NUKE 101

PROFESSIONAL COMPOSITING AND VISUAL EFFECTS

Ron Ganbar

NUKE 101

Professional Compositing and Visual Effects

Ron Ganbar

Peachpit Press
1249 Eighth Street
Berkeley, CA 94710
510/524-2178
Fax: 510/524-2221

Find us on the Web at www.peachpit.com
To report errors, please send a note to errata@peachpit.com
Peachpit Press is a division of Pearson Education

Senior Editor: Karyn Johnson
Development Editor: Corbin Collins
Production Editor: Cory Borman
Technical Editor: Mark Christiansen
Copyeditor: Kelly Kordes Anton
Proofreader: Scout Festa
Indexer: Valerie Haynes Perry
Interior Design and Composition: Kim Scott, Bumpy Design
Cover Design: Charlene Charles-Will
Cover Illustration: Alicia Buelow
Cover images: *Adrenalin Lemmings* (2008) by Crew 972, www.crew972.com

ISBN-13: 978-0-321-73347-4
ISBN-10: 0-321-73347-9

9 8 7 6 5 4 3 2 1

Printed and bound in the United States of America

This page intentionally left blank

“I don’t always like writing, but I very much like having written.”
—William Gibson

CONTENTS

Introduction ix

CHAPTER 1: Getting Started with Nuke 1

Components of the Graphic User Interface 2
- The Content menu 3
- A rundown of the various panels 3
- The menu bar 4
- The contextual menu 6
- Hot keys 6

Nodes 7
- Creating a node 7
- The Read node 9
- The File Browser 9

The Viewer 11
- Navigating the Viewer 12
- Using the Viewer 13
- Viewer inputs 14
- Playing a clip in the Viewer 15

Framecycler 16

CHAPTER 2: Touring the Interface with a Basic Composite 19

Working with Process Trees 20
Creating a Simple Process Tree 22
Merging Images 24
- Merging premultiplied images 26
- Saving Nuke scripts 29

Inserting and Manipulating Nodes in the Tree 30
- Inserting, creating, branching, and replacing nodes 30
- Connecting nodes 32
- Selecting nodes 33
- Arranging nodes 34
- Disabling and deleting nodes 34

Changing Properties 35
- Color correcting the image 36
- Using the Properties Bin 37

Adjusting properties, knobs, and sliders 38
Using the Color Wheel and Color Sliders panel 40
Using the Animation menu 41
Rendering 42
Using the Write node 42
Naming file sequences 43
Delving Deeper into the Merge Node 47
Using the Shuffle node 50
Viewing a composite without rendering 51
Creating Animation with Keyframes 52
Indicators on nodes 56
Rendering a new version and comparing 57
Comparing images 58

CHAPTER 3: Compositing CGI with Bigger Node Trees 61

Working with Channels 62
Bringing in a 3D render 63
Viewing channel sets with the Viewer 63
Working with Contact Sheets 65
Using the Bounding Box to Speed Up Processing 66
Linking Properties with Expressions 69
Slapping Things Together: Foreground Over Background 71
Building the Beauty Pass 73
Working down the pipe 73
Splitting the tree 76
Using the ShuffleCopy Node 84
Manipulating Passes 85
Adding Other CGI Images 87
Placing CGI Over Live Background 91
Using the Mask Input 93

CHAPTER 4: Color Correction 97

Understanding Nuke's Approach to Color 98
Color Manipulation Building Blocks 99
Dynamic range 100
Using an I/O Graph to Visualize Color Operations 105
Creating Curves with ColorLookup 111
Color Matching with the Grade Node 113
Using the Grade node 113
Using CurveTool to match black and white points 118
Matching midtones by eye 122

Achieving a “Look” with the ColorCorrect Node 125
Using the ColorCorrect node 125
Using the mask input to color correct a portion of the image 131

CHAPTER 5: 2D Tracking 135

Tracker Node Basics 136
Stabilizing a Shot 137
Tracking Four Points 143
Understanding tracking points 143
Tracking a picture in the frame 144
Changing the Tracker settings 146
Replacing the picture 148
Adjusting the source pins 152
Adding motion blur 154

CHAPTER 6: RotoPaint 159

Introducing RotoPaint’s Interface 160
Painting strokes 162
Editing strokes 163
Painting in vectors 164
Erasing and deleting strokes 165
Drawing and editing shapes 166
Animating a shape 168
The Curve Editor 170
Painting in Practice 174
Using Paint for wire removal 175
Dust removal with Paint 176
Split-Screening Twins with Roto 178
Combining Paint, Roto, and Animation 185
Compositing with the KeyMix node 187
Working with the Stroke/Shape List window 188
Using the Dope Sheet 191

CHAPTER 7: Keying 197

Introducing Nuke’s Keying Nodes 199
HueKeyer 201
The IBK: Image Based Keyer 203
Keylight 210
Combining Keyer Nodes Using the Tree 216
Erode, Dilate, and Erode 217
Spill suppressing with HueCorrect 220

CHAPTER 8: Compositing Hi-Res Stereo Images 225

Using the Project Settings Panel 226
The Root tab 226
Nonlinear images and lookup tables (LUTs) 227
The Views tab 229
Setting Up a High-Res Stereo Script 230
Setting formats 231
Working with LUTs 233
Stereo views 235
Using proxies 238
Creating stereo-view proxies efficiently 243
Compositing a Stereo Project 246
Retiming elements 246
Compositing the two elements together 249
Changing properties for a single view 251
Rendering and Viewing Stereo Trees 253

CHAPTER 9: The Nuke 3D Engine 257

3D Scene Setups 258
Moving Images with a 3D Scene 259
Setting up a Nuke 3D scene 259
Navigating the 3D world 261
Importing a camera 263
Creating a cube 265
Reconcile3D: Transforming 3D Data into 2D Data 268
Setting up a Reconcile3D node 268
Using Reconcile3D's output with a Tracker node 273
Final Disclosure 277

CHAPTER 10: Camera Tracking 279

Calculating Reflection Movement Using Camera Tracking 280
3D Tracking in Nuke 281
Tracking features 282
Solving the camera 284
Creating the scene 286
Loading a Pre-Generated CameraTracker Node 287
Aligning the Scene 287
Creating the Reflection 292
ScanlineRender nodes 293
Creating the reflective surface 296
Environment light and specular material 298
Cutting the reflection to size 302

CHAPTER 11: Camera Projection 305

Building a Camera Projection Scene 306
Tweaking the Geometry 315
Animating the Camera 316
Tweaking the Texture 318
Using a SphericalTransform to Replace Sky 320
Compositing Outside the ScanlineRender Node 323
Cloning nodes 323
Final adjustments 325
2D Compositing Inside 3D Scenes 326
Importing Photoshop layers 326
Creating the frame 332
Compositing the screen into the 3D scene 334
Rendering the Scene 336

CHAPTER 12: Customizing Nuke with Gizmos 337

About Safety Areas 338
Building the Gizmo's Tree 339
Creating User Knobs 342
Scripting with a Little TCL 348
Testing the Gizmo's Tree 352
Wrapping in Groups 353
Manipulating the Nuke Script in a Text Editor 359
Turning a Group into a Gizmo 362
Installing the Gizmo 363
Testing the Gizmo 363
Using the Viewer Input Process 364
More About Gizmos 365

APPENDIX I: Customizing Nuke with Python 367

Python Scripting Basics 368
Creating a Button with Python 369
Adding a Hot Key 371
Making Customization Stick with the Menu.py File 372
Creating and adding an icon 372
Other uses for menu.py 375

Index 377

See last page of this eBook for instructions on downloading your lesson files.

INTRODUCTION

The Foundry's Nuke is fast becoming the industry leader in compositing software for film and TV. Virtually all the leading visual effects studios—ILM, Digital Domain, Weta Digital, MPC, Framestore, The Mill, and Sony Pictures Imageworks—now use Nuke as their main compositing tool. This is not surprising, as Nuke offers a flexible node-based approach to compositing, has a native multi-channel workflow, and boasts a powerful integrated 3D compositing environment that delivers on the artist's needs.

Nuke was first developed as the in-house compositing tool at Digital Domain, the visual effects studio behind the *Terminator* series, *The Fifth Element*, *Tron: Legacy*, *The Curious Case of Benjamin Button*, and other major films. The software has been developed by artists for artists to meet the immediate needs of actual top-level productions. Nuke is now developed by The Foundry (www.thefoundry.co.uk), which remains committed to making Nuke the best tool for compositing artists working in the trenches.

ABOUT THIS BOOK

Learning Nuke is a must for visual effects artists who want to master high-end compositing techniques and artistry. My goal with this book is to get you up-and-running with the program and give you the skills you need for doing your own compositing projects in Nuke.

Who this book is for

This book is for anyone interested in learning Nuke. Whether you're an artist experienced in using Adobe After Effects, Autodesk Flame, Apple Shake, or Eyeon Fusion, or you only have a basic understanding of compositing and image manipulation concepts, this book will guide you through the necessary theory and practice you need to use Nuke—from a basic level to Nuke's more advanced toolset.

How this book is organized

This book was written as a series of lessons, each focusing on a part of the interface, a tool, or a technique. Chapters 1 through 3 discuss Nuke basics, which are important for understanding where things are and how to create simple composites. Chapters 4 through 7 cover important tools and techniques. In Chapter 8 and onwards, advanced tools and techniques are explained.

What this book covers

This book teaches how to use Nuke from its very basic interface to its very advanced toolsets, including the 3D engine, Camera Projection, and Camera Tracking. Although the book teaches a fair amount of compositing theory, there is not enough space here to cover that topic in depth. Some of the theory discussed in this book may be new to you, but my intention is to cover just enough to understand how to use Nuke. If you want to dive further into the theory, two of my favorite books are Ron Brinkmann's *The Art and Science of Digital Compositing* and Steve Wright's *Digital Compositing for Film and Video*.

How to use this book

As you advance through the chapters in this book, the later lessons rely on knowledge you learned in the previous lessons. Chapter 2 relies on what you learned in Chapter 1, and so on. Because of this, I recommend completing the exercises in the chapters in order.

In the book you will find explanatory text and numbered steps. Ideally, you should complete each numbered step exactly as it is written—without doing anything else (such as adding your own steps). Following the steps exactly as written will give you a smooth experience. Not going through the steps as they are written might result in the next step not working properly, and could well lead to a frustrating experience. Each series of steps is also designed to introduce you to new concepts and techniques. As you perform the steps, pay attention to why you are clicking where you are clicking and doing what you are doing, as that will truly make your experience a worthwhile one.

You can use this book on your own through self-study or in a classroom.

- **Using the book for self-study:** If you're reading this book at your own pace, follow the instructions in the previous paragraph for your first read-through of the chapters. However, as you are not limited by any time frame, I recommend going through chapters a second time, trying to do as much of the work without reading the steps. Doing so can help you better understand the concepts and tools being taught. Also, the book leaves a lot of room for further experimentation. Feel free to use the tools you're learning to take your compositions further the second time you run through a chapter.
- **Using the book in a classroom setting:** You can use this book to teach Nuke in a classroom. As a course, the material is designed to run for roughly 40 hours, or five eight-hour days. I suggest the trainer run through a chapter with the students listening and writing down notes, explaining the steps as they are shown on-screen to the class while taking questions and expanding on the text where

necessary. Once a chapter has been presented from start to finish, give students time to run through the same chapter on their own in the classroom in front of a computer, using the book to read the instructions and follow the steps. This second pass will reiterate everything the trainer has explained and, through actual experience, show the students how to use the software with the trainer still there to answer questions and help when things go wrong.

INSTALLING NUKE

While this book was originally written for Nuke version 6.2v1, The Foundry updates Nuke on a regular basis and the lessons can be followed using more recent updates. Small interface and behavior updates might slightly alter the Nuke interface from version to version, especially for so-called "point" updates (such as if Nuke version 6.3 were released). I recommend using this book with Nuke version 6.2v1 if you haven't already downloaded the most current version and you want the exact results as shown in the book.

You can download Nuke in a variety of versions from The Foundry's web site at www.thefoundry.co.uk as discussed in the next sections.

Different flavors of Nuke

Nuke comes in three different flavors with different features at different prices. There is only a single installation file for Nuke, but the license you purchase determines which type of Nuke you will be running. The Foundry offers a 15-day trial license, so you can try it before you purchase it (see "Getting a trial license" later in this section).

Here are the three flavors of Nuke.

1. **Nuke PLE (Personal Learning Edition):** This license (or lack of) is free—as in, you pay nothing. You can install Nuke on your computer and not purchase a license. With the PLE you can use Nuke as much as you want, although certain limitations apply. These include the placement of a watermark on the Viewer and on renders, and the disabling of WriteGeo, Primatte, Framecycler, and Monitor Output.
2. **Nuke:** This is regular Nuke—the flavor this book covers. Nuke requires a trial license or regular paid license, which should cost about $4,500.
3. **NukeX:** This license includes all the regular Nuke features with a few additional high-end tools. These tools include the CameraTracker, PointCloudGenerator, LensDistortion, DepthGenerator, Modeler, FurnaceCore plug-ins, and PRmanRender (allowing for Renderman integration). NukeX costs in the region of $7,000. Chapter 10 covers the Camera Tracker and shows how to use it under the NukeX license; however, the exercises in the chapter can also be done without a NukeX license.

Downloading Nuke

To download Nuke, follow these steps.

1. Go to www.thefoundry.co.uk/products/nuke/product-downloads/.
2. Select the latest copy of Nuke for your operating system (Mac, Windows, or Linux). You can also download older versions of Nuke if necessary.
3. If you're downloading the latest version, you need to register and then you are directed to the download page. (Downloading older versions from the archive does not require registration.)
4. Follow the instructions for installation on your specific operating system.

Getting a trial license

After successfully installing Nuke, when you double-click the Nuke icon it explains that because you don't have a license yet, you can use Nuke under the PLE license. If you would like to use a fully licensed Nuke or NukeX, you will have to buy Nuke, rent Nuke (both available on the Foundry's web site shown below), or get a free 15-day trial license.

As there is no functional difference between getting a Nuke trial license or a NukeX trial license, I recommend getting a NukeX trial license. To get your free 15-day trial NukeX license, do the following:

1. Go to www.thefoundry.co.uk/products/nuke/try/.
2. Log in or sign up to get to the Free Trial page.
3. In the Free Trial page, fill in the form.

The System ID, which is the first entry to fill, is the unique code of your computer—the free license will be locked to that computer. The link to the right of the entry field explains where to find this number on your computer.

4. After you complete the form and click Continue, follow the rest of the instructions on the Foundry's web site for how to install the license on your operating system.

Staying up to date

Because Nuke is updated often, please visit the web site at www.peachpit.com/nuke101 for periodic updates to the book. I will strive to publish notes about behavior in new versions of Nuke as they come out if they differ from what's written in the book.

Also, I will periodically publish additional information and more tutorials on the web site that will add further content to the book.

ADDITIONAL TECHNICAL REQUIREMENTS

Nuke is a very powerful piece of software, even though its system requirements are pretty low. If you bought your computer in the last couple of years, you are probably OK. The requirements are listed on The Foundry web site but there are three things you should really check:

- Workstation-class graphics card, such as NVIDIA Quadro series, ATI FireGL series, R3D Rocket, or newer. Driver support for OpenGL 2.0.
- Display with at least 1280x1024 pixel resolution and 24-bit color.
- Three-button mouse. This kind of mouse is really a must as Nuke uses the middle mouse button extensively. A scroll wheel, by the way, can serve as the middle mouse button.

