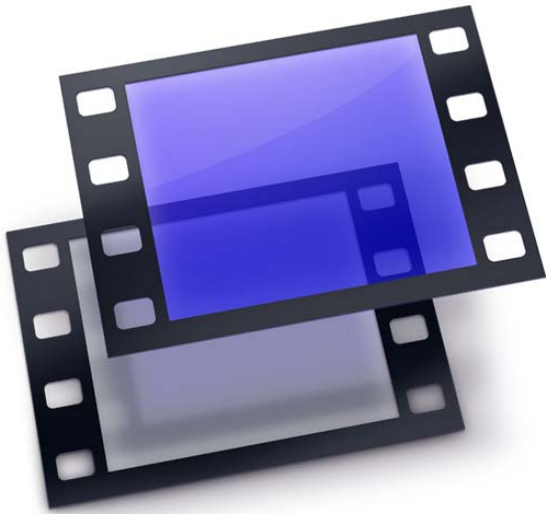





Shake 4 Tutorials



 Apple Computer, Inc.
© 2005 Apple Computer, Inc. All rights reserved.

Under the copyright laws, this manual may not be copied, in whole or in part, without the written consent of Apple. Your rights to the software are governed by the accompanying software license agreement.

The Apple logo is a trademark of Apple Computer, Inc., registered in the U.S. and other countries. Use of the keyboard Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

Every effort has been made to ensure that the information in this manual is accurate. Apple Computer, Inc. is not responsible for printing or clerical errors.

Apple Computer, Inc.
1 Infinite Loop
Cupertino, CA 95014-2084
408-996-1010
www.apple.com

Apple, the Apple logo, Final Cut, Final Cut Pro, FireWire, Mac, Macintosh, Mac OS, Nothing Real, QuickTime, Shake, and TrueType are trademarks of Apple Computer, Inc., registered in the U.S. and other countries.

Adobe and Photoshop are trademarks or registered trademarks of Adobe Systems Incorporated in the U.S. and/or other countries.

Cineon is a registered trademark of Eastman Kodak Company.

Maya, Alias, and Alias|Wavefront are trademarks or registered trademarks of Alias Systems Corp. in the U.S. and/or other countries.

IRIX is a registered trademark of Silicon Graphics, Inc.

3ds Max is a registered trademark of Autodesk Inc.

Other company and product names mentioned herein are trademarks of their respective companies. Mention of third-party products is for informational purposes only and constitutes neither an endorsement nor a recommendation. Apple assumes no responsibility with regard to the performance or use of these products.

Contents

Preface	7	Welcome to Shake 4
	7	The Tutorial Lessons
	8	Installing the Tutorial Media
	8	Mac OS X Notes
Chapter 1	11	Shake Basics
	11	Tutorial Summary
	12	A Tour of the Basics
	16	Loading Images
	19	Viewing Images, Parameters, and Channels
	24	Working With Windows
	24	Launching a Flipbook
	26	Compositing Elements
	31	Setting Resolution
	33	Filtering and Masking
	34	Tuning Parameters
	37	Working With Layer Nodes
	40	Transforming an Image
	41	Fading an Element
	43	Rendering a Sequence
Chapter 2	47	Intermediate Skills
	47	Tutorial Summary
	48	Inserting Nodes Into a Tree
	50	Grouping Nodes and Using SetDOD
	55	Using the Time View
	66	Creating Motion Blur
	71	Importing Photoshop Files
	75	Keyframe Animation and the Curve Editor
	80	Color Correction
Chapter 3	91	Depth Compositing
	91	Tutorial Summary
	92	Simulated Depth and 3D Compositing

93	Working With Z Channels
98	Creating Composites With ZCompose
102	Color Correcting Premultiplied Images
107	Fading With Distance
111	3D Compositing With the MultiPlane Node
121	Animating a MultiPlane Camera
125	Importing Camera and Animation Data

Chapter 4

133	Working With Expressions
133	Tutorial Summary
133	Creating the Fan Composite
136	Creating a Light Source With RGrad
138	Looping Frames in the Time View
139	Using Local Variables and Expressions
144	Simulating Volumetrics With RBlur
153	Concatenating Color Adjustments
156	Adding Motion Blur to Pre-Animated Elements

Chapter 5

163	Using Keylight
163	Tutorial Summary
164	Using Keylight to Pull a Key
165	Testing the Mask With a Viewer Script
167	Adjusting the Mask With Parameters
169	Masking
172	Color Correcting the Foreground Image
175	Advanced Keylight Techniques
178	Using fgBias to Remove Blue Spill
179	Using a Holdout Matte

Chapter 6

183	Using Primatte
183	Tutorial Summary
183	The Basics of Pulling a Key in Primatte
186	Inner Mechanics of Primatte
191	Masking Primatte
194	Spill Suppression in Primatte
200	Compositing Outside of Primatte

Chapter 7

201	Tracking and Stabilization
201	Tutorial Summary
201	Tracking and Stabilizing Nodes
202	Stabilizing an Image Sequence
206	Converting Stabilization Data to MatchMove Data
208	Using the MatchMove Node

- 215 Position the Foreground Element
- 218 Color Correct the Foreground Element

Chapter 8

- 225 **Working With Macros**
- 225 Tutorial Summary
- 225 What Is a Macro?
- 226 Creating a Handmade Macro
- 232 Saving and Testing the Macro
- 234 Adding a Button to the Interface
- 237 How to Set Slider Ranges
- 238 Creating Macros With MacroMaker
- 241 Creating Sliders in MacroMaker

Chapter 9

- 243 **Creating Clean Plates**
- 243 Tutorial Summary
- 243 Stitching Images
- 247 Stabilizing and Stitching Background Plates
- 252 Creating a Clean Plate With QuickPaint

Welcome to Shake 4

This guide includes hands-on tutorials, demonstrations, and explanations of Shake features and workflow.

In addition to the fundamental topics, this guide also explains specialized topics, such as 3D compositing, expressions, keying, and tracking. Check the lesson summaries below for a quick overview of each tutorial.

For further study, you'll want to explore the *Shake 4 User Manual*. This is a two-volume book (also available in PDF format from the onscreen Help menu and in the *Shake/doc* directory) that contains detailed information about color correction, keying and spill suppression, masking, transforms, premultiplication, bit depth, logarithmic color space, caching and optimization, and other Shake features. The *Shake 4 User Manual* also includes a helpful "Cookbook" chapter with additional tips and macros to improve your workflow and productivity.

The Tutorial Lessons

- *Tutorial 1: "Shake Basics"*—This tutorial introduces Shake through a series of common tasks, including loading and compositing images, tuning parameters, transforming images, adding masks, and rendering.
- *Tutorial 2: "Intermediate Skills"*—This tutorial shows how to optimize your workflow with the *SetDOD* node, and how to use the Shake Time View, and the Curve Editor. You will also learn how to add motion blur, how to color-match the elements in a composite, and how to import Photoshop files as layers in a composite.
- *Tutorial 3: "Depth Compositing"*—This tutorial demonstrates different methods for creating "real" and simulated depth in your composites. You'll start with Z channels and filtering options. Then you'll work with Shake's *MultiPlane* node.
- *Tutorial 4: "Working With Expressions"*—This tutorial shows how to generate animation with expressions, rather than keyframes.
- *Tutorial 5: "Using Keylight"*—This two-part lesson covers the basics of using the *Keylight* node: pulling keys, applying masking, creating holdout mattes, and performing spill suppression.

- *Tutorial 6: “Using Primatte”*—This lesson describes the basic use and mechanics of the Photron Primatte keying plug-in, as well as masking and spill suppression.
- *Tutorial 7: “Tracking and Stabilization”*—This tutorial demonstrates the primary uses for Shake’s tracking technology, including removing unwanted motion from an image sequence and “matchmoving” an element to the motion of another element in the composite.
- *Tutorial 8: “Working With Macros”*—This tutorial demonstrates how to create reusable groups of commands, called macros. In this example, you’ll set up a basic macro for a motion blur effect that is adjustable to any angle.
- *Tutorial 9: “Creating Clean Plates”*—This tutorial demonstrates how to stitch images with the *AutoAlign* node, and how to use the *SmoothCam* node to stabilize footage. You will also use the *QuickPaint* node to create a clean background plate.

Installing the Tutorial Media

Before you continue with the tutorials, you need to install the tutorial media. The sample files for the lessons are located on the Shake Installation disk, in the *Documentation/Tutorial_Media* directory. Licensed users can also download these files from the Shake Installation website.

- *Installation CD*: Copy the *Tutorial_Media* folder from the *Documentation* directory to your *\$HOME/nreal* directory.
- *Online (Linux/IRIX Users)*: Contact your system administrator for the URL and password to access the download site for the Shake tutorial media.

Note: You can install the tutorial media files anywhere you like, but the *\$HOME/nreal/Tutorial_Media* directory is used in this guide to simplify the process of instruction.

Mac OS X Notes

The following information applies to Shake on the Mac OS X platform:

Using the Three-Button Mouse

You must use a three-button mouse with Shake as many functions are not possible with a single- or two-button mouse. The middle scroll wheel commonly serves as the middle mouse button. Many commands in Shake require you to “middle-click.”

The Delete Key

The Macintosh Delete key located below the F12 key is the equivalent of the Linux Backspace key; the Macintosh Delete key grouped with the Help, Home, and End keys is the equivalent of the Linux Delete key.

Important: Macintosh users should bear in mind that the Delete key used in Shake is *not* the key located below the F12 key but, rather, the one grouped with the Help, Home, and End keys. If you are using a smaller Macintosh keyboard without the second Delete, use Option-Delete (again, the key below F12).

Keyboard Command Differences Between Platforms

Some keyboard commands are different between the Mac OS X platform and the Linux or IRIX platform. In most cases in this documentation, the Macintosh keyboard command is cited first, followed by the Linux/IRIX command.

Control vs. Command

On the Mac OS X platform, you can use the Control or Command key interchangeably. For example, use Control-C or Command-C to copy an object.

Launching Shake in the Terminal

Mac OS X wraps up binaries and their contents into one icon in the Finder. Click the Shake icon in Mac OS X to launch Shake, or right-click the Shake icon to obtain menu options. For example, right-click the Shake icon and choose Show Package Contents from the shortcut menu to open the subdirectories. Alternatively, you can use the Terminal to navigate to *shake.app/Contents/MacOS/* to find the actual binary files.

This tutorial introduces Shake through a series of common tasks, including loading and compositing images, tuning parameters, transforming images, adding masks, and rendering.



Tutorial Summary

- A tour of the basics
- Loading images
- Viewing images, parameters, and channels
- Working with windows
- Launching a Flipbook
- Compositing elements

- Setting resolution
- Creating a new element
- Tuning parameters
- Working with layer nodes
- Transforming an image
- Applying a mask
- Rendering a sequence

A Tour of the Basics

In this tutorial you'll create a composite with images created at Big Sister's Watching, NY by Brandon Robinson and Melissa Graff.

You'll start by layering the images for the composite. Then you'll incorporate additional elements for soft shadows and lighting. Before you begin, let's review some basic Shake operations.

Launching Shake

You can launch the Shake application from your desktop or from the command line, assuming the Shake binary files are located in the directory created for Shake during installation.

To launch Shake:

- On the Mac OS X platform, browse to the application directory and double-click the Shake icon, or simply click the Shake icon in the Dock.
- On the Linux or IRIX platform, enter the following in any shell:

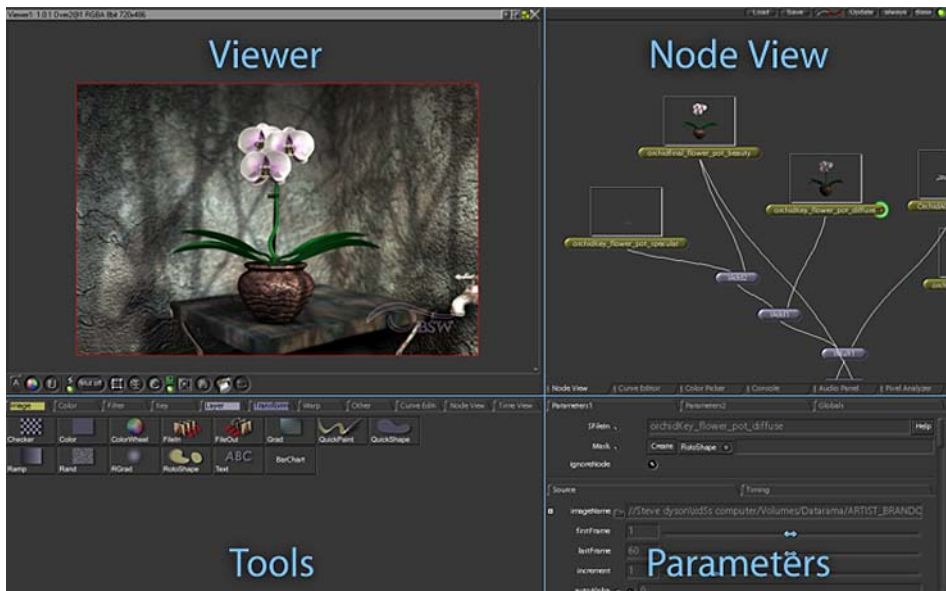
```
shake
```

For you Mac users, the Shake icon may appear on the Dock. If it's not there, then you can drag it to the dock from the application directory. You can also launch from the Mac OS X Terminal, but you must type the complete path to Shake (that is, *Applications/Shake/shake.app/Contents/MacOS/shake*), or set the appropriate environment variables.

Note: For more information, see "Environment Variables for Shake" in Chapter 14 of the *Shake 4 User Manual*.

Using the Shake Panels

When you start Shake, you'll notice that the interface is divided into four panels: Viewer, Node View, Tools, and Parameters.



Each image process in Shake is accomplished by connecting items, called *nodes*, as a tree structure in the Node View. The result is an overview of the images, layers, and processes in your project. Shake projects are called *scripts* because the results are stored as a list of sequential commands in a script file.

So, where do the nodes come from? The Tools panel lists the objects and functions—the nodes—that you can add to your script. The Viewer shows the output of a selected node in the script. The Viewer is also the place where you use interactive controls to transform images and create shapes.

In the Parameters panel, you edit a selected node or change project settings on the Globals tab. And, speaking of tabs, three of the four panels are divided into a number of tabs that allow access to commands, additional parameters, and other functional windows, like the Curve Editor and Audio Panel.

You don't need to know all the screen controls at this point, but you'll probably have some questions while you're working through this tutorial. Being the advanced compositing artist that you are, you never crack open the user documentation—except for this tutorial guide, of course—so how can you figure out what all the screen items do? Contextual help, that's how!

Contextual Help

Move the mouse pointer over a control or parameter to display brief help messages in the Info field at the bottom of the screen. These messages explain what each item does and also show any relevant hot keys.



The Info field displays context-sensitive help.

For example, suppose you want to know what would happen if you clicked on the ColorWheel command on the Image tab. Move your pointer over the command and the message *Create node: "ColorWheel()"* appears in the Info field. Go ahead, click it if you want. You'll add a ColorWheel node to the Node View.



Your first color wheel!

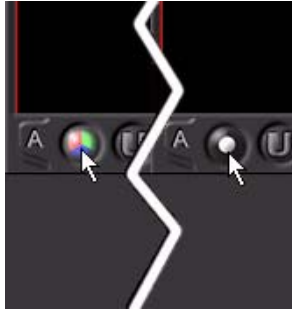
Shortcut Menus and Lists

In addition to the contextual help, you can right-click many controls to display shortcut menus with additional commands and functions. Try this: Right-click the View Channel button. A shortcut menu appears with the hot keys for each channel display option.



For many onscreen controls, there are also click-and-hold behaviors:

- Click a button to toggle between its two default states. For example, click the View Channel button to toggle between RGB and alpha channel views.
- Press and *hold* a button to select an option from a pop-up list. When you press and hold the View Channel button, for example, you can select from a list of available channel options.



Click to toggle between RGB and alpha views.



Press and hold to choose from a pop-up menu.

Overriding the Default Button Choices

To override the default choices, Control-click and hold to choose your next option. For example, the default button behavior for the View Channel button is to toggle between RGB view and alpha view. To modify the behavior to toggle from RGB view to red channel view to alpha channel view and then back to RGB view, perform the following steps:

- 1 Make sure the View Channel button is set to RGB (Color) View.
- 2 Control-click and hold the button, then choose the Red Channel button from the pop-up menu.
- 3 Control-click and hold the button, then choose the Alpha Channel button.

Because the alpha view already toggles to RGB view, you do not have to Control-click and hold again to toggle back to RGB view.

To save this behavior, choose File > Save Interface Settings.

Loading Images

To load images into your project, you use the *FileIn* node from the Image Tool tab. In these tutorials, the notation for creating nodes is identified with this format: Tab Name-Node Name, as in “insert an Image-FileIn node”.



Note: Do not confuse the *FileIn* node with the Load and Save buttons at the upper-right corner of the screen. The Load and Save buttons are for retrieving and saving Shake scripts.

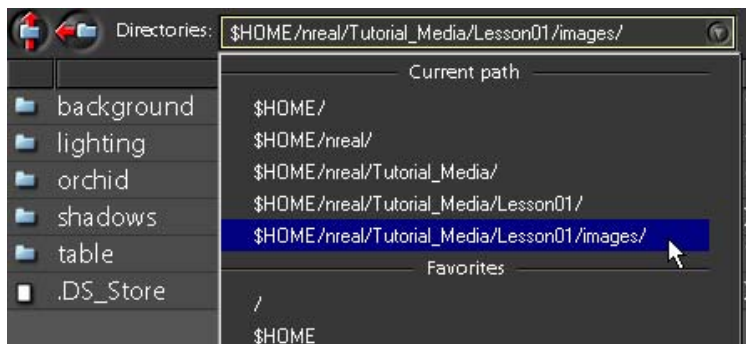


If you still have that ColorWheel node in the Node View, you need to remove it. Click once on the node to select it, then press the Delete (Mac OS X) or Backspace (IRIX/Linux) key.

To load the images:

- 1 In the Image tab, click *FileIn* to launch the File Browser.
- 2 Browse to the `$HOME/nreal/Tutorial_Media/Tutorial_01/images` directory.

Note: See “Browsing Tips” below for information on saving this directory to a list of favorites.

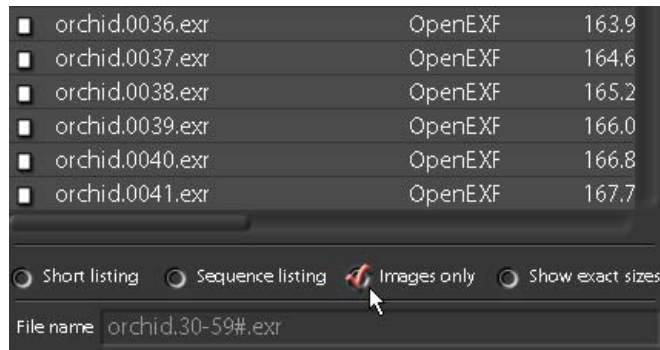


- 3 Double-click the *background* directory to open it, select the *background.30-59@.jpg* image sequence, then click Next at the bottom of the browser (or press the Space bar).
You click Next instead of OK to select additional files. When you're done, you'll load all the images at once.
- 4 Click the Up Directory button, and browse to the *orchid* directory where you'll find the *orchid.30-59#.exr* image sequence.



The "30-59" tells you this is a sequence of 30 frames. The # symbol indicates frame numbers padded with zeros, so that all frames have four-digit numbers. Don't believe it?

- 5 Just for fun, click to deselect the "Sequence listing" checkbox.



You'll see the individual files in the image sequence. This is a good thing to know if you ever need to load just one frame from a sequence.

- 6 Click to select the Sequence listing box again to switch back to the original view.

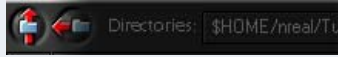


- 7 Select the *orchid.30-59#.sgi* image sequence, then click Next.
- 8 Click the Up Directory button and navigate to the *table* directory, where you'll find the *table.30-59@.exr* image sequence.
This sequence name includes the @ symbol instead of the # symbol. The @ symbol indicates these frame numbers are not padded with leading zeros. Toggle the "Sequence listing" option and you'll see.
- 9 To load the sequence, activate "Sequence listing," then select *table.30-59@.exr*. This time, do *not* click Next!
- 10 Click OK to include the last file and close the browser.

Browsing Tips

There are several ways to navigate within the File Browser:

- To move up one level, click the Up Directory button or press Delete (Mac OS X), or press Backspace (IRIX or Linux).



- To move back to the previously viewed directory, click the Previous Directory button.



- To move down one level, double-click a directory.
- To scroll by file, click in the file list, then press the Up Arrow or Down Arrow key to scroll up or down.
- To scroll using the mouse, middle-click and drag or Option-click and drag (Mac OS X), or Alt-click and drag (you do not have to use the scroll bar on the right). You can also use the mouse scroll wheel.
- To jump to a file name, type a letter to move to the first file that starts with that letter.
- To review browsing history, use the Directories pop-up menu, which lists the files in your *\$HOME* directory, as well as recently visited directories and favorite project directories.
- To save a favorite location, click the Bookmark button near the top of the File Browser. The current location is saved to a list of favorites.



- To import multiple files, click Next, or press the Space bar. On the last file, click OK to import all selected files and close the File Browser.

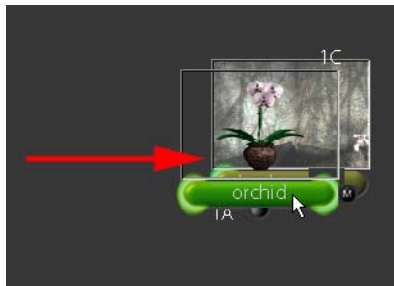
After you click OK, the selected images appear as *FileIn* nodes in the Node View. They might overlap each other, but this is easy to fix.

- 11 Drag a selection box around all the nodes, then press L on your keyboard to line them up. You can also right-click in the Node View and choose Node Layout > Layout Selected from the shortcut menu.



Viewing Images, Parameters, and Channels

Each node represents a function or operation that can be viewed or modified. In this case, these are *FileIn* nodes that reference images from your disk directories. As shown in the previous illustration, image thumbnails appear above the nodes. If there is an accompanying alpha channel, a thumbnail includes transparency, as well. To test this, drag the *mid* node over the *background* node in the node tree and you'll create a mini-composite. Does this help you composite at all? No, but it gives you a quick preview of what the composite might look like.



To create an actual composite, you must connect the nodes. This happens in a moment, so stop fidgeting.

Working With Thumbnails

When working with thumbnails, bear in mind the following:

- The thumbnails represent the frame at the time of file loading.
- To refresh for the current frame, select the node and press R with the pointer in the Node View.

- To view the alpha channel, place the pointer over the thumbnail and press A. To return to the RGB view, place the pointer over the thumbnail and press C for “color.”



- Any node can have a thumbnail—select the node and press T.
Note: The thumbnails do not dynamically update (see below).
- To hide thumbnails, select the nodes and press T. Press T again to show the thumbnails.
- Additional controls for the thumbnails are located in the Globals tab.

Shake does not dynamically update thumbnails because it can be inefficient and inaccurate. For example, if you’re working on 6K plates, do you really want to spend all of your time resizing 6K plates down to tiny icons? (Please say “no.”)

Suppose your script has 900 nodes, which I think we can all agree is not unlikely. Continually updating all thumbnails would require... well, that’s a lot of coffee breaks. The most efficient and accurate way to check a node is to load it into the Viewer.

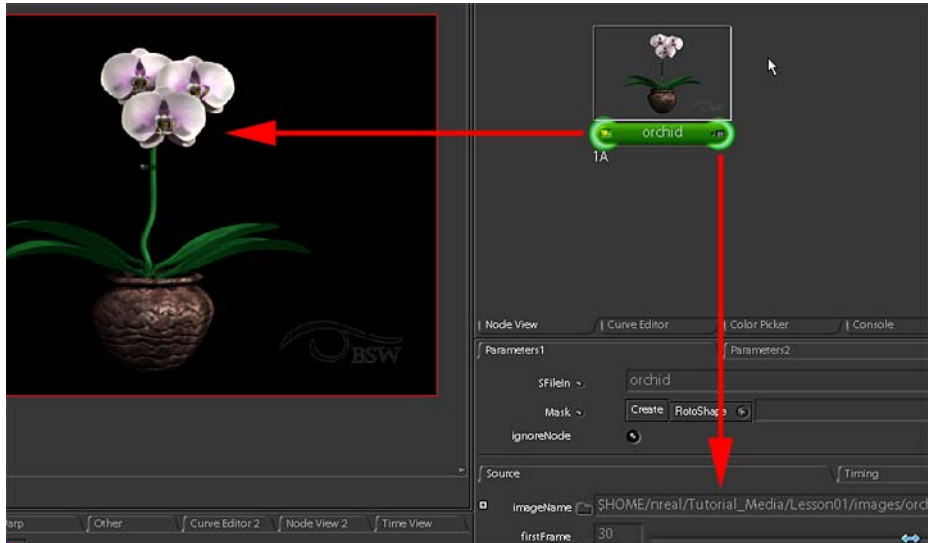
This leads us to the next topic, loading and viewing nodes.

Viewing a Node or Loading Its Parameters

There are several ways to activate the controls on a node:

- To load a node into the Viewer, click the left side of the node.

- To load the parameters into the Parameters tab, click the right side of the node.



- To load a node into the Viewer and the Parameters tab simultaneously, double-click the node.

Sometimes, you need to edit the parameters of one node while viewing the output from another. For example, you'll often want to adjust a color correction while looking at the result in the final composite.



In the illustration above, the *orchid* node is loaded into the Viewer—the highlight on the left side of the node is the Viewer indicator. The *1A* label also appears below the node to indicate it is loaded into Viewer 1, buffer A (stay tuned for information on Viewer buffers).

The gray square on the right side of the *background* node—it's supposed to be a tiny text field—indicates that this node is selected for editing in the Parameters tab. So why is this useful? In a real composite—you'll have one soon—you'll often want to adjust a node while viewing the end result at the final output node in your script.

We mentioned contextual help for screen controls, and this also works to get information about nodes. As you pass the pointer over a node (no need to click), the resolution, bit depth, node name and type, and channels are displayed in the Info field of the Shake window. For example, move the pointer over the *FileIn* node named *background*, and you'll see that it stores an 8-bit image, with RGB channels, and a resolution of 720 x 486 pixels.

Displaying Different Channels in the Viewer

Use the View Channel button to toggle the display of different channels in the Viewer. This is important when you want to check the quality of matte edges or transparencies in the alpha channel. You'll also need to view independent R, G, or B channels for many color-correction operations. For example, click the View Channel button to toggle to the alpha channel view.



There you see the alpha channel. However, nobody actually uses the View Channel button. Remember when you right-clicked this button to see the hot keys?



Use the hot keys when the pointer is in the Viewer to quickly view a channel (C, R, G, B, or A).

Panning, Zooming, and Framing

While working in Shake, you'll need to pan, zoom, and frame the contents of the Node View, the Viewer, and other windows in Shake.

- *To pan:* Drag the pointer over a window while pressing the middle mouse button. Or, drag the pointer while pressing Option (Mac OS X) or Alt (Linux/IRIX)
- *To zoom:* Drag the pointer over the window while pressing Control-Option (Mac OS X) or Control-Alt (Linux/IRIX). You can also press plus (+) or minus (-) to zoom in or out.
- *To frame:* Move the mouse pointer over the window and press F.
- *To expand or shrink the window:* Move the pointer over a panel and press the Space bar This toggles the window between full-screen and standard view.

Setting the Frame Range

There are two places to set the frame range in Shake. The first and most important is in the Globals tab. The Globals tab lists all script settings—the frame range, proxy settings, default resolutions, GUI settings, and quality settings. The first Parameters tab contains a listing of parameters for a selected node.

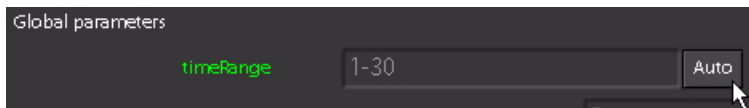
There are two ways to show the Globals tab:

- Click the Globals tab.
- Double-click an empty area in the Node View.

The first parameter in the Globals tab is the *timeRange* parameter. This frame range determines which frames are rendered. Although it is saved in the script, you can override it in the command line. Click the Auto button to examine the *FileIn* nodes and determine the frame range automatically.

To enter the time range:

- Click Auto in the timeRange parameter.



The timeRange parameter is extremely flexible because you can customize the range:

Entry	Calculates
1-56	Frames 1 to 56
20-30	11 images from frames 20 to 30
1-56x3	Frames 1, 4, 7, and so on
1,10,20-30x2	Frames 1, 10, 20, 22, 24, 26, 28, and 30

Working With Windows

Now that you have a few images loaded, this is a good time to practice methods of working with the Shake windows.

Function	Keyboard	Notes
Expand a window full screen	Space bar	Press the Space bar again to zoom back to normal view.
Pan a window	Middle-click and drag, or Option-click and drag (Mac OS X); Alt-click and drag (Linux/IRIX)	Works in all windows, including the File Browser.
Zoom a window	Command–middle-click and drag, Control–middle-click and drag, or Control–Option-click and drag (Mac OS X); Control–Alt-click and drag (Linux/IRIX)	Works in the Node View, the Viewer, the Time Bar, and the Curve Editor.
Zoom in on a Viewer	+ / – (under the Mac OS X function keys); Backspace key (Linux/IRIX)	Gives you an integer-based zoom so you have fewer round-off artifacts on your display. The zoom follows the pointer.
Reset a View	Home	Works in the Node View, the Viewer, the Time Bar, and the Time View.
Resize a pane		Grab the border between two panes and drag to resize the window.

Launching a Flipbook

In the Viewer shelf, click the Flipbook button to render a Flipbook for the node displayed in the current Viewer.



If a *FileOut* node is selected, the actual *FileOut* is not executed. To render to disk, use the Render command (right-click in the Node View, or use the Render menu). Otherwise, the sequence is rendered into memory and you can play it back.

Flipbook Controls

The following table contains several Flipbook controls.

Control	Action
. (Period; think of it as the > key.)	Play forward.
, (Comma; think of it as the < key.)	Play backward.
R, G, B, A, C	Show the red, green, blue, alpha, and color channels.
Shift-drag	Scrub through the animation.
Escape (Esc)	Close the window.

You can have as many Flipbooks as memory allows, but once closed, all links to the Flipbook are lost—you cannot save or use the images again. You can stow the Flipbooks and retrieve them later, but they take up memory.

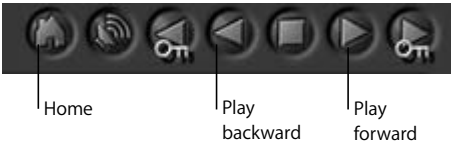
You can also use the Time Bar to indicate a frame range and obtain playback. The Time Bar displays the range that you want to concentrate on, and does not get saved into the script.

To set the playback range for the Time Bar:

- Place the pointer over the Time Bar and press Home on the keyboard (or click the Home button in the Time Bar) to load the script’s timeRange into the Time Bar.

To play the sequence in the Viewer:

- Click the Play forward button in the Time Bar to play through the sequence. Playback does not occur in not real time, but does place the images into the memory cache, optimizing future calculations in the interface.



- Shift-click the Play Forward button to playback from the images cached to disk from the Time Bar.

For more information on the Flipbook, see “The Flipbook, Monitor Previews, and Color Calibration” in Chapter 11 of the *Shake 4 User Manual*.

Compositing Elements

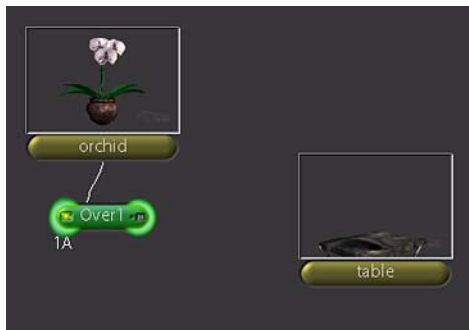
Now that you know how to navigate through the Shake interface, you're ready to composite the elements. Finally! Arrange the *FileIn* nodes like this, in the order that you want to composite them:



To begin the composite:

- 1 Click the *orchid* node to select it.
- 2 In the Tool tabs, click the Layer tab, then click *Over*.

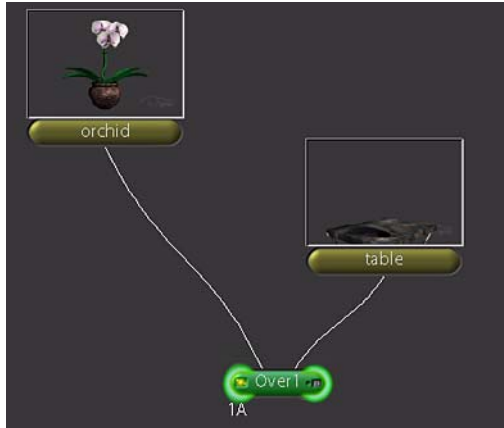
Over1 is automatically attached to the *orchid*, because that node was selected when you added the new node.



The *Over1* node has two inputs on the top of the node, although you won't see them until you place the pointer over the node. The *orchid* node is attached to the first input.

- 3 Move the *Over1* node down a little. Then, drag the second input from the top of the *Over1* node to the bottom of the *table* node to connect the two nodes.

You can also drag from the bottom of *table* to the second input on *Over1*.



Information flows downward in the Node View like a stream—the image data is passed from the *orchid* and *table* nodes, and fed into the *Over1* node to create the composite.



orchid node

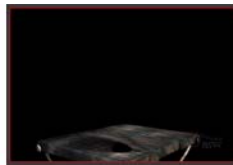
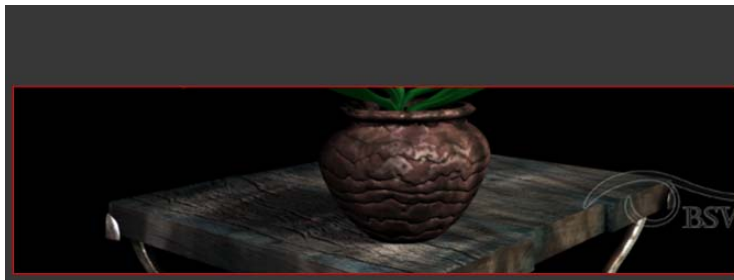


table node

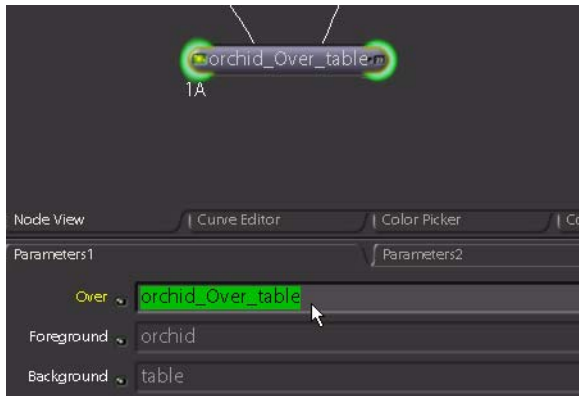


Over1 node

What happened? The top half of the image is gone. We'll fix this in a minute—and explain what it means.



Look in the Parameters tab, where the *Over1* parameters now appear. The first parameter is the same for all nodes: the name of the node. By default, Shake assigns a generic name and appends a number to it, which allows each node to have a unique name. You can type a different name in the text field and make it more descriptive.

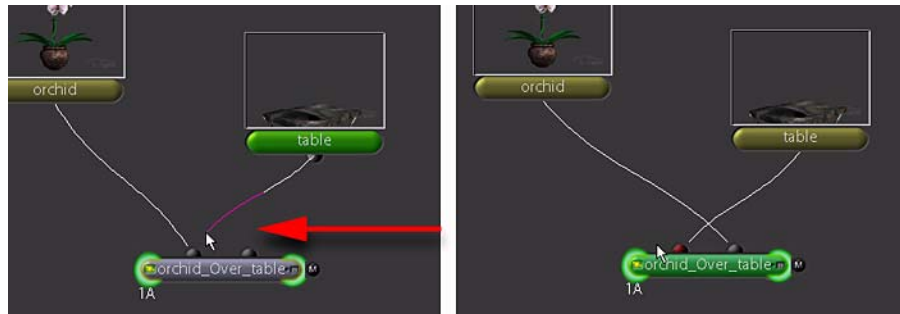


- 4 Click in the first parameter field for *Over1*, and type “orchid_Over_table.”

The new name reflects the compositing logic for the node: “Input 1 is Over Input 2.” If you switch the inputs, then the compositing order is reversed.

Note: When you move the pointer over the line—called a *noodle*—that connects two nodes, the end changes color (magenta = lower end, yellow=upper end) to show that you can drag or delete the connection.

- 5 Drag the lower end of the *table* noodle to the first input on *orchid_Over_table*.



The node logic is switched, which completely changes the result of the composite. The orchid now appears behind the table when it should be on top.



- 6 Switch the inputs again (or press Command-Z or Control-Z to undo your previous operation). The *orchid* image appears over the *table* image again.

As you work with the different layer nodes, you'll find they have distinct methods for creating the layered output. The *Over* node, for example, follows the logic of "Input 1 is Over Input 2"—or more specifically, "The pixel values of Input 1 are placed Over the pixel values of Input 2." As you'll soon see, other layer nodes use different logic to create their output, such as "The pixel values of Input 1 are *Added* to the pixel values of Input 2" or "The pixel values of Input 1 are *Multiplied* by the pixel values of Input 2."

Breaking Connections Between Nodes

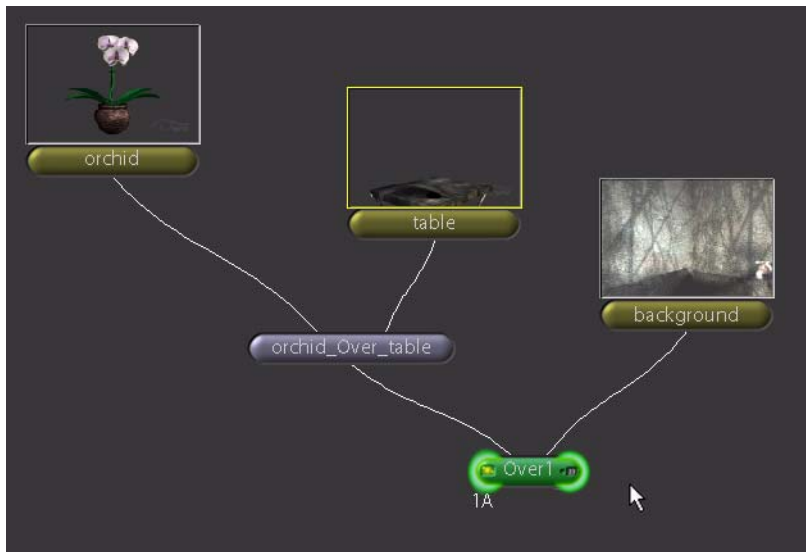
Use one of the following methods when you need to break a connection between nodes:

- Move the pointer over one end of a noodle and press Delete (Mac OS X) or Backspace (IRIX and Linux). You can also Control-click the noodle.



- Select a node and press E on your keyboard to extract the node—and appropriately break all its connections. If necessary, you can always drag any node back over a noodle to insert it again.
- Take advantage of Shake's namesake and quickly drag and shake a node to break its connection.

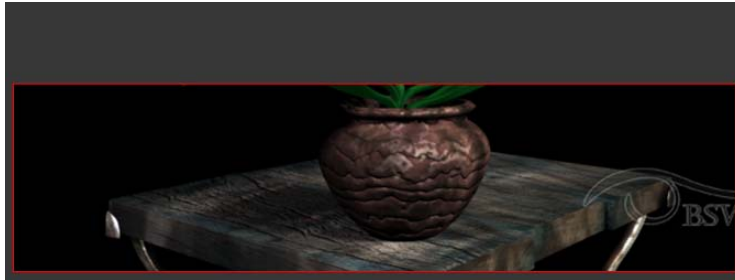
- 7 In the Node View, select the *orchid_Over_table* node and add one more *Over* node. Use the following illustration as a guide to connect the background element.



The result appears in the Viewer. We can thank the folks at Big Sister's Watching, NY for their impressive work. But it didn't always look like this, did it?



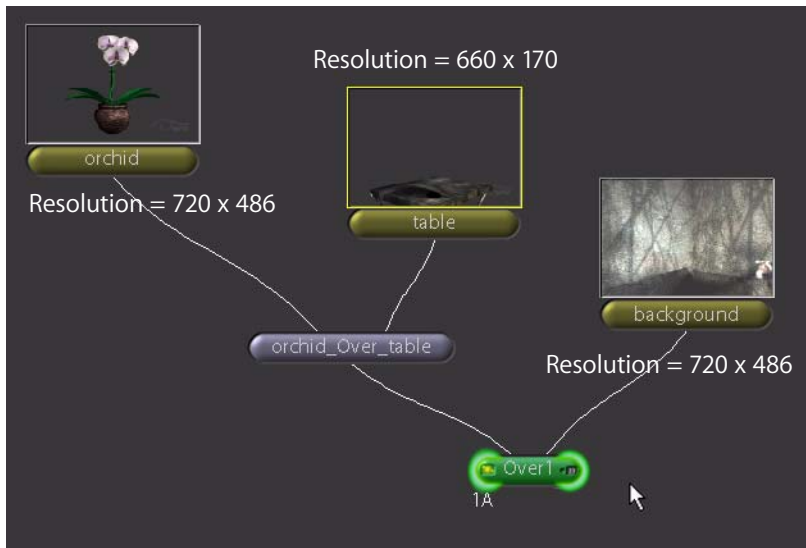
If you were paying attention when you connected images to that first *Over* node, your composite looked like this for a brief period:



Although everything turned out fine, it's important to understand why Shake clipped the image—it's a feature, honest—and how this can help you. This, and other mysteries, are explained in the next section.

Setting Resolution

Shake supports an *Infinite Workspace*, which means you can dynamically change resolution during the compositing process and Shake will handle it. For example, you can simultaneously output an HD image and a 601 video image and your compositing process will be independent of any specific resolution—yes, even independent of the resolution you set up in the Globals tab. You can change resolution at any place, as many times as you need, along the node tree.



Setting Resolution in a Composite

There are several ways to set the resolution for your composite:

- Composite elements so that one of the elements is already set to the resolution you want to use. Then, use the `clipMode` parameter in the layer node (*Over*, *lAdd*, *lMult*, and so on) to indicate which image you want to pass as the resolution for the next node in the tree.
- From the Transform tab, use the *Fit*, *Resize*, or *Zoom* node to scale your images.
- From the Transform tab, use the *Crop*, *Viewport*, or *Window* node to change the size (that is, resolution) of the workspace you're passing down the node tree. These nodes do not resize the image, but instead reset the framing of the image.

Notice that none of these methods involve the Globals tab, where you'd expect all global project settings to be defined. The resolution set in the Globals tab does determine the initial resolution Shake-generated elements such as rotoshapes, ramps and gradients, but it does not change the resolution of images read into the script from outside source files.

So, what does this have to do with the clipped image in our composite? What you saw earlier was the result of Shake's automatic method for adjusting resolution according to the images you're using. Take another look at your composite to see how this information applies.

To set the resolution with the `clipMode` parameter:

- 1 Move the mouse pointer over the *orchid* node.

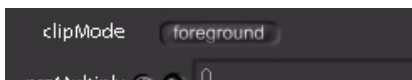
In the Info field at the bottom of the screen, you'll see that this image has a resolution of 720 x 486.

- 2 Now click the left side of the *table* node to load the image into the Viewer.

This image is 660 x 170 pixels—different dimensions than the *orchid* image. When you use a layer node, Shake assumes you want to pass the resolution of the second image (called the “background” image) to the next node in the tree. In this case, Shake took the resolution of the *table* image and passed it downstream. Unfortunately, the smaller resolution also framed out part of the *orchid* image. Don't fret; you can change which input controls the resolution.

- 3 Double-click the *orchid_Over_table* node in the Node View to simultaneously view that node and load its parameters.

In the Parameters tab, you'll see setting called `clipMode`. When `clipMode` is set to “foreground,” the resolution of the first image is used; when set to “background,” the resolution of the second image is used.



- 4 To ensure that the resolution is 720 x 486 pixels, set clipMode to foreground.



The image is no longer clipped.

- 5 Double-click the *Over1* node to view the full composite again.

In this situation, the *Over1* node restores the *orchid* image, even without the clipMode fix. This is because *orchid* isn't truly clipped. The Shake Infinite Workspace ensures that images are never permanently clipped due to framing. If an image is clipped at one point in the node tree, the image data will still be there when the resolution is increased further down the tree.

Filtering and Masking

All the elements in this project are in sharp focus. Softening a portion of the background will add an illusion of depth to the shot. You don't have a "soft-focus" version of the background, but you can create one with a *Blur* node and an *RGrad* mask. Add the blur effect first, then mask it to create a depth-of-focus effect.

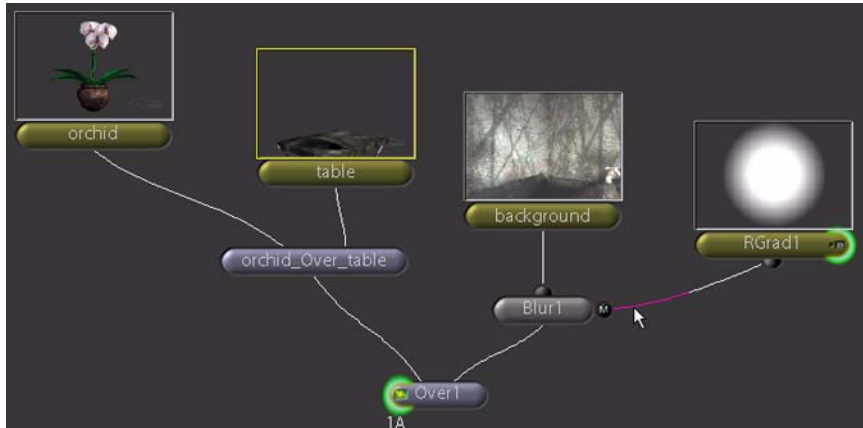
To add the blur effect to the background:

- 1 Select the *background* node, then insert a *Filter-Blur* node.
- 2 Click the left side of the *Over1* node to load it into the Viewer.
- 3 Click the right side of the *Blur1* node to load it into the Parameters tab.
- 4 In the *Blur1* parameters, drag the slider beneath the first pixels value field (the slider appears when you move the pointer over it), and set the blur pixels to 20.



Not bad, but the angle of the background walls means the sides which are closer to the camera should gradually draw into focus. You can fix this with a mask.

- 5 Using the following illustration as a guide, add a new Image-*RGrad* node and attach it to the side mask port on the *Blur1* node.



The blur effect is now controlled by the pixel values in the *RGrad1* alpha channel. Lighter pixels allow the effect to be applied to the background image. Darker pixels block the effect.

Tuning Parameters

So now you've got your basic composite. The mask needs some fine tuning to improve the effect, and this will give you some practice with the Parameter controls. Node controls may include parameters for numerical entry, sliders to set values, color controls, and Viewer overlays (onscreen transform controls) for interactive control. The *RGrad1* node has all these parameters and controls.

Note: If you are using a stylus, open the Globals tab, then turn on `virtualSliderMode` in the `guiSettings` subtree. This assists you with the virtual slider. You should also assign one of the stylus buttons as the right-mouse button.

To adjust the placement and shape of the gradient:

- 1 Click the left side of the *Over1* node to load it into the Viewer.

- 2 Click the right side of the *RGrad1* node to load it into the Parameters tab.



- 3 In the *RGrad1* Parameters tab, set radius to 100, and falloffRadius to 400.
- 4 Display the subtree for center, then change the xCenter value to 360.
- 5 Change aspectRatio to 1.5.

The result will look similar to this:



The overlay circles and crosshairs are interactive transform controls that allow you to make adjustments in the Viewer.

- 6 Drag the inner and outer circles to change the radius and falloffRadius, then drag the crosshairs to move the center of the gradient, adjusting the appearance of focal range.

In the illustration below, the radius is set to 312 and the falloffRadius is set to 370.



It's looking better, but there's still a big difference between the sharp area and the blurred area. Adjust the gradient colors to fix this, then set the *Blur1* node to use a color channel from *RGrad1* as the mask, instead of the alpha channel. When you change the gradient colors and substitute the alpha channel with one of the color channels, you can adjust the opacity of the mask and its effect on the blur filter.

- 7 Click the right side of the *Blur1* node to load its parameters, expand the Mask subtree, then set maskChannel to R (red).



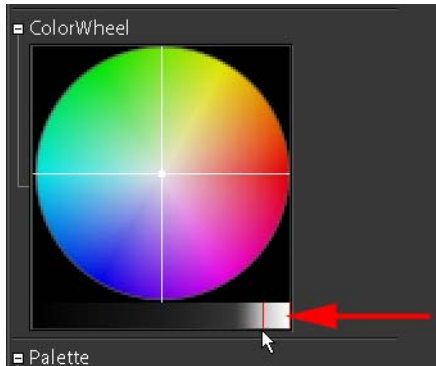
It doesn't matter which of the three color channels you specify. You just need to choose one of the color channels to use as the mask.

- 8 Now click the right side of the *RGrad1* node to load it back into the Parameters tab.
- 9 Click the centerColor control.

The Color Picker opens.



- 10 Drag in the luminance bar, under the ColorWheel, until you see a softer focus for the background in the Viewer.



