2 Pulling Mattes

The first step towards any composite is to pull a matte. Although your software probably has one or two excellent keyers, such as Ultimatte, there are many occasions on which they don't work well or are not applicable at all. This chapter describes several alternative methods of creating mattes, some of which work with bluescreens and some of which work with arbitrary backgrounds. The purpose here is to provide you with as many different methods as possible for creating a matte for either compositing or element isolation – to put as many arrows in your quiver as possible. No single matte extraction procedure is good for all situations, so the more different approaches at your disposal, the better the chances are that you can pull a good matte quickly. Subsequent chapters will address the topics of refining the matte, despill methods, and the actual compositing operation itself.

A very important point to keep in mind is that pulling a matte from an image, even a perfectly shot bluescreen, is in fact a clever cheat – it is not mathematically valid. It could be characterized as a fundamentally flawed process that merely appears to work fairly well under most circumstances, so we should not be surprised when things go badly – which they do frequently. Technically, a matte is supposed to be an "opacity map" of the foreground object of interest within an image. It is supposed to correctly reflect semi-transparencies at the blending edges and any other partially transparent regions. We will see in the color difference matte section why even the best bluescreen matte is a poor approximation of this definition. Because the matte extraction process is intrinsically flawed, to be truly effective it is necessary to understand how it works in detail in order to devise effective workarounds for the inevitable failures of the process. Hopefully the following techniques will provide you with a wide range of approaches to help you navigate through the perverse terrain of your personal workspace.

2.1 LUMA-KEY MATTES

First up on the matte extraction hit parade is the ever popular luma key. Luma-key mattes get their name from the world of video, where the video signal is naturally separated into luminance and chrominance. The luminance (brightness) part of the video is commonly used to create a matte (or "key," in video-speak) in order to isolate some item for special treatment. In digital compositing we might refer to the same process as a luminance matte; however, most compositing packages will label this node as a luma-key node. Regardless of what you call it, it works the same way in both worlds. Some portion of the luminance of an image is used to create a matte. This matte (luma key) can then be used in any number of ways to isolate some item of interest for selected manipulation.

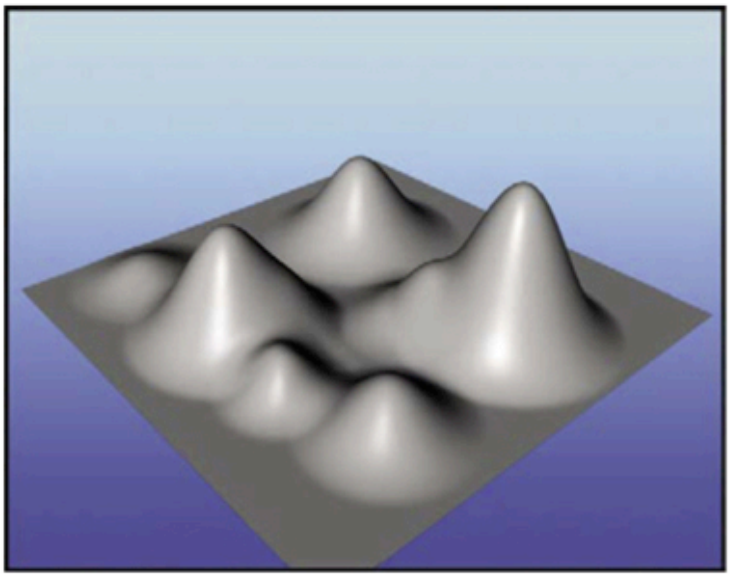
Luma-key mattes are simple to create and flexible in their use, because the item of interest is often darker or lighter than the rest of the picture. This section describes how they work, their strengths and weaknesses, and how to make your own customized versions to solve special problems.

2.1.1 How Luma-Key Mattes Work

A luma-keyer takes in an RGB image and computes a luminance version of it, which is a monochrome (one-channel grayscale) image. A threshold value is set, and all pixel values greater than or equal to the threshold are set to 100% white and all pixel values less than the threshold are set to black. Of course, this produces a hard-edged matte that is unsuitable for just about everything, so some luma-keyers offer a second threshold setting to introduce a soft edge to the matte.

My favorite visualization of a luminance key is to imagine the luminance image laid down flat on a table and the pixels pulled up into mountains. The height of each "mountain" is based on how bright the pixels are. Bright pixels make tall mountain peaks, medium gray pixels make low rolling hills, and dark pixels make valleys like those shown in Figure 2-1. This three-dimensional representation of two-dimensional images is a very useful way of looking at RGB images in general, which we will use a lot in this book.

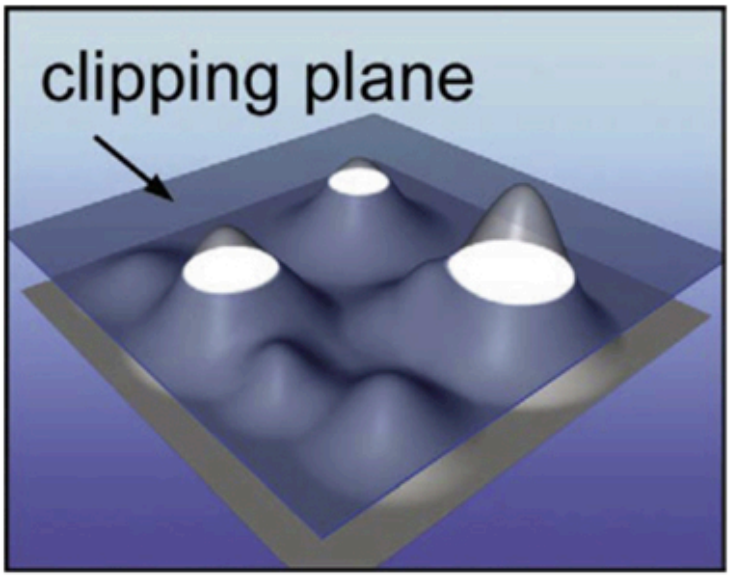
Figure 2-1 Monochrome image visualized as brightness "mountains"



You can now imagine the threshold point as a clipping plane that slices off the peaks of the mountains, like Figure 2-2. The white areas are the intersection of the

clipping plane with the mountain peaks and form the resulting luminance mask.

Figure 2-2 Clipping plane clips the tallest (brightest) mountain peaks

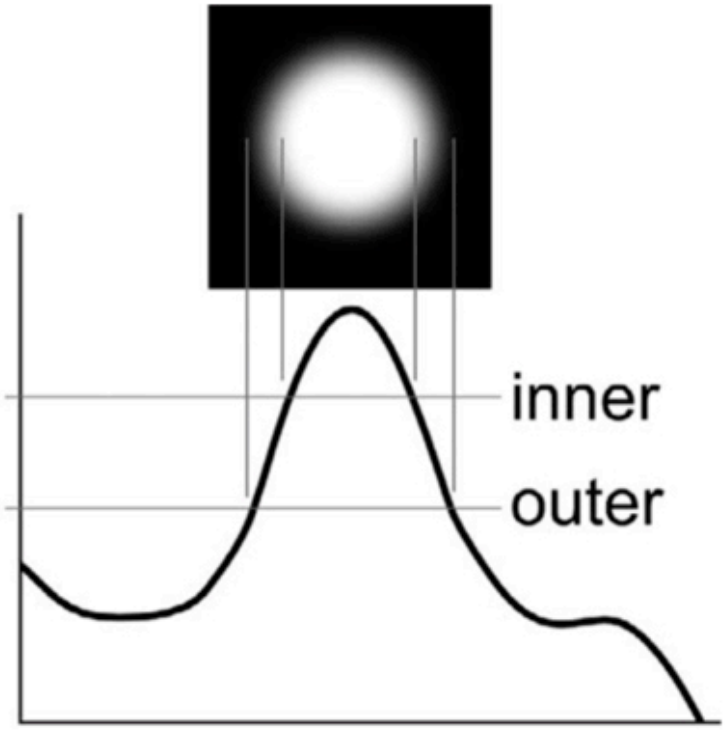


This metaphor provides a number of interesting insights to the luminance matte. The first and most obvious issue is how there are several different mountain peaks sliced off by the threshold point clipping plane, not just the one we may be interested in. This means that the luminance matte will have "snared" unwanted regions of the picture. Some method will have to be devised to isolate just the one item we are interested in. Another point is how the clipped regions will get larger if the threshold point (clipping plane) is lowered, and smaller if it is raised, which expands and contracts the size of the matte.