To copy the exercise files to your computer, you will need a DVD drive as well.

For a full list of Nuke's system requirements, visit www.thefoundry.co.uk/products/nuke/system-requirements.

ABOUT THE BOOK'S DISC FILES

At the back of the book you will find a DVD-ROM disc containing the files you need to complete the exercises in this book (or if you bought the ebook version, you'll be presented with a link to the files). The files are a mix of a real production that I or my colleagues created in recent years and some shot elements intended for use with this specific book.

What's on the disc

Each chapter has its own directory. Copy these directories to your hard drive to use the files properly. Some chapters use files from other chapters, so you need to copy all the directories to your hard drive.

How to install the files

1. Insert the Nuke101 DVD into your DVD drive.
2. Create a directory on your hard drive and name it **NukeChapters**.
3. Drag the chapters directory from the DVD into the NukeChapters directory on your hard drive.

ACKNOWLEDGMENTS

I've been teaching compositing since 2001. When Nuke started becoming the tool of choice for a lot of the studios around me, I decided to write a course that focused on it. I started writing the course in the spring of 2009 with help from The Foundry, whose staff was very kind and forthcoming. I would specifically like to thank Vikki Hamilton, Ben Minall, Lucy Cooper, and Matt Plec.

I finished writing the original course around the autumn of 2009. I taught it several times at Soho Editors Training in London, which was kind enough to let me try out the new course at their training facility. The course was well received, so between sessions I updated, corrected, and expanded on the original course.

About a year after that I approached Peachpit Press with the idea of turning the course into a book. Karyn Johnson, the book's senior editor, took on the project and after a long digestion period I sat down and started adapting the course into a book. Karyn made sure I had the best support I could possibly have, and with the help of the wonderful team at Peachpit, including Corbin Collins and Kelly Kordes Anton, I managed to complete the book to the high standard Peachpit expects of their writers. Thanks also go out to the kind friends and colleagues who gave me materials to use for the book: Alex Orrelle, Alex Norris, Hector Berrebi, Dror Revach, Assaf Evron, Menashe Morobuse, and Michal Boico.

It was quite a ride. You can see it took me three paragraphs to cover it all. But throughout this long period, which sometimes felt like it would last forever, my wife, Maya, and my two sons had to bear with my long days and long nights of writing, gave me the quiet and solitude I needed, and believed (and prayed) that I would finish the book. And so I have.

In so many ways, this book is for them.

Throughout this title you will see references to lesson or resource files on a disc. Please note that these files are available to eBook readers via high-speed download. Please **click here** to go to last page in this eBook for the download location and instructions.

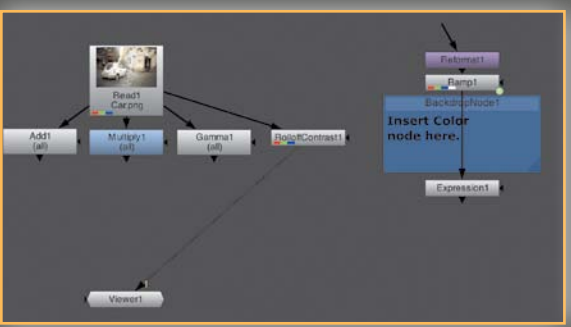
This page intentionally left blank

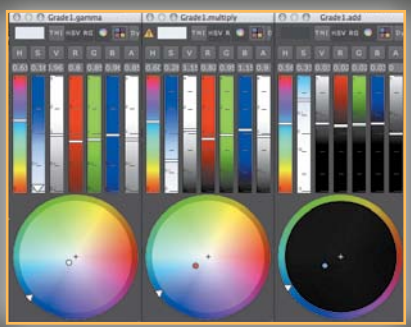
4

COLOR CORRECTION

Wow. This is a bit naive. Calling a lesson “Color Correction.” It should be a whole course on its own. But this book is about more than that, and limited space reduces color correction to a chapter. So let me start by explaining what color correction means.

Color correction is one of the most fundamental things you can do to an image. It refers to any change to the perceived color of an image. Making an image lighter, more saturated, changing the contrast, making it bluer—all of this is color correction. There are a lot of uses for color correction. The most obvious one is to make an image look different as a result of a stylistic decision. But you can also color correct to combine two images so they feel like part of the same scene. This is performed often in compositing when the foreground and the background should have colors that work well together. There are plenty





more uses for changing the color of an image. An image might be a mask or an alpha channel that needs to have a different color in some way—to lose softness and give it more contrast, for example.

Whatever reason you have for color correcting an image, the color correction will work according to the way Nuke handles color. Nuke is a very advanced system that uses cutting-edge technology and theory to work with color. It is important to understand Nuke's approach to color to understand color correcting within Nuke.

UNDERSTANDING NUKE'S APPROACH TO COLOR

Nuke is a 32-bit float linear color compositing application. A bit of a fancy description there, with potentially new words. I explain this bit by bit:

- *32-bit*: That's the amount of bits used to hold colors. Most compositing and image-manipulation programs are 8-bit, allowing for 256 variations of color per channel (resulting in what's referred to as "million of colors" when combining the three color channels). This is normally fine for displaying color, but is not good enough for some calculations of operations, and may result in unwanted results such as *banding*—inaccurate display of gradients where changes in color happen abruptly instead of smoothly. 32-bit allows for a whopping 4,294,967,296 variations of color per channel. That's a staggering amount that results in *much* more accurate display of images and calculations of operations. 8- or 16-bit images brought into Nuke will be bumped up to 32-bit, although that doesn't add any detail, it just enables better calculations from that point onwards.

- *Float:* Normally the color of an image is represented between black and white. In 8-bit images, for example, the 256 color variations are split evenly between black and white—so the value 1 is black, the value 256 is white, and the value 128 is a middle gray. But what about colors that are brighter than white? Surely the whiteness in the middle of a lit light bulb is brighter than a white piece of paper? For that reason, there are colors that are brighter than white called super-whites. There are also colors that are darker than black called sub-blacks (but there isn't a real-world analogy that can be used here short of black holes). Using 8 bits to describe an image simply doesn't allow enough room to describe colors beyond black and white. These colors get *clipped* and are simply represented as black or white. However, in 32-bit color, there is plenty of room and these colors become representable. As mentioned before, 8-bit color is normally enough to display images on-screen. Furthermore, the computer monitor can still display only white—and nothing brighter. However it is still very important to have access to those colors beyond white, especially when color correcting. Darkening an image that has both a piece of white paper and a light bulb in it will leave the light bulb white, while darkening the paper to a gray color results in an image that mimics

real-world behavior and looks good and believable. Doing the same with a non-floating image will result in the white paper and the light bulb looking the same gray color—which will be unconvincing.

- *Linear:* Linear can mean lots of things. In terms of color, I mean *linear color space*. A computer monitor doesn't show an image as the image appears in reality, because the monitor is not a *linear* display device. It has a mathematical curve called *gamma* that it uses to display images. Different monitors can have different curves, but most often they have a gamma curve called *sRGB*. Because the monitor is not showing the image as it appears in reality, images need to be "corrected" for this. This is usually done automatically because most image capture devices are applying an sRGB curve too, in the opposite direction. Displaying a middle gray pixel on a monitor only shows you middle gray as it's being affected by the gamma curve. Because your scanner, camera, and image processing applications all know this, they color correct by applying the *reverse* gamma curve on this gray pixel that negates the monitor's effect. This process represents basic *color management*. However, if your image's middle gray value isn't middle gray because a gamma curve has been applied to it, it will react differently to color correction and might produce odd results. Most applications work in this way, and most people dealing with color have become accustomed to this. This is primarily because computer graphics is a relatively new industry that relies on computers that, until recently, were very slow. The correct way to manipulate imagery—in whatever way—is *before* the gamma curve has been applied to an image. The correct way is to take a linear image, color correct it, composite it, transform it, and then apply a reverse gamma curve to the image to view it correctly (as the monitor is applying gamma correction as well and negating the correction you just applied). Luckily, this is how Nuke works by default.

Still confused? Here's a recap: Nuke creates very accurate representations of color and can store colors that are brighter than white and darker than black. It also calculates all the compositing operations in linear color space, resulting in more realistic and more mathematically correct results.

Nuke has many color correction nodes, but they are all built out of basic mathematical building blocks, which are the same in every software application. The next section looks at those building blocks.

NOTE Nuke color values are displayed and are calculated in what's called "normalized values." This means that instead of defining black at a value of 0 and white at a value of 255, black is still 0, but white is 1. It's a very easy thing to remember that makes understanding the math easier.

COLOR MANIPULATION BUILDING BLOCKS

Color correction is a somewhat intuitive process. Often compositors just try something until they get it right. Understanding the math behind color correction can help you pick the right tool for the job when attempting to reach a specific result—which is better than trial and error. **TABLE 4.1** explains most of these building blocks.

TABLE 4.1 Basic Color Correction Functions

Math function	Node	Explanation	Names in Nuke	Other known names
Add	Add, Grade, ColorCorrect	Adds a constant value to a channel.	Add, Offset	
Multiply	Multiply, Grade, ColorCorrect	Multiplies the channel by a constant value.	Gain, Multiply	Brightness, Contrast, Exposure, Input/Output White
Gamma	Gamma, Grade, ColorCorrect	Applies a gamma curve to a channel.	Gamma	
Contrast	RolloffContrast, ColorCorrect	Applies a contrast curve to a channel. This is also a form of multiplication.	Contrast, RolloffContrast	
Lift	Grade	This function is similar to Multiply and Contrast. It's a contrast curve with a center at white. More below.	Lift	Pedestal, Input/Output Black
Lookup	ColorLookup	Applies a user-defined curve to a channel.	ColorLookup	Curves
Saturation	Adjusts the color intensity by reducing the differences between the RGB channels.	Saturation		

Dynamic range

When dealing with color correction, I usually talk about dynamic range and its parts. *Dynamic range* means all the colors that exist in your image, from the darkest to the brightest color. The dynamic range changes from image to image, but usually you are working with an image that has black and white and everything in between. The parts of the dynamic range, as mentioned, are split according to their brightness value as follows:

- The shadows or lowlights, meaning the darkest colors in the image
- The midtones, meaning the colors in the image that are neither dark nor bright
- The highlights, meaning the brightest colors

In Nuke, and in other applications that support colors beyond white and black (float), there are two more potential parts to the dynamic range: the super-whites, or colors that are brighter than white, and the sub-blacks, colors that are darker than black.

Let's look at these building blocks in several scenarios to really understand what they do and why you might choose one over another.

1. Launch Nuke.
2. Bring in a clip called Car.png by pressing R and navigating to the chapter04 directory.
3. Click Read1, then press 1 on the keyboard to view it.

   It's an image of a car. Did that catch you by surprise?
4. With Read1 selected, go to the Color toolbox and click Add in the Math folder.

   You have now inserted a basic color-correcting node after the car image. Let's use it to change the color of the image and see its effect.
5. In Add1's Properties panel, click the Color Picker button to display the Color Wheel and Color Sliders panel. Play with the R, G, and B colors to see the changes (**FIGURE 4.1**).

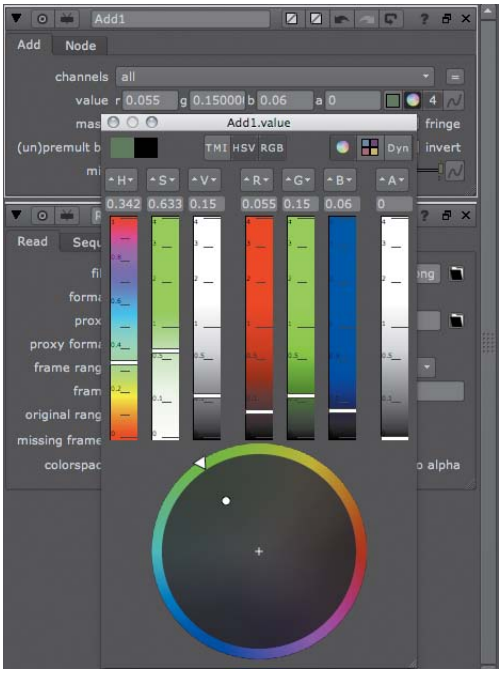


FIGURE 4.1 Using the Color Wheel and Color Sliders panel.

You can see everything changes when playing with an Add node—the highlights, midtones, and even blacks (**FIGURE 4.2**). An Add operation adds color to everything uniformly—the whole dynamic range. Every part of the image gets brighter or darker.

FIGURE 4.2 The whole image is becoming brighter.

6. When you're finished, close the Color Wheel and Color Sliders panel.
7. Select Read1 again and branch out by holding the Shift key and clicking a Multiply node from the Math folder in the Color toolbox.
8. While Multiply1 is selected, press 1 on the keyboard to view it.
9. In Multiply1's Properties panel, click the Color Picker button to display the Color Wheel and Color Sliders panel and experiment with the colors (**FIGURE 4.3**).

FIGURE 4.3 The changes affect the highlights more than the rest of the image.

You can see very different results here. The highlights get a strong boost very quickly while the blacks are virtually untouched.

10. Repeat the above process for the Gamma node. Remember to branch from Read1 (**FIGURE 4.4**).

You can see that gamma mainly deals with midtones. The bright areas remain untouched and so do the dark areas.

FIGURE 4.4 The mid-tones change the most when changing gamma.

You should now have three different, basic, math-based color correctors in your Node Graph that produce three very different results as shown in **FIGURE 4.5**.

FIGURE 4.5 The results from changing Add, Multiply, and Gamma.

Your DAG should look a little like **FIGURE 4.6**.

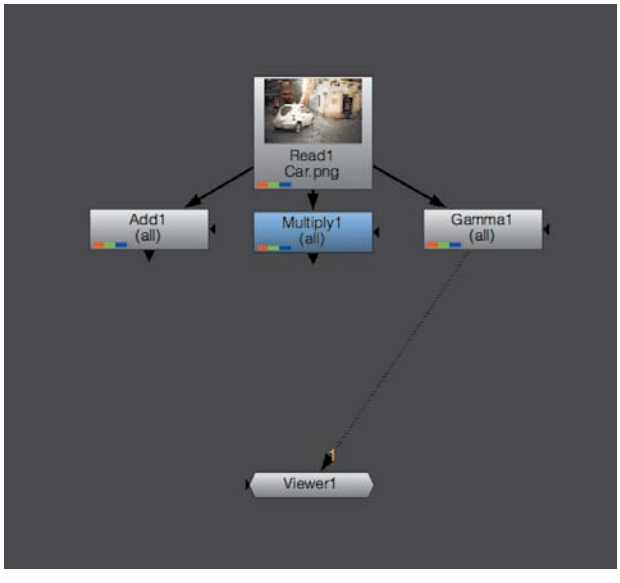


FIGURE 4.6 Branching three color correctors from a node.

Let's try some more color correction nodes.

11. Select Read1 and then Shift-click RolloffContrast in the Color toolbox to create another branch.

    I find it really annoying that they chose to call the Contrast node RolloffContrast, especially since it makes opening it via the Tab key so much harder because typing "contrast" won't display this node.

12. While viewing RolloffContrast1, open its Properties panel and play with the Contrast value (**FIGURE 4.7**).

FIGURE 4.7 A high Contrast value produces a high-contrast image.