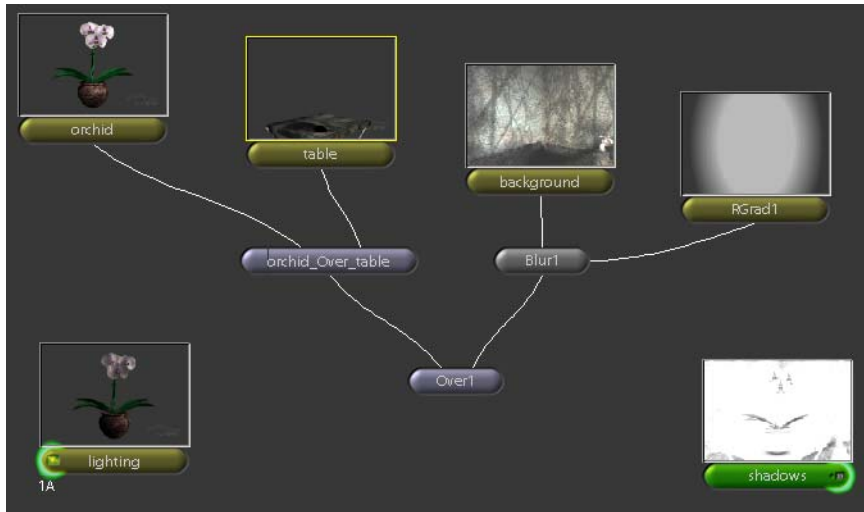
Working With Layer Nodes

This example includes other elements for soft shadows and lighting. Instead of the *Over* node, you'll use *Layer-Mult* and *Layer-Screen* to blend these elements into the composite you created in the previous exercise.

To load the lighting and shadow images:

- 1 Add an *Image-FileIn* node to the node tree.
- 2 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_01/images/lighting` directory, select `lighting.30-59@.exr`, then click Next (or press the Space bar).
- 3 Move up one directory, then open the `shadows` directory, select `shadows.30-59@.exr`, then click OK.

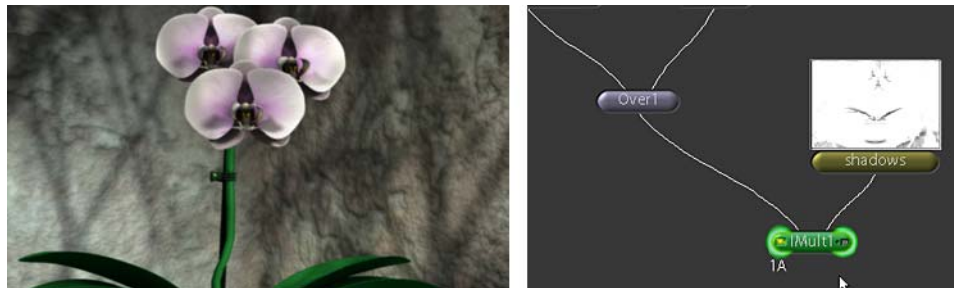
- 4 Arrange the nodes as shown in the illustration below.



Composite the soft shadows first.

- 5 Select the *Over1* node, then add a *Layer-Mult* node to the tree.
- 6 Connect the *shadows* image to the second input.

Rather than place one image over another, the *IMult* node multiplies pixel values together. For this reason, it doesn't matter that the *shadows* image is connected to the second input. The shadow image is multiplied into the existing image, not placed on top of it. Here is the result:



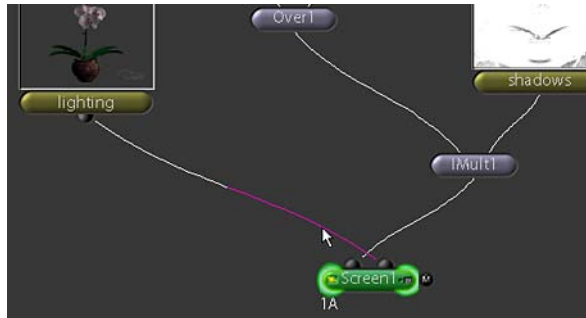
Now you have some nice soft shadows in the nooks and crannies, but they might be a little too dark. You can adjust the amount that the first input is multiplied by the second input.

- 7 In the *IMult1* parameters, drag the percent slider to set it to 65.

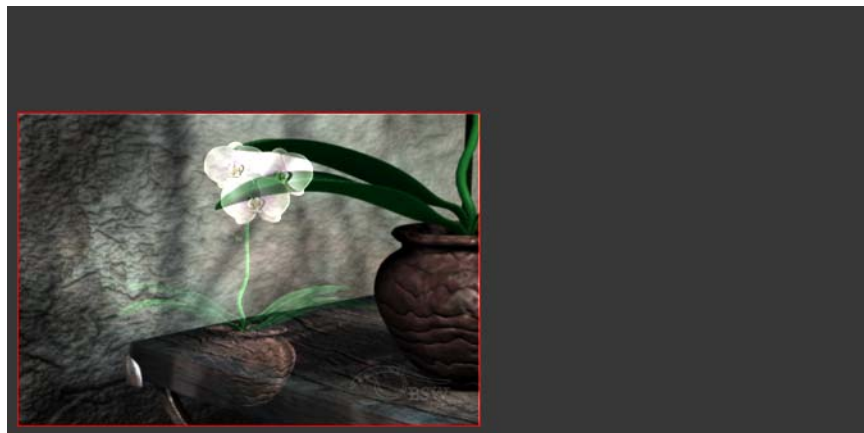


Next, composite the lighting pass.

- 8 Select the *IMult1* node and insert a Layer-Screen node.
- 9 Connect the *lighting* node to the second input of the *Screen1* node.

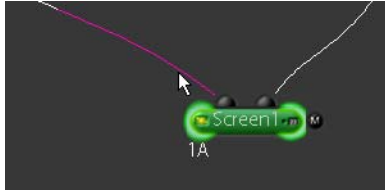


Not again! In the Viewer, you'll see a clipped image, similar to what you saw earlier in this tutorial.



Move the pointer over the *lighting* node and you'll see it has a resolution of 360 x 243. This resolution is obviously not the desired output.

- 10 Drag the noodle from the second input on *Screen1* to the first input.



That fixed the clipping situation but didn't change the size of the *lighting* image to match the rest of the composite. You could re-render this element at the full resolution, but who has time to do that? Alternatively, you can resize it with a transform node (just don't tell your CG supervisor).

Transforming an Image

Most of the transform nodes display onscreen controls that let you interactively scale, move, and even distort the image. In this example, use the *Move2D* node to resize and position the *lighting* image.

You can speed up interactive transforms by changing the Viewer update method. Press and hold the "always" button in the upper-right corner of the Node View, then choose "release" from the pop-up menu.



This tells Shake to update the Viewer only after you've released the mouse button.

To scale and position the *lighting* image:

- 1 Select the *lighting* node, then insert a Transform–*Move2D* node.
- 2 Locate the scale parameter in the Parameters tab, then, type "2" in the first value field (xScale).

Or, in the Viewer, drag a corner of the Move2D transform control until the scale parameter is set to 2.



The size is right, but the image is not in the correct position.

- 3 Drag the triangles or the crosshairs to position the image to match the rest of the composite.
- 4 Open the pan subtree in the *Move2D1* parameters, then adjust the values so that $xPan = 181$ and $yPan = 123$.

The lighting pass should now be in the right place.



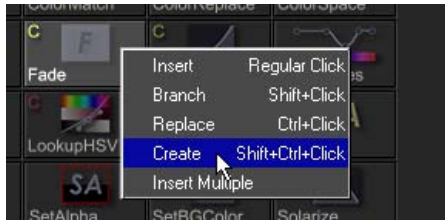
Fading an Element

You're almost done. The final step in this exercise—before rendering the animation—is to adjust the lighting pass. Right now, it's a little washed-out, so you'll use a *Fade* node to control its effect on the rest of the image. *Fade* operates on all channels in the image, including the alpha channel. This time you'll use the shortcut menu to insert the required node.

To adjust the opacity of an image:

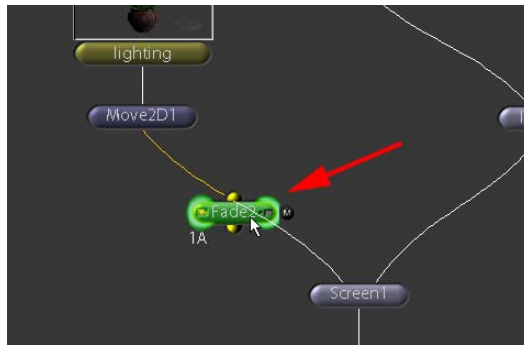
- 1 Click the *Color* tab to display the color command nodes.

- 2 Right-click *Fade*, then choose *Create* from the shortcut menu (or, you can click *Fade* while holding down the Control and Shift keys).



This inserts a new *Fade1* node that is not connected to any existing nodes.

- 3 Drag *Fade1* over the connection between the *Move2D1* and the *Screen1* nodes. When the inputs turn yellow, release the mouse button and *Fade1* is inserted between the two nodes.



- 4 In the *Fade1* parameters, drag the value slider to 0.25.
This reduces the opacity of the lighting pass to 25 percent.



The finished image will look similar to this:



Rendering a Sequence

When you're ready to create the final images for your composite, you insert at least one *FileOut* node at the place in the node tree where you want to render. This is often at the bottom of the node tree, because this is where the final result is composited. However, you can place as many *FileOut* nodes as you need to output from different places along the tree, and send rendered output to multiple directories and formats.

To add a *FileOut* node:

- 1 In the Globals tab, click the Auto button to ensure that the timeRange is set to 1-30.
- 2 Attach an Image-*FileOut* node to *Over1*.
The File Browser opens.
- 3 In the File Browser, navigate to the desired directory.
- 4 In the File Name text field at the bottom of the browser, type "comp.#.sgi" as the file name.

About File Names

When you enter the file name for render output, the name should include a combination of the following:

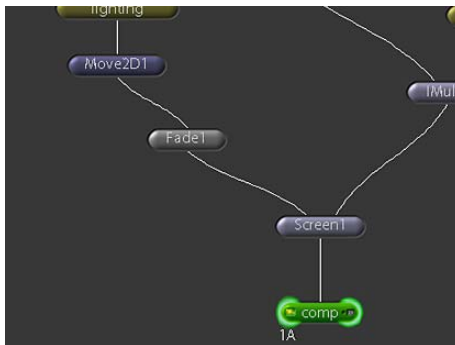
- The name of the file.
- A frame-numbering symbol for image sequences. When you render to an image sequence, include a frame numbering symbol—either @ or #. A single @ is an unpadding number—that is, 1, 2, 3, 4. A single # is 4-place padding—that is, 0001, 0002, 0003. Either symbol used multiple times indicates that many places of padding, that is, @@@@ is 00001, 00002, 00003.
- A .mov extension for QuickTime files. When you render to a QuickTime file, omit the frame-numbering symbol and append the file name with the .mov extension. The QuickTime format options appear in the *FileOut* parameters.
- A file format extension, such as .cin, .sgi, .jpg, .iff, and so on, is required.

For example:

Enter *comp.#.sgi* as your file name, and the rendered images are written as *comp.0001.sgi*, *comp.0002.sgi*, *comp.0003.sgi*, and so on.



- 5 Once you have entered your file name settings in the File Browser, click OK.



To save the script:

- 1 Click the Save button in the upper-right corner of the Shake window (or press Command-S or Control-S).

Because this is the first time you are saving the script, the Save Script window appears. Next time you click Save, Shake updates the existing file with the latest changes.

- 2 In the File Name text field, type “orchid.shk” as the name for the script, then click OK.

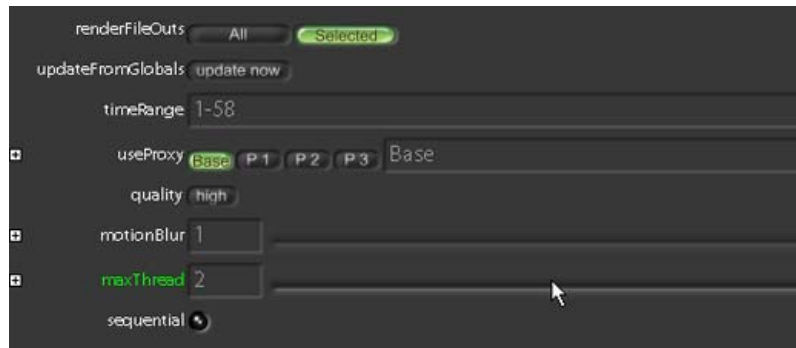
Note: To Save As, use Command-Shift-S or Control-Shift-S, or choose File > Save As.

To render files to disk:

- 1 From the menu bar, choose Render > Render FileOut Nodes.

The Render window appears.

Note: You can also right-click in the Node View, then choose Render > Render FileOut Nodes from the shortcut menu.



In the Render window, you can enter a new time range, proxy scale, quality level, motion blur, and the number of CPUs to use for the render. To render all *FileOut* nodes, enable All in the renderFileOuts parameter.

To render only active FileOuts, enable Selected. You can also select a *FileOut* node after the Render Parameters window is opened.

- 2 Click the Auto button to copy the time range from the Globals tab.
- 3 Click Render.

The images are rendered to disk, and a 320 x 243 snapshot view of the current frame is displayed. The snapshot is always 320 x 243, regardless of input resolution, and only shows the currently rendering frame.

Batch Rendering in a Shell

You can choose to render on the command line. Click the Save button at the top of the Node View (or press Command-Shift-S or Control-Shift-S) to save the script. If you have not yet saved the script, you are prompted for a script name; otherwise it writes over what you had before.

Note: If you are working in a shell, you have usually set environment variables. For more information, see “Environment Variables for Shake” in Chapter 14 of the *Shake 4 User Manual*.

- 1 Save your script and hide or quit Shake.

2 Open Terminal and navigate to the directory where your *orchid.shk* script is saved.

3 Type:

```
shake -exec orchid.shk -v
```

Note: In OS X, if you do not have environment variables set for Shake, you must type the full Shake path, for example:

```
/Applications/Shake/shake.app/Contents/MacOS/shake -exec orchid.shk -v
```

Usually, you set your environment variables to use the *shake* command with a lower-case “s.” For more information, see “Environment Variables for Shake” in Chapter 14 of the *Shake 4 User Manual*.

On a Linux or IRIX system, type the following (the Shake command begins with a lower-case “s”):

```
shake -exec orchid.shk -v
```

The -v means “verbose,” so the render status is displayed. For more information on the command line, see Appendix B, “The Shake Command-Line Manual,” in the *Shake 4 User Manual*.

4 To override the time range, use the -t option:

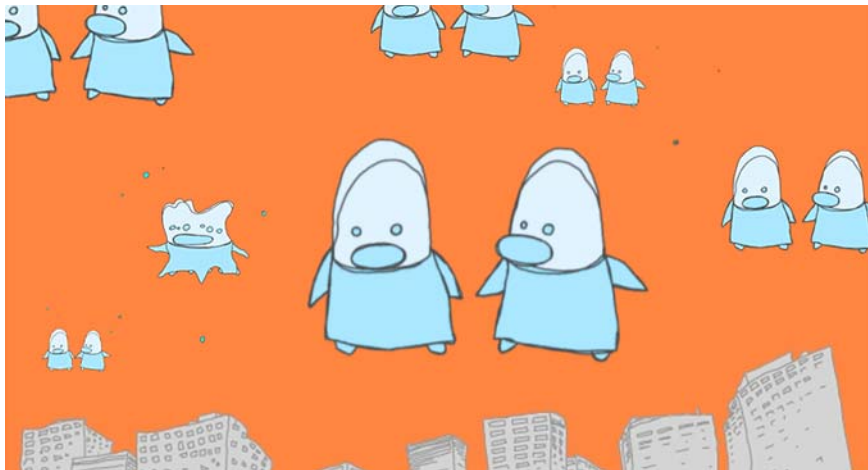
```
shake -exec orchid.shk -v -t 10-20
```

or

```
shake -exec orchid.shk -v -t 1-56x2
```

The above command renders every other frame.

This tutorial shows how to optimize your workflow with the SetDOD node, and how to use the Shake Time View, and the Curve Editor. You will also learn how to add motion blur, how to color-match the elements in a composite, and how to import Photoshop files as layers in a composite.



Tutorial Summary

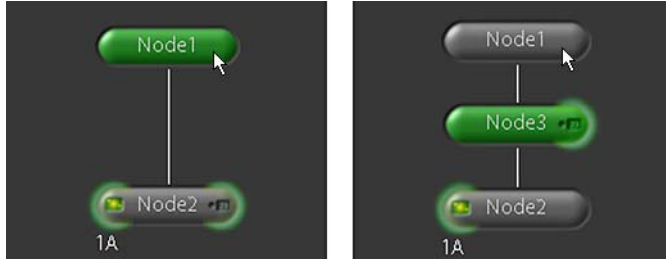
- Inserting nodes into a tree
- Using the Time View
- Grouping nodes and using *SetDOD*
- Creating motion blur
- Importing Photoshop files
- Keyframe animation and the Curve Editor
- Color correction

To explore the topics of this tutorial, you'll work with images from the Beanfield music video *Tides*, kindly provided by the artist, Maximilian Graenitz.

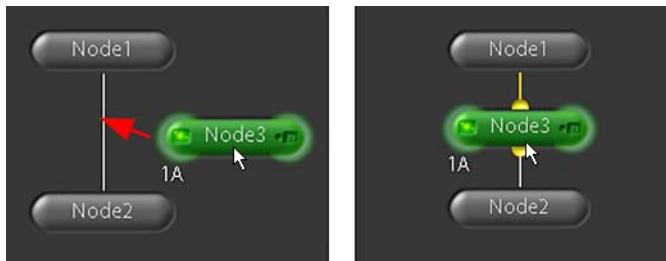
Inserting Nodes Into a Tree

Before you continue with the tutorial, take a moment to review different ways to insert and manage items in the Node View. The illustrations show a generic “node,” but you can try these with any node from the Tool tabs.

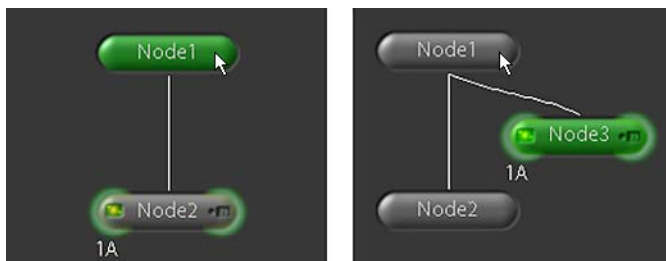
- *Insert, Method 1:* Select the parent node, then click a node button in the Tool tabs.



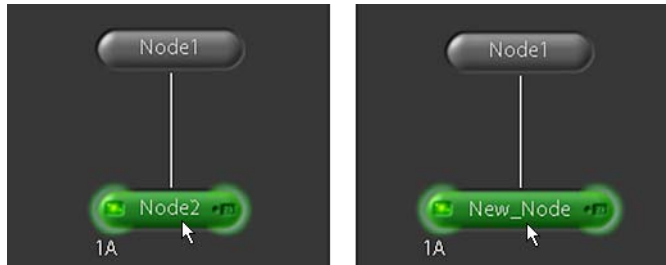
- *Insert, Method 2:* Drag an existing node onto a noodle to insert it between connected nodes.



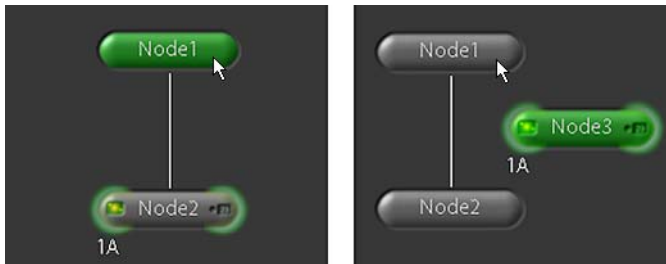
- *Branch:* Select the parent node, then Shift-click a node button in the Tool tabs.



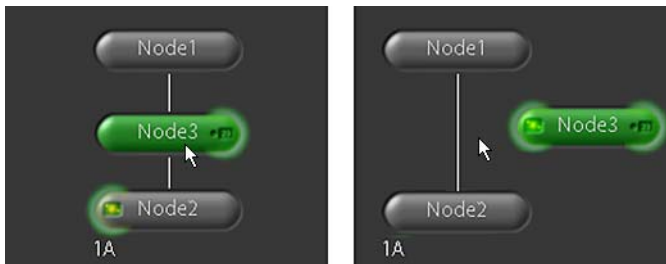
- *Replace*: Select the node you want to replace, then Control-click a node button in the Tool tabs.



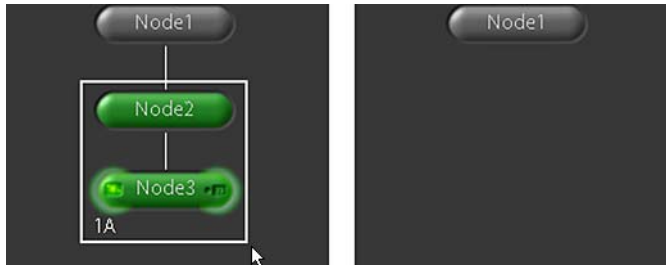
- *Insert Unconnected*: Control-Shift-click a node button in the Tool tabs. The new node appears in the Node View unconnected to any other node.



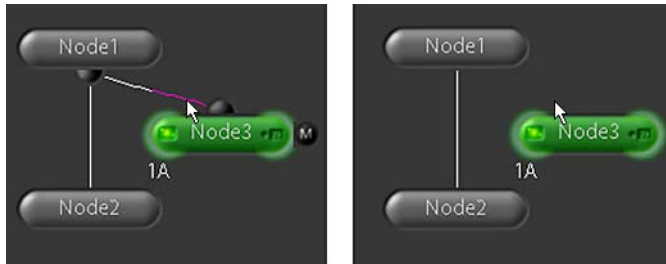
- *Extract*: Select the node, then press E (for Extract). The node is disconnected from the tree.



- *Delete*: Select the nodes by clicking, Shift-clicking or dragging a selection box, then press Delete.



- *Delete Connection*: Control-click the noodle, or move the pointer over one end of the noodle (it turns magenta at the bottom or yellow at the top), then press Delete.

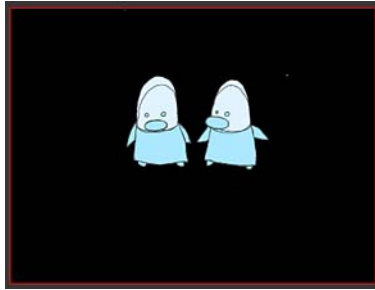


Grouping Nodes and Using SetDOD

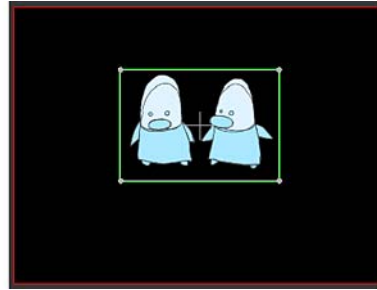
When your script includes many elements, two features can help you control the organization and rendering of the nodes: the *SetDOD* node and the Groups command. *SetDOD* provides a powerful optimization step that reduces rendering time, I/O activity, and memory use. The Groups commands lets you combine two or more nodes into a “grouped” node in the tree.

Optimizing With SetDOD

A *DOD* (Domain of Definition) is a rectangular region around an element that defines the part of the image you want to include in render calculations. Usually, the DOD is used to exclude empty image areas from render calculations.

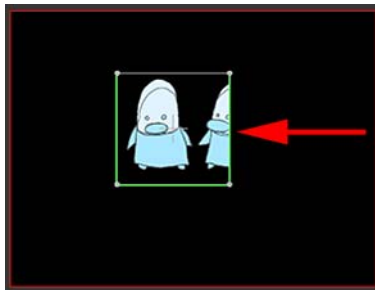


Before *SetDOD*: Whole image area is calculated and rendered.



After *SetDOD*: Only image area inside DOD is calculated and rendered.

But you can also use *SetDOD* to crop out parts of an image that you don't want to include in your composition. Give it a try.



To open the *little_guys.shk* script:

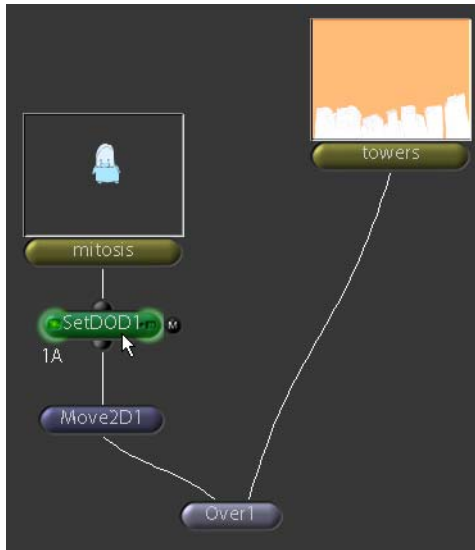
- 1 Click the Load button at the top of the Node View.



- 2 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_02/scripts` directory.
- 3 Select the *little_guys.shk* file and click OK.

To add the *SetDOD* node:

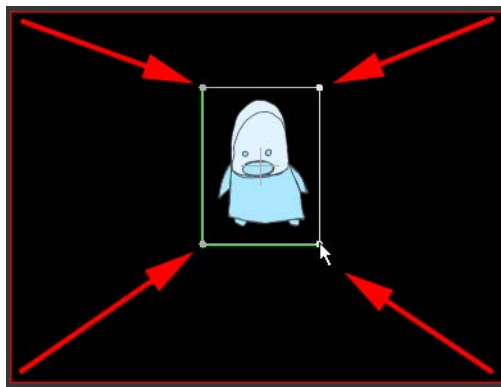
- 1 In the Node View, select the *mitosis* node.
- 2 Insert a Transform-*SetDOD* node.



- 3 Make sure the onscreen controls are enabled for the Viewer.



- 4 Drag the DOD corners or edges in the Viewer to frame the "little guy" character.



This tells Shake to calculate only the image information within the DOD boundaries. The rest of the image is ignored, thereby reducing memory usage, I/O activity, and processing time, both upstream and downstream in the node tree.

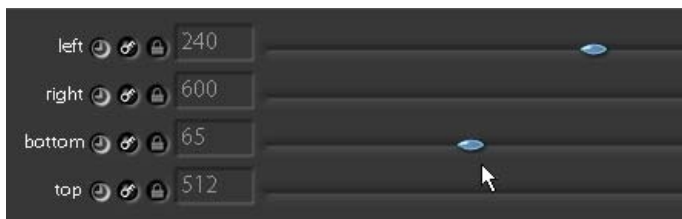
- 5 Click the Flipbook button in the Viewer shelf to create a preview of the image with the DOD applied.



When you play back the preview (press the period key), you'll see that the DOD crops some of the animation as the clip plays. To fix this, make an adjustment to the border.

To adjust the *SetDOD* border:

- 1 Load the *SetDOD1* node into the Parameters tab.
- 2 Change the border setting to these values: left = 240, right = 600, bottom = 65, and top = 512.



This allows enough border space to include all the animation from the clip.

Note: You can further optimize rendering for each frame by animating the DOD borders at different places along the duration of the clip. Don't do this for the current tutorial—it will cause problems later in the example—but keep it in mind for future reference.

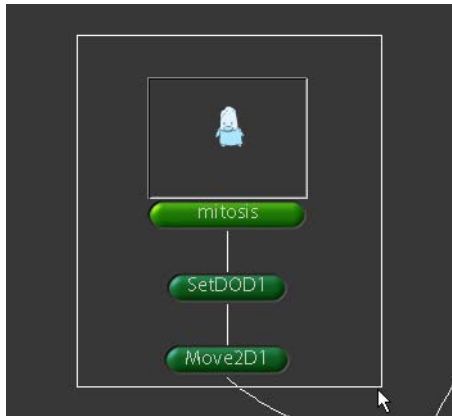
The *SetDOD* node will save you a small amount of render time for each frame. As the number of elements in your composite increases, so does the potential render time. Additional *SetDOD* nodes can dramatically optimize a complex composite.

Grouping Nodes

Next you'll use Shake's Groups commands to combine the *mitosis*, *SetDOD1*, and *Move2D1* nodes as one node in the tree. This will help you copy and manage these elements in the script.

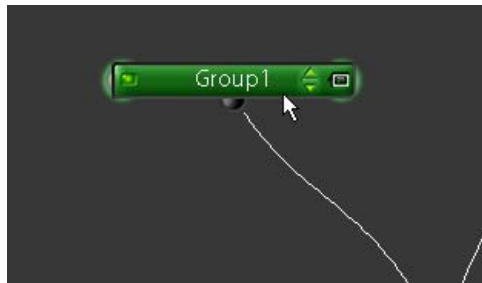
To group the *mitosis*, *setDOD1*, and *Move2D1* nodes:

- 1 In Node View, drag to select the *mitosis*, *setDOD1*, and *Move2D1* nodes.



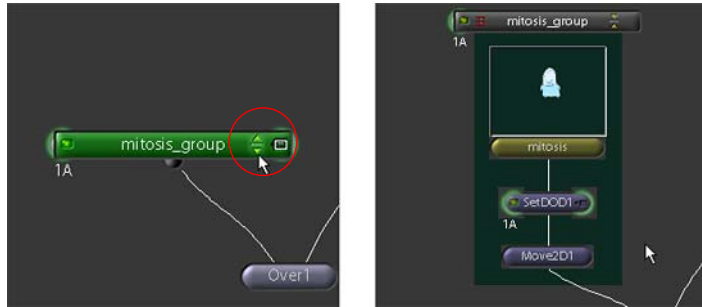
- 2 Right-click in the Node View, then choose Groups > Group Selected Nodes from the shortcut menu.

The new *Group1* node appears in the Node View.



- 3 In the Parameters tab, rename the node to “mitosis_group.”

- 4 Click the Expand button on the *mitosis_group* node to show its contents.



When a group is expanded, you can select and edit the nodes inside the group.

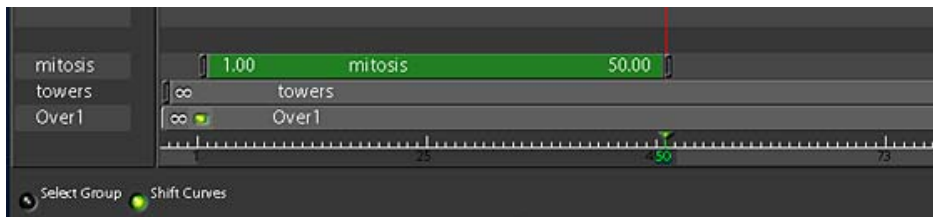
- 5 Click the Expand button again to hide its contents.

Using the Time View

In the next example, you'll make a few adjustments to the *mitosis* clip in the Time View, where you can modify the location and duration of clips.

To display the Time View:

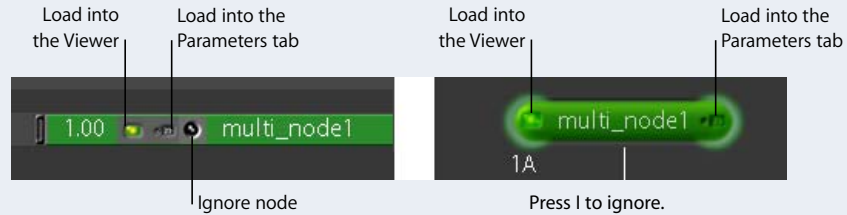
- Click the Time View tab (on the right side of the Tool tabs, underneath the Viewer), then then press the Space bar to expand this pane to full screen size.



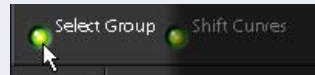
Each image node and layer node in the composite is represented by a horizontal bar, lined up with the frames in the Time Bar.

Tips for Working in the Time View

- The Time View shares similar functions with the Node View. You can select nodes, load parameters, and ignore nodes by clicking the controls on the bars in the Time View.



- Turn on the Select Group button to load only the active nodes into the Time View.



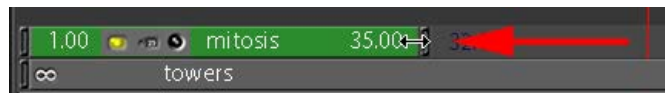
- To scale the Time View, press Command or Control while holding down the middle mouse button, and drag left or right.
- To pan the Time View, hold down the middle mouse button and drag left or right.

The *mitosis* clip starts at frame 1 and ends at frame 50. The *towers* bar is a still image with no start or end frame—notice the “infinity” (∞) symbols at each end of the *towers* bar. The *Over1* bar also has the “infinity” symbols at each end. You change the length of a clip by dragging its timing handles, located on either end of the bar.

Note: The *mitosis_group* node does not appear in the Time View because groups do not have In/Out points in time. The Time View shows only images, image sequences, movie files, and layer nodes.

To modify the duration of the *mitosis* clip:

- Drag the right timing handle to the left until its number reads 35.



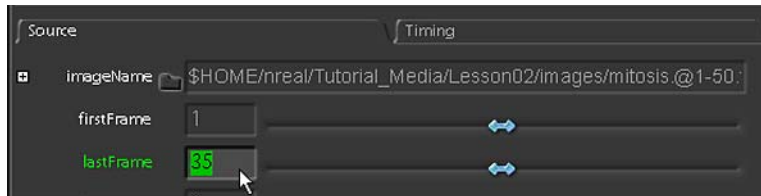
- Press the Space bar again to restore the original size of the Time View.

You’ve just shortened the length of the clip. The new start/end frames also appear in the *mitosis* Parameters tab.

To load the *mitosis* node into the Parameters tab, do one of the following:

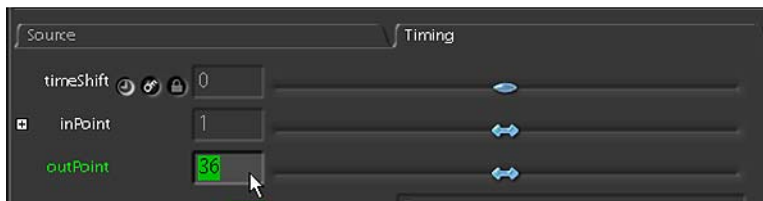
- In the Node View click the right side of the *mitosis* node.
- In the Time Bar, click the parameters indicator on the *mitosis* bar.

The *mitosis* parameters appear in the Parameters tab.



Under the Source subtab, the firstFrame parameter is set to 1 and the lastFrame parameter is set to 35. These are the source frames read from the image sequence on disk.

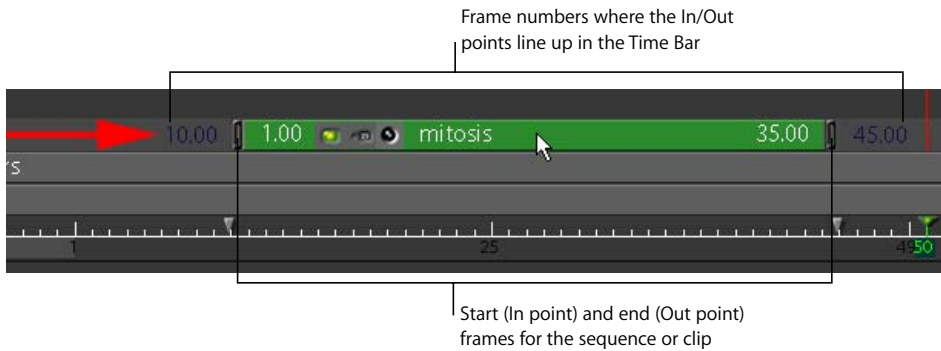
Click the Timing subtab and you'll see how the clip lines up with the Time Bar frames. The inPoint is set to 1 and the outPoint is set to 36, which is one frame *after* the last frame in your clip.



Suppose you want to change where the clip begins in the Time Bar, but you don't want to change the number of frames used from the clip.

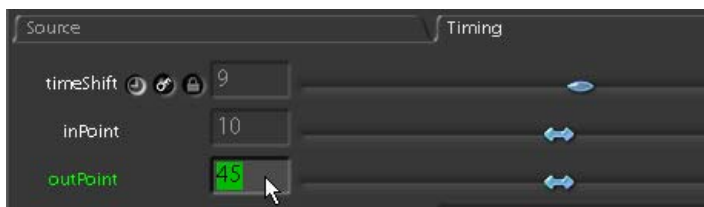
To modify the clip's position in the Time Bar:

- In the Time View, drag the middle of the *mitosis* bar to the right.
The entire clip moves to the right.



As you drag the *mitosis* bar, dark blue numbers appear outside the ends of the bar to show where the start and end frames of the clip line up with the Time Bar frames.

Check the Timing subtab, and you'll see the new time shift indicated there also.



The first parameter, timeShift, shows the difference, if any, between the frame numbers of the clip and the frame numbers in the Time Bar. Positive numbers shift the clip forward and negative numbers shift the clip backward.

The next two parameters, inPoint and outPoint, correspond to the dark blue numbers you saw while dragging the clip in the Time View.

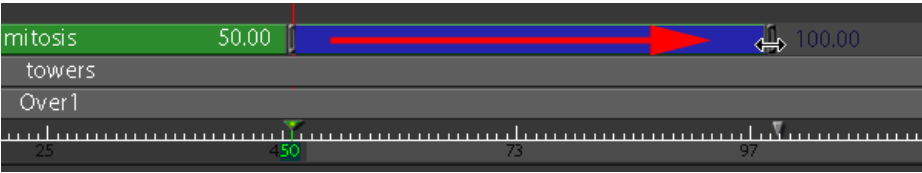
To restore the *mitosis* clip to its original Source and Timing parameters:

- 1 In the Source subtab, set firstFrame to 1 and lastFrame to 50.
- 2 In the Timing subtab, set timeShift to 0, inPoint to 1, and outPoint to 51.

Continuing with this lesson, after extending the length of the *mitosis* clip, you'll create several copies of it, then slide those copies in the Time Bar to stagger their start and end frames. This ensures that the clips are not synchronized, lending the illusion of multiple "little guys" spawning independently.

To extend the length of the *mitosis* clip:

- 1 In the Time View, press the Control or Command key and drag the right timing handle of the *mitosis* clip until the Out point (dark blue number) reads 101.



Pressing the Control or Command key while dragging a timing handle breaks the relationship between the disk clip and the duration of the clip in the composite. The newly extended area of the clip is calculated in one of several repeat modes. The default repeat mode—represented by a blue bar—is a freeze frame: Shake fills the expanded time range in the clip with a freeze frame of the last frame in the clip.




Note: When you drag a timing handle *without* pressing Control or Command, you extend the length of the sequence in the composite without modifying the extended area. In other words, if you drag a timing handle beyond the number of frames in the source clip, the output after the last frame in the source clip will consist of empty black frames.



For the purposes of this example, change the repeat mode from a freeze frame to a loop (or mirror) playback.

- 2 In the Timing subtab of the *mitosis* parameters, set the outMode control to Mirror.



The inMode and outMode parameters tell Shake what to do with a clip that is extended beyond the number of frames available from the image sequence on disk. In this example, the animation wasn't designed to loop, so use the Mirror outMode to repeat the frames in reverse order. In all, there are five repeat modes:

Button	Mode	Result
	Black	No frames repeat.
	Freeze	Last (or first) frame is held and repeated indefinitely.
	Repeat	The entire clip repeats, starting at the first frame.

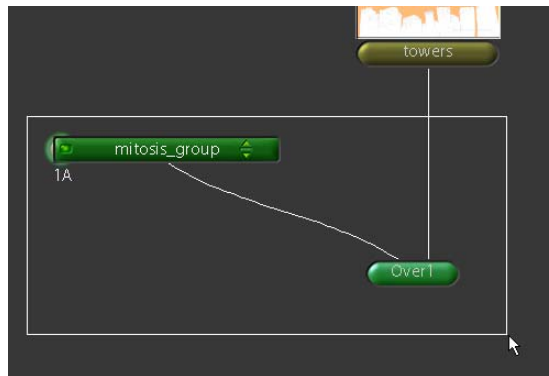
Button	Mode	Result
	Mirror	The clip repeats, first in reverse order, and then forward, indefinitely. To provide a smooth transition, the first frame does not repeat between the loops.
	InclusiveMirror	The entire clip repeats, first in reverse order, and then forward, indefinitely.

You now have everything set up for the first original copy of our “little guys” element. The next step is to make a copy of all the nodes for this element, and paste four or more copies to populate the composite.

To make copies of the elements:

- 1 In the Node View, select both the *mitosis_group* node and the *Over1* node.

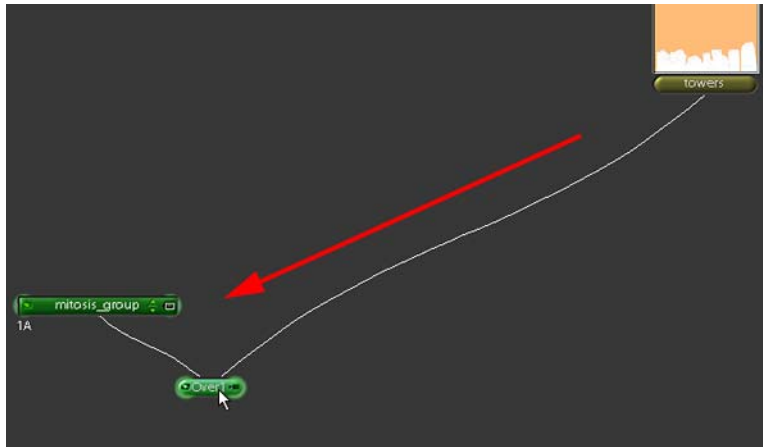
Note: To collapse the *mitosis_group* node, click the Expand/Collapse button on the right side of the group header.



- 2 Right-click in the Node View, then choose Edit > Copy from the shortcut menu.

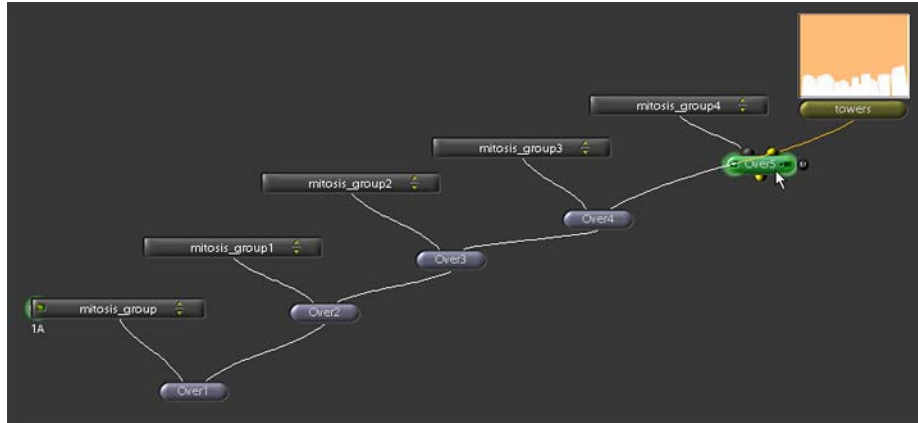


- 3 Drag the selected nodes to the left in the Node View to make room for the copies.



- 4 Right-click in the Node View, then choose Edit > Paste from the shortcut menu (or press Control-V) to place the first copy.
- 5 Repeat Step 4 to paste at least three additional copies.

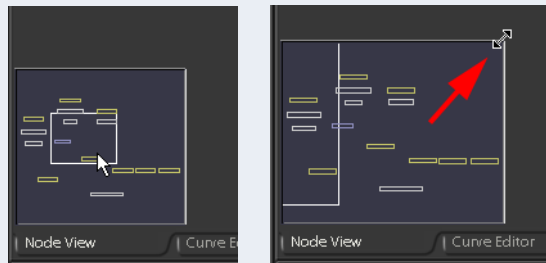
- 6 Using the illustration below as a guide, connect the copies to the rest of the node tree.



Tips for Navigating in the Node View

The Node View can get a little unwieldy when you add a large number of nodes to your script. Here are some tips to help you work more efficiently. With the pointer over the Node View:

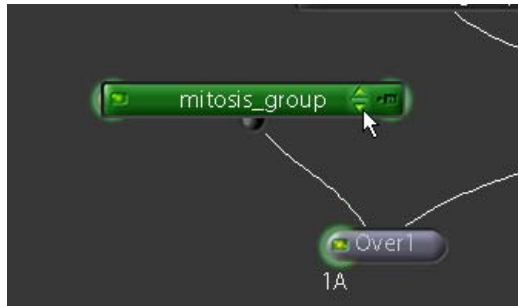
- Press the plus key (+) to zoom in. Press the minus key (–) to zoom out.
- Press F on your keyboard to frame all selected nodes. If none is selected, press F to frame the entire node tree.
- Press O to display the node overview. You can drag inside the overview frame to pan to different areas of the node tree, and also drag the overview border to resize it.



So you have your copies. But they all occupy the same position in the Viewer. Use all those *Move2D* nodes to position the “little guys” at different points in the Viewer.

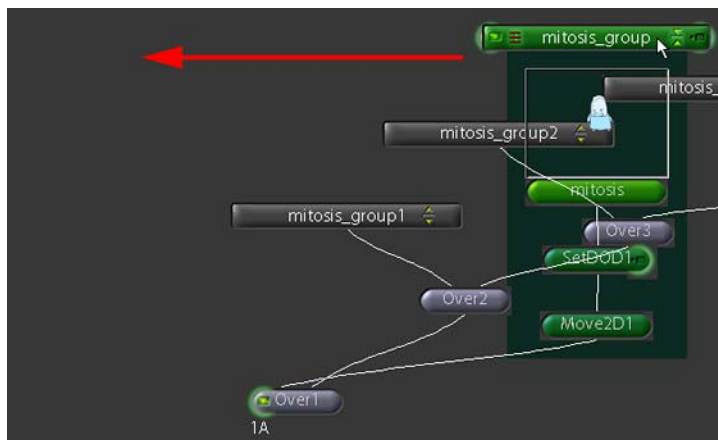
To position the “little guys”:

- 1 Load the leftmost *Over1* node into the Viewer.
- 2 Click the Expand/Collapse button on the leftmost *mitosis_group* node to show its contents.



The group window expands (and jumps to its original paste position).

- 3 Drag the group header to the left to see its contents more clearly.



- 4 Click the right side of the *Move2D1* node to load its parameters into the Parameters tab.



Keep that *Over1* node, at the bottom of the node tree, displayed in the Viewer.

- 5 Drag a corner of the *Move2D* onscreen controls (the box overlay) to change the size of the element.
- 6 Drag the top and side triangles to position the element within the frame.



It doesn't matter where you place the element or how big it is. Just resize it and move it away from the center.

- 7 Click the Expand/Collapse button on the *mitosis_group* node to hide its contents.
- 8 Repeat the previous steps to size and place all the copies of the *mitosis* element.

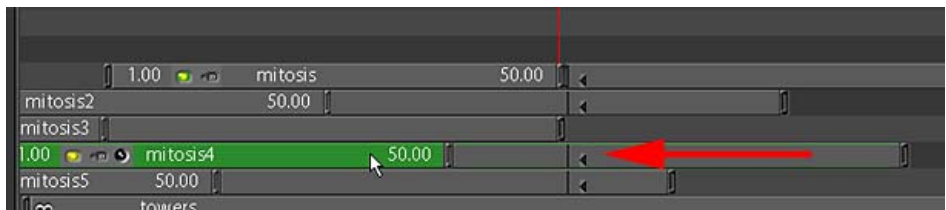
Now you need to stagger the copies in the Time View, to start the animation for each copy at different places in the time line.

To stagger the *mitosis* clips in the Time View:

- 1 Set the Out frame in the Time Bar to 50.



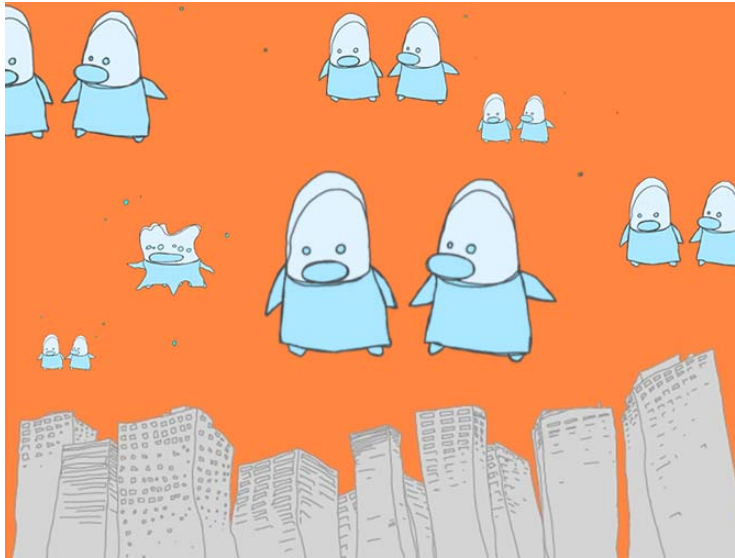
- 2 Open the Time View and drag each copy of the *mitosis* clip—but *not* the original—so that the bars are randomly staggered, like this:



This causes the animation of each copy to start at a unique spot along the Time Bar.

- 3 Double-click the *Over1* node (the last *Over* node at the bottom of the node tree), to see the results of the composite.

When you are finished, your composite should look similar to this:

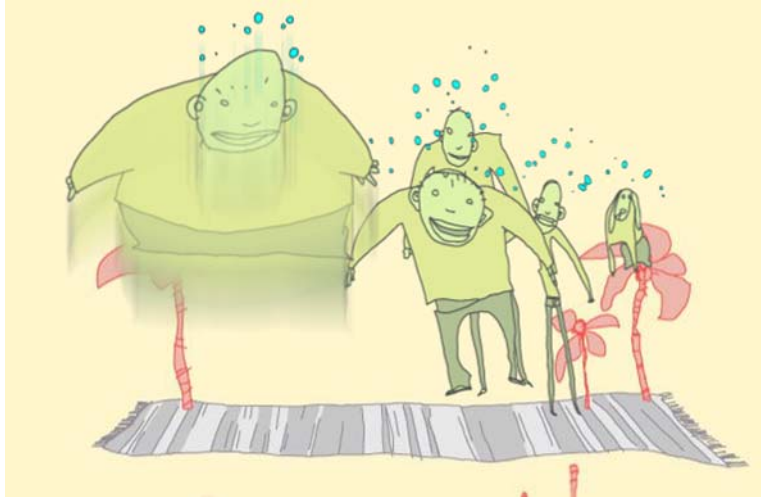


- 4 Click the Flipbook button to preview the finished composite.

That's it for this example. In the next project, you'll see how to add motion blur to elements in a composite.

