 so that you see the transition between the two cycles. Make sure that you see "L" and "R" by the cursor when you drag the yellow markers.

Viewing the Original Function Curves

Is the rotation of her hips is a bit too much? Is her head looking down too much? Or do the feet need modifications? You can change any of these movements because the original function curves are still available.

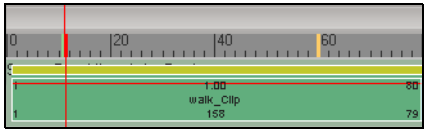
16. Right-click on the clip and choose **Source** to edit the action's source (not just the clip).

17. In the Source property editor, click the **Edit Source Data** button to open the animation editor.

18. Select any of the seemingly thousands of curves to see that they are all raw function curves (one key per frame, indicating their motion capture origins. You are going to adjust these curves in a few easy steps.

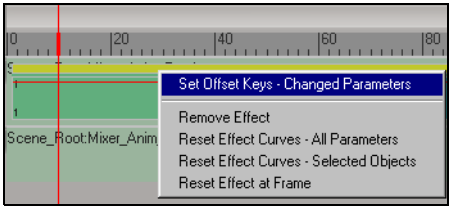
If these curves would have been simpler splines, you could have edited the curves in the animation editor. Instead, you are going to use the offset effect to make adjustments without destroying the original function curves. See the next tutorial, The Commander, for tips on managing motion capture files.

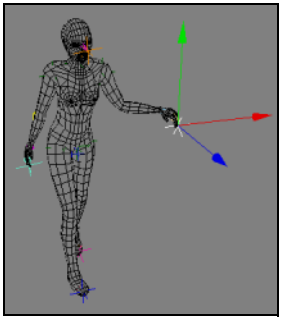
Adding an Offset Effect to the Original Animation



The offset effect allows you add any animation "on top" of the action clip without altering the original animation. You are going to create an offset effect for the hand position's adjustment

19. Select the walk clip and choose **Effect > Create Offset Effect** from the mixer's command bar. This creates the clip effect, which is displayed as a yellow bar above the clip.
20. Select one of Jaiqua' hand effectors.
21. Go to frame 10, right-click on the yellow bar and choose **Set Offset Keys - Changed Parameters**. This sets a keyframe for the position of the effector.





22. Without changing the effector's transformation, go to frame 30, right-click on the yellow bar and choose **Set Offset Keys - Changed Parameters** again to set another keyframe.
23. Now go to frame 20, translate the hand up as in the illustration, and again set a keyframe for the effect using **Set Offset Keys - Changed Parameters**. The keyframe saves only the parameters that have differed from the previous keyframe.
24. Drag the playback cursor on the timeline to see the results.

Conclusion

You have learned how to cycle a clip and add an offset to the existing animation. You can also add an offset using the **Offset Map** button on the mixer's command bar or type in an offset directly in a clip's value map. To do the latter, right-click a clip and choose **Clip Properties**. On the Value Mapping Expressions page, add the offset value for the parameter you want to change.

For more information, see *Chapter 14: Actions* in the *Animating* guide.

Tutorial 11: The Animation Mixer—The Commander Attacks

Your producer has provided you with animations from SOFTIMAGE|3D. You must mix and modify these motions in XSI.

The animation mixer can simplify part of the process when dealing with complex animation. Motion capture is a good example because of its multitude of function curves. You can use the raw motion capture data, and modify it non-destructively, without having to deal with all the function curves in the animation editor.

Take advantage of the non-linear animation capabilities of the animation mixer to quickly make the changes without touching a single function curve.

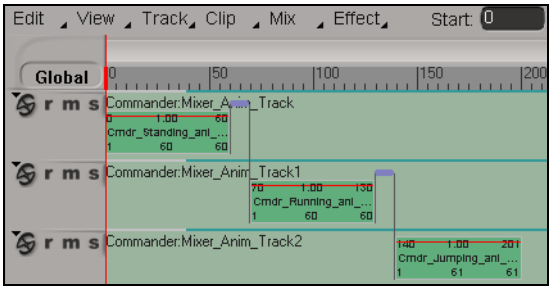
This tutorial shows you how:

- Use the animation mixer as your main animation tool
- Mix, sequence, and equalize actions
- Create transitions between actions.

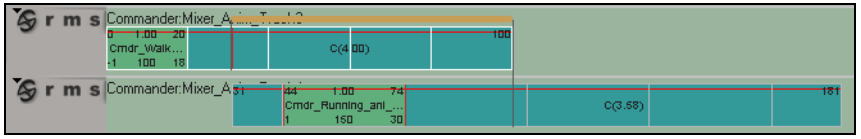
Overview

1 **Load predefined action clips.**

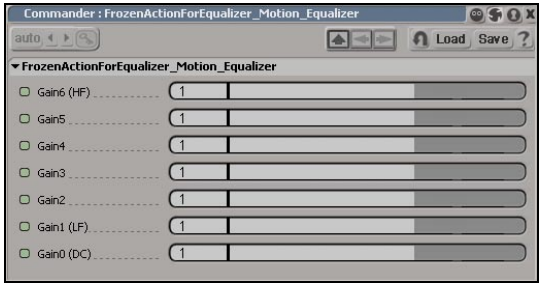
2 **Create transitions between clips.**



3 **Use a bridge transition to change from a walk to a run.**

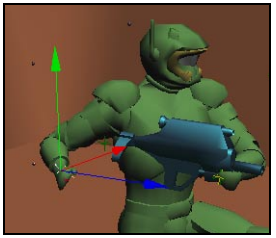


4 **Break down animation into frequencies (equalize).**

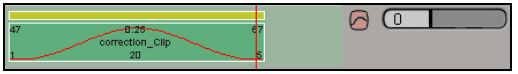


5 **Load a sniper action and add a wave to it.**

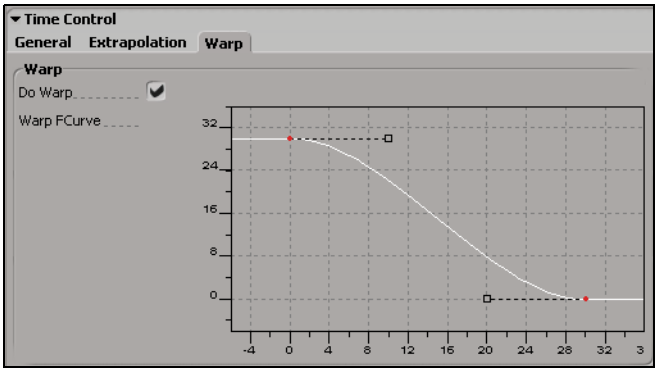
6 **Correct the arm's movement with offsets.**



7 **Keyframe the weight of a clip.**

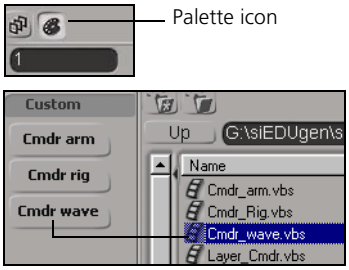


8 **Reverse an action with a timewarp.**



Loading Scripted Buttons to Automate Selection

Before you begin this lesson, you can load a group of scripted buttons that have already been created. These buttons will speed up your work by relieving you of the need to search through the explorer, select objects, or activate layers.

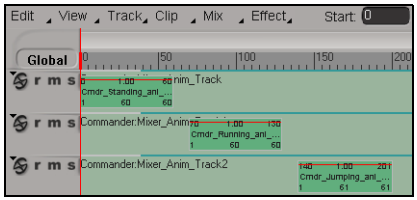


1. At the bottom of the toolbar, click the Palette icon to open the Palette/Custom toolbar.
2. Open a browser and open the **Tutorial_Project\Scripts** folder.
3. Select each script that is prefixed with "Cmdr" and drag them, one by one, to the Custom part of the toolbar. When you drag the scripts over, you can give them more descriptive names.
   - **Cmder rig** selects the hierarchy of nulls that controls the Commander character.
   - **Cmdr arm** selects the null that controls the Commander's right arm movement.
   - **Cmdr wave** displays/hides the reference spheres that indicate the reference position for the keyframes to be set for the wave movement.
4. Save this scene.

Loading the Action Clips

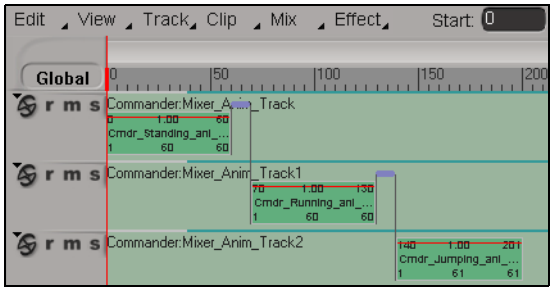
5. Open the **AN2_e2_commander** scene. This scene contains a helmeted, jumpsuited, beweaponed military tough-guy called "the Commander."

   Drag the playback cursor and you'll see that there is no animation applied to him.
6. Click the **Cmdr rig** button on the toolbar to select the commander rig. This rig will drive the character's animation.
7. Switch back to the Animate toolbar (click the toolbar icon at the bottom of the toolbar).
8. Click the **Update** button in the animation mixer to display tracks for the Commander.
9. Right-click on a track and choose **Add** > **Track** > **Animation** or press Shift+a.
10. Load the following clips from the Animation folder onto three separate mixer tracks, leaving 10 empty frames between each clip (you can easily drag them into exact position later). To load the clips, right-click each track, choose **Load Source > From File**, and select the clip:
    - **Cmdr_Standing.ani** is an animation of the commander standing still with a chest motion to simulate breathing.
    - **Cmdr_Running.ani** contains a run cycle.
    - **Cmdr_Jumping.ani** contains a jump action.



Creating Transitions

11. In the mixer, choose **Mix** > **Standard Transition Tool**. Click the first action clip, and then the second. Notice the purple transition bar between the two.
12. Click the third clip to create another transition. Right-click to finish creating transitions.



Move the playback cursor over the transitions to see the results.

You can see that the transition between standing and running is not natural-looking. The Commander appears to be moonwalking, and the second transition doesn't look so great either. You need to trim the clips to make better transitions.

13. Right-click the run clip and choose **Time Properties**. In the **Source Clipping In** text box, specify 12 frames (this is a suggestion—try a different one if the results are better).

    Move the playback cursor over the transitions to see the results.

Now trim the run clip to blend with the jump clip.

14. Drag the playback cursor until the commander in his running position has one foot beneath his hips (i.e., match the foot position of the first frame of the jump clip).
15. Select the clip and choose **Clip** > **Trim After** in the mixer's command bar to trim the clip after the position of the playback cursor.
16. Reposition the jump clip so it starts approximately 10 frames after the run clip ends.

    Again, move the playback cursor over the transition to see the results.

You can right-click on the transition bar, choose **Properties**, and select another transition type. For example, you could choose the Cardinal type to use a Cardinal transition curve as opposed to the linear interpolation of the Standard transition.

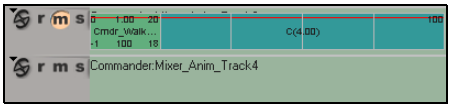
Using the Bridge Transition

The bridge transition processes the corresponding animation parameters between two actions and creates a function curve for each parameter (for example, one curve for scale x, y, and z). This results in a smoother transition that you can control more.

Here are a few tips you should consider before using the bridge transition:

- Both clips need to overlap in time.
- The “From” clip has to start before the “To” clip.
- There should be only 1 cycle (or period) in the “cyclable” imported action.
- Synchronize the movements between the actions for the best results (for example, the right foot touches the ground at approximately the same frame for both clips).

17. Mute the current tracks and add two more action tracks.
18. Solo viewport A (click its letter). Then, in viewport A, hide 3D geometry and nulls (click the eye icon and deselect these items from the menu). Only the commander’s skeleton should be visible.
19. On the first available track, load the **Cmdr_Walk** action from the Animation folder.

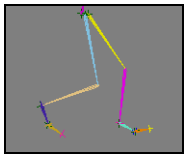


20. Cycle it four times and mute its track.
21. Reload **Cmdr_Running** on the next track.

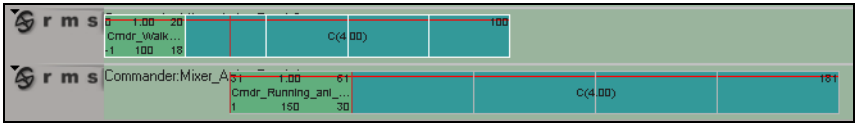
Since there are cycles (or periods) within that action, you must trim it.

22. Move the playback cursor to the beginning of the clip, and select the clip and the four leg bones of the commander (use the Shift key).
23. From the mixer’s command bar, choose **Clip** > **Find Cycles** and accept the default values. Click OK for the message that appears.
24. Deselect the leg bones.

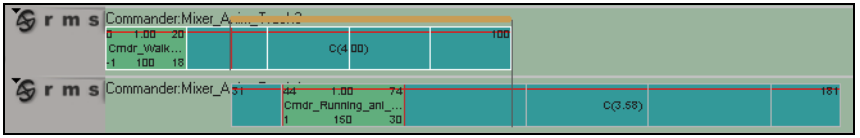
25. Move the playback cursor to the frame where the marker is and press the] (end bracket) key to trim the clip after the playback cursor.
26. Right-click on the newly trimmed clip, choose **Set in Out Loop**, and play the animation to verify that the clip is still cycling. Click on the loop icon in the playback controls when you’re done to remove the loop.
27. Select the run clip and create four cycles for it.
28. Unmute the track for the Walk clip.



29. For each clip, select it and add a marker (press the **m** key) to indicate where the right foot is leaving the ground but is still touching it, as shown in the image on the left. Drag the run clip to the left to match its marker to the walk clip's.



30. Choose **Mix** > **Bridge Transition Tool** and pick the walk clip, and then the run clip. Right-click to end. The bridge processes the clips and creates a transition between them shown in light orange.



Separating a Clip into Animation Frequencies

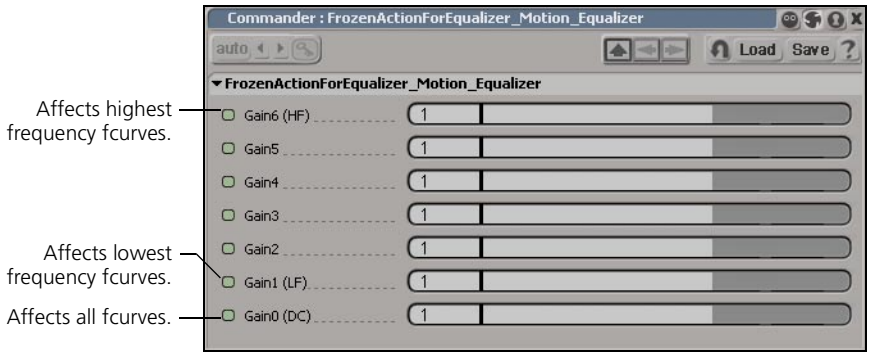
By decomposing animation into its base frequencies, you can control each frequency gain (weight) individually, in a way similar to using an audio equalizer to control individual audio waveforms. For example, you may want to filter out or isolate high frequencies in an animation sequence.

31. Select the walk and run clips and choose **Clip** > **Equalize** from the mixer's command bar.

32. This creates a compound clip on a new track and mutes the contributing clips. The Equalizer dialog box is displayed in which you can use the **Gain** sliders for adjusting the weight of each animation frequency.

33. Create a loop on the timeline and play with the equalizer's sliders. For the most dramatic and "interesting" results, drag the Gain 0 slider, which affects all function curves.

You can also animate any of these frequencies—note the animation icons (green boxes) beside each slider.



Double-click the equalized compound clip to see a "subclip" for each frequency that is created. Setting the weight sliders at the end of each subclip's track is the same as setting the Gain sliders in the Equalizer dialog box.

Add a Sniper Action

34. Open the **AN2_e2_sniper** scene. In this scene, the Commander has keyframe animation already applied to him.

35. Click the palette icon to open the custom toolbar and click the **Cmdr_Rig** button.

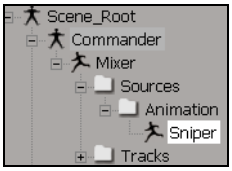
36. Back on the Animate toolbar, choose **Actions** > **Store** > **Transformations - Fcurves**. This stores an action source for the local transformation properties that are animated with function curves (keyframes).

37. Name the new action `Sniper` and leave the rest of the settings as they are and click OK.

You are leaving **Remove Animation** on because you want to deal only with actions in the animation mixer: this option removes all the function curves from the rig, storing all the animation in the action.

Move the playback cursor across the timeline—you'll see that there is no animation applied to the Commander because you selected **Remove Animation**. All the animation is in the action source you just created.

38. Open a floating explorer (press the 8 key) and make sure that **Mixers Only** is selected in its command bar.



39. Under the Commander, expand the **Mixer** node and look under **Sources** > **Animation** to find the **sniper** action source.

If you need to reapply the function curves to the Commander rig, select the **sniper** action source from the explorer, and then choose **Actions** > **Apply Action**. The function curves are assigned back to the Commander rig. To remove the animation, select the rig, and choose **Animation** > **Remove All Animation** in the Animation panel.

40. With the rig selected, click the **Update** button in the animation mixer to display tracks for the Commander.
41. Right-click on a track and choose **Load Source** > **Sniper**.
42. Hold the mouse pointer over the right edge of the sniper clip so that an orange box appears. Drag to the left to squeeze the clip's length down to 20 frames.

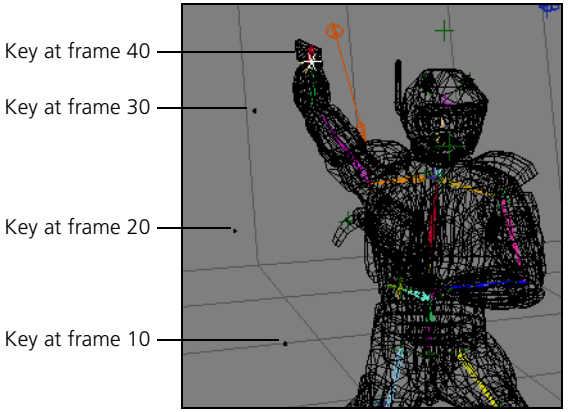
Mixing the Kneel with a Wave

Your next step is to add a new action to the Commander's motion. As he kneels down to take his shot, he will also wave, as though to encourage his fellow troops to charge on past him, right into the line of fire!

Bring the Commander to his knees:

43. Drag the playback cursor in the animation mixer to frame 20 (the last frame of the sniper clip) and mute that track.
44. Click the **Cmdr_Wave** custom button to make visible the wave reference pose layer. Four small points appear in the viewport: these are reference points that you can use to animate the waving arm (click the eye icon in the viewport and make sure that **Nulls** are selected).
45. Select the null that controls the commander's right arm movement by clicking the **Cmdr_arm** custom button.
46. Mark the translation in X, Y, and Z by clicking on the **t** button in the Transform panel.
47. Drag the playback cursor to frame 0 and save a keyframe (click the key icon or press the **k** key).
48. Expand a viewport so you can see what you are doing more clearly.

49. Move the Commander's arm to align with each of the reference points, starting from the lowest one, while advancing the playback cursor 10 frames each time. Save keyframes at frames 10, 20, 30, and 40. Don't forget that you can use **autokey** to make this easier, but also don't forget to turn it off once you are done!

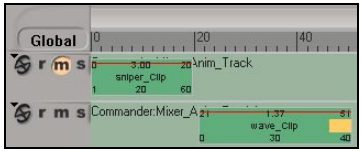


Now the only part of the rig that has function curve animation is the Commander's arm: the rest of his motion is saved as an action.

50. Make sure that **XYZ** Translation is still marked in the main command area and choose **Actions** > **Store** > **Marked Parameters - Fcurves**.

51. In the Store Action property editor, name the new stored parameter `Wave` and click OK.

52. In the animation mixer, right-click on the second track, and choose **Load Source** > **Wave**.

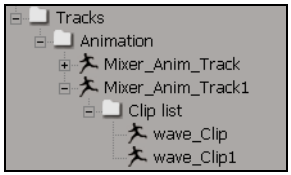


53. Scale the Wave clip to be 30 frames long and move it on the track so that it follows the Sniper clip directly.

54. Unmute the Sniper clip and drag the playback cursor to see how the Commander first kneels, then waves.

If you move the playback cursor to the end of the animation, and then drag backward very quickly, the Commander's arm may be stuck. This happens because, early in track 2, the location of the Commander's arm is not defined so it stays in its last known position. To fix this, move the playback cursor back and forth a few times from the first to last frames of the clip.