    You can see how, when increasing the contrast above 1, the lowlights get pushed down and the highlights are pushed up.

13. Keep the Contrast property above 1 and bring the Center value down to 0.

    The Center property changed what is considered to be the highlight or lowlight. Colors above the Center value will be considered bright and pushed up, and colors below the Center value will be considered dark and pushed down.

You can see the result of the RolloffContrast operation now is very similar to that of the Multiply node. In fact, they are virtually the same. When setting the center value at 0, you lock that value in place. The value 0 is locked in place when multiplying as well.

14. Bring the Center value up to 1.

You haven't gone through an operation called Lift yet, but the RolloffContrast operation is virtually the same as that operation. With Lift, the value 1 is locked in place and the farther the values are away from 1, the bigger the effect. You will go through Lift when you learn about the Grade node later in this chapter.

To wrap up this part of the color introduction, here's an overall explanation:

- When dealing with color, there's usually a need to control the lowlights, midtones, and highlights separately.
- The Add operation adds the same amount of color to every part of the dynamic range.
- The Multiply operation multiplies the dynamic range by a value. This means that a perfect black doesn't change, lowlights are barely touched, midtones are affected by some degree, and highlights are affected the most. It is good to mention that a Multiply operation is virtually the same as changing the exposure in a camera or increasing light. It is the most commonly used color operation.
- The Gamma control is a specific curve designed to manipulate the part of the dynamic range between 0 and 1 (black and white, remember?), without touching 0 or 1.
- Contrast is actually very similar to Multiply, but has a center control. If you place the center point at 0 you get a Multiply node.

USING AN I/O GRAPH TO VISUALIZE COLOR OPERATIONS

And *I/O graph* (input versus output graph) is a great way to understand color operations. The X axis represents the color coming in, and the Y axis represents the color going out. A perfectly diagonal line therefore represents no color correction. The graph shows what the color operation is doing and the changes to the dynamic range.

To view an I/O graph like this, you can bring in a pre-made script I made.

1. Choose File > Import Script to load another script from the disk and merge it with the script you have been building.
2. In the File Browser that opens, navigate to chapter04 and click IO_graph.nk to import it into your current script.

   Notice that when you imported the script (which is only four nodes) all of its nodes were selected. This is very convenient as you can immediately move the newly imported tree to a suitable place in your Node Graph.
3. Make sure the imported tree is not sitting on top of your existing tree. Move it aside to somewhere suitable as in **FIGURE 4.8**.

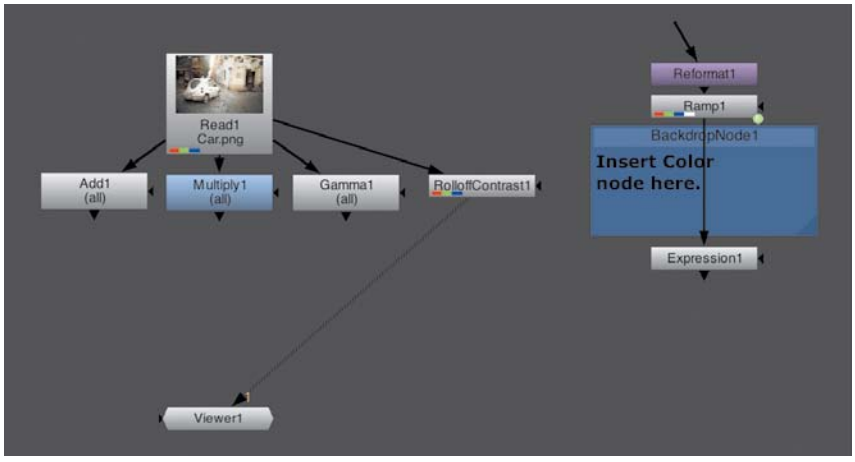


FIGURE 4.8 You now have two trees in your DAG.

4. Make sure you are viewing the output of Expression1.

A quick explanation of the script you imported, node by node:

- The first node is a Reformat node, which defines the resolution of your image. In this case, 256x256. Notice that its input isn't connected to anything. This is a good way to set a resolution for your tree.
- The second node is a Ramp. This can be created from the Draw toolbox. This node generates ramps—in this case, a black to white horizontal ramp from edge to edge.
- The third node is a Backdrop node, used to highlight areas in the tree. You can find it in the toolbox called Other. It indicates where to add your color correction nodes in the next step.
- The fourth and last node is an Expression node. This is a very powerful node. It can be found in the Color > Math toolbox. It lets the user write an expression with which to draw an image. You can do a lot of things with this node, from simple color operations (such as adding or multiplying, though this is wasteful) to complex warps or redrawing of different kinds of images all together. In this case, you use this node to draw values of a horizontal black to white ramp (you have the ramp from above) on-screen as white pixels in the corresponding height in the image. A value of 0.5 in the ramp will generate a white pixel halfway up the Y resolution in the output of the Expression node. The left-most pixel is black in the ramp, and shows as a white pixel at the bottom of your screen. The middle pixel is a value of 0.5 and so shows as a white pixel in the middle of the screen. The right-most pixel is a value of 1 and so draws a white pixel at the top of the screen. All these white pixels together form a diagonal line (**FIGURE 4.9**). Changing the color of the ramp will change the line. This happens on each of the three color channels individually.

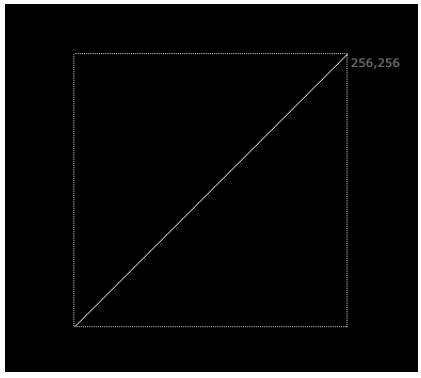


FIGURE 4.9 The I/O graph at its default state.

Let's start using this I/O Graph tree. You will insert a Color node in between Ramp1 and Expression1 and look at the resulting I/O graph.

5. Insert an Add node from the Color > Math toolbox after Ramp1 as shown in **FIGURE 4.10**.

6. Bring the value of Add2's value property to around 0.1.

   You can see, as in **FIGURE 4.11**, that the Add operation changes the whole dynamic range of your graph, and therefore, for any image.

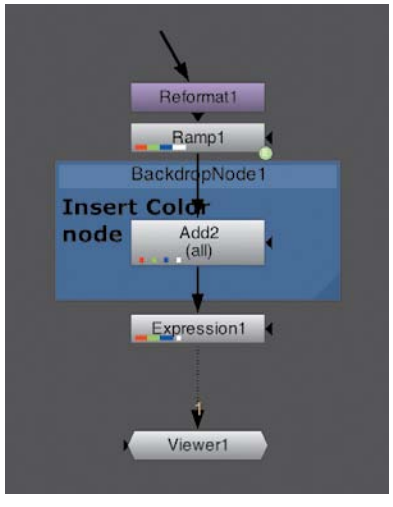


FIGURE 4.10 Add2 has been inserted after Ramp1 and will change your I/O graph.

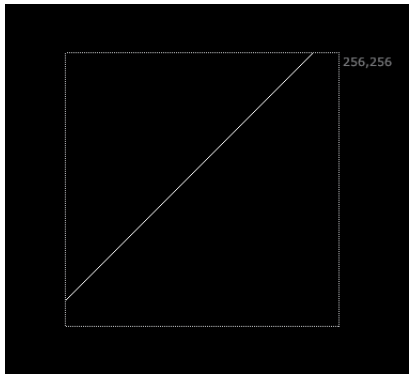


FIGURE 4.11 The whole graph is raised or lowered in unison.

Let's replace your Add node with a Multiply node. You've never done this before, so pay attention.

7. With Add2 selected, Ctrl/Cmd-click the Multiply node in the Color > Math toolbox to replace the selected node with the newly created one.

8. Increase and decrease Multiply2's value.
9. You can also click the color wheel and change the RGB channels individually (**FIGURE 4.12**).

   The Multiply operation has more effect on the highlights than the lowlights. You can see when you are moving the slider that the 0 point stays put, and the further away you go from 0 the effect becomes stronger.

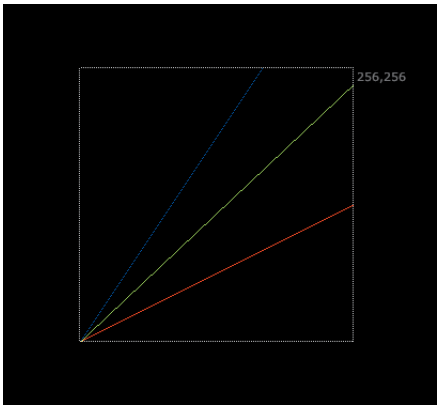


FIGURE 4.12 The graph changes more the further away it is from 0.

Let's try Gamma. Maybe you don't know what a Gamma curve looks like. Well, here's your chance.

10. Replace Multiply2 with a Gamma node from the Color or Math toolbox by holding down Ctrl/Cmd and clicking Gamma from the Color > Math toolbox.
11. Load Gamma2's Color Wheel and Color Sliders panel and play with the sliders for R, G, and B.

    You should now get a similar result to **FIGURE 4.13**.

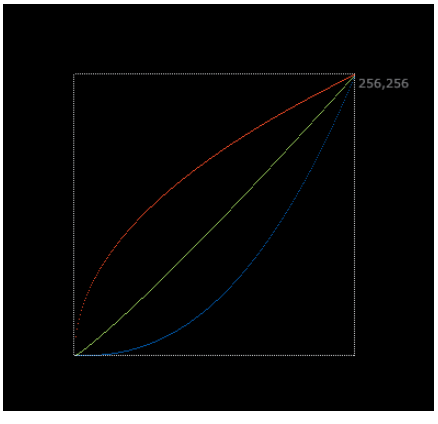


FIGURE 4.13 Notice that only the middle part of the graph moves.

The Gamma operation changes the midtones without changing the blacks or whites. You can tell that the point at the furthest left and at the furthest right are not moving.

Contrast is next.

12. Replace Gamma2 with a RolloffContrast node in the Color toolbox.

13. Bring RolloffContrast2's contrast value to 1.5.

The contrast operation pushes the two parts of the dynamic range away from one another (**FIGURE 4.14**).

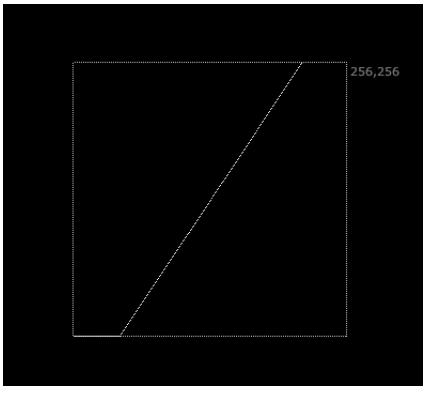


FIGURE 4.14 A basic contrast curve.

14. Play around with RolloffContrast2's Center property. When you are finished, set the value to 0.

    Here you can see what actually happens when you play with the Center slider. It moves the point that defines where the lowlights and highlights are. When leaving the center at 1, you can see that the curve is identical to a Multiply curve (**FIGURE 4.15**).

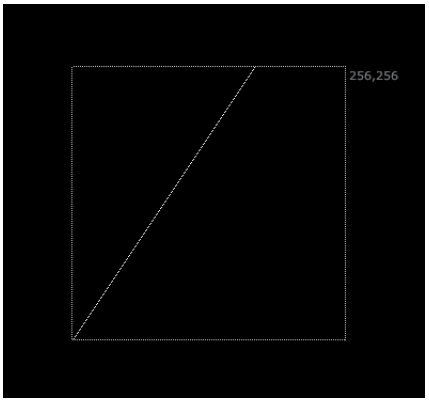


FIGURE 4.15 A Center value of 0 makes Contrast behave like Multiply.

15. Move the Center slider up to 1 (**FIGURE 4.16**).

    This is a Lift operation, which is covered later in this chapter. Your white point is locked, while everything else changes—the opposite of Multiply.

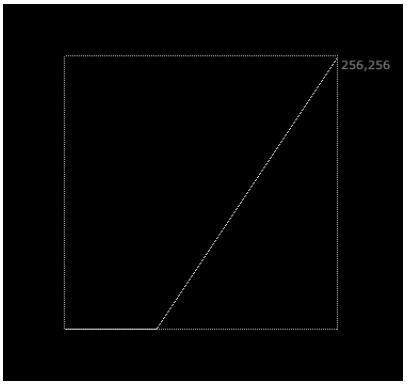


FIGURE 4.16 Moving the slider up to 1 is actually a Lift operation.

RolloffContrast has one other property you can see in the I/O graph. This property, called Soft Clip, is the property that gives this node its name. This property smooths out the edges of the curve so that colors don't all of a sudden turn to black or white and result in a harsh transition.

16. Move the center slider to 0.5 and start to increase the Soft Clip slider. I stopped at 0.55.

    FIGURE 4.17 shows what happens when you increase the soft clip. This creates a much more appealing result, which is unique to this node.

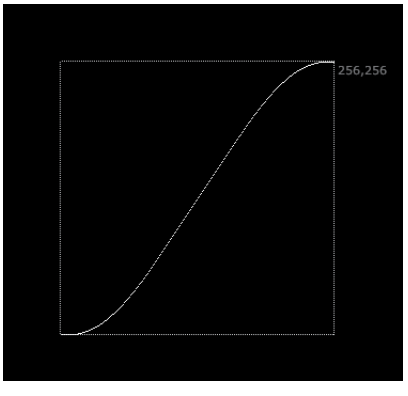


FIGURE 4.17 This smooth edge to the curve is what gives RolloffContrast its name.

If you have a fair amount of experience, you must have noticed that the I/O graph looks a lot like a tool you may have used in the past—something applications such as Adobe After Effects call Curves. In Nuke, this is called ColorLookup, and it is discussed in the next section.

CREATING CURVES WITH COLORLOOKUP

The ColorLookup node mentioned at the beginning of this lesson is actually an I/O graph you can control directly. This makes it the operation with the most amount of control. However, it's actually the hardest to control and keyframe due to its more complicated user interface. After all, it's easier to set a slider and keyframe it than move points on a graph.

Let's try this node on both the image and the I/O graph itself.

1. Replace RolloffContrast2 with a ColorLookup node in the Color toolbox (**FIGURE 4.18**).

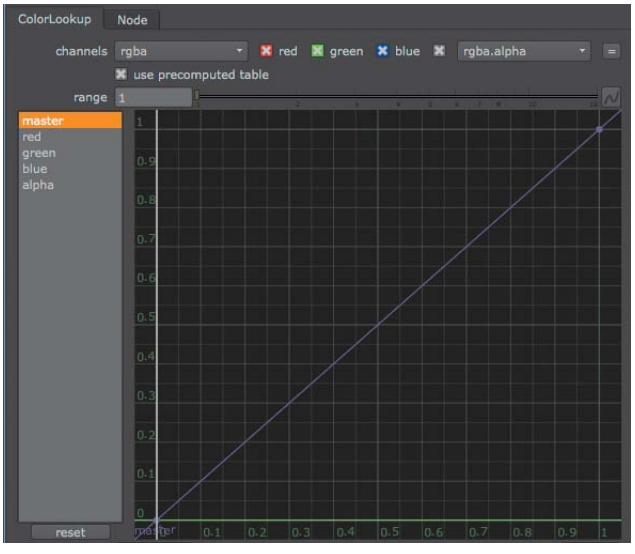


FIGURE 4.18
The ColorLookup interface.