Creating Motion Blur

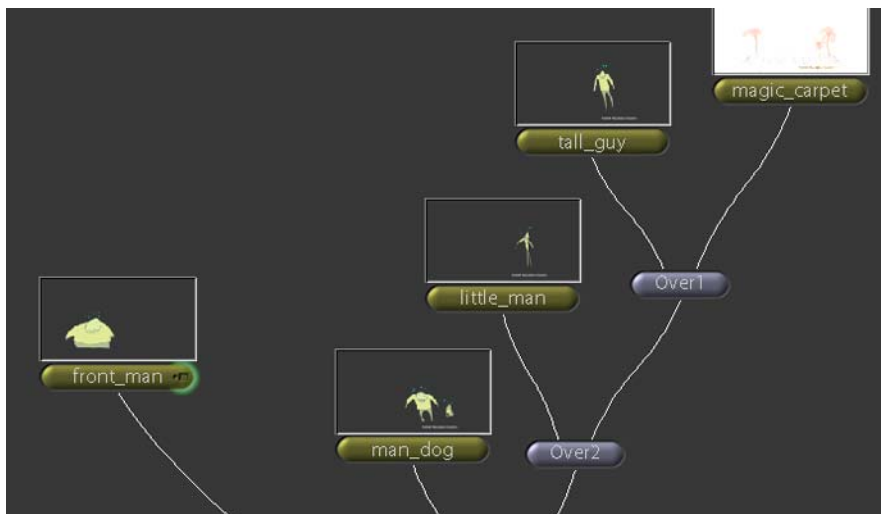
The motion blur feature lets you simulate the effect you see with moving objects in real photography. Shake relies on changes in position or scaling created with one of the Transform nodes—*Move2D*, *Move3D*, *CornerPin*, and so on—to generate motion blur for an element.



In this example, you'll open the “magic carpet” script and add motion blur to one of the elements.

To open the “magic carpet” script:

- 1 Click the Load button at the top of the Node View, browse to the `$HOME/nreal/Tutorial_Media/Tutorial_02/scripts` directory, then open the `magic_carpet.shk` script.



- 2 Click the Globals tab and expand the motionBlur subtree.



Note: The *Move2D*, *Move3D*, *CornerPin*, *CameraShake*, *Pan*, *Rotate*, *Scale*, *Stabilize*, and *MatchMove* transform nodes also have motion blur parameters.

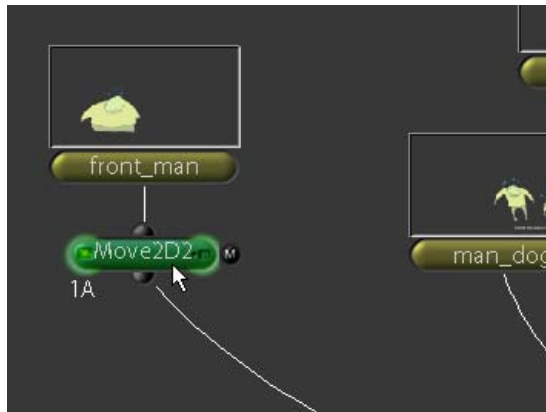
The motion blur parameter has no effect until you apply an animated transformation to an element. Shake creates the motion blur by tracking each pixel through the transformation.

Motion Blur Parameters

- *motionBlur*: Controls motion blur quality. When set to 0, the blur effect is turned off. When set to 1 or greater, the blur effect is of high quality. Lower the motionBlur value to 0.5 for acceptable quality, or to 0.1 for draft previewing.
- *shutterTiming*: Specifies the fraction of the frame exposed for the motion blur. The default setting of 0.5 indicates that only half of the frame of movement is exposed. This mimics the shutter exposure of 178 degrees out of 360 for a real film camera, rounded up to 0.5. You can also enter 178/360 to be exact. To match a video camera with a shutter setting of 1/300, enter 1/300 in the shutterTiming parameter. This resolves the setting to 0.0033 and creates a tiny amount of motion blur. When shutterTiming is set to 0, motion blur is disabled. If set to 1, the entire frame is calculated. You can also set this number higher than 1 to calculate later movement into the current frame.
- *shutterOffset*: Specifies an offset from the current frame, from which the blur effect should be calculated. The default of 0 calculates motion blur starting with the current frame. If, for example, you set the shutterOffset to -1, blur is calculated from the previous frame. This can be useful when adjusting motion blur for rotoshapes.

To create a transformation for motion blur:

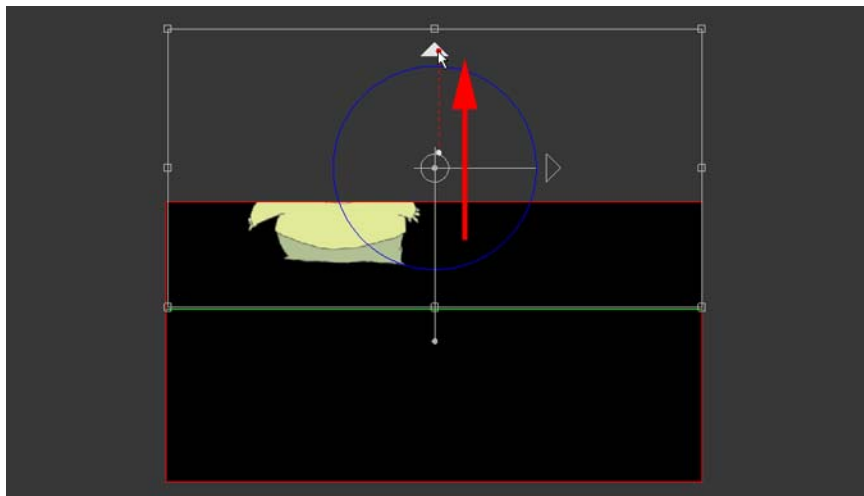
- 1 Select the *front_man* node, then insert a Transform–Move2D node.



- 2 Move the playhead to frame 25 in the Time Bar.
- 3 Enable the Autokey button in the Viewer shelf.



- 4 Move the playhead to frame 30, then drag the onscreen transform control up until the "front man" character moves out of view.



This animated transformation between frames 25 and 30 creates the movement Shake needs to generate motion blur.

- 5 Click the right side of the *Over4* node to load the final output into the Viewer.

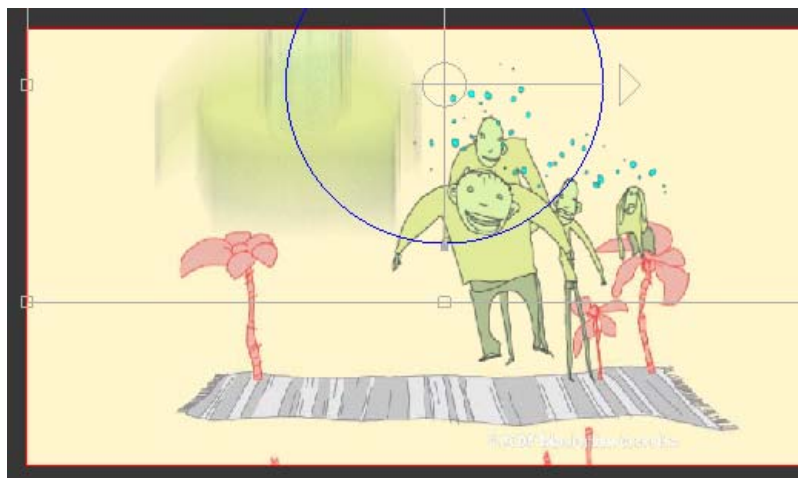


No motion blur, right? You still need to specify the motion blur settings. (See “Motion Blur Parameters” on page 67 to review the Shake motion blur settings.)

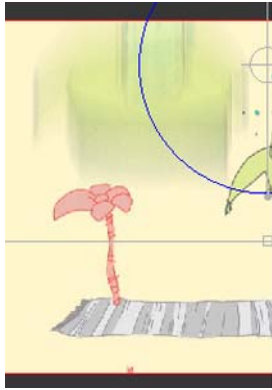
To activate the motion blur:

- 1 Scroll to the bottom of the *Move2D1* parameters and expand the motionBlur subtree.
- 2 Set motionBlur to 1.

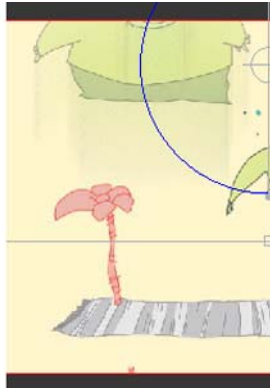
You should now see some motion blur action.



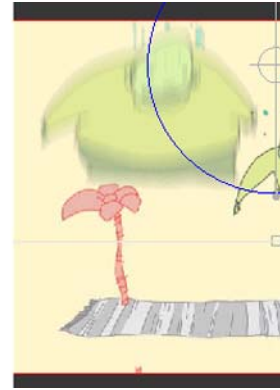
- 3 Set motionBlur to 0.25 to lower the blur quality.
- 4 Enter a few different settings for shutterTiming to test the effect of this parameter.



shutterTiming = 0.5



shutterTiming = 1.5



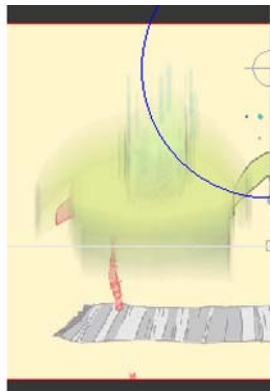
shutterTiming = 0.15

Time to settle on one of these.

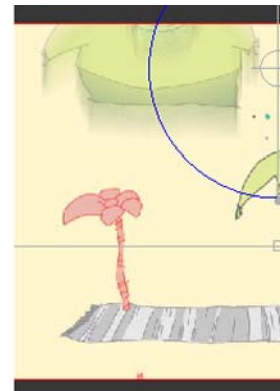
- 5 Set shutterTiming to 0.35.
- 6 Test the effect of shutterOffset by entering different values for this parameter.



shutterOffset = 0.5



shutterOffset = -2



shutterOffset = 1.25

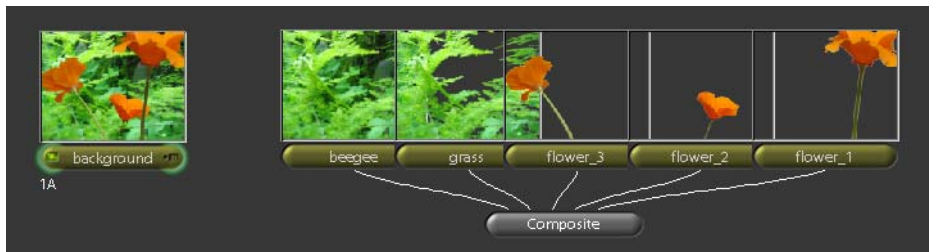
- 7 Reset shutterOffset to 0.
- 8 Click the Flipbook button to generate a preview of the completed composite.

Note: Generally, it's better to define motion blur settings for individual transform nodes. You can set motionBlur to 0 in the Globals tab to temporarily turn off all motion blur calculations, and thereby speed up your workflow. Just remember to reset the global motionBlur parameter to 1 when you need to preview the effect in the Viewer or render final output.

The first two motion blur parameters in the Globals tab are multipliers of the motion blur settings for the individual transform nodes. Entering 0 for global motionBlur, for example, internally multiplies all the motionBlur settings in the transform nodes by 0, which effectively disables all motion blur. On the other hand, entering a global motionBlur setting of 2 doubles the values. Set the global motionBlur to 1 to restore motionBlur settings on the individual nodes.

Importing Photoshop Files

Shake handles several different file formats. The *FileIn* node is the standard method for “reading in” (importing) images into the Node View. However, Photoshop files are handled a bit differently. You can use the *FileIn* node to read in Photoshop files, but if your file has multiple layers, it's better to use the import feature designed specifically for Photoshop. Give it a try.



First you'll use the *FileIn* node to read in a Photoshop file. Then you'll compare the results to the Photoshop import feature.

To import a Photoshop file using the *FileIn* node:

- 1 Create a new script by choosing New Script from the File menu.
- 2 Add an Image-*FileIn* node.
- 3 Browse to the `$HOME/nreal/Tutorial_Media/Tutorial_02/images/field_wawas` directory, then double-click the `background.psd` file.

Shake inserts the Photoshop file as a single image.

Now use the Photoshop import feature to read in the same file.

To import a Photoshop file using the Import command:

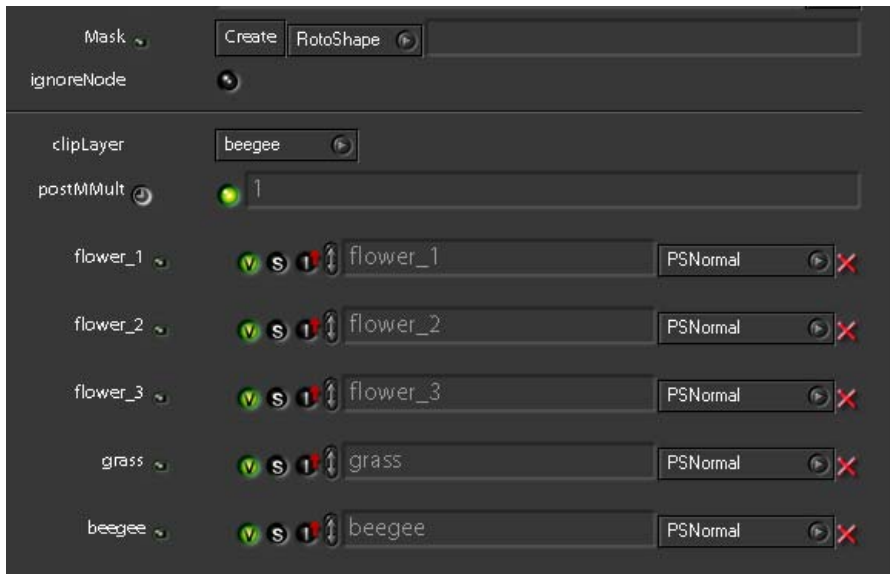
- 1 Choose Import Photoshop File from the File menu.
- 2 Browse to the same directory as before (`$HOME/nreal/Tutorial_Media/Tutorial_02/images/field_wawas`) and double-click the *background.psd* file.

This time, Shake organizes each Photoshop layer in the file as a separate image and inserts a *Composite* node to create the same result as the *background* node that you inserted.

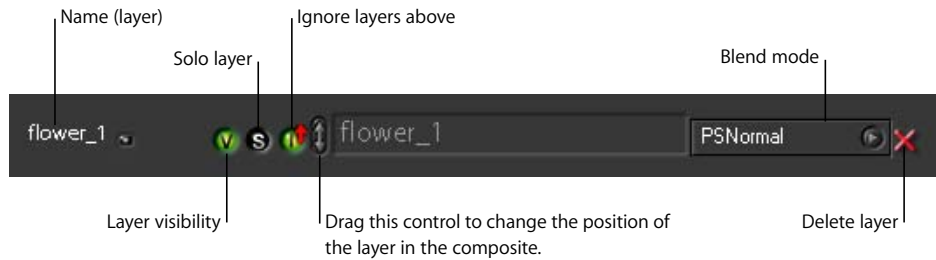
Note: The *Composite* node in this example is actually a *Layer-Multilayer* node, inserted with the Photoshop import command and renamed “Composite.”

What’s so great about this? Now you can expand the node tree and insert other elements between the Photoshop layers. You can also edit the layers in the original Photoshop file, and the changes will be updated in your Shake script.

- 3 Double-click the *Composite* node to load it into the Viewer and Parameters tab.

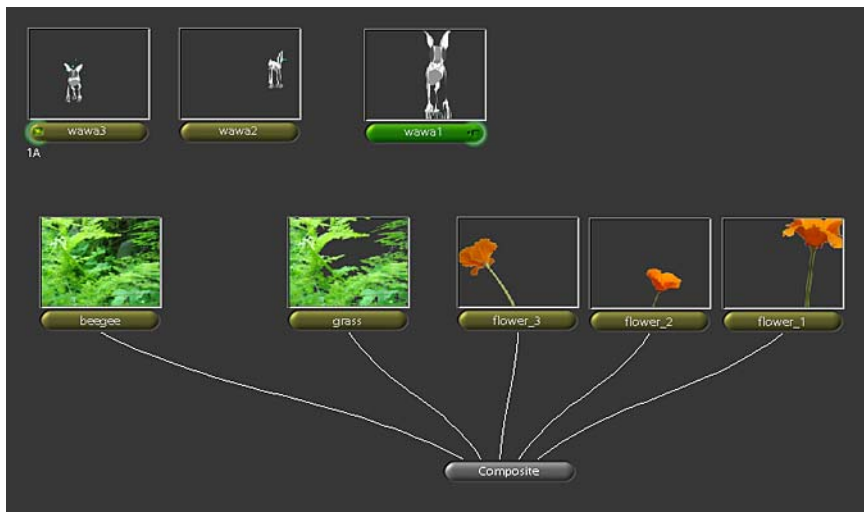


As you can see, each Photoshop layer has its own set of parameters inside the *Composite* node. The *clipLayer* parameter controls the output resolution of the *Composite* node.

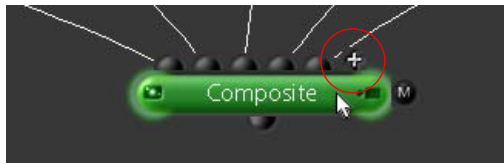


Now let's add some new elements and incorporate them into the composite.

- 4 In Node View, delete the *background* node.
- 5 Insert an *Image-FileIn* node, then browse to the `$HOME/nreal/Tutorial_Media/Tutorial_02/images/field_wawas` directory.
- 6 Shift-click *wawa1.1-30@.tif*, *wawa2.1-30@.tif*, and *wawa3.1-30@.tif* to select these image sequences, then click OK.
- 7 Arrange the nodes as shown in the following illustration.



Zoom in on the *Composite* node (press Command-middle-click or Control-middle-click and drag right). You'll see the plus sign (+) on the top-right edge of the *Composite* node.

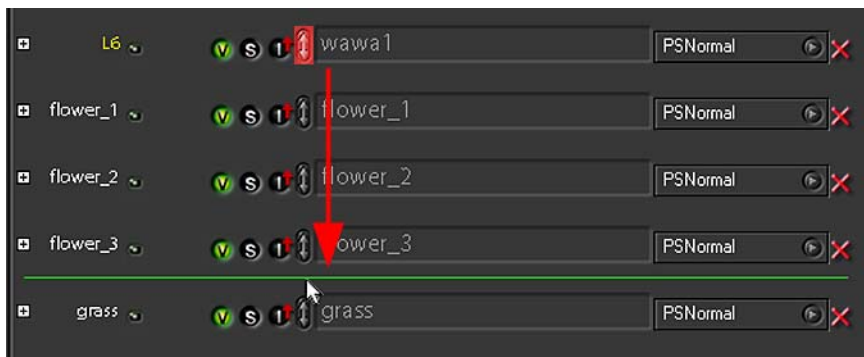


Drag any noodle to the plus sign and you'll add new inputs—in this case, new layers—to the node.

- 8 Drag a noodle from the *wawa1* node to the + sign on the *Composite* node.

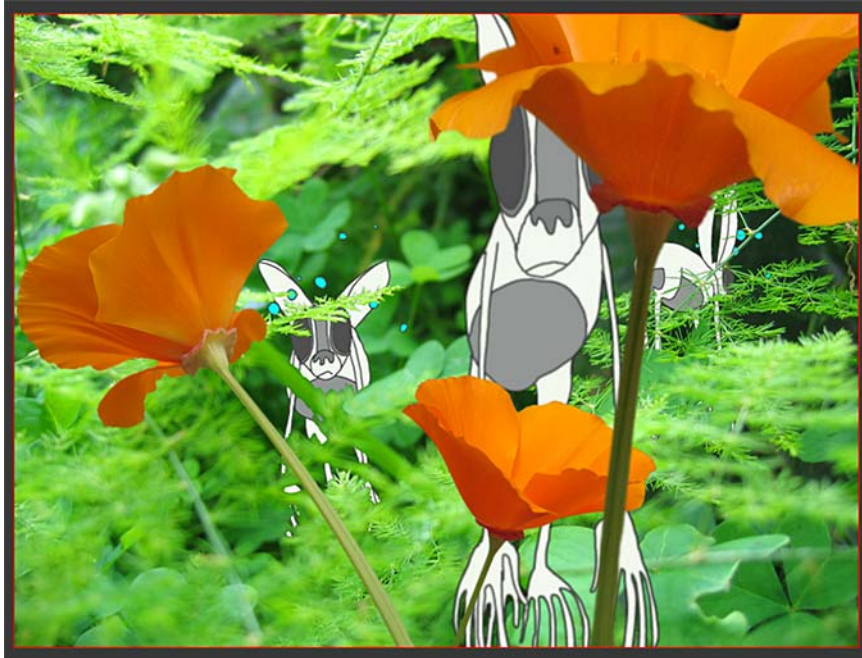


- 9 Load the *Composite* parameters.
Note that a new layer is stacked on top of the existing layers in the Photoshop file. Move it behind all the "flower" layers.
- 10 Using the Reposition control for the *wawa1* layer, drag that layer down below the *flower_3* layer.



- 11 Drag noodles from both the *wawa2* and *wawa3* nodes to the plus sign on the *Composite* node.

- 12 In the Parameters tab, use the Reposition controls to drag the *wawa2* and *wawa3* layers below the *grass* layer.



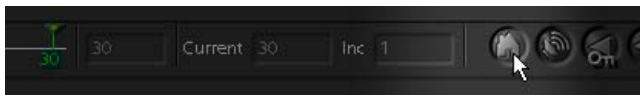
- 13 Click the Save button at the top of the Node View, and save the script as *field_wawas.shk*.

Keyframe Animation and the Curve Editor

This section continues with the script you created in the previous section. You'll add animation to create motion that doesn't exist in the original image sequences. Then, you'll use the Curve Editor to adjust the animation keyframes.

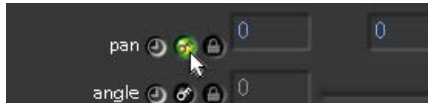
To animate the "wawa" characters:

- 1 In the Globals tab, enter 1-30 as the timeRange, then click the Home button in the Viewer playback controls to match the Time Bar to the global time range.

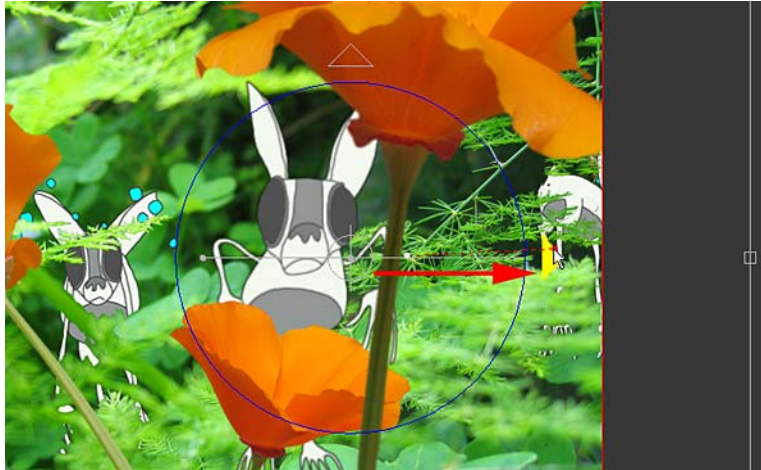


- 2 Select the *wawa2* node and insert a Transform–Move2D node.
Before completing the next step, make sure the playhead is set to frame 1.

In the *Move2D1* Parameters tab, click the AutoKey button next to the pan parameters.



- 3 Move the playhead to frame 30, then, using the onscreen transform controls, drag the *wawa2* character to the right edge of the frame.



- 4 Again, using the tranform controls of the *Move2D1* node, adjust the vertical position of the *wawa2* element at frames 10, 20, and 30 to create a “bouncing” movement. It should look like the character is hopping through the grass.

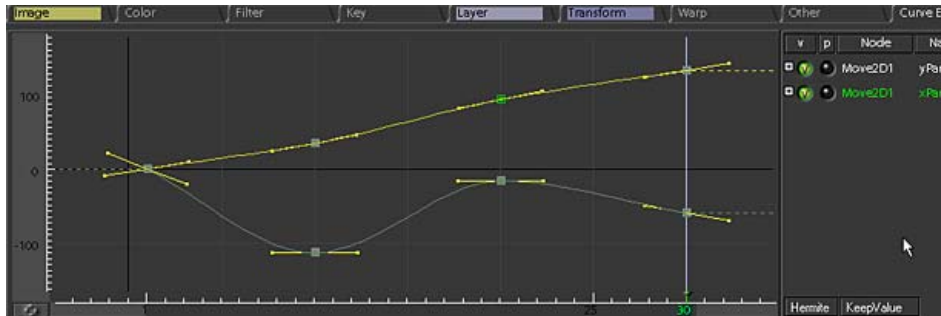
No need to be precise. The result is two animation curves, visible in the Curve Editor. The first curve (xPan) makes the “wawa” move from left to right. The second curve (yPan) creates the up-and-down movement to make the character look like it’s hopping. In a moment, you’ll edit the animation in the Curve Editor.

- 5 In the *Move2D1* Parameters tab, click the small clock button next to the pan parameters.



This is the Load Curve button. A red checkmark over the button indicates that the *Move2D1* pan parameter is loaded in the Curve Editor. Usually the checkmark appears whenever you activate AutoKey for a parameter. In this case, it didn’t because there are more parameters in the pan subtree.

- Click the Curve Editor 2 tab (in the Tool tabs) to view the animation curves.



Note: To expand the window, position the pointer in the Curve Editor and press the Space bar. Press the Space bar again to reset the Curve Editor window.

Here you'll see the curves for the two pan parameters that you animated: xPan and yPan. The points on the curves indicate the places where you set keyframes in the Time Bar. To the right of the curves, you'll see the individual curve channels.

- Click the Visibility button for the yPan parameter to hide this curve.

	v	p	Node	Name	Val	Type	Cycle
			Move2D1	yPan	-37	Hermite	KeepValue
			Move2D1	xPan	132	Hermite	KeepValue

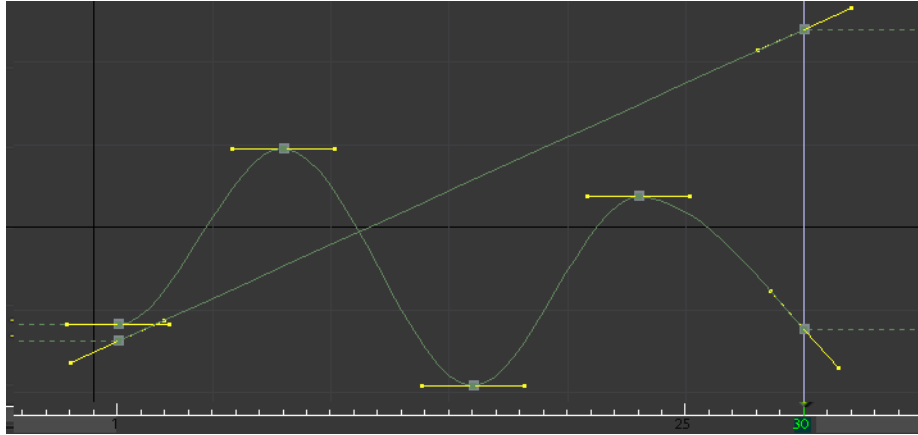
Notice that the xPan curve basically follows a linear path. Yours might look different, but in the example shown here, the xPan was animated so that *wawa2* moves from left to right as it "hops." As you can see, the xPan curve actually has more keyframes than it needs to create that linear movement.

- Drag a selection box around all the keyframes—excluding the first and last keyframes—on the xPan curve. Then, press Delete (near the Page Up/Page Down keys) or Backspace to remove the extra keyframes.



- Click the Visibility button for the yPan parameter to display this curve again.

- 10 Drag the yPan keyframes until your curves look similar to the following illustration.



Now make use of what you've already learned about motion blur.

- 11 In the *Move2D1* parameters, expand the motionBlur subtree, set motionBlur to 1, and shutterTiming to 1.25.

If you move the playhead, you'll see motion blur on the *wawa2* character. Note that the blur effect disappears at frame 30. You can create an offset to fix this.

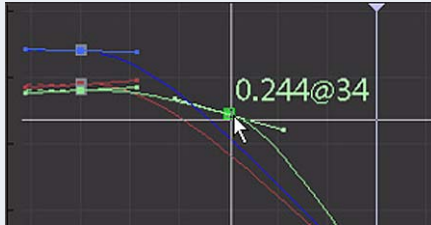
- 12 In the motionBlur subtree, set shutterOffset to -1.

If you need practice on animation and the Curve Editor, add another *Move2D* node for *wawa3* and animate that, as well.

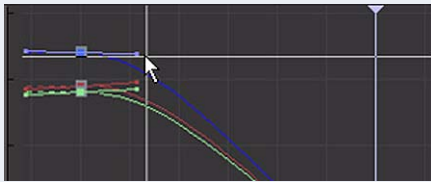
- 13 Before you continue, click the Save button at the top of the Node View to save the changes you've made.

Curve Editing Tips

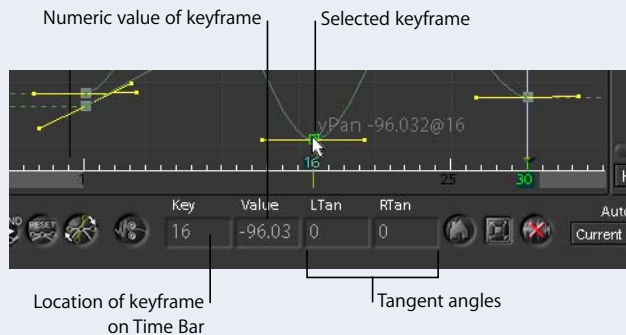
- To move a keyframe, click the keyframe and drag.
- To insert a keyframe, Shift-click a curve segment. You can also enter a value at the desired frame in the Val field of the Curve Editor.



- To remove a keyframe, select the keyframe and press Delete (near the Page Up/ Page Down keys) or Backspace.
- To adjust the tangents, click and drag a tangent handle.
A “flatter” tangent creates a smoother ease-in/ease-out effect for the animation at that point in the curve.



- To break the tangents of a curve, Control-click a tangent handle.
- To restore broken tangents, Shift-click a tangent handle.
- To perform a precise edit on a keyframe, select its point on the curve, then manually enter the frame number and value in the Key and Value fields at the bottom of the Curve Editor.



Color Correction

You've got a nice composite with the animated "wawas." Now it's time to blend the elements using color correction. First, you need to match the base colors of the animation to the photographic layers. Second, you want to vary the luminosity of the characters to mimic the lighting from the background.



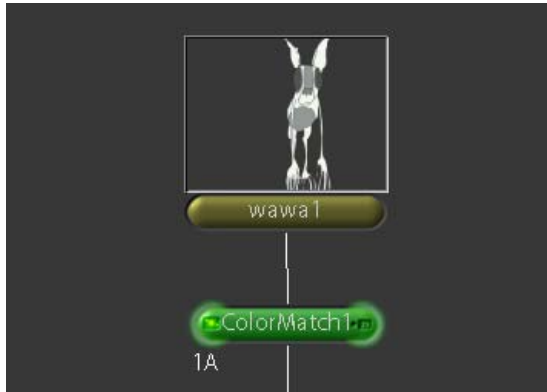
Many color-correction nodes reflect Shake's command-line heritage by applying color corrections as single-function mathematical operations—*Add*, *Brightness*, *Gamma*, and *Multiply* all perform single-function color corrections. Other nodes work as master color-correction nodes by applying the combined result of multiple color functions. The *ColorMatch* node falls in the latter category.

In this next example, you'll use the *ColorMatch* node to match the base colors of one of the animated characters to the photographic background. Then you'll apply clones of this color correction to the other characters, so that any changes you make to the original *ColorMatch* node will ripple down to the other characters.

To color correct *wawa1*:

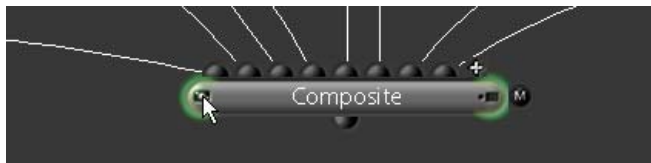
- 1 In Node View, select the *wawa1* node.
- 2 Insert a Color-*ColorMatch* node (choose *ColorMatch* from the Color command tab).

A *ColorMatch* node appears after the *wawa1* node.



ColorMatch1 is already loaded into parameters, but we need to see the result of the composite to match our character to the background.

- 3 Click the left end of the *Composite* node to load its output into the Viewer.



In the *ColorMatch* parameters, you'll see controls that let you specify the low, mid, and high colors of the image (the "Source") that you want to color correct. Typically, the low color is black or the darkest color in the image. The high color is white or the lightest color. The mid color is neutral gray or a value between the low and high values you've selected.



Choosing the low, mid, and high values from the source image you want to color correct

The *ColorMatch* Parameters tab also contains controls for colors you want to match (the “Dest” or destination colors), which are also low, mid, and high. When you first insert the *ColorMatch* node, the color selections for Source and Dest will be the same.

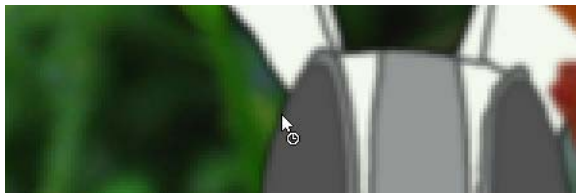


Choosing the low, mid, and high values from the destination image that the source image will match

As you adjust the Source and Dest colors, the *ColorMatch* node will color correct the source image to conform to similar colors in the destination image. This is a good way to match the base colors of your animated elements to the background.

- 4 Zoom into the Viewer (press + a few times) to get a closer view of the *wawa1* character.
- 5 In the *ColorMatch* parameters, click the lowSource color control and click the black outline of the “wawa.”

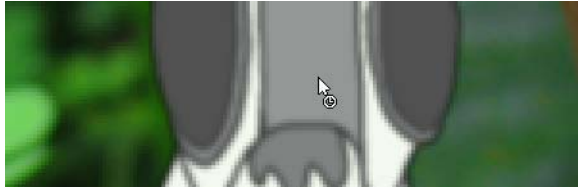
Try to get the darkest pixel you can find on the outline.



- 6 Click to select the highSource color control, then drag inside of the white area of wawa's ear.



- 7 Click to select the midSource color control, then drag inside a gray area between the wawa's nose. This should be the most neutral area between the low and high values you've selected.



You won't see a difference in the Viewer because you haven't specified the Destination colors yet. As you continue with the next steps, notice how the image changes with each of your selections.

- 8 Click to select the lowDest color control, then drag inside the darkest area you see in the background.









- 9 Click to select the highDest color control, then drag inside the brightest area from the background.



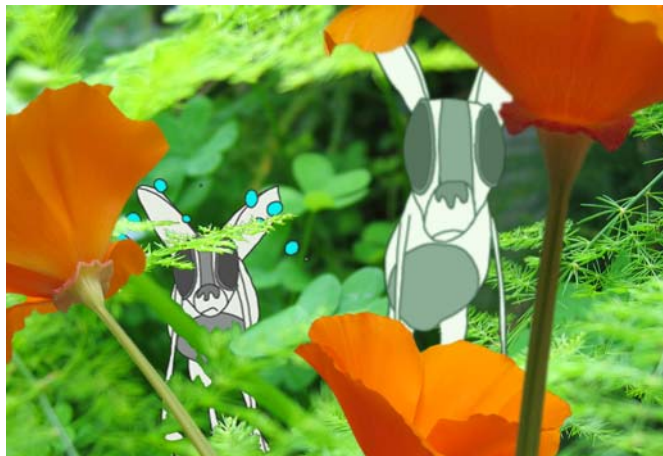
- 10 Click the midDest color control, then drag inside a neutral area or a midrange value between the lightest and darkest areas in the background.



Your *ColorMatch* settings will look similar to this:

lowSource		0	0	0
lowDest		0.1411	0.2078	0.145
midSource		0.5	0.5	0.5
midDest		0.43	0.6065	0.4718
highSource		1	1	1
highDest		0.8666	1	0.8117

- 11 Click the Home button in the Viewer shelf to see the entire image.



The image doesn't look quite matched. So let's make adjustments to the color selections. You can do this with precise channel sliders, found in the subtrees beneath the color controls. You can also drag over the color control while pressing keystroke modifiers to focus adjustments to specific values.

To adjust color selections with subtree sliders:

- 1 Expand the first subtree beneath the lowDest color control.



Beneath the color control, you'll see a slider with several buttons—R, G, B, H, S, V, T, M, and L. The slider acts like a master control to adjust a specific color property: red (R), green (G), blue (B), hue (H), saturation (S), value (V), temperature (T), magenta (M), or luminosity (L).

- 2 Make sure the Luminosity (L) button is selected, then move the slider left to change the Luminosity setting to 0.1.
- 3 Expand the first and second subtrees under highDest, to show the channel sliders beneath the property buttons. Set the rHighD slider to 0.85, and the gHighD slider to 0.98.



This is definitely an improvement, and adds little more contrast, but the green color is still too strong. Let's adjust the midrange color. This time, instead of the color sliders, you'll use the keystroke modifiers.

To adjust colors with the keystroke modifiers:

- 1 Hold down the G key on your keyboard while dragging over the midDest color control, until the rgb channels are set to 0.4, 0.5, and 0.4.



- 2 Using the keystroke modifiers or the color sliders, continue to make adjustments to the low, mid, and high destination values until you get a result that you like.

Keystroke Modifiers for Color Controls

Hold down one of the following modifier keys while dragging inside a color control to adjust the color according to a specific property.

R = Red, G = Green, B = Blue, O= Offset, H=Hue, S=Saturation, V=Value, T=Temperature, C=Cyan, M=Magenta, Y=Yellow, and L=Luminance.

You should have a fairly good match between the background and *wawa1*. Now apply that color correction to the other characters.

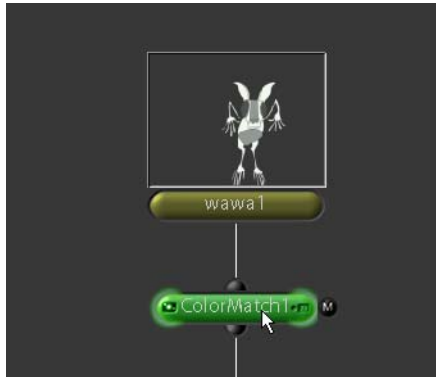


You could copy and paste the node to apply it to the other characters. However, if you decide to make changes, you'll need to update every instance of the *ColorMatch* node.

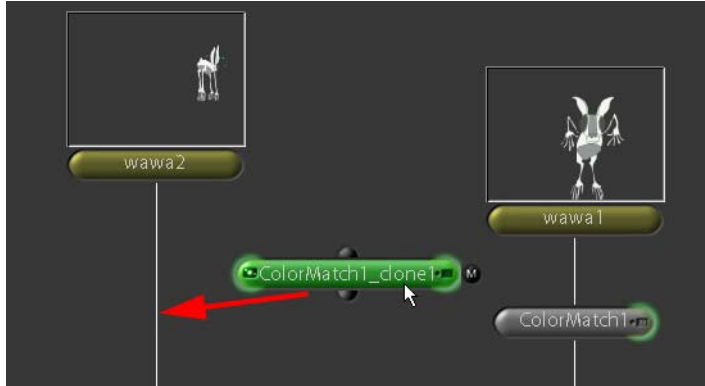
Since you have multiple characters—and they’re composited at different places in the node tree, a better method is to copy and *clone* the *ColorMatch* node. Then, when you adjust the original color correction, your changes will apply to the base colors of the other characters, also.

To clone the color correction node:

- 1 Select the *ColorMatch1* node and press Command-C or Control-C to copy it.

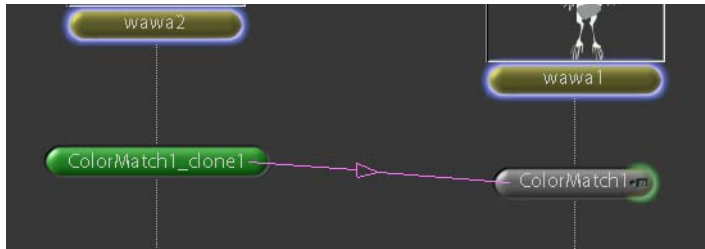


- 2 Press Shift-Command-V or Shift-Control-V to paste a copy of node as a clone of *ColorMatch1*.

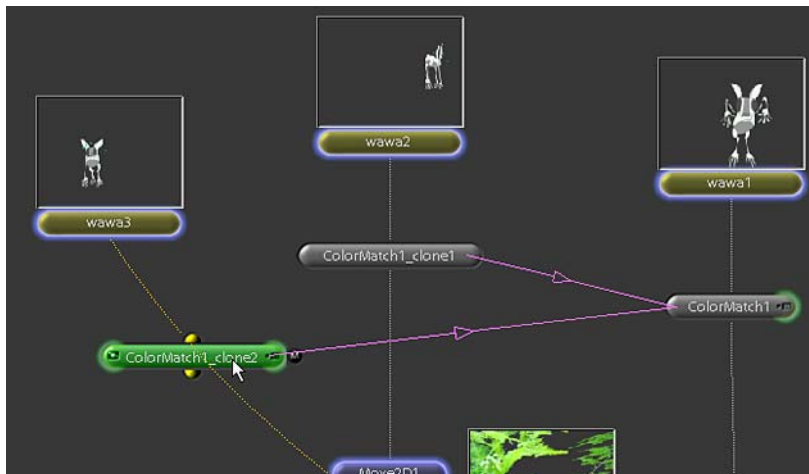


- 3 Drag the copy over the noodle that connects the *wawa2* node to the *Composite* node. There! You have a new node called *ColorMatch1_clone*. If you need more proof that it’s a clone—and a visual reminder of the node it’s linked to—turn on the enhanced Node View.
- 4 Right-click in the Node View, then choose Enhanced Node View from the shortcut menu (or press Control-E).

You see a connection between the original *ColorMatch* node and the clone, with an arrow pointing toward the parent.



- 5 Press Shift-Control-V again to paste another *ColorMatch1* clone, and drag it to the noodle under *wawa3*.
- 6 Press Control-E to turn off enhanced Node View.



Changes in *ColorMatch1* will be applied also to *ColorMatch1_clone1*, and any other clones of *ColorMatch1*—as long as you do not change the parameters of the clones. If, for example, you manually adjust the *ColorMatch1_clone3* parameters, they will not update with future changes from *ColorMatch1*.

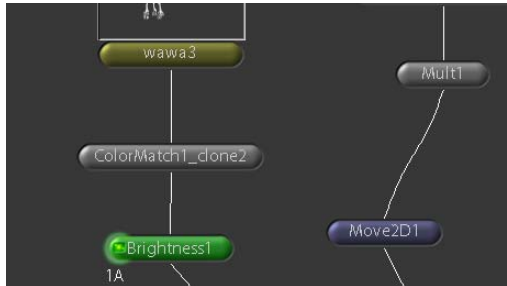
Note: When you create a clone from a node, Shake links all parameters of the clone to its parent. When you change individual parameters in the clone, this breaks the link to the parent for the changed parameters only; any unchanged parameters remain linked to the parent. The magenta connection line appears in Enhanced Node View as long as at least one parameter is linked between clone and parent.

Extended View also includes other display changes to give you a better overview of the current script and the relationships between the nodes. For more information, see Chapter 7, “Using the Node View” in the *Shake 4 User Manual*.

Now that you've color corrected the base colors of the characters, you can add other nodes to simulate lighting for their positions at various distances from the camera.

To adjust brightness for *wawa3*:

- 1 Select the *wawa3* node and insert a *Color-Brightness* node.
- 2 In the *Brightness1* parameters, set the value slider to 0.88.



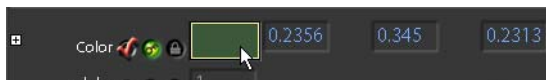
The final character, *wawa2*, is moving into a shadowed area of the background. Use the Mult color-correction node and animate its settings to darken the character as it moves out of frame.

To animate the color correction for *wawa2*:

- 1 Select the *wawa2* node and insert a *Color-Mult* node (not *MMult*).
- 2 Drag the playhead to frame 15.
- 3 In the *Mult1* node parameters, click the *AutoKey* button next to the *Mult* color control. This sets the first keyframe at frame 15.



- 4 Drag the playhead to frame 30.
- 5 While holding down the L key, drag over the *Mult1* color control, until the RGB values look similar to this:



It doesn't need to be exactly the same as the illustration, but the color correction on *wawa2*, should match the dark area in the background. The goal is to animate this color correction so that the wawa appears to be retreating into the shadows.



You're finished with the field wawas! Again, good work!

- 6 Click the Save button to save your changes to the file on disk.
- 7 Click the Flipbook button to preview the completed composite.

This tutorial demonstrates different methods for creating “real” and simulated depth in your composites. You’ll start with Z channels and filtering options. Then you’ll work with the MultiPlane node.



Tutorial Summary

- Simulated depth and 3D compositing
- Working with Z channels
- Creating composites with *ZCompose* node
- Color correcting premultiplied images
- Fading with distance
- 3D compositing and the *MultiPlane* node
- Animating a *MultiPlane* camera
- Importing camera and animation data

Simulated Depth and 3D Compositing

In this tutorial, you'll use various methods to create the illusion of depth in composites. You can simulate 3D transformations by positioning, scaling, and rotating the images in the composite. Shake can also perform Z channel operations to filter and mask elements, according to the perceived distance from the "camera" or viewing plane.

The *MultiPlane* node provides another method by compositing elements in 3D space (with x, y, and z axes). Animated camera moves simulate the change in depth and perspective. You can also import animation data from third-party software, and utilize the "camera" and point cloud data in the *MultiPlane* 3D compositing space.

The images for this tutorial were produced by Adonic Film, Los Angeles.

Setting Up the Depth Composite

Start by reading in the images for the first project. You'll use these images to review basic information about compositing for simulated depth.

To read in the images for a Z channel composite:

- 1 In a new Shake script, insert an Image-FileIn node.
- 2 When the File Browser appears, navigate to `$HOME/nreal/Tutorial_Media/Tutorial_03/images/depth`.

Shift-click to select *balloons.iff*, *city_bg.jpg*, and *city_bg_depth.jpg*. Then click OK.

You'll see the following nodes in the Node View. The *balloons.iff* image has the Z channel embedded in the file.



Launching Shake Projects From Terminal

Here's something you can try to speed up your workflow in Shake: Open a Terminal window and enter simple commands that perform the compositing functions before you launch the Shake interface.

The following steps provide an alternative way to read images into Shake:

- 1 Open Terminal, found in the *Applications/Utilities* folder.
- 2 In the Terminal window, type the following:

```
cd nreal/Tutorial_Media/Tutorial_03/images/depth
```

The “cd” command tells Terminal to change directories according to the specified path.

- 3 At the Terminal prompt, type:

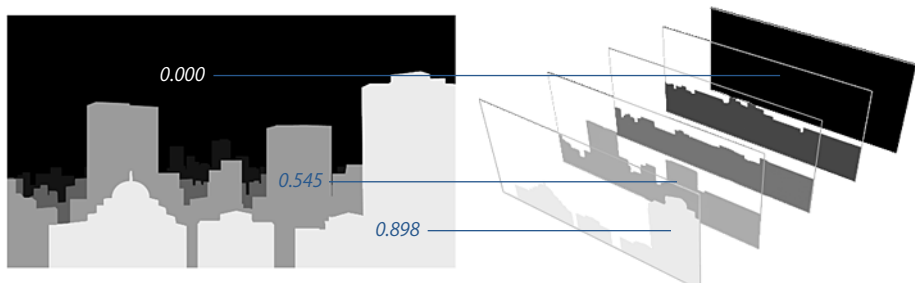
```
shake balloons.iff city_bg.jpg city_bg_depth.jpg -gui
```

- 4 Press Return.

The “shake” command tells Terminal to read in three image files, and the “-gui” command at the end of the line prompts Terminal to open the Shake graphical user interface.

Working With Z Channels

Most compositing depends on the information in the R, G, B, and alpha channels to create a final composite. An optional channel, called the *Z channel*, stores depth information as a grayscale image. Unlike the alpha channel or a roto shape, the Z channel represents perceived distances from the viewing plane and image masks in varying degrees, allowing elements to appear in front of *and* behind other elements in the composite. Lighter values are interpreted as being closer to the viewing plane. Progressively darker values indicate increased distance from the viewing plane.



As shown in the illustration above, Shake orders these pixels and outputs the higher values when it layers the images for the composite. Z channels are usually rendered from 3D animation software, where each pixel has a known distance from the camera.

Note: You may need to make adjustments—inverting the grayscale image, for example—to make the Z channel from your animation software work in a Shake composite. This is because there are different methods for processing depth information and your software may not match the method Shake uses to handle Z channel data.

A Z channel can be rendered into the same file as the RGBA channels. You can also generate and save a Z channel file—either with 3D animation software or a paint program—and then use the Shake *Reorder* and *Copy* nodes to incorporate the Z channel into an image. This latter method is useful when working with live-action plates and other elements that do not have a rendered Z channel.

These images show the Z channels for the images in the first example of this tutorial:



balloons.iff includes five channels: RGBA (left) and Z (right).



city_bg.jpg include three channels: RGB.

The hand-painted *city_bg_depth.jpg* will be added to *city_bg.jpg* as a Z channel.

The Z channel for *balloons.iff* was rendered from the original 3D scene file that created that image, and is stored as one of the channels in the file. The Z channel for the city background was hand-painted in an image editor. In this tutorial, you'll incorporate this separate image as the Z channel for the background.

Displaying Z Channels

The Z channel—or lack of it—in the background image is a problem because some of the balloons and dirigibles should appear in front of the buildings, while others should appear behind buildings.



Background image, no Z channel



Background image, with Z channel mask

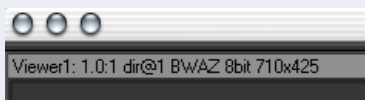
This background is a 3D image. You could render a Z channel for it, when you have the animation file. If the background were a live-action plate, however, that wouldn't be an option—not without recreating the entire scene in 3D.