Adding an Offset to the Waving



55. Right-click on the second track near the end of the wave clip and choose **Load Source** > **Wave**. A second wave clip appears on the second track.

    Look in the explorer and note that there are now two instances of the wave clip in the **Tracks\Animation\Clip list** folder.

56. Position the second wave clip so that there is a 5-frame gap between it and the first wave clip.

57. Move the playback cursor to the beginning of the first wave clip and zoom in on the Commander's gun and hand.

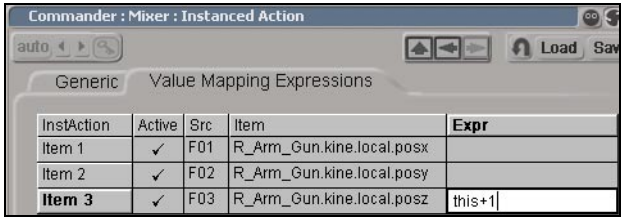
Note that his hand is not correctly positioned on the grip. You need to correct this:

58. Right-click on the first wave clip and choose **Clip Properties**.

59. Click the **Value Mapping Expressions** tab in the Instanced Action property editor.

You will add an offset value to the function curve controlling the Commander's hand. This will move the hand down a little bit without affecting the function curve itself.

60. In Item 3 (the **posZ** entry), enter `this + n` —where *n* is a numerical value (try 1). The Commander's hand is lowered by the value specified.



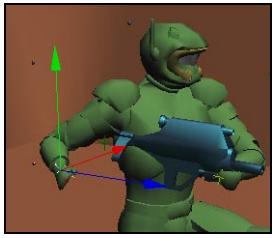
As soon as you add an expression for the action's value map, a clip effect is created. A clip effect is simply an "add-on" to the stored animation—it is represented as a yellow bar above the clip. This lets you easily add animation information to a clip without changing the original animation.

Correcting the Arm Movement

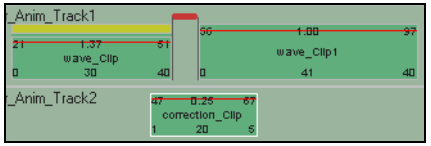
Now you need to create a transition between the two wave actions.

61. Choose **Mix** > **Cardinal Transition Tool** for a smoother transition. This creates a Cardinal interpolation for the transition curve rather than a spline interpolation, as used by the Standard transition. Pick the first wave clip, then the second wave clip, and right-click to end picking. A red transition appears between the two clips.

A small problem remains: at the transition point, the Commander's arm swings down and back toward the gun. However, this is not a natural motion: if the Commander were waving repeatedly, his hand would not return to his gun each time, so you need to add another offset to the second Wave clip to move his hand away from his gun.



62. Click the Cmdr_arm custom button to select the arm control null and move the playback cursor to the beginning of the second wave clip.
63. Click the **t** button on the Transform panel to activate Translation mode (or press the **v** key), and move the Commander's hand to his right, away from his body.
64. Choose **Actions** > **Store** > **Marked Parameters - Current** and call the new stored parameter `Correction`. Click OK.
65. Add a new action track to the animation mixer. Right-click the new track and choose **Load Source** > **Correction**.
66. Make the correction clip 20 frames long and move it on the track so that it straddles the first wave clip, the transition, and the second wave clip.



Move the playback cursor to the correction clip, and note that the new offset actually is only half the distance that you specified. Why? Because SOFTIMAGE|XSI is interpolating between the two clips and producing an averaged position. You can correct the position of the arm with another offset.

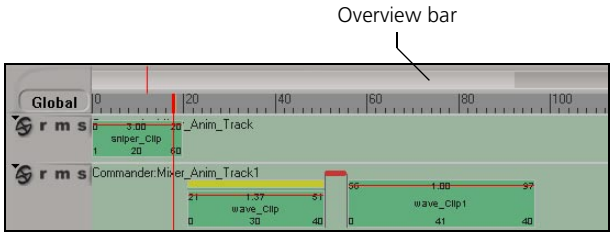
67. Right-click on the correction clip and choose **Clip Properties**.
68. In the Instanced Action property editor, click the **Value Mapping Expressions** tab.
69. In Item 1 (the **posX** entry), enter `this - 6`.

The arm now moves further away from the body—drag the playback cursor to see.

Change the range of view

The overview bar is the light gray bar above the timeline in the animation mixer. Use it to change the amount (range of frames) of the timeline you can see at once.

Use the **s** supra key to zoom in the mixer. This modify the track height. Use the **z** supra key to modify the height of tracks as well as the range.



70. Click and drag the right edge of the overview bar to the left. This zooms in on the timeline, revealing detail, and creates a kind of horizontal scroll bar. Experiment dragging the bar right and left.

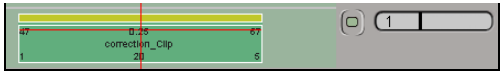
71. Change the value of the **End** frame in the timeline to 200. Note that dragging the bar now lets you scroll across 200 frames.

Changing the Weight

You want to make the second wave action look more natural by easing in and out of the correction clip. To create the ease in/out, you will vary the weight of the correction clip over time.

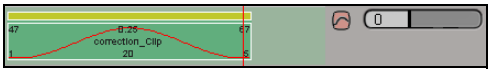
The point of greatest weight for the correction clip must be at its center, which also happens to be the first frame of the second wave clip. Move to that point with the playback cursor.

72. In the mixer, choose **View** > **Weight Curves** (if not already selected) to see the red weight curves on the clips.



73. With the playback cursor in the center of the correction clip, set the **Weight** slider at the right end of the track to 1. Click the animation icon (green box) beside the slider to set a keyframe.

74. Move to the beginning of the Correction clip, set the **Weight** slider to 0, and click the animation icon again.

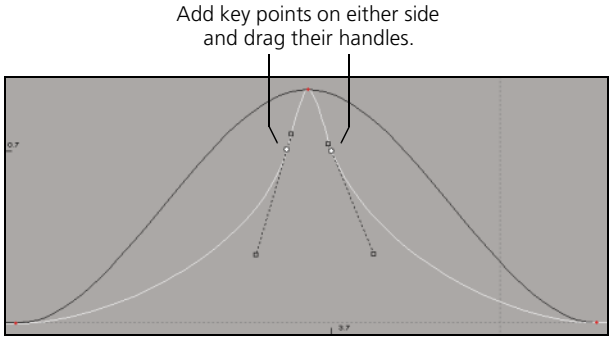


75. Move to the end of the correction clip and do the same so that there is a “bell curve” shown on the clip.

You now want to tweak the weight curve more.

76. Right-click the correction clip and choose **Animation Editor**.

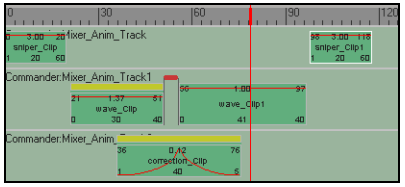
77. Add a key point on either side of the function curve and drag the key points' handles to change the shape of the curve's slope as shown here. Close the animation editor when you're done.



78. Scale the correction clip larger so that it covers the enter transition between the two wave clips. Make sure that the point of greatest weight is still at the beginning of the second Wave clip.

Reversing the Animation with a Timewarp and a Bounce

Next, you'll have the Commander stand up again after he waves his heroic troops into battle. He, of course, will follow them ... eventually!



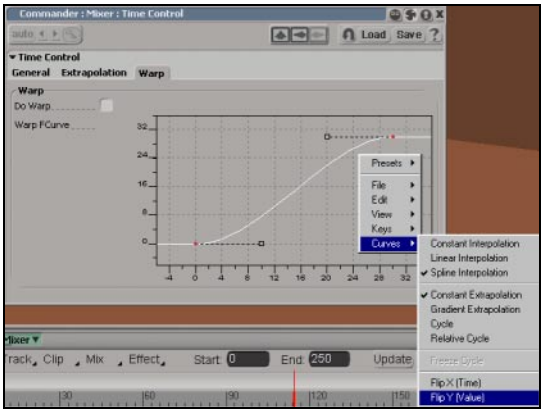
79. Ctrl+drag the sniper clip to the right along the track to duplicate the clip. Move the new sniper clip along the first track to start just after the second wave.

Now reverse the action so that the Commander starts in the kneeling position and then stands up.

80. Right-clip the second Sniper clip and choose **Time Properties**.

81. Click the **Warp** tab in the property editor. The **Warp** page shows a graph with a curve plotted on it from bottom-left to top-right.

82. Select the curve, right-click on the graph, and choose **Curves** > **Flip Y** (**Value**). This will flip the curve, thus reversing the animation.



You can add points to the curve by pressing the **a** key while clicking the curve. This would be useful, for example, during the jump action to change the speed of his jump or to prolong his time in the air.

83. To apply the warp, select the **Do Warp** option and close the property editor. Move the playback cursor to see the effect.

Bounce the arm action

One final thing to fix: You may notice that the Commander's arm is stuck somewhere above his head as he rises in the second sniper clip. His arm is starting to go numb and would like you to bring it back to his side. To do this, you'll apply a **bounce** to the wave action to reverse it.

84. Drag the upper-right corner of the second wave clip to the right. When you release the mouse button, a reverse-direction duplicate clip (bounce) appears after the wave clip.

Conclusion

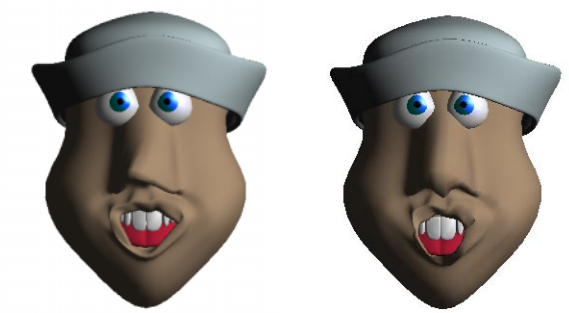
You have seen how the animation mixer is a powerful tool for animating non-destructively. None of the steps you performed in this lesson actually changed the animation sources, so you can always return to them. Now that you have created a setup for one character, set up a connection-mapping template and copy the animation to another character (such as Jaiqua) and offset their animation.

For more information, see the following chapters in the *Animating* guide:

- *Chapter 13: The Animation Mixer*
- *Chapter 14: Actions*

Tutorial 12: Shape Animation—Hi Sailor!

Use the animation mixer to explore different methods of applying shape animation to the sailor's mouth.

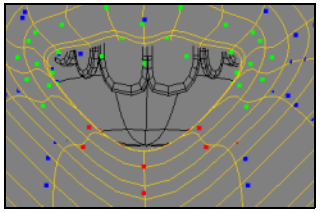


This tutorial shows you how to:

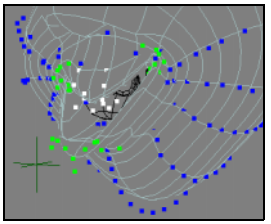
- Use standard transition mode to model and save shape keys as you go.
- Use mixed weight mode to save shape keys using predefined shapes.
- Save, store, select, and replace shape keys.
- Use clusters with a cluster center on the lower lip.
- Mix shape clips in the animation mixer.
- Control shapes with expressions driving custom parameters.

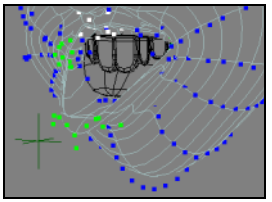
Overview

1 Create clusters on the sailor's lip.

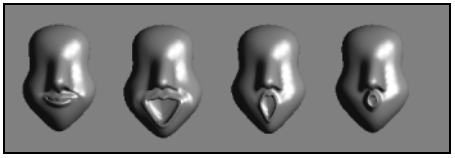


2 Save shape keys in transition mode.

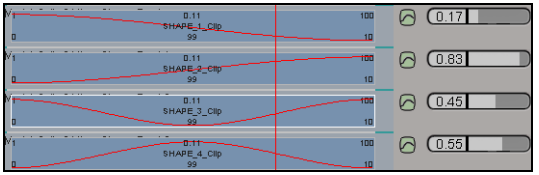




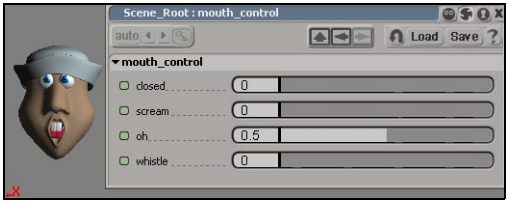
3 Select shapes in mixed weight mode.



4 Mix the weights in the animation mixer.

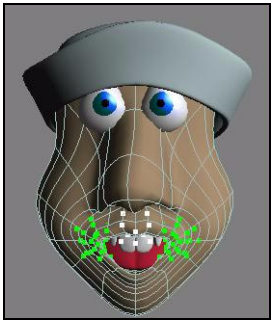


5 Create a custom parameter set to weight the mouth shapes.



In SOFTIMAGE|XSI, there are four modes in which you can create shape animation clips:

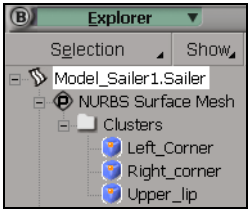
- **Standard transition mode** is usually used for "pose-to-pose" shape animation. It creates shape clips that are a single frame (transition points) and are instantiated on the same track with transition curves automatically generated between them. The transitions are linear, moving directly from one shape to the next.
- **Cardinal transition mode** is similar to transition mode: one-frame clips are instantiated along a single track with transitions. However, the interpolation in space between shapes is Cardinal instead of linear, resulting in a smooth, arcing transition that passes through each shape key. You need at least three keys for a Cardinal curve, and the curve is fully defined when there are at least four keys.
- **Mixed weight mode** creates shape clips that are the same length as your scene. They are instantiated on separate tracks with default weight curves. This is useful if you want to control shapes by mixing the weights of different shapes, often when you have a predefined set of shapes to pick from.
- **Instance only mode** creates shape clips that are 10 frames long. They are instantiated on separate tracks within a compound shape action, but with no transitions or weight curves. You can blend, scale, or create your own transitions.



SOFTIMAGE|XSI keeps track of shape animations differently from SOFTIMAGE|3D. In SOFTIMAGE|3D, each time a shape is modified, the position of every point on the shape is recorded. In XSI, when a shape is animated, only the changes (offset) of the affected points are recorded. This saves considerable amounts of memory for complex animations.

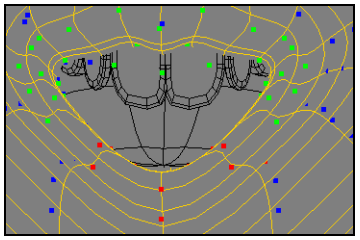
1. Open the **AN2_e3_sailor** scene from the Tutorials project.
2. Select the head of the sailor, and isolate it in the explorer by pressing **e**. Expand the **NURBS Surface Mesh** > **Clusters** node.

There are three defined clusters that you'll use to animate the mouse. Select each one in turn, and note the location of the clusters in the viewport (click the eye icon in the viewport and make sure **Clusters** is selected).



Defining a Cluster for the Lower Lip

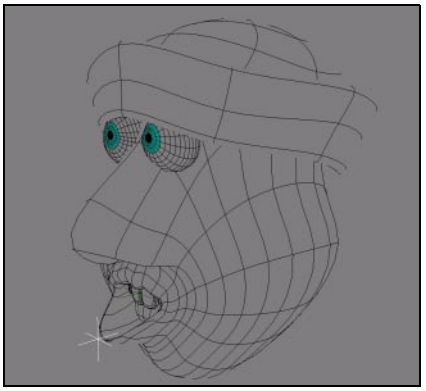
3. Click the eye icon in the viewport and select **Points**, then view the head in **Hidden Line Removal** display mode.
4. Zoom in to the mouth area.
5. Press **t** and tag the points around the lower lip. There are about 10 points in total (shown in red in the illustration).



Use the lasso selection tool (press F8) for selecting points: it uses raycasting, which prevents you from accidentally selecting unwanted points on the back of the head.

6. In the Edit panel in the main command area, click the **Cluster** button to create a cluster from the points you selected.
7. Switch back to **Object** mode and notice that the new cluster has been added to the explorer.
8. Right-click on the new cluster and rename it `Lower Lip`.

Creating a Cluster Center Handle



To make it easier to manipulate the cluster, you can create a "handle" to control the cluster.

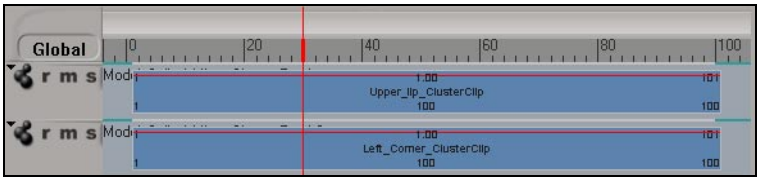
9. Get a primitive null into the scene (click the eye icon in the viewport and select **Nulls**) and move it close to the sailor's lower lip for the sake of convenience.
10. Select the lower lip cluster and choose **Deform** > **Deform** > **Cluster Center** from the Animate toolbar. Pick the null and close the property editor.

This constrains the center of the cluster to the null. As you move the null, the cluster deforms, so you can animate the shape more easily.

Using Transition Mode

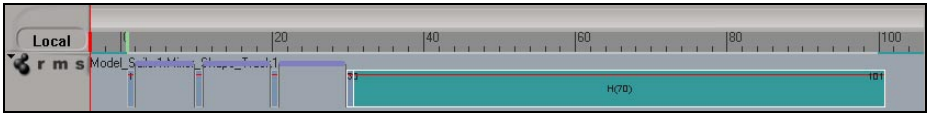
With Transition mode, you can change the shape on a model “as you go,” then save shape keys for the shapes you want to keep.

11. Select the Upper Lip cluster and choose **Deform** > **Shape** > **Standard Transition Mode** from the Animate toolbar.
12. Move the playback cursor to frame 1 and chose **Deform** > **Shape** > **Save Shape Key** to save a shape key. This creates a shape source and a corresponding shape clip in the animation mixer.
13. Move the playback cursor to frame 10, move the Upper Lip cluster down, and save another shape key (middle-click the **Shape** menu button).
14. Advance 10 frames, move the Upper Lip cluster again, and save another shape keyframe. Do this once more so that four shape keys are saved.
15. Select the **Left_Corner** cluster, and set keys for this cluster at the same frames.
16. Select the head and click the **Update** button in the animation mixer. The two tracks for the shape animations you just created are displayed.

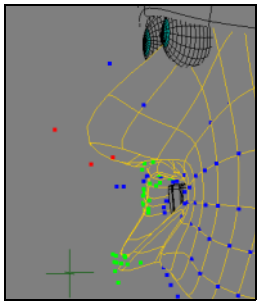


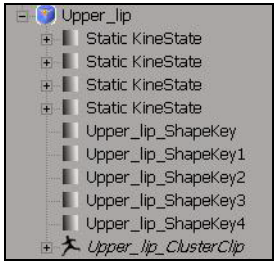
17. Double-click the first track. The details of the animation for the upper lip cluster appear:
    - Blue vertical lines indicate keyframes (they’re only one frame long).
    - Purple bars indicate automatically generated standard transitions.
    - The turquoise bar indicates that the last keyframe is being held for the remainder of the track.

You can continue to animate the shape while the track is displayed.



18. To return to the two-track view (which you saw earlier), double-click on the blank light-gray area of the track. The second track contains the shape keys of the Left_Corner cluster.





Animating Individual Points

19. Tag the group of points at the tip of the nose. Move the playback cursor to frame 1 and save a keyframe as you did for the other shapes.

    A new track is added to the animation mixer for these points.

20. Move the playback cursor to frame 10, deform the nose, and save another keyframe.

    This is just to show that you can animate individual points without needing to define a cluster first: XSI creates a cluster for you.

21. In the explorer, expand the node for one of the clusters and look at the shape keys, such as for the upper-lip cluster.

Each keyframed deformation is listed in the tree. These entries are not recordings of the entire cluster at each keyframe but, rather, are a list of the changes (differences). The first keyframed entry (the Reference Key Shape called Upper_lip_ShapeKey) contains the locations of all the points in the entire cluster. The subsequent entries contain only the offsets from that reference. This means that if you change the original reference entry, the entire animation for that cluster is modified.

For example:

- Select the Upper_lip cluster.
- Move the playback cursor to the first frame.
- Move the cluster.
- Choose **Deform > Shape > Replace Shape Key**.