   The interface for this node has the narrow curves list on the left, and the curve area on the right. Choosing a curve at left displays that curve at right, which enables you to manipulate it. There are five curves. The first controls all the channels, and the next four control the R, G, B, and alpha separately. You can have more than one curve display in the graph window on the right by Shift-clicking or Ctrl/Cmd-clicking them in the list.

2. Click the Master curve in the list at left.

   In the graph (Figure 4.18), you can now see a curve (a linear one at the moment). It has two points that define it, one at the bottom left and one at the top right. Moving them will change the color. For example, moving the top one will create a Multiply operation.

The ColorLookup's strength lies in making curves that you can't create using regular math functions. For that you need to create more points.

3. To create more points on the curve, Ctrl/Cmd-Alt/Option-click the curve in the graph window. It doesn't matter where on the curve you click.

   You've just created another point. You can move it around and play with its handles. If you look at the I/O graph on the Viewer, you can see that it mimics what you did in the ColorLookup node. They are exactly the same (**FIGURE 4.19**).

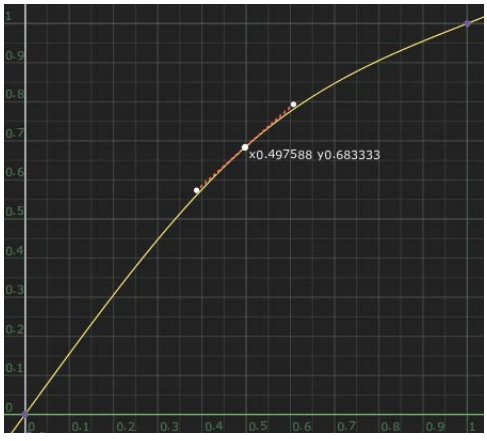


FIGURE 4.19 Changing the curve is just like working with an I/O graph.

Now let's use ColorLookup on the car image.

4. Select Read1 and Shift-click the ColorLookup node in the Color toolbox to branch another output.
5. Click ColorLookup2 and press the 1 key to view it in the Viewer.
6. Play around with ColorLookup2's curves. You can play with the separate RGB curves as well.

I ended up with **FIGURE 4.20**—pretty drastic. But that's the level of control you have with ColorLookup. The Reset button at bottom left allows me to reset this mess.

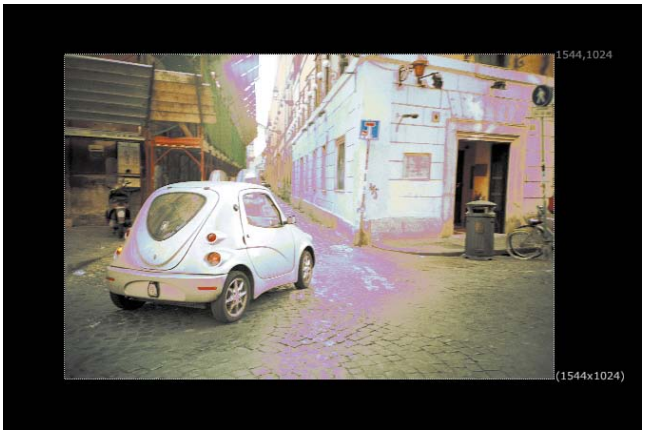


FIGURE 4.20 Extreme color correction courtesy of ColorLookup.

COLOR MATCHING WITH THE GRADE NODE

The Grade node is specifically built to make some color correction operations easier. One of these operations is matching colors from one image to another.

When matching colors, the normal operation is to match black and white points between the foreground and background (only changing the foreground), and then match the level of the midtones gray, and finally match the midtone hue and saturation.

Using the Grade node

The Grade node is made out of a few of the building blocks mentioned earlier. **TABLE 4.2** shows a list of its seven properties.

TABLE 4.2 Grade Node Properties

Property	Definition
Blackpoint	This is the reverse operation to Lift. It works in the same way, but higher numbers will result in stronger blacks instead of lighter blacks. Basically, the color chosen here will turn to black.
Whitepoint	This is the reverse operation to Multiply. It works in the same way, but higher numbers will result in lower highlights instead of stronger highlights. Basically, the color chosen here will turn to white.
Lift	A Lift operation.
Gain	A Multiply operation.
Multiply	Another Multiply operation.
Offset	An Add operation.
Gamma	A Gamma operation.

By using Blackpoint and Whitepoint to set a perfect black and a perfect white, you can stretch the image to a full dynamic range. When you have a full dynamic range you then can easily set the blackpoint and whitepoint to match those of the background using Lift and Gain. You then have Multiply, Offset, and Gamma to match midtones and for final tweaking.

Let's practice color matching, starting with a fresh script.

1. If you want, you can save your script. When you are finished, press Ctrl/Cmd-W to close the script and leave Nuke open with an empty script.
2. From your chapter04 folder bring in two images: CarAlpha.png and IcyRoad.png.
3. Make sure that CarAlpha.png is called Read1 and IcyRoad.png is Read2. You can change the name of a node in the top-most property.

NOTE If Nuke quits altogether, just start Nuke again.

You will quickly composite these images together and then take your time in color matching the foreground image to the background.

4. Select Read1 and press the M key to insert a Merge node after it.

5. Connect Merge1's B input to Read2 and view Merge1 in the Viewer (**FIGURE 4.21**).

FIGURE 4.21 The car is over the dashboard—this is wrong.

The composite is almost ready. You just need to punch a hole in the foreground car so it appears to be behind the snow that's piling on the windshield. For that, you'll bring another image in (you will learn how to creates mattes yourself in Chapter 6).

6. From your chapter04 folder bring in Windshield.png and display it in the Viewer.

Here you can see this is a matte of the snow. It is a four-channel image with the same image in the R, G, B, and alpha. You need to use this image to punch a hole in your foreground branch. To do that you will need another Merge node.

7. Select Read3 and insert a Merge node after it.

8. Drag Merge2 on the pipe between Read1 and Merge1 until the pipe highlights. When it does, release the mouse button to insert Merge2 on that pipe (**FIGURE 4.22**).

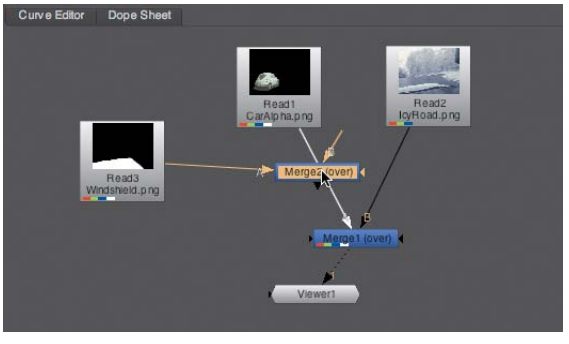


FIGURE 4.22 Inserting a node on an existing pipe.

9. View Merge1 (**FIGURE 4.23**).

FIGURE 4.23 All that white on the dashboard shouldn't to be there.

You can see here that this is not the desired result (**FIGURE 4.23**). You still need to change the Merge2 operation to something that will cut the B image with the A image. This operation is called Stencil. Stencil is the reverse operation from Mask, which you used in Chapter 3. Mask held image B inside the alpha channel of image A, and Stencil will hold image B outside image A.

10. In Merge2's Properties panel, choose Stencil from the Operation drop-down menu.

FIGURE 4.24 The car is now correctly located behind the dashboard.

Looking at your comp now, you can see that it works—short of a color difference between the foreground and background (**FIGURE 4.24**). Let's use a Grade node to fix this shift.

11. Select Read1 and press the G key to insert a Grade node after it.

As you know from Chapter 2, you are not allowed to color correct premultiplied images. It is often hard to tell if an image is premultiplied or not, but in this case it is. You can also look at the RGB versus the alpha channels and see that the areas that are black in the alpha are also black in the RGB.

Since you can't color correct premultiplied images you have to unpremult them. You can do this in one of two ways: using an Unpremult node before the color correction (in this case, Grade1) and then a Premult node after it, or using the (Un)premult By Switch in your Color nodes. Let's practice both.

12. Bring Grade1's Offset property up to around 0.4.

You can see that the whole image, except the dashboard area, turned brighter, even though you are only correcting the car image (**FIGURE 4.25**). This is due to the lack of proper premultiplication. Let's do the two-node method first.

FIGURE 4.25 The whole image turned brighter.

13. Click Read1 and from the Merge toolbox add an Unpremult node.

14. Click Grade1 and from the Merge toolbox add a Premult node and look at the Viewer (**FIGURE 4.26**).

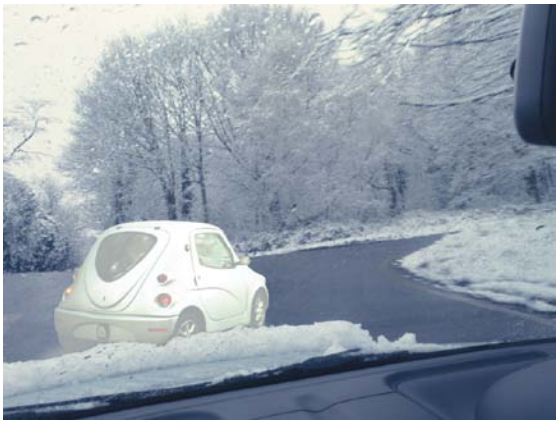


FIGURE 4.26 The proper premultiplication fixed the problem.

The problem has been fixed. This is one way to use proper premultiplication. Let's look at another.

15. Select Unpremult1 and Premult1 and press the Delete key.

16. In Grade1's Properties panel, choose rgba.alpha from the (Un)premult By menu; this automatically selects the associated check box (**FIGURE 4.27**).

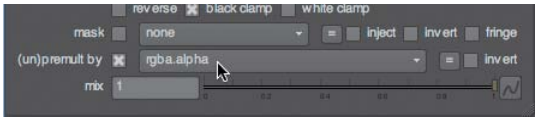


FIGURE 4.27 Using the (Un)premult By property does the same thing as the Unpremult and Premult nodes workflow.

The resulting image looks exactly as before (Figure 4.26). This technique does exactly the same thing as the first method, just without using other nodes. I usually prefer the first method as it shows clearly in the DAG that the premultiplication issues are handled. However, if you look at Grade1 in the DAG now, you will see that, although a smaller indication, Grade1 is showing that it is dividing the RGB channels with the alpha channel. The label now says "rgb/alpha" (**FIGURE 4.28**).

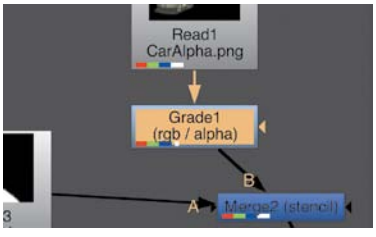


FIGURE 4.28 The node's label changes to show the Unpremult and Premult operations are happening inside the node.

Let's use the second method you have set up already. You will now be color correcting an unpremultiplied image, but outputting a premultiplied image. After a little rearranging, the tree should look like that in **FIGURE 4.29**.

17. Bring the Offset property back to 0.

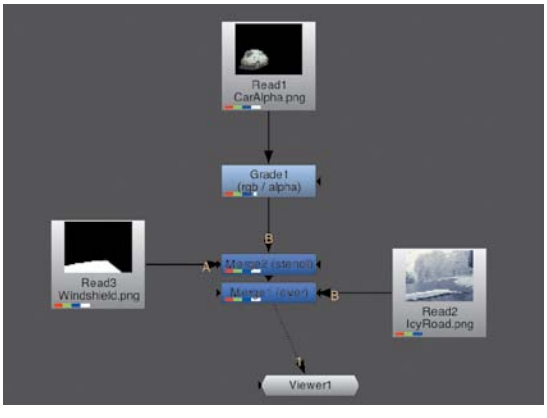


FIGURE 4.29 Your tree should look like this at this point.

Using CurveTool to match black and white points

Thinking back to the introduction of this section, how are you going to find the darkest and lightest points in these two images to match them together? One way, which is valid and happens often, is using your eyes to gauge which are the darkest and brightest pixels. However, the computer is so much better at these kinds of things, and doesn't have to contend with light reflections on the screen, etc.

The Node to use for this is the CurveTool node, which you used in Chapter 3 to find the edges of the lemming element. You can also use this node to find other color-related stuff about your image. Let's bring a CurveTool node in to gauge the darkest and brightest point in the foreground and use that data to stretch the foreground image to a full dynamic range.

1. Select Read1 and branch out by Shift-clicking a CurveTool node in the Image toolbox.

This time you are going to use the Max Luma Pixel Curve Type. This finds the brightest and darkest pixels in the image.

2. In CurveTool1's Properties panel, switch the Curve Type drop-down menu to Max Luma Pixel.
3. Click the Go! button.
4. In the dialog box that opens, click OK as you only want to process one frame.
5. Switch to the MaxLumaData tab and view CurveTool1 in the Viewer (**FIGURE 4.30**).

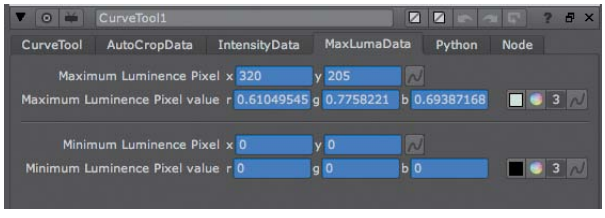


FIGURE 4.30 The MaxLumaData tab's two sections.

The purpose of this operation is to find the darkest and lightest pixels in the image. When switching to this tab you see two sections, the one showing the lightest pixel (Maximum) and the darkest pixel (Minimum). For each, the X and Y location and RGB values display.

NOTE A Constant node creates a solid color with a chosen resolution.

Looking closely you can see that the value of the minimum pixel is 0 in every property. This is because this image is a premultiplied image, and as far as CurveTool is concerned, all that black in the image is as much a part of the image as any other part of it. You need to find a way to disregard that black area. Let's do the following.

6. From the Image toolbox, create a Constant node.

7. Change Constant1's Color value to 0.5.
8. Select Read1 and branch a Merge node from it by pressing Shift-M.
9. Connect Merge3's B input to Constant1, and then view Merge3 in the Viewer (**FIGURE 4.31**).

FIGURE 4.31 The car is on a gray background.

What you did here was replace, momentarily, the black background with a middle gray background. This way you are getting rid of the black and replacing it with a color that is not the darkest nor the lightest in the image. This new image is the image you want to gauge using the CurveTool. You'll need to move the pipe coming in to CurveTool1 (**FIGURE 4.32**).

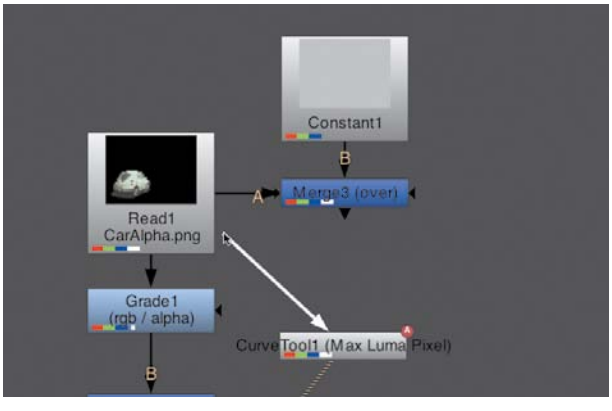


FIGURE 4.32 Moving the pipe from Read1's output to Merge3's output.