There are two approaches to the problem of snaring more regions than you want. The first is to garbage matte the area of interest. Crude, but effective, if the garbage mattes do not require too much time to make. The second approach entails altering the heights of the mountain peaks in the luminance image so that the one you are interested in is the tallest. This is covered in the following section, "2.1.2 Making Your Own Luminance Image."

Simply binarizing the image (separating it into just black and white pixel values) at a single threshold value creates a very hard-edged matte; most applications require a soft edge. The soft edge issue is addressed by having two threshold values in the luma-keyer settings. One setting is the inner, 100% density edge, and the other is the outer 0% density, with a gradient between them. Switching to a cross-section view of one of the "mountain peaks" in Figure 2-3, you can see how the inner and outer settings create a soft-edged matte. Everything greater than the inner threshold value is pulled up to 100% white. Everything below the outer threshold value is pulled down to a zero black. The pixels in between these two values take on various shades of gray, which creates a soft-edged matte shown in the inset above the mountain peak.

Figure 2-3 Inner and outer thresholds for soft-edged matte



2.1.2 Making Your Own Luminance Image

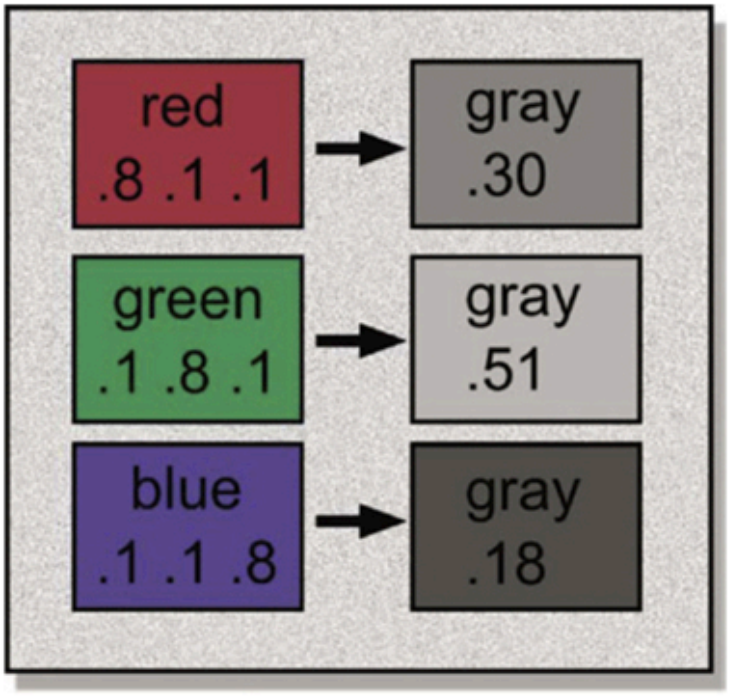
There are two important things to be aware of about the luminance image itself that is generated internally by the luma-keyer. The first is that there are a variety of ways to calculate this luminance image, and some of these variations might isolate your item of interest better than the default. The second thing to be aware of is that it does not have to be a luminance image at all. Your luma-keyer will most likely accept monochrome images in addition to RGB images as inputs. We can take serious advantage of this fact by feeding it a customized monochrome image that better isolates the item of interest. If it doesn't accept monochrome inputs, then we will make our own luma-keyer.

2.1.2.1 Variations on the Luminance Equation

The normal idea of a luminance image is to convert a three-channel color image to a monochrome image such that the apparent brightness of the monochrome image matches the apparent brightness of the color image. As usual, this is more complicated than it first seems. The obvious approach of simply taking ⅓ of each of the three channels and summing them together doesn't work, because the eye's sensitivity to each of the three primary colors is different. The eye is most sensitive to green, so it appears much brighter to the eye than equal values of red or blue.

Figure 2-4 shows how the eye responds differently to the three primary colors as luminance by taking as an example a set of three color chips made up of 80% of one primary color and 10% of the other two. The chip labeled red is made of the RGB values **0.8 0.1 0.1**, and when converted to luminance results in a gray value of **0.30**. The green chip has the exact same RGB values (except for green being the dominant color, of course), but its gray value is **0.51**, much brighter than the red. The poor blue chip hardly makes any brightness at all, with a paltry **0.18**.

Figure 2-4 Color converted to luminance

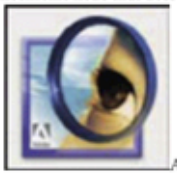


Here's the point: if you do not create the luminance version with the proper proportions of the red, green, and blue values that correctly correspond to the eye's sensitivities, the resulting luminance image will look wrong. A simple ⅓ averaging of all three colors would result, for example, in a blue sky looking too bright, because the blue would be over-represented. A green forest would appear too dark, because the green would be under-represented. One standard equation that mixes the right proportions of RGB values to create a luminance image on a color monitor is:

$$\text{Luminance} = 0.30\,R + 0.59\,G + 0.11\,B \qquad (2\text{-}1)$$

This means each luminance pixel is the sum of 30% of the red plus 59% of the green plus 11% of the blue. Be advised that the exact proportions differ slightly depending on what color space you are working in, so they may not be the exact values used in your luma-keyer. And, of course, these percentage mixes are different for media other than color monitors. If your software has a channel math node, then you can use it to create your own luminance images using Equation 2-1.

But making a luminance image that looks right to the eye is not the objective here. What we really want is to make a luminance image that best separates the item of interest from the surrounding picture. Armed with this information, you can now think about "rolling your own" luminance images that better separate your item of interest. Some luma-keyers allow you to juggle the luminance ratios. Try different settings to make your target item stand out from the background better. If your luma-keyer does not permit you to tweak the luminance equation, then feed it a luminance image you created externally with some other method such as a monochrome node that does allow you to adjust the mix ratios.



Adobe Photoshop Users – you too can create your own custom luminance images by making a custom grayscale image with the Channel Mixer. Go to Image > Adjustments > Channel Mixer, check the "Monochrome" box at the bottom of the dialog box (Output Channel goes to "Gray"), and then dial up your custom mix of colors to make your very own luminance image.

If your monochrome node does not permit you to change the RGB mix proportions, then another way you can "roll your own" luminance image is with a color curve node. The flowgraph in Figure 2-5 shows the operations. The RGB image goes into the color curve node where each color channel is scaled to the desired percentage mix value as shown in Figure 2-6. The three-channel output goes to a channel split node, so the R, G, and B channels can be separated then summed together to create a custom luminance image with the attributes you need. This luminance image is then connected to the luma-key node to pull the actual matte.