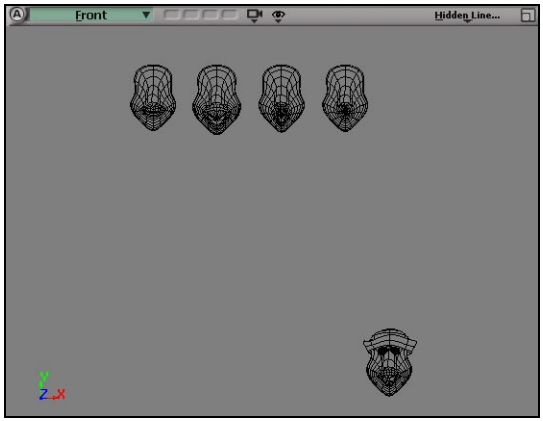
By doing this, you have changed the starting point for the animation. Since all subsequent keyframes are merely offsets of the starting point, the changes ripples down throughout the entire animation. Move the playback cursor and see.

Selecting Shape and Using Mixed Weight Mode

Now try using mixed weight mode to shape-animate using a series of predefined shapes. To continue, you need to display a "fresh" unmodified model. There is one conveniently prepared for you on a separate hidden layer in the scene.

22. Choose **Layers** > **Layer Control** in the main command area. In the Layer control box, deselect the View for the Default layer and select **View** for the Weight_Example layer.

    This layer contains four additional sailor heads with mouth shapes that correspond to different phonemes (zoom out in the Front view to see them).



23. Pick the main sailor head and choose **Deform** > **Shape** > **Mixed Weight Mode**. This mode lets you weight shapes as is done in SOFTIMAGE|3D.

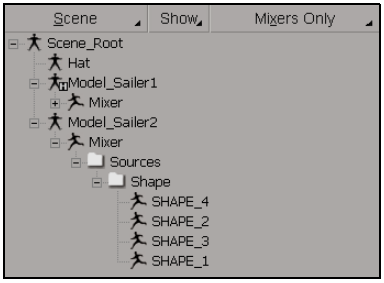
24. Choose **Deform** > **Shape** > **Select Shape Key** (at any frame). **Select Shape Key** lets you select many existing shapes, and saves each shape in the explorer for use in animating the original shape. It does not create a clip in the animation mixer.

25. Pick each of the four additional sailor heads so that all are selected. Right-click to quit the pick mode.

26. In the animation mixer, choose **View** > **Clear All**. This erases any information from the previous Transition method.

27. Open the explorer and locate Model_Sailor2. Lock the explorer.

28. Expand the tree for **Mixer** > **Sources** > **Shape**. The entries for the four shapes of the sailor's head are displayed (make sure **Mixers Only** is selected in the explorer's command bar).



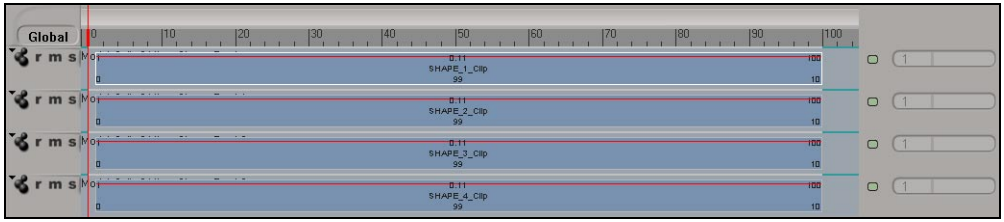
29. Select the main sailor head in the viewport and choose **View** > **Update From Selected** in the mixer.

Two animation tracks appear by default. Delete them (right-click on each and choose **Delete Track**), because you will be creating shape tracks.

30. Choose **Track** > **Add Shape Track** in the animation mixer or press Shift+s—add four shape tracks in total.

31. Drag and drop each of the shapes from the explorer onto their own tracks in the animation mixer.

32. Move each shape clip on its track to start at frame 1, and scale each to be 100 frames long (press Ctrl and select the clips, then scale them all at once).



You can mute a track (click the **m** button on the left of the track). Muting all tracks gives you the default shape.

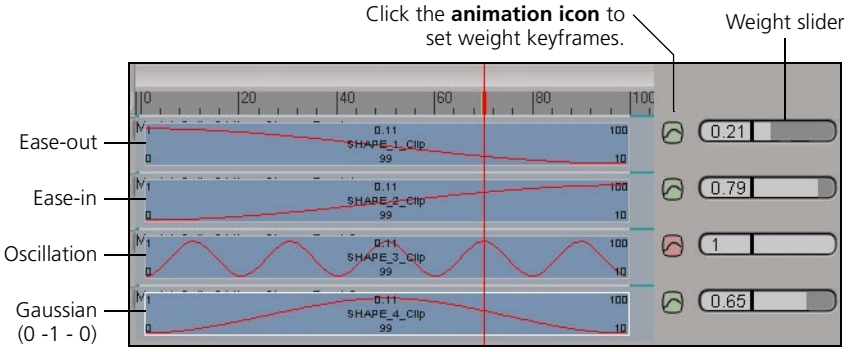
To the right of each track is a weight slider and a green animation icon. The sliders lets you mix the shape actions together, and the animation icon lets you set a key. The greater the value in the slider, the more weight that shape has in the mix.

33. Move the playback cursor, pick a spot, change the values of the weight sliders, and keyframe each track. Note how the shape action's weight curve is updated on the clip.

34. Repeat this process a few times using different slider values. Note how the mouth motions of the sailor's head change.

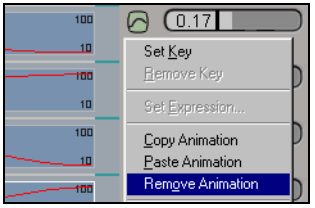
35. Use some of the preset weights in the **Clip** > **Preset Weights** menu. Preset weights can save you the effort of keying for some types of weighting.

In the following illustration (obviously not a typical setup!), the ease-in and out, oscillation, and Gaussian preset weights were used.



If you want to fine-tune any of the shape weight function curves, right-click on the animation icon for that track and choose **Animation Editor**. Any changes you make are reflected in the shape clip in the animation mixer.

Creating Custom Parameters for Controlling the Mouth

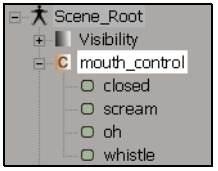


Now you will use custom parameters to control the shape action's weights.

36. Remove the animation from each of the tracks by right-clicking each animation icon and choosing **Remove Animation**.

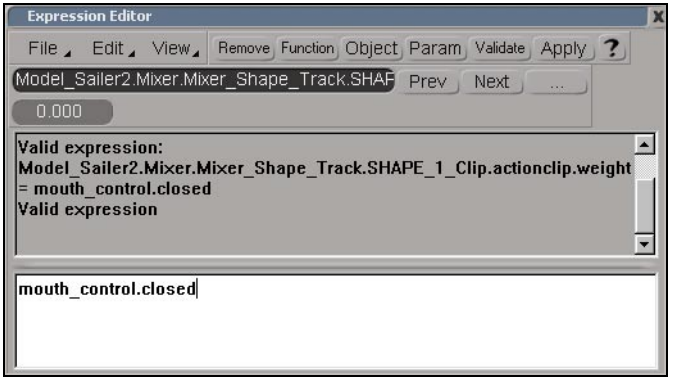
Your first step is to create a set of custom parameters—one custom parameter for each expression to be used on each of the sailor heads.

37. Open an explorer and select the root of the scene.
38. From the Animate toolbar, choose **Create** > **Parameter** > **New Custom Parameter Set**. Call the new custom parameter set `Mouth Control`.



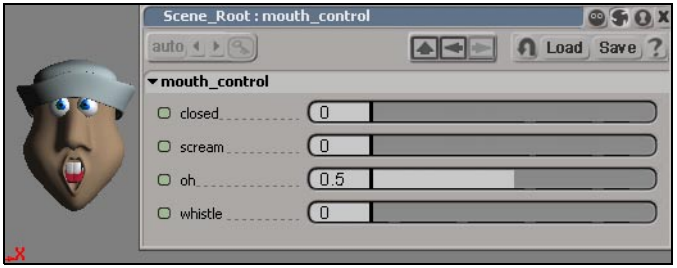
39. With the Mouth Control custom parameter set selected (select the All+Animatable parameters filter in the explorer), choose **Create** > **Parameter** > **New Custom Parameter** and call the new parameter `Closed`. Leave everything else as default.
40. Create three more custom parameters called `Scream`, `Oh`, and `Whistle`. You now have one custom parameter for each of the four sailor heads.
41. Select the first track, right-click on the animation icon and choose **Set Expression**.

42. In the expression editor, delete any text that might be lingering from a past session.
43. Click the **Object** button and select the **Closed** custom parameter from the tree. This links the first shape clip to the Closed custom parameter.
44. Click **Validate** and **Apply**, and keep the expression editor open.



45. Assign the Scream, Oh, and Whistle custom parameters to the other three shape clips the same way as you did for Closed.

To view your set of custom parameters, double-click the Mouth Control node in the explorer. A property editor containing four sliders appears. Here, you can animate and keyframe the mouth motions of the sailor. This is the same as using the weight slider for each track in the mixer.



Conclusion

You have seen how to create very basic shape animation from either saving shape keys or selecting shape keys from predefined shapes to be applied to a model. You've also seen how you can set up your own custom slider panel for controlling shapes. To go further, add some audio tracks in the mixer and synchronize the sound with the mouth shapes.

For more information, see the following chapters in the *Animating* guide:

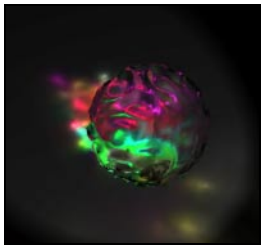
- *Chapter 10: Custom and Proxy Parameters*
- *Chapter 13: The Animation Mixer*
- *Chapter 15: Shape Animation*
- *Chapter 16: Sound and Animation*

Section 8

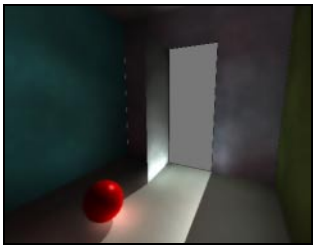
Caustics, Global Illumination & Final Gathering

Fake it? Forget that!

With global illumination, caustics, and final gathering, you can light a 3D scene using true lighting principles based on photon emissions and bouncing, as opposed to using lighting and texturing tricks to fake the phenomena.



Caustic lighting



Global illumination

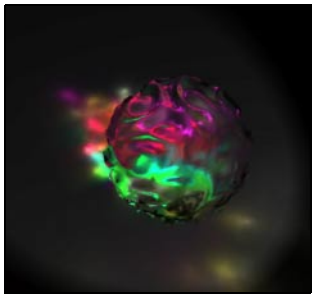
Final gathering

SOFTIMAGE|XSI provides support for caustics, global illumination, and final gathering, thanks to the new features of mental ray® rendering software version 2.1.

The following three tutorials show you the basics of each feature.

Tutorial 13: Caustic Lighting—Psyched-Out Billiard Ball

Caustics are created by the focusing and dispersion of light by specular reflection or refraction onto a diffuse surface.

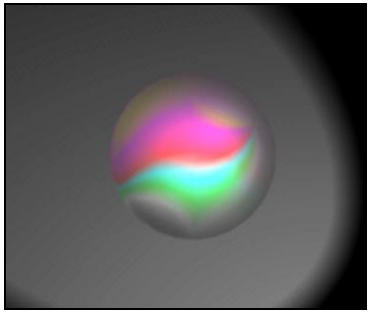


This tutorial shows you how to:

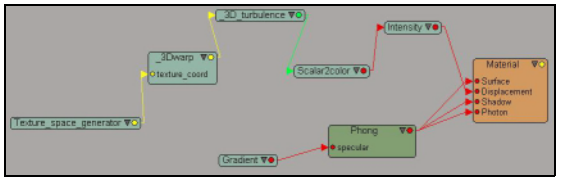
- Use the render region to interactively preview lighting effects.
- Create caustics emitted from a sphere with displacement and a specular gradient.

Overview

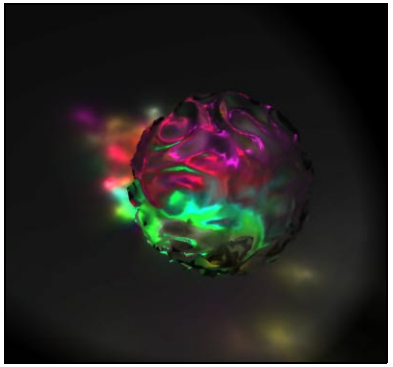
1 **Create an object with a gradient-textured specular.**



2 **Give the object displacement using the render tree**



3 **Enable the caustic effects**



Caustics

In XSI, caustics are controlled from both the light property editor and the render properties property editor. The following list describes the most useful parameters to use when creating a caustic effect.

Caustics parameters

These parameters also apply to global illumination.

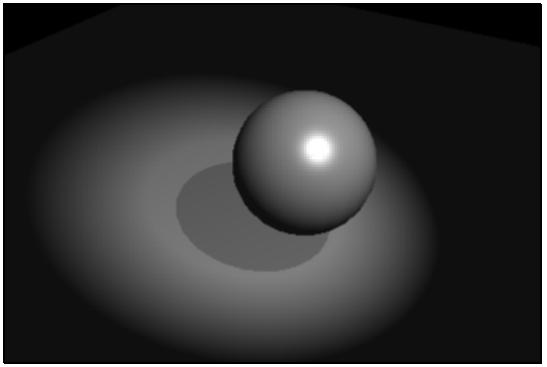
- **Photon energy** is the power (wattage) of the light source. The greater the energy, the farther the photons will travel. The rate at which the photon loses its energy is inversely proportional to the square of the distance travelled. Doubling the distance traveled reduces the energy level of the photons by four. Photons also lose energy each time they bounce off objects in the scene.
- **Number of Photons** is the precision of the indirect illumination effect (for either caustics or global illumination).
- **Accuracy** is the number of photons considered by the material shader that examines the photon map to obtain the caustic or global illumination contribution. (This option is located in the Render Options property editor.)
- **Accuracy Radius** is the maximum distance within which mental ray searches for photons. (This option is also located in the Render Options property editor.)

Building the Set

In order to demonstrate the basic concepts and functions of caustic lighting, you will build a simple scene that uses a sphere's specular value to create caustic lighting.

1. Begin with a new scene.
2. If you have another scene already open, choose **File** > **New Scene** or press **Ctrl+n**.
3. Get a sphere and choose **Get** > **Primitive** > **Surface** > **Sphere** from the Model toolbar.
4. Give it its own Phong material using **Get** > **Material** > **Phong**.
5. Get a grid and place it beneath the sphere. You may want to scale the grid so it is several times larger than the sphere and makes a good backdrop.
6. Select the new scene's default light and delete it.

7. Get a spotlight and position it above the sphere. From the light's property editor (**Modify** > **Shader**), narrow the light's cone and switch on its shadows (Umbra = 0.2). So far, your scene should look more or less like this (below):



8. In order to create striking caustics, you will have to eliminate the scene's Ambient Lighting. Do this by choosing **Modify** > **Ambience** from the Render toolbar and setting the Ambient Color to black (0, 0, 0). Ctrl+click to drag all the sliders at once.

When creating lights and placing them, you may find it useful to work in the Shaded view mode.

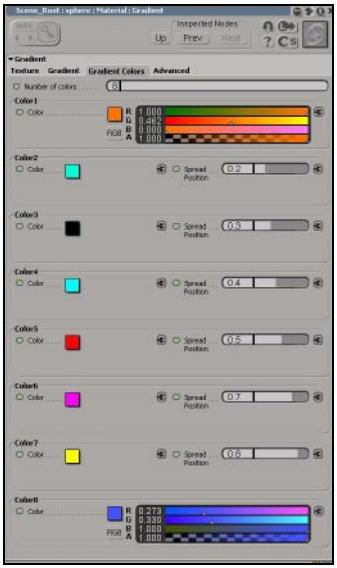
Polishing the Specular

Because caustics are driven by specular values, you will use a color gradient to drive the specular value of the Phong you applied to the sphere.

9. Select the sphere and open its property editor using **Modify** > **Shader** on the Render toolbar.

Connection Icon

10. Click the connection icon to the right of the Specular color sliders. A pop-up menu opens listing shaders that you can connect to the Specular parameter. Choose the **Gradient** shader.
11. The Gradient shader's property editor will appear. From the Texture tab, create a new texture projection for the texture. A spherical projection will probably work best here.
12. Click on the **Gradient** tab and define a **Radial Rainbow Gradient Mode**.

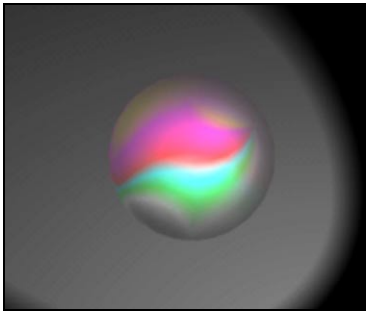


Use the Gradient shader to define a ramp or gradient of colors to drive the Specular parameter.

13. Next, select the **Gradient Colors** tab and start choosing your colors. You can leave the Spread values as they are. Also, try not to use any white colors: they may dilute the final effect.

To help tweak your colors, you can open a render tree view (in the C viewport through the Views menu), select the sphere, press update, select the gradient node, and use the Preview mode (press **p**). Now draw a render region in the B viewport. Don't forget to press **p** again when you are finished editing the shader's values.

14. From the **Advanced** tab, define a Repeats value of 2, 2, 2 to repeat the texture twice in all axes.
15. Click the **Up** button in the Gradient's property page
16. Set the **Phong's Specular Decay** to 7, **Transparency** (RGB value) to 0.7 (use the Ctrl key to move all sliders at once), and the **Index of Refraction** to 1.2. To speed up rendering a little, switch off the Reflection parameter.



Once you have edited the Phong shader's parameters, the sphere should resemble the one depicted at right. If it doesn't, make sure you have defined a texture projection for the texture shader.

Setting Up Displacement

In order to create some funky caustic effects, give the sphere some displacement so the photons will have a little more to "bite into."

17. Open a render tree view, select the sphere, and press **Update**.
18. From the **Nodes** drop-down menu, select the following three lesser-known shaders. You'll use them to create displacement on the sphere's surface.

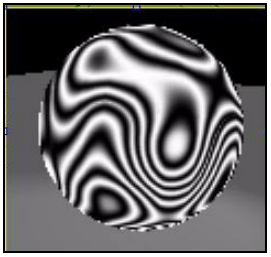
    - Texture Space Generators > Generator
    - Texture Space Controller > 3D_Warp
    - Texture Generators > Turbulence

19. Connect the Texture Space Generator to the 3D_Warp's **texture coord** input by clicking and dragging an arrow from the node's dot on the upper right of the shader. Then connect the 3D_Warp to the Turbulence's **coord** input.

20. Before you connect this branch to the sphere's Displacement, connect the branch to the sphere's Surface input. This will help you visualize what is going to be applied to the displacement input. In order to do this, you will need the Scalar2Color shader . Get it from the **Nodes** > **Conversion** menu. This shader allows you to connect two shaders of incompatible input/outputs.

You can also select the **p** key for automatic preview at this point on the tree. Make sure you turn it off; otherwise, it will try to preview from any selected nodes.

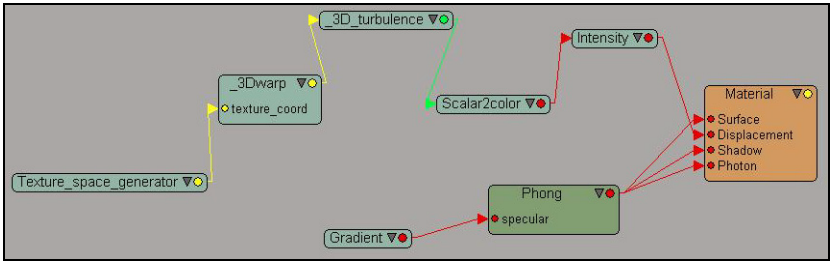
The texture space generator creates a the texture space needed to apply the turbulence pattern. The warp shader is used to warp the texture space used by the turbulence shader. Your sphere should more or less resemble the image below if this branch is connected to the Surface input:



Before connecting the warping effect to the displacement input, connect it to the Surface input (using the Scalar2Color shader) so you can get an idea of what will happen once this texture is used as a displacement map.