10. Click the top half of the pipe going into CurveTool1, which will enable you to move it to the output of Merge3.

11. Double-click CurveTool1 to display its Properties panel in the Properties Bin. Switch to the CurveTool tab (the first one), click Go! again, and click OK.

12. Switch to the MaxLumaData tab again and have a look (**FIGURE 4.33**).

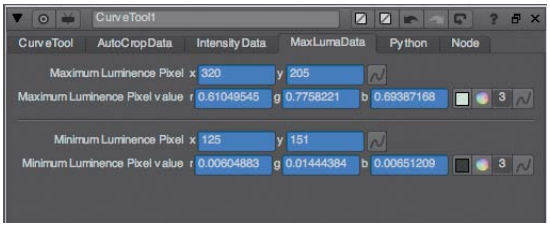


FIGURE 4.33 The updated CurveTool1's MaxLumaData tab.

You can see now that the minimum values are far from being all 0. You are now getting a true result showing the lightest and darkest pixels. Let's make use of them.

13. Close all Properties panels in the Properties Bin to clear some room.

14. Double-click CurveTool1, and then double-click Grade1.

15. View Merge1 in the Viewer.

16. Click the 4 icon next to Grade1's Blackpoint, Whitepoint, Lift, and Gain to enable the four fields.

17. Ctrl/Cmd-drag from CurveTool1's Minimum Luminance Pixel value's Animation menu to Grade1's Blackpoint Animation menu and release the mouse button to create an expression link between them.

18. Do the same from Maximum Luminance Pixel value to Whitepoint (**FIGURE 4.34**).

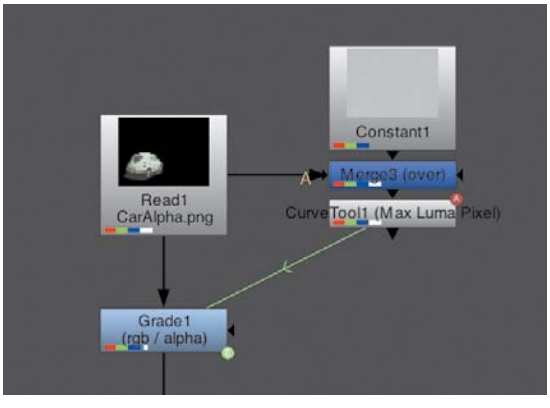


FIGURE 4.34 The green arrow shows the expression link between the two nodes.

The foreground image's dynamic range now spans from a perfect black to a perfect white. This enables you to push those colors to new black and white points to match these points to the background image. You'll need to use another CurveTool to find those points in the background image.

19. Click Read2 and by Shift-clicking, branch out another CurveTool from it.

This time there is no alpha and no black background to worry about. You can simply proceed to finding the black and white points.

20. In CurveTool2's Properties panel, choose Max Luma Pixel from the Curve Type drop-down menu.

21. Click Go! When asked, click OK.

22. When the processing is finished (you should see a quick flash of the Progress Bar) switch to the MaxLumaData tab.

You now have two sets of data to match to: new black points and white points. Let's link them to your Grade node.

23. Close all Properties panels in the Properties Bin to clear some room.

24. Double-click CurveTool2, then double-click Grade1.

25. Ctrl/Cmd-drag from CurveTool2's Minimum Luminance Pixel value's Animation menu to Grade1's Lift Animation menu to create an expression link between them.

26. Do the same from the Maximum Luminance Pixel value to Gain (**FIGURE 4.35**).

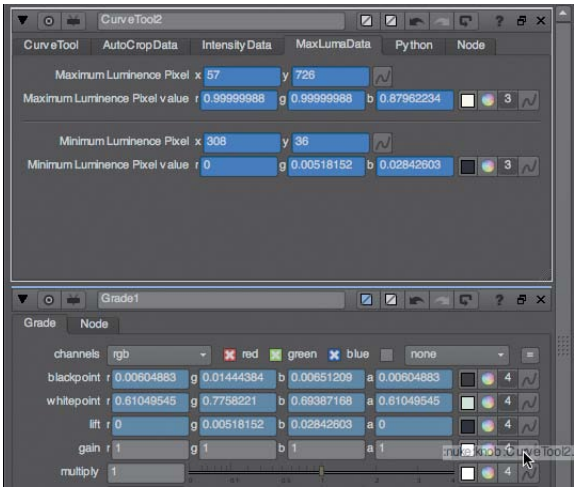


FIGURE 4.35 Dragging while holding Ctrl/Cmd creates a linking expression.

You have now matched the foreground's shadows and highlights to those of the background (**FIGURE 4.36**).

FIGURE 4.36 Shadows and highlights now match.

As you can see from the image, the shadows and highlights are matched, but the image is far from looking matched. The midtones, in this case, make a lot of difference.

Matching midtones by eye

You now need to match the midtones. This is a much more difficult task. You'll start by matching its luma level by eye. Because it is hard to tell what the midtones are, though, you are going to view the luminance of the image in the Viewer.

1. Hover your mouse pointer in the Viewer and press the Y key to view the luminance.

To change the midtones now, you will use the Gamma property. You can see that the whitish snow on the right is a darker gray than the whitish car. Let's bring down the whitish car to that level.

2. Start dragging the Gamma slider down. I stopped at around 0.6.

Notice that the midtones don't match well with a higher Gamma value. Now, however, the lower midtones aren't matching well. I need to use the Multiply property to produce a good match.

3. Bring the Gamma slider up to 0.85 and bring the Multiply slider down a bit to 0.9 (**FIGURE 4.37**).
4. Hover your mouse pointer in the Viewer and press the Y key to view the RGB channels (**FIGURE 4.38**).

FIGURE 4.37 The midtones match better now.

FIGURE 4.38 There is still work to be done on the color of the midtones.

OK, so the midtones' brightness is now better, but you need to change the color of the car's midtones. At the moment, the car is too warm for this winter's day. Matching color is a lot more difficult as you always have three options: red, green, and blue. Matching gray is a lot easier as you only need to decide whether to brighten or darken it. However, as each color image is made out of three gray channels, you can do that to match color too. Here's how.

5. Hover your mouse pointer in the Viewer and press the R key to view the red channel (**FIGURE 4.39**).

FIGURE 4.39 Viewing the red channel.

Now you are looking only at levels of gray. If you now change the red sliders, you will better match the color while still looking only at gray.

6. Display the Color Wheel and Color Sliders panel for the Gamma property by clicking the Color Wheel button.

You will also want to change the Multiply and Offset values to achieve a perfect result. This is because, even though you matched the black point and white point, the distance of the car from the camera means the black point will be higher and the white point lower. At the end of the day, it will only look right when it does, math aside.

Let's display those extra color wheels.

7. Ctrl/Cmd-click the Color Wheel button for the Multiply and Offset properties. Your screen should look like **FIGURE 4.40**.

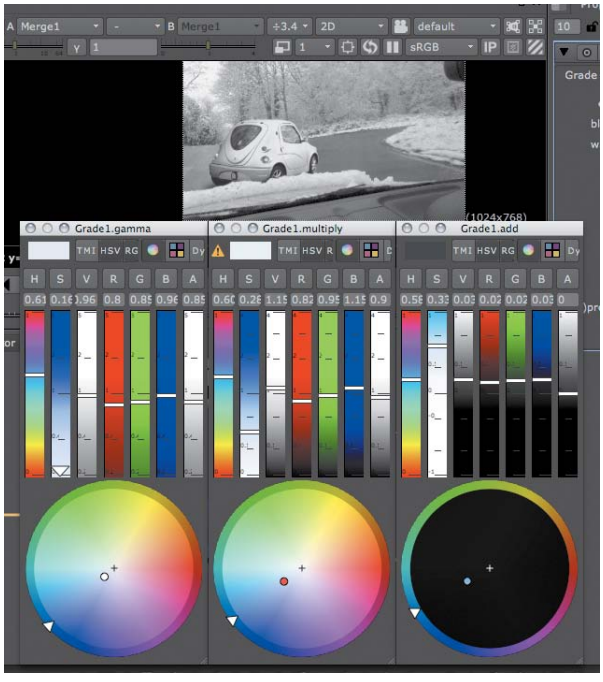


FIGURE 4.40 Opening three color wheels to easily control three properties.

8. Since you are looking at the red channel in the Viewer, you should change the red sliders for Gamma, Multiply, and Offset until you are happy with the result. Little changes go a long way. I left mine at Gamma: 0.8, Multiply: 0.82, and Offset: 0.02.

9. Display the green channel in the Viewer, and then move the green sliders to change the level of green in your image. Mine is Gamma: 0.85, Multiply: 0.95, and Offset: 0.025.

10. Do the same for the blue channel. Mine is Gamma: 0.96, Multiply: 0.95, and Offset: 0.03.

11. Switch back to viewing the RGB channels (**FIGURE 4.41**).

FIGURE 4.41 Not a bad result at the end of it all.

This is as far as I will take this comp. Of course, you can use your already somewhat-developed skills to make this a better comp, but I'll leave that to you.

Save your script if you wish, and we will move on.

ACHIEVING A "LOOK" WITH THE COLORCORRECT NODE

Giving an image a "look" is a very different practice than matching color. While matching color has a very specific purpose and methodology, giving an image a look refers to an artistic practice that gives an image a different feel to how it was shot. For example, you might want it to look brighter, warmer, or colder, depending on the feeling you want to create.

Using the ColorCorrect node

The ColorCorrect node is a very good tool to use for this as it has a lot of control over the different parts of the image—even more control than the Grade node. But as with everything else, it is still made out of the basic mathematical building blocks covered in the beginning of this chapter.

Let's bring in an image and give it a look.

1. Press Ctrl/Cmd-W to close the color matching script and start a new one.
2. Press the R key and bring in, from the chapter04 folder, the Car.png image again.
3. While the newly imported Read1 node is selected, press the C key to create a ColorCorrect node. You can also find the ColorCorrect node in the Color toolbox.
4. View ColorCorrect1 in the Viewer (**FIGURE 4.42**).

NOTE If Nuke quits altogether, just start Nuke again.

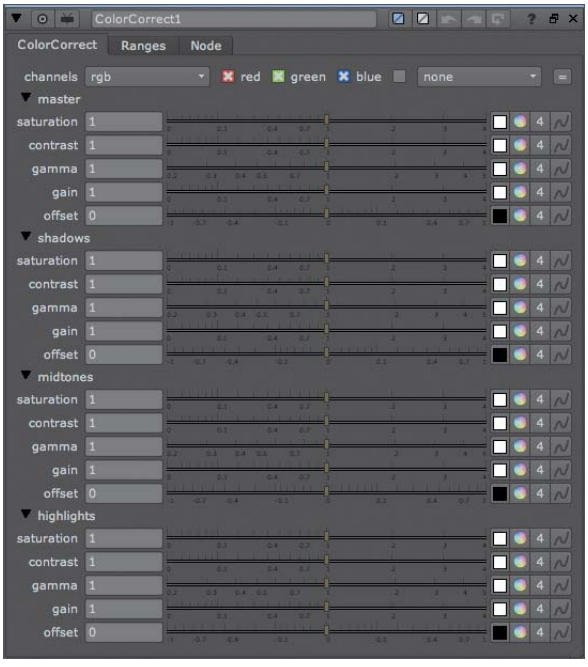


FIGURE 4.42 The ColorCorrect node's Properties panel.

As you can see in ColorCorrect1's Properties panel, the ColorCorrect node includes controls for Saturation, Contrast, Gamma, Gain (Multiply), and Offset (Add). This is performed over either the whole dynamic range—called Master—or parts of the dynamic range called Shadows, Midtones, and Highlights. This makes creating a look somewhat easier.

This idea of midtones, highlights, and shadows changes from image to image. An image of a dark room will have no whites, but in that darkness one can still define that the brighter areas will be that image's highlights, the slightly lighter blacks will be midtones, and the darkest colors shadows. This can also be defined in the ColorCorrect node's Ranges tab.

5. Click the Ranges tab in ColorCorrect1's Properties panel.

   In this tab (similar to ColorLookup, isn't it?) you have three graphs, all selected. One represents the shadows, another the midtones, and a third the highlights (**FIGURE 4.43**).

6. Check the Test check box at the top of the graph.

   This shows a representation in the Viewer of what parts of the image are shadow, midtone, and highlight. Highlights are represented by white, midtones as gray, and shadows as black. Green and magenta areas represent a mix of two ranges (**FIGURE 4.44**).

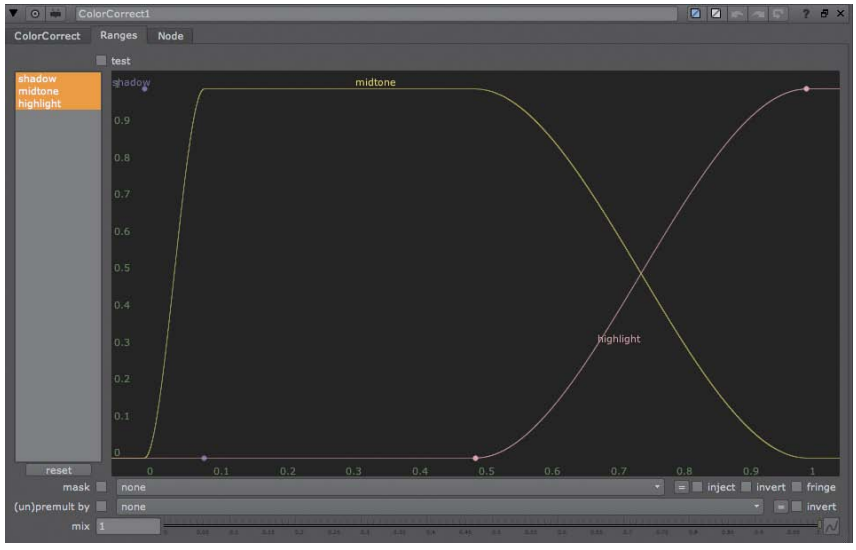


FIGURE 4.43 ColorCorrect's Ranges is a lookup curve that defines the brightness ranges.

FIGURE 4.44 The test shows the parts of the dynamic range in the Viewer.

7. Click the Test button at the top of the graph again to turn it off.

   The ranges are fine for this image, so we won't change anything and we will continue working.

8. Switch back to the ColorCorrect tab.

You will now give this image a dreamy, car-commercial look—all soft pseudo blues and bright highlights. If you don't define the "look" you are after in the beginning, you can lose yourself very quickly.

Before changing the color of this image, I'll show you my preferred interface setup for color correcting.

9. In ColorCorrect1's Properties panel, click the Float controls button (looks like two boxes). This will float the Properties panel instead of docking it in the Properties Bin (**FIGURE 4.45**).

FIGURE 4.45 Click this button to float the Properties panel.

10. Hover your mouse pointer in the Viewer and press the Spacebar to maximize the Viewer to the size of the whole interface (**FIGURE 4.46**).

FIGURE 4.46 This is a good way to set the interface for color correction.

Since the Properties panel is floating, it is still there. This way, you can look at the image at its maximum size without wasting space on things like the DAG yet still be able to manipulate the ColorCorrect node.