Figure 2-5 Flowgraph for generating a custom luminance image

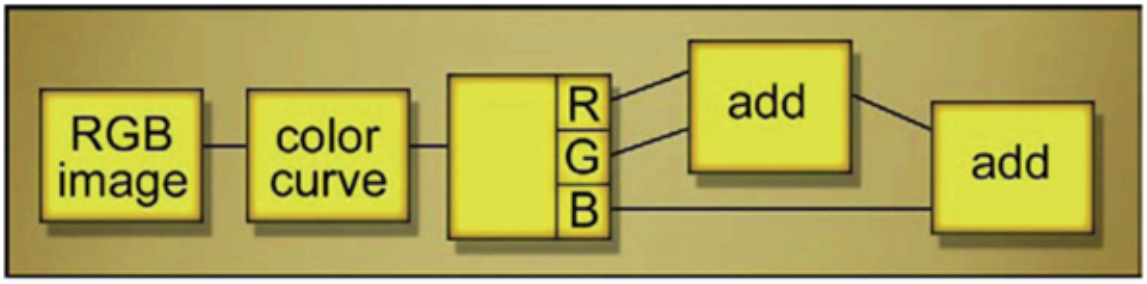
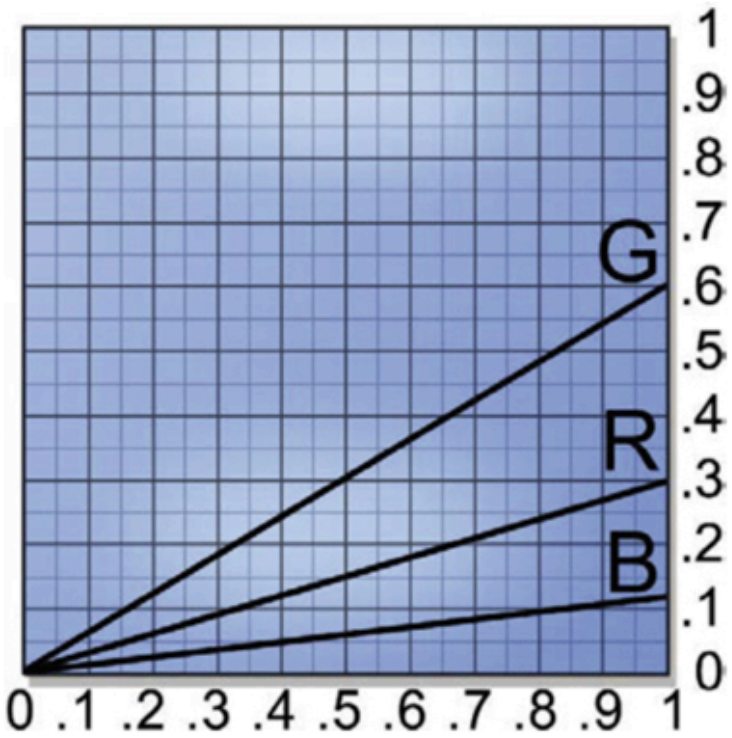


Figure 2-6 RGB scaling in the color curve



Adobe Photoshop Users – the luminance image is referred to as a "grayscale" image. You can mix your own grayscale image using the Channel Mixer (Image > Adjust > Channel Mixer). Click the monochrome button on and dial up the percentage mix of each color channel.

2.1.2.2 Nonluminance Monochrome Images

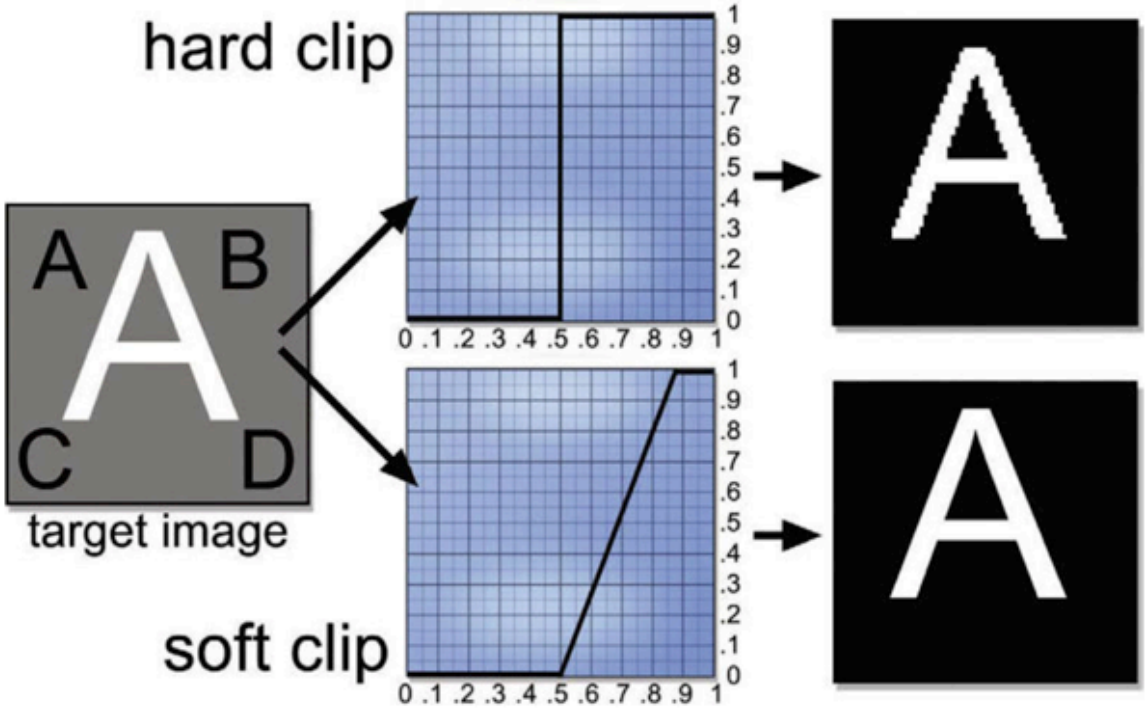
Another approach is to not use a luminance image at all. The lumakey process is simply based on the brightness values in a monochrome image. This one-channel image is usually created by making a luminance version of the color image, but it is not against the laws of man or nature to make the one-channel image in other ways. For example, just separate the green channel and pipe it into the luma-keyer. Perhaps the blue channel has a better separation of your item of interest from the rest of the picture, or maybe a mix of 50% of the red channel plus 50% of the green channel will work best. It totally depends on the color content of your target and the surrounding pixels. Unfortunately, the RGB color space of an image is so complex that you cannot simply analyze it to determine the best approach. Experience and a lot of trial and error are invariably required. Fortunately, the iterations can be very quick and a good approach can usually be discovered in a few minutes.

2.1.3 Making Your Own Luma-Key Matte

Perhaps you are using some primitive software that does not have a lumakeyer, or you don't want to use your luma-keyer for political or religious reasons. Maybe you just want to demonstrate your prowess over pixels. Fine. You can make your own luma-keyer using a color curve node, which every software package has. The first step is to make your best monochrome image that separates the item of interest from the background using one of the techniques described earlier, then pipe it to a color curve node to scale it to a "hicon" (high-contrast) matte.

How to set up the color curve is demonstrated in Figure 2-7. The item of interest is the white letter "A" in the target image on the left, a deliberately easy target for demo purposes. The color curve labeled "hard clip" uses a single threshold value, but this results in a hard matte with jaggy edges (effect exaggerated for demo purposes). The color curve labeled "soft clip" shows a slope introduced to the color curve to create a soft edge for the matte. The gentler the slope of the line, the softer the edge. Of course, nothing says that the target object will be the white object. A black object could be the target, so in that event the color curve would be set to pull all pixels *lighter* than the black target pixels up to 100% white. This produces a black matte on a white background, which can be inverted later if needed. Alternately, the color curve could pull the black pixels up to 100% white and all others down to black to produce a white matte on a black background directly.

Figure 2-7 Using a color curve to make a luma-key matte



Very often the target object is a middle gray, like the one in Figure 2-8. The pixel values of the target object are pulled up to 100% white with the color curve, and all pixel values greater *and* less than the target are both pulled down to black. The sloped sides ensure a nice soft edge to the matte. Occasionally the luma key is pulled on the darkest or lightest element in the frame so the simple soft clip shown in Figure 2-7 can be used, but most often it will be some middle gray like this example, where you have to suppress pixels that are both lighter and darker than the target.