21. Double-click the Texture Generator node to open its property editor. Give it a texture projection of your choice. To obtain something similar to the image above, try setting the Texture Space to Intersection and the Projection Method to Lollipop.

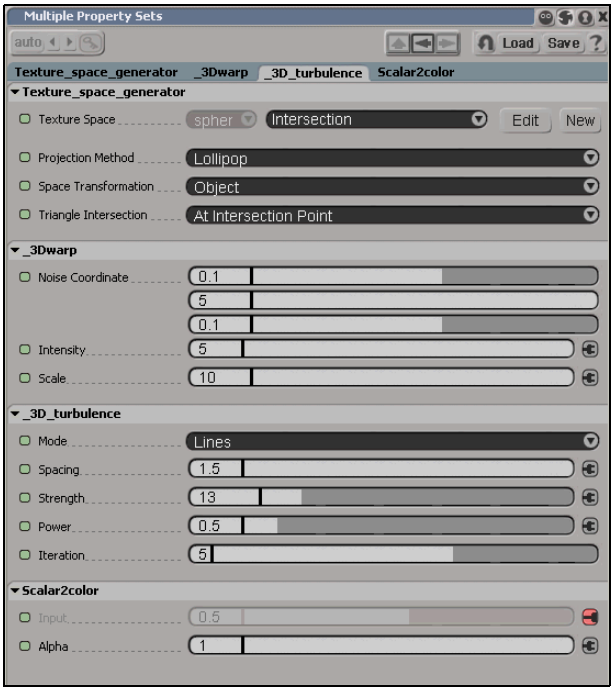
22. Again from the **Nodes** menu, get the **Conversion** > **Scalar2Color** shader. Use this shader to connect the 3D Turbulence shader to the Displacement input of the material node.



23. You can further control the turbulence/warp effect by using an **Image Processing** > **Intensity** shader node after the Turbulence shader. Once connected to the Displacement input, your render tree should resemble the tree above.

24. Open the **Intensity** shader property editor and set the **Factor** parameter to 0.5 to smooth the warping effect.

25. Use the following values to guide you in editing these three shaders:



You can display several property editors at once:

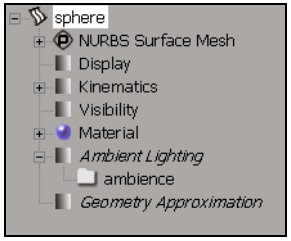
Ctrl+Select the nodes from the render tree work area and press Return. A multi Property Sets property editor will open, listing the shaders in the order they were chosen.

You can also branch-select a section or a whole render tree and display its property editor.

26. In order to smooth the displacement effect a little, you may wish to edit the sphere's geometry approximation. Simply select the sphere and click on the Selection button in the main control area. Select **Geometry Approximation** and make a local copy. Under the **Surface** tab, select **Parametric** and enter 2 for both the U and V steps. Under the **Displacement** tab, select **Parametric** and enter 2 in subdivision steps. Keep in mind, however, that this will increase rendering time, as displacement accuracy is increased.

Define caustics

After creating and coloring this funky sphere, you will finally define the caustic receivers and transmitters.



Selecting the Visibility icon opens the Visibility property editor.

27. Select the sphere and click the **Selection** button from the Selection panel of the main command area. Once the explorer view of the selection opens, click the Visibility icon.
28. Define the sphere as a Caustic transmitter by checking the **Caustics Transmitter** box (you can also check Caustic Receiver in conjunction).
29. Repeat the process for the grid, but select **Caustics Receiver** only.

Define caustic light

30. Select the spotlight and press **Modify** > **Shader** from the Render toolbar to open the light shader's property editor. Lock the property editor open by clicking the lock/pin icon.
31. Before activting caustics, reduce the light's Intensity to 0 so only photons will affect the scene's lighting. You will bring the intensity back up when the caustic effect is completed.
32. Click the **Photon** tab and switch on **Caustics**. If you have a render region open, you will notice that nothing is rendering. This is because you have to activate caustics for the render region.

Activate caustics

33. Open the render region options property editor by selecting **Render** > **Region** > **Options**. Keep this property editor open as well by locking/pinning it open.
34. Select the **Photon** tab and switch on **Caustics**.
35. You should see caustic effects with the default values, but try playing around with them some. You may want to set the render region to not Auto-Refresh (**Render** > **Region** > **Auto-Refresh**) so you can simply update the parameters and click **Refresh** when ready.

Playing with the Caustic and Lighting Settings

Caustic settings in the View Render Options property editor:

- **Accuracy** specifies the resolution of the photon map. Try a value around 200.
- **Radius** specifies the radius of the photons as they strike the receiving object. Larger photons illuminate more of the object. A value of zero allows mental ray to make a "best guess" value for the scene. A value of 0.05 lets you see individual photons. Increase the value by small increments to see the differences. Try a value of 2.
- **Filter Type** specifies the type of blurring effect applied to the photons.

Energy and number of photons settings in the Light property editor:

- **Color** is the color of the light transmitted by the caustic emitter.
- **Intensity** is the energy level of the photons. Increasing this makes everything brighter. Try a value of 30 000.
- **Number of Emitted Photons** specifies the quality of the caustic effect. Try a value of 50 000.

36. You can now bring in a bit of direct illumination by setting your spot light's intensity to 0.2.

You can open up the GLB_e2_caustics_done scene which has the completed scene

Conclusion

You created caustic lighting that is driven by an object's specular value and, to a lesser extent, its transparency. Common uses for caustic lighting include an illuminated water bottle (a refractive object) that casts a focused caustic effect on a diffuse table surface. A similar example is the pattern of light cast on the bottom of a swimming pool by the wave pattern on the water's (refractive) surface. Caustics can also be caused by reflections, as in the case of a mirror.

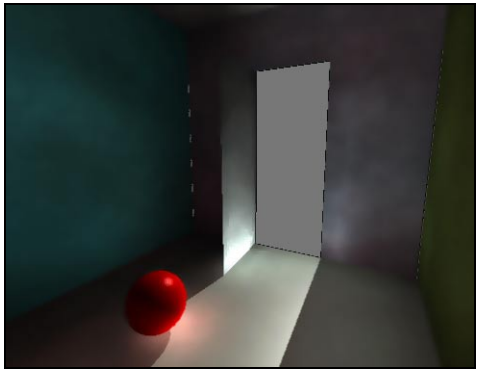
Applying caustics to a scene is very processor-intensive, so, to reduce the rendering time, the caustic effects should be localized. You accomplish this in XSI by specifying the light source, the caustic emitter, and the receiving object(s) in your scene. The caustic effect is then limited to these objects.

For more information, see *Chapter 6: Global Illumination & Caustics* in the *Shaders, Lights & Cameras* guide.

Tutorial 14: Global Illumination—The Ball Room

Global illumination simulates the lighting phenomenon that occurs when light reflects off objects and causes the surroundings to inherit some of the object's color.

A good example of what global illumination brings to the scenes that you create in SOFTIMAGE|XSI can be seen in a photographer's studio. A photographer uses reflectors and diffusers to illuminate the subject with reflected light. The color of the reflector colors the light bouncing off it and onto the subject, thereby coloring the subject.

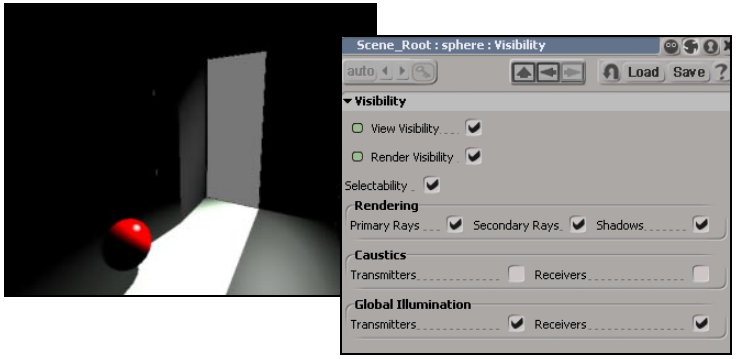


This tutorial shows you how to:

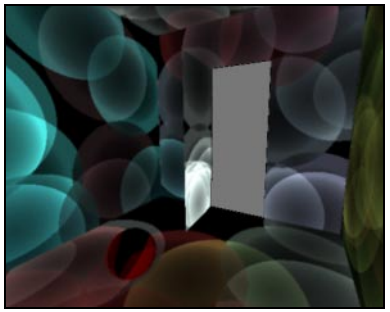
- Use the render region to interactively preview lighting effects.
- Define global illumination to add reflected light effects from a sphere within a closed "room."

Overview

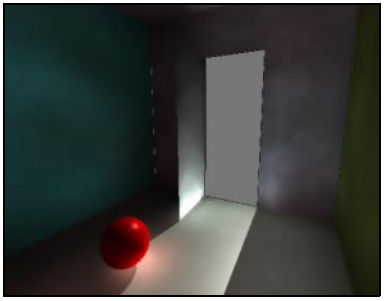
1 Define the transmitters and receivers.



2 Set the light to emit global-illumination photons.



3 Adjust the global-illumination effect.

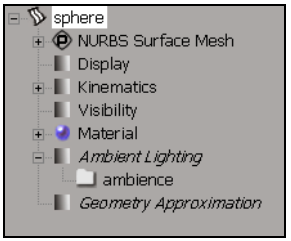


Applying Global Illumination

1. Open the **GLB_e1_room** scene from the Tutorials project.
2. Draw a render region. You can see how the single light source outside the room illuminates the floor and red sphere.

Normally, you would bring a scene's Ambience to 0 before applying any type of global illumination effect. Although this scene's is already at 0, you can easily define the Ambience by selecting **Modify** > **Ambience** from the Render toolbar.

3. Multi-select all the objects.



Clicking the Visibility icon opens the visibility property editor, where you can decide whether your object(s) will receive or transmit photons, or both.

Setting the Transmitters and Receivers

You first want to attach the receiving and transmitting property for global illumination to the objects in the scene:

4. Open a **Property Editor** view in the C viewport. This view will "dock" a property editor inside a viewport.
5. In the main command area, choose **Selection** from the Select panel and click the Visibility icon. The visibility property editor for all the selected objects in the property editor viewport opens.
6. In the property editor, activate **Global Illumination Receiver** and **Global Illumination Transmitters.** This will define every selected object as both a receiver and transmitter of global illumination.

Because global illumination and caustic effects are render intensive, you may wish to swirch off the Auto-Refresh of the render region so a new render is not launched every time you tweak a parameter. To do this, select **Render** > **Region** > **Auto-Refresh**.

7. Now activate global illumination in the light source: Select the light and choose **Modify** > **Shader.** The light shader property editor will open in a floating window or in the property editor viewport if it is still open.
8. Switch on **Global Illumination** in the light's property editor and lock the editor (using the lock icon).

Seting Up the Render

9. From the Render toolbar, choose **Render** > **Region** > **Options**.
10. In the property editor that opens, click the **Photon** tab and activate **Global Illumination**. You will probably want to keep this property editor open as well.
11. In the Render toolbar, choose **Render** > **Refresh** to see the result.
12. Try tweaking the settings (Number of photons, Accuracy, Accuracy Radius in the Light and Render property pages) to create different results. You will probably want to start experimenting with very few photons; for example, 100.

As soon as you start to see an effect, bring down your light's intensity to 0 so you are seeing global illumination lighting only.

13. Once you have a number of photons you are happy with, edit the **Radius**. Try a value of 1 to diffuse the photon effect. This parameter commands the renderer to search for more photons in the scene.
14. Use the global illumination Accuracy parameter to smooth the effect, much like a sampling.

In the render region, you may notice that the red sphere is very saturated and unrealistic. You can edit the photon characteristics of specific objects by simply opening their property editor.

15. Select the sphere. From the Render toolbar, select **Get** > **Shader** > **Photon** > **Edit**. This will open the Phong's photon property page.
16. Reduce the intensity of the red from the **Diffuse** parameter and notice the difference in the render region.

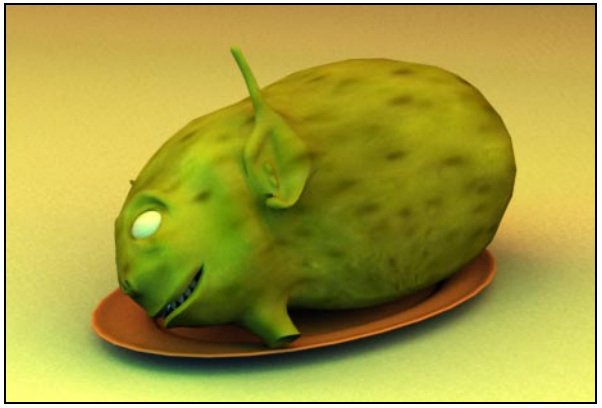
Conclusion

Global illumination is a simple solution to achieving real-world lighting. In this tutorial you used a colored sphere to influence the lighting in a colorless room. When global illumination is used in a larger scene with many objects, you can simulate how real light affects every object it touches, taking into consideration their proximity to other objects, their distance from the light, as well as other lighting in the scene.

For more information, see *Chapter 6: Global Illumination & Caustics* in the *Shaders, Lights & Cameras* guide.

Tutorial 15: Final Gathering—Making an Alien Look Real

Final gathering computes single bounces of indirect light on diffuse surfaces. It gives you details in areas of the scene where there are many edges and sharp light variations. It can be used alone on scenes with a lot of direct light or in conjunction with global illumination. The rule of thumb suggests that if the strength of the direct light in a scene is greater than 50 percent, then final gathering can be useful.



When using final gathering alone, you don't need to set anything on the Photon property page. In fact, most light options are redundant, since the main source of lighting will be reflected light.

Final gathering effects are achieved from the render options property editor.

This tutorial sees the unfortunate return of Frank the alien who retired from tutorial life a few pages ago. One can only imagine his gambling troubles caught up with him.

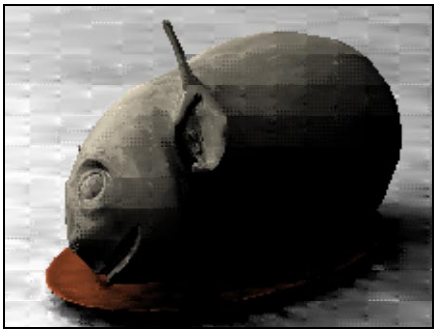
Poor Frank.

This tutorial shows you how to:

- Set up lights for use with final gathering
- Use a reflector to bounce light
- Create a radiosity map
- Simulate "real word" lighting without using reflected light and no photons.

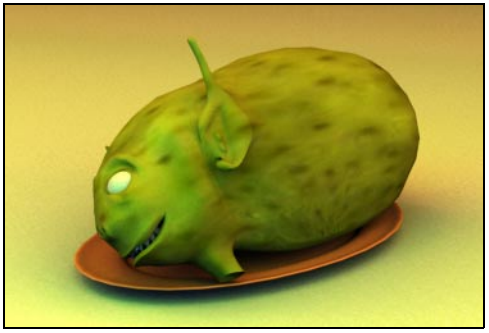
Overview

1. **Set lights and activate final gathering.**



2. **Add a texture to create a radiosity map.**

3. **Tweak final settings to smooth final-gathering effect.**



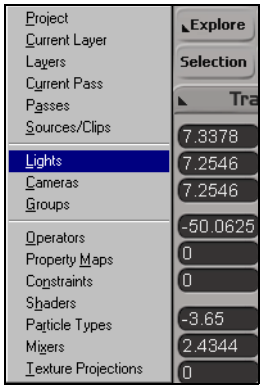
Using the Reflector Technique

A constant shaded surface can replace the presence of a light in a final-gathering scene. For example, if you have a white constant dome or grid over your scene, you will be simulating an overall, diffused lighting. If no constant surfaces are in the scene, you will then need at least a single light to trigger the final-gathering effect.

1. Open the **GLB_e3_alien** scene from the Tutorials project.
2. Press **a** while placing the cursor over a viewport. This will "zoom out" the viewport view to Frame All elements of a scene. Notice how a simple curved grid object surrounds the alien head.
3. Select the overhead grid-like object and apply a Constant shader to it using **Get** > **Material** > **Constant**. Because it has a Constant material shader applied to it, the grid-like object acts as a large reflector in the scene, bouncing light rays to illuminate the alien's head. You can leave the shader's default values as is and close the property editor.

- The Reflector technique is good to use when an object is in a wide, open space. Using an entire sphere would cause unnecessary calculations underneath the scene's floor, hence create longer render times.
- This technique is very similar to the white-board lighting method used in cinematic and photographic lighting, where large reflective surfaces are used to bounce sunlight into a scene or on a subject.

Setting the Non-Lights



4. Select the scene's infinite light (using the **Explore** > **Lights** command from the Select panel in the main command area), and open its property editor by selecting **Modify** > **Shader**.
5. Reduce the light's Intensity to 0. When applying final gathering, no active light is needed; you need only a single light with no intensity.
6. You can close the lights' property editor by pressing **Ctrl**+~.
7. If a render region isn't yet open in the camera viewport, create one now using the **q** key.

Because final gathering can cause extended render times on some computers, you may wish to deactivate the Auto-Refresh option: From the Render toolbar, choose **Render** > **Region** > **Auto-Refresh**. The render region's outline will change to red. You can then refresh it by clicking the **Refresh** button or middle-click when in region mode (**q** key).

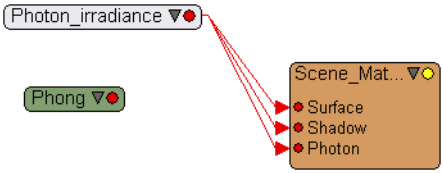
8. Eliminate the scene's Ambience by selecting **Modify** > **Ambience** from the Render toolbar. Reduce the Ambience value to 0,0,0. This step should eliminate all lighting from your scene and produce a beautiful, black rendering.

To see the final-gathering effect, make sure the render region is set to display the RGB only, not the Alpha.

Setting the Materials

Most of the scene's materials and textures have already been defined. You won't need to bother about the alien head and the plate for the time being.

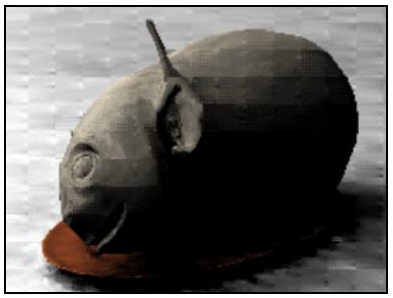
9. Select the grid that acts as a floor underneath the plate.
10. Open a render tree view in a viewport or in a floating window (press 7). If not already displayed, press **Update** to display the grid's render tree.
11. Select the **Nodes** drop-down menu and choose **Illumination** > **Photon Irridiance**.



12. Connect the Photon Irridiance shader to the Surface, Photon, and Shadow inputs of the material node (as shown below). This will disconnect the Phong shader node. You can leave this shader there. Someone will sweep it away later.

Activating Final Gathering

13. Open the Render Region Options property editor by selecting **Render** > **Region** > **Options** from the Render toolbar.
14. Select the **Photon** tab and activate **Final Gathering**. The render region will then render using final gathering. The render doesn't use any direct illumination—all the light is created by the reflection caused by the Constant white material of the overhead grid.



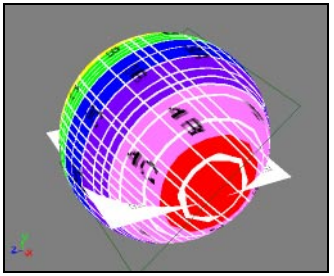
Controlling the Reflected Light

15. Select the reflector grid above the scene and press Return to open its Primitive property editor.
16. You will notice that this object is in fact a sphere that has been opened, and not a grid. You can control how "open" or "closed" the sphere will be by playing with the Start and End U and V parameters. As you open the sphere, notice how more light is reflected throughout the scene. For this tutorial, open the sphere completely using 0 and 360 as the Start and End U values and 0 and 180 as the Start and End V values.

    Notice how using the full sphere blasts more light into the scene since more light is reflected by the white Constant material.

Creating a Radiosity Map

Instead of using a white Constant shader to reflect light into your scene, you can define a texture. You could use a panoramic image to simulate real lighting in a given environment. In this tutorial, you'll simply use the colorful no icon.pic to illustrate how the color from a texture is reflected using final-gathering light rays.



The No Icon texture will surround the scene, reflecting its colors using final-gathering light rays.