You could create a bunch of roto shape masks to handle the overlapping elements, but just look at all those balloons and buildings. Do you really want to roto them? No. Fortunately, there's a much easier solution. You can build your own the Z channel using a hand-painted mask of the background image, then combine it with the existing Z channel in the balloon image.

When you load an image containing a Z channel into the Viewer, “RGBAZ” appears in the Viewer title bar. To display the Z channel in the Viewer, you must activate a Viewer script.

Additional Viewing and Framing Tips

- With the pointer over the Viewer, press the Home key to reset an image to a 1:1 view ratio. Any visual artifacts caused by a non-integer zoom are removed.
- Press + / – (near the Delete / Backspace key) to zoom the image. The location of the pointer determines the point of zoom.
- Shift-F ensures that the Viewer title bar is visible. The Viewer title bar displays information about the image.



For example, “BWAZ” in the Viewer title bar tells you that you are looking at an 8-bit black-and-white image with alpha and Z channels.

To view the Z channel of an image:

- 1 Load the *balloons* node into the Viewer.
- 2 Press and hold the Viewer Script button, then choose the View Z button from the pop-up menu.



The Viewer displays the Z channel embedded in the *balloons.iff* file.

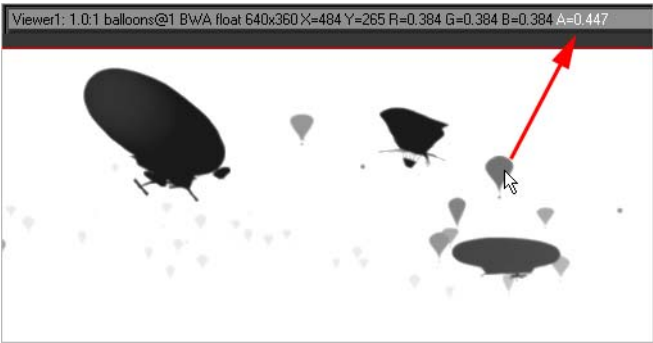
- 3 To see the numerical Z values, right-click the View Z button, then choose Load Viewer Script Controls into Parameters2 Tab from the shortcut menu.
- 4 In the Parameters2 tab, set the floatZinA parameter to Original or Distance.



The parameters for the View Z script also include display options to normalize Z values for Maya or 3ds Max files, and to adjust the depth range.

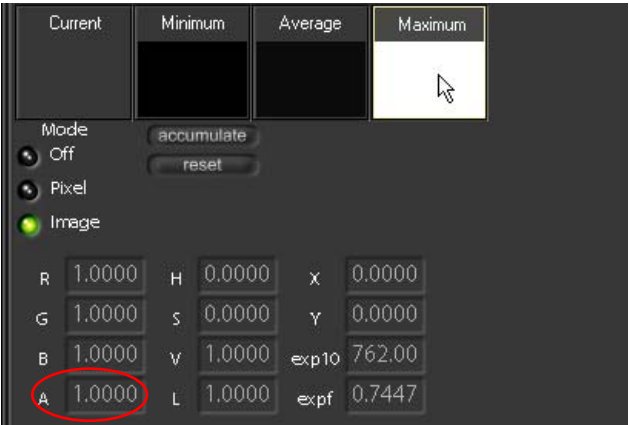
- 5 In the Viewer shelf, click the View Channel button to toggle to alpha channel view. The Viewer displays an inverted view of the Z channel for *balloons.iff*.

The real Z values are placed into the alpha channel for analysis. The inverted display is a result of the Alpha channel view.



- 6 Click a pixel or drag over the image to display the Z distance value at the top of the Viewer, shown as the “alpha” value.

When the View Z script is active, you can use the Pixel Analyzer to get the minimum and maximum values in terms of distance.
- 7 Click the Pixel Analyzer tab (in the row of tabs beneath the Node View).
The Pixel Analyzer opens.
- 8 Set Mode to Image, to analyze all displayed pixels, then click the Maximum color control.



The maximum distance value is displayed in the Alpha channel field. Click any of the other Value Range options to change the display range (for example, 0-255, 0-1023, and so on).

- 9 Click the Minimum color control to display the minimum distance value in the Alpha channel field.

Some software processes Z channel data differently than Shake. If you're using 3ds Max, for example, you'll see negative values in the Z channel of the rendered image. If you're using a Maya-rendered image, you'll see values calculated as $-1/\text{distance}$. Setting the `floatZinA` parameter to Distance returns the actual distance from the camera. You can also choose to "normalize" the Z channel data for either Maya or 3ds Max by selecting the appropriate item in the `zNormalize` parameter.

- 10 Click the View Z button to disable the Viewer script, then press C to reset the Viewer to display all color channels.

Creating Composites With ZCompose

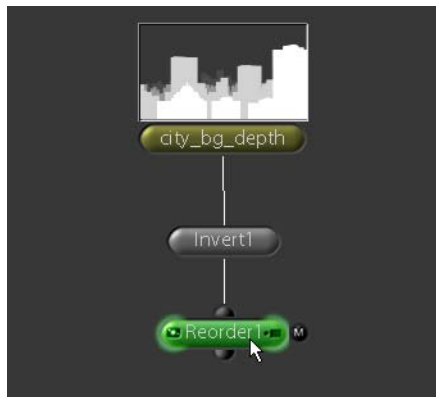
In the next example, use the *ZCompose* node to composite these elements together. The *ZCompose* node is designed to consider five channels—R, G, B, A, and Z—when layering images. A little preparation is required to convert the painted Z channel (*city_bg_depth.jpg*) into a proper Z channel for the background image.

To prepare the Z channel for the *city_bg_depth* image:

- 1 In the Node View, select the *city_bg_depth* node.
- 2 Apply a Color-Invert node.

Why the Invert node? The Z data must be inverted for one of the images, because the Z channels don't match. In the *city_bg_depth* image, the pixel values go from light to dark as distance increases, while in *balloons.iff*, the inverse is true. Therefore, you need to invert the *city_bg_depth* image to match the Z channel of the other image.

- 3 With the *Invert1* node selected, add a Color-Reorder node.

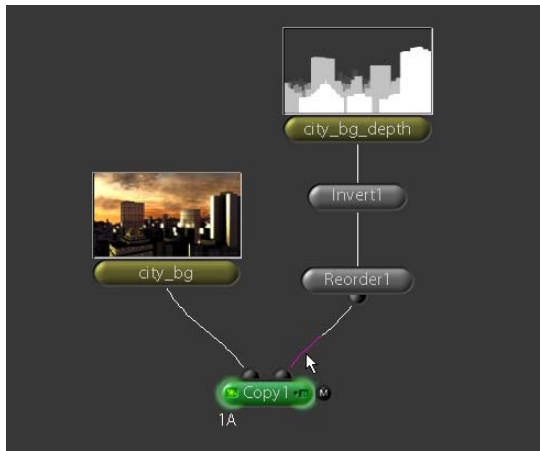


The *Reorder* node will create a Z channel for this image by copying another channel (red) to the Z channel position.

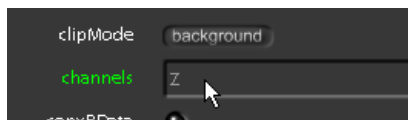
- 4 In the Reorder parameters, enter “rgbar” in the channels value field. This string specifies that red goes to the red channel, green to the green channel, blue to the blue channel, alpha to the alpha channels, and red again to the fifth channel—the Z channel.
You now have a Z channel for the background. Next, you’ll copy this Z channel to the *city_bg* image.

To copy the Z channel to the *city_bg* image:

- 1 In the Node View, select the *city_bg* node.
- 2 Attach a Layer-Copy node.
- 3 Connect the *Reorder1* node to the second input of *Copy1*.



- 4 In the *Copy1* parameters, enter “z” in the channels field. This copies the Z channel from the *Reorder1* node to the *city_bg* image.

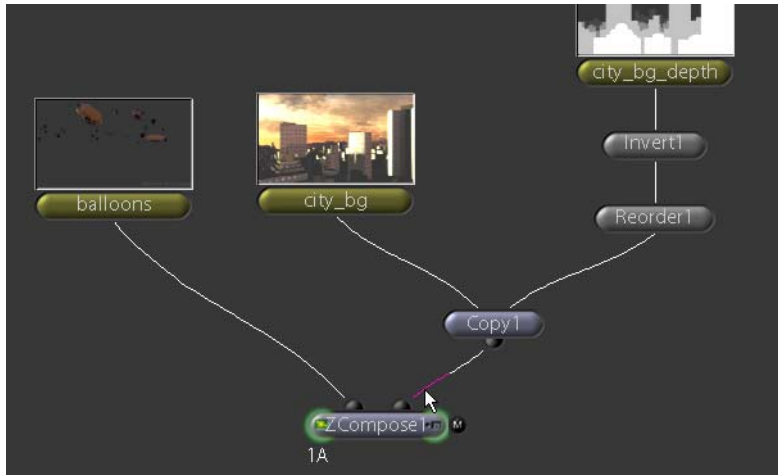


Now, with the Z channels in place, you can create the composite using a *ZCompose* node. *ZCompose* takes two images and uses the Z channel values to “stack” the RGBA pixels by comparing which image has the lower Z value. The order in which the layers are composited does not matter. Instead, the pixels with the higher Z values are composited on top and passed down the tree.

To composite *balloons* and *city_bg* using *ZCompose*:

- 1 In the Node View, select the *balloons* node.
- 2 Attach a Layer-*ZCompose* node.

- 3 Connect the *Copy1* node to the second input of *ZCompose1*.



So what's the result? Probably different than you expected. There are several places where the balloon images are not masked by the Z channel.



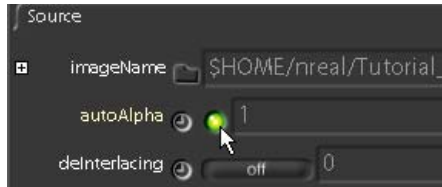
ZCompose will not work properly unless all images have an alpha channel. In this case, you need to create an alpha channel for the *city_bg.jpg* image.

To create an alpha channel for *city_bg*:

- 1 Double-click the *city_bg* node to load it into the Viewer and into the Parameters tab.
- 2 Click the View Channel button in the Viewer shelf to display the alpha channel.

The screen goes black, which indicates that the *city_bg* image does not have an alpha channel.

- 3 In the *city_bg* parameters, click the button to enable autoAlpha.



A solid white alpha channel is created for the image.

Note: As an alternative, you can also attach a *Color-SetAlpha* node after the *city_bg* node.

- 4 Click the left side of the *ZCompose* node to load the corrected composite into the Viewer.

Adjusting Z Depth

When you combine the effect of two or more Z channels, as in this example, the distance values do not always correspond to the result you want in the composite. The *Color-Mult* node (not *IMult*) includes a depth parameter that lets you “push” or “pull” the depth values. You don’t need to do this for the current example, but it’s worth noting for future reference.



To adjust the Z depth, add a *Color-Mult* node to an image, then adjust the depth slider. Give it a try and you’ll see the image forward or backward according to depth.

Color Correcting Premultiplied Images

To create a more realistic lighting effect, you can color correct the foreground objects to “reflect” the ambient colors of the background. Use the *Color-Add*, *Color-ContrastLum*, and *Color-Brightness* nodes to correct the *balloons* image.

The images in this tutorial are *premultiplied*, which means that the RGB pixel values are appropriate for the transparency indicated by the alpha channel.



Premultiplied: RGB pixel values are multiplied by the alpha pixel values and interpreted as

For example, an RGB pixel value of 0.75 multiplied by an alpha value of 1.0 (a white pixel) results in the full value of that pixel being passed to the composite. An RGB pixel value of 0.75 multiplied by an alpha value of 0.0 (a black pixel) would become zero (and result in 0 percent opacity). Gray values in the alpha result in varying levels of opacity.

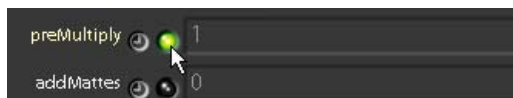


No premultiplication: RGB and alpha channels exist in the file, but there is no transparency.

For images that are not premultiplied, an alpha channel may be present but there is no transparency “baked into” the RGB channels.

When performing color corrections in Shake, premultiplication can be an issue when the correction alters the RGB channels that correspond to semi-transparent areas of the alpha channel. This can affect the layering operations in your composite.

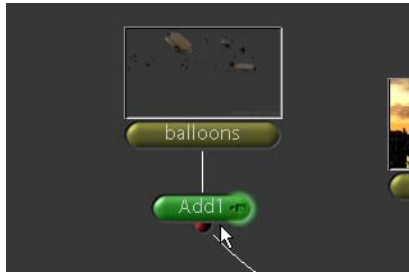
One way to deal with premultiplied images is to toggle the *preMultiply* control in the layer nodes you’re using.



Another technique is to manage premultiplied images with *MDiv* and *MMult* nodes when performing color correction.

To color correct the *balloons* image:

- 1 In the Node View, select *balloons*, then insert a Color-Add node.



- 2 In the *Add1* parameters, click the Color control to open the Color Picker.
- 3 In the Viewer, scrub over the dark orange horizon in the *city_bg* image to add an orange tint to the *balloons* image.



Everything from the balloons to the background turns bright orange—and check out the artifacts around the edges! This is obviously not good.

The entire image is brightened because you’ve color corrected a premultiplied image. The Color-Add node adds whatever color you choose to the pixel values of all channels, including the alpha. This returns incorrect pixel values—and the corresponding transparency values—when multiplied with the alpha channel.



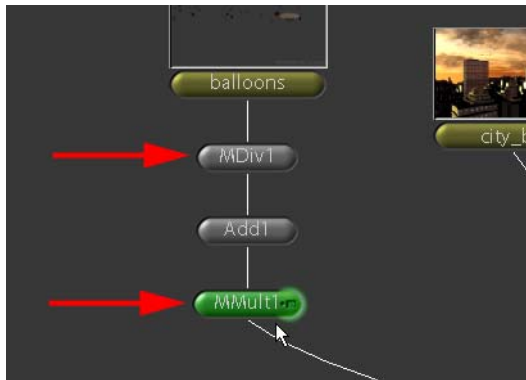
The result is a compromised alpha which generates a tinted matte, rather than the full transparency we need for the *balloons* image.

The *Add1* node applies a tint to the balloons and the sky, but doesn't affect the buildings from the background image. This is because the composite also includes Z-channel masking, which prevents color correction on the closer elements (in Z space).

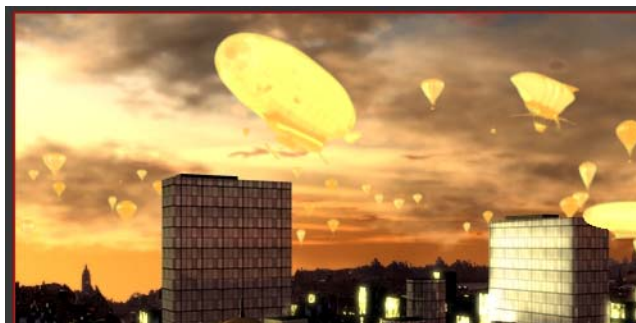
To fix this problem, you need to isolate the color-correction nodes for this branch of the node tree using *MDiv* and *MMult* nodes. *MDiv* removes the alpha channel from calculations, which prevents the alpha pixels from being color corrected. *MMult* brings the alpha back after color correction is done.

To isolate color correction for a premultiplied image:

- 1 In the Node View, select the *balloons* node.
- 2 Insert a Color-*MDiv* node.
- 3 Select the *Add1* node.
- 4 Insert a Color-*MMult* (not *Mult*) node.



If you add new color correction nodes for this element, make sure you insert them between the *MDiv1* and *MMult1* nodes.



Now the color correction affects only the *balloons* image, without compromising its alpha channel.

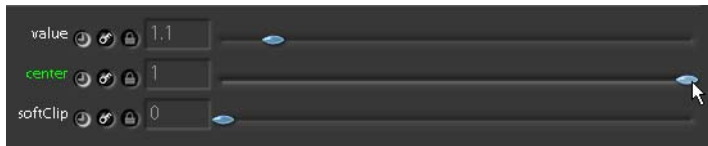
This is an extreme color change, but it helps illustrate the issue with premultiplied images. Time to tone it down a bit.

- 5 Load the *Add1* parameters (click the right side of the node), then set the RGB color values to 0.2, 0.11, and 0.07.



That's a little better, but the blacks could use more contrast.

- 6 Select the *Add1* node and insert a *Color-ContrastLum* node.
- 7 In the *ContrastLum1* parameters, drag the value slider to 1.1, then drag the center slider to 1.



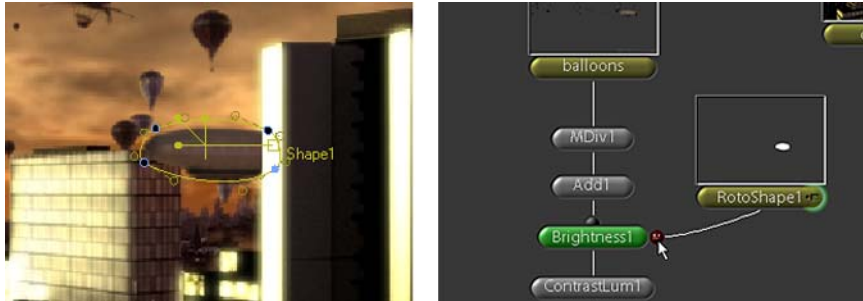
This color correction works well for most of the balloons. One exception is the zeppelin on the far right, which is too bright for the rest of the composite.



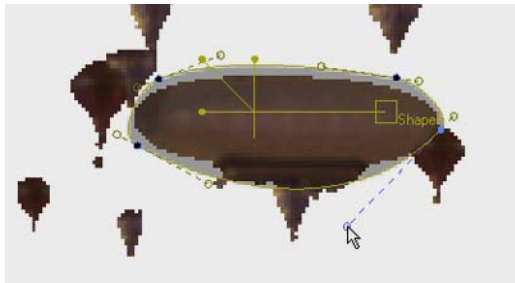
You can use a roto shape mask to limit color correction for the small zeppelin.

To limit color correction with a rotoShape mask:

- 1 Insert a *Color-Brightness* node after the *Add1* node.



- 2 Insert a *Color-RotoShape* node, then drag a noodle from *RotoShape1* to the mask input on *Brightness1*.
- 3 Click the left side of *Brightness1* to load it into the Viewer, then click the right side of *RotoShape1* to load it into the Parameters tab.
- 4 Using the *RotoShape* tools in the Viewer shelf, draw a shape around the small zeppelin.



- 5 Load the *Brightness1* parameters, then set the value slider at 0.75 to darken the zeppelin.

Now the image should work better as a composite. This is a good time to go back and adjust the color corrections to suit your own preferences.

Fading With Distance

In this section, you'll add some depth cueing by varying the color correction between the foreground and background elements. There are many ways to do this. One way is to pull a depth-based key from the *balloons* image and use it as a mask for a color-correction node. Again, there are many options for the type of color correction.



Without fading



With fading

In this example, use *Fade* node with the *DepthKey* node to allow the background to show through as distance increases, creating the illusion that some of the balloons are actually behind the clouds on the horizon.

To add a fade effect:

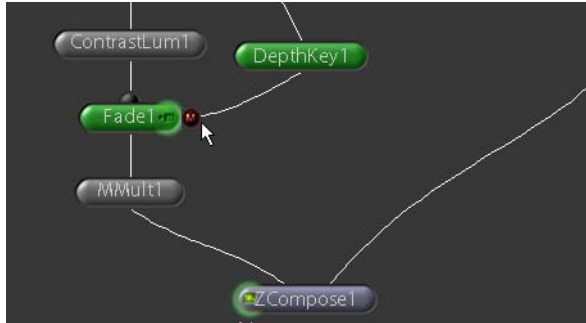
- 1 In the Node View, select the *ContrastLum1* node, then insert a *Color-Fade* node.
- 2 In the *Fade1* parameters, set the value slider to 0.3.



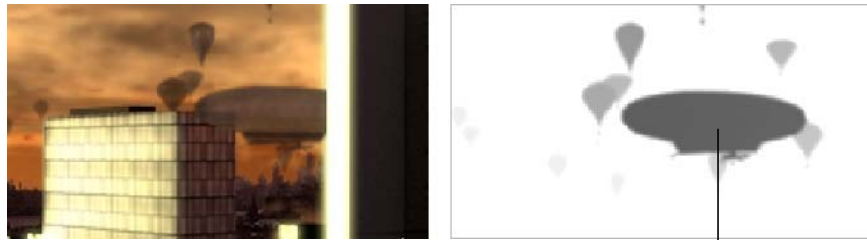
This fades all the balloons equally. Next, add a mask to vary the effect according to depth (or distance).

- 3 Select the *balloons* node.
- 4 In the Key tab, Shift-click the *DepthKey* command.
A new branch is created with the *DepthKey1* node.

- 5 Connect the *DepthKey* output to the mask input of the *Fade1* node.



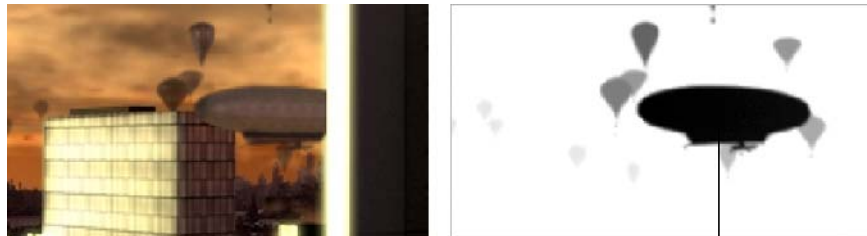
- 6 Click the left side of *ZCompose1* to load the result into the Viewer.



This part of the mask is partially transparent.

The fade effect is definitely varied, but take a close look and you'll notice that some elements are fading too soon. You need to adjust the range of the *DepthKey*.

- 7 In the *DepthKey1* parameters, set *loVal* to approximately 0.25.



After adjustment, the mask is opaque.

To verify the key, view the alpha channel of the *DepthKey1* node. You can adjust the *loVal* and *hiVal* settings to experiment with the range for the fade effect.

If you're happy with the results at this point, then you're done!



Otherwise, you can add some additional nodes to improve the composite. The *depth-DONE.shk* script, located in your `$HOME/nreal/Tutorial_Media/Tutorial_03/scripts` directory, includes these final changes.

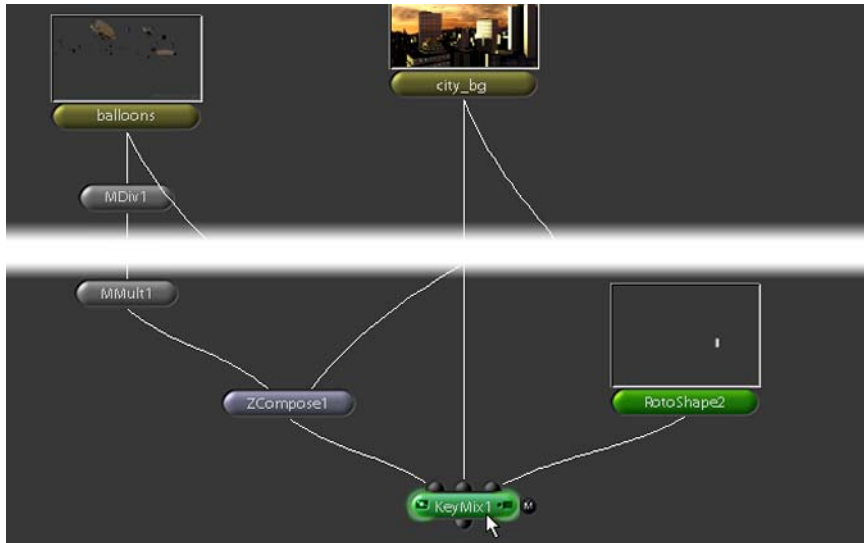
Layering Background Elements to Smooth Edges

Our little zeppelin has turned out to be problematic in the composite. Notice the hard edge between the right end of the zeppelin and the building in front of it. This happens because of the soft edge created by the reflective glow of the building.



You can create a smoother transition between the elements by layering a cropped copy of the building's edge (from the *city_bg* image) over the *balloons* image. If this were an image sequence or clip, you'd need to animate the roto shape to any movement in the clip.

This is one example of the node tree you could create for this solution:

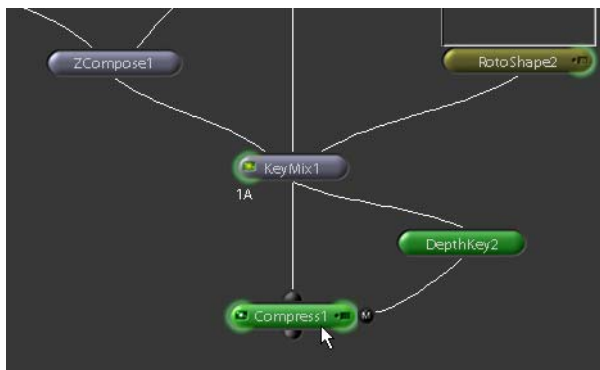


The *KeyMix1* node layers a copy of *city_bg*, which is masked with *RotoShape2*, over the finished composite from *ZCompose1*. *KeyMix* has a mask input, like all other nodes, but also has a unique input for the mask you want to combine with the other elements.

Adding Haze to the Entire Image

When you applied the *Fade* node, the result was a “fade-to-distance” effect. This works for the balloons because there’s a background for them to fade into. Not true for the city background; it will just fade to black because there’s nothing behind the *city_bg* image.

To create a similar fading effect for the city horizon, you add a *Compress* node combined with a *DepthKey* node, as shown below, to simulate haze:



This *DepthKey2* node pulls a key from the combined Z channel of *balloons* and *city_bg*. The *Compress* node uses *DepthKey2* as a mask to apply a subtle haze that increases with depth.



Original image

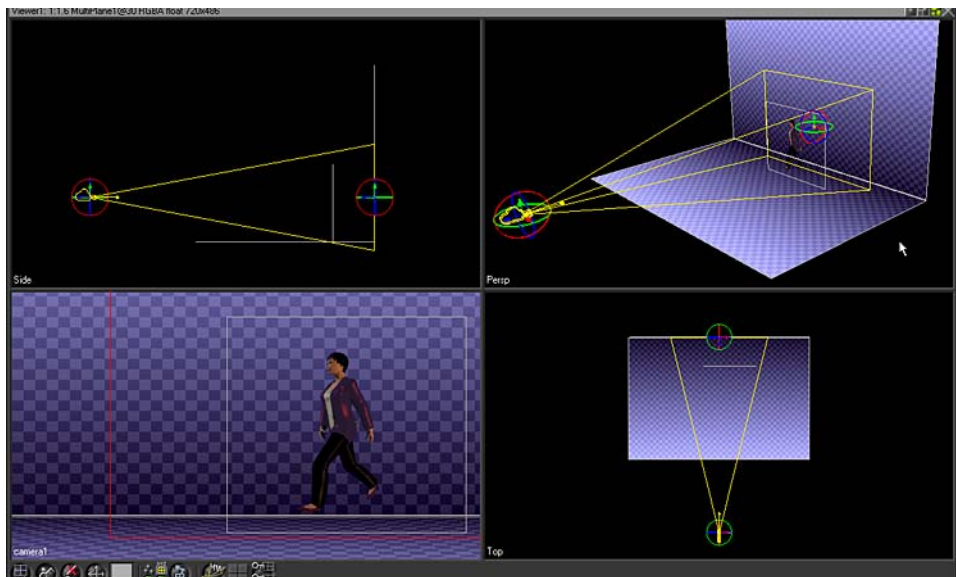


Haze added with Compress and DepthKey

Compress1 adds the hazy color. In this example, the settings are Low Color = 0.25, 0.15, 0.115, and High Color is set to 1, 1, 1.

3D Compositing With the MultiPlane Node

In the previous example, you created the illusion of depth with 2D images. This section demonstrates a simple project with the *MultiPlane* node, which composites in 3D space, with x, y, and z axes.



You'll create an environment with 2D planes, animate a camera, then composite an animated clip into the environment. Start from a new script.

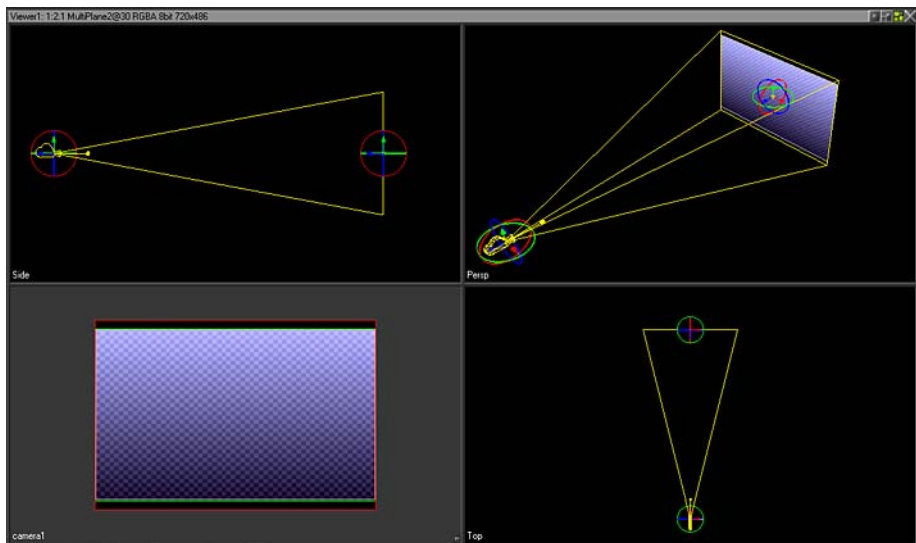
To set up the 3D scene:

- 1 Add an Image-FileIn node to the Node View.
- 2 Browse to `$HOME/nreal/Tutorial_Media/Tutorial_03/images/betty`, select the `checker_plane.jpg` image, then click OK.
- 3 In `checker_plane` parameters, click the `SFileIn` field and rename the node `wall_plane`.
- 4 This image does not have an alpha channel, so turn on the `autoAlpha` parameter.
- 5 Select the `wall_plane` node, press Control-C to copy it, then press Control-V to paste the copy.
- 6 Rename the copy of the node `floor_plane`.

These image nodes will be the two elements that create the environment.



- 7 Select the `wall_plane` node, then add a Layer-MultiPlane node to the node tree. The Viewer is now divided into four panes: Side, Perspective, Camera1, and Top.



Think of the *MultiPlane* environment as a cube enclosing all the layers connected to it. Each pane shows you one possible view into the cube.

The wireframe camera lets you adjust the position of the rendering camera, which becomes the output of the *MultiPlane* node. The Camera1 view is the default, but you can specify a different camera for output.

Each panel shows an independent view of your scene, and you can use the standard onscreen controls to pan and zoom inside them.

To navigate in the multi-pane Viewer:

- 1 To pan, hold down the middle mouse button and drag in the Side pane.

You can also hold down the Option or Alt key and drag inside the pane.

It may appear that you're moving the camera, but this only pans the editing view of the scene.

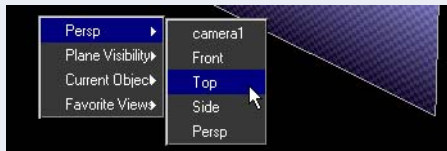
- 2 To zoom, press Control-Option or Control-Alt and drag inside the Top pane.

You can also move the pointer over a pane and press the plus (+) or minus (–) key to zoom in increments.

- 3 To rotate the perspective, press X while dragging the pointer in the Perspective pane.

Tips for Navigating in the Multi-Pane Viewer

- The standard pan and zoom controls for the Viewer work in the *MultiPlane* panes.
- In Perspective view, press X and drag the pointer to rotate the view.
- Press F (for “frame”) over a pane to fit all scene elements in the view.
- Right-click in a pane, then choose the desired 3D view from the shortcut menu: Top, Bottom, Side, Perspective, or one of the cameras associated with the scene.

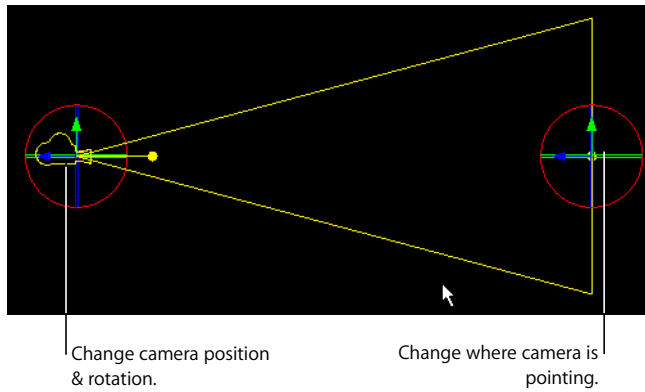


- Click the Viewer Layout button to select a different pane configuration.

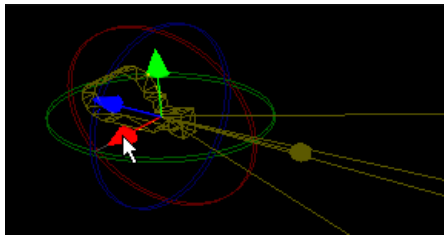


Note: The multi-pane display becomes active whenever the *MultiPlane* node is loaded into the Parameters tab, even when you explicitly load another node into the Viewer. In these cases, the displayed Camera panel is the equivalent of the standard Shake Viewer.

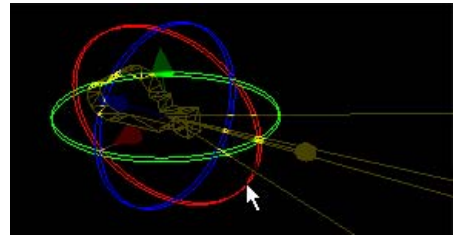
Click the camera to select it, and you'll see two sets of transform controls: one set at the center of the camera, and one set for the target, or *filmback*, of the camera.



Move the camera controls to change the camera position. Move the filmback controls to change where the camera is pointing.



Drag the pan controls to change camera position.

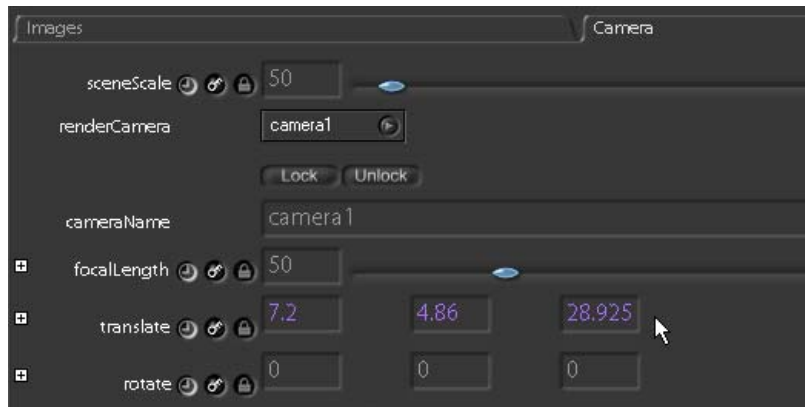


Drag the rotate controls to rotate the camera.

Green handles control translation and rotation for the Y-axis. Blue handles control the X-axis. Red handles control the Z-axis settings.

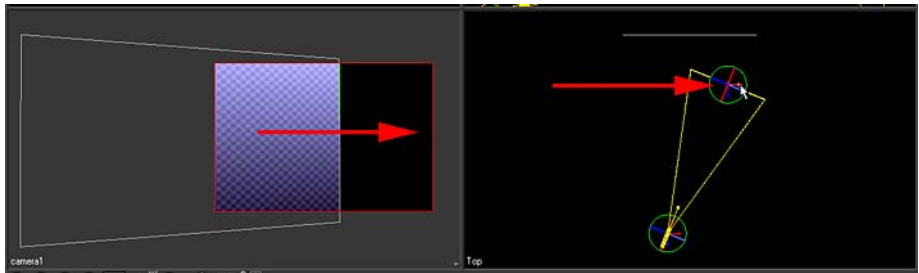
To adjust the position of the camera:

- 1 If it is not already selected, click the yellow wireframe camera in the Perspective view.
- 2 In the Parameters tab, click the Camera subtab and find the translate and rotate controls.



You can use the onscreen controls described earlier to interactively change the camera position. Or you can enter specific numeric values in the Parameters tab.

- 3 In the Viewer, drag the pan controls on the camera (the red, green, and blue arrows) to change its position. Drag the rotation controls to change its view angle.



As you drag the handles, notice that the translate (pan) and rotate parameters change in the Camera tab. The adjusted view appears in the Camera1 panel in the Viewer. Use the Side, Perspective, and Top views to make the adjustment, as it can be difficult to get precise positioning from just one or two views.

If those pesky rings make it difficult to see (and select) the translate handles, click the XYZ Angle button to remove the rotation handles from the Viewer.



Now that you've had some practice manipulating the camera, you need to reset it to the default position.

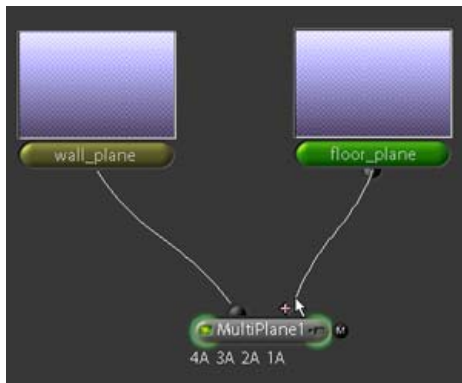
- 4 Right-click in the Parameters tab, then choose Reset All Values from the shortcut menu.

You can make this scene more interesting by adding additional elements. The *MultiPlane* node lets you add as many inputs to the node as you require.

To adjust elements in 3D compositing space:

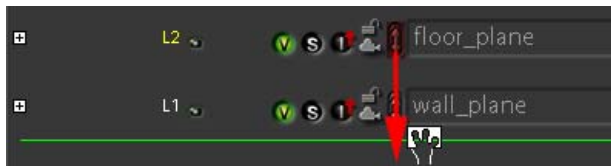
- 1 Drag a noodle from the *floor_plane* node to the + (plus) sign at the top of the *MultiPlane1* node.

This adds a new input to the node.



You've connected the *floor_plane*, which now occupies the same space as the *wall_plane* in the 3D compositing space. You'll need to rotate and move the floor to see it in the composite.

- 2 Open the Images subtab of the Parameters tab, and drag the control for layer order, so that *L2 floor_plane* appears below *L1 wall_plane* in the list of elements.



- 3 Open the L2 subtree, then, in the L2_angle parameter, set xAngle to 90.
- 4 Set the L2_scale parameter as follows: xScale = 2, yScale = 2, zScale = 2.



The floor_plane appears, but it's in the wrong position. Before you start moving anything around, it's a good idea to turn on hardware rendering and make use of the OpenGL rendering engine.

- 5 Click the Rendering button in the Viewer shelf to turn on hardware rendering.

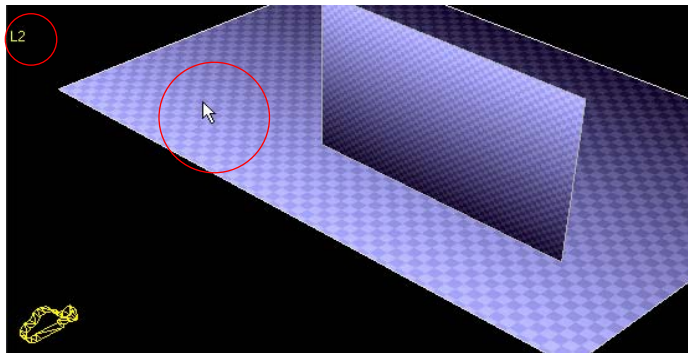


This will speed up the display response.

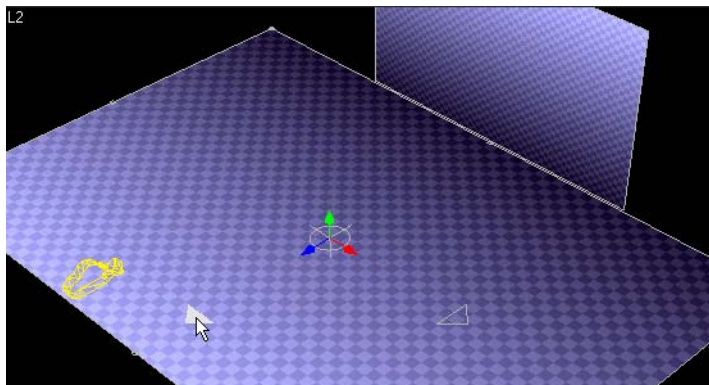
- 6 Click in an empty space inside the Perspective (Persp) view pane, then roll the pointer over the *L2 floor_plane*.

When you see the “L2” label in the pan, you know the pointer is in the right spot.

- 7 Click once to select the *L2 floor_plane*.

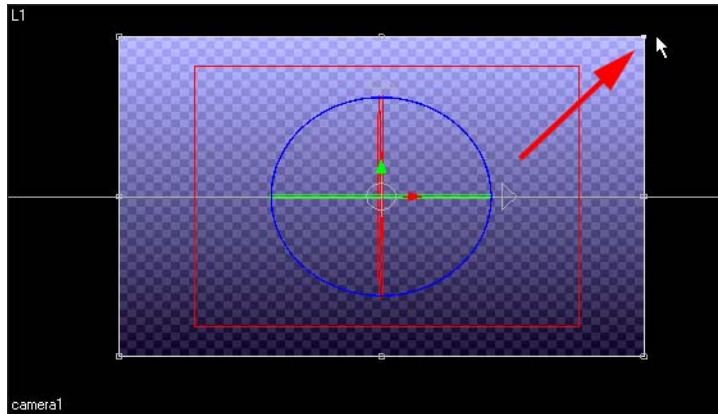


- 8 Use the pan controls and white triangles to line up the back edge of the floor with the bottom edge of the wall.

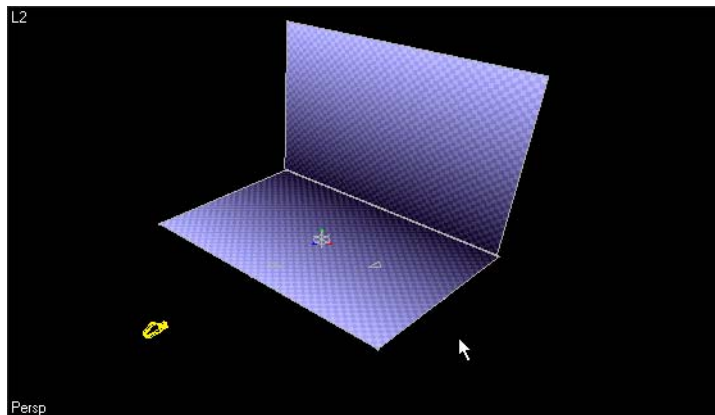


Make sure you check all the view panes while you move, because it's practically impossible to get precise positioning from a single view.

- 9 Select the *L1 wall_plane* element in the Camera1 pane, then drag a corner to resize it to the same width as the floor.



- 10 Use the pan controls to adjust the wall at the back edge of the floor. When you're done, the image should look similar to this:

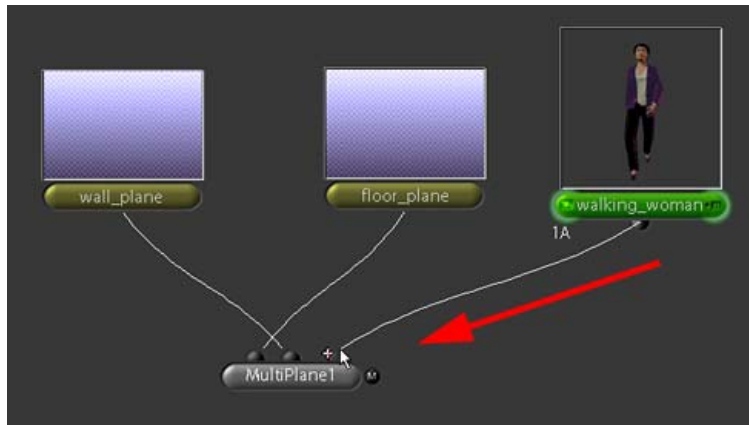


Now let's add the star of your shot—Virtual Betty.

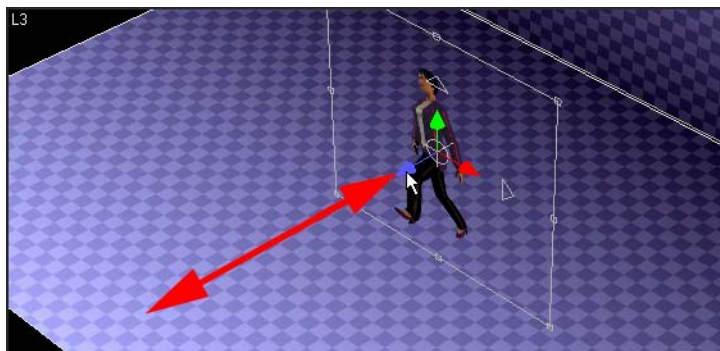
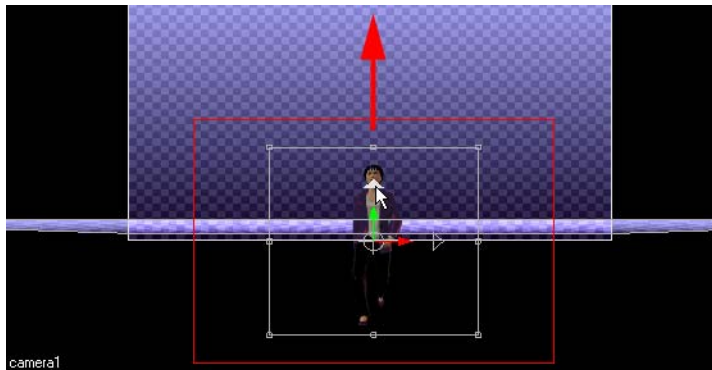
To add a new element and adjust the camera position:

- 1 Add an *Image-FileIn* node, then browse to *\$HOME/nreal/Tutorial_Media/Tutorial_03/images/betty*.
- 2 Select the *walking_woman.1-62@.exr* image sequence, then click OK.

- 3 Drag a noodle from the *walking_woman* node to the + (plus) sign on the *MultiPlane1* node.



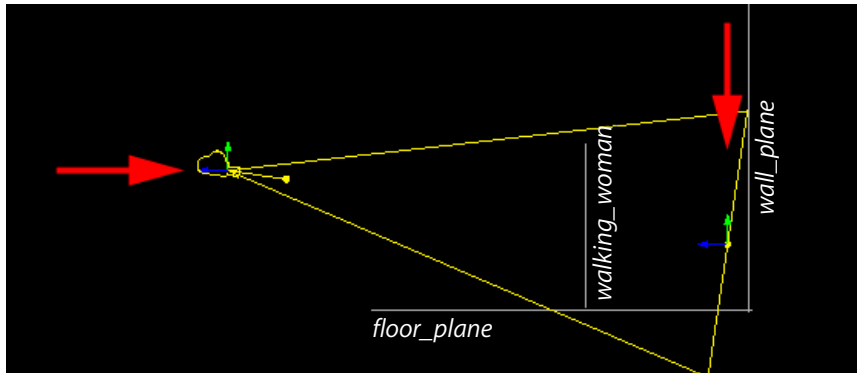
- 4 In the Camera1 pane, select the *L3 walking_woman* element—your Virtual Betty—and adjust her position so that she appears to be standing on the floor.



It won't look perfect because the camera is in the wrong position. So that's the next step—place the camera in a position that matches the camera that rendered our *walking_woman* image.

Shake can import precise camera data from an animation program, which you'll do in the next example. For this project, however, make the camera adjustment manually.

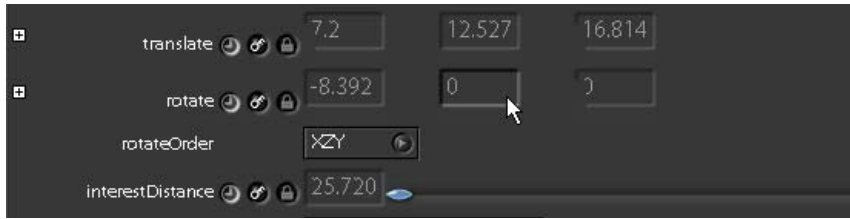
- 5 In the Side pane, select the camera, then use the pan controls to move the camera up and forward until the scene is framed.



In the Camera1 pane, the red border shows what will appear within the rendered frame. You might need to make some minor adjustments to match the illustrations shown here.



Here are the camera settings for the project shown in the illustrations:



- 6 Open the Globals tab, then click the Auto button to adjust the timeRange parameter for the length of the *walking_woman* clip.
- 7 Click the Flipbook button in the Viewer shelf to render a preview the composite (and keep the Flipbook window open for the next section).
- 8 Press the period key (.) to play the Flipbook.

Note: Pressing period (.) plays forward in the Flipbook. Comma (,) plays in reverse. The Space bar stops/starts playback.

The results are entertaining. Virtual Betty floats over the floor, as if trucking along in zero gravity. The problem is that Betty's clip was rendered with a camera move, but the *MultiPlane* camera is stationary. As a result, the wall and floor elements don't match up with the action in the animated clip. In the next section, you'll fix this by repositioning and animating the camera.

- 9 Click the Save button at the top of the Node View.
- 10 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_03/scripts` directory, then save the script as *Betty.shk*.

Animating a MultiPlane Camera

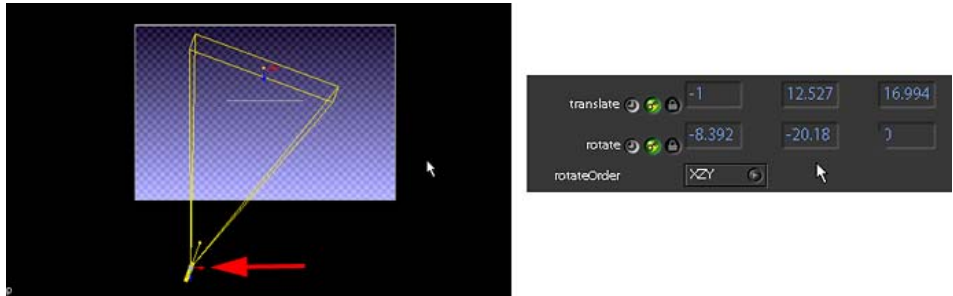
Take another look at the Flipbook preview. When you play back the Flipbook, the Virtual Betty animation indicates a camera move that swings around the girl.