Figure 2-8 Luma-key matte for a middle gray target

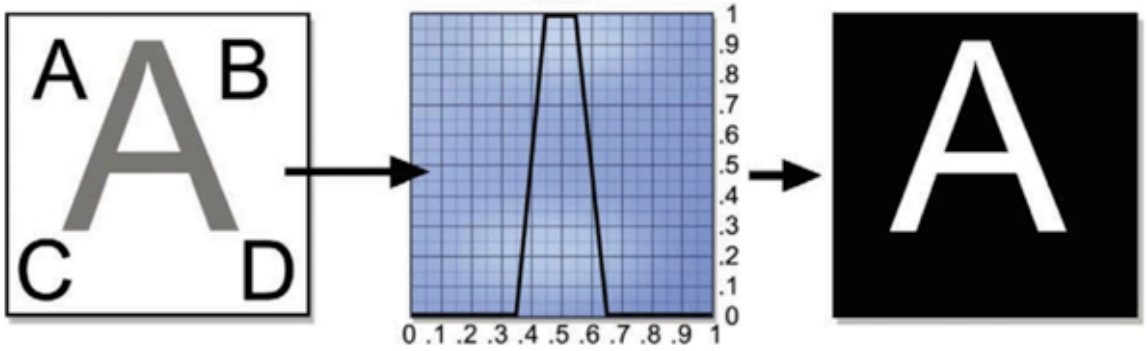
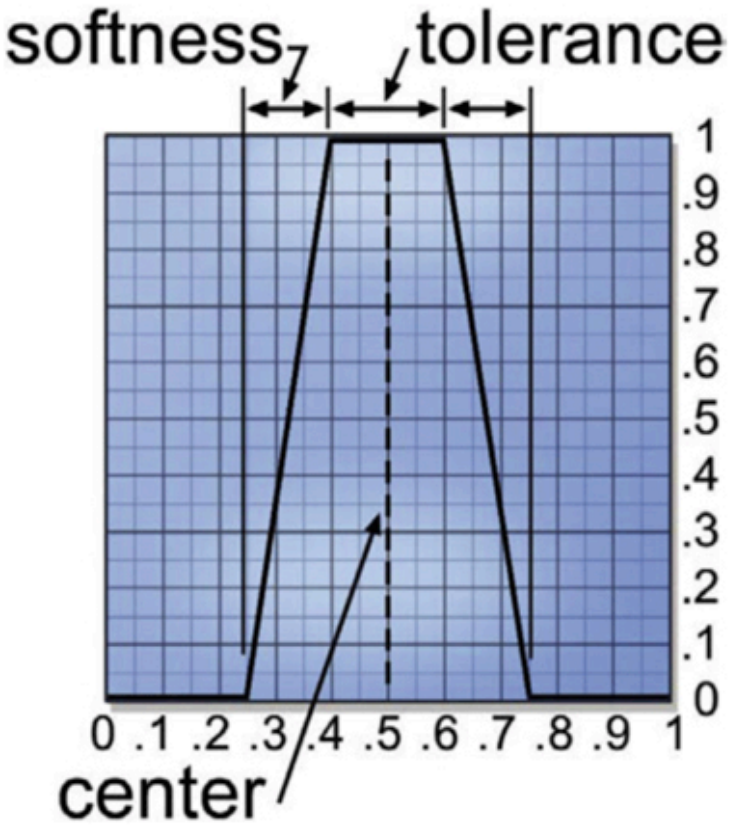


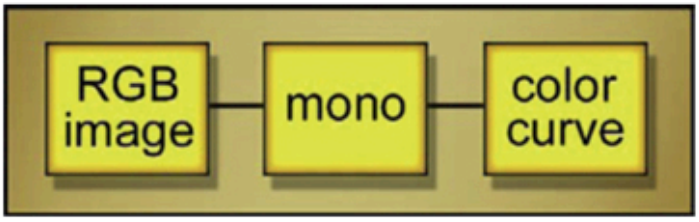
Figure 2-9 shows the details of shaping the matte with a curve in the color curve node. The "center" represents the average luminance value of the target. The "tolerance" is how wide a range of luminance will end up in the maximum density region of the resulting matte. The "softness" reflects the steepness of the slope between black and white, and the steeper this slope the harder the matte edges will be. Only four control points are needed to adjust the tolerance and softness. Measuring a few pixel values will give you a starting point, but you will probably end up adjusting the four control points visually while watching the resulting matte in order to see the effects of the changes to the tolerance and softness.

Figure 2-9 Details of a luma-key scaling operation



Ex. 2-1 To summarize the homemade luma-key process, Figure 2-10 shows a flowgraph of the operations described previously. The RGB image is piped into a "monochrome" node, which makes a luminance version of color images. Hopefully yours has internal settings that can be tweaked for best results. The mono image is then piped to a color curve to scale it up to the final hicon matte. Alternately, instead of the mono node, a single channel of the RGB image might be used, or perhaps some mix of two channels as described earlier for making your own luma keys.

Figure 2-10 Flowgraph of luma-key operations



2.2 CHROMA-KEY MATTES

Chroma-key mattes also get their name from video work, where again, the video signal is naturally separated into luminance and chrominance. The chrominance (color) part of the video is used to create a matte (key) in order to isolate some item for special treatment. The same principles are used in digital compositing. Because the chroma key is based on the color of an object, it can be more selective and hence more flexible than the simple luma key. The chroma key also comes with a list of limitations and liabilities, which are explored in this section along with some workarounds and how to make your own chroma-key type mattes.

2.2.1 How Chroma-Key Mattes Work

The chroma-keyer node takes in the RGB image and converts it to an HSV (Hue, Saturation, Value or, if you prefer, color, saturation, brightness) or HSL type of representation internally. The reason that the original RGB version is not used internally is because the keyer needs to distinguish between, say, levels of saturation, which is not an RGB attribute, but is an HSV attribute. A starting RGB value is set which repeesents the "center" of the chromakey matte, then additional tolerances are set that permit the addition of saturation and value (brightness) ranges to be included in the matte. A good chroma-keyer will also permit some sort of "tolerance"

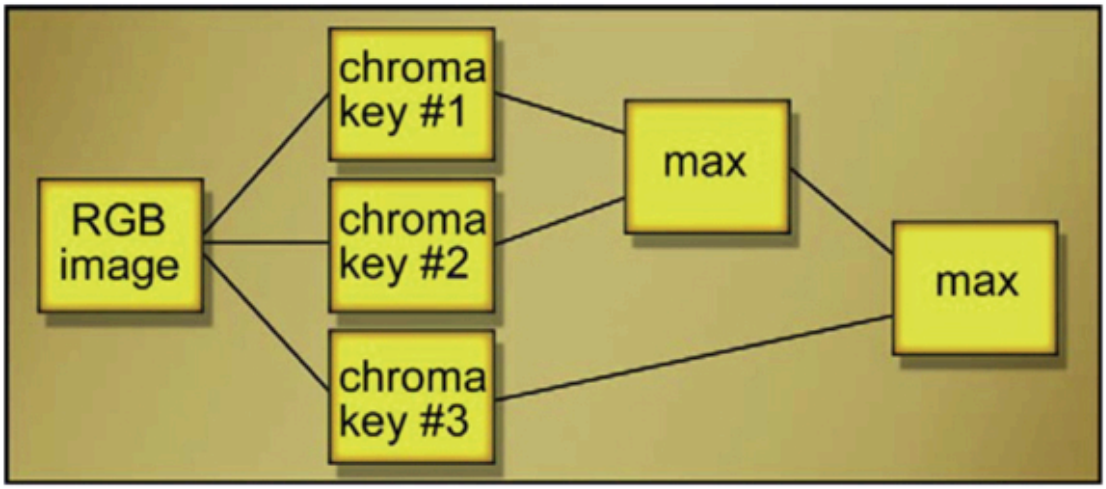
settings to allow for a graceful transition with soft edges.

The chroma key is very flexible for two important reasons. First, it allows you to go after any arbitrary color. It does not have to be a carefully controlled bluescreen, for example. Second, it accommodates the natural variations in the color of real surfaces by having saturation and value ranges to expand its window of acceptance. If you examine the color of skin, for example, you will find that the shadow regions not only get darker, but also have lower saturation. To pull a matte on skin, therefore, you need something that will not only select the RGB color you picked for the main skin tone, but also follow it down into the shadows with lower saturation and value.

Having said all that, let me also say that the chroma key is not a very high-quality matte. It is prone to creating mattes with hard edges that require blurring, eroding, and other edge processing to try and clean up. It also does not do semi-transparent regions very well at all. You must also never use it for bluescreen work, except to pull garbage mattes in preparation for more sophisticated matte extraction processes. For bluescreen work it is terrible with edge-blended pixels – those wisps of blonde hair mixed with the bluescreen, for example.

Some of the shortcomings of the chroma key can be compensated for by pulling several chroma-key mattes and "maxing" them together. That is to say, use a maximum operation to merge several chroma-key mattes together as shown in the flowgraph in Figure 2-11. Each chroma key pulls a matte from a slightly different region of color space so that when combined, they cover the entire item of interest.