What I am aiming for is something like that in **FIGURE 4.47**. You can try to reach that yourself, or you can follow my steps point by point.

11. I'll start by desaturating the whole image a little, so in the Master set of properties I set the Saturation property to 0.5.

12. Now for the shadows. I would like to color the shadows a little bluer than normal. Click the Color Wheel button for shadows.gamma.

FIGURE 4.47 This is the image look I am referring to.

13. From the Hue slider, choose a blue hue. I selected 0.6. Now display Saturation for the shadows.gamma color. I set it to 0.31. Finally, adjust the brightness, or Value, slider in the Color Wheel and Color Sliders panel. I have 1.22 (**FIGURE 4.48**).

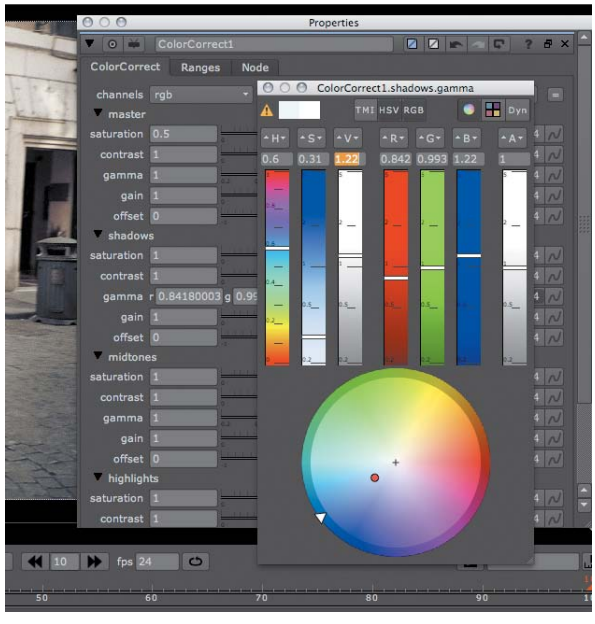


FIGURE 4.48 Setting the shadow's Gamma properties using the Color Wheel and Color Sliders panel.

This results in RGB values of 0.8418, 0.993, and 1.22, respectively. It gives the image a nice-looking blue shadow tint. Notice that there are actually no hue and saturation sliders in the real Properties. The hue and saturation sliders in the Color Wheel and Color Sliders panel are only there so it will be easier to set the RGB sliders.

14. Close this Color Wheel and Color Sliders panel.

15. You have a lot more work in the midtones. First, set the Saturation to 0, so that the midtones are tinted black and white.

16. To create a flatter palette to work on, set the Contrast for midtones at 0.9.

17. To darken the midtones, set the Gamma to 0.69.

18. Use the Gain property to tint the midtones by clicking the Color Wheel button for midtones.gain.

NOTE If you need to make the Color Wheel and Color Sliders panel bigger, drag the bottom-right corner of the panel.

19. In the Color Wheel and Color Sliders panel that opens, click the TMI button at the top to enable the TMI sliders (**FIGURE 4.49**).

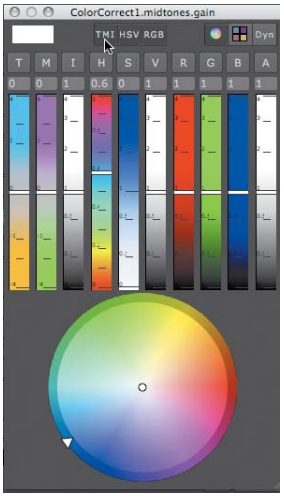


FIGURE 4.49 Turning on the TMI sliders.

20. Now, for a cooler-looking shot, drag the T (temperature) slider up towards the blues. I stopped at 0.72.

21. To correct the hue of the blue, use the M (magenta) slider to make this blue either have more magenta or more green in it. I went towards the green and left it at –0.11.

NOTE As always, only the RGB values affect the image. You just used TMI sliders to set the RGB values in an easier way.

22. Close the Color Wheel and Color Sliders panel (**FIGURE 4.50**).

FIGURE 4.50 The values are always in RGB.

23. You will now increase the highlights a little, so let's start by setting the Contrast to 1.5.

24. To color correct the highlights, first click the 4 icon to enable the Gain slider.

25. Click in the right side of Gain's first field (for the red channel) and use the arrow up and down keys to change the red value. I left it on 0.75 (**FIGURE 4.51**).

FIGURE 4.51 The arrow keys make it easy to nudge values in fields.

26. Leave the next field (green) where it is, but use the arrow keys in the blue field to increase blue. Because I want everything to be a little bluer, I left mine at 1.5.

The first stage of the color correction is finished. Let's bring the rest of the interface back.

27. Close the ColorCorrect1 Properties panel (**FIGURE 4.52**).

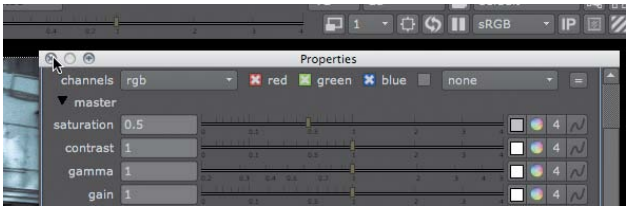


FIGURE 4.52 First, close the floating Properties panel.

28. Press the Spacebar to bring back all your panes.

Using the mask input to color correct a portion of the image

Let's say that a movie director asks for the wheels to pop out of the image and have high contrast. To do this secondary color correction, you will need to first define an area to apply the color correction to, then use another Color node and use this area in its mask input.

You haven't learned to create complex mattes yet, but in this case you only really need two radial mattes. You can easily create those using the Radial node in the Draw toolbox.

First, to brighten up the wheels, you will use the Grade node.

1. Select ColorCorrect1 and insert a Grade node after it.

   If you use the Grade node as it is, the whole image will get brighter. You'll need to use Grade1's mask input to define the area to work in.

2. With nothing selected, create a Radial node from the Draw toolbox (**FIGURE 4.53**).

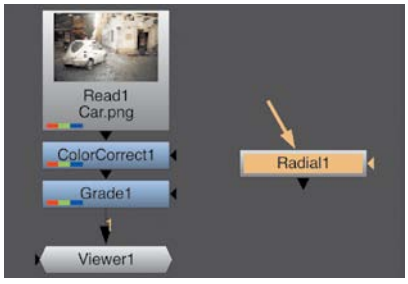


FIGURE 4.53 Creating an unattached Radial node.

3. View Radial1.

   It creates a radial, see? I told you. By moving the edges of the radial box, you can change its shape and location.

4. View Grade1.

5. Drag Radial1's edges until it encompasses the back wheel (**FIGURE 4.54**).

FIGURE 4.54 Radial1 encompasses the back wheel.

You'll need another Radial node to define the second wheel. (You can add as many Radial nodes as you need. Everything in Nuke is a node, remember?)

6. With Radial1 selected, insert another Radial node after it.

7. Adjust Radial2 to encompass the front wheel (**FIGURE 4.55**).

8. To make use of the radials, you will take the mask input for Grade1 and attach it to the output of Radial2, as in **FIGURE 4.56**.

FIGURE 4.55 Using Radial nodes to create masks for color correction

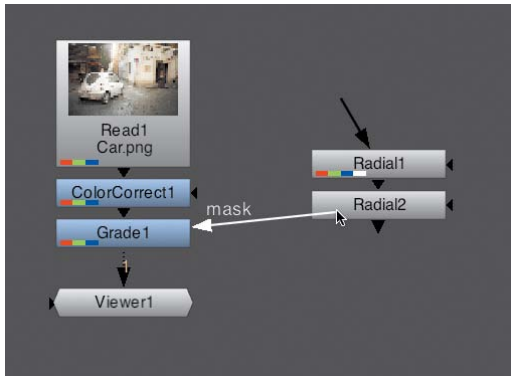


FIGURE 4.56 Attaching the mask input to the mask image.

This means whatever you now do in Grade1 will affect only where the radial's branch is white.

9. Increase the whites by bringing the Whitepoint property for Grade1 down to around 0.51.

10. Some of the deep blacks have become a little too gray, so decrease a little on the Blackpoint property. I left mine at 0.022.

At this point, the grading is finished. Mask inputs can be very important in color correction because a lot of times you only want to color correct an area of the image. But remember not to confuse mask inputs with mattes or alpha channels. The use of the mask input is solely to limit an effect—not to composite one image over another or copy an alpha channel across.

This page intentionally left blank

INDEX

Numbers

2D compositing, inside 3D scenes, 326–333
2D data, transforming 3D data into, 268–277
2D movements, 143
2D tracking, 136. *See also* tracking
2D view, switching to, 270
2K, working in, 238
3D applications, scale and rotation in, 265
3D data, transforming into 2D data, 268–277
3D render, bringing in, 63
3D scenes. *See also* Photoshop layers
  2D compositing inside, 326–333
  camera element, 258
  changing View Selection, 260
  compositing screen into, 334–335
  creating cubes, 265–268
  geometry element, 258
  Grid node, 330–332
  importing cameras, 263–265
  ScanlineRender node, 258, 260
  Scene node, 258
  setting up, 259–260
3D setup, 258–260
3D Studio Max, using for camera projection, 309
3D toolbox, described, 8
3D tracking, applying, 283
3D view, accessing, 260
3D world
  camera in, 261
  displaying camera axis, 262
  navigating, 261–263
  rotation controls, 262–263
8-bit color, explained, 98
32-bit color, explained, 98
2048×2048 image, defining, 320

Symbols

symbol, using with numbers, 43–44
. (period), using to create Dots, 76, 78, 87, 312–313

A

Action Safe area, attaching text to, 349
Action Safe Area Knob, 357
ActionSafe property, using with User Knobs, 344–345. *See also* safety areas
Add color-correction function, 100, 105
Add node, replacing with Multiply node, 107–108
Adrenalin Lemming, 63
All Plugins submenu, using with Gizmos, 363–364
alpha channel, viewing, 25
Ambient Occlusion pass
  combining with comp, 92
  described, 65
anaglyph glasses, using with stereo trees, 254–255
animation
  combining with paint and roto, 185–190
  identifying, 69
  using keyframes in, 52–54
Animation menu, using, 41
Animator curve, example of, 54–55
ASCII file, explained, 29
assigning variables, 369
AutoCropData tab, 68
autosaves, setting preferences for, 29
axis, repositioning, 58–59
Axis node
  connecting to Camera node, 265
  using in camera projection, 316–317

B

Backdrop node, 106
background, placing foreground on, 71–72
background input, connecting, 24–25
background.####.png file, using with process tree, 22–23
beauty pass
  building, 73
  described, 65
  layering passes, 73–76
  premultiplied, 85
  splitting process tree, 76–83
bg.####.png file, using in 3D scene, 259
bg##.png file, opening, 178
bg.png file, using with keying nodes, 200
black and white points, matching, 118–122
blackpoint property, 113
Blending Mode property, using in RotoPaint, 190
Blur, inserting for Reconcile3D nodes, 277
bounding box
  tightening, 70–71
  using, 66–68
branches
  making for passes, 76–83
  starting from nodes, 31
brightness values, 100
Buddha image, placing in frame, 148
bullet_%V.####.png file, naming, 253
bulletBG_left.####.dpx file, opening, 230
bulletBG_right.####.dpx file, opening, 230
bulletCG_left.####.ext file, 243
bulletCG_right.####.ext file, 243
buttons. *See* Python buttons

C

cache file, explained, 15
camera axis, displaying in 3D world, 262
camera element, in 3D scenes, 258
Camera node
  connecting Axis node to, 265
  using in camera projection, 312–314
camera projection
  3D Studio Max, 309
  720p format, 306–307
  animating camera, 316–317
  applying texture to buildings, 311–312
  Axis node, 316–317
  Axis1 connected to ShotCamera, 316–317
  Camera node, 312–314
  camera selection, 308–309
  changing scale, 310
  changing text, 308
  Dots, 312–313
  explained, 310
  geometry in 3D Viewer, 308
  importing geometry, 307
  importing image, 306
  manipulating Translate.x field, 310
  mirror ball map, 320–321
  near clipping plain, 314
  Near property, 314
  painting tips, 319–320
  project setup, 306
  Project3D node, 310–311
  proxy mode for rendering, 317
  relocating camera, 309–310
  rendering speed, 317
  replacing sky, 320–322
  RotoPaint node, 318–320
  ScanlineRender node, 312–314
  Scene node, 312–314
  shade geometry, 310–311
  square_2k format, 320
  texture setup for geometry, 310–312

tracking shot, 318–320
tweaking geometry, 315–316
tweaking texture, 318–320
Write node for rendering, 336
Camera Tracker. *See also* reflection; tracking
2D and 3D scenes in Viewers, 289
3D view, 288
aligning scenes, 287–292
CameraTracker tab, 283
connecting Viewer, 289
Create Scene, 281
creating scenes, 286–287
creating Viewer in pane, 289
defining ground plane on axes, 290
defining points, 290
Film Back Size property, 285
improving tracks, 282
Mask input, 283
nodes, 286–287
point cloud in 3D Viewer, 288
points representing table, 292
removing extra Viewer, 292
running tracking process, 283
selecting points for Z axis, 291
selecting points on table, 290
setting Z axis for ground plane, 291
Solve Camera, 281, 285
solved tracks, 285
Solver tab, 284–285
splitting pane horizontally, 288
Track Features, 281, 283
tracking features, 282–284
Tracking tab, 282
using, 281
Viewers side by side, 290
camera tracking, calculating reflection movement, 280–281
cameras
importing in 3D scenes, 263–265
viewing in 3D world, 261
nodal, 277
CameraTracker node, 280, 287
candle flames, adding flares to, 140–142
car commercial look, creating, 127–128
Card node, using with reflective surface, 295–298
CarWindow.png file, opening, 185
CGI images
adding, 87–91
placing over live backgrounds, 91–93
channel display box, 13
channel sets
combining, 74
JPEG file format, 62
listing, 64
PNG file format, 62
PSD file format, 62
pulling information from, 91
RGBA, 62
TIFF file format, 62
viewing with Viewer, 63–65
Channel toolbox, described, 7
channels
copying between branches, 84
maximum number of, 62
mixing from channel sets, 77
moving, 50–51
using Mask input with, 93–95
Check Box Knob
linking to expression, 362
setting up, 361
Christmas2K file, naming, 232
chroma-keying, explained, 198. *See also* HueKeyer keying node
CLEAN ME, writing on car window, 189
clipped color, explained, 99
clips
playing in Viewer, 15–16
slowing down, 246
Clone brush, using in RotoPaint, 175–176
Cmd key, 7
Col pass, adding, 81
color channels, viewing, 13–14
color correcting
highlights, 130–131
images, 36
portions of images, 131–133
unpremultiplied images, 117
color correction
dynamic range, 100–105
functions, 100
interface setup, 127–128
limitation of, 26–27
nodes, 36, 103–104
color difference-keying, explained, 198
color management, 227
color matching, using Grade node, 113–117
color operations, visualizing, 105–110
Color pass, 65
Color Picker button, locating, 86
color pickers, unchecking, 212
Color Sliders panel
closing, 86
enlarging, 130
using, 40–41
using in dynamic range, 101
color space conversion, applying, 234–235
Color toolbox, described, 7
Color Wheels
closing, 86
displaying, 124
enlarging, 130
using, 40–41
using in dynamic range, 101
ColorBars node, using with Gizmo's tree, 352
ColorCorrect node
brightness ranges, 126–127
Properties panel, 126
Ranges tab, 126–127
using, 125–131
ColorLookup tool, creating curves with, 110–112
colors
32-bit, 98
capturing from screen, 212
clipped, 99
dealing with, 105
normalized values, 99
sampling from Viewer, 86
Colorspace option, using with stereo script, 234
Colorspace property, in Proxy System, 241
Command key, 7
commands
creating in Python, 371
expressions as, 69–71
composites
building for process trees, 22–23
creating, 24–25
starting, 20
viewing without rendering, 51–52
compositing, defined, 24
compression property, changing, 44
Constant node, creating, 118–119
ContactSheet node, using, 65–66
Content menu, 3
contextual menu, 6
Contrast color-correction function, 100, 105
contrast operation, performing, 109
controls, 35
copying Gizmos, 364
CornerPin node
specifying source corners, 152–154
using in Fly Ginza example, 334
using with Tracker, 149–150, 152
Crop node, using, 69–71
Ctrl key, 7
cubes
applying checkerboard texture, 265–266
creating in 3D scenes, 265–268
Downrez option, 267
Proxy mode for playback, 267
relocating, 266–267

seeing over background, 267
tracked camera, 267
Current Info panel, using with scripts, 29–30
Curve Editor, 3. *See also* RotoPaint
vs. Dope Sheet, 192
moving around in, 173
switching between Node Graph, 3
using, 54–55, 170–174
curve interpolation, 172
curves
creating, 170
creating points on, 112, 173
fitting to window, 171
loading onto Curve Editor, 41
smoothing edges of, 110
CurveTool, using, 68, 118–122