To animate the camera to match the *walking_woman* animation:

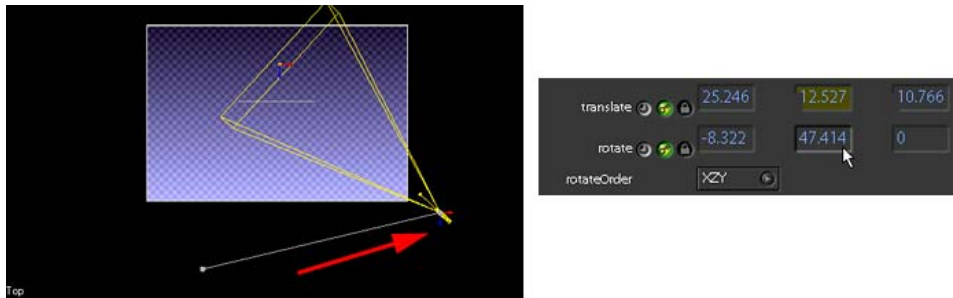
- 1 Click the Home button in the Time Bar and make sure the playhead is at frame 1.
- 2 Click the Camera tab in the *MultiPlane* parameters, then click the Autokey buttons next to the translate and rotate parameters.



- 3 In the Top pane, move the camera position—but do not move the camera target—to the left edge of the floor.



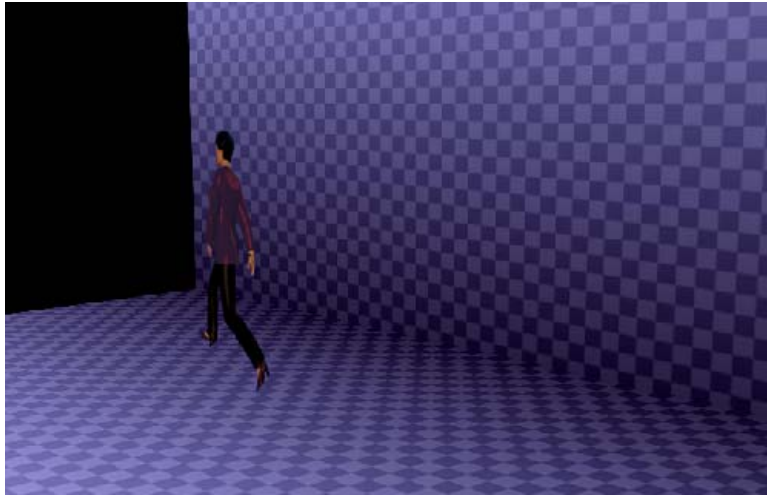
- 4 Move the playhead to frame 62.
- 5 Now move the camera to the opposite end of the floor to create the camera move.



For the first keyframe, the camera settings should be as follows: translate = -1, 12.527, 16.994 and rotate = -8.392, -20.18, 0. At frame 62 the settings should be: translate = 25.246, 12.527, 10.766 and rotate = -8.322, 47.414, 0.

- 6 Click the Flipbook button to preview the results.

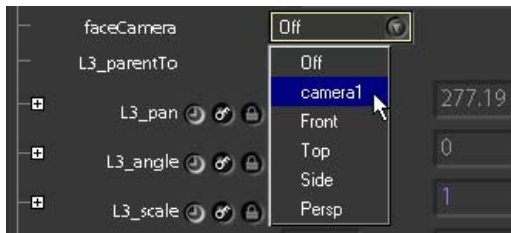
The first 10 frames look okay. At about frame 40, however, we begin to see just how two-dimensional and superficial Betty really is.



Fortunately, Shake has a useful control that will take care of Betty's 3D shortcomings: `faceCamera`. This parameter automatically rotates an element in the 3D space to face the camera, at every frame.

To force a 2D plane to face the camera:

- 1 In the *MultiPlane* parameters, click the Image tab and open the subtree for the *L3 walking_woman* layer.
- 2 Choose `camera1` from the `faceCamera` pop-up list.



- 3 Click the Flipbook button again to generate a new preview of the scene.

The results are better, but you'll notice that Betty's footsteps don't seem to move in relation to the floor. This is because the current `faceCamera` settings are a rough approximation of the camera move from the animated scene.

If you had camera data for Betty's animation, you could import it into Shake and get a better lock on the camera move.



The *betty-DONE.shk* file, located in the `$HOME/nreal/Tutorial_Media/Tutorial_03/scripts/` directory, takes this project a little further by adding simulated lighting and shadow effects to the composite.

Importing Camera and Animation Data

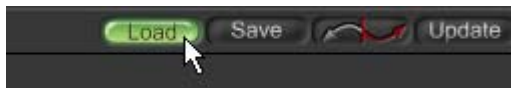
In this example, continue with the *MultiPlane* node, and create a composite by combining 3D and 2D elements with camera data imported from 3D animation software.



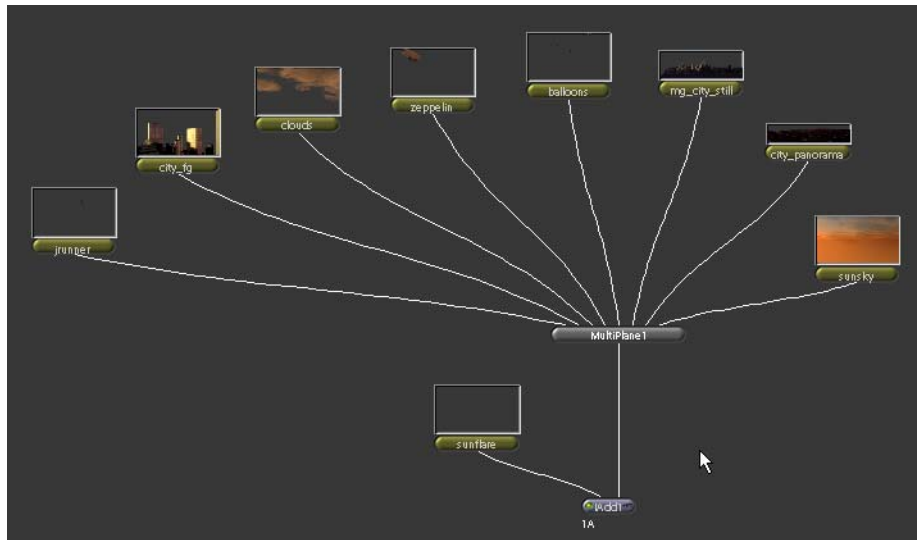
A script is already set up to help you get started.

To load and preview the *MultiPlane* example:

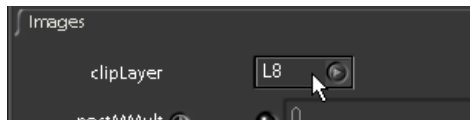
- 1 Click the Load button at the top of the Node View.



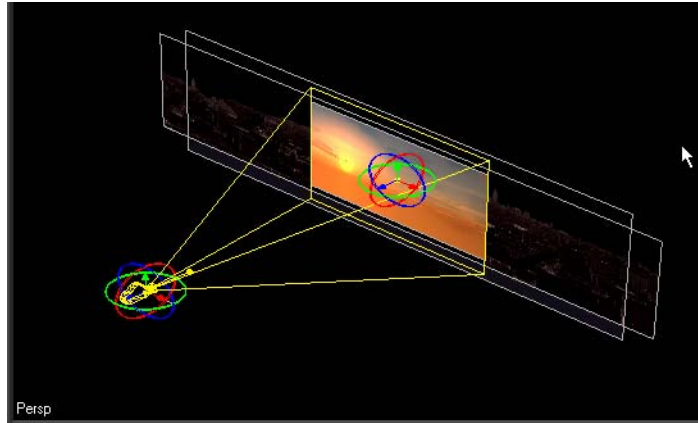
- 2 Browse to the `$HOME/nreal/Tutorial_Media/Tutorial_03/scripts` directory, select `multiplane-start.shk`, then click OK.



- 3 Double-click the *MultiPlane1* node to review the structure of the composite.
Remember the clipMode option from Tutorial 1? The *MultiPlane* node has a similar feature, called clipLayer. This lets you select the layer that has the resolution you want to pass to the next node in the tree.
- 4 In the *MultiPlane1* parameters, open the Images tab, then change clipLayer to L8 (*sunsy*).



If you look in the Perspective pane, you'll now see these elements:

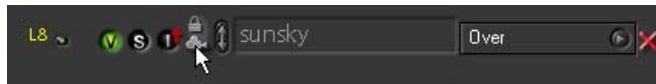


The image sequences include moving objects and other elements that have a noticeable change in perspective. For everything else—the detailed city background elements—you'll use simple still images that must be matched to the animated camera move.

Only the still images will make use of the *MultiPlane* camera move. The animated image sequences already have the camera move “baked in,” and so, to avoid a perspective change on these, you need to lock them to the camera view.

To lock elements to the camera:

- 1 In the *MultiPlane1* parameters, scroll down to the L8 (*sunsky*) layer, then click the Attach Layer to Camera button to lock this element to the camera perspective.



- 2 Repeat the previous step for each of the following layers to lock them to the camera: L5 (*balloons*), L4 (*zeppelin*), L3 (*clouds*), L2 (*city_fg*), and L1 (*jrunner*).

- Expand the subtree beneath the L8 (*sunsky*) layer and set the camera distance value to 2500.



Note that this changes the position of the layer along the Z axis, but the camera lock ensures that the view from the camera's perspective does not change. In this case, it allows the sunsky layer to appear behind everything else in the composite.

- Adjust the camera distance for the other layers, to match these settings:

- L5_cameraDistance = 1500
- L4_cameraDistance = 1200
- L3_cameraDistance = 1000
- L2_cameraDistance = 800
- L1_cameraDistance = 700

This takes care of the distance/position for each of the elements locked to the camera. These are just arbitrary numbers to space the layers; on a real shot you'll want to be more precise. Now you need to adjust the still images that will be affected by the camera move.

- Now preview the results by rendering a Flipbook preview.

When you play back the Flipbook sequence, things look fine at frame 95, but the composite breaks down before frame 155. The two still images are not positioned correctly, and do not match the camera move.



It's time to import the camera data from the 3D animation project to create the proper perspective change for the 2D images.

Before you import camera data, it's very important that the in/out frames on the Time Bar match the start/stop frames for the camera data. In our example, our camera data starts at frame 95 and ends at frame 156—and the project Time Bar is already adjusted accordingly.

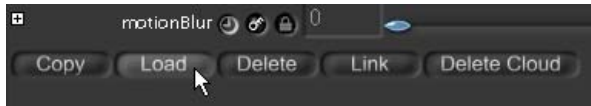
To import 3D camera data:

- 1 Display the Time View and turn off the Shift Curves option.

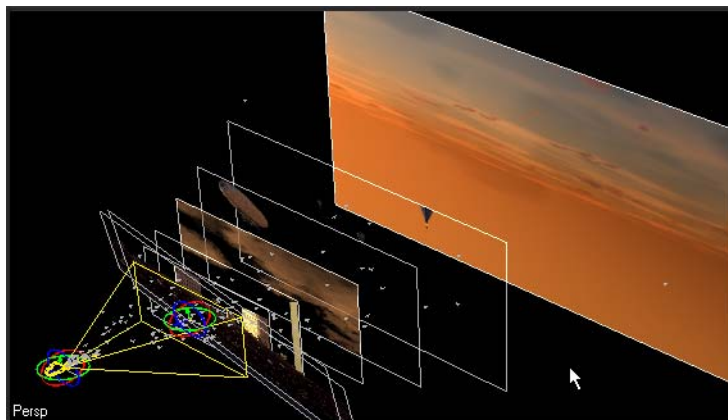


This step is very important, so that Shake will not adjust the frame data from the imported camera.

- 2 In *MultiPlane1* parameters, click the Camera tab.
- 3 Scroll down and click the Load button.



- 4 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_03/images` directory, then select the `track_data.ma` file and click OK.



You'll now see the point cloud data and the updated camera (Camera_1_1) added from the 3D data file.

Now, before you preview, adjust the position of the still images.

To adjust the position of the still images:

- 1 Display the Image tab in the *MultiPlane1* parameters.
- 2 Expand the subtree beneath the L7 (*city_panorama*) layer, then adjust the pan values to -1580, -258, and -1300.



Or, just select the layer in the perspective panel and move it with the translate handles.

- 3 Set the L7_scale to 2, 2, 1.



- 4 Set the L7 layer faceCamera option to "Camera_1_1."
- 5 Expand the subtree beneath the L6 (*mg_city_still*) layer, and set its pan values to -1032, -366, and -1370.
- 6 Set the L6_scale to 1.75, 1.75.

To preview the finished composite:

- 1 In the Time Bar, scrub to frames 20, 25, and 62 on the Time Bar to view the change in the camera position resulting from the imported data.

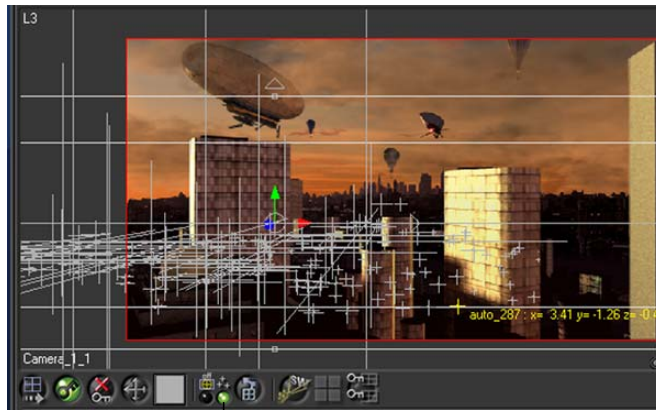


- 2 Now click the Flipbook button to generate a new preview.

This time, you'll see a better lock on the camera movement for all the layers. There may still be some of the "2D skew" on the still images, similar to what you saw in our "Betty" example.

You can solve this again by choosing the "Camera1_1" (the rendering camera) for the faceCamera option on the still images. As before, this doesn't change the location of the elements, but it does force them to rotate perpendicular to the camera view, on every frame.

The point cloud data can help you determine the placement for layers in the composite and verify the camera move. You can move the pointer over a data point to get the precise location of the point. You can hide the point cloud by clicking the Point Cloud Display button:

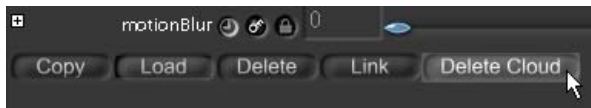


Click here to hide the point cloud data.

You can also completely remove the point cloud data from the project.

To clear the point cloud data:

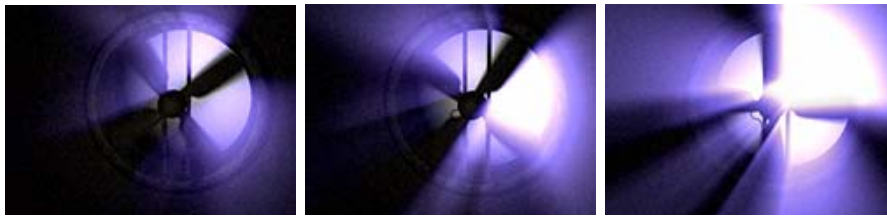
- 1 In the *MultiPlane* parameters, scroll to the bottom of the Camera tab.
- 2 Click the Delete Cloud button.



You can continue to apply some of the other tricks shown earlier in this chapter to add haze, color correction and other effects.

This tutorial shows how to generate animation with expressions, rather than keyframes.

Starting with one image of a ceiling and five frames of a rotating fan, you will create an animated effect using filters, local variables and expressions.



Tutorial Summary

- Creating the fan composite
- Creating a light source with the *RGrad* node
- Looping frames in the Time View
- Using local variables and expressions
- Simulating volumetrics with *RBlur*
- Concatenating color adjustments
- Adding motion blur to pre-animated elements

Creating the Fan Composite

The fan composite is divided into four steps:

Step 1: Set up the basic composition, both in the frame and in time. Since you have only five images of the fan, loop the animation using the Time View. Also, add a *RGrad* gradient behind the ceiling image to act as a light.

Step 2: Using expressions, add a flickering effect to the *RGrad* node, and use local variables to better control the expressions.

Step 3: Add a radial blur (*RBlur*) effect to create a volume-rendering feel.

Step 4: Although it is already animated, add motion blur to the fan because it's cool.

To read in the lesson files:

- 1 In the Image tab, click the *FileIn* button, browse to the `$HOME/nreal/Tutorial_Media/Tutorial_04/images` directory, then load *ceiling.iff* and *fan.1-5@.iff*.

In the File Browser, use one of the following methods to read in the files:

- Select one file, hold down the Shift key, then double-click the second file.
- Drag to select both files, then click OK.
- Select *ceiling.iff*, press the Space bar, select *fan.1-5@.iff*, then click OK.

Note: Pressing the Space bar in the File Browser is the equivalent of using the Next button, and it's best used when you need to import multiple files that are located in different directories.

The two *FileIn* nodes, *ceiling* and *fan*, are added.



- 2 Add an Image-*RGrad* node.

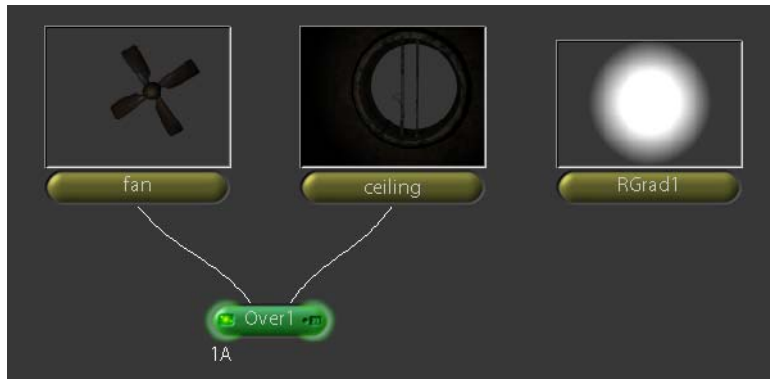


- 3 In the Globals tab, set the timeRange for the project to 1-75.

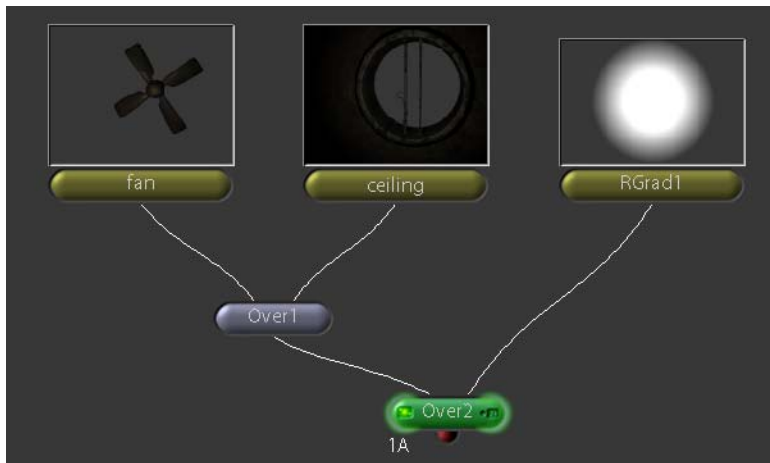
- 4 Click the Home button in the playback controls to load the new frame range into the Time Bar.

To composite the images together:

- 1 In Node View, select the *fan* node and add an Layer-Over node.
- 2 Attach the *ceiling* node to the second input (background) of the *Over1* node.
The *fan* image is layered over the *ceiling* image.



- 3 Select the *Over1* node, add a second Over node, and attach *RGrad1* as the background.



- 4 Your copy of *RGrad1* may be a different resolution than the other elements in the composite. Click the right side of the *RGrad1* node, then enter a resolution of 400 by 300.



- 5 Toggle the clipMode parameter of *Over2* to set the resolution to foreground.



The image should now look like this:



Creating a Light Source With RGrad

Since you have only five frames of material, you can add a little kick to the composite by animating the position of the *RGrad* gradient shape to simulate a moving spotlight.

To animate the *RGrad* node:

- 1 Go to frame 1.
- 2 Click the left side of the *Over2* node to load it into the Viewer, then click the right side of the *RGrad1* node to load its parameters.

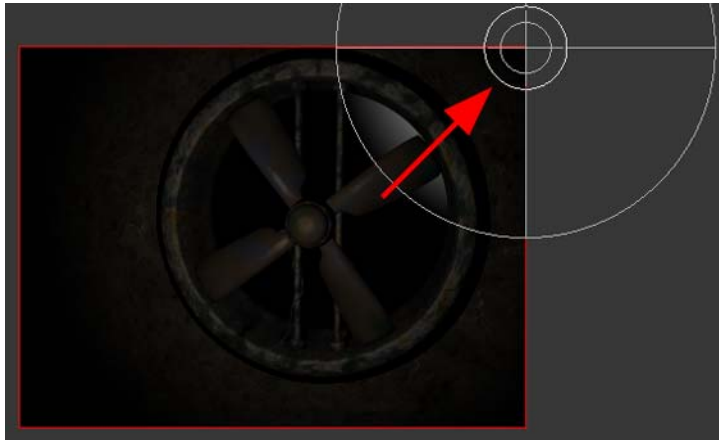
The *RGrad1* onscreen controls appear. You might need to zoom out a bit; move the pointer over the Viewer and press the minus key (–) a few times.

- 3 In the Viewer, turn on Autokey.

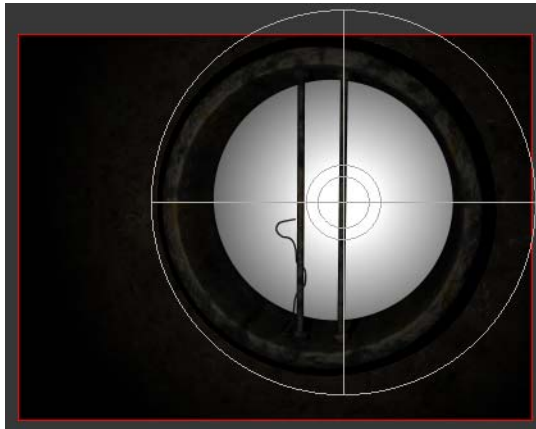


A position keyframe for the *RGrad* gradient shape is created.

- 4 Using the onscreen controls, position the gradient shape in the upper-right corner, then drag the middle ring to reduce the size of the RGrad.



- 5 Go to frame 75.
- 6 Using the following illustration as a guide, move the center of the gradient to create a second position keyframe.



Now drag along the Time Bar to preview the dramatic animation. You'll note that the fan blades disappear after frame 5, somewhat reducing the realism of the scene! You can fix this in the Time View.

Looping Frames in the Time View

Click the Time View tab to view the clip length of your nodes. Only image generators, such as *FileIn* and *Grad*, and layer nodes that combine two branches together (*Over*, *IDisplace*, and so on) appear in the Time View.



As we saw in Lesson 2, the infinity symbols (∞) on the left and right edges of *ceiling* and *RGrad1* indicate that these clips—or more accurately, the output of these nodes—have no start or end frames and extend the full length of the frame range set in the Globals tab. The short *fan* clip, however, has a finite length of five frames. At frame 6, the clip disappears.



To extend the length of a clip:

- 1 Holding down the Control or Command key, drag the right edge of the clip to frame 75.

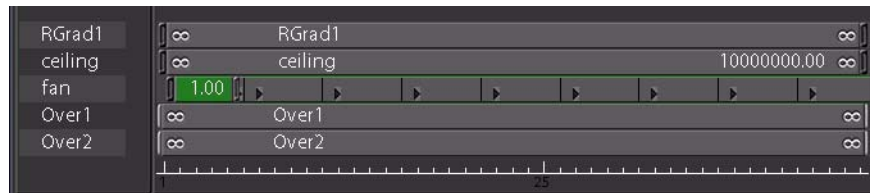


- 2 In the *fan* node parameters, open the Timing tab.
- 3 Set the outMode to Repeat.



The inMode and outMode parameters tell Shake what to do with a clip that is extended beyond the number of frames available from the image sequence on disk.

In the Time View, you can see that the fan clip repeats; the triangles mark the looping points.



- 4 In the Time Bar, click the forward arrow button to play back the sequence.



This button differs from the normal Viewer Flipbook button because the images are not loaded into RAM. Shake renders and plays as quickly as possible. You should see the fan blades turning.

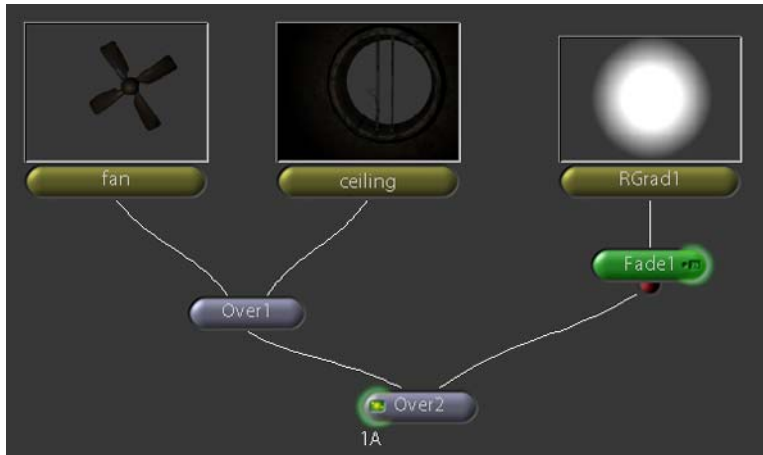
Using Local Variables and Expressions

An expression is a simple formula for calculating values. Expressions can be used to control animation parameters. A variable is an alias that stores or “points to” a value. The name of the variable stays the same, but the value stored in the variable may change.

In this section, you’ll continue the project by creating expressions with variables to create a flickering effect on the background light. This can be accomplished with keyframe animation, but you’ll soon see that expressions can help you do this more efficiently.

Creating Flicker on the Light

- In the Node View, select the *RGrad1* node and insert a *Color-Fade* node between the *RGrad1* and *Over2* nodes.



You can, of course, enable Autokey for the *Fade1* value, then manually animate the fade for a flicker effect on the light. You can automate this with an expression.

To insert a function into the value text field:

- 1 In the *Fade1* parameters, click the value text field to select the current value.
- 2 Enter this simple expression: *time*



A plus sign (+) appears next to the value parameter. Click the plus sign (it changes to a minus sign after you click it) to open the expression for editing.



The keyword *time* tells Shake to take the current frame number and use it for value. Drag along the Time Bar to test it. As the playback slider moves to higher frames, the fade value increases, too. This of course brightens the image too much, causing it to blow out before you get to the end of the shot. Not exactly a great flicker effect.

You can put any expression into the value expression field. To create the flickering light effect, you need to make small edit using the random function (*rnd*).

- 3 Edit the value expression so that it reads like this: `rnd(time)`



This returns a random value between 0 and 1 with time as your seed. Without the time seed—or a variable with changing values, the `rnd()` function is not truly random. Use a static number for the seed, and the fade value will be identical for every frame. By using *time* as your seed, you guarantee that the values differ from frame to frame.

Note: Additional functions can be found in Chapter 31, “Expressions, and Scripting,” in the *Shake 4 User Manual*.

- 4 Drag the value slider to remove the expression.

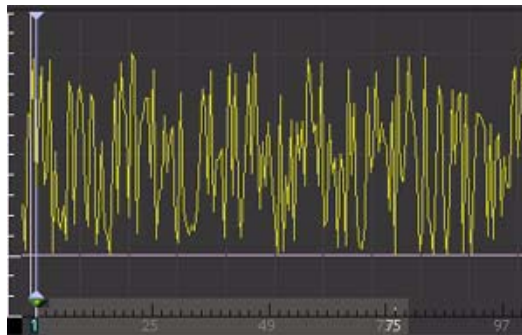
When you manually set a parameter that is linked to an expression, the expression is deleted, except when the expression is an animated curve (as in, *Hermite(0,[1,0,0]@1,[0.5,0,0]@100)*). Although curves are actually expressions, changing a parameter linked to a keyframed curve merely adds a new keyframe or modifies an existing keyframe.

- 5 Type the expression again, or click Undo.



Note: You can also press Command-Z or Control-Z to undo.

- 6 Click the Load Curve button, then open the Curve Editor to view the animation curve for the *Fade1* value parameter.



You can't edit this curve in the Curve Editor because each value is generated by the expression. However, you can create additional variables, which are "local" to the *Fade1* node, then use them to control the values generated by your original expression. The result will be a curve or "envelope" that shapes the random values.

To help control the expression:

- 1 Right-click in the *Fade1* Parameters tab, then choose Create Local Variable from the shortcut menu.

The Local Variable Parameters window appears.

- 2 Enter "rndVal" as the Variable name.

When you create a new variable, you also need to choose the type of information you want to store in the variable:

- *float*: A number with a decimal place (0.0, .1, .5024, 1.00, and so forth)
- *int*: A rounded number (0,1, 2, and so forth)
- *string*: Characters ("Hey, what's up?")

- 3 Set this variable to float, to specify a decimal value.
- 4 Click Next to create the next variable.
- 5 Replace the rndVal name with animVal, set this variable to float also, then click OK.

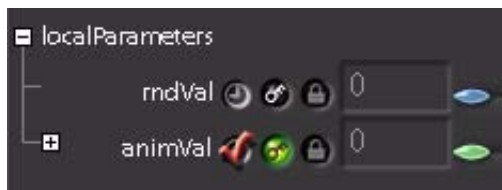
A new subtree named localParameters appears at the bottom of the *Fade1* parameters list.

- 6 Open the localParameters subtree to access the new local variables.



If you adjust the rndVal slider, you'll see that absolutely nothing happens. Why? You need to make a few more changes before this parameter becomes active. If you adjusted the rndVal slider, set it back to 0 now.

- 7 Turn on Autokey for the animVal parameter.



- 8 Set keyframes for animVal at frames 1, 40, and 75. One of the keyframes—it doesn't matter which one—should be set to a value of 1. The other two can be any decimal number between 0 and 1.

The only parameter that matters to the *Fade1* node is the value parameter. The other parameters are not hooked into the value parameter yet, so they have no effect. To make things a little easier to work with, swap the *rnd* function from the value parameter to the rndVal parameter.

- 9 Click the value expression field that reads *rnd(time)*. Copy the whole expression (press Command-C or Control-C) and paste it (press Command-V or Control-V) into the box for the rndVal local variable (in the localParameters subtree).
- 10 In the Fade1 value expression field, change the expression to:

*rndVal * animVal*

This calls the two values from rndVal and animVal and multiplies them:

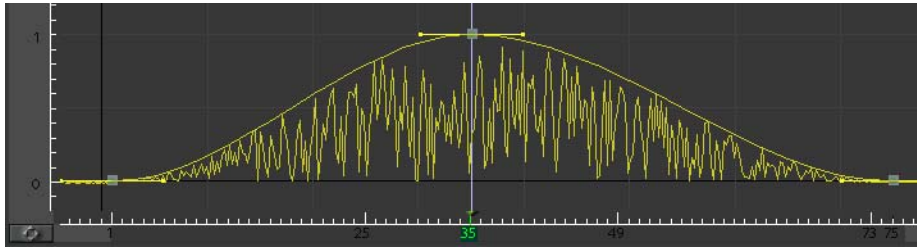
- value is set to *rndVal * animVal*.
- rndVal is set to *rnd(time)*.
- animVal is set to an animated keyframed curve, which modifies the shape of the otherwise random curve.



Note: When you create these expressions, the use of uppercase and lowercase does matter. Any variables or parameters referenced in an expression must be typed with the precise names.

- 11 Make sure value and animVal, but not rndVal, are loaded into the Curve Editor. (Click the Load Curve button next to the parameter name.)
- 12 In the Curve Editor, edit the keyframes for animVal to change the form of the random shape. If you do not see both curves, make sure the Load Curve button is toggled on for the value and animVal parameters, and that at least one of the animVal curve keyframes is set to 1.

Note: To frame the curves, position the pointer in the Curve Editor and press the Home key.



This trick makes everybody warm and happy. When you change the shape of the *animVal* curve, you'll see how it controls the shape of the random values. This creates a flickering light with random values that are more “organic.”

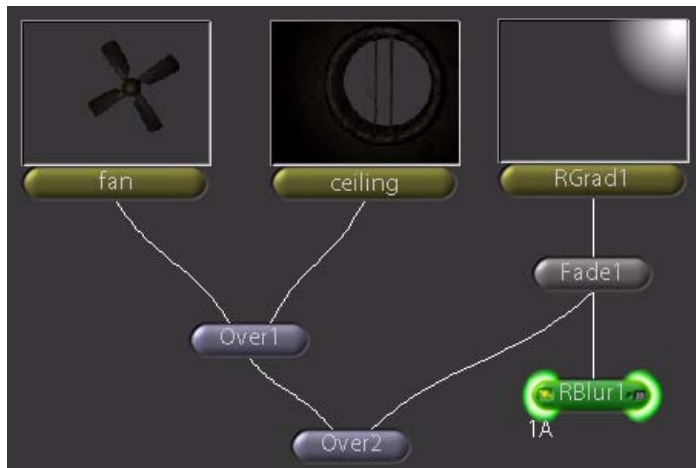
If you are using the playback button, close the Curve Editor by switching to a different tab—when displayed, the Curve Editor and the Color Picker slow down playback speed.

Simulating Volumetrics With RBlur

The following steps build the atmosphere with the *Filter-RBlur* function. *RBlur* stands for Radial Blur.

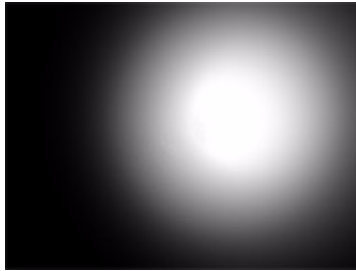
To insert the *RBlur* node:

- 1 In Node View, select the *Fade1* node.
- 2 Shift-click *Filter-RBlur*. This creates a new branch from the *Fade1* node.



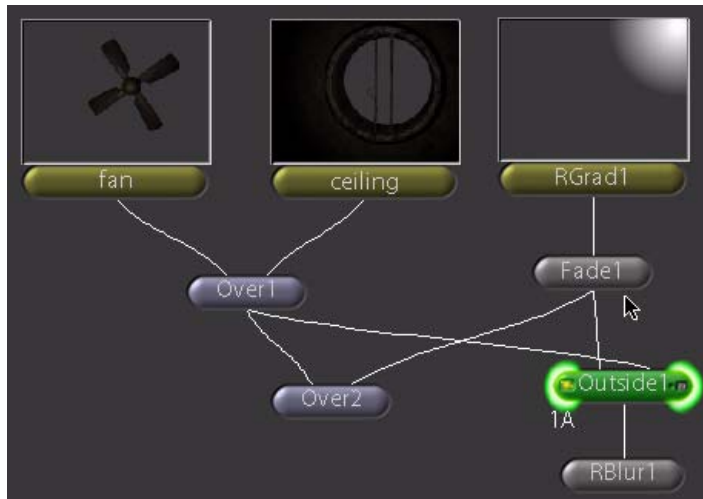
- 3 In the *RBlur* parameters, set the *iRadius* to 35.

The default values do not greatly modify the *RGrad*, so you need to make some adjustments.

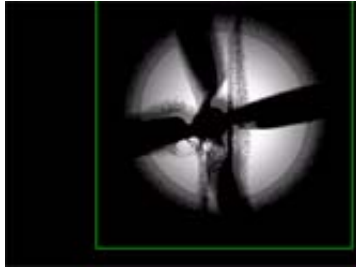


To give the *RBlur* more definition:

- 1 In Node View, select the *Fade1* node.
- 2 Shift-click Layer–*Outside* to insert a new layer node.
- 3 Drag a second connection from the *Over1* node to the second input of the *Outside1* node.
- 4 Connect *RBlur1* to *Outside1*, as shown below.

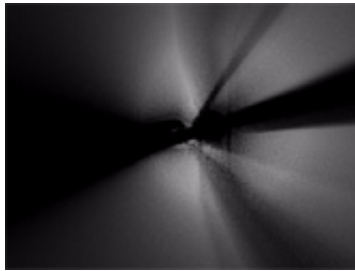


As a result, the *RGrad* gradient shape appears only where there is no alpha in the ceiling/fan composite. *RBlur1* now has a slightly more interesting effect.



5 In the *RBlur1* parameters, do the following:

- Set *iRadius* to 0. (There is no area of the center that is non-blurred.)
- Set *oRadius* to 300. (This sets the blur amount.)
- Set *amplitude* to 1. (This is a multiplier on *oRadius*—the greater the number, the more blur; also affects the rendering speed.)
- Set *quality* to 0.1. (The quality is lowered to speed rendering.)



Linking the *RBlur* effect to the *RGrad* shape

Although the light is moving, the rays are emanating from the same static position. To improve the effect, animate the center of the *RBlur*, and match it to the center of *RGrad1*. The best way to do this is to link the *xCenter* and *yCenter* parameters of *RBlur1* to the *xCenter* and *yCenter* parameters of *RGrad1*. Then, if *RGrad1* is modified, *RBlur1* is automatically adjusted.

To link a parameter to another parameter within the same node, type the name of the parameter. To link to a parameter in a different node, type the name of the node, a period, and then the parameter name (*NodeName.parameterName*). Node names and parameters are case-sensitive, and Shake does not tolerate typos!

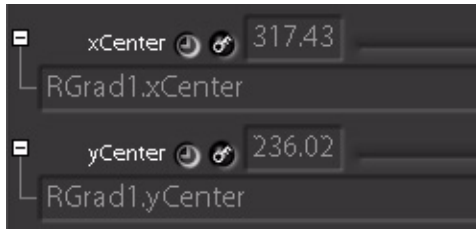
To link the node parameters:

- 1 In the *RBlur1* parameters, expand the center parameter subtree, then type the following in the xCenter field:

RGrad1.xCenter

- 2 Type the following in the yCenter field:

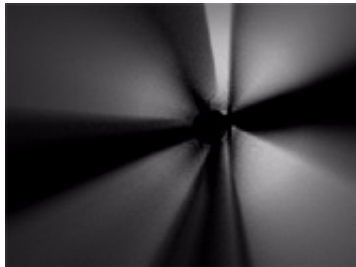
RGrad1.yCenter



These expressions are similar to *rnd(time)* or *animVal * rndVal*, except that they check the xCenter and yCenter in the *RGrad1* node for their values.

Note: If you enter new values for xCenter or yCenter, or drag the sliders for these parameters, this breaks the links to *RGrad1*. You can restore the link by clicking Undo.

The origin point of the rays is now animated to match the center of the light.



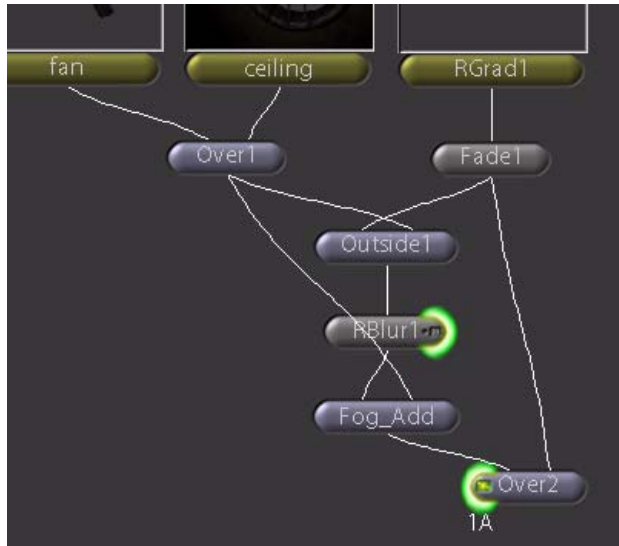
Connecting the RBlur to the Composite

The following steps show you how to make all the node connections for the *RBlur*.

To connect the *RBlur*:

- 1 In Node View, select the *Over2* node and press E to extract it from the script. Move it down to make space for new nodes.
- 2 Select the *RBlur1* node, and add a new *Layer-Add* node.
- 3 Rename the new *IAdd1* node to *Fog_Add*.
- 4 Drag a connection from the *Over1* node to the second input of *Fog_Add*.
- 5 Connect the output of *Fog_Add* to the first input of *Over2*.

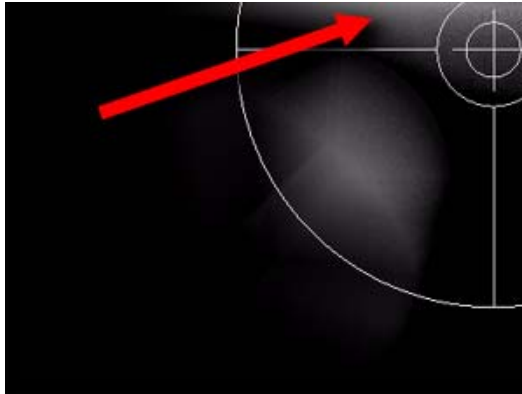
- 6 Connect the output of *Fade1* to the second input of *Over2*.



The *Fog_Add (IAdd)* node adds the two images together. *Over2* places the volumetric effect and the fan/ceiling elements over the *RGrad*.



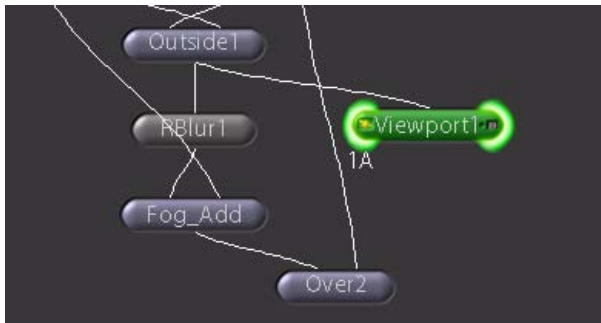
If you scrub through the Time Bar, you might see a frame where the *RGrad* is mostly outside the workspace (probably an early frame), and light improperly bleeds in from the edges. Where does this come from?



The Shake engine has a normally divine feature called the Infinite Workspace. With the Infinite Workspace, nothing is cropped if it extends outside the frame. *Outside1* should cut off the *RGrad*, except when viewed through the hole in the ceiling element. However, the *RGrad* is larger than the ceiling element, so it is not completely obscured by the ceiling element.

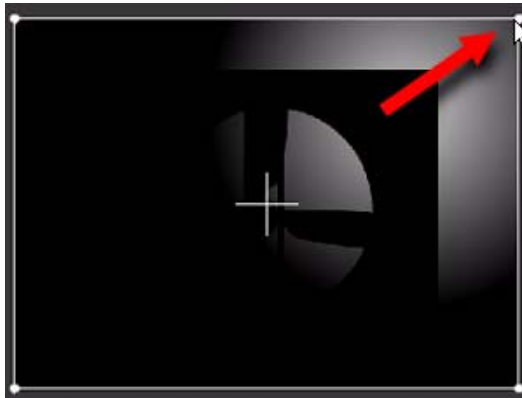
To check the extended canvas:

- 1 In Node View, select *Outside1*, click the Transform tab, then Shift-click *Viewport*.



Viewport will help you view the extended canvas.

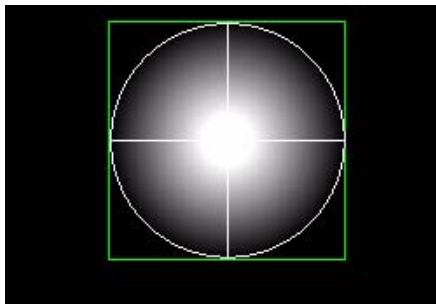
- 2 Drag the *Viewport* onscreen controls to enlarge the canvas to reveal that the *RGrad* exists outside of the frame boundary. You can drag a corner or an side of the transform overlay.



To correct this, use one of the following (but not both) solutions:

Solution 1: Control the Background Color

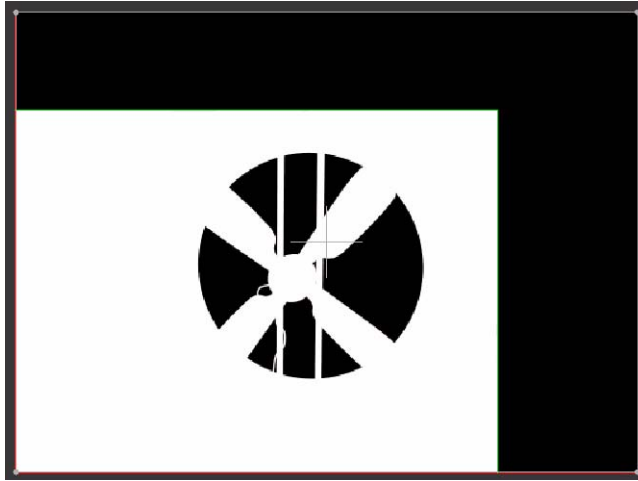
Background Color is considered everything outside the Domain of Definition (DOD) of the image. The DOD is normally the frame of the image, but could be smaller. For example, when you view *RGrad1*, the DOD is described by the *RGrad* itself, and is the area inside of the green box. The green box is the DOD. The DOD of the ceiling element is the frame border itself.



To add a *SetBGColor* node:

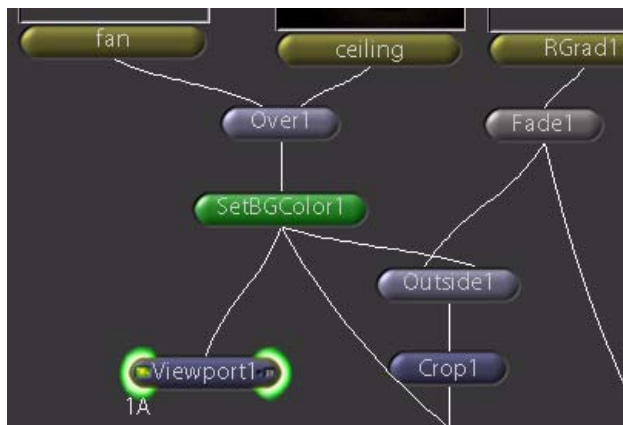
- 1 Rewire the *Viewport* (created earlier) to the *Over1* element (see the illustration on the following page).

- 2 Display the alpha channel (press A with the pointer over the Viewer) and it will look similar to this:



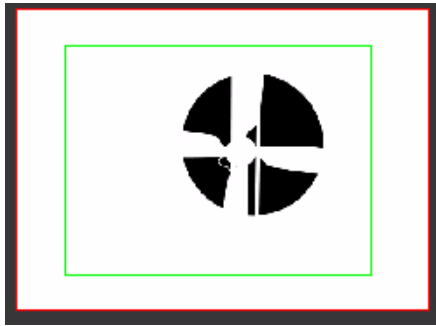
This area outside of the DOD is called the Background Color, and its color can be controlled with `Color-SetBGColor`.

- 3 In Node View, select *Over1*.
- 4 Insert a *SetBGColor* node from the Color tab.

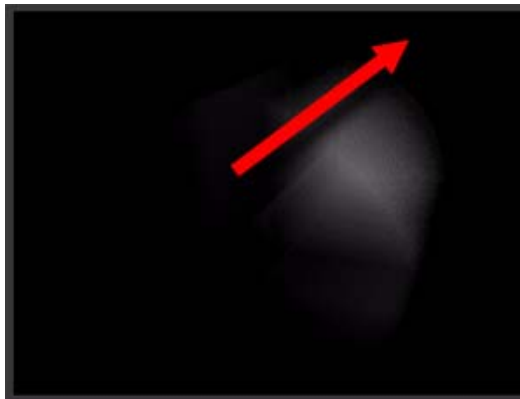


- 5 In the *SetBGColor1* parameters, set the alpha value to 1.

The solid alpha value is extended infinitely, and masks the *RGrad* outside of the ceiling frame.



- 6 View *RBlur1* (click the left side of the node).
The bleeding from the edges disappears.



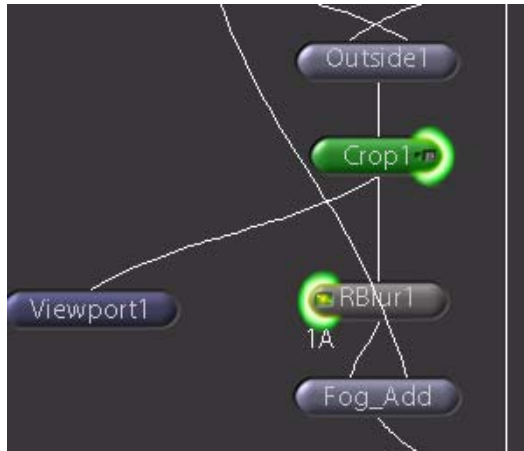
Solution 2: Clip the Infinite Workspace

The second solution is to clip the Infinite Workspace with a *Crop* node. This removes everything outside of the frame, and returns the frame to black.

To add a *Crop* node:

- 1 If you already followed Solution 1, delete the *SetBGColor1* node and reconnect the *Viewport1* node to the *Outside1* node.
- 2 Select the *Outside1* node and insert a *Transform–Crop* node.

You are finished with this stage.



- 3 Display the *Viewport1* node and compare the difference between the output of the *Outside1* node and the output of the *Crop1* node. Just reconnect *Viewport1* to check the output of each.



Outside1



Crop1

- 4 When you're finished, delete the *Viewport* node.

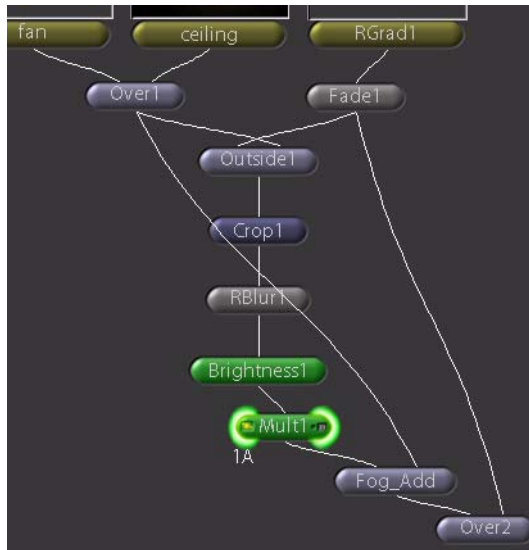
Concatenating Color Adjustments

Use the following steps to punch up the effect, add a bit of color to the flaring, and brighten the image.