Figure 2-11 Flowgraph for combining multiple chroma keys




Adobe Photoshop Users – the "Magic Wand Tool" is in fact a manually operated chroma-key tool.

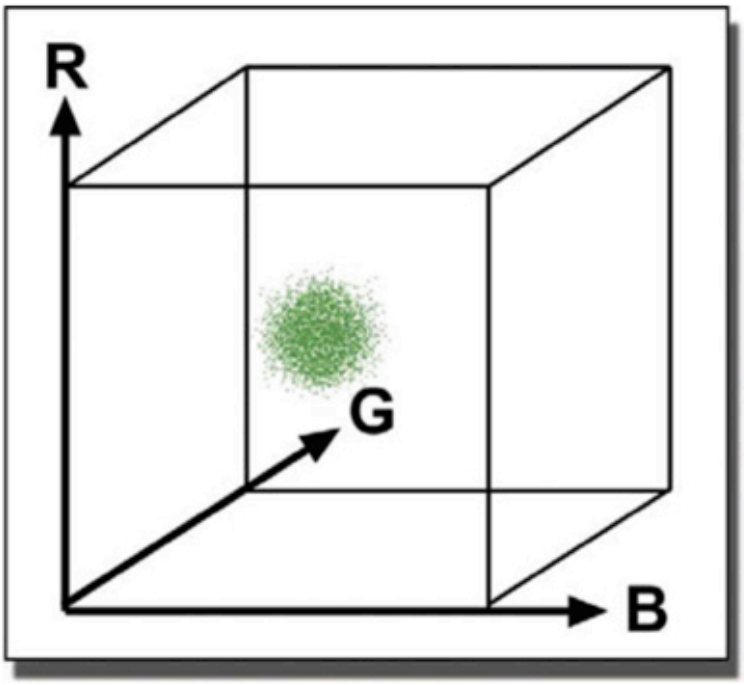
2.2.2 Making Your Own Chroma-Keyer

Here is an interesting and flexible variation on the chroma-keyer theme that you can make at home. Although it is based on utterly different principles than the classic chroma-keyer, which converts the image to an HSV type of representation for manipulation, it is still based on the specific color of the item of interest, not simply its luminance. Because it is based on making a matte for any arbitrary color, I have tossed it into the chroma-key category. It is actually a three-dimensional color keyer that uses a much simplified version of the operating principles of Primatte.

As you probably know, any given pixel can be located in RGB color space by its RGB value. If we imagine a normal three-dimensional coordinate system labeled RGB instead of XYZ, we can imagine any given pixel being located somewhere in this "RGB cube" by using its RGB values as its XYZ location coordinates. We could then take a greenscreen plate like the one in Figure 2-12 and plot the location of just the green backing pixels in this RGB cube. As all of the pixels in the green backing part of the picture have very similar RGB values, they would all cluster together into a fuzzy green blob like the one shown in the RGB cube of Figure 2-13.

Figure 2-12 Greenscreen plate

Figure 2-13 Green backing pixels plotted in RGB color cube



Now, what if we could isolate this little constellation of green pixels? Suppose we picked a point right in the center of the green pixel constellation by choosing the appropriate RGB value, then measured the distance between this center point and all of the pixels in the image. The green backing pixels all hover in the vicinity of the center point, so the distance to them would be zero or close to it. All other pixels would be located much further from the center point, giving them distances much greater than zero.

Suppose, further, that we created a new version of the greenscreen image that showed the distance each pixel is from this center point – call it a distance map. Pixels very close to the center point would be black because their distance to the center point is at or near zero. The entire green backing region would therefore be black, yet the skin pixels, being further away, would have some shade of gray. What we wind up with is a grayscale image of the same picture with the brightness of each pixel indicating its distance from the target color, like the distance map in Figure 2-14. The closer a pixel is to the center point RGB value the blacker it is. By using a color curve to separate the black "near" pixels from the gray "far" pixels, a hicon matte can be made like the one in Figure 2-15.

Figure 2-14 Distance map of greenscreen plate

Figure 2-15 Hicon version of distance map

This three-dimensional color matte can be created for any arbitrary RGB value, not just a greenscreen or bluescreen, which is what makes it as flexible as the chroma key. The target color's RGB value is used as the center point, then the distance map is created relative to it. If the skin tones were used for the center point, the skin would have been black and all other pixels shifted in gray value to reflect their "RGB distance" from this new center point in the color cube.

Now the icky part – the math. It is simply based on the trigonometry equation for calculating the distance between two points in 3D space. You remember from your favorite trig class, for point 1 located at x_1, y_1, z_1 and point 2 located at x_2, y_2, z_2, the distance between point 1 and point 2 is

$$\text{Distance} = \text{square root}((x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2) \qquad (2\text{-}2)$$

To apply this equation to our matte, point 1 becomes the center point RGB value and point 2 becomes the RGB value of each pixel in the image, with all RGB values normalized to between 0 and 1.0. This means that the smallest distance value we can get is 0 and the largest is 1.0, which results in a grayscale image with pixel values ranging between 0 and 1.0. To reformat the trig equation for RGB color space, it becomes

$$\text{Gray value} = \text{square root}\left((R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2\right) \qquad (2\text{-}3)$$

where $R_1G_1B_1$ is the normalized value of the center point – that is, the color you want to isolate – and $R_2G_2B_2$ is the normalized value of each pixel in the image. This equation can be entered in a math node or some other expression format that your software supports and you can create your own threedimensional color mattes. Good hunting!

Adobe Photoshop Users – unfortunately Photoshop does not support this ability to write your own math equations. Lucky you.

2.3 DIFFERENCE MATTES

Difference mattes are one of those things that sounds really great until you try it. They don't work very well under most circumstances and have nasty edges in almost all circumstances. Their most suitable application is usually garbage matting. However, for those occasional situations where they might be helpful, the difference matte is herein presented.

2.3.1 How Difference Mattes Work

The difference matte works by detecting the difference between two plates: one plate with the item of interest (the "target") to be isolated, and the other a clean plate without the item of interest. For example, a scene with the actor in it, and the same scene without the actor. This obviously requires the scene to be shot twice with either a locked off camera or, if the camera is moving, two passes with a motion control camera. In the real world, this rarely happens, but under some circumstances you can create your own clean plate by pasting pieces of several frames together. However it is done, a clean plate is required.

A target over its background is illustrated with the monochrome image in Figure 2-16. This simplified example will make the process much easier to follow than a color image with complex content. The background is the speckled region and our target is the dark gray circle, which is standing in for the actor. Figure 2-17 represents the clean plate. To create the raw difference matte, the difference is taken between the target and clean plates. For those pixels where both plates share the common background, the difference is zero. Those pixels within the target have some difference compared to the clean plate. That difference may be small or large, depending on the image content of the target relative to the clean plate.

Figure 2-16 Target plate

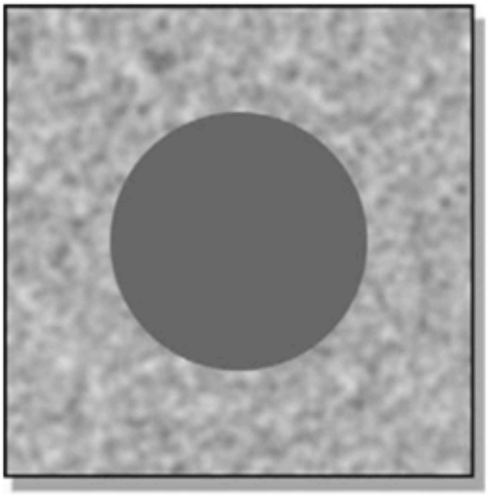
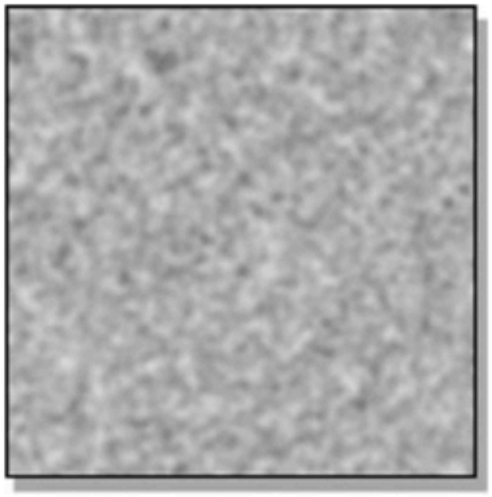


Figure 2-17 Clean plate



In the real world, the background region in the target and clean plates are never actually identical due to film grain plus any other differences in the clean plate such as a small change in lighting, a leaf blown by the wind, or a camera that has been bumped. This will show up as contamination of the black background region.