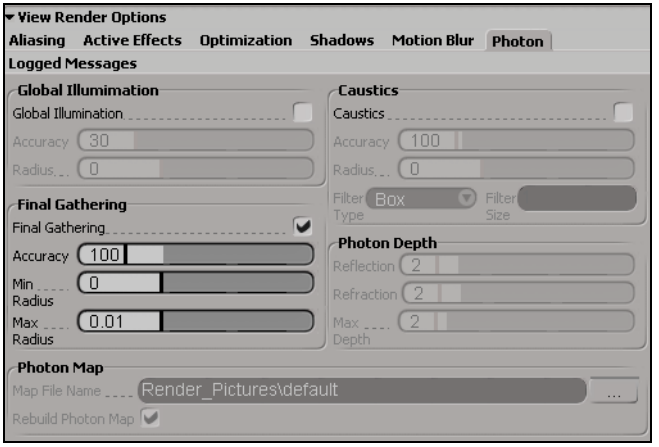
17. Select the sphere that surrounds your scene.
18. From the Render toolbar, choose **Modify > Shader** to open the Constant shader's property editor.
19. In the property editor, click the connection icon beside the Color parameter's sliders. A pop-up menu opens, where you can select a shader to connect to the Color parameter. Select the **Image** shader. This opens the Image shader property editor.
20. From the Image shader property editor, Use the **No_Icon.pic** and define a Planar XY projection. A No Icon texture will now surround the scene and no longer reflect white light everywhere. Your render region will resemble the following image:

21. The rendering is very blocky, so now you can start defining the accuracy settings of the final-gathering effect. In fact, this is where the final-gathering effect is really achieved. From the Render toolbar, select **Render** > **Region** > **Options** to open the Render Region Options property editor.

22. Use the settings appearing in the following illustration to help you set the final-gathering Accuracy and Radii:

Using the final-gathering render settings is the most influential way to refine the final effect. Often, artifacts are visible in shadow areas of a scene, but these can be minimized by reducing the **Minimum** and **Maximum Radius** settings.

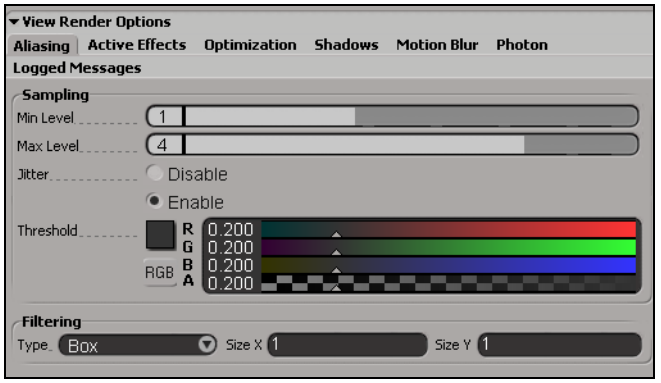
The **Radius** parameters define how detailed the final-gathering effect will be. But be careful: values less than 1 may result in extended render times.The **Accuracy** parameter works in conjunction with the **Radius** parameter to smooth out the final-gathering effect.



Try keeping the Accuracy setting low (< 100) while tweaking the effect. Once you are ready to render, bring it back up. A high Accuracy value will blend the noise onto the surface. If you have a powerful machine, you may wish to push the Accuracy parameter even higher (> 500).

Even though you may be using small Radius values and a higher Accuracy setting, you may still need to optimize the render by increasing the **Sampling Minimum** and **Maximum Levels** as shown. Of course, the higher the sampling values, the longer the render time will be.

In some cases, activating the **Jitter** parameter may help reduce artifacts. You can also try activating **Dithering** from the Active Effects property page. This can reduce color banding.



23. To change the lighting effect, try translating or rotating the sphere or its texture support object. For example, try a translation of the texture support in X (3 units).

Adding Textures to a Final-Gathering Effect

Rather than leave the alien's head with a standard Phong material surface, you can introduce textures into the scene. Here, you will simply connect a texture to the alien head's Diffuse parameter.

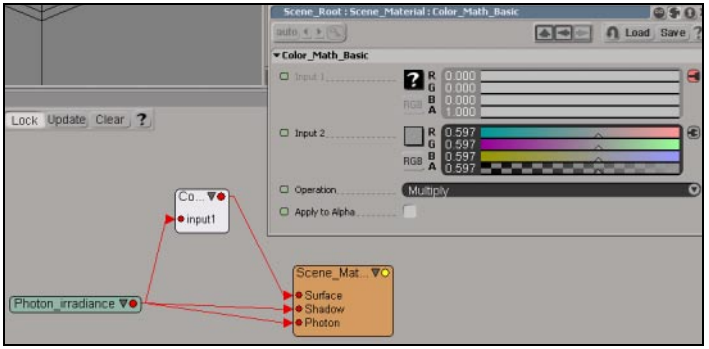
24. Select the head.
25. From the Render toolbar, choose **Get** > **Material** > **Phong**.
26. Once the Phong property editor opens, click the connection icon beside the Diffuse parameter and select the Image shader from the pop-up menu.

27. This applies an Image shader to the Diffuse parameter only. In the Image shader property editor, define a texture image to use. Click **New** > **New from File...** . Select the **Poire_pic** from the Tutorials picture folder and click Ok.
28. Rather than create a new texture projection, select **Texture Projection 1** from the Texture Projection drop-down menu in the Image shader property editor.
29. To give it more realism, switch on Bump Mapping and set the Bump Factor to –5.

Do the floor

To fine-tune the intensity of the floor, you can apply a tool shader to the grid using the following steps.

30. Select the grid under the plate.
31. Press **7** to open a floating render tree view.
32. From the Nodes menu, select the **Nodes** > **Math** > **Color Math Basic** shader.
33. Connect it in between the Photon Irridiance shader and the Material Node's Surface input as shown (following).



Conclusion

Once you find the proper balance of radius size, accuracy, and sampling, final gathering can help you create realistic lighting without photons. And because photons aren't used with this process, you can, with a bit of testing, create scenes with photorealistic lighting in a fraction of the time it takes to achieve through more conventional methods (such as global illumination).

For more information, see *Chapter 6: Global Illumination & Caustics* in the *Shaders, Lights & Cameras* guide.

Section 9 **Light & Effects**

Tutorial 16: Basic Lighting—Frank's Bust

Take a fantastic model, with incredible texturing, and put it under mediocre lighting. What a waste! Lighting can make all the difference when it comes to the look of your models and scenes. This tutorial will quickly explain the basics of good lighting in order to get the most out of your object or scene.

This tutorial shows you how to:

- Use the render region to interactively preview lighting effects.
- Use the Spotlight view to edit your spotlight's cone and point of view.
- Create lights.
- Position and edit lights in order to get the most out of your scene or object.

Overview

1 **Create and place the Key light.**

2 **Create and place the Fill light.**

3 **Create and place the Back lights.**

The Three Basic Lights

Frank the alien has finally retired from tutorial workbooks and has been inducted into the Tutorial Hall of Fame despite a gambling scandal. In his honor, a bust was created and placed within the Hall. Unfortunately, the Hall's caretakers know nothing about lighting (as demonstrated by this ghoulish setup). It is up to you to bring out Frank's best features one last time.

When lighting a scene or object, three basic lights are used:

- Key light
- Fill light
- Back light

Usually, each light type is created with a spotlight. The size of the light cone, light spread, and falloff distances are all relative to your scene.

Because every scene is different, for this example we will use a simple model with a simple background. The lessons taught in this example can be carried over to any other scene or object.

Key light

The key light is used as the main source of illumination. It should not only light your subject but create shadows as well. For example, a scene set by a sunlit window would have the sunlight as its key light.

The key light shouldn't be too close to the camera or it will wash out details on the object and kill shadows. Instead, the light should create an outline of the object against the background.

Fill light

This is used to fill shadows or shaded areas created by the key light. The fill light will always be weaker than the key light. By what percentage depend on the type of mood you wish to create. Standard lighting requires the fill to be 1/3 as strong as the key light. For example, if your key light has an intensity of 1, the fill should set at 0.33.

Back light

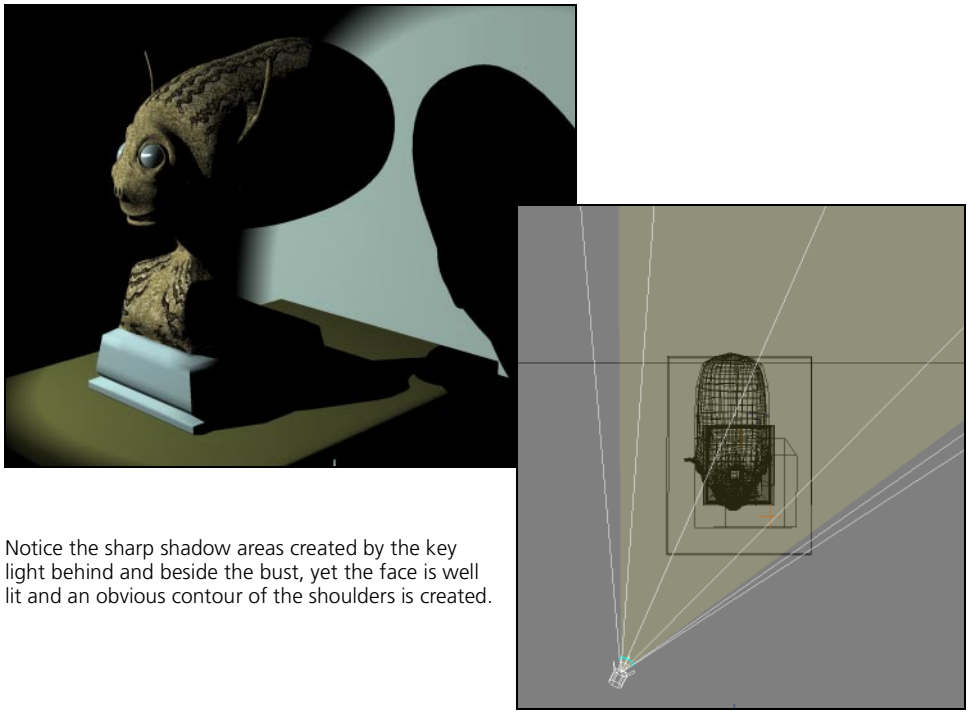
Also called the ramp light, this light is used to create speculars in the scene. The back light doesn't create shadows and has the same intensity as the key light. The back light often faces the camera but is placed at a high angle within the scene so a flare isn't created.

1. Open the **LIT_e1_alien** scene from the Tutorials project.
2. Any time you want to light a scene or object from "scratch," make sure that all of the lights in the scene are deleted, including the default light. You can quickly see what lights are present in a scene by selecting **Explore** > **Lights** from Select panel of the main command area. Simply select the light(s) and press Delete to remove them from the scene.
3. Once all lights are deleted, reduce the scene's ambience to 0. You can open the scene ambience property editor by choosing **Modify** > **Ambience** from the Render toolbar. Reduce the Ambient value to 0, 0, 0. Your scene should be black now.

When working to position lights and determine their intensities, you may find it very helpful to work in the Shaded view mode.

Defining the Key Light

4. Create the key light by selecting **Get** > **Light** > **Spotlight** from the toolbar. In the light's property editor, set the light's Umbra to 0 and the Intensity to 2.0. Also, make sure Shadows are on. Position it so you are lighting the alien bust well but creating shadows at the same time. Use the image below as a guide.



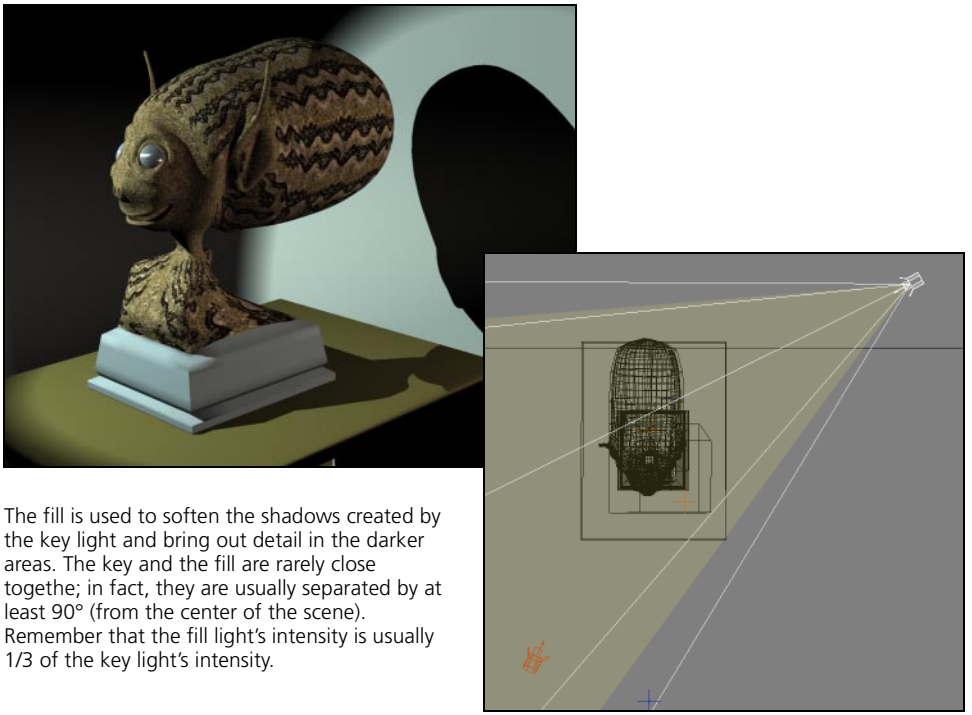
Notice the sharp shadow areas created by the key light behind and beside the bust, yet the face is well lit and an obvious contour of the shoulders is created.

A great way to "see" what your spotlight sees is to change the viewport view to **Spot Lights** (Views menu). Circles within the view show you the start and stop light falloff distances.

5. You may want to adjust the light's cone spread to achieve a more accurate lighting. To do this, select the light and press **b** to activate the manipulators. You can cycle through the various types of manipulators by pressing **TAB**. The manipulators are interactive: simply click and drag them to the proper setting.

Defining the Fill Light

6. Next, create the fill light by selecting **Get > Light > Spot**. In the light's property editor, set the fill light's Intensity to 0.66 and make sure that shadows remain disabled. Position it so you can just start to see details in the shadows. Use the image below as a guide:

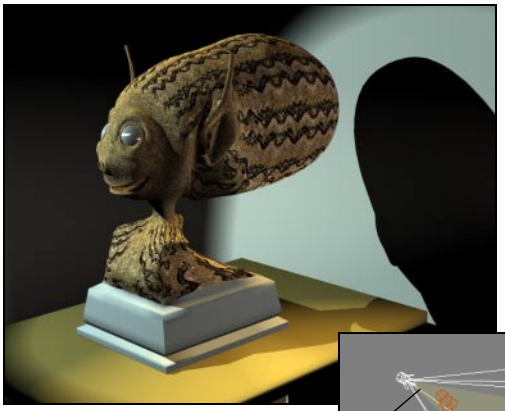


The fill is used to soften the shadows created by the key light and bring out detail in the darker areas. The key and the fill are rarely close togethe; in fact, they are usually separated by at least 90° (from the center of the scene). Remember that the fill light's intensity is usually 1/3 of the key light's intensity.

In order to add a touch of realism to your scene, try coloring the key light with a little blue and the fill with a little yellow. Experiment with different shades of these colors.

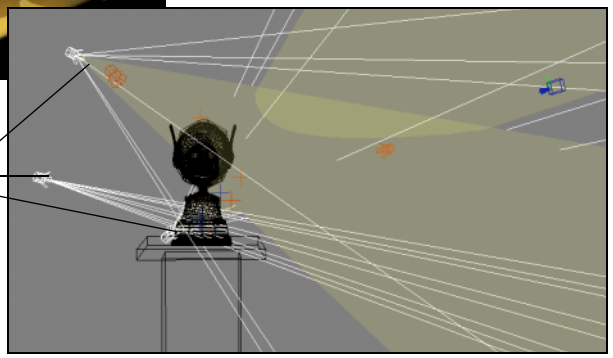
Defining the Back Light

7. Create the back light by selecting **Get** > **Light** > **Spot** from a toolbar. In the light's property editor, disable Shadows and increase the Intensity to 2.0.
8. Translate the light to a high angle, facing the camera a little.
9. Depending on your object, you may need to use several back lights. This scene uses three. Use the scene below as a guide:



The final image uses three back lights at the same intensity as the key light. Because the bust was so close to the wall, the key light was given an exponential falloff value of 2.

Frank can finally rest in peace knowing that he has optimal lighting.



Three backlights were used to illuminate different areas of the alien's extra-long head. Notice how the positioning of the backlights (right) is directed toward different pieces of the alien's head. This is where using the light's manipulators can come in very handy.

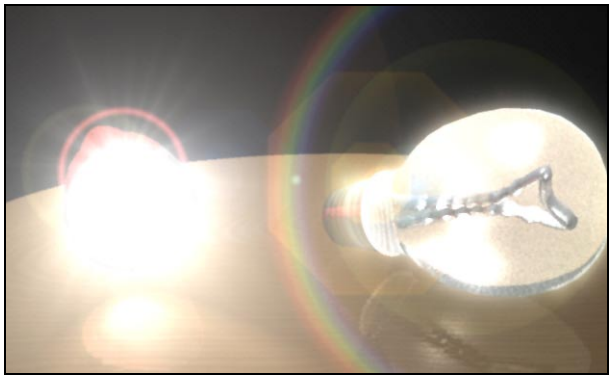
Conclusion

Each one of these light types has a specific function. If a light is positioned to fill more than one role, you may create scattered, hap-hazzard lighting. Using these three types of lights is meant to enhance the three-dimensional aspect of a scene. You are trying to reinforce a sense of foreground, middle ground, and background; in other words, give an object a sense of space.

For more information, see *Chapter 5: Working with Lights* in the *Shaders, Lights & Cameras* guide.

Tutorial 17: Light Effects—How Many Animators Does It Take to Change a Lightbulb?

Your scene lacks photorealism—the lighting is the main culprit. You decide to add lens flares where appropriate, as well as volumic lights, glow, and shadow maps. You are prepared to burn the midnight oil to fix the problem, but you will soon realize how easy the task really is.

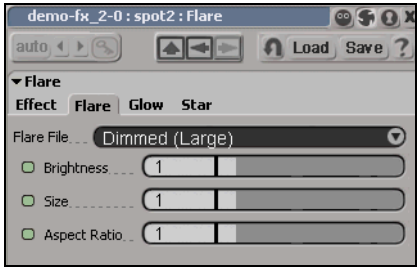


This tutorial shows you how to:

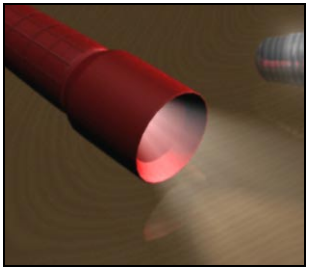
- Apply a Lens flare, Volumic effect, and Glow to a flashlight.
- Use Inclusive/Exclusive lighting to exclude the table from the light.

Overview

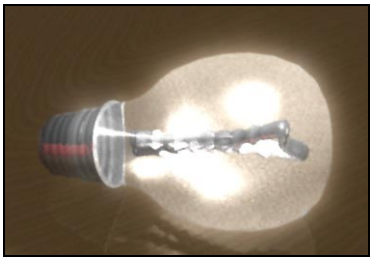
1 **Apply a Lens Flare.**



2 **Apply a Volumic effect.**



3 **Apply a Glow effect.**

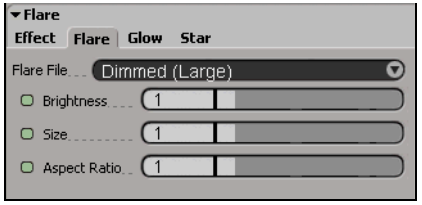


Adding Flare to a Flashlight

1. Open the **LIT_e2_FX** scene from the Tutorials project.
2. Position the camera in the **B** viewport so that the camera is looking down the "barrel" of the flashlight.

You can use the multifunction **s** key to orbit, dolly, and pan (left-, middle-, and right-click, respectively).

3. Select the light source inside the flashlight body—change the Selection filter to **Light** to make this easier.
4. Draw a render region around the flashlight.
5. In the Render toolbar, choose **Get**> **Property** > **Lens Flare**.
6. On the Flare property page, click the **Flare File** menu and choose **Dimmed** (**Large**). Adjust the settings as you wish, then close the property page:
   - **Brightness**
   - **Size**—the absolute size of the flare, which does not change with distance.
   - **Aspect Ratio**—affects the shape of the flare; wider aspect ratios result in flattened Flares.