To concatenate the color corrections:

- 1 In the Node View, select the *RBlur1* node.

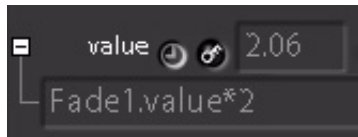
- 2 Insert a *Brightness* node and a *Mult* node from the Color tab, then set *Brightness1* as the active node in Parameters.



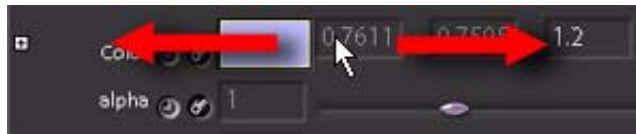
Both nodes should be between *RBlur1* and *Fog_Add*. You'll now create an expression to take the animation values from *Fade1*, double them, then apply the result for the *Brightness1* value.

- 3 Enter this for the *Brightness1.value* parameter:

*Fade1.value * 2*



- 4 In the *Mult1* parameters, set the color to blue using the Virtual Color Picker. Hold down the B key and drag to the right to boost the blue color.

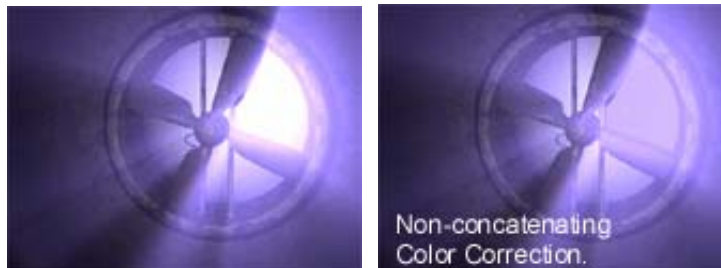


It is rather subtle, but *Brightness1* and *Mult1* are concatenating their color corrections. The *Mult* should tint the entire *RBlur* blue, but you can see that the center remains white because *Brightness1* internally raises the values of the center to around 1.5, 1.5, 1.5 (numbers are a theoretical example). When *Mult1* multiplies the color by .7, .7, 1 in the RGB channels, it lowers the color to 1.05, 1.05, 1.5. At 8 bits, this rounds down to 1, 1, 1, which is white.

To view what occurs without concatenation:

- 1 Insert a *Filter-Blur* node between *Brightness1* and *Mult1*.

With the *Blur* node between the *Brightness1* and *Mult1* nodes, everything turns a flat, disagreeable blue.



Concatenating

Non-Concatenating

- 2 Delete the *Blur* node to preserve the concatenation.

Adding Motion Blur to Pre-Animated Elements

As you view the animation, the fan movement appears too... well... mechanical. The problem is a notable lack of motion blur in the image of the moving fan. Although it's best to render this sort of thing into the animated images, Shake has a few tricks to help you add this, post-render.

There is one caveat: Shake generates motion blur only when it's creating motion with one of its transform nodes. The rotation of the fan is "baked into" the clip—movement within a clip doesn't give Shake anything from which to create a motion blur.

To solve this, you'll create and animate a placeholder that matches the shape and movement of the fan, then apply its motion blur to the fan clip. You'll use the *Move3D* node to animate the placeholder—the letter "X"—and generate the motion blur.

Note: At this point, your script includes several elements, and a preview with the playback button might be a little slow. From this point forward, consider using **Render > Flipbook** to create previews. Don't forget to set the frame range to 1-75.

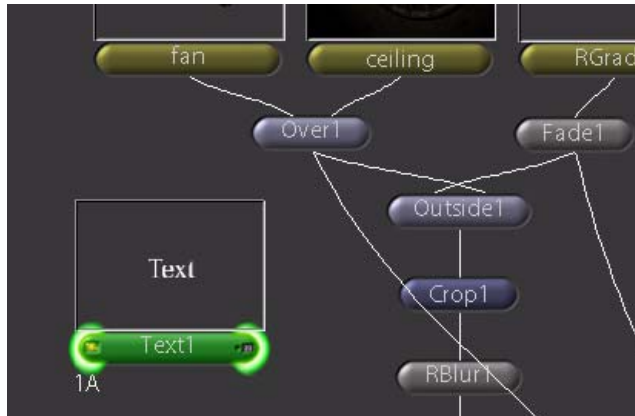
Create a Motion Blur Placeholder

You'll use the *Text* node and the *Move3D* node to create a temporary placeholder for motion blur. Normally, when a *Move3D* node is applied to an element, it transforms the element. In this case, you will toggle a *Move3D* parameter, called *useReference*, to apply the motion blur, but not the transform.

To create the placeholder:

- 1 Create an *Image-Text* node.
- 2 Set the *Text* node parameters:
 - a In the *Text* node parameters, expand the *Res* subtree, then set width and height to 160.
 - b In the text field, type "X" as your text.
 - c Click the font pop-up menu and select *Courier*.
 - d Expand *fontScale* subtree, then set the *xFontScale* to 307 and set the *yFontScale* to 275.

- Under the position parameter, set yPos to 140.



Your result should look like this:

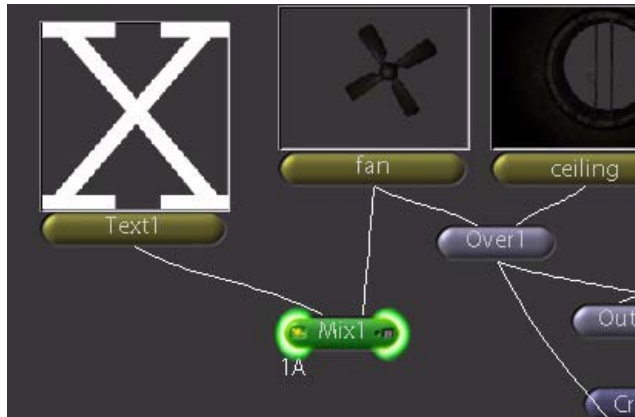


Note: To update the *Text* thumbnail in the Node View to reflect your changes, select the node and press R.

To mix the *Text* node with the *fan* node:

- 1 In the Node View, select the *Text1* node.
- 2 Attach a Layer-Mix node. This is a temporary node that will let you see the placeholder over the fan clip.
- 3 Connect the second input of the *Mix1* node to the *fan* node.

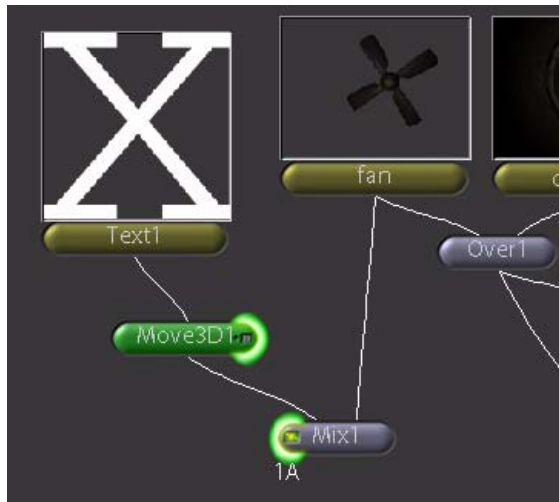
Because the fan is dark, boost the *fan* node brightness (with a *Brightness* node), or view the alpha channel. Because this subtree is temporary, do not insert it into the rest of the tree.



Next, match the fan animation. Start with the initial position.

To match the placeholder to the fan's position:

- 1 Go to frame 1.
 - In the Node View, select the *Text1* node and insert a *Transform–Move3D* node.



- 2 Expand the *Move3D1* subtrees for pan and angle, then set the following parameters:
 - Set *fieldOfView* to 15. (This parameter simulates the lens distortion of the *Move3D* “camera.”)
 - Set *xPan* to 153.
 - Set *yPan* to 87.

- Set xAngle to -75 .
- Set yAngle to -25 .
- Set zAngle to -195 .
- Leave center (x, y) at 80.

The result looks similar to this:



To animate the X element:

- 1 In the *Move3D* parameters, click the zAngle Autokey button to set a keyframe.



- 2 Go to frame 5.
- 3 Set the zAngle parameter to -122 .



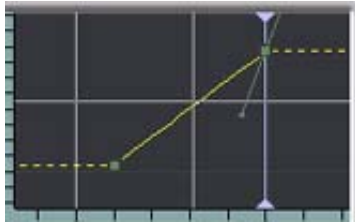
- 4 Use the Left Arrow and Right Arrow keys to step through the animation.

The animation matches between frames 1 and 5, but stops at frame 5. Use the Curve Editor to correct the problem.

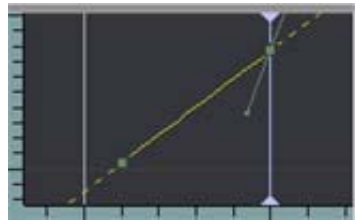
Note: When Autokey is enabled, the Load Curve button is activated, so the zAngle curve is already loaded into the Curve Editor.

- 5 Click the Curve Editor tab.
- 6 To select the curve, drag across the curve or select the curve in the curve list.
- 7 In the Curve Editor, change the curve Cycle mode from KeepValue (default) to KeepSlope.

Instead of a flat value before and after the two keyframes, the `zAngle` parameter keeps the same slope into infinity.



KeepValue mode



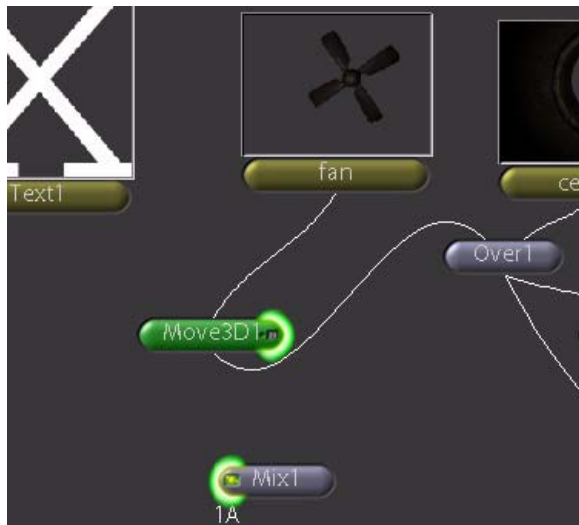
KeepSlope mode

- 8 Test the animation again with the Left Arrow and Right Arrow keys.
The animation continues and is matched beyond frame 5.

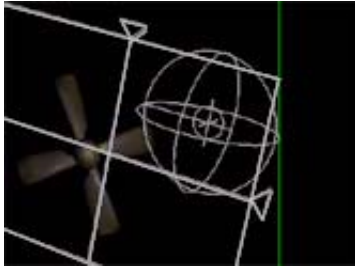
You are almost done. Honest.

To delete the *Text* node and apply the animation to the fan:

- 1 In the Node View, select the *Move3D1* node and press E to extract the node.
The node is pulled from the tree, but not deleted.
- 2 Attach *Move3D1* between the *fan* node and the *Over1* node.
- 3 Delete *Text1* and *Mix1*.



When you view *Move3D1*, the transformation moves the *fan* image, but that's not what you want. You want to use the *Move3D1* to generate motion blur only, so now you're going to tell Shake to use the input as a reference, but ignore the transform.



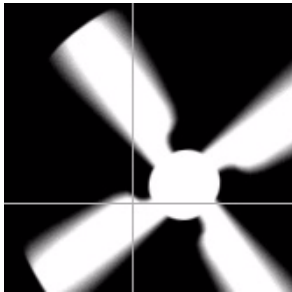
To apply fake motion blur to the fan:

- 1 In the *Move3D1* parameters, set *motionBlur* to 0.5.

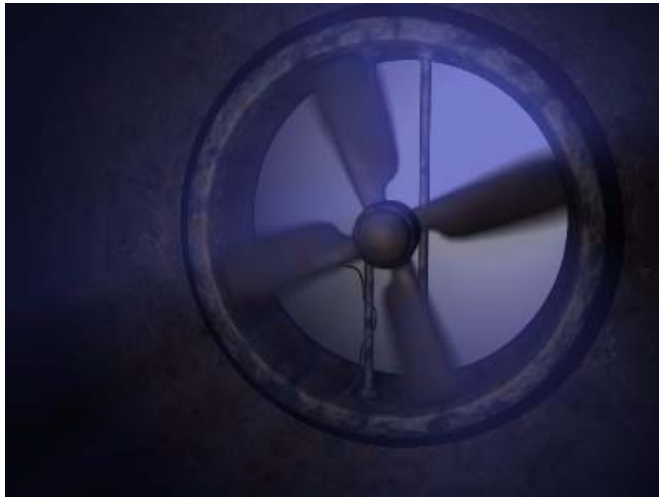
The result still is not correct—the *fan* is transformed.

- 2 At the bottom of the *Move3D1* parameters, enable *useReference*.

The *fan* snaps back to its original position and the motion blur is applied to the *fan*.



That's it. Now you've learned how to create simple volumetric effects and animation through expressions. Well done!



This tutorial covers the basics of using the Keylight node, including how to pull keys, apply masking, create holdout mattes, and perform spill suppression.



The *Keylight* node is a plug-in developed by Framestore CFC. In addition to pulling keys in the examples, you'll learn a few masking and color-correction tricks that can be applied to other nodes as well.

The first example in this tutorial uses images from the television miniseries, *Merlin*, courtesy of Lions Gate Entertainment. The second example uses images from the motion picture *The Saint*, provided by Framestore CFC and Paramount British Pictures Ltd.

Tutorial Summary

- Using the *Keylight* node to pull a key
- Testing the mask with a Viewer script
- Adjusting the mask with parameters
- Masking
- Color correcting the foreground image
- Removing blue spill
- Using a holdout mask

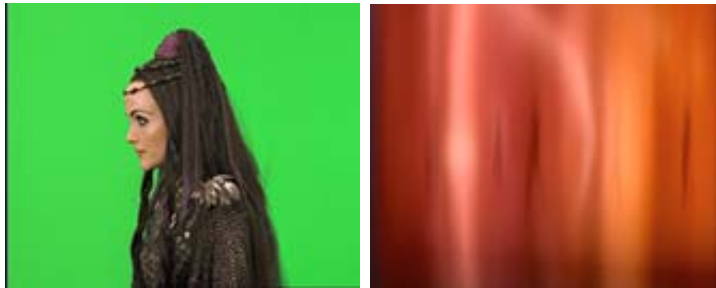
Using Keylight to Pull a Key

The first section of this tutorial demonstrates the basic steps for using the *Keylight* node.

To pull the key on the “merlin” image:

- 1 Add an *Image-FileIn* node, then browse to `$HOME/nreal/Tutorial_Media/Tutorial_05/images`.
- 2 Read in the *merlin_fg.jpg* and *merlin_bg.jpg* images.

Note: To read in both files simultaneously, drag in the File Browser window to select both files, then click OK.

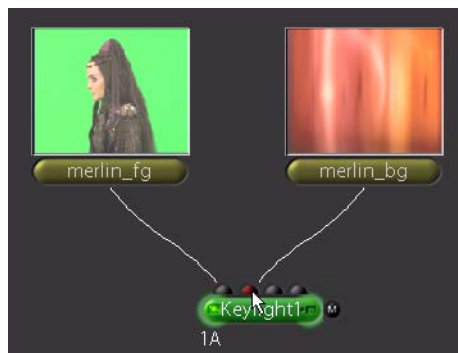


merlin_fg.jpg

merlin_bg.jpg

You are correct: That's not Merlin.

- 3 In the Node View, select the *merlin_fg* node, then add a *Key-Keylight* node.
- 4 Connect the *merlin_bg* node to the second input (the background input) on the *Keylight* node.



Keylight has several color controls. In this lesson, the important one is *screenColour*.

- 5 In the *Keylight* parameters, click the `screenColour` control (the blue-colored box), then drag the pointer across the lower portion of the greenscreen color in the Viewer.



The greenscreen color appears in the `screenColour`. (Although *Keylight* also works with bluescreen backgrounds, it does not work well with secondary colors such as cyan, magenta, or yellow.)

The result of your initial color pick is a fairly good key.



- 6 To view the alpha channel, position the pointer in the Viewer, then press A.



Testing the Mask With a Viewer Script

The quality of your mask cannot always be gauged visually for a number of reasons. For one, monitor default settings differ from system to system. Depending on your gamma settings, the your results match those shown in this tutorials. Therefore, when pulling a key, it's a good idea to test your results by applying a custom gamma lookup table.

To test the mask using a Viewer lookup table:

- 1 Click and hold the VLUT (Viewer Lookup Table) button in the Viewer shelf, then choose VLUT2 from the pop-up menu.



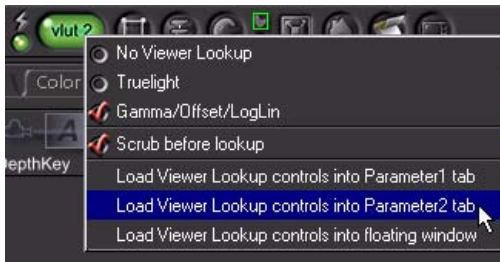
If you right-click the VLUT2 button, you'll see that the Gamma/Offset/LogLin VLUT is now active.



Note: There are only two Viewer lookup tables installed by default—Gamma/Offset/LogLin and Truelight—but you can add your own. For more information, see “Viewer Lookups, Viewer Scripts, and the Viewer DOD” in Chapter 1 of the *Shake 4 User Manual*.

You won't see a difference in the Viewer yet, because none of the three conversions—gamma, offset, or LogLin—is active when you first choose the VLUT2 option.

- 2 Right-click the VLUT 2 button, then choose Load Viewer Lookup Controls into Parameters2 Tab from the shortcut menu.



- 3 In the Parameters2 tab, boost the Viewer gamma level by typing “2.2” into the viewerGamma value field.



viewerGamma = 1



viewerGamma = 2.2

The heightened gamma level reveals noise in the image. If you lower the viewerGamma, holes are revealed in the mask. You probably don't see much until the value drops below 0.5, so don't worry about the holes for this composite. However, try to correct the background noise.

Primatte allows you to combine several pixel samples to allocate foreground and background color. *Keylight*, however, relies on sliders and masking to tune the relationship between foreground and background. In this example, there are two parameters to help you. (The possibility exists that your mask is fine, by the way.)

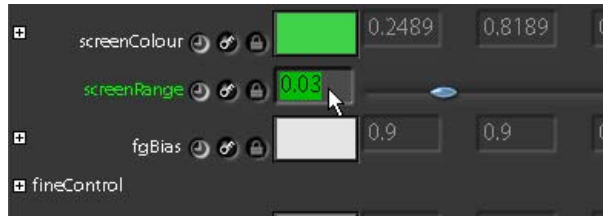
Adjusting the Mask With Parameters

The first parameter you can use to adjust the mask is screenRange. Raising the screenRange value increases the contrast in the mask. This slider is typically used for garbage masks that are later fed into other masks, as it tends to remove fine detail.

To test the effect of adjusting the screenRange values:

- 1 In the Viewer shelf, click the VLUT2 button to deactivate the lookup table.
The VLUT2 button changes to VLUT Off.
- 2 Ensure that the *Keylight* node is selected in the Node View.
- 3 In the *Keylight* Parameters1 tab, set screenRange to 0.3.
The mask is filled, but the edge quality degrades.

- 4 Set screenRange to 0.03.



The edges soften, and the mask is filled in, but some of the hair detail drops off.

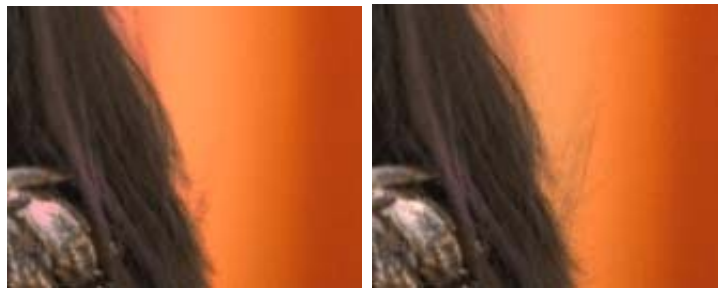
- 5 Return the screenRange value to 0.

You can see that this parameter can be clumsy. The second set of mask opacity controls are the Gain parameters found in the fineControl subtree. Raise or lower the Gain parameters to increase or decrease the mask in the shadow, midtone, or highlight areas.

- 6 In the Parameters1 tab, click the plus sign (+) next to fineControl to expand the fineControl subtree.
- 7 Set the highlightGain subparameter to 0.4.

The noise disappears. (You can turn on the VLUT2 button to see the result.) Unfortunately, the Viewer reveals that hair detail has been removed. This is bad.

- 8 To restore the hair detail, set highlightGain back to 0.



highlightGain = 0.4

highlightGain = 0

Note: To toggle the Viewer between the alpha and color channel views, position the pointer in the Viewer and press C (for color) or A (for alpha).

So right about now, you may be thinking that these parameters are useless and ought never be touched. To the contrary, these parameters are often useful, but this example reveals problems you must consider when keying—noise and edge detail. In this case, another approach should be taken, which is the use of masks. Hair is one of the hardest keys to pull. Filling in the inside of the mask is one of the easiest things to do. Therefore, it is better to use masks to fill in holes instead of risking losing hair detail.

Masking

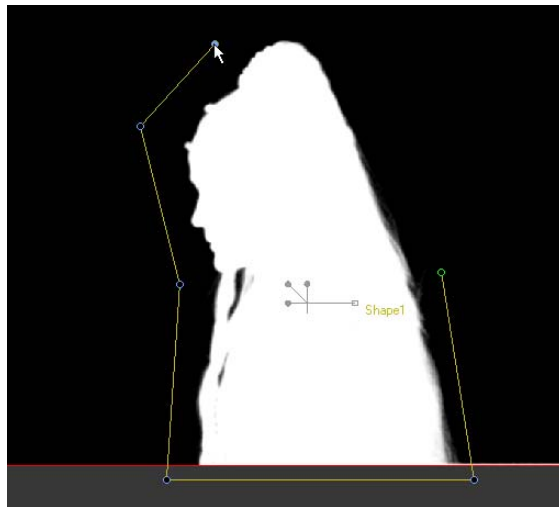
In this step, draw a shape to get rid of most of the noise and still maintain the hair detail. This mask is then also used to get rid of the line on the left side of the image.

To add a rotoShape:

- 1 Display the alpha channel view in the Viewer.
- 2 From the Image tab, Shift-Control-click the *RotoShape* command button.
This inserts the node without connecting it to anything in the node tree.
- 3 Load the *Keylight1* node into the Viewer (click the left side of the node), then load the *RotoShape1* parameters into the Parameters1 tab (click the right side of the *RotoShape1* node).

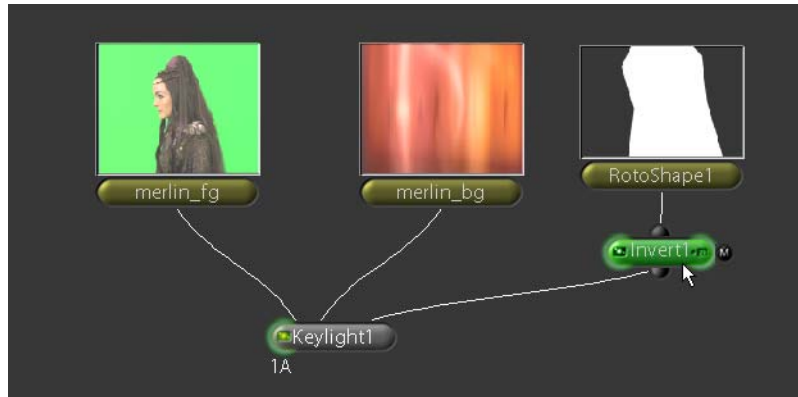
Rotoshape controls appear in the Viewer shelf, and the Add Shapes button is enabled by default.

- 4 Click in the Viewer to draw a loose rotoShape around the figure.

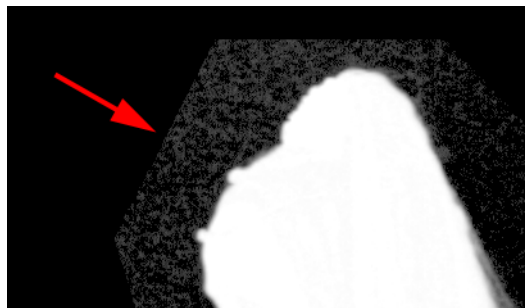


- 5 Click on the first point to close the shape.

- 6 Connect the *RotoShape* node to the fourth input (GarbageMatte) of the *Keylight* node to remove the gunk on the side of the image.
Now the rotoShape matte must be inverted.
- 7 In the Node View, select the *RotoShape1* node.
- 8 Apply a *Color-Invert* node to the *RotoShape* node.
The mask is reversed.

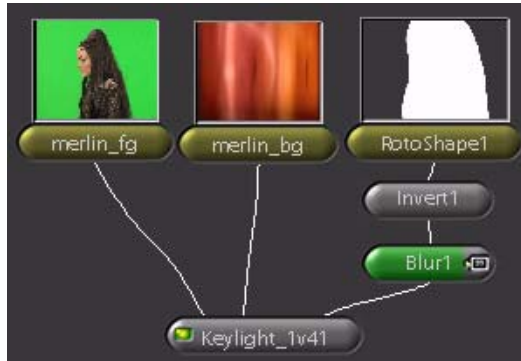


- 9 Activate the VLUT2 button.
Hold down the button, then choose VLUT2 from the pop-up menu. You do not have to modify the Viewer lookup values; they are saved for the session.
When VLUT2 is enabled, you'll see an edge on the mask. You can soften this with a *Blur* node.



- 10 Select the *Invert1* node and insert a *Filter-Blur* node.

The *Blur* node is faster than using the *RotoShape* feathering.



- 11 In the *Blur1* parameters, adjust the values of the xPixels and yPixels.

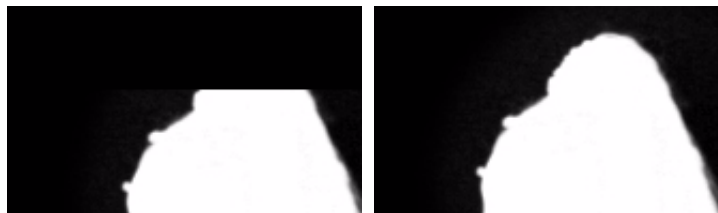
As soon as the blur is activated, the top of the head gets clipped off. What's up with that? The *RotoShape* node is 486 pixels high by default, while the other images are 540 pixels. Be aware that *Blur* is the only function that let's you work with the Infinite Workspace on or off. By default, it is turned off so you can blur full-frame images without creating a black edge on the border.

- 12 Click the spread button to toggle the *Blur* spread parameter to Outside Frame.



The Infinite Workspace is enabled and the clipping is removed.

Note: You can also adjust the *RotoShape1* height parameter to 540 (in the *Rotoshape1* parameters). As always, there are many ways to correct things in Shake.



spread = In Frame Only

spread = Outside Frame

Color Correcting the Foreground Image

To venture away from keying for a moment, this section discusses color correcting the foreground. Since the woman is composited into a red room, cast the woman with a red hue. You do not want to color correct *after* the *Keylight* composite, because that would alter the background. Likewise, you cannot color correct *before* the *Keylight* node, because that would change the greenscreen. You seem to be in a pickle. Fortunately, the *Keylight* node handles this problem in two ways.

First, *Keylight* contains built-in color correction tools, for exposure (the equivalent of the Shake *Mult* node), gamma, and saturation. Feel free to use these tools if they are sufficient for your color correction. An advantage of the built-in tools is that they are calculated into the keying lookup table, so you have a slight speed advantage over using extra nodes. However, to do other operations, such as transformations or color-correction curves, you need to do something different.

It is very rare that you read in your elements, key, and composite with only one node. To apply other effects to the foreground, you must break the compositing out of the keying operations and use *Over* nodes instead. In the *Keylight* parameters, notice that there are several output buttons.



The fourth button from the left is called “unpremult”—an unpremultiplied image. With this button selected, the key is knocked out of the foreground, but is then divided by its mask. This setting allows you to apply color corrections before compositing over the background with an *Over* node. If you are only applying filters or transforms, set the output parameter to “on Black.”

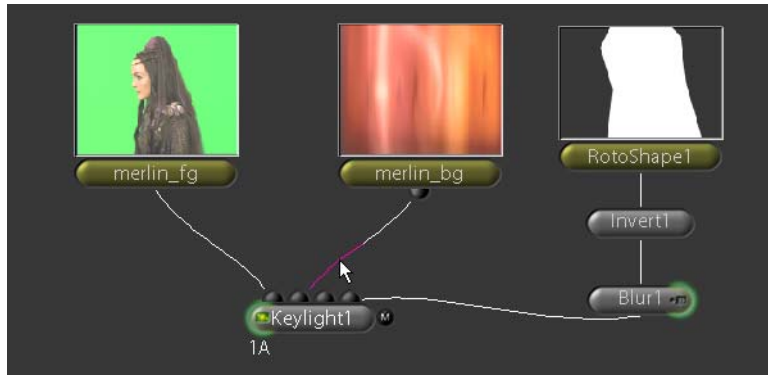
To perform color correction, but not compositing, in the *Keylight1* node:

- 1 In the *Keylight1* parameters, set output to “unpremult.”



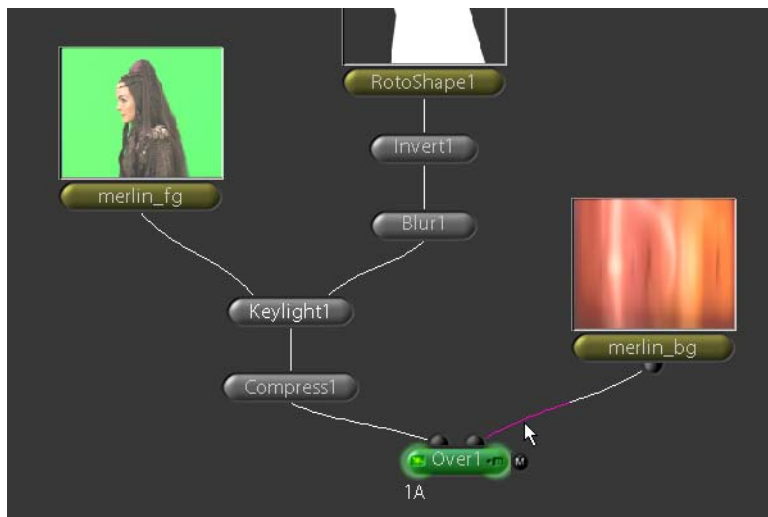
When the output is switched to “unpremult,” the composite looks terrible. This unpremultiplied state is temporary, until you composite with an *Over* node.

- 2 To remove the background image input from *Keylight1*, Command-click or Control-click the noodle.



Now use a *Compress* node with an *Over* node to re-composite the background.

- 3 In the Node View, select the *Keylight1* node.
- 4 Attach a Color-*Compress* node.
The *Compress* node is used for the color correction.
- 5 Attach a Layer-*Over* node to the *Compress1* node.
- 6 Connect the *merlin_bg* image to the second input of the *Over1* node.



Here is the crucial trick:

- 7 In the *Over1* parameters, activate *preMultiply*.
The nasty gunk around the character is removed.



For the color correction:

Once again, you are matching a foreground image to a limited background image.

- 8 Load the *Compress* parameters by clicking on the right side of the node.



- 9 In the *Compress* node parameters, click the Low Color control and scrub in the dark corner of the background.
The low color is selected.
- 10 To select the high color, click the High Color control and scrub on a bright portion of the background.



Note: In the Color Picker, Use Source Buffer is off. This is good, since you want to scrub color from the composite rather than from the unmultiplied image coming out of *Keylight*.

Although this is an extreme color correction used for illustration purposes, the principal concept here is that you spit out an unpremultiplied image, do your color corrections or transformations, then composite with preMultiply enabled. You can alternatively use On Black for your output from *Keylight*, apply a Color-*MDiv*, do your color corrections, then apply a Color-*MMult* to get the same result, but this complicates the tree.

Testing a Color Correction

The following is a handy trick. Use the viewer lookup table (VLUT2) to gauge the color correction. If it is a good correction, the foreground blacks should be somewhat similar to the background blacks when the gamma is boosted, and the brights should be similar when the gamma is lowered below 1.

To evaluate the color correction:

- 1 In the Viewer shelf, turn on the VLUT2 button.
- 2 Right-click the VLUT 2 button, then choose Load Viewer Lookup Controls into Parameters2 Tab from the shortcut menu.
- 3 In the Parameters2 tab, adjust the viewerGamma slider from .1 to 4 to evaluate the correction.



A Bad Correction

Highlights Are Good

Lows Are Good

If things look good, turn off the VLUT2 button.

You have now pulled a basic key using *Keylight* and tested some of its parameters to adjust the mask. Since the parameters were not sufficient to remove the grain of the image but keep the hair detail, a garbage mask was created, thereby removing most of the noise. Afterward, a color correction was applied. You used a viewer lookup table (VLUT) to test both the key and the color correction.

Advanced Keylight Techniques

This part of the tutorial focuses on more *Keylight* parameters and also discusses the use of anamorphic images.

For the following example, use the five foreground and background images (at video resolution) in the `$HOME/nreal/Tutorial_Media/Tutorial_05/images` directory.

To begin the composite:

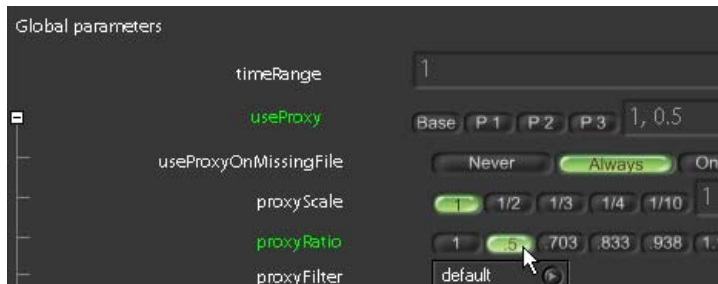
- 1 Add an Image-FileIn node, then load the *saint_fg.1-5#.jpg* and *saint_bg.1-5#.jpg* image sequences.

The Joy of JPEG

Never use JPEG images for any footage you have to key. The compression causes major problems, as you will find in this tutorial. Also, do not use JPEG images for video interlaced footage, as the fields become corrupted.

The first thing you notice is that these are anamorphic frames, so everything is squeezed. Additionally, the element is originally a scanned Cineon plate, so it is in logarithmic color space. First, focus on the aspect ratio issue.

- 2 In the Globals tab, open the useProxy subtree and set the proxyRatio to 0.5.



Note: You can also set the proxy in the useProxy value field as 1, 0.5.

The images are squeezed by half in the Y axis.



proxyRatio = 1



proxyRatio = 0.5

This squeezing can be turned on and off very quickly with the proxy button in the upper-right corner of the Shake interface. "Other" (highlighted in green) indicates that you are using a custom proxy setting rather than a preset (P1, P2, P3). The proxy scaling is just temporary while you work with the script. When finished with the composite, set the proxyRatio back to 1.

- 3 In the Node View, select the *saint_fg* node and apply a Key–Keylight node.
- 4 Connect the *saint_bg* node to the second input (background) of the *Keylight1* node.

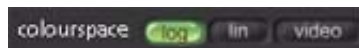


- 5 In the *Keylight1* parameters, click the screenColour control and scrub the back window of the car. This is a good place to scrub because there are no reflections (like the side window).



The lesson images are logarithmic (that is, filmic, rather than linear digital or video) color-space images. Don't fret—you can specify the color space the node is working in in the *Keylight1* Parameters tab.

- 6 Set colourspace mode to “log” for this composite.



Note: All masking data is dependent on your monitor gamma. Use the viewer lookup table to ensure you have an accurate key as you proceed with the tutorial.

Using fgBias to Remove Blue Spill

Some blue spill appears on the actor's hair, on the right side.



Since the blue spill is subtle, you can linearize the “log” plates to return them to something resembling the real world.

To linearize the log plates:

- 1 In the Viewer shelf, turn on the VLUT2 button.
- 2 Right-click the VLUT 2 button, then choose Load Viewer Lookup Controls into Parameters2 Tab from the shortcut menu.
- 3 In the Parameters2 tab, set the viewerLogLin parameter to LogToLin.
- 4 The viewerGamma value should be set between 1 and 1.5.

To minimize the blue spill:

- 1 In the *Keylight1* parameters, click the fgBias color control; then, in the Viewer, drag along the actor's hair (the blond part, not the blue spill part).



Scrubbing Ms. Shue's hair



With spill suppression

The blue is removed. (A ringing effect may occur due to the JPEG compression on the foreground element. Nothing you can do about that for this tutorial. Sorry.)

To minimize any yellowness that may occur in the image, you can decrease the fgBias saturation. The stronger the saturation of the fgBias color, the stronger the yellow tint.

- 2 Position the pointer over the fgBias color control, press and hold S, then drag left.
The saturation of the color is decreased.



You can also set the balance parameters to 1 to reduce the yellow tint. It does not reduce all of the yellow—her scarf and map may still retain some yellow—but it helps. The yellow color correction is further adjusted in the “Using the replaceColour” section, below.

Using a Holdout Matte

Show the alpha channel in the Viewer (position the pointer in the Viewer and press A), and you are no doubt appalled at what fgBias has done to the matte. Return to the color view (press C), and notice the snow on the road is visible behind the seat belt.

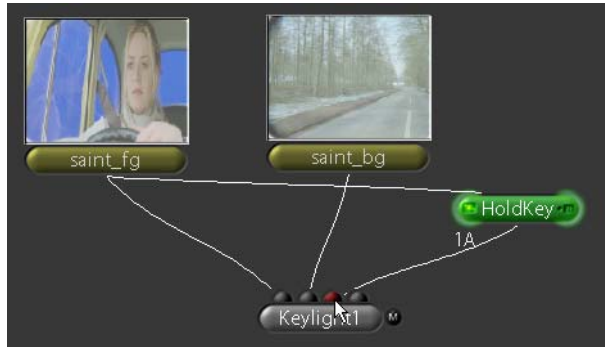


A common technique to fix this problem is to copy your first *Keylight* node, boost its contrast, reduce the size of the mask by “chewing into it,” then feed it back into the first *Keylight* node.

To create a holdout matte:

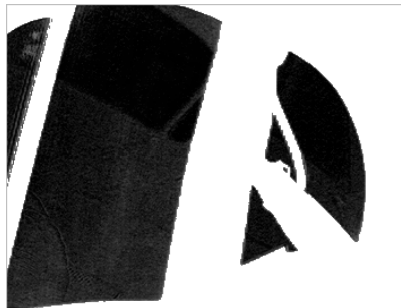
- 1 Clone the *Keylight1* node:
 - a Select the node, then press Command-C or Control-C to copy it.
 - b Press Command-V or Control-V to paste the cloned node.
- 2 In the Parameters tab of the cloned node, enter a new name: *HoldKey*.
- 3 Connect the *saint_fg* node to the first input of the *HoldKey* node.

- 4 Connect the *HoldKey* node to the third input of the *Keylight* node.

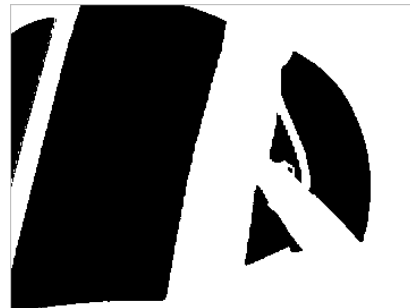


- 5 In the *HoldKey* parameters, boost the *screenRange* to 0.3.
The mask becomes solid.

When the holdout matte is plugged in, it acts as a mask for the foreground. You are, of course, not obliged to use *Keylight*. You can also create a holdout matte using other nodes, including *RotoShape*, *Primatte*, *ChromaKey*, *LumaKey*, and so on. The following illustrations show the *Keylight1* alpha channel and the *HoldKey* alpha channel. *HoldKey* contains no gray tones; *Keylight1* retains the window reflections.



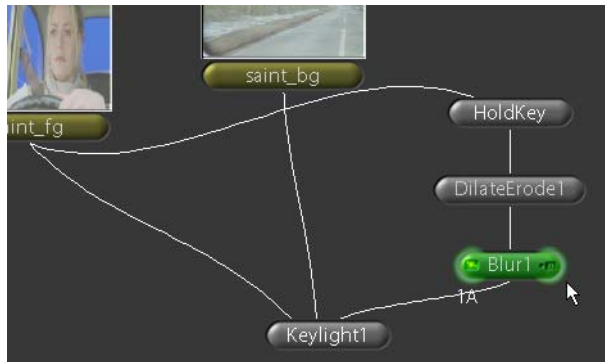
Keylight1 alpha



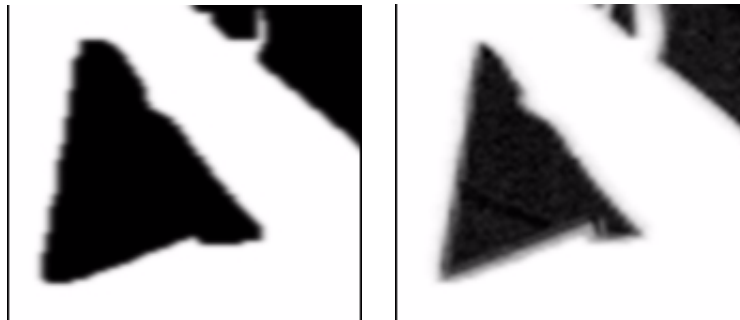
HoldKey alpha

The holdout matte has destroyed the edge quality of *Keylight1*. To correct the edge quality, “chew into the edge” of the holdout matte.

- 6 Insert a Filter-*DilateErode* node and a Filter-*Blur* node between the *HoldKey* and *Keylight1* nodes.



- 7 In the *DilateErode1* parameters, set xPixels and yPixels to approximately -3 to chew into the matte.
- 8 In the *Blur1* parameters, set xPixels and yPixels to approximately 6 to soften the matte.



HoldKey matte with dilate and

Keylight

The holdout matte strengthens the normal key pulled by *Keylight1*. Because the *DilateErode1* node reduces the size of the *HoldKey* holdout matte, the matte does not interfere with the edges.

To test the effect of the holdout matte:

- In the lower portion of the *Keylight1* parameters, expand the plumbing subtree, then enable `useHoldOutMatte`.



Without holdout matte



With holdout matte

Using the `replaceColour` Parameter

You can also use the holdout matte with the *Keylight* `replaceColour` parameter to help eliminate the yellowish tint. This is for areas that are keyed out by mistake because they are lit with more blue light than white light, and are also included in the holdout matte area.

To help correct the lighting with `replaceColour`:

- 1 In the Viewer, show the composite (press C with the pointer in the Viewer).
- 2 In the *Keylight1* parameters, click the `replaceColour` color control.
- 3 In the Viewer, select a color that is the opposite of the warm yellow—a mid-saturation purple.

In the following split illustration, the uncorrected yellowish image is on the left. The image with the replace color is on the right.



As an end note, like all keyers, *Keylight* is not a magic bullet—a key rarely works with a single pixel scrub. Feel free to use *Keylight* in conjunction with other mask techniques and keyers, and combine the keys through the use of the *KeyMix* function, or through the addition or subtraction of mattes with tools such as the *IMult*, *IAdd*, *Outside*, *Inside*, *KeyMix*, and *Max* nodes.

This lesson describes the basic use and mechanics of the Photron *Primatte* keying plug-in, as well as masking and spill suppression.

Tutorial Summary

- The basics of pulling a key in *Primatte*
- Inner mechanics of *Primatte*
- Masking *Primatte*
- Spill suppression in *Primatte*
- Alternatives

The Basics of Pulling a Key in Primatte

The example images for this lesson are located in the `$HOME/nreal/Tutorial_Media/Tutorial_06/images` directory.

Important: Before you begin this tutorial, open the Globals tab and make sure that the `proxyRatio` parameter is set to 1.

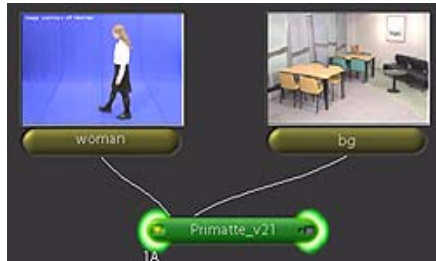
To pull the *Primatte* key:

- 1 Add an Image-FileIn node and read in the *woman.iff* and *bg.jpg* images.



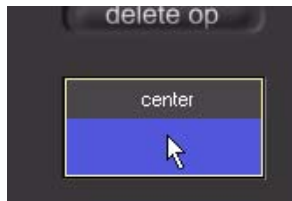
- 2 In the Node View, select the *woman* node.

- 3 Add a *Key-Primatte* node, then connect the *bg* node to the second input (background) of the *Primatte* node.



In *Primatte*, you select a “center” value that is the average color of the key you want to pull. (The concept of center is explained in “[Inner Mechanics of Primatte](#)” on page 186.) Unlike *Keylight*, *Primatte* is ready to scrub your background color as soon as you add the node.

- 4 In the *Primatte1* Parameters tab, click the center color control to select it, if it’s not already selected (a yellow border appears around the center box).



- 5 In the Viewer, drag across the bluescreen in the background.



When you drag (or *scrub*) over the background, *Primatte* calculates an average (or *center* sampling) of the scrubbed pixels. Note that in the Viewer, nothing appears to change. This is because, by default, only the alpha channel is modified, as indicated in the *Primatte* output parameter.

- 6 To view the alpha channel, position the pointer in the Viewer and press A.

The alpha channel reveals the basic key. The majority of the background should appear black in the alpha channel.

Note: If most of the background is medium gray, the center control was not selected when you scrubbed for the bluescreen color. Make sure the center control is active, then scrub the bluescreen again.

Output, one of the first parameters in *Primatte*, determines what is passed out of the node. The output is set to “alpha only” by default.



Primatte contains the following output parameters:

- *alpha only*: Only the alpha channel is modified; no spill suppression is performed.
- *on Black*: Foreground is composited over black and spill suppression is performed.
- *comp*: Foreground is composited over the background image with spill suppression.
- *status*: Displays *Primatte*'s pixel analysis, allowing you to see how pixels are being processed under the parameter settings. In this reference mode, pixels are displayed in four categories or *zones*, represented by the following (arbitrary) colors:
 - Black represents pure background.
 - Red represents pure foreground.
 - Blue represents transparent areas.
 - Green represents transparent areas for spill suppression.

For more information about status color zones, see [“Inner Mechanics of Primatte”](#) below.

To set the output mode to view the composite:

- In the *Primatte* output parameters, enable “comp.”

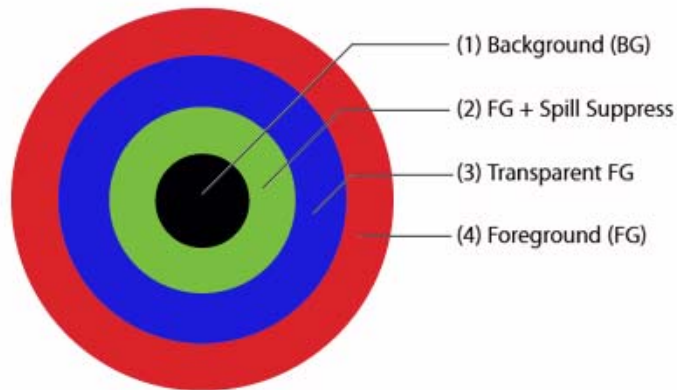


The composite appears in the Viewer.



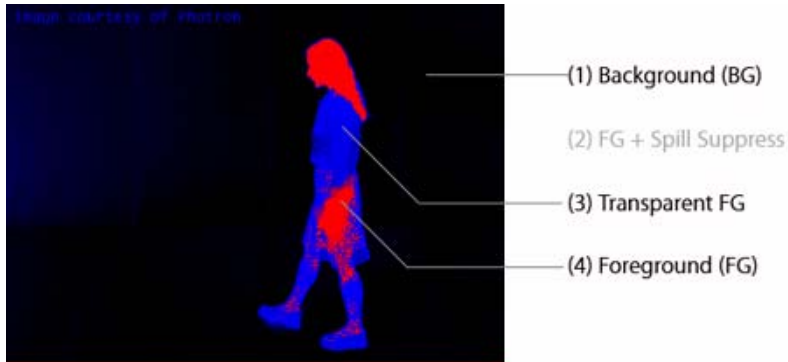
Inner Mechanics of Primatte

So, what is Primatte doing to the image pixels? Whereas *Keylight* is based on a model of light and its properties, *Primatte* works by assigning color to one of four different zones through a series of color scrubs. Each zone has its own qualities. These zones are constructed in a 3D space created with the three RGB axes. The following illustration is a 2D representation of this 3D space.



To view the color zones in the composite:

- Click the “status” button in the *Primatte* output parameter.



As you can see, the pixels of the bluescreen—ideally, any pure blue pixels—fall inside zone 1 (Background).

Note: While we’re talking about blue, don’t confuse the blue pixels from the keyed image with the blue color-coding that *Primatte* uses in status mode to display pixels assigned to zone 3. The color zones in status mode are merely a way to show you how *Primatte* is allocating pixels in a composite.

Pixels *close* to pure blue—but not pure blue—fall inside zone 3 (Transparent FG). Spill suppression colors are assigned to zone 2. The rest of the colors are in zone 4 (Foreground).

To return to composite mode, click the “comp” button in the *Primatte* output parameter.

To assign a color in the image to one of the different zones, click an operator button—background, foreground, spill sponge, and so on—then scrub the color in the keyed image. In the current image, for example, you can see that the initial key operation has made parts of the woman’s shirt transparent; you can use the foreground operator to “pull” these pixels back into the full opacity of the foreground.

To add more of the shirt to the foreground zone:

- 1 With the pointer over the Viewer, press A to view the alpha channel.
- 2 In the *Primatte1* parameters, click the foreground operator button.

The foreground operator is assigned and a new color control is automatically activated.