The raw difference matte produced by the difference operation can be seen in Figure 2-18. Note that it is not a nice 100% white clean matte ready to go. It has all kinds of variations in it due to the fact that the background pixels are of varying value and the foreground pixels are also of varying value, so the results of their differences will also be of varying values. To convert the low density raw difference matte into a solid hicon matte like Figure 2-19, it is typically "binarized" around a threshold value. To binarize an image simply means to divide it into two (binary) values, zero and 1.0, at some threshold value. All pixels at or above the threshold go to 100% white; those below go to zero. Of course, this single-threshold approach will produce a jaggy hard-edged matte. A difference matte node worth its pixels will have two threshold settings that you can adjust to select the range of differences that appear in the matte versus the background in order to introduce a softer edge.

Figure 2-18 Raw difference matte

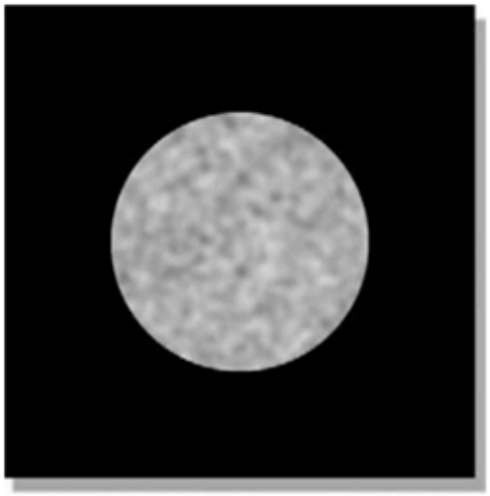
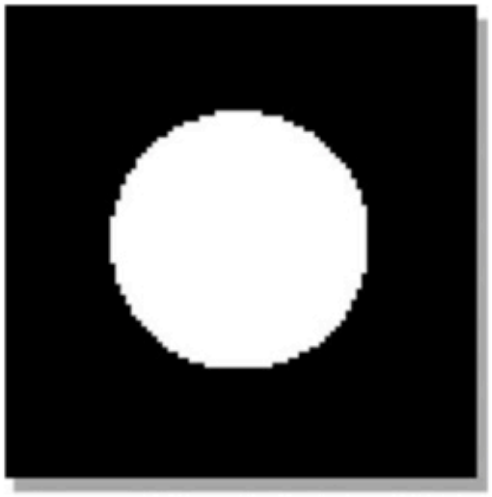


Figure 2-19 Binarized hicon difference matte



Now for the bad news. Remember those pixels in the target mentioned earlier that were close in value to the pixels in the clean plate? They and their close associates spoil the difference matte. Why this is so can be seen in Figure 2-20, which represents a slightly more complex target. This target has pixels that are both lighter and darker than the background, and some that are equal to it. When the raw difference matte is derived in Figure 2-21, there is a black stripe down the center, because those pixels in the target that are close to the pixel values in the background result in differences at or near zero. Both the left and right ends are brighter, because they are much darker and lighter than the background, resulting in greater difference values.

Figure 2-20 Complex target

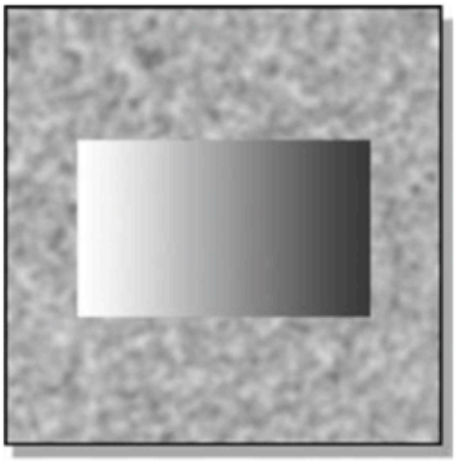
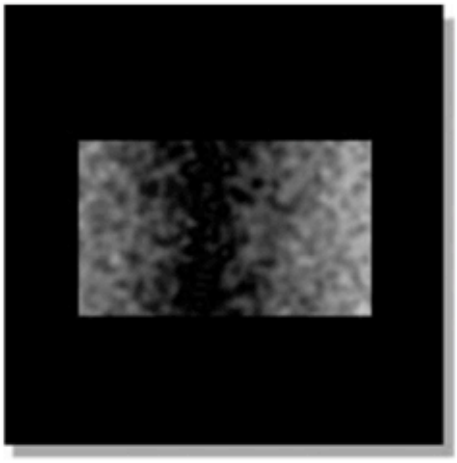


Figure 2-21 Raw difference matte



When the raw difference matte is scaled up to create the solid hicon version shown in Figure 2-22, there is a serious "hole" in the matte. And this is the problem with difference mattes. The target object will very often have pixel values that are very close to the background, resulting in holes in the matte. The raw difference

matte normally must be scaled up hard to get a solid matte for these small difference pixels, and this produces a very hard matte overall. However, there are occasions where the difference matte will work fine, so it should not be totally dismissed.

Figure 2-22 Hicon difference matte

2.3.2 Making Your Own Difference Matte

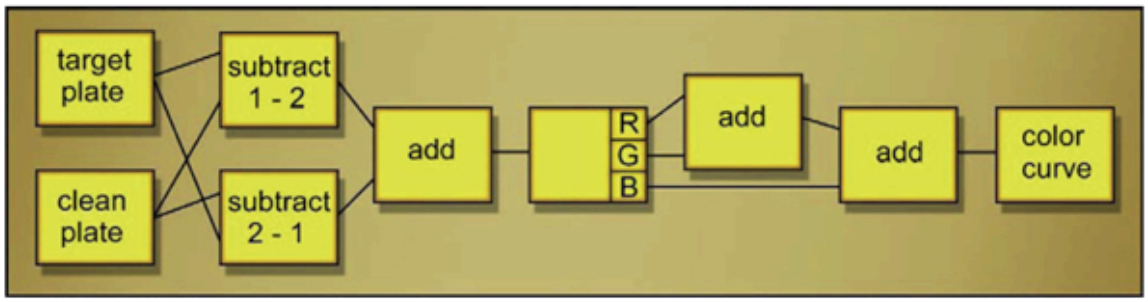
For reasons that are now all too obvious, many software packages do not offer a difference matte feature, but you can make your own. To create the difference matte, the target and clean plates are subtracted twice and the two results are summed together. Two subtractions are required because some of the target pixels might be greater than the background and some less than the background. In "pixel math," negative pixel values are normally not permitted. If you tried to subtract 100 – 150 you will get 0 as an output, not –50. We want this –50 pixel for our matte, but the pixel math operation clipped it to zero. Unless your software has an "absolute value" operator, it will take two subtractions to get both sets of pixels into the matte. So our math operation would look like this:

$$\text{Raw difference matte} = (\text{target plate} - \text{clean plate}) + (\text{clean plate} - \text{target plate}) \quad (2\text{-}4)$$

The target pixels that are *greater* than the background pixels in the clean plate will be found with the first subtraction (target plate – clean plate). The target pixels that are *less* than the background pixels in the clean plate will be found with the second subtraction (clean plate – target plate). These two sets of pixels are then summed together to make the raw difference matte. As you saw earlier, any target pixels that are equal to the background will show up as holes in the matte. As for the background surrounding the target, theoretically, these pixels are identical in the two plates so their differences become zero. Theoretically.