D

DAG (Directed Acyclic Graph), 2–3, 23, 235–236
decimal point numbers, changing, 54
deleting
keyframes for tracks, 146
nodes, 34
strokes, 165
Difference keying node, 199
different-keying, explained, 198
diffuse pass, creating, 74
Display Gain slider, 276. *See also* Gain property
Display Gamma slider, 276. *See also* Gamma property
display resolution, scaling down, 238. *See also* resolution
doll image
alpha channel, 25
changing color of, 38–39
color correcting, 36
making darker, 41
placing over background image, 24–30
Dope Sheet, 3, 249
vs. Curve Editor, 192
displaying timing in, 249
using, 191–196
Dots
creating, 87
using in camera projection, 312–313
using to organize nodes, 76–79
using to organize trees, 92
using with Groups, 355
Downrez option, using with cube, 267
Downscale Resolution option, 238, 240
Draw toolbox, described, 7
Drop Shadow pass, described, 65
dust, removing with paint, 176–178
dynamic range
brightness values, 100
highlights, 100, 102
midtones, 100, 102–103
shadows, 100

E

EdgeDetect node, using with Gizmos, 339
Edit Expression panel, using with TCL, 349
Edit menu, 4
erode algorithms, using with mattes, 217–221
Erode filter, using on sky, 325–326
Export as Gizmo button, 363
Expression node, locating, 106
Expression panel, using with Tracker1 node, 151
expressions
changing for Groups, 358–359
changing for User Knobs, 345, 347
linking properties with, 69–71, 151
writing, 151–152
writing for Gizmo's tree, 341
eyes, adding to lemming, 89–91
Eyes pass, described, 65

F

Feather Falloff property, using in RotoPaint, 167–168
fg##.png file, opening, 178
File Browser
anatomy of, 9
launching for RotoPaint, 183
multiple-selecting files in, 22
using, 10–11
using with process tree, 22
viewer, 10
File menu, 4
file sequences
changing compression settings, 44
frame padding structure, 43
LZQ compressor, 44
naming, 43–46
rendering, 43–46
Filter toolbox, described, 8
filtering, occurrence of, 142
filters, applying, 86
Flare node, using, 140
flipbooks, loading in Framecycler, 16–17
float, explained, 98–99
flows, trees as, 20
Fly Ginza example
adding highlight to frame, 335
background layers, 331
compositing screen in 3D scene, 334–335
CornerPin node, 334
creating frame for screen, 332–333
matte added to buildings, 335
Merge node, 334
Ramp node, 334–335
rendering scene, 336
screen in perspective, 334
screen tree, 333
setting Alpha value, 335
X and Y Number fields for grid, 331
Fly Ginza logo, 329
Fly Ginza poster, 327
footage, importing, 9
foreground, placing over background, 71–72
format, determining for stereo script, 232
four point tracking, 143
FPS (frames per second) field, displaying in Viewer, 15
setting, 15
frame padding structure, applying to files, 43
frame range
changing in Project Settings panel, 248
controlling for Viewer, 226
frame_v01.nk file, using with camera tracking, 280–281
frame_v02.nk file, saving, 281
Framecycler playback system
defining frame ranges, 17
features of, 16
loading flipbooks, 16–17
maximizing, 18
shortcuts, 17–18
frames
advancing, 54
creating for pictures, 148–149
indicating keyframes on, 54
inventing in-between, 246
Furnace Core toolbox, described, 8

G

Gain property, using with stereo projects, 252. *See also* Display Gain slider
Gamma color-correction function, 100, 105
Gamma curve, using, 108–109
Gamma property. *See also* Display Gamma slider
knob for, 39
using with midtones, 122–125

Gamma slider, turning into numeric fields, 40
garbage matte, creating, 216, 219
geometry element
in 3D scenes, 258
for reflection, 301
ginza_moving.####.png render, naming, 336
Ginza.jpg, opening for camera projection, 306
Gizmo. *See also* TCL scripting language; User Knobs
creating User Knobs, 342–348
Export as Gizmo button, 363
NoOp node, 342
percentage values, 342–348
User Knobs panel, 342–343
Gizmo hot key, creating, 371
Gizmos. *See also* nodes
converting to Groups, 365–366
copying, 364
creating icons, 374
defined, 338
installing for Groups, 363
safety areas, 338–339
testing for Groups, 363–364
tips, 365–366
turning Groups into, 362–364
Viewer Input Process, 364–365
Gizmo's tree
building, 339–342
ColorBars node, 352
connecting Rectangle nodes, 352–353
EdgeDetect node, 339
expression, 341
outline for Action Safe guide, 340
Rectangle1, 340, 342
removing overlays, 341
testing, 352–353
X property, 341
good_track script, importing, 148. *See also* tracking
Grade node, 36
blackpoint property, 113
gain property, 113
gamma property, 113
lift property, 113
multiply property, 113
offset property, 113
properties, 113
unpremultiplying images, 250
using for color matching, 113–117
using Mask input with, 94
whitepoint property, 113
graphic user interface (GUI). *See* GUI (graphic user interface)
Grid nodes, using in 3D scene, 330–332
Group nodes, Show button in, 354
Group Output panel, 354
Group1 Node Graph, 355
Groups. *See also* trees
adding functions to, 360
changing expressions, 358–359
converting Gizmos to, 365–366
creating, 354
defined, 353
turning into Gizmos, 362–364
User Knobs in, 353, 355
using Dots with, 355
using Knobs with, 355–356
Viewer Input Process, 364–365
gs.tif file, using with keying nodes, 200
GUI (graphic user interface)
components, 2–3
Context menu, 3
contextual menu, 6
Curve Editor, 3
customizing, 4–6
DAG (Directed Acyclic Graph), 2–3
default layout, 2
Dope Sheet, 3, 249
getting back, 3
hot keys, 6–7
menu bar, 4–6
New Viewer, 4
Node Graph, 2–3
Nodes Toolbar panel, 2–3
panels, 3–4
panes, 2
Progress Bar, 4
Properties Bin, 3
Script Editor, 4
Viewer, 2

H

hair, adding to lemming, 87–89
Hairs pass, described, 65
Help menu, 4
high contrast image, producing, 104
highlights, 100
color correcting, 130–131
matching shadows with, 122
representing, 126–127
hi-res images. *See also* stereo script
using proxies with, 238–242
Viewer Downscale Resolution option, 238, 240
hot keys, 6–7
3D view, 261
adding with Python, 371
Bézier in RotoPaint, 167
creating for Gizmo, 371
Current Info panel, 29
Edit Expression panel, 349
identifying, 29
keyframe selection, 171
Merge node, 24
overlays, 341
for playing clips in Viewer, 15
rendering versions, 57
RotoPaint, 168
for saving, 6
saving scripts, 45
for Viewer, 13
HSV slider, using, 40
HueCorrect node, using to remove spills, 220–222
HueKeyer keying node. *See also* chroma-keying
changing point on graph, 203
default alpha channel, 201
editing curves, 202–203
editing graphs, 202
explained, 199
using, 201–203

I

IBK (Image Based Keyer) keying node, 199–200
Blue/Green Weight property, 207, 209
clean plate, 203–205
Darks and Lights, 205
Erode property, 206
IBKColour node, 203–204
IBKGizmo node, 203–204, 207–209
Luminance Level property, 207
Luminance Match property, 207, 209
Merge node, 208
Patch Black property, 207
plate, 203
Red Weight property, 207, 209
result of, 216
setting up IBK tree, 204
using, 203–210
Weight properties, 208
icons, creating and adding with Python, 372–375
ID pass, described, 65
image appearance. *See* process trees
image area, defining for processing, 67–68
image resolution
defining, 106
displaying, 23
Image toolbox, described, 7
images. *See also* nonlinear images; premultiplied images; shots
color correcting, 36
comparing, 58–59
finding darkest pixels in, 118–122
finding lightest pixels in, 118–122
merging, 24–30
playing in realtime, 15

premultiplied, 25–28
previewing, 10
shrinking to half resolution, 72
viewing, 13–14
importing footage, 9
in-between frames, inventing, 246
Incidence (Inc) pass
adding to composite, 81
changing color of, 85
described, 65
Indirect Light pass, described, 65
I/O graph
default state, 107
using with color operations, 105–110

J

JPEG file format, channel sets, 62

K

keyboard shortcuts. *See* hot keys
Keyer keying node, 200
Keyer toolbox, described, 8
Keyframe window, keyframing, 193
keyframes
changing timing of, 191–196
creating, 41
deleting for tracks, 146
indicating on frames, 54
selecting all, 171
setting with Write On End field, 191
using in animation, 52–54
keying nodes
combining using tree, 216–223
Difference, 199
HueKeyer, 199, 201–203
IBK (Image Based Keyer), 199–200, 203–210
Keyer, 200
Keylight, 200, 210–215
Primatte, 200
Ultimatte, 200
keying terminology, 198
Keylight keying node, 200. *See also* mattes; spills
Bg input, 211–212
choosing output modes, 214–215
Clip Black property, 214
Clip Rollback property, 214
controls, 213–214
drag action, 212
Gain controls, 213
garbage matte input, 211, 216
holdout matte input, 211
InM (inside matte) input, 211
locating controls, 213
OutM (output matte) input, 211
result of, 216
sampling colors, 212
Screen Gain property, 212–213
Screen Matte options, 214
Source input, 211
Tuning submenu, 213
using, 210–215
KeyMix node
compositing with, 187–188
using with Reconcile3D, 276
KnifeCU.###.png file, opening, 174
Knobs. *See also* properties; User Knobs
adjusting, 38–39
for check box, 361
creating for Which property, 361
Pick Knobs to Add panel, 356
as properties, 342
properties for, 356
using with Groups, 355–356

L

LayerContactSheet node, using, 65–66
layers. *See* channel sets; Photoshop layers
Layout menu, 4
lemming
adding hair and eyes to, 87–91
compositing over background, 71–72
lemming render passes, 65
lemming tree, placing over background, 91–93
lemming_v01.####.png, using in 3D scene, 259
Lgt pass branch, 78
Lift color-correction function, 100
Lift operation, 104, 110
Light pass, described, 65
linear, converting log images to, 234
linear color, explained, 99
lines, removing, 88–89
linking expression, creating, 69
log to linear, converting images to, 234
Lookup color-correction function, 100
luma-keying, explained, 198
LUT (lookup table) settings, 227–229
displaying graphs, 228–229
for editing, 228–229
for image types, 228–229
using with stereo script, 233–235
LZQ compressor, using with file sequences, 44

M

Mask input
color correcting parts of images, 131–133
using, 93–95
using with Camera Tracker, 283
mask operation, performing, 90–91
mattes. *See also* Keylight keying node
creating for unwanted areas, 216
crunchy type of, 216
dilating and eroding, 217–221
using, 47–52
Max Luma Pixel Curve Type, 118
maya.png file, using with process tree, 22–23
menu bar, 4–6
menu.py file, using with Python, 372–375
Merge nodes
connecting for beauty pass, 73
creating, 24
layering operations, 24, 48
Over layering option, 48
Stencil operation, 49
treatment of foreground images, 26–27
treatment of premultiplied images, 26–27
using in Fly Ginza example, 334
using with Ambient Occlusion pass, 92
using with channel sets, 91
using with mattes, 219
using with RotoPaint, 186
Merge toolbox, described, 8
merging
images, 24–30
premultiplied images, 26–28
Metadata toolbox, described, 8
midtones, 100, 102–103
matching by eye, 122–125
representing, 126–127
using Gamma property with, 122–125
mirror ball map, using with sky, 320–321
mix slider, using, 74–75
motion blur
adding, 154–157
properties, 154–155
turning on, 184
VectorBlur filter node, 156
motion vectors, generating, 155
MotionBlur2D node, connecting, 155
movement, creating with Tracker mode, 143–144
Multiply color-correction function, 100, 105
Multiply node, replacing Add node with, 107–108
multi-view node, identifying, 244

N

New Viewer, 4
nodal camera, 277
Node Graph, 2–3
  enlarging, 3
  switching between Curve Editor, 3
node indicators, using, 54–57
nodes. *See also* Gizmos; process trees
  anatomy of, 22
  arranging, 34
  branching, 30–32
  cloning ScanlineRender, 323–325
  connecting, 32–33
  connecting to Viewer, 11–12
  creating, 7–8, 30–32
  deleting, 34
  deselecting, 33
  disabling, 28, 34
  inserting, 30–32
  inserting to pipes, 49
  multi-view, 244
  naming, 8
  organizing, 78
  placing on pipes, 114
  replacing, 31–32
  resetting properties of, 35
  selecting, 33
  Shuffle, 50–51
  starting branches from, 31
Nodes menu, calling in Python, 369–370
Nodes Toolbar, 2–3, 7–8
noise, removing, 219
Noise node, using with RotoPaint, 185–186
nonlinear images. *See also* images
  Log, 228
  settings for, 227–229
  sRGB, 228
NoOp node, using in Gizmo, 342, 344–345
normalized values, explained, 99
.nuke directory, locating, 363
Nuke interface. *See* GUI (graphic user interface)
Nuke scripts. *See* scripts
NukeX license, 280
numbers, formatting, 43

O

OFlow (optical flow) node, using with stereo projects, 246–247
one point tracking, 143
OpenEXR file format, channel sets, 62–63
Other toolbox, described, 8