- 3 In the Viewer, scrub the shirt area.



Note: The Undo function and *Primatte* are not best friends. To remove an operator, use the delete op button in the *Primatte* parameters. To pick a different color, click the color control (under the delete op button), then scrub in the Viewer.

- 4 To sample more foreground area, click the foreground button again and scrub in the Viewer.
- 5 To re-scrub, ensure that the color control is active, then scrub again.

When using *Primatte*, knowing when to stop is important. The more scrubs you perform with the foreground and background operators, the crunchier the matte edge becomes (this is bad).

For example, you cannot eliminate the dark area under the woman's arm. Kind of like that nasty corner in your shower, you can scrub on it all day but it doesn't disappear. This is a clue that this particular color is buried way down in zone 1 (pure background). If you try to assert this color as foreground, you are going to upset the entire keying model (that is also bad).

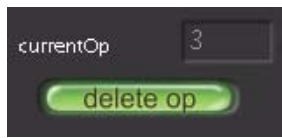
When scrubbing fails, it may be time to apply some rotoscoping to the key. In this case, a holdout matte will work better to mask problem areas such as the dark portion under the woman's arm. This technique is covered in the "Masking Primatte" section below.



If you tried to remove the dark area with a foreground operator and failed miserably, delete the operator.

To delete an operator:

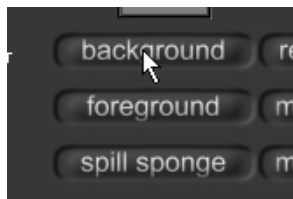
- In the *Primatte1* Parameters tab, click the delete op button, below the currentOp parameter.



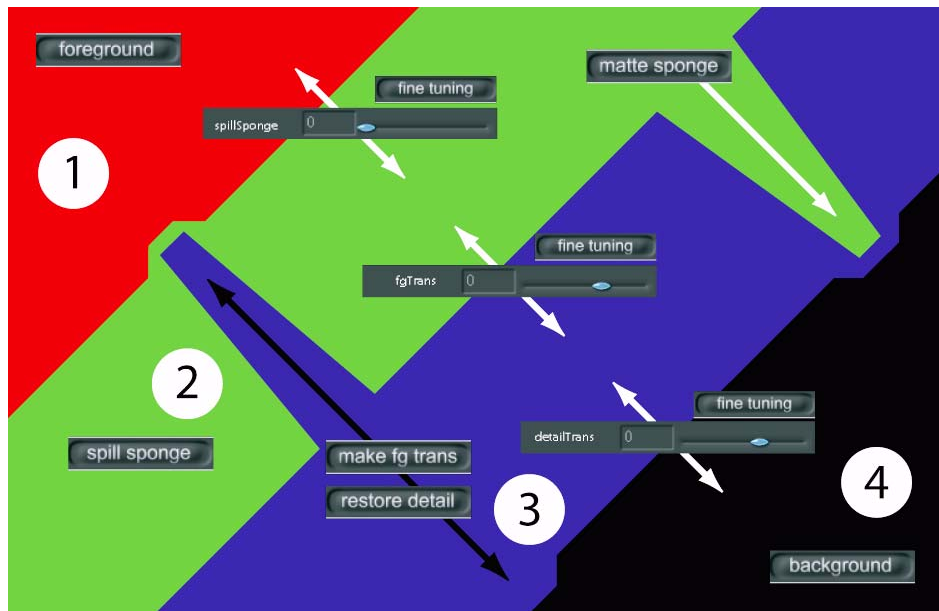
Note that the background of the key, especially on the left side, needs work.

To modify the background of the key:

- In the *Primatte1* Parameters tab, click the background operator button, then scrub the left side of the image.



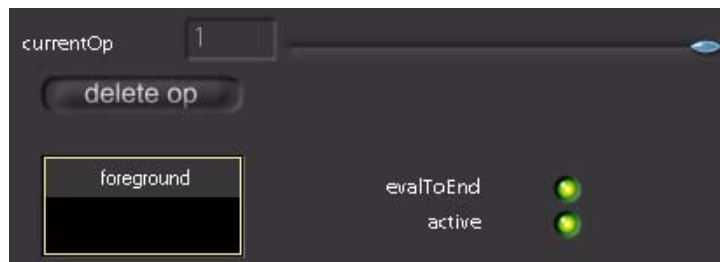
By using the foreground and background operators, you have assigned pixels to either zone 1 or 4. Picking any of the eight operators assigns color to a different zone. The following image is a chart of the operator/zone assignments.



This illustration demonstrates (roughly) how each operator dimples the surface of the zones through the use of successive color picks. You may have seen implementations of *Primatte* in other software using a real 3D representation of the color space using polygons. While cool, only about three people on the planet seem to understand it.

Managing Operators

One swell thing about Shake's implementation of *Primatte* is that you can review, modify, or delete previous operators at any time. You do not have to start over from scratch if you are not happy with the key. Use the parameters shown in the following illustration to review, modify, or delete previous operators.



The currentOp slider scrolls from 0 (the Center pick) to the number of applied operators. Each operation can be accessed by using the slider. The Center pick is always 0, and cannot be deleted.

To modify previous operators:

- Use the currentOp slider to select the operation you want to modify. Operations are numbered in the order applied. For example, if you applied a foreground operation, a second foreground operation, and then a background operation, the first foreground operator is 1, the second foreground operator is 2, and the background operator is 3.
- Enable evalToEnd to view the effect of all applied operators. When disabled, only the operations up to the currentOp are displayed. For example, if you have 10 operators and your currentOp is set to 3, only operations 0 through 3 are calculated by *Primatte*.
- Use the “active” parameter to quickly toggle an operation on and off to test its effect.
- To re-scrub a color, click the color control and scrub in the Viewer.
- To remove an operator, press the delete op button.

Masking Primatte

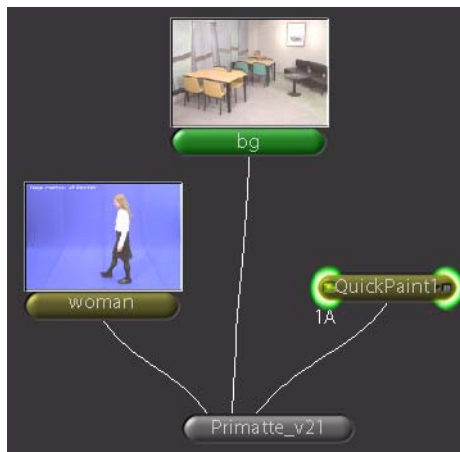
In this section, use a holdout matte to clean up the key under the woman’s arm, and a garbage matte to eliminate the “image courtesy of Photron” text. Holdout mattes add to opacity, garbage masks add to transparency.

To attach a garbage mask:

- 1 Click the Image tab, press Command-Shift or Control-Shift, then click the *QuickPaint* command button.

An unattached *QuickPaint1* node is created.

- 2 Connect the *QuickPaint1* node to the third input (garbageMatte) of the *Primatte* node.



- 3 Show the composite (*Primatte* node) in the Viewer (click the left side of the node).
- 4 Make sure the *QuickPaint1* node parameters are loaded (click the right side of the node).
- 5 In the *QuickPaint* controls in the Viewer shelf, click the Hard/Soft brush button (this is the default) to toggle the selection to Hard.



- 6 Drag the brush across the text to “erase” it.



Note: As an alternative, you can connect the *QuickPaint1* node between the *woman* node and the *Primatte1* node, then paint over the text using a color selected from the bluescreen. *Primatte1* then automatically makes that area invisible as it keys out the blue color.

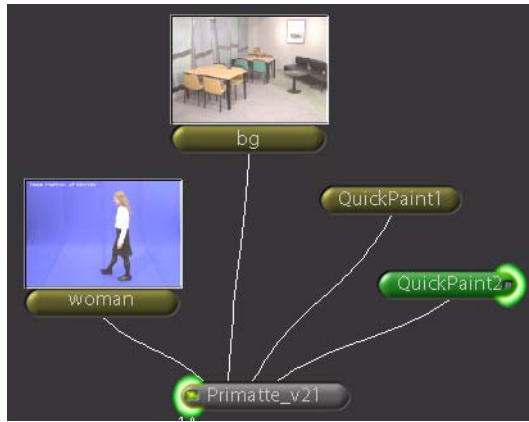
To select the channel for the garbage matte, use the *gMatteChannel* settings in the *Primatte1* parameters.

Attach a Holdout Matte

Use a holdout matte to promote areas that should be visible, but are keyed out because the color is close to the bluescreen or greenscreen center color. For example, use a holdout matte to eliminate the line under the woman’s arm (since the foreground operators could not completely eliminate it earlier).

To create a holdout matte:

- 1 Add a second *QuickPaint* node (a *RotoShape* node would also do the job), and connect the node to the fourth input (holdOutMatte) of the *Primatte1* node.



Note: To identify inputs on the top of a node, position the pointer over the input and read its name in the Info field at the bottom-right portion of the interface.

- 2 Using the brush in the *QuickPaint* node, paint a holdout matte along the line under the woman's arm.



- 3 In the *Primatte1* parameters, set the hMatteChannel masking channel for the holdout matte.

In this case, hMatteChannel can be set to any of the channels—r, g, b, or a—because the paint strokes of the *QuickPaint* node appear in all channels. However, if you use an image that does not have an alpha channel, or you want to use a specific channel, make sure the channel you want to use is selected for the hMatteChannel.

A final parameter that helps with multiple masks is the arithmetic parameter. The arithmetic parameter indicates how the *Primatte* key interacts with the alpha channel of the foreground.

By default, *Primatte* replaces the alpha channel, but you can also add, subtract, or multiply the two masks. With this technique, you can create several masks and combine them in *Primatte*. Since there is no incoming mask in this example, the arithmetic parameter has no effect.

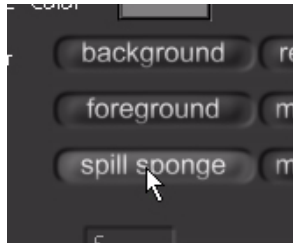
In this example, there are three primary areas where the blue spill is apparent: The edge of the woman's hair, the shadows on her shirt, and possibly the little transparent bit of her skirt at her knees. Each time you use the foreground operator, you create more areas that you must spill suppress. On the flip side, the fewer foreground operators you use, the more rotoscoping or paint touch up you must do. An alternate—and common—way to handle this is to target specific areas of the image by pulling multiple keys with separate nodes, then combine the keys into one node.

Spill Suppression in Primatte

Spill suppression is the removal of reflected light (blue or green spill) on the foreground material that comes from the bluescreen itself. In this example, blue spill is immediately evident on the woman's white shirt. Although the suppression tools in *Primatte* are interesting and useful, as a general principle you are encouraged to perform the suppression outside of the *Primatte* node using nodes such as *HueCurves*, *SpillSuppress*, and *ColorReplace*. After all, why would you want to tackle two difficult tasks at once?

To use the spill sponge operator:

- 1 In the *Primatte1* Parameters tab, click the spill sponge operator button.



- 2 In the Viewer, drag along the purplish edge of the woman's hair.



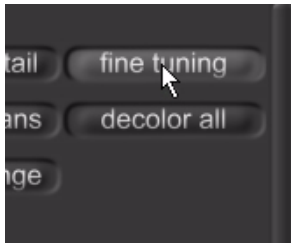
- 3 You may need to apply two or three spill sponge operators. The spill sponge operator applies a precalculated amount of spill suppression, and replaces the blue color with an alternative color.

For the spill on the shirt, you *could* use additional spill sponge operators. But, for this example, use the fine tuning operator instead.

The fine tuning operator combines the functions of the spill sponge, restore detail, and make fg trans operators. The restore detail operator brings back fine details such as individual hair strands. The make fg trans operator pushes foreground material into the slightly transparent zone. The fine tuning operator is much easier to use than the spill sponge, restore detail, and make fg trans operators, because it has sliders to precisely tune the combination of the three effects.

To use the fine tuning operator:

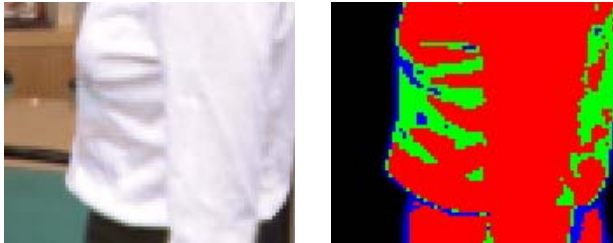
- 1 In the *Primatte1* Parameters tab, click the fine tuning operator.



- 2 In the Viewer, drag along the bluish part of her shirt.



- 3 Adjust the spillSponge slider (under the color control).
The more you slide, the greater the range of suppressed area.
- 4 To view the affected areas, toggle output to “status,” then move the spillSponge slider up and down.
The green zone (spill suppression) increases and decreases in size.

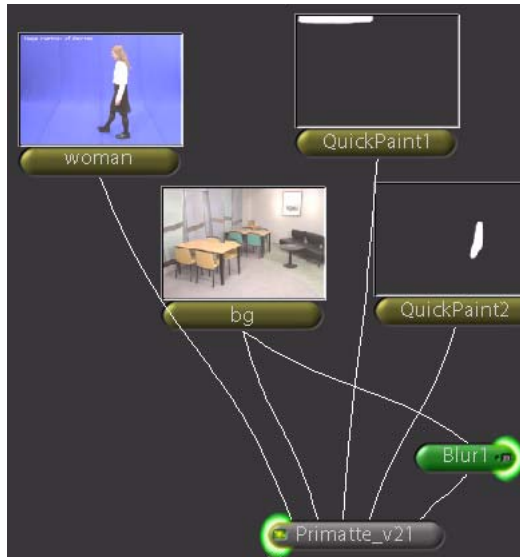


The spill-suppressed image appears slightly transparent when you return the output operator to comp. This is because *Primatte* pulls in background color to replace the blue color. Here is the logic: If you stand in front of a bluescreen and you have blue spill, then conceivably if you stand in front of a red wall, you have red spill. Therefore, *Primatte* sucks in the background image color to replace the removed bluescreen color. This is sometimes sufficient, but often needs a little help. If the definition of the background is too sharp (for example, the black top of the chair appears through her shirt), you must modify the image from which *Primatte* pulls the replaced color. By default, *Primatte* pulls it from the background image. This can be replaced by inserting an image into the sixth input on the *Primatte* node.

To control spill using the *replacesImage* input:

- 1 In the *Primatte1* parameters, set the output to comp.
- 2 In the Node View, select the *bg* node.
- 3 Shift-click the Filter-Blur node button.
- 4 In the *Blur* parameters, set the xPixels and yPixels to 200 (an arbitrary value in this case).

- 5 Connect the *Blur* node to the *replaceImage* input (the last input) on the *Primatte1* node.



Since the black chair backing has lost its definition, the shirt appears less transparent. The alpha channel remains solid.



An alternative to spill suppression is to set the *replaceMode* parameter to “use color,” which fills the spill areas of the image with a solid color specified in the *Replace Color* parameter.

To control spill with the “use color” *replaceMode* setting:

- 1 Make sure the composite is loaded in the Viewer.
- 2 In the *Primatte1* parameters, toggle *replaceMode* to “use color.”
- 3 Click the *Replace Color* control, then scrub the color of the wall.

Under the right circumstances, this can reduce blue or green spill quite nicely. Under the wrong circumstances, it can create a fuzzy halo around your edges—so use with caution.

- 4 Set the `replaceMode` parameter back to “use image.”

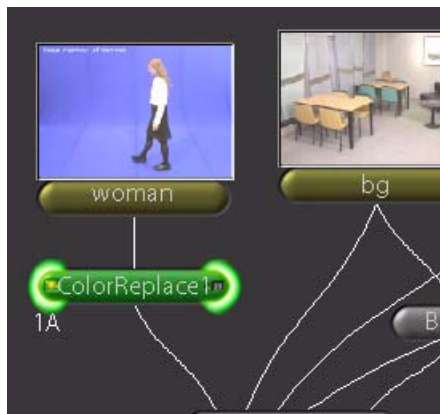
You can also remove blue (or green) spill, *before* keying, by replacing it with a color from your background. Other nodes that can remove spill suppression include *Key-SpillSuppress*, *Color-AdjustHSV*, *Filter-Blur*, or *Color-Monochrome* applied to the foreground image.

As you can see, there are many different ways to handle keying and spill suppression. One popular trick is to place a *Color-Monochrome* node on the foreground image and connect it as the `replaceImage` instead of the *Blur* node shown earlier.

The next set of steps show how to use a *ColorReplace* node to replace the blue spill on the woman's shirt with a color from the background image.

To use *ColorReplace* for spill suppression:

- 1 Insert a *Color-ColorReplace* node between the *woman* node and the *Primatte1* node.



- 2 In the *ColorReplace1* parameters, click to select the Source Color control, then drag over the blue spill areas on the front edge of the woman's shirt.
- 3 Click the Replace Color control again, then drag over a medium gray area of the background.

- 4 Turn on the `affectAlpha` button, then view the alpha channel of the node to test the area you are modifying.

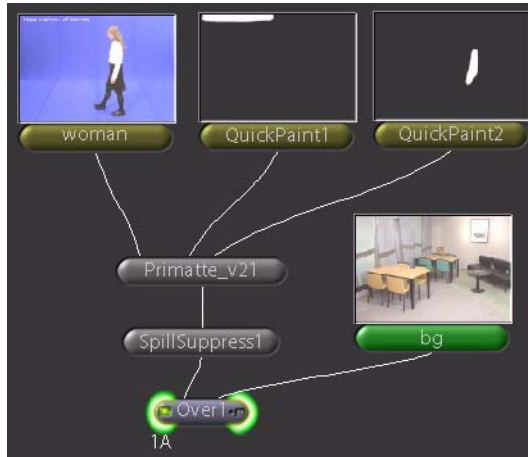


- 5 Set all the `hueRange`, `satRange`, and `valRange` parameters to 0.
- 6 Set the `hueFalloff`, `satFalloff` and `valFalloff` parameters to 0.2.
- 7 Load the *Primatte1* node into the Viewer and you'll see the results.

You may need to do some fine tuning operations in the *Primatte1* parameters after you've used *ColorReplace* for spill suppression.

Compositing Outside of Primatte

Primatte has a great deal of flexibility because you can composite within the node or pass its alpha channel to other layer nodes for compositing. You have the most flexibility when you do not composite inside of *Primatte*, because you can control how and where color-correction and effects are applied to your images. In the following process tree, the *Primatte* output is set to “alpha only,” and *preMultiply* is enabled in the *Over* node. In this tree, *Primatte* does not process the edge with color suppression operators and the edge stays much cleaner.



Note: Always supply the background image unless you are setting output to “alpha only.”

Although you can output *Primatte* with a premultiplied foreground and *without* a background image, you should supply the background input if possible, as *Primatte* still calculates some of this information. If you do not supply the background image, black ringing occurs around the edges. If this is impractical, toggle *replaceMode* to use *Color* and supply an appropriate *Replace Color* setting.

This has a corollary: Save yourself the trouble and set your output to alpha only. No background image is needed. Then do your spill suppression with a *HueCurves* or *SpillSuppress* node. This also ensures your image is unpremultiplied, making it ready to color correct.

This tutorial demonstrates the primary uses for Shake's tracking technology, including removing unwanted motion from an image sequence and “matchmoving” an element to the motion of another element in the composite.

Tracking allows you to do two things. First, you can eliminate unwanted movement in a plate, such as the movement caused by film gate weave. This is known as stabilizing. Second, you can match a stable element to a moving element, such as swapping the heads of two actors. This is called matchmoving. You often use the two together. For example, to do a head replacement, you typically first stabilize the head you want to grab, then matchmove it to the body you want to place it on. Both involve tracking patterns in an image sequence from frame to frame. This can be done manually, but that gets really tiresome, really fast—which is why Shake provides tracking nodes to do it for you.

Tutorial Summary

- Tracking and stabilizing nodes
- Stabilizing an image sequence
- Converting stabilization data to matchmove data
- Using the *MatchMove* node (with a four-point track)
- Positioning a foreground element
- Color-correcting a foreground element

Tracking and Stabilizing Nodes

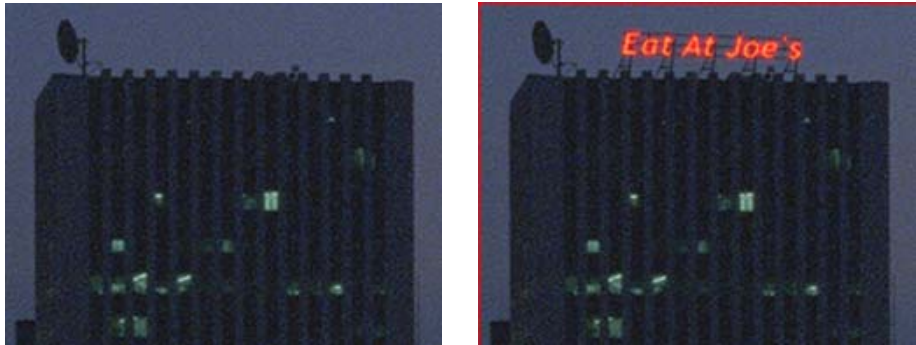
In Shake, three nodes track sequences of images:

- *Transform–Tracker*: This node generates unlimited tracking points, but does not apply movement to the element.

- **Transform–Stabilize:** This node generates one, two, or four tracking points, or uses points generated by other tracking nodes. It performs either stabilization or matchmoving.
- **Transform–MatchMove:** This node performs matchmoves and composites. It can also create its own one-, two-, or four-point tracks.

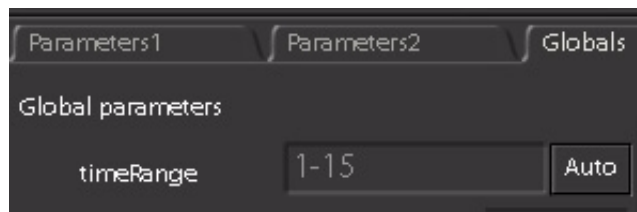
Stabilizing an Image Sequence

In the first section, stabilize a moving clip. Later, track the “Eat At Joe’s” sign onto the image.



To stabilize the building clip:

- 1 Add an **Image–FileIn** node and read in `$HOME/nreal/Tutorial_Media/Tutorial_07/images/stabilize/stab.1-15#.jpg` (a 15-frame clip).
- 2 In the **Globals** tab, click the **Auto** button in the **timeRange** parameter.
Your script length is automatically set to the number of frames in the clip, 15 in this case.

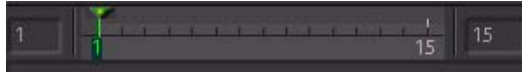


- 3 Click the **Home** button in the Time Bar playback controls.



Home

The Time Bar range is set from frame 1 to frame 15.



This does not affect your rendered frames, but helps you adjust your track points.

You have by now no doubt discovered that the plate drifts to the left. This is a prime candidate for stabilization, which locks the image down.

- 4 In the Node View, select the *stab* node, then add a Transform–*Stabilize* node.



A tracker appears in the Viewer.

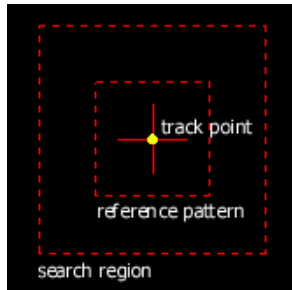


When the pointer is positioned over the tracker box, the tracker is highlighted. You can drag a highlighted tracker box in the Viewer.

- 5 Drag the tracker box to a high-contrast area— for example, the large window in the middle of the image.



The tracker is composed of the *track point* (the center), the *reference pattern* (the inner square), and the search region (the outer square). Here's how it works: Shake analyzes the reference pattern (the area of your image isolated within the inner square of the tracking box), then in subsequent frames looks for that pattern within the search region (the outer square of the tracking box). When a match is found, Shake records a track point (keyframe) at that coordinate.



- 6 Go to frame 1.
- 7 Click the Track Forward button in the Viewer shelf.



The 15 frames are tracked.

Note: You can also go to frame 15 and click the Track Backward button.



- 8 When the tracking is finished, activate applyTransform in the Parameters tab.

Note: When tracking large plates, turn on `limitProcessing` in the *Stabilize* parameters to decrease image load times.



- 9 Press Play to test the stabilized plate.



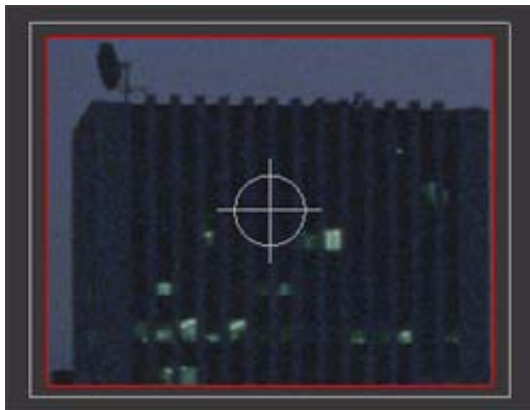
Note: You can also test the stabilized plate by launching a Flipbook.



The window stays in the same position for the duration of the 15-frame sequence.

When the plate drifts, the *Stabilize* node moves it back to the original point. This eventually causes a black border to appear around the image. The simple (lazy) way to fix this is with a *Scale* node to adjust the plate size. The *Scale* node concatenates with the *Stabilize* node. This is good.

- 10 In the Node View, select the *Stabilize1* node and connect a Transform–*Scale* node.
- 11 Using the onscreen transform controls or the controls in the *Scale* Parameters tab, scale the image up slightly. Examine frame 15 to gauge the correct scale.



As this is the “Lazy Man’s Method,” and therefore appealing to compositors across the world, it is not the ideal method. Instead, a clean plate should be constructed and placed behind the image. See “Creating Clean Plates” on page 243.

Converting Stabilization Data to MatchMove Data

Use the same building clip to build a simple matchmove. In this case, remove the stabilization data from the building clip, invert the stabilization data, then apply the data to a second image.

To convert stabilization data to *MatchMove* data:

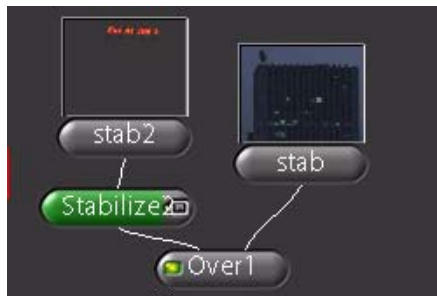
- 1 Add another *FileIn* node (do not delete the node tree built in the previous section) and read in *\$HOME/nreal/Tutorial_Media/Tutorial_07images/stabilize/eatatjoes.iff*.

Note that because the image is saved as a Shake IFF (.iff) file, the Domain of Definition (DOD) data is retained.



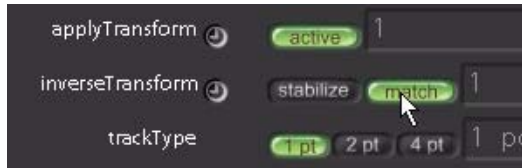
Since you have already tracked the background, you can attach the *Stabilize1* node to the *EatAtJoes* node, then use a *Layer-Over* node to composite the two images together.

- 2 Extract the *Stabilize* node (select the node in the Node View and press E) from the *stab* (building) node.
- 3 Connect the extracted *Stabilize* node to the *eatatjoes* node.
- 4 Add a *Layer-Over* node to the *Stabilize* node.
- 5 Connect the *stab* (building) node to the second input (background) of the *Over* node.



The *eatatjoes* image drifts away from the background image. The *Stabilize* node is trying to stabilize the image—the opposite of what you want. To match the *eatatjoes* image to the movement of the building clip, invert the transform by setting the shrewdly named *invertTransform* button to “match.”

- 6 In the *Stabilize* Parameters tab, set *inverseTransform* to “match.”



The *eatatjoes* sign moves with the background plate.



As shown, you can use the *Stabilize* node for stabilization and matchmoving. Others lurking around the office prefer to use the *MatchMove* node, described in the next section. There is one gotcha with using *Stabilize* for matchmoving: You must turn off the transform and connect the *Stabilize* node to the background (tracked clip) in order to re-track.

Note: When destabilizing, the *inverseTransform* toggle can also be used to reapply the same jitter that was removed. You may have a slight jitter due to gate weave. *Stabilize* the shot, apply all of your elements to the stabilized plate, then copy and paste the original *Stabilize* node to the end of the tree. Then, set *inverseTransform* to match, and you arrive with the same natural-feeling jittery movement.

Using the MatchMove Node

The next example uses more complicated tracking information to match a background clip that changes perspective. The *MatchMove* node is used to match the movement of the background. In this example, read in a clip of a moving bus and an advertisement image, then apply the image to the side of the moving bus. This tutorial also discusses some solutions for common tracking problems.

To start the *MatchMove* script:

- 1 Choose File > New Script, or press Command-N or Control-N to clear the existing nodes and begin a new script.
- 2 Add an Image-FileIn node and read in the *bus2.40-80#.jpg* clip and the *sign.jpg* image from the *\$HOME/nreal/Tutorial_Media/Tutorial_07/images/stabilize* directory.

The sign image is 720 x 210 pixels, and the bus clip is 720 x 486 pixels.



The following illustration shows the result of this lesson.



Before



After

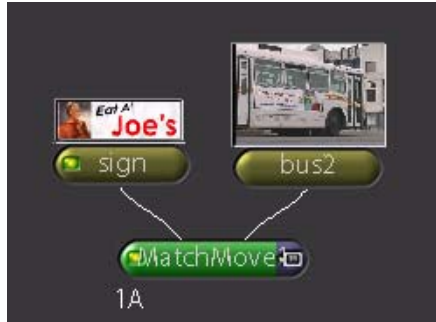
- 3 Examine the bus clip to identify good search patterns. (This should always be the first step.)

Luckily, the sign of the bus is visible during the entire clip, although there is some distortion due to perspective shifts.

- 4 In the Node View, select the *sign* node and add a Transform-*MatchMove* node.

Unlike other transform nodes, *MatchMove* allows you to composite within the node. The tracking is done on the second input (the background).

- 5 Connect the *bus2* node to the second input of the *MatchMove* node.



Although this project calls for a 4-point track (one for each corner), begin with only one tracker to keep things simple. Several intentional mistakes occur in this tutorial to help you better recognize problems with tracking your own footage.

- 6 Go to frame 1.
- 7 Position the tracker box in the lower-left corner of the sign on the side of the bus. Leave the tracker at the default size, and precisely position the track point on the corner. Use the + or – keys to zoom in, with the pointer as an aiming device.

Note: Press the Home button in the Viewer shelf to return the Viewer to the normal size.



- 8 In the Viewer shelf, click the Track Forward button.

Doh! The tracker advances a few frames and stops. The last number in the text next to the tracker box reads “c=0.2” (approximately). This indicates the correlation, or accuracy of the track. A score of 1 means a perfect correlation with the reference pattern between frame 2 and frame 1 (frame 1 as the reference pattern). A score of 0 is amazingly bad. On frame 2, you have a correlation of 0.2. This is bad.

The search region is too small to encompass the movement in the clip—the bus moves too far to the right. In this illustration, the desired area, artificially marked in green, extends beyond the zone of the tracker search region (the outer red box).



If the search region is too large, you waste time processing. If the search region is too small, the tracker does not find successful matches. There is also a black line around the sign along the bottom of the pattern. There is a chance due to the perspective shift in later frames that the tracker will confuse the two lines. Adjust the search area with these problem areas in mind.



- 9 Go back to frame 1.
- 10 Grab the corners of the search region and scale it outward, but shrink the bottom up a bit. Also, scale down the reference pattern to better fit the corner.



Note: When adjusting the tracker boxes, grab an edge to adjust just that edge. Grab a corner to scale the entire box. Grab the center to move everything. What is adjusted is highlighted in yellow.

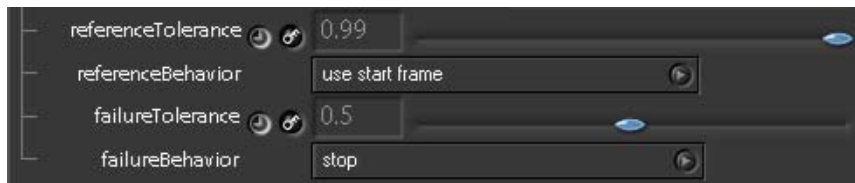
- 11 Click the Track Forward button.
Any previous keyframes are written over.

Gee, everything was going fine until around frame 34, right? At that frame (results vary), the tracker spitefully decided that the words on the sign were the best match to the original reference pattern at frame 1.



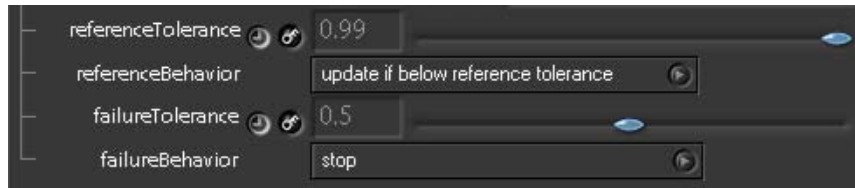
The tracker continually compares the new images back to frame 1 (or whichever frame you used to start the track). As the bus moves away, the sign gets smaller. At a certain point, the words match the original thickness of the sign border from frame 1 better than the sign border itself.

This matching behavior is controlled by the `referenceBehavior` setting in the `tolerances` subtree. The `referenceBehavior` pop-up menu is set to “use start frame” by default—it compares the new samples to the original start frame (not necessarily frame 1 if you start at a later frame).



Two settings in the `referenceBehavior` pop-up menu can help with scaling changes in the image sequence. The settings are “update every frame,” which uses the sample of the previous frame for the reference pattern, and “update if below reference tolerance,” which checks the correlation of every frame. If the correlation falls below the `referenceTolerance` value (the parameter directly above `referenceBehavior`), Shake uses the previous frame as the reference pattern. Shake continues to use that pattern until the correlation once again dips below the `referenceTolerance`. The second setting (“update if below reference tolerance”) is more accurate because you get inherent drift with “update every frame” as tiny errors accumulate.

- 12 In the *MatchMove* Parameters tab, set the referenceBehavior pop-up menu to “update if below reference tolerance.”



- 13 Find a frame with a high correlation. Although you can track from frame 33 (the frame before the tracker got confused), it is better to find a frame with a high correlation. The correlation is displayed in the yellow text next to the tracker box. Ideally, this is of course frame 1, but you may find a high correlation at approximately frame 14.

- 14 Click Track Forward.

With the above settings, you should get a successful track.

Note: You can also manually enter track positions with the use of the Autokey button in the Viewer shelf. Note that this is often the only way to get a track working.

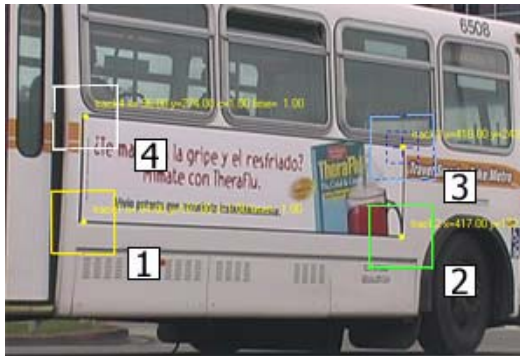
- 15 If the track fails again, repeat these steps, or manually reposition the tracker at the missed frame and restart the tracking.

To apply the other three trackers:

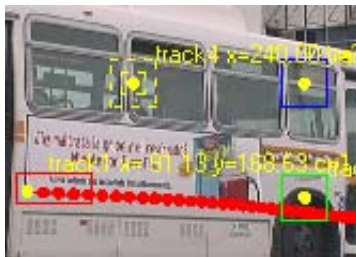
- 1 In the *MatchMove* parameters, set the trackType to “4 pt.”



The three extra trackers are activated. Each tracker is located in the relative corner where it should stay. For example, the upper-left corner of the foreground image is connected to track4 of the *MatchMove*. Shake uses Cartesian coordinates, so the trackers are ordered according to the following image.



Yes, you are right, this is different from some other prominent systems. Deal with it. The first track point, track1, is already set, and its keyframes are visible.



Every time you start a track, all visible trackers analyze the image and create new keyframes. Therefore, turn off the visibility of track1. The four trackers are listed in the lower portion of the *MatchMove* Parameters tab. You can click the track name text field (that is, where it says “track1”) to change the name. You can change the display color of a tracker by clicking on the color control. The “V” button toggles the visibility of a tracker.

- 2 Deactivate track1 by toggling its visibility.

This ensures the track is not overwritten.



An active tracker is green in the tracker list.



- 3 Go to frame 1.
- 4 Following the same principles of the first tracker, position and shape each of the remaining trackers on the remaining sign corners.



- 5 Click the Track Forward button to start the tracking analysis.

When finished, the three points should be reasonably accurate. If not, use the visibility toggles to isolate problem trackers and repeat the above steps to correct your track.



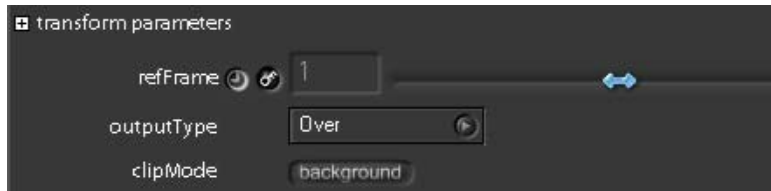
Note: Manual tracking? Don't ever ever ever believe a product demonstration—tracking, like keying, is rarely straightforward and usually involves a lot of manual labor. Make sure Autokey is on and away you go. You frequently must adjust points by hand. Welcome to the exciting world of production!

Position the Foreground Element

With the four tracks plugged into the *MatchMove* node, you can now apply the foreground element.

In the *MatchMove* parameters, the *outputType* pop-up menu contains several compositing options. Background is selected by default to allow you to immediately start tracking. To test the track, switch the *outputType* to a compositing operation.

- 1 In the *MatchMove* parameters, set the *outputType* to *Over*.



This produces a somewhat less-than-convincing composite.



As with *Stabilize*, you must activate the transformation.

- 2 Toggle applyTransform to active.



If an image appears brighter in an *Over* operation (in effect adding the two images), you should immediately suspect the absence of a mask in the foreground. Because the sign image is a JPEG and does not support alpha channels, you must add an alpha channel to the image.

- 3 In the *sign* node parameters (the *FileIn* node), enable autoAlpha.

The *autoAlpha* parameter examines the input image for an alpha channel. If no alpha channel is found, it sets the entire alpha channel to 1. If an alpha channel is present, it is left untouched by *autoAlpha*.



This is an improvement. The four corners of the foreground image are snapped to the four corners of the trackers. If the image is twisted around, you have done something such as put track 3 in track 4's corner. You can unravel that in the next step.

Adjust the Sign Position

The sign may not fit exactly into the background's sign limits. A little bit peeks out from behind.



To adjust the foreground sign position:

- 1 Go to frame 1.
- 2 Ensure that the Viewer Autokey button is deactivated (unless you want to animate the image, which you do not).
- 3 Click the BG/FG button on the Viewer to show the foreground display.



This displays four corner markers in the Viewer. These points correspond to what is matched to the four trackers. By default they are located in the four corners of the image.



- 4 Pull the corners in.

The corners represent the track location points. Therefore, bringing the points in expands the image size in the composition.



- 5 Toggle the Viewer to BG mode.

Note: You may need to switch your outputType back to *Over*. Occasionally this parameter is not reset properly.

The image is now slightly larger in the composition.



Color Correct the Foreground Element

Anybody who has visited Los Angeles knows that a clean bus is a mythical beast. As the final step, color correct the sign to better match it to the bus.



Before



After

To build the color correction:

- 1 In the Node View, select the *sign* node.
- 2 Using the following illustration as a guide, insert these nodes from the Color tab: *Compress*, *Mult*, *Saturation*, and *SetBGColor*.
- 3 Add an *Other-AddShadow* node.

Note: You do not need a *MDiv/MMult* set because you are working on the full frame.



For the color correction, first adjust the whites and blacks with the *Compress* node. Because the sign has pure whites and blacks, and the bus has good representations of what should be white and black, *Compress* works well.

- 4 In the *Compress1* parameters, click the Low Color control.
- 5 Scrub color in the Viewer that represents the blackest part of the bus, for example, under the wheel well of the bus.
The blacks are raised to around 18 percent.
- 6 Click the High Color Picker box, and select a bright color from the side of the bus.
As the bus changes its lighting levels as it turns, animate the High Color parameter.

To animate the *Compress* parameters:

- 1 Go to frame 1.
- 2 In the *Compress1* parameters, click the Autokey button in the High Color parameters.
- 3 Go to frame 39.
- 4 Scrub the same white area. Since Autokey is already activated, a keyframe is created.



The sign is a neutral grey color. Some warm color makes it stick out a bit and draws the viewer's attention (though not too much—it still has to be integrated into the scene). You can tint it yellow with *Compress1*, but you then must have the same modification for all of your keyframes, which is cumbersome. A better idea is to use the *Mult* node to tint the image. As *Compress* and *Mult* concatenate, there is no loss of color quality. This is good. Very good.

5 Load the *Mult1* parameters.

6 Using the Color control, tint the sign yellow.

The sign is now yellowish, but the red of “Joe’s” is too bright. Use the *Saturation* node to lower the overall saturation of the sign. You may need to boost the *Mult1* node to compensate.

7 In the *Saturation1* parameters, lower the saturation value.

Note: *Saturation* does not concatenate with other nodes.

The three nodes (*Compress1*, *Mult1*, and *Saturation1*) demonstrate an advantage of working with nodes and concatenation. Because *Compress1* is animated, it is awkward to apply adjustments to the overall color for the entire clip length within the *Compress* node. By breaking these steps into separate nodes, you do not have to adjust each keyframe.

Next, because the *Compress* node raised the black levels in infinite space, the Color–*SetBGColor* is used to set the area outside of the image area to a specific color (black by default). If the sign has a soft mask, you must use a *MDiv/MMult* pair.

Test the effect of the *SetBGColor1* node:

- In the Node View, select the *SetBGColor1* node and press I to ignore it.
- Press I again to activate the *SetBGColor1* node.

The sign is color balanced in the scene, but the shadow, created with the *AddShadow* node, needs to be adjusted. Without this node, the sign has no feeling of depth.

To adjust the sign shadow:

- 1 In the *AddShadow1* parameters, set the xOffset and yOffset to approximately –5.
- 2 Set fuzziness to approximately 35.

- 3 Set shadowOpacity to approximately .7.



No shadow



With AddShadow

Note: For additional fine tuning, look at the script `bus_track2.shk` script in the `$HOME/nreal/Tutorial_Media/Tutorial_07/scripts` directory. This script adds some color correction and masking to take care of the reflections that go across the side of the bus from frames 36 to 40.

Stabilize as an Alternative to MatchMove

You may have noticed that tuning the corners of the sign using the FG/BG Viewer controls was less than ideal. Some might use the word “maddening and suicide-inducing,” as there is no interactive feedback. An added drawback: Onscreen controls from upstream are not properly adjusted by the *MatchMove* node. This is one advantage that the *Stabilize* node has over the *MatchMove* node. As demonstrated in the Eat At Joe’s tutorial, you can use *Stabilize* instead of *MatchMove* to create matchmoves. This gives you a more intuitive control mechanism, but requires more plumbing in the Node View. Note that for this exercise, you are copying the tracks from *MatchMove1*, but you can generate the tracks with a *Stabilize* node.

To use *Stabilize* for creating a matchmove:

- 1 Attach *Stabilize* to the node you want to track (in this example, the bus image), and generate your tracks.
- 2 Extract the node (select the *Stabilize* node and press E), and attach it to the image you want to transform (in this example, the sign).
- 3 In the *Stabilize* parameters, activate the transformation with `applyTransform` and set `inverseTransform` to *match*.
- 4 Composite the *Stabilize* over the background with an *Over* node.
- 5 Optional Step (not necessary for this tutorial): Insert a *Transform-Viewport* node above the *Stabilize* node to adjust the frame around what you want to track. Or use a *CornerPin* node, and invert its transform. Either of these techniques readjusts the corner positions to match the image corners in the next step. This is useful only when you are adjusting an image area that does not match the frame borders.

- 6 Insert a *CornerPin* node above the *Stabilize* node. *CornerPin* is used to position the foreground over the background.

This technique has several advantages:

- More flexibility in your compositing
- Better control over transform concatenation of the foreground
- Better control over premultiplication of the foreground
- Accurate pass-through of upstream onscreen controls
- Intuitive control of foreground positioning
- Access to the rotation and scaling values for two-point transforms

To switch the tracking from the *MatchMove* node to a *Stabilize* node:

- 1 In the Node View, select *AddShadow1*.

- 2 Shift-click Transform–*CornerPin*.

The Shift key modifier installs *CornerPin* off as a new branch in the node tree.

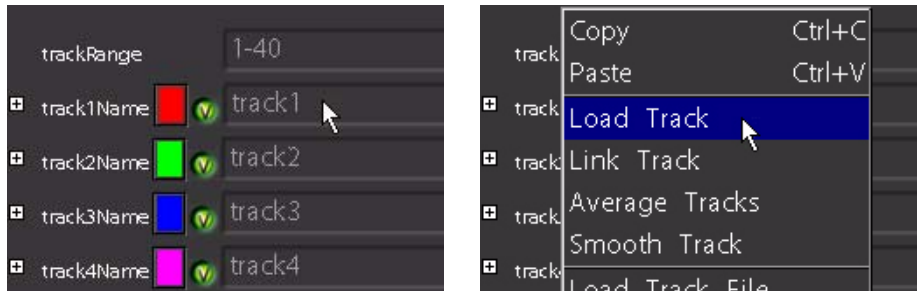
- 3 Connect a Transform–*Stabilize* node to the *CornerPin* node. Normally, you attach the *Stabilize* node to the *bus2* node to start tracking, but you already have your tracks in *MatchMove1*.
- 4 Connect an *Over* node to *Stabilize1*.
- 5 Connect *bus2* to the second input of *Over1*.



Next, copy the tracks from *MatchMove1* to *Stabilize1*.

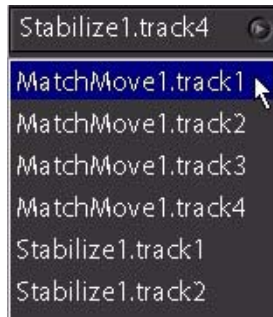
To copy the tracks into *Stabilize*:

- 1 Right-click the trackNames, then choose *Load Track* from the shortcut menu:

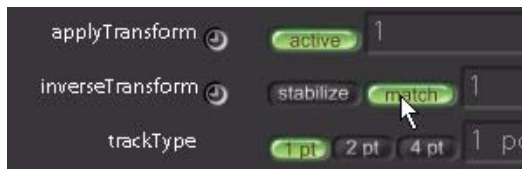


The Select Track window opens.

- 2 Choose *MatchMove1.track1* from the pop-up menu, then click OK.
Track1 is copied from *MatchMove1* to *Stabilize1*.



- 3 Repeat the process for track2 through track4.
- 4 Activate the transformation with `applyTransform`.
- 5 Set the `inverseTransform` parameter of the *Stabilize* node to *match*.



You are currently applying the motion to only one point, so toggle your trackType to "4 pt."



Things don't quite line up in the Viewer. Oops.



Frame 1



Frame 30

MatchMove has controls for adjusting the foreground position. *Stabilize*, by default, assumes that frame 1 has the relative position that you want. The selected frame can be modified with the *referenceFrame* parameter, but it still doesn't modify the relative position. You must therefore rely on external nodes to adjust the foreground—in this case, the *CornerPin* node.

- 6 Make sure the Viewer Autokey button is deactivated.
- 7 Load the *CornerPin1* parameters.
- 8 Move the corners of the *CornerPin* to match the side of the bus.



The controls are now completely intuitive, as well as interactive.

This tutorial demonstrates how to create reusable groups of commands, called macros. In this example, you'll set up a basic macro for a motion blur effect that is adjustable to any angle.



Tutorial Summary

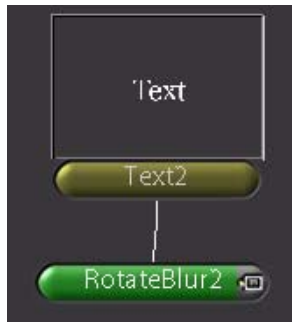
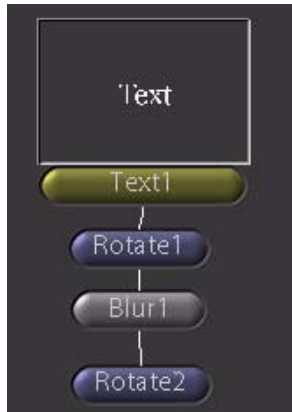
- What is a macro?
- Creating a handmade macro
- Saving and testing the macro
- Adding a button to the interface
- How to set slider ranges
- Creating macros with MacroMaker
- Creating sliders in MacroMaker

What Is a Macro?

So, what is a macro? Macros are commonly used nodes that are combined to create new nodes. You control which parameters are exposed, and hide the ones you do not need. Macros are extremely powerful ways of modifying Shake to suit your own needs.

Note: Use the onscreen version of this tutorial to allow easy copy and paste of the scripting sections.

The following images demonstrate the creation of a macro. The first image shows four connected nodes—the *Text* node and the three nodes that combine to create a blur effect. Following a long, arduous process, the second image shows the completed macro attached to an icon (*RotateBlur* icon) in the Filter Tool tab. The third image shows the completed macro, the *RotateBlur* node, applied in the tree—it looks like any other node. The last image demonstrates that only the angle and the blur parameters from the original nodes are exposed in the Parameters tab of the new node.



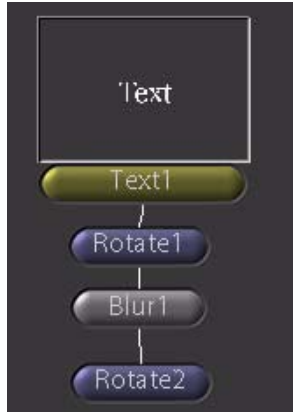
Creating a Handmade Macro

The first step in a macro is to build the desired effect—for the purposes of this example, a “*RotateBlur*” node—using normal nodes.