You will undoubtedly be performing this operation on three-channel color images instead of the monochrome examples so far. You can perform the two subtractions and one addition operation shown in Equation 2-4 on the full color images with just add and subtract nodes as shown in the flowgraph in Figure 2-23. When the two subtractions are summed together, however, you will have a three-channel image that needs to become a one-channel image. Just separate the three channels and sum them together to make the monochrome raw difference matte, then use a color curve to scale it up to a hicon matte.

Figure 2-23 Flowgraph of homemade difference matte



If you have a channel math node, the entire operation can be put into one node with the following equation:

$$\text{Raw difference matte} = (\text{abs}(R_1 - R_2)) + (\text{abs}(G_1 - G_2)) + (\text{abs}(B_1 - B_2)) \quad (2\text{-}5)$$

where "abs" means absolute value, and $R_1G_1B_1$ are the RGB values of one plate and $R_2G_2B_2$ are the RGB values of the other plate. It matters not which is plate 1 and which is plate 2. And, of course, this raw difference matte must then be scaled with a color curve to make the final hicon matte. Good luck.

Adobe Photoshop Users – you can add and subtract color channels using the Calculations command (Image > Calculations). The "Difference" blending mode does not use the same math as described here, so it does not produce as robust a difference matte. It might occasionally work well enough to be worth a try, however. In Photoshop, you only have to worry about one frame rather than a sequence of moving frames, so the magic wand is your best friend here.

2.4 BUMP MATTES

Here is a sweet little number that you will find indispensable perhaps only one or two times a year, but when you need it, there is no other game in town. The situation is that you want to pull a matte on the bumps of a rough surface – a stippled ceiling, the textured grass of a lawn, the rough bark on a tree, and so on. You want to isolate and treat the bumpy bits, perhaps to suppress or highlight them. The bump matte will give you a matte of just the bumps – and here is the really sweet part – even when they are on an uneven surface. A luminance matte will not do. Because of the uneven brightness, many of the bump tips are darker than the brighter regions of the surface. You need a method that just lifts out the bumps, regardless of their absolute brightness. The bump matte is based on the *local* brightness of the region immediately surrounding each bump, so it continuously adapts to the local brightness everywhere in the picture.

The idea is to make a luminance version of the target plate that makes the bumps stand out as much as possible. A blurred version of the target plate is then generated, and this blurred version is subtracted from the luminance version. Wherever the bumps stick up above the blurred plate, you will get a matte point. This is an entirely different result from performing an edge detect. The edge detection would actually draw circles around all the bumps, while the bump matte actually gets the bumps themselves.

Figure 2-24 represents a generic bumpy surface. Note how the left edge is a medium brightness; it gets brighter towards the center, then gets much darker towards the right edge. This is not a flat, evenly lit surface. Figure 2-25 shows the blurred version of the original bumpy plate in Figure 2-24, and Figure 2-26 shows the resulting bump matte. Note how each bump became a matte point, even though the surface was unevenly lit. It is as if the bumps were "shaved off" the main surface in Figure 2-24 and laid down on a flat surface.

Figure 2-24 Target bumpy surface

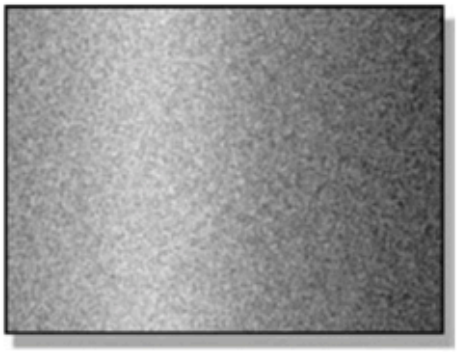
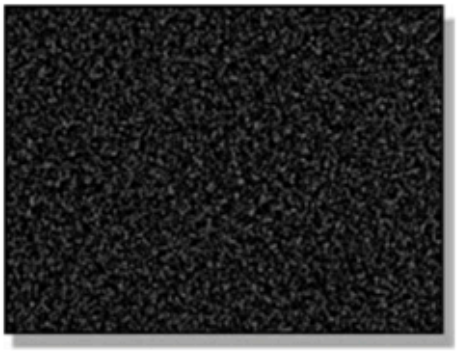


Figure 2-25 Blurred bumpy surface

Figure 2-26 The resulting bump matte



The action can be followed in the slice graph in Figure 2-27, where a slice line has been drawn horizontally across the bumpy target image in Figure 2-24. The bumps in the image create the bumpy slice line. After the image is blurred, the bumps are all smoothed down, but the overall contours of the surface are retained in the slice line labeled "blurred." The blurred surface will in fact be the average of the bumpy surface. When the blurred plate is subtracted from the bumpy plate all of the bumps that stick up above the blurred version are left as the bump matte at the bottom of Figure 2-27.

Figure 2-27 Bump matte

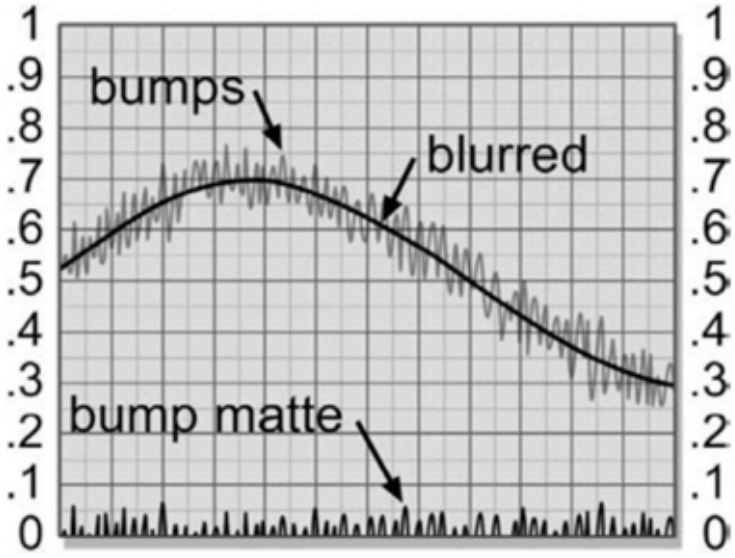
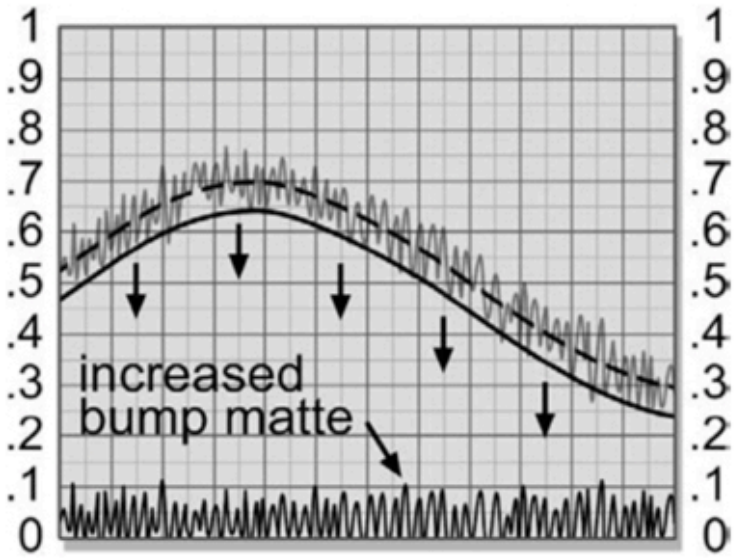


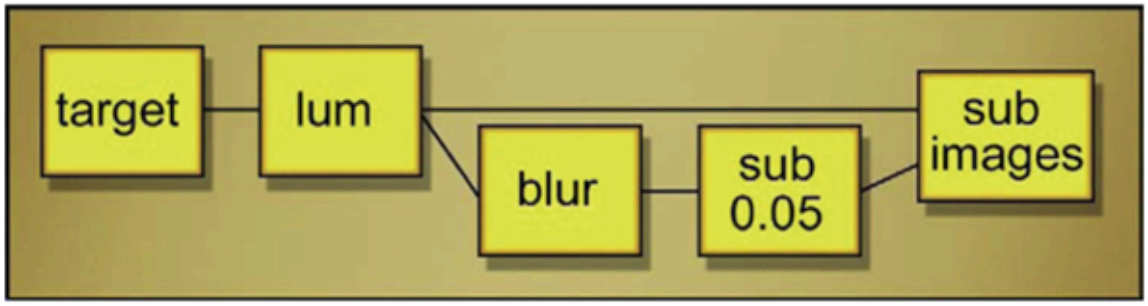
Figure 2-28 shows an important refinement of the matte where the matte points have been increased by first lowering the blurred plate a bit before the plate subtraction. Simply subtracting a small constant value (such as 0.05) from the blurred plate "lowers" it so that the remaining bumps stick up higher. The resulting matte has both larger and "taller" (brighter) matte points. Increasing and decreasing the amount of blur is another refinement. The greater the blur, the larger the bumps that will be trapped in the bump matte.