P

paint
  combining with roto and animation, 185–190
  dust removal with, 176–178
  using for wire removal, 175–176
painting in practice, 174–178. *See also* RotoPaint
Panasonic HVX-200 camera, 285
panels
  Curve Editor, 3
  locating names of, 3
  vs. panes, 2
panes, 2
  moving divide between, 3
  vs. panels, 2
  splitting, 5
panorama. *See* reflection maps
parallax, defined, 280
passes
  combining, 74–76
  connecting, 82
  defined, 62
  diffuse, 74
  layering, 73–76
  making branches for, 76–83
  manipulating, 85–87
  rendering, 64–65
  unpremultiplying, 78
pattern box, use with tracker point, 136
period (.), using to create Dots, 76, 78, 87, 312–313
perspective movement, 143
Photoshop layers. *See also* 3D scenes
  alpha channel, 328–329
  channel sets, 327
  determining opacity, 329
  importing, 326–333
  layering operations, 327
  splitting, 328
picture
  placing in frame, 148–152
  tracking in frame, 144–146
pipes
  changing course of, 76–77
  moving between nodes, 32–33
  moving for HueCorrect, 222
  placing nodes on, 114
pixel, defined, 62
pixel aspect ratio value, entering, 233
pixel-based properties, using in Proxy System, 239
PNG file format, channel sets, 62
PNG file sequence, rendering, 253–254
points
  changing locations of, 173
  creating on curves, 112, 173
position data
  accumulating, 136
  applying, 136
Premult option, accessing, 27–28
premultiplied images. *See also* images
  merging, 26–28
  stereo-view proxies, 244
  unpremulting, 116
  unpremultiplying images, 250
previewing images, 10
Primatte keying node, 200
process trees, 20–21. *See also* Groups; nodes
  adding functions to, 360
  controlling timing in, 191–196
  creating, 22–23
  displaying image resolution, 23
  flow of, 20–22
  highlighting areas in, 106
  organizing, 76–77, 92
  rendering, 42–43
  splitting, 76–83
  Viewer Input Process, 364–365
processing, speeding up, 66–68, 70
Progress Bar, 4
  moving, 4–6
  panels, 5, 45
project files. *See* scripts
Project Settings panel
  changing frame range in, 248
  displaying, 226
  LUTs (lookup tables), 227–229
  nonlinear images, 227–229
  Root tab, 226–227
  Views tab, 229–230
project speed, determining, 226–227
properties. *See also* Knobs
  adjusting, 38–39
  changing, 35–41
  keyframing, 52–54
  as knobs, 342
  linking with expressions, 69–71
  manipulating, 38–39
Properties Bin, 3
  clearing, 38
  editing properties in, 38
  loading Properties panel in, 37
  locking, 38
  panels in, 37
  removing Properties panels from, 38
  using, 37–38
Properties panels, 11
  Float panel button, 38
  floating, 128
  opening, 38
proxies. *See also* stereo-view proxies
  using with hi-res images, 238–242
  using with stereo script, 238–242

Proxy files, renaming, 242
proxy images, generating, 240–241
Proxy mode
  effect of, 72
  toggling on and off, 240
  using in camera projection, 317
  using with cube, 267
Proxy System
  Colorspace property, 241
  pixel-based properties, 239
  Proxy Scale property, 239
  Read node, 238
  rendering views in, 241–242
  turning on, 240
  Viewer, 239
PSD file format, channel sets, 62
Python buttons
  Clear History, 369
  Clear Output, 369
  creating, 369–371
  Load Script, 369
  Next Script, 369
  Previous Script, 369
  Run the Current Script, 369
  Save Script, 369
  Show Both Input and Output, 369
  Show Input Only, 369
  Show Output Only, 369
  Source Script, 369
Python scripting language, 368. *See also* TCL scripting language
  adding hot keys, 371
  calling up Gizmo, 370
  calling up Nodes menu, 369–370
  creating and adding icons, 372–375
  creating buttons, 368
  creating commands, 371
  creating menus, 370
  customization, 372–375
  menu.py file, 372–375
  menus, 368
  Nodes toolbar, 368
  running commands, 370
  Script Editor, 368
  User menu, 371

R

Radial node, creating, 131–132
Ramp node, 106, 334–335
Raw Data option, using with stereo script, 233–234
ray tracing, using in reflection, 298–299
Read nodes
  connecting into Multi-view stream, 237
  connecting via Views, 236
  creating, 9–10, 47
  in DAG, 23
  in DAG (Directed Acyclic Graph), 235–236
  in Proxy System, 238
  using versioning system with, 57
  using with process tree, 22
  using with stereo script, 243–246
  Write node as, 45
realtime, playing images in, 15
Reconcile3D nodes
  axis placement, 268–270
  connecting parts of, 270–271
  creating RotoPaint node, 273–274
  determining image resolution, 271
  Display Gain slider, 276
  Display Gamma slider, 276
  Execute panel, 272
  identifying reference frame, 275
  inserting Blur, 277
  KeyMix node, 276
  setting up, 268–273
  Translate properties, 273
  Translate.x values, 270
  turning on Tracker 2, 275
  using output with Tracker node, 273–277
Rectangle nodes, connecting for Safe Area tool, 352–353
redo functionality, 38
reflection. *See also* Camera Tracker
  adding to frame image, 302–303
  creating, 292
  cutting to size, 302–303
  environment light, 298–302
  image projected on Card node, 301
  importing script, 292–293
  masking, 302–303
  Merge operations, 302–304
  optimizing tree, 301
  panorama image, 300
  placing over composite, 302–303
  ray tracing, 298–299
  rotating image, 302
  ScanlineRender node, 293–296, 300–301
  Scene node, 299
  specular materials, 298–302
  Transform node, 301
reflection maps
  panorama in 2D, 295
  panorama in 3D Viewer, 294
  qualities of, 293
reflection movement, calculating, 280–281
reflective surface
  Card node, 295–298
  creating, 296–298
Reformat node, 106, 295–296
Render button, locating, 44
Render dialog box, using with RotoPaint, 184
Render menu, 4
render passes, 65
rendering
  choosing paths for, 43
  file sequences, 43–46
  moving images, 43–46
  naming file sequences, 43–46
  new versions and comparing, 57
  with Write node, 42–43
resolution. *See also* display resolution
  defining for stereo script, 231–233
  setting for images, 227
Retime node, using, 247–248
retiming elements, 246–249
Reveal brush, using in RotoPaint, 177–178
RGB slider, using, 40
RGBA channel set, 62
RolloffContrast operation, 104, 110
Rotate properties, using on sky, 325
roto, combining with paint and animation, 185–190
RotoPaint. *See also* Curve Editor; painting in practice
  animating shapes, 168–169
  Bézier hot keys, 167
  Bézier tool, 161
  Blending Mode property, 190
  Blur tool, 161
  Brush tool, 161
  B-Spline tool, 161
  Burn tool, 162
  changing length of strokes, 163–164
  Clone brush, 175–176
  Clone tool, 161
  compositing with KeyMix node, 187–188
  connecting Viewer inputs, 181
  deleting strokes, 165
  Dodge tool, 162
  Dope Sheet, 191–196
  drawing shapes, 166–168
  editing shapes, 166–168
  editing strokes, 163–165
  Ellipse tool, 161
  Eraser tool, 161
  erasing strokes, 165
  Feather Falloff property, 167–168
  hot keys, 168
  launching File Browser, 183
  Lifetime properties, 163–164
  locking shapes, 181
  Merge node, 186
  motion blur, 184
  Noise node, 185–186

on-screen controls, 160
painting in practice, 174–178
painting in vectors, 164–165
painting strokes, 162–163
pulling out soft edge, 168
Rectangle tool, 161
Render dialog box, 184
reordering strokes, 165
Replace option, 185
Reveal brush, 177–178
Reveal tool, 161
scaling brush sizes, 162
shape-drawing tools, 161
Sharpen tool, 162
ShuffleCopy node, 182
Smear tool, 162
split screening twins, 178–184
stroke-drawing tools, 161–162
Stroke/Shape List window, 188–190
switching between buffers, 181
Tool Settings bar, 161–162
Toolbar, 161
transforming groups of points, 167
RotoPaint nodes
creating, 176
loading, 160
using in camera projection, 318–320
using with Reconcile3D, 273–274

S

Safe Area tool, beta testing, 352–353
Safe Area tree, displaying, 354
SafeAreas Group, naming, 362
safety areas, 338–339. *See also* ActionSafe property
boxes for, 347–348
creating for titles, 346
moving boxes for, 349
Saturation color-correction function, 100
Save New Version option, 29–30
saving scripts, 29–30
ScanlineRender node
in 3D scenes, 258, 260
cloning nodes, 323–325
Projection Mode, 294
Reformat node, 295–296
using for reflection, 293–296
using in camera projection, 312–314
using with reflection, 300–301
Scene node
in 3D scenes, 258
using in camera projection, 312–314
Script Editor, 4
scripts. *See also* stereo script; TCL script
manipulating in text editors, 359–362
saving, 29–30, 45
Self Shadow pass, described, 65
sequences. *See* file sequences
settings. *See* Project Settings panel
Shadow pass, subtractive, 82
shadows, 100
matching with highlights, 122
representing, 126–129
setting Gamma properties, 129
shapes
animating, 168–169
drawing, 166–168
editing, 166–168
shots, stabilizing, 137–142. *See also* images
Shuffle node, using, 50–51
ShuffleCopy node
using, 84–85
using with HueCorrect, 221–222
using with RotoPaint, 182
sky
adjusting color of, 325
applying Rotate properties to, 325
Erode filter, 325–326
replacing with SphericalTransform, 320–322
SkyDome.png file, opening, 320
slap comps
in stereo projects, 250
using, 72
sliders
adjusting, 38–39
switching between fields, 40–41
slowing down elements, 246–249
Soft Clip property, 110
specular, defined, 299
Specular pass
branch, 87
described, 65
SphericalTransform, using to replace sky, 320–322
spills, suppressing with HueCorrect node, 220–222. *See also* Keylight keying node
SplitAndJoin node, using with stereo projects, 251
split-screen shape, 180
SplitScreen.####.png sequence, naming, 183
spoon position, tracking, 137–139
square_2k format, using with sky, 320
stabilize.####.tif sequence, loading, 137
statue.jpg file, opening, 148
Stencil operation
using with Grade node, 115
using with mattes, 219
using with Merge node, 49
stereo projects
changing properties for views, 251–253
compositing, 245–253
compositing elements together, 249–251
Gain property, 252
JoinViews node, 251–252
manipulating views apart from properties, 252
OneView node, 251–252
Optical Flow, 246
placing foreground over background, 249–250
retiming elements, 246–249
slap comp, 250
SplitAndJoin node, 251
unpremultiplying images, 250, 253
stereo script. *See also* hi-res images; scripts
applying color space conversion, 235
Colorscript drop-down menu, 234
defining resolution, 231–233
LUT (lookup table) settings, 233–235
setting formats, 231–233
using proxies, 238–242
stereo trees
Colorspace property, 253
Data Type property, 253
rendering and viewing, 253–255
using anaglyph glasses, 254
stereo views, 235–238
stereo-view proxies. *See also* proxies
alpha channel, 244–245
applying color management, 244
creating, 243–246
premultiplied images, 244
views for rendering, 245
Write nodes, 245
streams
connecting in multi-view tree, 237
creating via branching, 31
strokes
deleting, 165
editing in RotoPaint, 163–165
erasing, 165
painting in RotoPaint, 162–163
reordering, 165
Stroke/Shape List window, using in RotoPaint, 188–190
student_files folder, creating, 42–43
Switch node, connecting, 360–361
Switch Properties panel, 361

T

TCL script, typing, 77. *See also* scripts
TCL scripting language. *See also* Gizmo; Python scripting language
  Action Safe text in Viewer, 350
  adjusted expression, 350–351
  attaching text to Action Safe area, 349
  inserting Text node, 348
  linking expression, 349
  Message input field, 350
  moving box for safe area, 349
  Title Safe area, 350–351
  Transform group, 348–349
  using, 348–351
text editors
  adding functions, 360
  Check Box Knob, 361–362
  connecting Switch node, 360–361
  creating Knob for Which property, 361
  editing expressions, 361–362
  manipulating scripts in, 359–362
  search and replace functions, 359–360
Text node, inserting for TCL, 348
TextEdit, Find dialog box, 359–360
TIFF file format, channel sets, 62
Time toolbox, described, 7
timing of elements
  dealing with, 246–249
  showing in Dope Sheet, 249
Title Safe area, creating, 350–351
Title Safe area Knob, 358
TMI slider
  turning on, 130
  using, 40
Track Forward button, 138
tracked data, changing to transformations, 139
Tracker node
  features of, 136–137
  position data, 136
  Properties panel, 137–138
  using Reconcile3D's output with, 273–277
  using to create movement, 143–144
  using with CornerPin node, 149–150
tracker point, anatomy of, 136
Tracker Properties panel, Transform tab, 139
Tracker Settings tab
  Epsilon parameter, 146–147
  max_error, 147
  properties on, 146–148
tracking. *See also* 2D tracking; Camera Tracker; good_track script
  improving, 145–146
  picture in frame, 144–146
  types of, 143
tracking points
  overview, 143–144
  tracking, 144–146
tracks
  deleting keyframes for, 146
  stopping, 146
Transform group, using with TCL, 348–349
Transform node
  geometry, 301
  inserting, 30–31
  using with reflection, 301
Transform toolbox, described, 8
Transformation controls, using, 35–36
transparencies. *See* Keylight keying node
trees, 20–21. *See also* Groups; nodes
  adding functions to, 360
  controlling timing in, 191–196
  creating, 22–23
  displaying image resolution, 23
  flow of, 20–22
  highlighting areas in, 106
  organizing, 76–77, 92
  rendering, 42–43
  splitting, 76–83
  Viewer Input Process, 364–365
TV screen, creating, 326–335
two point tracking, 143

U

Ultimatte keying node, 200
undo/redo functionality, 38
updating Viewer, 235
User Knobs. *See also* Gizmo; Knobs
  ActionSafe property, 344–345
  changing expressions, 345, 347
  defining properties, 343–344
  in Groups, 353, 355
  naming tabs, 344
  replacing percentages, 345
  safety area for titles, 346
  settings for, 346
  shuffled alpha channel, 347
  TitleSafe, 347
  using in Gizmo, 342–348
  Width/Height Knob panel, 343, 345
User Knobs panel
  entries in, 356–357
  Operation Knob, 358
  ordering Knobs in, 358

V

values, manipulating, 54
VectorBlur filter node, using, 156
vectors, painting in, 164–165
versioning system, using with nodes, 57
versions, comparing, 58–59
Viewer, 2
  anatomy of, 12
  Channel buttons on, 65
  connecting nodes to, 11–12
  controlling frame range of, 226
  enlarging, 3
  features of, 11
  fps field, 15
  hot keys, 13, 15
  inputs, 14–15
  navigating, 12–13
  opening, 4
  pausing, 235
  playing clips in, 15–16
  in Proxy System, 239
  sampling colors from, 86
  updating, 235
  using, 13–14
Viewer Downscale Resolution option, 238, 240
Viewer Input Process, using, 364–365
viewers, adding and connecting, 4
Views
  buttons in Viewer, 237
  listing, 236
  managing, 230
  manipulating apart from properties, 252
  using to connect Read nodes, 236
Views toolbox, described, 8
View-specific nodes, location of, 237

W

wheels, brightening up, 131–132
white and black points, matching, 118–122
whitepoint property, 113
window layout, controlling, 3–6
windows, turning into Node Graph, 3
Wipe option, turning on, 58–59
wire removal, using paint for, 175–176
Write nodes
  as Read nodes, 45
  stereo-view proxies, 245
  using, 42–43
  using in camera projection, 336
  using versioning system with, 57
  with thumbnails, 45–46