Click on the arrowhead at the end of the Flare File drop-down menu and choose from dozens of flare types.

Adding a Volumic Effect to the Light

7. Orbit the camera so that the B viewport shows the flashlight from above. Draw a render region.
8. Make sure the spotlight inside the flashlight is still selected.
9. In the Render toolbar, choose **Property** > **Volumic**. In the property editor that opens, make sure the Volumic effect is active. Adjust the parameters as you like.

If you wish, you can widen the light cone. Note that the width of the rendered cone is limited by the width of the barrel of the flashlight.

10. Select the large lightbulb object (not the light source within the bulb).
11. From the Render toolbar, choose **Properties** > **Glow**. In the property editor, adjust the settings as you like.
12. The Glow shader is a post-rendering effect, so you won't see its effect in the rendered image until the rendering is complete.

Excluding the Table from the Light

Now exclude the table from the lighting cast by the large lightbulb.

13. Pick the light source inside the large lightbulb object
14. In an explorer, isolate the light (press **e**), expand the light's tree, and lock the explorer view.
15. Select the table object and, in another explorer (viewport) view, isolate the table.
16. Drag the name of the table from that explorer to the icon for **Associated Models** in the other explorer.
17. Expand the tree for **Associated Models**. Note that the table object has now been added.

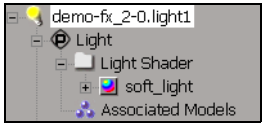
By default, any association between the light and the table is inclusive. This means that the light is illuminating only the table and no other object. Now change that:

Click the light's property icon to open the light property editor.

18. Click the light property icon in the explorer to open the light property page (or select the light and click **Modify** > **Shader** from the Render toolbar).
19. Change the **Selective Light** option to **Exclusive**.

Notice how the table is now the only object in the scene that is *not* receiving light from the bulb.

To remove the association, right-click the table object in the **Associated Models** tree and choose **Remove from Group**.



The Associated Models node contains all of the objects in a scene that you wish to be affected by a selective light.

Conclusion

In SOFTIMAGE|3D, using lens flares in a scene involved applying a shader to the camer, and then selecting the object that would be the source of the lens flare. In SOFTIMAGE|XSI, the workflow is more straightforward.

The workflow to add volumic effects is a bit different in XSI, compared to SOFTIMAGE|3D. To apply a volumic lighting effect in SOFTIMAGE|3D, you must apply a volumic shader effect to the scene, then specify which light(s) would generate the volumic effect.

Since lights can now be made child objects, lighting properties will automatically propagate down through object trees. For example, if you are working on a truck that has four headlights, and you want to apply the same lighting attributes to all four lights simultaneously, you simply select the truck in branch mode and apply the settings.

For more information, see *Chapter 8: Blurs, Flares & Other Effects* in the *Shaders, Lights & Cameras* guide.

Section 10 **Render Tree & Textures**

Tutorial 18: The Render Tree—Texturing, with a Twist of Lemon

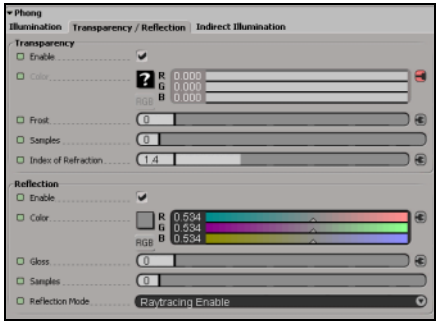
This exercise uses the render tree to mix materials, apply a texture, and create a displacement map on a single object.

This tutorial shows you how to:

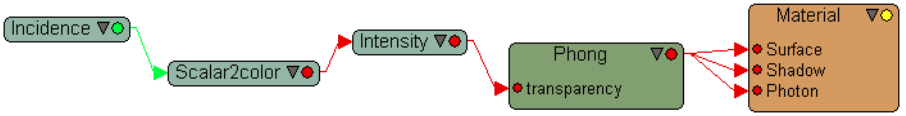
- Use the render region to interactively preview a finished-quality render of your scene or pass.
- Work with shader nodes in the render tree.
- Apply textures.

Overview

1 **Apply a material from the toolbar.**



2 **Get and connect nodes in the render tree.**



3 **Blend shaders and apply effects using the render tree.**

Applying a Surface

1. Open the **REN_e1_Bottle** scene from the Tutorials project.
2. Click and drag a render region over the bottle in the scene. The bottle looks like a gray blob for the moment, because it only has a basic Phong material applied to it.
3. Select the bottle object and open a render tree view of it in either a viewport or a floating window (press **7**).
4. Notice the basic surface—a Phong shader—applied to the bottle's Surface input. Let's get this bottle looking more like a bottle. Double-click the Phong's shader node in the render tree to open its property editor.
5. Select the **Transparency/Reflection** tab and give the Phong some transparency; for example, about 0.5 or 0.6. Then give it some refraction. A value of 1.4 is physically accurate, so why not put that one? In the render region you'll see the new values update immediately.

In XSI, the transparency is defined with colors, not with a single slider anymore. To achieve results of 0.6, for instance, do one of the following:

- Enter 0.6 for each channel of RGB.

  or

- Ctrl+click and drag over one slider until you reach 0.6.

  or

- Switch to HLS mode by clicking once on the RGB mode and set the luminance **L** factor to 0.6.

6. To get a better rendering, open the Render Region options by selecting **Render** > **Region** > **Options**. Click the **Optimization** tab and increase the raytracing values to 4, 4, 8. Close the property editor and notice how the refractions are more accurate in the render region.

Creating a Glass Effect

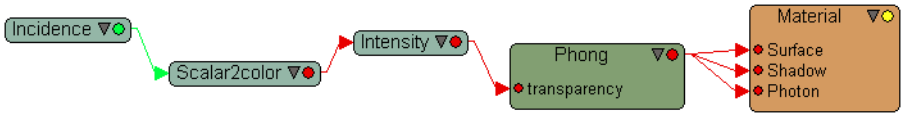
7. Back in the render tree work area, select the **Incidence** shader from the **Nodes** > **Illumination** menu.

   The Incidence shader is used to control the transparency of the bottle. The edges will be made more opaque, according to its normals and the camera's angle.

8. Also from the Nodes button, select the **Scalar to Color** shader from the Conversion menu. This shader is used to convert the Incidence shader's scalar output to color so it can be connected to a color parameter (i.e., surface, transparency, etc.).

9. Next, once again using the Nodes button, select the **Image-Processing** > **Intensity** shader. This shader will be used to control the intensity of the Incidence effect, hence the transparency of the bottle.

10. Connect the shaders by clicking on the dots in the top-right corner of the shader node and dragging an arrow to the shader you wish to connect it to. Connect the shaders as shown below:



11. Now you must set the values of the various shaders to achieve a realistic-looking glass. Open the Incidence shader's property editor and set the Incidence Exponent to 0.6.

12. Open the Intensity shader's property editor and set the Intensity Factor to a value around 1.3.

13. Notice the change in the render region. You can further enhance the glass of the bottle by giving it a blue tinge. Do this by editing the Diffuse color of the Phong shader and, perhaps, playing with the Specular Decay.

14. In the render tree work area, add a Lambert illumination shader (Nodes > Illumination). Either select it from the Nodes menu or drag and drop it from a browser window.

    The Lambert material will be used as a basis for the bottle's label. Don't connect it just yet.

15. From the Nodes drop-down menu, get a Texture > Image shader. Connect the Image shader to the Lambert shader's Diffuse and Ambient inputs.

Mixing Material Shaders

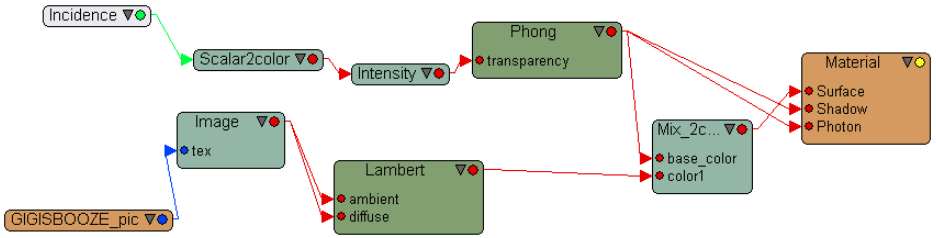
16. Next, you'll want to use both the Lambert and Phong shaders on the bottle, but you'll need to mix them first. From the **Nodes** > **Mixers** menu, select the Mix2Colors shader. Connect the Phong into the Base Color input and the Lambert into the Color 1 input.

17. Double-click the Image shader and select a new image clip to use instead of the default no_icon color bar image. Choose GIGISBOOZE. Once selected, you'll see a thumbnail of the image in the shader's property editor. In the render tree, you'll see that a new image clip has attached itself to the Image shader. The no_icon image node is left in the work area to die a lonely death.

18. Connect the Mix2Colors output to the Surface input of the Material Node.

In order for the texture to be displayed correctly, it will need the correct texture projection. Choose the texture projection that has already been defined for the bottle.

19. The render region should now display a mix of both the transparent Phong and the bottle's label, and your render tree should look like this:

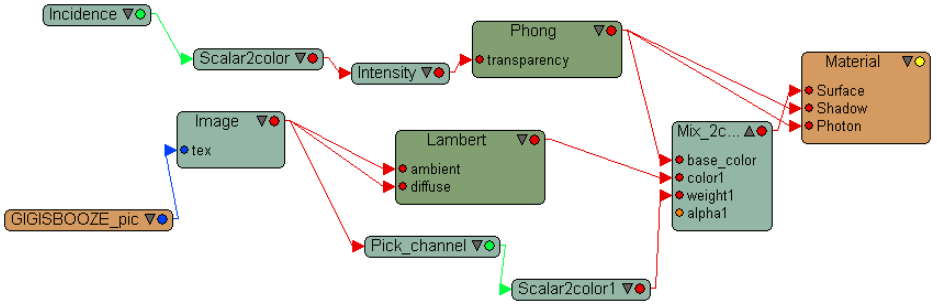


20. In order to get the proper mix happening on this bottle, you'll have to affect the mix weighting. In order to make this a perfect mix, use the alpha channel of the label image to drive the mix weighting of both materials.

Extracting the Alpha

21. In order to extract the alpha channel from the image, select the **Color Channels** > **Picker** shader. Connect the output of the Image shader to the Picker's input. You need not disconnect the Image shader from the Lambert. A shader can output to any number of other shaders.

22. Connect the output of the Picker shader to the Weight parameter of the Mix2Colors shader using another Scalar to Color conversion shader. Your render tree should resemble this:



If you wish to see what the Picker shader is extracting, simply select the shader and press the **p** key. This will enter you in Preview mode and temporarily attach the selected shader to the Surface input of the material node. Don't forget to press **p** again before continuing work on your render tree.

23. Open the Picker shader's property editor and select the Alpha in the channel parameter of the picker property page. The render region should now display the Lambert-shaded label on the Phong-shaded bottle.

24. As a final touch, try adding some Reflections and Gloss, and increase the sampling of the Phong shader.

Extra Bonus Steps!!

You can add displacement to the bottle in a few easy steps:

25. Select an Image shader and open its property editor. Select the **gigibooze_displacement** image. In the thumbnail, you'll notice that this image will add small ridges to the bottle's cap and slightly elevate the label.

26. Use the same texture projection defined for the image of the label.

27. Connect the Image shader to another Intensity shader. Open the Intensity shader's property page and lower the Intensity value.

28. Connect the Intensity shader to the Displacement input on the material node.
29. Once connected, you'll see the displacement directly in the render region. In order to smooth it out a little, make sure the bottle is selected and click **Get** > **Property** > **Geometric Approximation**.
30. Once the Geometric Approximation property editor opens, select the **Displacement** tab and edit the **Step** setting to 2—or higher, if you don't mind a longer render.
31. Also, you can increase the anti-aliasing in the render region by letting the mouse cursor rest on the small rectangle on the upper-right side of the render region's frame. Move the slider higher to define a higher anti-aliasing setting.

Conclusion

You have used the render tree to blend textures, combine materials, create displacement, and edit all of their values. The render tree lets you use any number of image clips or tool shaders to create almost any effect you desire. This tutorial concentrated on a geometric object, but the render tree can be used on cameras and lights as well.

For more information, see the following chapters in the *Shaders, Lights & Cameras* guide:

- *Chapter 3: Material & Texture Basics*
- *Chapter 4: Advanced Materials & Textures*
- *Chapter 9: The Render Tree*

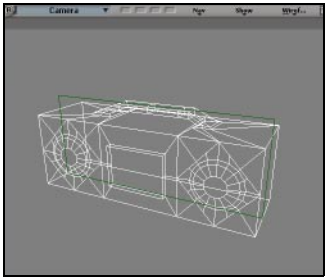
Tutorial 19: The Texture Editor—Dressing Up the Boombox

Now that you can construct effects and manipulate mapping using the render tree, you'll learn how to accurately map textures to selected polygons using the texture editor. The texture editor superimposes your object's wireframe UV coordinates onto a texture. You can then manipulate any part of your object to fit onto any part of the texture.

This tutorial shows you how to edit the UV coordinates using the texture editor.

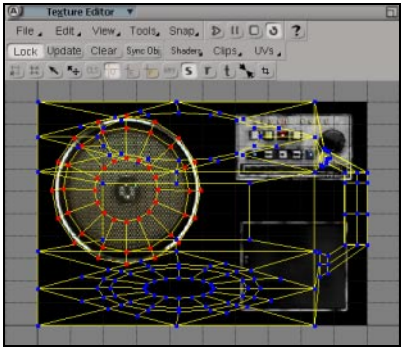
Overview

1 **Apply a texture projection to the object.**



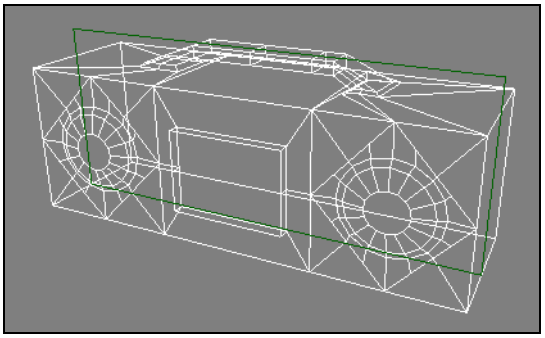
2 **Apply a texture to the object.**

3 **Edit the texture's placement on the object.**



Applying a Texture

1. Open the **REN_e2_boombox** scene from the Tutorials project. (We know— the boombox doesn't have a CD player, but retro is in.)
2. Define a texture projection for the boombox using **Get** > **Property** > **Texture Projection** > **XY** from the Render toolbar. It makes sense to use a planar projection so that most of the object (speakers and tape deck controls) are covered by the projection (in this case, an XY projection).



3. Then, apply a texture to the boombox's geometry by selecting the object and choosing **Get** > **Texture** > **Image** from the Render toolbar. Press Yes when prompted.

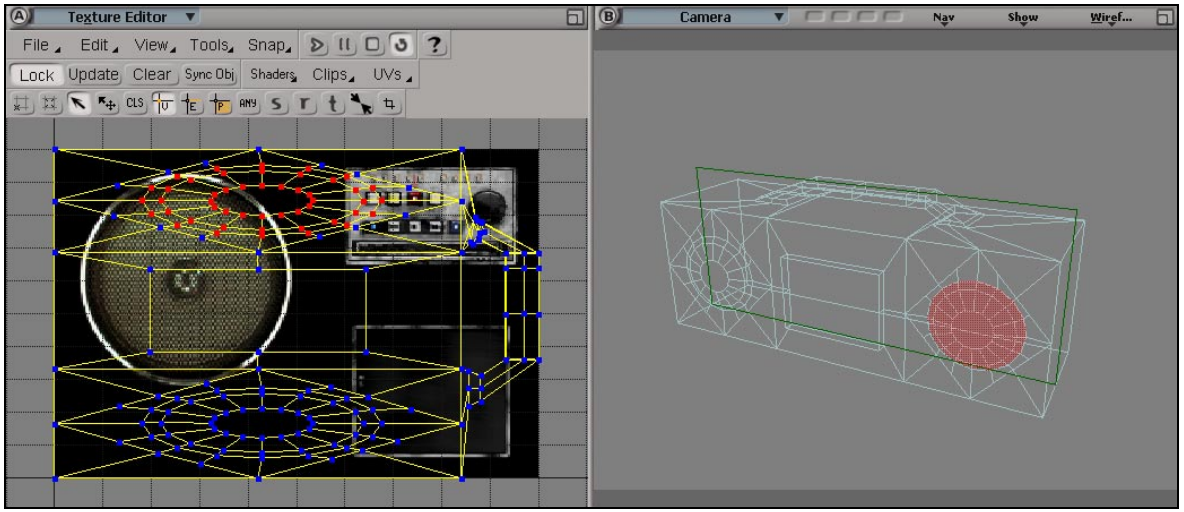
4. Once the Image shader property editor appears, choose **New from File** and select the **boombox.pic** sequence from the picture folder of the Tutorials project.

When selecting a sequence of images, the XSI browser will display the sequence as a folder icon. You can open the folder to select individual files or select the **animated_boombox** folder to use the whole sequence.

Using the Texture Editor

5. Use the Texture Projection text box to make the Image shader use the texture projection you defined in Step 2.

6. With the boombox geometry still selected, open the texture editor and press **Update**. You should see the UV texture coordinates over the texture, as shown below.



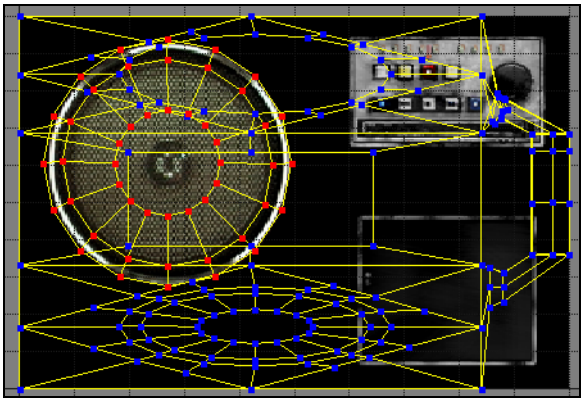
7. Select one of the speakers' polygons using the raycast selection tool (press **u**). The polygon is filled with red in the Texture Editor view.

To add or omit a polygon from the raycast select, use **Ctrl**+**u**. Otherwise, every time you press **u** and make a selection, any selected polygon(s) will be unselected.

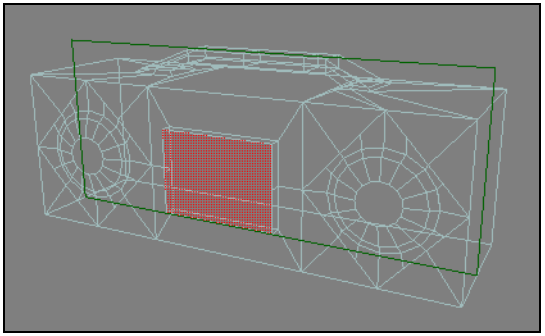
8. If you click **Update** you should see the selected polygon vertices highlighted (as shown above, left).

Moving Polygons to Fit a Texture

9. Before you translate your selection in the texture editor, make sure that you are in Polygon Selection mode (the button with a small **p**). Otherwise, when you try to translate a selection, it will remain attached to its neighboring vertices.
10. Press **v** (translation) and move the polygon coordinates in position over the speaker texture. Then use **x** (scale) to scale the polygon selection to the size as the speaker part of the texture map.

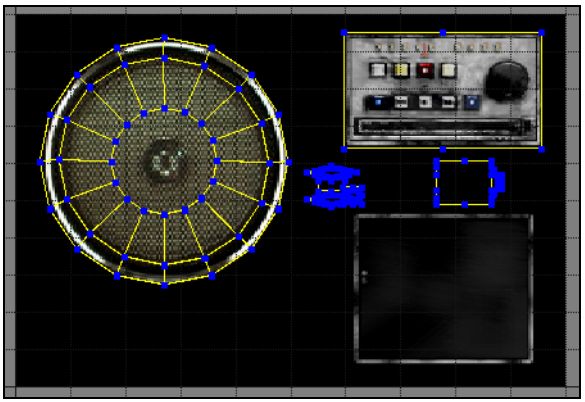


11. Repeat the same steps for the other speaker.
12. Using the raycast tool again, select the middle rectangular polygon (below).