To build the node tree:

- 1 Create an Image–*Text* node.
- 2 Attach a Transform–*Rotate* node.
- 3 Attach a Filter–*Blur* node.

- 4 Attach a second *Rotate* node.



In this example, the algorithm works by rotating the image, blurring the image on the X axis, then rotating the image back to the start position. Therefore, if *Rotate1* has an angle of 45 degrees, *Rotate2* must have an angle of -45 degrees. The best way to make this happen is by entering an expression to link *Rotate1* to *Rotate2*.

To link the *Rotate1* and *Rotate2* angle parameters:

- 1 Click the right side of the *Rotate1* node to load its parameters into the Parameters tab.
- 2 In the angle value field, type the following expression:

```
-Rotate2.angle
```

This expression tells Shake to make the angle of the *Rotate1* node the inverse of the angle of the *Rotate2* node. As you modify *Rotate2*, *Rotate1* automatically updates.

Note: Case sensitivity is important. In Shake, each word in a node name has initial capitalization. For example, *QuickPaint* has an uppercase “Q” and “P.” Parameter names follow the same basic rule, with one exception: The first letter in the first word of a Shake parameter is lowercase. So, for example, in the parameter framesPerSecond, only the “P” and the “S” are capitalized—the “f” is lowercase. As you develop macros, it is good form to follow these guidelines, so that you never have to wonder which letters are capitalized.

To set the *Blur* parameters:

- 1 Load the *Blur1* node parameters into the Parameters tab.
- 2 Open the pixels subtree and set xPixels to 200.
- 3 Set yPixels to 0.
- 4 Toggle the spread parameters button to Outside Frame (1).

To test the effect:

- 1 Load the *Rotate2* parameters into the Parameters tab.
- 2 Adjust the angle parameter.

The blurring rotates, but the text stays in the same spot. If the *Rotate2* angle parameter is 67 degrees, then the *Rotate1* angle is -67.

Warning: Do not modify *Rotate1*, because that will break the link to *Rotate2*.

If the effect is not working for some reason, don't sweat it—delete your *Rotate* and *Blur* nodes, select and copy the following text (press Command-C or Control-C), then paste it into the Node View (press Command-V or Control-V):

```
Rotatel = Rotate(0, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);
Blur1 = Blur(Rotatel, 200, 0, 1, "gauss", xFilter, "rgba");
Rotate2 = Rotate(Blur1, -36.9218, 1, width/2, height/2, 0, 0.5, 0);
```

Because Shake is a compiler, you can copy nodes from text and paste the text into the interface.

Save the script for later use:

- Choose File > Save Script As and save the script as *rotateblur_tree.shk*.

You are now ready to start creating the macro. By default, Shake places macros, preference settings, and interface modifications in your *\$HOME* directory (for example, *Users/john*). In the following steps, you will add a series of subdirectories to the *\$HOME* directory using the Terminal application.

To make the macro:

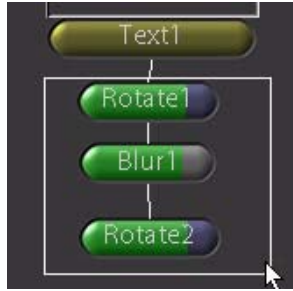
- 1 Open Terminal, found in the *Applications/Utilities* directory.
- 2 Type:

```
mkdir -p $HOME/nreal/include/startup/ui
mkdir -p $HOME/nreal/icons
```

Note: You can add files to other directories as well using an environment variable. For more information on setting environment variables, see in Chapter 14 of the *Shake 4 User Manual*.

- 3 Launch a text editor (TextEdit, vi, emacs, or the like).

- 4 In Shake, drag to select the *Rotate1*, *Blur1*, and *Rotate2* nodes in the process tree.



- 5 Press Command-C or Control-C to copy the nodes.

Note: You can also select Copy with the right-click shortcut menu in the Node View.

- 6 In the text editor, paste the nodes (press Command-V or Control-V).

The Shake source code is pasted into the editor, and it looks something like the following:

```
Rotate1 = Rotate(0, -Rotate2.angle, 1, width/2, height/2, 0, .5, 0);
Blur1 = Blur(Rotate1, 200, 0, 1, "gauss", xFilter, "rgba");
Rotate2 = Rotate(Blur1, 42.61728, 1, width/2, height/2, 0, 0.5, 0);

// User Interface settings

SetKey(
    "nodeView.Blur1.x", "290",
    "nodeView.Blur1.y", "227",
    "nodeView.Rotate1.x", "290",
    "nodeView.Rotate1.y", "263",
    "nodeView.Rotate2.x", "290",
    "nodeView.Rotate2.y", "191"
);
```

Copying Text Into the Node View

If you copy the first two lines in the text editor:

```
Rotate1 = Rotate(0, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);
Blur1 = Blur(Rotate1, 200, 0, 1, "gauss", xFilter, "rgba");
```

you can paste it back into the Node View.

This demonstrates that you can copy between script and interface.

If you already know C programming, you may want to jump down to the finished macro; otherwise, keep reading.

The text in your editor is raw script, meaning it is not formatted as a macro. If you paste it back in, you get three more nodes. You need to format the text as a single function—a macro.

- 7 Delete the layout information—that is, everything in the text editor window including and below the “User interface settings” line.

These lines tell Shake where to place the nodes if you paste them back in again. You do not need the lines, so remove them.

You are left with this:

```
Rotate1 = Rotate(0, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);
Blur1 = Blur(Rotate1, 200, 0, 1, "gauss", xFilter, "rgba");
Rotate2 = Rotate(Blur1, -36.9218, 1, width/2, height/2, 0, 0.5, 0);
```

The left side of the argument supplies a variable name. In the above example, there are three variables: *Rotate1*, *Blur1*, and *Rotate2*. These are assigned to Shake functions, the *Rotate* and the *Blur* functions. (Inside the Shake interface, these are referred to as “nodes.” Inside a script, they are referred to as “functions,” as you can have other types of functions.) Next are the values for the functions. The first argument is generally the incoming image, so you can see that the incoming image for *Blur1* is the result of the first *Rotate* function, *Rotate1*. *Rotate1* has a value of 0 as the incoming image—there is no expected input. If you look these functions up in the documentation, you find the scripting format for each function.

If you change the variable names *Rotate1*, *Blur1*, and *Rotate2*, you must modify every reference to them.

The following is used as an example only (it is not a step in the tutorial):

```
Marshall = Rotate(0, -Holly.angle, 1, width/2, height/2, 0, .5, 0);
Will = Blur(Marshall, 200, 0, 1, "gauss", xFilter, "rgba");
Holly = Rotate(Will, 42.61728, 1, width/2, height/2, 0, 0.5, 0);
```

Notice how the strings “gauss” and “rgba” are in quotation marks because they are strings, meaning letter data rather than numeric data. Compare these to what you see in the interface parameters tab to get an idea of how Shake saves its parameters in a script.

Starting the Macro Format

Next, add some formatting. You must declare the type of the macro—if the macro spits out an image, an integer (int), a number with decimal place (float), or a word (string). Because this macro modifies an image, it needs to return an image. After that, add the macro name. In this case, name the macro *RotateBlur*, capitalizing each word for consistency.

Also, include two sets of parentheses. Incoming variables are placed in the first set, and the second set {the braces} surrounds the body of the macro.

- 1 Add the formatting for the macro body and declare the macro type and name:

```
image RotateBlur( )
{
  Rotate1 = Rotate(0, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);
  Blur1 = Blur(Rotate1, 200, 0, 1, "gauss", xFilter, "rgba");
  Rotate2 = Rotate(Blur1, 42.61728, 1, width/2, height/2, 0, 0.5, 0);
}
```

Next, you need to declare the information that is passed into the macro. You can potentially change all parameters—the rotate angle, center of rotation, motion blur parameters, blur filter, and so on. Practically speaking, only the rotate angle and the blur amount need to be set, giving you two sliders. You must also declare that an image is passed into the macro to be modified. If you do not have an image, the macro assumes that you are somehow creating a new image, such as a *Ramp* or *ColorWheel* node. Each time you add an image input variable, an input (connection point) is created on top of the node in the Node View (so you can plug in an image). Each time you add a float, string, or int, a new slider or text field is created in the Parameters tab. When you add these variables, declare the type of variable (again—either image, int, float, or string), then the name. The names of the parameters should follow that pesky rule mentioned earlier—the first letter in the parameter name is lowercase.

- 2 Add the input variables, which later become sliders or image inputs on the node:

```
image RotateBlur(
  image input,
  float angle,
  float blur
)
{
  Rotate1 = Rotate(0, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);
  Blur1 = Blur(Rotate1, 200, 0, 1, "gauss", xFilter, "rgba");
  Rotate2 = Rotate(Blur1, 42.61728, 1, width/2, height/2, 0, 0.5, 0);
}
```

- 3 Plug the variables into the right spot inside the macro body. Instead of the number or word copied from Shake, use the variable name. These values are later modified with sliders.

Substitute the variables in the macro body:

```
image RotateBlur(
  image input,
  float angle,
  float blur
)
{
  Rotate1 = Rotate(input, -Rotate2.angle, 1, width/2, height/2, 0, .5, 0);
  Blur1 = Blur(Rotate1, blur, 0, 1, "gauss", xFilter, "rgba");
  Rotate2 = Rotate(Blur1, angle, 1, width/2, height/2, 0, 0.5, 0);
}
```

- 4 Indicate what you want to spit out of the macro with the return line. This is the final image (or float, int, string) that you want to extract out of the macro function. Remember the semicolon at the end of the line—in this case, *return Rotate2*, as it has the result you need.

Add the return line:

```
image RotateBlur(  
    image input,  
    float angle,  
    float blur  
)  
{  
    Rotate1 = Rotate(input, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);  
    Blur1 = Blur(Rotate1, blur, 0, 1, "gauss", xFilter, "rgba");  
    Rotate2 = Rotate(Bur1, angle, 1, width/2, height/2, 0, 0.5, 0);  
    return Rotate2;  
}
```

- 5 The following is the final macro. As a final touch, you should add default values for any parameter, so it launches without having to add values in the command line.

Add the default values:

Note: Take special care with the commas.

```
image RotateBlur(  
    image input,  
    float angle = 45,  
    float blur = 150  
)  
{  
    Rotate1 = Rotate(input, -Rotate2.angle, 1, width/2, height/2, 0, .5, 0);  
    Blur1 = Blur(Rotate1, blur, 0, 1, "gauss", xFilter, "rgba");  
    Rotate2 = Rotate(Bur1, angle, 1, width/2, height/2, 0, 0.5, 0);  
    return Rotate2;  
}
```

Saving and Testing the Macro

When this is done, save the file in your *startup* directory. Make sure the file name has a .h file extension. Some text editors automatically append an extension. This is bad. There is no correlation between the file name and the macro name.

To save the macro:

- Save the file in your *\$HOME/nreal/include/startup* directory:
/Users/Me/nreal/include/startup/my_macro.h

To test the macro:

- Go to the command line in Terminal.

You might be saying to yourself right now, “Hey, why am I testing this in a Terminal?” All you have done so far is create the macro—you have not plugged it into the Shake interface. You have not written anything that tells Shake to build this function into the interface. You do that in a moment.

- Test the function by typing in Terminal:

```
shake -help rotate
```

If everything worked properly, you should get a listing of all functions that contain the word “rotate,” including the new function, *RotateBlur*.

Note: The command line is the only place where capitalization does not matter, because it saves you the effort of having to press Shift all the time.

```
-rotate [angle] [aspectRatio] [xCenter] [yCenter] [motionBlur] [shutterTiming] [shutterOffset]  
-rotateblur [angle] [blur]
```

If you do not get this help message, Terminal returns an error message that tells you what went wrong.

Note: When working in Terminal, error messages are printed in that window. When working in the interface, error messages are printed in the terminal that launched Shake. If you launched Shake with the Dock, the messages appear in the Console. The errors generally detail where you have a mistake or if you have a variable that is incorrectly named.

If you cannot tell what is wrong from the error message, check the following list of possible problems:

- Match the capitalization and spelling of the variable names to where they are plugged in. For example, did you declare the variable *blur* but use *bluramount* by mistake later on?
- Use normal parentheses for the variable declarations.
- Use braces—{}—for the body of the macro.
- Include a semicolon at the end of the return statement.
- Do not include an extra comma at the end of your variables on the last line of the variables. If you did, remove it.
- Make sure the .h file was saved in your *startup* directory.
- Make sure the saved file has a .h extension on the name, and that it is not named *nreal.h* or *nrui.h*.

If you are still unable to figure it out, copy the finished macro from above into your *startup* directory and save it as *my_macro.h*.

Now try your new macro on an image. If you do not have an image, use the sample image in the `$HOME/nreal/Tutorial_Media/Tutorial_08` directory.

To test your new macro on an image in the Tutorial_08 directory:

- 1 In the Terminal, type the following:

```
cd nreal/tutorial_media/tutorial_08/images
```

The “cd” command at the beginning of the line tells Terminal to *change directories* to the specified path—in this case, the `Tutorial_08/images` directory.

- 2 Press Return.

Terminal changes directories according to the specified path.

- 3 At the command-line prompt, type any of the following commands (or cut and paste them, if you are reading the onscreen version of this tutorial):

- `shake traffic_bg.jpg -rotateblur`
- `shake traffic_bg.jpg -rotateblur 100 300`
- `shake traffic_bg.jpg -rotateblur "Linear(1,0@1,360@11)" 200 -t 1-10`
- `shake traffic_bg.jpg -rotateblur "Linear(1,0@1,360@11)" 200 -gui`

Terminal Shortcuts for Repeating Commands

If you press the Up Arrow key, the last command typed in Terminal is repeated. Press the Up Arrow again to list the previous command on the list, and so on. Use the Left Arrow and Right Arrow keys to edit the command.

The “Linear...” used for the angle of the blur is an animation curve, using a Linear format. The first “1” indicates the curve is extended into infinity. The second set of numbers indicates you have a value of 0 at frame 1, and then a value of 360 at frame 11.

The last command launches the tree into the interface. You can see your new *RotateBlur* node in action. However, you still cannot actually click anything to create a second node. For that, you need to create a button and attach an icon.

Adding a Button to the Interface

This section shows you how to create a button in the interface and attach a node to the button. The icons for the tabs have two qualities:

- The icon size is 75 x 40 pixels.
- The icons are saved in the `nreal/icons` directory as `TabName.Name.nri`.

To make an icon with which to apply the *RotateBlur* function:

- 1 In Terminal, type the following:

```
shake -addtext RotateBlur -rotateblur -gui
```

The text is launched in the interface with your *RotateBlur* node attached.

- 2 Attach a Transform–Fit node to the *RotateBlur1* node in the Node View.

- 3 Set the xSize parameter to 75.
- 4 Set the ySize parameter to 40.

The result is an image that is 75 x 40 pixels.

To save the icon file:

- 1 Attach an Image-FileOut node to the *Fit* node.
- 2 In the File Browser, use the Directories pop-up menu to navigate to your *\$HOME/nreal* directory.
- 3 Open the *icons* directory.

You should have created this directory in the first step of the tutorial. If not, use the Create Directory button to make this directory.



- 4 Type "Filter.RotateBlur.nri" in the File Name field, then click OK.

The new icon is assigned to the Filter tab. (The .nri extension stands for "Nothing Real Icon.")

- 5 Choose Render > Render FileOut Nodes.
- 6 In the Render Parameters window, set renderFileOuts to All.
- 7 Click Render.

A 320 x 243 thumbnail appears. (Yes, correct, it is rather bizarre to have a thumbnail that is larger than the actual image being rendered.)

You now have the two necessary elements: the macro and the icon. You now need to write the code to place both in the interface.

- 8 Quit Shake. (You must restart Shake to write UI code.)
- 9 Launch a text editor.
- 10 Paste (or type if you are reading the printed version of this tutorial) the following into the text editor:

```
nuiPushToolBox("Filter");  
nuiToolBoxItem("RotateBlur", RotateBlur(0,0,0));  
nuiPopToolBox();  
//These set the slider ranges for the blur parameter  
//nuiDefSlider("RotateBlur.blur", 0, 400 );
```

The first line indicates the Tool tab where the macro is placed—the Filter tab in this case. If the tab does not exist yet (for example, the MySwankFunctions tab), Shake creates it for you. The second line adds the function into the tab. The first occurrence of "RotateBlur" calls the icon file that you just created.

Important: Shake assumes you have matched the tab name with the file name, so if you actually did place this into the *MySwankFunctions* tab, you have to rename the icon file on your disk to *MySwankFunctions.RotateBlur.nri*.

Creating a Node Without an Icon

If you do not have an icon, create a plain button with text on the button by placing an @ sign before the word:

```
nuiToolBoxItem( "@OopsNoIcon", RotateBlur(0,0,0) );
```



As this example shows, the button name can differ from its actual function.

The next part of the code, *RotateBlur(0,0,0)*, is the function call, followed by its default values. Because Shake does not know what image it is attached to (until you actually press it), place a 0 as the first (image) variable. The second and third zeroes set the angle and blur parameters to 0. The third line closes up the tab.

- 11 Save the text file as *rotateblur_ui.h* in your *\$HOME/nreal/include/startup/ui* directory.
- 12 This directory contains all instructions to modify the interface.
- 13 Launch Shake.

In the Filter tab, the new *RotateBlur* button appears.



If there is a problem, check the following list:

- If you only see the "Missing Artwork" icon, it means Shake cannot find the icon for some reason. Make sure the icon is named *<TabName>.<IconName>.nri* and is saved in the *\$HOME/nreal/icons* directory. Make sure that the spelling and capitalization of the file name and the call to the file match.

- If there is nothing at all, you have made an error in your ui file. Check the error messages for an indication.
- If the icon is there, but nothing happens when you click the button, check the spelling, capitalization, and number of arguments of the call to the *RotateBlur* function. Make sure there are no extra commas.
- Make sure you saved your ui file in the *ui* directory, and that it ends with a .h extension.

How to Set Slider Ranges

This section discusses how to test the sliders created in your macro.

To test the sliders:

- 1 Add the new *RotateBlur* node.
- 2 Move the angle and blur sliders.



Notice that the angle slider values range from -360 to 360, but that the blur slider ranges from 0 to 1, forcing you to use the virtual sliders (Control-drag in the value field) to go beyond 1. Set your own slider range with a bit of code already included in the example from above. On the last line of your ui file—and only the last line—delete the double forward slashes://. When “commented out” by double slashes, a line of code is not read by Shake upon startup of the application. By removing the comment characters, you reactivate the line.

To uncomment the slider definition:

- 1 Remove the // from the last line in your *ui/rotateblur_ui.h* file.

The code should now look like this:

```
nuiPushToolBox("Filter");
nuiToolBoxItem("RotateBlur", RotateBlur(0,0,0));
nuiPopToolBox();
//These set the slider ranges for the blur parameter
nuiDefSlider("RotateBlur.blur", 0, 400 );
```

This sets the slider range from 0 to 400 for the blur parameter of the *RotateBlur* function. Spelling and capitalization are important, which is why that tiresome rule explained earlier starts to make sense.

You do not have to set a range for angle, because it is already set between -360 and 360 in the default Shake setup files. To change the angle slider range, duplicate the blur line and modify it to say *angle*. This example sets it from 0 to 180:

```
//These set the slider ranges for the blur parameter
nuiDefSlider("RotateBlur.blur", 0, 400 );
nuiDefSlider("RotateBlur.angle", 0, 180 );
```

- 2 Save your file, then launch Shake again.

The sliders are now set to 0 to 400 and 0 to 180.

For more information on customizing parameters, see Chapter 14, “Customizing Shake,” in the *Shake 4 User Manual*.

Creating Macros With MacroMaker

This next section of the tutorial duplicates your labor by using the MacroMaker. It is a fast way of creating reusable macros, but it is important to understand the files it creates so that you can modify them. Creating the macro by hand (above) helps demystify the process that the MacroMaker follows.

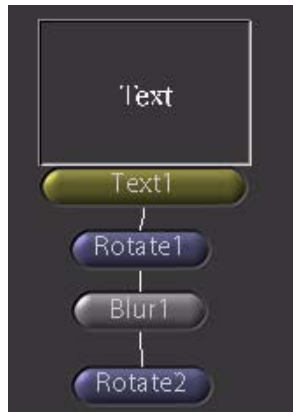
To create a macro with the MacroMaker:

- 1 Open Shake, and, if you followed the above parts of the tutorial, load the script you created called *rotateblur_tree.shk*.

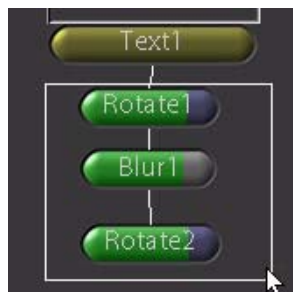
Note: If you do not have this script, copy and paste the following code into the Node View:

```
Text1 = Text(300, 200, 1, "Text", "Courier", 100,xFontScale, 1, width/2,
    height/2, 0, 2, 2, 1, 1, 1, 1, 0, 0, 0, 45);
Rotate1 = Rotate(Text1, -Rotate2.angle, 1, width/2, height/2, 0, 0.5, 0);
Blur1 = Blur(Rotate1, 181.9876, 0, 1, "gauss", xFilter, "rgba");
Rotate2 = Rotate(Blur1, 42.64842, 1, width/2, height/2, 0, 0.5, 0);
```

The node tree appears in the Node View.



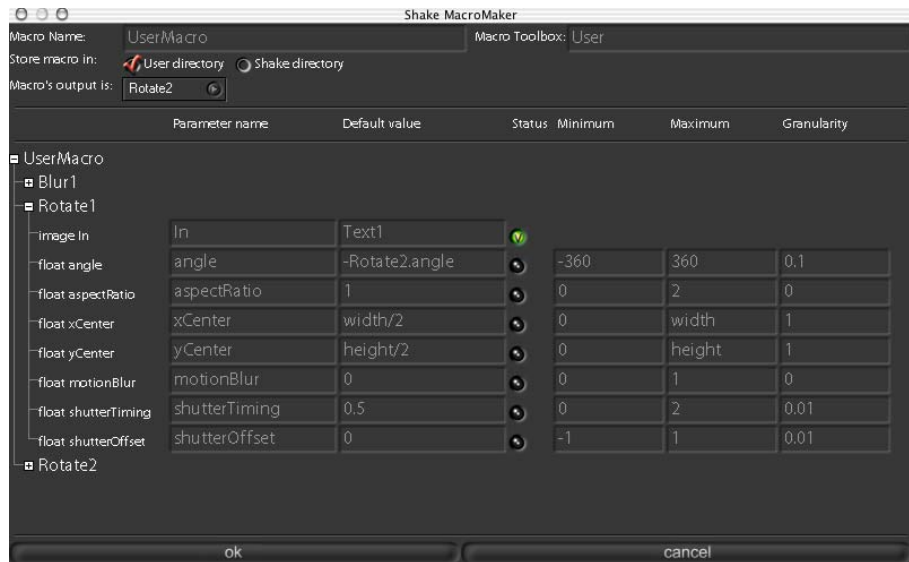
- 2 In the Node View, select the *Rotate1*, *Blur1*, and *Rotate2* nodes that build the macro.



- 3 With the pointer in the Node View, press Shift-M.

Note: You can also right-click in the Node View, then choose Macro > Make Macro from the shortcut menu.

The Shake MacroMaker is launched.



The top portion of the Shake MacroMaker window contains text fields for the name of the macro, and specifies in which tab the macro is placed. By default, MacroMaker assigns a name, "UserMacro," and assigns the macro to a new Tool tab called "User." You can change the name of the macro and its tab.

- 4 Name the macro "RotateBlur." Shake notifies you this name is already used because you already (manually) created a macro called *RotateBlur*, and therefore appends an A to the end of the name.



- 5 In the Macro Toolbox text field, type "Filter."
 - 6 Leave "Store macro in" set to the default, "User directory."
- The "Store macro in" setting indicates where the macro is saved:

- The "User directory" is `$HOME/nreal/include/startup`. Macros stored here are only available to you on the local machine.

- The “Shake directory” is the `<ShakeDirectory>/include/startup` directory. When this option is checked, the macro is available to all users who open Shake using that specific binary in that specific directory. You may have permissions problems writing to this directory.

The “Macro’s output is” pop-up menu tells you which node is the final output node of the new macro. Shake usually makes a good guess, but you may need to explicitly specify the node if you have multiple branches selected. In this example, *Rotate2* is the proper output. This is the equivalent to the line, *return Rotate2*; that you used in the first macro.

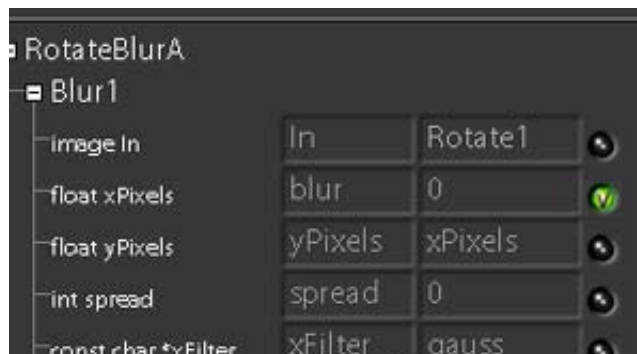
Creating Sliders in MacroMaker

The next area of the MacroMaker window contains the list of parameters you can make available to the user, including the name of the parameter, the default values, and the slider range.

To expose the *Blur1* xPixels parameter and the *Rotate2* angle parameter:

- 1 In the MacroMaker window, open the *Blur1* subtree.
- 2 Change the name of the float xPixels variable to “blur.” (This does not conflict with the *Blur* function because of capitalization.)
- 3 Set the default float xPixels value to 0.
- 4 Make sure the float xPixels Visibility button is on.

Each visible parameter will appear in the interface as an input or a slider.



- 5 In the *Rotate2* (not *Rotate1*) subtree, change the name of the float angle variable to “angle.”
- 6 Set the default float angle value to 0.

- 7 Make sure the float angle Visibility button is on.



The *Rotate1* “image In” parameter is activated automatically. This is because an image is fed into it, but is not accounted for by the nodes that you selected. This can cause problems (though not in this case), as all unattached image inputs are activated.

- 8 Click OK.

A new button called *RotateBlurA* appears in the Filter tab, identical to the macro you previously created.



This process creates two files, which are similar to the files you previously created by hand:

- `$HOME/nreal/include/startup/RotateBlurA.h`
- `$HOME/nreal/include/startup/ui/RotateBlurA_UI.h`

Typically, use the MacroMaker to create the initial files, then modify the files with a text editor.

You have created two macros, one by hand and an identical macro with the MacroMaker. Both of these processes create two nearly identical sets of files. The first files create the function, and appear in the `$HOME/nreal/include/startup` directory. The second set of files loads the macro into the interface, and is found in the `startup/ui` subdirectory. The interface files are separate, as you may want the file to appear only in the command line.

For more information, see Chapter 14, “Customizing Shake,” and Chapter 30, “Installing and Creating Macros,” in the *Shake 4 User Manual*.

This tutorial demonstrates how to stitch images with the *AutoAlign* feature, and how to use *SmoothCam* to stabilize footage. You will also use the *QuickPaint* node to create a clean background plate.



Tutorial Summary

- Stitching images
- Stabilizing and stitching background plates
- Creating a clean plate with *QuickPaint*

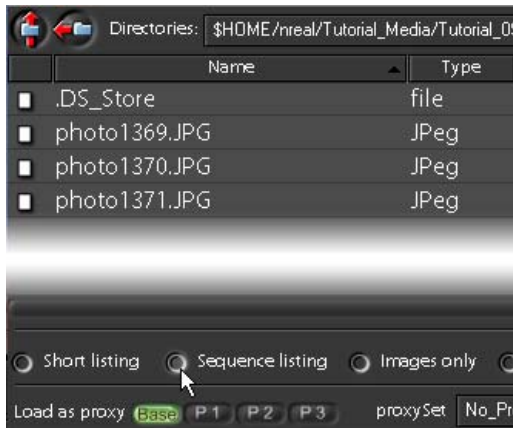
Stitching Images

The *AutoAlign* node lets you stitch up to three images to create a continuous image for backgrounds, panoramas, and other similar elements. The example in this tutorial uses still images, but you can also combine moving footage and *Shake* will stitch the separate plates together as one moving image.

To stitch images using the *AutoAlign* node:

- 1 From a new Shake script, insert an *Image-FileIn* node.
- 2 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_09/images/stitching` directory.

You'll see one image sequence, `photo1369-1371@.JPG`. By default, Shake assumes that sequentially numbered frames are part of an image sequence, and displays a single file name in the File Browser. These are the still images you want to read in, but you don't want to read them as an image sequence.
- 3 Deselect the Sequence Listing button to view the directory contents as individual files.



- 4 Select `photo1369.JPG`, `photo1370.JPG`, and `photo1371.JPG`, then click OK.
- 5 Insert a *Transform-AutoAlign* node, then connect the photo images to it, like this:



Now that the the images are connected, you can tell *AutoAlign* to analyze and stitch them.

There are two different modes for analyzing the pixel data: precise and robust.

- 6 In the *AutoAlign1* parameters, change mode to robust.



- 7 Click the analyze button.



When Shake completes the analysis, you'll see a simple stitch operation, without blending or color matching.

Note: If you do not see acceptable results, switch from robust mode to precise mode, then click the analyze button again.

- 8 In the *AutoAlign1* parameters, change the blendMode to smartBlend.

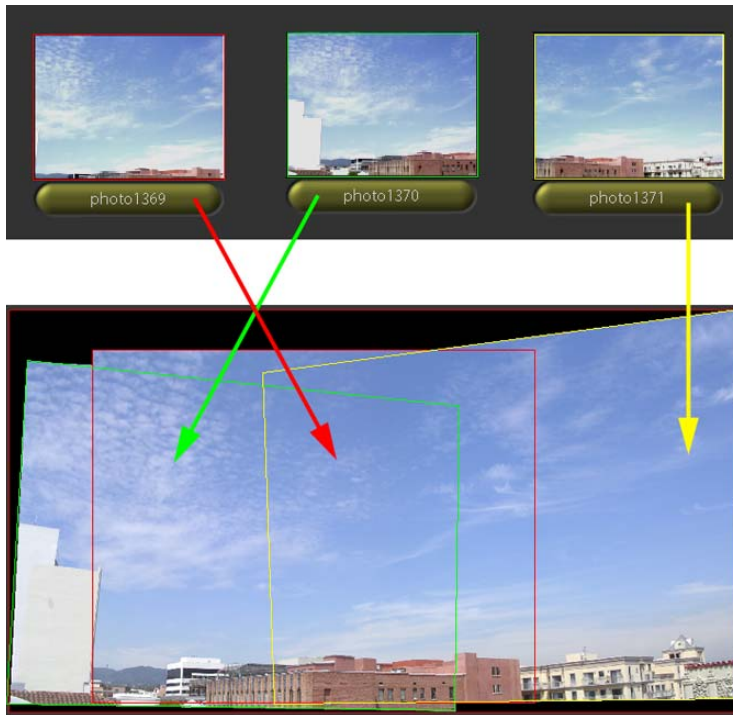
Depending on your system, it may take a few moments to recalculate. You may notice a few places where you can see the seams between the images.



- 9 Turn on the matchIllum option to fix the color match between images.



Now you should have a fairly good background plate. Notice that Shake arranged and stitched the images correctly, even though they were not connected in the order that they should appear in the panorama.



This is possible because there is enough overlap between the images (about 20 to 30 percent) to allow accurate calculation and positioning.

Also notice that two of the images are distorted to conform to the “locked” image, *photo1369*, represented by the red border in the above illustration.

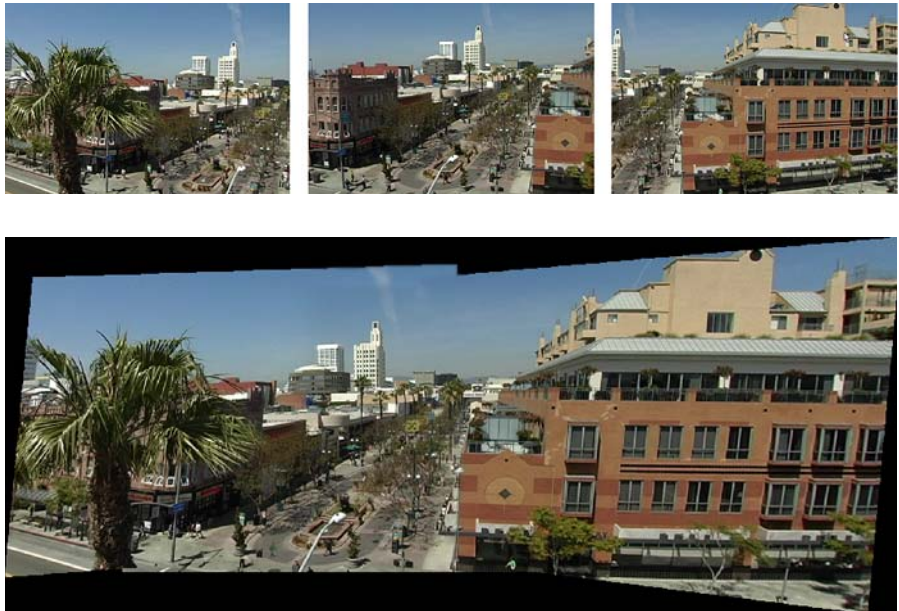
This image is attached to the first input of the *AutoAlign1* node, and is designated as the “locked” plate—meaning that it will not be distorted.



From the lockedPlate list, you can choose an alternate image and the other images will be distorted to match it.

Stabilizing and Stitching Background Plates

AutoAlign also stitches moving images. In this section, the process is almost identical to the previous example, except you will first stabilize the images to eliminate unwanted movement. You will open an existing script, stabilize the images with the *SmoothCam* node, then stitch them together with *AutoAlign*.



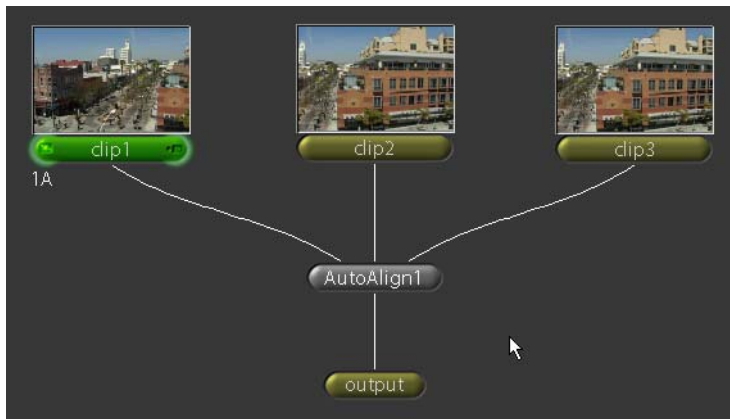
Note: If you do not stabilize your footage before using *AutoAlign*, then the images should be matched to the movement of the image specified as the “locked plate.”

You may already be familiar with the *Stabilize* node and wonder why we're not using that to stabilize the footage. These clips were all shot without the benefit of a tripod or camera mount—that's right, they're handheld. The *SmoothCam* node can stabilize and adjust for translation on all three axes (x, y, and z), and can also stabilize for zoom and perspective changes. The *Stabilize* node adjusts for transformations on the x- and y-axes only.

To open the *stitch-2_start.shk* script:

- 1 Click the Load button at the top of the Node View.
- 2 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_09/scripts` directory.
- 3 Select the *stitch-2_start.shk* script and click OK.

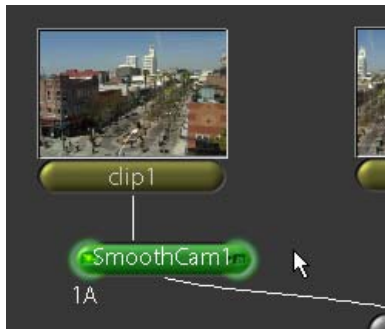
This node tree looks similar to the previous example.



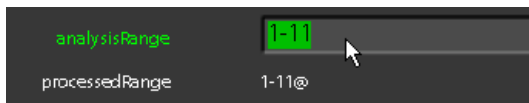
If you stitch these images now, the *AutoAlign* node will incorporate all the camera movement into the stitched background. If you don't want movement in the clips, you need to stabilize them first.

To stabilize the *clip1* image:

- 1 Select the *clip1* node, then insert a Transform–*SmoothCam* node.



- 2 In the *SmoothCam1* parameters, the frame range (1-11) should appear at the top of the parameters. (If not, type “1-11” in the analysisRange field.)



- 3 Set the steadyMode to lock.

Choose lock instead of smooth because you want to remove all camera movement from the shot, if possible. Now, you need to specify which types of movement to eliminate.

- 4 Expand the lockdown subtree.
- 5 Turn on translateLock, rotateLock, zoomLock, and perspectiveLock.



In the “real” world, you’ll want to review your footage and see which of these options are necessary. If, for example, there’s no zooming in the shot, then you wouldn’t need to spend calculation time on zoomLock.

- 6 Click the analyze button and sit back while the *SmoothCam* node stabilizes your footage.

- 7 Now insert Transform–*SmoothCam* nodes for the *clip2* and *clip3* nodes, and repeat the stabilization process for these images.

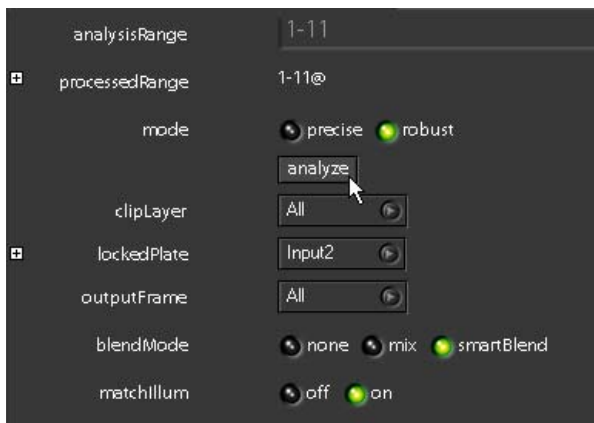
When you're finished, you may want to load each *SmoothCam* node into the Viewer and then create Flipbooks to preview the results.

To stitch the clips together:

- 1 Double-click the *AutoAlign1* node to load it into the Viewer and the Parameters tab.
- 2 Again, check the frame range for analysis. If it's not correct, type "1-11" in the analysisRange field.



- 3 Set up your options as shown below.



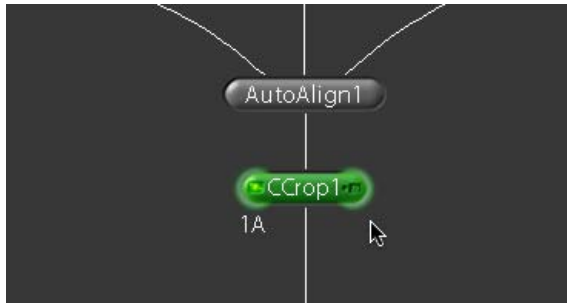
- 4 Click the analyze button to calculate the results.

So far, things should be working pretty well and you should see a decent background plate.

Now add a few more nodes to crop out the edges and remove some of the lens distortion on the stitched image.

To crop the stitched plate:

- 1 Select the *AutoAlign1* node, then insert a *Transform-Crop* node.



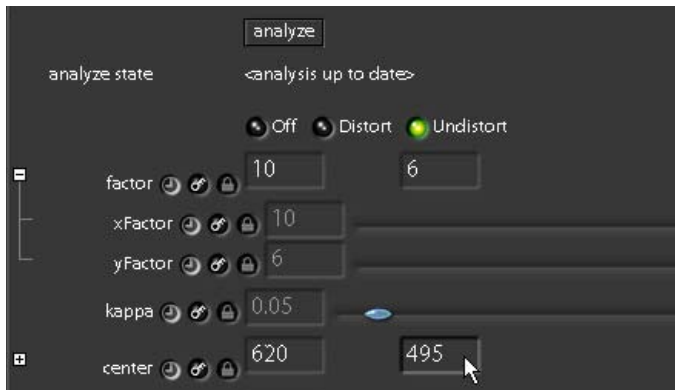
- 2 In the Viewer, drag the borders to crop the portion of the image you want to use in a composite.



Now you'll remove some of the distortion in the image. You can remove lens distortion before you stabilize and stitch; otherwise, the locked plate image influences any motion or perspective in the stitched image. In the spirit of simplification, use the *LensWarp* node at this point to remove some of the distortion.

To adjust for lens distortion:

- 1 Select the *CCrop1* node, then insert a *Warp-LensWarp* node.
- 2 Load *LensWarp1* into the Parameters tab and select Undistort.



- 3 Expand the factor subtree and set xFactor to 10, and yFactor to 6.
 - 4 Set kappa to 0.05.
 - 5 Change the center to 620 (xCenter) and 495 (yCenter).
 - 6 Click the analyze button.
- That's should do it. It's not perfect, but it's a quick little fix.
- 7 Double-click the *output* node and you'll see the result.



Creating a Clean Plate With QuickPaint

A clean plate is a background image with unwanted material removed—a wire rig, boom mike, street clutter, and so on. To create a clean plate, you typically erase the parts of the image you don't want to see and replace them with a “cleaned” version of the background image.

There are many ways to accomplish this. You might immediately think “Clone brush” in *QuickPaint*, and you are in fact partly correct. You can create a clean plate by cloning image information, but you’ll also use the Reveal brush to layer an image from another frame over the frame you want to clean.



The images for this example are courtesy of Mr. Ron Brinkmann from his film, *Hope*, © 2002 Precipice Pictures.

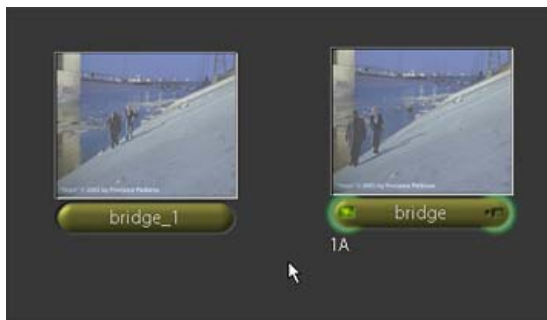
To load the tutorial images:

- 1 Start from a new Shake script, then insert an *Image-FileIn* node.
- 2 In the File Browser, navigate to the `$HOME/nreal/Tutorial_Media/Tutorial_09/images/paintfix` directory.

As before, you will see the two images listed as one sequence, *bridge.1,38#.jpg*.

- 3 Deselect the Sequence Listing option.
- 4 Select *bridge.001.jpg* and *bridge.038.jpg* from the file list, then click OK.

The two images are read into the script.



Rename the nodes for purposes of clarity in this tutorial.

- 5 Double-click the node that reads in the *bridge.0001.jpg* image, and rename it *bridge001*.
- 6 Double-click the node that reads in the *bridge.0038.jpg* image, and rename it *bridge038*.

When creating a clean plate, try to select frames with as little overlap of the foreground characters as possible. Because this is not always practical, you should always request a clean plate shot whenever possible (and save yourself the tedious task of going through this tutorial).



bridge001



bridge038

You will modify frame 1 (in this case, named *bridge001*) because the people are smaller, which means less area to clean up.

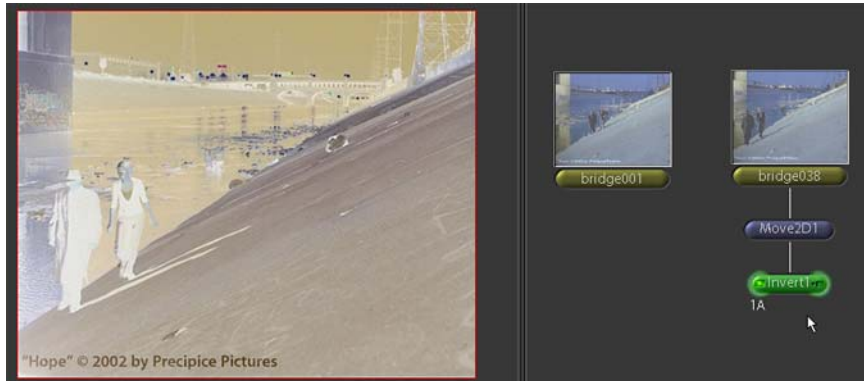
Before you can use one image to fix the other, you need to align the two plates.

To align the *bridge001* and *bridge038* plates:

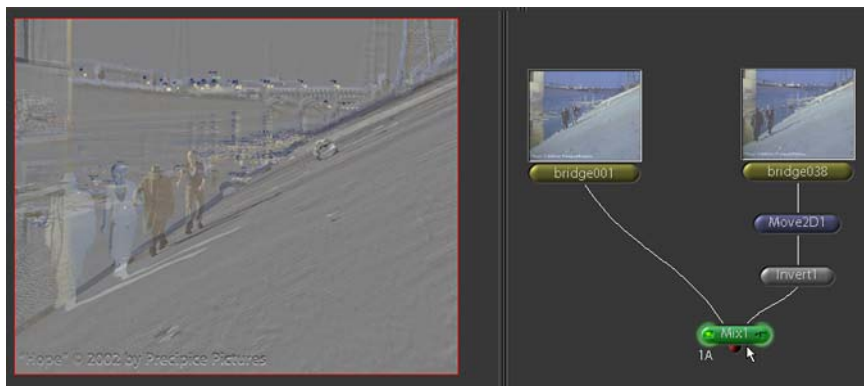
- 1 In the Node View, select the *bridge038* node, then attach a Transform–Move2D node.



- 2 Add a *Color-Invert* node to the *Move2D1* node.



- 3 Select the *bridge001* node and add a *Layer-Mix* node.
- 4 Connect the *Invert1* node to the second input of the *Mix1* node.



The combination of *Invert* and *Mix* let you clearly see the differences between the two images.

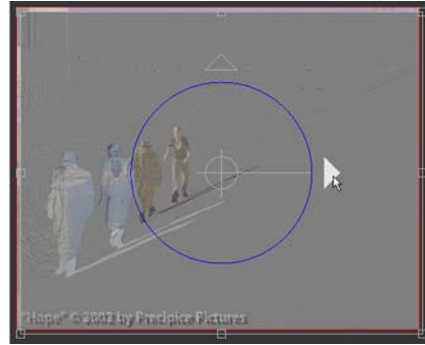
- 5 Load the *Mix1* node into the Viewer (click the left side of *Mix1*) and load the *Move2D1* node into the Parameters tab (click the right side of the *Move2D1* node).
The *Move2D* onscreen controls appear.
- 6 Adjust the *Move2D1* pan settings until the alignment of the plates is exact.



The images are aligned when most of the background noise disappears.



Not aligned

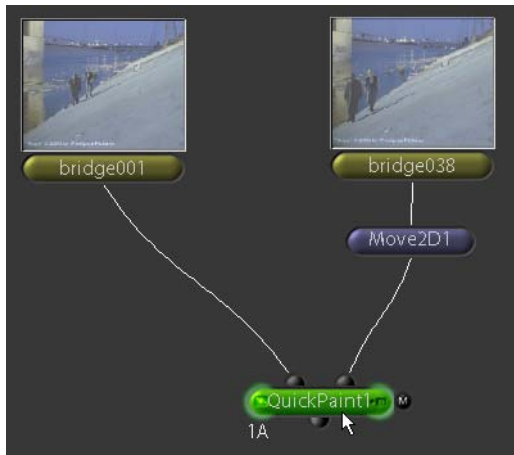


Aligned

- 7 In the Node View, select the *Invert1* node, then press Delete.
- 8 Select the *Mix1* node, then Command-click or Control-click the *QuickPaint* command button in the Image tab.



The *Mix1* node is replaced by the *QuickPaint1* node.



The next step is important:

- 9 In the *QuickPaint* controls in the Viewer shelf, click the “frame” button twice until it reads “persist.”



When Persist (Persistent) mode is set, the paint strokes remain on all frames. If you’re curious about the other paint modes, Frame places the paint strokes only on the frames displayed when you paint the strokes. The third mode, Interp (Interpolation), uses the frames with paint strokes as keyframes, and blends or animates the strokes between the frames.

To continue, you want to paint over the actors to create a clean plate. You’ll use the Reveal Brush to paint from the *bridge038* image into the first input image.

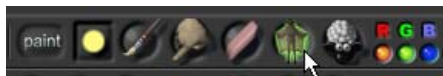
To paint with the Reveal Brush:

- 1 In the *QuickPaint* controls in the Viewer shelf, toggle to the hard-edged brush setting.



The hard-edged brush works better for images with lots of grain. Using a soft edge interpolates the color and softens the grain along the edges.

- 2 Select the Reveal Brush.



- 3 In the Viewer, Command-drag or Control-drag to resize the brush, if necessary.
- 4 Drag over the image to paint the strokes with image information from the *bridge038* node.



- 5 If you paint too far to the left, and reveal the actors in frame 38, use the Eraser to tidy up.



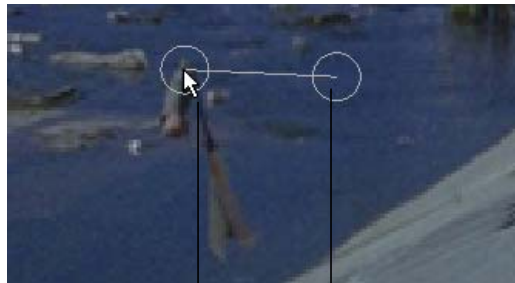
Unfortunately, there is a small area where the actors overlap in the frames, and the Reveal Brush cannot do the whole job. The Clone Brush can help you clean up that last bit by copying some of the nearby color from the water.

To paint with the Clone brush:

- 1 Click the Clone Brush button.



- 2 Press the Shift key and drag to relocate the brush from the clone source.



Shift-drag to position
paint stroke in relation
to clone source.

Clone source

You might need to paint a few strokes, adjusting the position of the clone source.



Once you have a clean plate, add an *Image-FileOut* node and render the clean plate as a new image or image sequence to include in the main composite.



Here you've cleaned one frame. You could apply the same techniques to clean an image sequence or clip. To do so, connect a copy of your original sequence and use it as the second input for the *QuickPaint* node; then, use the *Time View* to offset that copy in time (Tutorial 2 explains how to use the *Time View*), and paint from clean frames to the frames you want to modify.