13. Click **Update** and translate and scale the polygon over the play controls of the texture.

14. Select every other polygon not edited so far and **Collapse** the selection over a black area of the texture. That way, the remaining areas won't have extraneous pieces of the texture mapped to it, and they'll all be mapped to the same pixel. Once finished, the texture editor work area should look like this:



15. And that's it! Now keep the volume down... people are trying to work!

Conclusion

Using the texture editor, you can precisely place any section of a texture to any part of your object. This means you can create a map of several different textures using an image-editing software and apply them all to your object, regardless of their placement in the texture image.

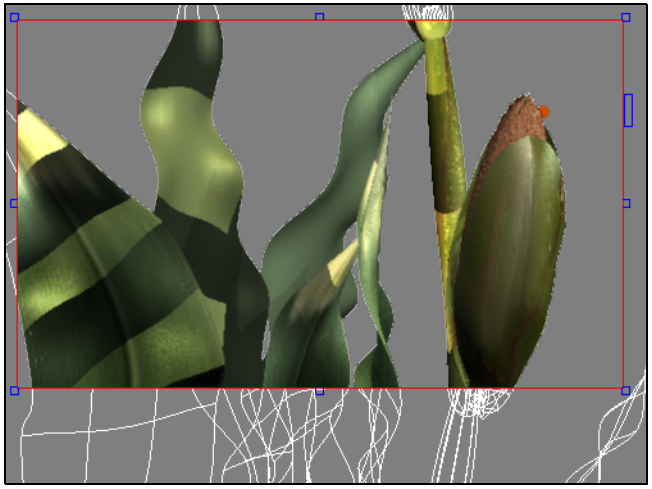
For more information, see the following chapters in the *Shaders, Lights & Cameras* guide:

- *Chapter 4: Advanced Materials & Textures*
- *Chapter 9: The Render Tree*

Section 11 Rendering

Tutorial 20: Creating Render Passes

Render passes allow you to divide your final render in separate layer in order to give flexibility to adjust each of these layer at the compositing stage. In this tutorial, you will create a highlight pass, a depth pass, and a shadow pass for rendering. These passes automatically become a part of the sequence you're working on and let you edit specific areas of your scene quickly and easily. Once defined, they will not have to be defined again.

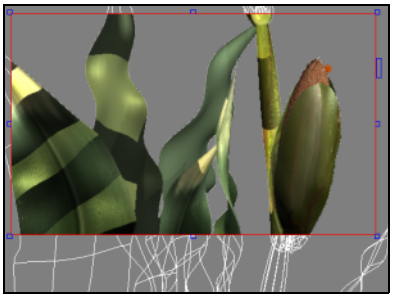


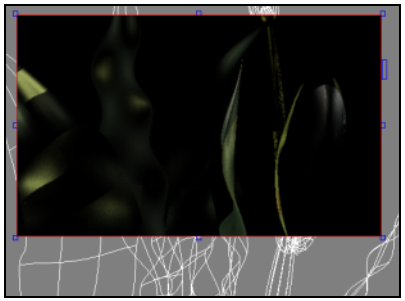
This tutorial shows you how to:

- Create a highlight pass.
- Create a depth pass.
- Create a shadow pass.
- Render all passes.

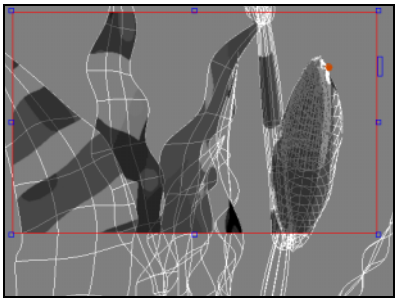
Overview

1 **Create a highlight pass of the scene's objects.**

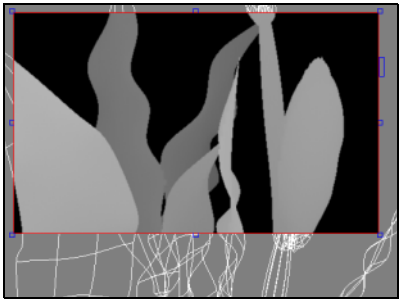




2 **Create a shadow pass.**



3 **Create a depth pass.**

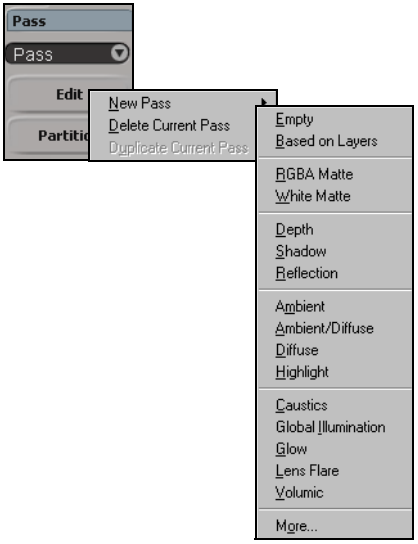


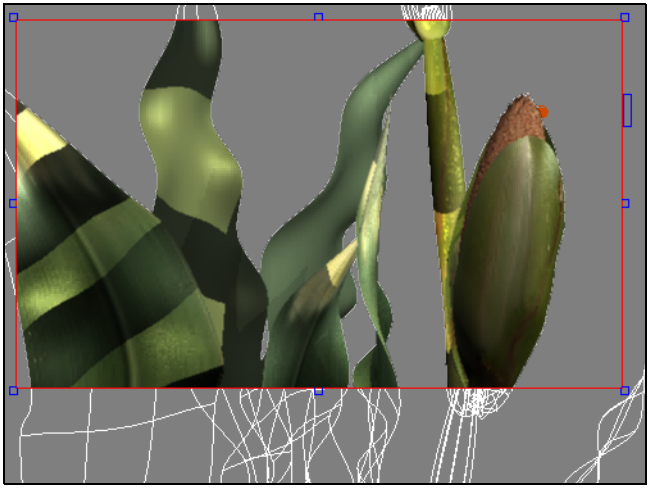
Creating a Highlight Pass

As you create new passes, the last pass defined becomes the current pass and you can immediately see what the rendered scene looks like in the render region.

The highlight pass uses all of a scene's objects and isolates their highlights. Also called a specular pass, this pass can help you tweak the brighter areas of a scene or even add shaders to the highlights while retaining the original materials of the model or object.

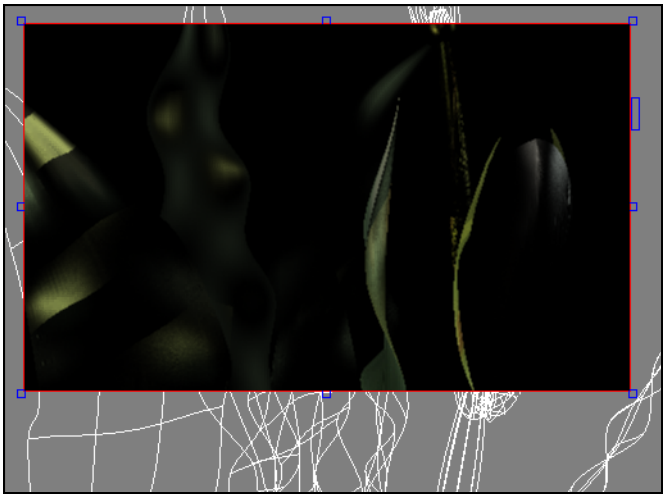
1. Open the **REN_e1_corn** scene from the tutorial database.





Default pass with render region set to RGB + Alpha.

2. Choose **Pass** > **Edit** > **New Pass** > **Highlights** from the Render toolbar to create the highlight pass for every object in the scene. The highlight pass is set as the current pass and the highlight pass property editor opens.
3. If you wish, enter an new **Name** for the pass in the property editor.
4. Click the Render Options tab to open the render options property page for the highlight pass. Set the options as desired and close the editor.
5. Draw a render region (press **q** and drag) in the camera viewport.
6. Set the region to show RGB only by choosing **Render** > **Region** > **Show RGB** from the Render toolbar.



Highlight pass with render region set to RGB.

Everything you create a new pass it becomes the current pass. You can specify the current pass by choosing it from the **Pass** pull-down menu on the Render toolbar.

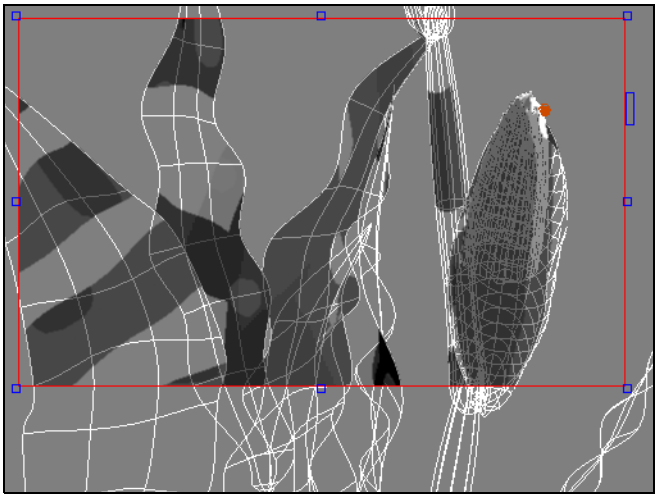
Creating a Shadow Pass

The shadow pass lets you isolate a scene or object's shadow and edit it in a way that can compliment the scene. For example, you could apply a transparency or blur to the shadow to make it more realistic instead of a sharp, opaque shadow.

7. Choose **Pass** > **Edit** > **New Pass** > **Shadow** from the Render toolbar to create the shadow pass for every object in the scene. The shadow pass is set as the current pass and the shadow pass property editor opens.
8. If you wish, you can enter an new **Name** for the pass in the property editor.
9. Click the Render Options tab to open the render options property page for the shadow pass. Set the options as desired and close the editor.

The render region updates with the new shadow pass.

10. Choose **Render** > **Region** > **Show RGB + Alpha** in the Render toolbar to see the altered alpha channel in the render region.



Shadow pass shown with render region set to RGB + Alpha

Creating a Depth Pass

A depth pass is used to create a depth effect for objects that are far from the camera. When composited with a rendered scene, objects that are far in the background have a grayish layer applied to them to reduce detail and simulate a type of depth fading.

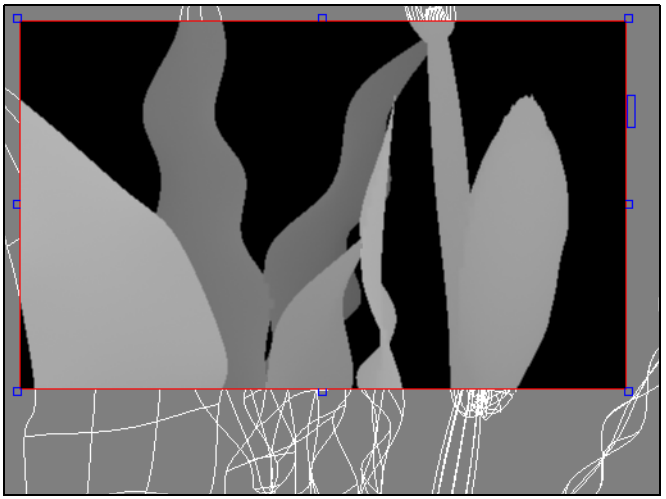
11. Choose **Pass** > **Edit** > **New Pass** > **Depth** from the Render toolbar to create a depth pass for all of the scene's objects. The depth pass is set as the current pass and the depth pass property editor opens.
12. If you wish, you can enter an new **Name** for the pass in the property editor.
13. Click the Render Options tab to open the Render Options property page for the depth pass. Set the options as desired and close the editor.

The render region updates with the new depth pass but renders black because the distance between the camera and objects is too long.

14. From the render toolbar, select **Pass** > **Edit** > **Current Pass** to open the pass property editor for the current pass.
15. Click the Volume shaders tab to expose the volume shader stack of the current (shadow) pass. You will see the **Constant Density** shader applied to the scene through its stack.
16. Select the name of the shader, and click the inspect button to open the shaders property editor.

17. Turn off **1:10** and turn on **1:100.** This sets the **Density** scale to one to a hundred. The scaling determines the depth used (in SOFTIMAGE units) to display the defined density. Use 1:100 to create a detailed depth pass of a scene that extends far from the camera, and 1:1 for a scene that is relatively flat.

18. From the Render toolbar, choose **Render** > **Region** > **Show RGB** for a clear view of what the depth pass will render.



Depth pass with render region set to RGB. Notice how objects closer to the camera are light and objects further from the camera are darker.

Rendering All Passes

Once you have edited all your passes and set the render options, you can render all the passes as separate files so they are ready for compositing.

19. Choose **Render** > **Render** > **All Passes.** Watch the passes being rendered one after the other without the need to re-tessellate each of them, but only at frame refresh.

Conclusion

You have created three of the most basic render passes a scene will use. In addition, you can create and edit a matte pass, a flare pass, a diffuse and ambient pass, and so on. Each pass gives you the ability to change how certain areas, effects, or properties will be rendered and to what extent they will contribute to the final look of your scene.

For more information, see the *Rendering* guide.

Section 12 Scripting

Tutorial 21: Scripting—Repeating Commands

Command box

Anything that you can do interactively can also be done by writing scripts. Simple scripts are a sequence of commands. More advanced scripts use a host language such as Visual Basic to combine SOFTIMAGE|XSI commands.

Every time you perform any action, the equivalent scripting command for the most recent action appears in the command box at the bottom-left of the window. There you can examine the syntax for any action you perform. You can later retrieve these commands and use them to create simple, one-command scripts that automate repetitive tasks, or use them as building blocks in your own, more elaborate scripts.

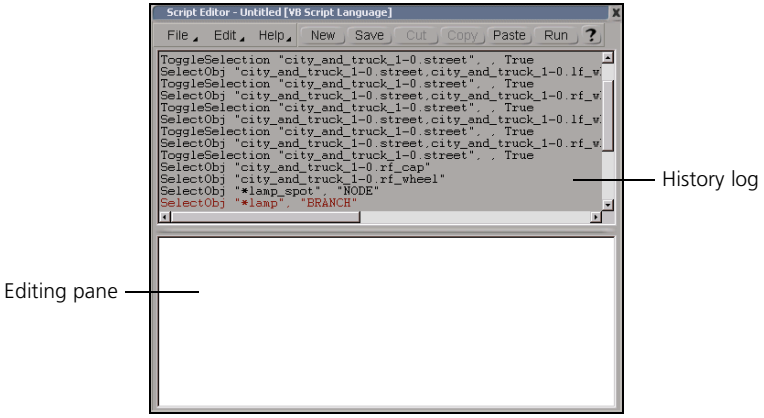
This tutorial shows you how to:

- Use the script editor's command box to repeat commands from the history.
- Create a button for a custom command.

Opens the script editor.

Using the Command History

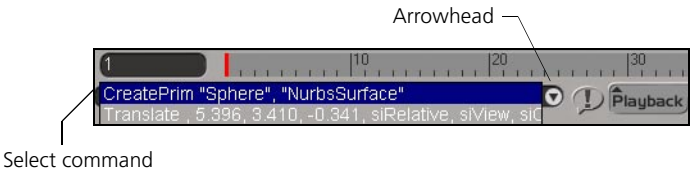
1. Open the script editor by clicking on the script editor icon—the exclamation point (**!**)—beside the Playback controls below the timeline.



2. It will be easier to see what's going on if you start with a clean slate. If you've been working in SOFTIMAGE|XSI and already have commands in the history log, clear it by choosing **Edit** > **Clear History Log** from the script editor command bar.

3. From any toolbar, choose **Get** > **Primitive** > **Surface** > **Sphere** to create a sphere. Notice how the command is instantly logged in the history pane of the script editor.
4. Translate the sphere in any direction.
5. If you just want to quickly repeat a single command, you don't need to use the script editor window. From the command box (at the extreme left below the timeline), click the arrowhead to show the list of recent commands and select the following line:

```
CreatePrim "Sphere", "NurbsSurface"
```



6. Press Enter. Another primitive sphere is created.

Creating Custom Commands

If you want to repeat more than a single command, you need to create a script. In this section you'll create a custom command that runs a very basic script—simply repeating the commands to get and translate a sphere.

You create custom commands by creating buttons on a custom toolbar either by dragging lines from the script editor or by dragging files from a browser.

7. Choose **View** > **Custom Toolbars** > **New Toolbar** from the main-menu bar and give the new toolbar any name you want. Click OK. Leave the toolbar open.
8. Select the first two lines in the history log:

```
CreatePrim "Sphere", "NurbsSurface"
Translate, xxxx, xxxx, xxxx, xxxx, xxxx, xxxx, xxxx
```

   These two commands create and translate a NURBS sphere.

9. Drag and drop these two lines onto the toolbar. The Add Script Command dialog box opens.
10. For the **Command/Button Name**, type **Quick Sphere**. This is the name that will appear on the button in the toolbar.
11. For the **Script Command Name**, type **Quicksphere**. This is the name that will be logged in the command history. You can also call your script from other scripts using this name.

12. Click OK to close the Add Script Command dialog box. The new button is added to the toolbar.
13. Start a new scene, either by choosing **File** > **New Scene** or by pressing Ctrl+n. You do not need to save the old scene.
14. Click on the **Quick Sphere** button in the toolbar. Your custom command creates and translates a sphere. Close the toolbar.
15. Delete the sphere.
16. In the command box, type `Quicksphere` and press Enter to invoke your saved script. This is another method of calling and executing a frequently used script.

Conclusion

This tutorial showed the simplest form of scripting: repeating commands. By itself, this can be a great timesaver. However, it is also possible to do much more with scripting. The next tutorials explore some of these possibilities.

For more information, see *Chapter 7: Commands & Scripts* of the *Fundamentals* guide.

Tutorial 22: Scripts for Automating Tasks

You can use scripts to automate virtually anything. In this example, you'll write one to toggle the viewport settings to speed up screen redrawing.

This tutorial goes one step beyond the simple repetition of commands that was presented in the previous tutorial.

This tutorial shows you how to:

- Modify the logged commands in the history so that they work with any selected objects.
- Use commands from the VBScript language to create a "for" loop that cycles through the selected elements and performs actions on them.

Loading a Fragment of VBScript Code

1. Open any scene, or start with a new scene and get several primitives.
2. Open the script editor.
3. Open the file **Opt_displ_templ.vbs** from the **Scripts** folder of the **Tutorial_Project**. This loads a template containing the following lines:

```
Set myselList = GetValue ("SelectionList")
For each elem in myssellist
next
```

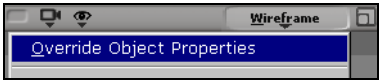
This fragment of code is a pattern that you'll use over and over in scripting.

- **SelectionList** is a special keyword that always refers to the collection of all the items that are currently selected. The first statement puts this collection in a user-defined variable called **mysellist**.
- The second statement is the beginning of a **for** loop. Any commands between this line and the statement **next** are performed once for each item in **mysellist**. Within this block of code, you can refer to the current item by the user-defined variable **elem**.
- Right now, there are no commands in the **for** loop yet. You'll complete this script by inserting commands in this loop.

Optimizing the Selected Object for Interaction

To optimize interactions in the viewports, you'll modify the selected object so that it is shaded when static and displayed as bounding boxes when you orbit, pan, etc.

4. Select any object in the scene.
5. On any toolbar, choose **Get** > **Property** > **Display**. Make the following settings:
   - Set all the **Static** options (**Selected**; **Unselected, Near**; **Unselected, Far**) set to **Shaded**.
   - Set all the **Interaction** options to **Bounding Box**.



6. In a viewport, turn off **Override Object Properties** in the display mode menu. When on, this option overrides the display properties of any individual objects: it must be off for your changes to have a visible effect.
7. Orbit in the viewport. Note that the affected objects turn into bounding boxes while the view changes, whereas they stay in shaded mode when the view is static.

Writing the Script

8. In the script editor, notice that the actions you performed have been logged in the history pane.
9. Copy and paste (Ctrl+c and Ctrl+v) the **AddProp** and **SetValue** statements from the history pane into the editing pane. Insert them between the **For each** and **Next** statements.

   Next, you'll modify these lines so that they work with whatever is currently selected, instead of the specific objects that were selected when these lines were logged.
10. Consult the online HTML command reference to get help on the **AddProp** command: move the text cursor to anywhere inside the AddProp keyword, then press F1 or click the **?** button. A browser opens, showing a reference page for the AddProp command.

    Read the page and notice that the default value of the **InputObj** parameter is the current selection. This means that if you leave this line as it is without specifying a value for InputObj, a local display property will be added to every item in the selection list every time the loop is executed. You'll fix this in the next step.
11. Add the following string to the end of the **AddProp** statement:

    ```
    , elem
    ```

    It should now look like this:

    ```
    AddProp "Display Property", elem
    ```

This specifies that the display property should be added to the current **elem** in the loop. Since the loop is executed only once for each **elem** in **mysellist**, the display property will be added exactly once to each selected item.

12. Remove the object name from each **SetValue** statement. Make sure you don't delete the double quote immediately before the object name nor the dot after it. For example:

```
SetValue ".display.staticsel", 1.000
```

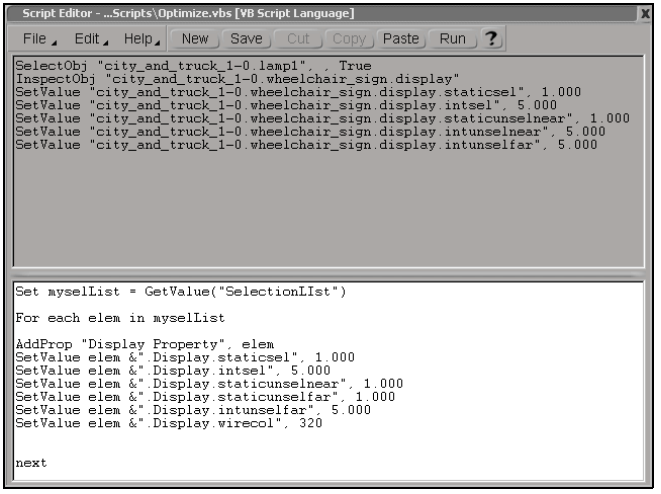
13. In each **SetValue** statement, insert the following string immediately before the first double quote character:

```
elem&
```

For example, the first **SetValue** statement should look like:

```
SetValue elem&".display.staticsel", 1.000
```

The ampersand **&** is the symbol for string concatenation in VBScript. This concatenates the current value of **elem** with the literal string inside the double quotes. Each time the loop is executed, the value of **elem** is the name of a different object in the selection list. Effectively, you have replaced the specific object name you deleted in step 12 with the names of whatever is currently selected.



14. Select some items in the viewport then click the **Run** button to run the script. The properties for all the selected objects are changed as though you had done it manually for each item.

If anything is selected, only the highlighted text will be parsed and executed when you click the **Run** button or press F5. This is useful for trying out new commands or stepping through the execution of a script step by step to track down errors.

15. If desired, add this script to a toolbar as you did in the previous tutorial.

Conclusion

Scripting is a powerful and flexible tool for doing almost anything. Almost all features in SOFTIMAGE|XSI are available through scripting. You can copy and paste commands from the history to automate repetitive tasks, or write your own scripts from scratch. And you are not limited to a specific scripting language—you can use any scripting language that supports ActiveX, including VBScript, JScript, PerlScript, and Python ActiveX Scripting.

For more information, see *Chapter 7: Commands & Scripts* in the *Fundamentals* guide.

Tutorial 23: Building Custom Tools

Created by **Matt Lind**
Animator, Technical Director, and Softimage Certified Instructor
speye_21@hotmail.com

The power of scripting does not stop at simply repeating commands you have already logged using the interface. You can also write custom tools that do new things. Your tools can even prompt for input and invoke picking sessions.

This tutorial shows you how to:

- Choose a task for a new tool.
- Plan how the tool will work.
- Validate the plan.
- Write the code. The script will use variables, subroutines, and arguments to accomplish its task.

This tutorial assumes you have basic knowledge of scripting concepts.

Choosing a Task to Script

As you work in SOFTIMAGE|XSI, you'll realize there are tasks you perform repetitively that can be consolidated into a single operation. This could be anything from creating a character's skeleton or loading primitives with specific settings, to saving files in a particular location with a specified naming convention on your hard disk drive. When designing a tool, the scope of its responsibilities must be determined: decide what the tool does and does not do.

For this tutorial, you'll write a tool that realigns an object to match the scale, orientation, and position of another object chosen from the scene. Such a tool would be useful when performing modeling operations, setting keyframes, adjusting lights and cameras, and replacing child objects in hierarchies.

Building tools for production is an art form in itself. Many things need to be considered:

- What is the goal of the tool?
- What is the scope of its responsibilities?
- What information does the tool need to perform its duties?
- How do users interact with the tool?
- Can the tool work in a general environment?
- Can other tools perform the same duties with minimal overhead?

Most important is whether the benefits reaped from a tool outweigh the time and resources needed to create it. The next few sections explore these design decisions as they relate to the match tool.

Designing the Tool

Once you have decided what your tool is going to do, you must decide how it is going to do it. In this case, the tool must set the transformations of the selected objects to match another object. There are several possible ways to achieve this:

- Manually adjust the SRT values.
- Apply then relax a constraint.
- Apply an expression.
- Set relative values with linked parameters.
- Set a keyframe, adjust the fcurve, then remove animation.
- Make a parent-child relationship, then reset the child's local transformation values to match its parent's transformations.

There are many options, but which is the best? If the tool is to be used for modeling, then setting temporary keyframes or using linked parameters probably won't be very helpful, especially if the object already has animation. Expressions may override other constraints, keyframes, and operators— that could ruin the scene. Applying then relaxing a constraint is a quick easy option often used in many 3D applications, but if the object already has a constraint, it may be difficult to remove it properly without destroying others already in place. The selected object could be made a child of the target, then have its SRT values zeroed out, forcing the object to cling to its parent—clever, but what if the selected object is already part of a hierarchy? That leaves the first option, adjusting SRT values—it allows modeling operations to be performed and objects to be adjusted without destroying other relations and elements in the scene, and keyframes can be set when necessary.

Now that you have a good understanding of the options, you must decide how to define which object(s) to affect. There are a few options:

- Apply to all selected objects.
- Let user interactively pick objects one by one.
- Type in the names of the objects to be modified.

The tool is meant to be a quick operation; therefore asking users to interactively navigate through the scene and click on several objects may be a bit tedious. Also, if objects are very close together, it may be difficult to isolate the desired target. Likewise, typing the names of objects into a browser can take just as long. That leaves the first option—modify the selected objects. For those who want to type, you can still use the selection box on the MCP to select the desired objects first.

Next, you must decide how to define the target object to which the selection will be aligned. The selection list can't be used, because it contains the objects to be affected. The remaining options are to pick a target interactively with the mouse, or to type the target's name into a browser. Clearly, picking is more intuitive and interactive for working quickly.

Testing the Theory

At this point, you should have a rough idea of the steps your script must perform to achieve the desired result. Now the methodology must be tested to see if it works. This can be done quickly by setting up some objects and manually performing the tasks within the interface. In this case, select an object and adjust its scale, orientation, and translation values to match that of a target object.

1. Get a primitive cube, sphere, and null.
2. Select the null and click the **Parent** button on the Constraint panel (main command area). Pick the cube to make it a child of the null. Click the right mouse button to exit parent mode.
3. Select the cube and translate it away from the origin to a random location. This will be the target.

4. Look up the position coordinates of the cube on the Transform panel. Take note of the values.
5. Select the sphere and enter the cube's position coordinates into the Translate boxes on the Transform panel so that its position matches the cube.

Notice the sphere now resides in the same position as the cube. The theory works so far, but it must be tested under more complicated conditions.

6. Select the null in branch mode with the middle mouse button.
7. Rotate the null around its Y axis by 30 degrees.
8. Select the cube and again take note of its position coordinates.
9. Select the sphere and copy the cube's position coordinates into the Transform panel.

What's this? The sphere's position does not match the cube! This is because the cube's coordinates are shown in local space relative to its parent, the null. Also, the coordinates listed in the Transform panel are relative to the manipulation mode that's currently set. In this case, the default View mode displays local coordinates. To get around this problem you need to retrieve the global coordinates of the cube, which represent the object's position relative to the world's origin.

10. Select the cube and click the **Global** button on the Transform panel. Jot down the cube's coordinates.
11. Select the sphere, then copy the cube's position into the Transform panel.

That's more like it! This is an example of how various settings can affect how data is perceived and processed. When building tools, such factors must be taken into account to obtain proper results. Many mistakes made during development occur because of such errors.

Before writing the code, examine one more situation that can affect how the tool works.

12. Select the sphere.

13. Constrain the sphere to the null by choosing **Constraint** > **Position** then clicking on the null.

14. As before, look up the cube's position and copy the values to the sphere.

Notice that the sphere cannot be moved because it is constrained to the null. However, if you activate compensation mode using the **Comp** button on the Constraint panel, the sphere can be moved just as before. Remember to turn off compensation mode, when finished, to keep the constraint with the new offset.

For verification, repeat the alignment tests for rotation and scaling. Also, select the null hierarchy and try to match its coordinates to the sphere in both branch and node selection modes. Take note of the different results.

Writing the Code

By now you should be convinced that the method for aligning objects is valid. It's time to start writing the script.

Comments

One of the single-most important aspects of programming is to document what you are doing. It lets anybody read your code and understand what you are attempting to do (including yourself when you've had to put the project on the shelf for an extended period of time and need to review where you left off).

Code documentation is usually achieved with comments—lines of text directly above or to the side of a command explaining the intent and purpose at that point in the code. Without comments, scripts can be very difficult to follow, especially when they grow over several hundred lines long.

15. Open the script editor and type the heading of your script using comments as shown below:

```
'*******************************************
'
' Match v1.0 - My first script!
'
'  This tool realigns selected scene objects
'  to match a picked target.
'
'*******************************************
```

In VBScript, anything that comes after the first apostrophe (') on a line is considered a comment. You can comment out an entire line as was done here, or you can comment out the last portion of a line to provide information about a specific command you are using.

Subroutines

Up to this point, the focus has been on the design of the tool and how it should work. Now you must finalize the details and make it happen.

Just as you addressed how to pick objects and manipulate them manually, we must take the same steps to create the functions that perform the operations. The general rule of thumb is to keep your code modular; that is, divide your code so that each routine or procedure performs a specific task. That way, code is minimized and can be reused by other functions or scripts. For this tool, you need modules to do the following:

- Get selected objects from the scene.
- Pick a target object interactively.
- Realign selected objects to match the picked target.
- Control overall program flow.

The first thing that is taught when programming with subprocedures is the use of the **main** routine. Its responsibility is to control the flow of the program—the glue that binds everything together.

16. Define a call to the main function by entering the following line:

```
Match_main        'Main Subprocedure call
```

    This subprocedure call is within the global code of your script; that is, it is not within any other subprocedure itself. All this line does is call and execute the subprocedure called **Match_main**. However, at this point, you haven't defined any procedure called **Match_main** yet. That's the next step.

17. Define the skeleton of your main procedure by entering the following lines:

```
sub Match_main    'Main Subprocedure calling other functions

end sub
```

    The word "main" has been prefixed with the name of the tool to prevent confusion with commands native to XSI or other procedures that may have a main routine (it's not mandatory, but it is good practice). The **sub** statement defines the start of the procedure, and the **end sub** statement defines the end. Only code between these statements (often referred to as a "block" of code) is executed when the subprocedure is called. You'll add this code in later steps.

Variables

The program structure has been defined, but now you need a means of storing and retrieving data for manipulation. You'll define variables to store data and write procedures to retrieve and manipulate them. Variables are areas of memory reserved for data, and can vary in size.

To realign the selected objects, you need to find out which objects are selected. SOFTIMAGE|XSI keeps track of the currently selected objects in the **SelectionList**. To retrieve its contents, use the **GetValue** command, which retrieves the value of a specified parameter in the project.

18. Retrieve the list of selected elements from the scene and store it in a variable, then print the variable's contents to screen to verify results.

```
Match_main        'Main Subprocedure call

sub Match_main    'Main Subprocedure calling other functions

   'Define a variable to store list of selected elements
   dim selected

   'Retrieve the selected elements from the scene
   set selected = GetValue( "SelectionList" )

   'Display contents to the screen
   LogMessage "Selected items: " & selected, siInfo

end sub
```

The above example illustrates another point. Variables should be defined before they are used. In VBScript, this is accomplished with the dim statement. This is done for several reasons:

- Code clarity
- Faster execution
- Fewer potential bugs

In programming languages such as C and C++, variables must be defined before they are used. With looser untyped languages such as VBscript and JScript, variables can be defined anywhere or upon its first use. However, to prevent unwanted side effects, it's best to define variables before they are used. In VBScript, you can also use the following statement in the global code at the beginning of your script to force variables to be defined before they are used:

```
Option Explicit
```

The **siInfo** parameter of **LogMessage** command specifies the type of message displayed. In this case, you're displaying information as opposed to a warning (**siWarning**) or error message (**siError**). The **LogMessage** command could also be written with the **GetValue** command because **GetValue** returns data in the form of a string (text):

```
LogMessage GetValue( "SelectionList" ), siInfo
```

19. Select various objects in the scene then run the script. Watch how the results change with each selection in the history log. Try running the script without anything selected.

At this point a question comes to mind—how do you know whether the selected elements are objects or whether anything is even selected? This is handled with the **SIFilter** command. The **SIFilter** command is a selection filter that allows you to retrieve elements that fit or do not fit specific criteria. Usually the criteria are based on object type.

20. Type **SIFilter** on a new line in the script editor, place the mouse cursor within the word, then press F1. The help page for the command opens.

    As you can see from the documentation, this command can filter out unwanted elements. Since you only want 3D objects, filter out non-object types.

21. Replace the **GetValue**() command with **SIFilter**() and test to see there is a valid selection. Click Run to test your script.

```
Match_main        'Main Sub Procedure call
sub Match_main    'Main Sub Procedure calling other functions

   'Define a variable to store list of selected elements
   dim selected

   'Retrieve the selected objects from the scene
   set selected = SIFilter( null, "sceneobject", true, siQuickSearch )

   'Is the selection valid?
   if typename(selected) = "Nothing" then
      selected="Nothing"
      LogMessage "Invalid selection : ", siError
      exit sub
   else
      LogMessage "Selected item(s) : " & selected, siInfo
   end if
end sub
```

Notice that the variable **selected**, the returned result from the **SIFilter** command, is tested before using it. This is to prevent an error, or worse, a crash due to improper usage. It's also more efficient because you can exit the subprocedure if your criteria has not been met, and thus not do any more work than necessary.

Pick sessions

The next task is to determine the target of the realign operation. As mentioned earlier, you'll use an interactive pick session. This is where the user will be required to navigate within the scene and click on the target. To keep your code modular, create a function just for this purpose. Make sure to define the function outside of your main routine.

22. Start a pick session and return the picked object. Use the **PickElement** command to enter the pick session and retrieve the values. Return an empty string if no object is picked or if the session is cancelled.

```
'*****************************************************
' Match_PickTarget()
' - Prompts user to pick an object
' - returns picked object, or empty string if cancelled.
'*****************************************************

function Match_PickTarget()

   dim picktarget_button, picktarget_target

   ' You can't display messages inside of PickElement(), so do it now and hope there isn't a refresh call
   LogMessage "Pick target for selected object(s).", siInfo

   ' Launch a pick session
   PickElement "OBJECT", "Pick target", "Pick target", picktarget_target, picktarget_button

   ' If the user canceled, return an empty string and exit
   if (picktarget_button = 0) then
      Match_PickTarget = ""

   ' Otherwise return the name of the picked element
   else
      Match_PickTarget = picktarget_target
   end if

end function
```

The **"OBJECT"** filter is used in the **PickElement** command to make sure only objects can be selected. If the user clicks on another element type, it will be ignored until a proper object is picked. The **PickElement** command returns two values: the variable **picktarget_target** stores the picked object and the variable **picktarget_button** stores the mouse button pressed (0 for right mouse button or Esc, 1 for left mouse button, 2 for middle mouse button). The **"Pick target"** strings are messages that appear in the mouse line to assist users, as this is the only feedback they'll receive while the pick session is active.

Notice that the two variables are prefixed with the name of the function. This is not necessary, but is very useful for ensuring the clarity of your code—it is obvious at a glance what the scope of a variable is. It also avoids any potential name conflicts with global variables; global variables are defined in global code and can be used by any procedure.

Once the pick session is finished, the mouse button is tested to see whether the user canceled or picked something. If the user canceled, then the function returns an empty string. If the left or middle mouse button was used, then you know an object was picked so the function returns its name.

Now that you have written the **Match_PickTarget** function, you need to call it from your main function:

```
Match_main          'Main Sub Procedure call

sub Match_main      'Main Sub Procedure calling other functions

   'Define a variable to store list of selected elements
   dim selected, target

   'Retrieve the selected objects from the scene
   set selected = SIFilter( null, "sceneobject", true, siQuickSearch )

   'Is the selection valid?
   if typename(selected) = "Nothing" then
      selected="Nothing"
      LogMessage "Invalid selection : ", siError
      exit sub
   else
      LogMessage "Selected item(s) : " & selected, siInfo
   end if

   ' Prompt to pick a target object
   target = Match_PickTarget

   'Did the user pick anything?
   if (target = "") then
```

```
        LogMessage "Match aborted.", siInfo
        exit sub
    end if

end sub
```

Notice also that the value returned by the **Match_PickTarget** function is tested before it is used. If it is empty, then the user must have canceled the pick session so a message is displayed and the main routine exits.

Arguments

Once it has been determined that a target has been picked, it is time to perform the realign operation. This will be done with a procedure, and requires *arguments* (also called *parameters*) to indicate which objects are to be matched to the target. Arguments are used to pass data from one routine to another, and isolate data so they aren't used out of context during program execution, for example, if one argument has the same name as another within the script.

23. Create a procedure to realign the selected objects according to the coordinates of the picked target. Again, make sure to define this procedure outside of any other procedure.

```
'*************************************************
' Match_Position
'
' - Aligns the selection to the target
'*************************************************

sub Match_Position( position_selected, position_target )

    dim vector

    set vector = CreateObject("Sumatra\Scripting\Math\SIVector3")

    vector.x = GetValue(position_target & ".kine.global.pos.posx")
    vector.y = GetValue(position_target & ".kine.global.pos.posy")
    vector.z = GetValue(position_target & ".kine.global.pos.posz")

    Translate position_selected, vector.x, vector.y, vector.z, siAbsolute,
siGlobal, siObj, siXYZ
end sub
```

The process of aligning the various objects is quite simple. The **Match_Position** function simply extracts the global coordinates of the target object then applies them to the selected objects as transformations in global space—the same as you did earlier with steps 1 through 14.

The **Match_Position** function illustrates another new concept: data objects. An object is a collection of various data types (variables) that are attached as properties. Properties are used to simplify and organize data storage, instead of defining a collection of variables individually. This allows many values to be transported across functions with a single assignment operation rather than copying them one by one. In this example, the **SIVector3** object stores the coordinates of the target object in its three properties **x**, **y**, and **z**. To access the properties of the object, use the **dot** notation as shown on the subsequent lines with the **GetValue** command.

Now that **Match_Position** is written, call it with the proper arguments from your main routine:

```
Match_main        'Main Sub Procedure call

sub Match_main    'Main Sub Procedure calling other functions

   'Define a variable to store list of selected elements
   dim selected, target

   'Retrieve the selected objects from the scene
   set selected = SIFilter( null, "sceneobject", true, siQuickSearch )

   'Is the selection valid?
   if typename(selected) = "Nothing" then
      selected="Nothing"
      LogMessage "Invalid selection : ", siError
      exit sub
   else
      LogMessage "Selected item(s) : " & selected, siInfo
   end if

   ' Prompt to pick a target object
   target = Match_PickTarget
   'Did the user pick anything?
   if (target = "") then
      LogMessage "Match aborted.", siInfo
      exit sub
   end if

   Match_Position selected, target

end sub
```

Further Work

To extend the capabilities of your script, write procedures to match the orientation and scale as well. To build realign routines for the scale and orientation parameters, simply substitute the **pos** parameters with **ori.euler.rot** (orientation) or **scl** (scale). Also substitute the **Translate** command with **Rotate** or **Scale**.

Conclusion

Scripting is a very powerful and often overlooked tool. Although you might think you need to know intricate math equations and formulas, this lesson shows that useful tools can be written without any math whatsoever.

Scripting is a method of telling SOFTIMAGE|XSI exactly what you want it to do. As you write more tools to simplify your workflow, you can focus on the higher level priorities of making your projects more polished and complete.

For more information, see *Chapter 7: Commands & Scripts* in the *Fundamentals* guide.

Index