Figure 2-28 Raised bump matte



Ex. 2-5 Figure 2-29 illustrates a flowgraph of the blur matte operations. Starting with the target image on the left, the luminance version is created. Next it is blurred, then a small constant such as 0.05 may be subtracted from it to increase the size of the bumps in the map. If you want to decrease the size of the bumps then you would add a constant instead. The adjusted blurred image is then subtracted from the luminance image of the target plate.

Figure 2-29 Flowgraph of bump matte procedure



2.5 KEYERS

There are several high-quality third-party "keyers" on the market such as Ultimatte, Primatte, and Keylight. What they all have in common is that they offer a turnkey solution to bluescreen compositing. You connect the foreground and background plates to the keyers node, adjust the internal parameters for matte and despill attributes, and they produce a finished, despilled composite. Ultimatte and Keylight both work on color difference principles, whereas Primatte is a very sophisticated three-dimensional chroma-keyer. So why not just use one of these all the time and not worry about pulling your own mattes?

There are several possible reasons. You may be working with problem elements that are giving the keyer problems, so you may have to assist the keyer by preprocessing the bluescreen plate to set it up for a better matte extraction. Another issue is when the despill operation goes wrong. A surprising number of edge discoloration artifacts are from the despill operation. Because it is internal to the keyer, when you cannot adjust the problem away, you have no way to substitute a different despill operation. If you cannot use the despill in the keyer, then you cannot use the keyer. You will have to pull the matte separately, use some other despill operation (see Chapter 4, "Despill") that does not introduce the artifact, then do the final composite yourself.

Sometimes the built-in color-correction will not solve the color-correction problem. Another problem may be that the built-in color-correction capability may not be adequate for certain cases. You may also have other operations to do to the foreground that must come after the despill but before the composite. You should not color-correct the raw bluescreen plate before giving it to the keyer, for several reasons. The first big problem is that if the color-correction has to be revised later, all of the keyer settings have to be completely redone. Second, the color-corrected version may disturb the relationships between the channels, which can degrade the matte extraction or the despill operations, or both.

Should you find that your favorite keyer is not doing the job, it still may be useful for pulling the original matte, then do the despill and composite externally. Virtually all keyers will output the matte that they develop internally for their own composite. This matte can be further refined if needed and used with a regular comp node to perform the actual composite. Although this approach may be necessary to solve a problem, be sure to always try to do the comp in the keyer first, as they usually do an excellent job and are much quicker to set up than creating your own color difference mattes.

2.6 COLOR DIFFERENCE MATTES

The color difference matte (not to be confused with the plain "difference matte") is one of the best matte extraction methods for bluescreens, because of its superior edge quality and reasonable semi-transparency characteristics. As mentioned earlier, the Ultimatte method is in fact a sophisticated color difference matte extraction and compositing technique. In this section, you will see how to "roll your own" color difference mattes. Though originally conceived for bluescreen matte extraction, the color difference matte technique can often be used to pull a matte on an arbitrary object that has a single dominant color, such as a red shirt or a blue sky, for example.

The importance of this section is twofold. First, by understanding the color difference matte process, you will be able to "assist" your favorite keyer when it's having trouble. Second, if you can't use a keyer, either because your software doesn't have one or it's having problems with a bluescreen plate, you can always make your own color difference matte. The methods outlined here can be used with any compositing software package or even Adobe Photoshop, as they are based on performing a series of simple image processing operations that are universally available.

This color difference matte section is extensive for two reasons. First, it is the single most important matte extraction procedure because it does the best job in the most common and critical application – bluescreen and greenscreen matte extraction. Second, it is also more complex and harder to understand than other matte extraction methods. Although the general term for this type of shot is a "bluescreen" shot regardless of the actual backing color, most of the examples in this section use greenscreens, simply because it is becoming the most common backing color used. The astute reader will be able to easily see how the principles apply equally to the other two primary colors, blue and red. Yes, you can shoot redscreens as well as bluescreens and greenscreens, with equally good results – assuming that the foreground objects do not have the same dominant primary color as the backing color, of course.

2.6.1 Extracting the Color Difference Matte

This first section is about how to extract or "pull" the basic color difference matte using simplistic test plates designed to avoid problems. After the basic principles of how the matte extraction works are clear, the sections that follow are concerned with fixing all the problems that invariably show up in real world mattes.

2.6.1.1 The Theory

Referring to Figure 2-12, note how a greenscreen plate consists of two regions, the foreground object to be composited and the backing region (the greenscreen). The whole idea of creating a color difference matte from a greenscreen is that in the green backing region, the *difference* between the green record and the other two records (red and blue) is relatively large, but zero (or very small) in the foreground region. This difference results in a raw matte of some partial density (perhaps 0.2 to 0.4) in the backing region, and zero (or nearly zero) density in the foreground region.

The different densities (brightness) of these two regions can be seen in the raw color difference matte in Figure 2-30. The dark gray partial density derived from the greenscreen backing color is then scaled up to 1.0, a fulldensity white, as shown in the final matte of Figure 2-31. If there are some nonzero pixels in the foreground region of the matte, these are pulled down to zero to get a solid foreground matte region. This results in a matte that is a solid black where the foreground object was and solid white where the backing color was. Some software packages need this reversed, with white for the foreground region and black for the backing, so the final matte can simply be inverted if necessary.

Figure 2-30 Raw color difference matte

Figure 2-31 Final color difference matte

2.6.1.2 Pulling the Raw Matte

The first step in the process is to pull the raw matte. This is done with a couple of simple math operations between the three color channels and can be done by any software package using simple math operators. We will look at a simplified version of the process first to get the basic idea, then look at a more realistic case that will reveal the full algorithm. Finally, there will be a messy real world example to ponder.

2.6.1.3 A Simplified Example

Figure 2-32 is our simplified greenscreen test plate with a gray circle on a green backing. We will refer to the green part of the plate as the backing region, as it contains the green backing color, and the gray circle as the foreground region, as it is the "foreground object" that we wish to matte out for compositing. Note that the edges are soft (blurry). This is an important detail, because in the real world, all edges are actually soft to some degree, and our methods must work well with soft edges. At video resolution, a sharp edge may be only one pixel wide, but at feature film resolutions, "sharp" edges are typically 3 to 5 pixels wide. Then, of course, there are actual

soft edges in film and video such as with motion blur and depth of field (out of focus), plus partial coverage such as fine hair and semi-transparent elements. It is these soft edges and transitions that cause other matte extraction procedures to fail. In fact, it is the color difference matte's superior ability to cope with these soft transitions and partial transparencies that makes it so important.

Figure 2-32 Gray circle on green backing

Figure 2-33 is a close-up of the greenscreen plate in Figure 2-32 with a slice line across it, which has been plotted in Figure 2-34. The slice line in Figure 2-33 starts at the left on the gray circle, crosses the edge transition, then over a piece of the greenscreen. The slice graph in Figure 2-34 graphs the pixel values under the slice line from left to right also. Reading the slice graph from left to right, it first shows that on the gray circle the pixel values are 0.5 0.5 0.5 (a neutral gray), so all three lines are on top of each other. Next, the graph transitions towards the greenscreen backing colors, so the RGB values can be seen to diverge. Finally, the samples across the green backing region show that backing color has RGB values of 0.4 0.6 0.3. So the green backing shows a high green value (0.6), a lower red (0.4) value, and a somewhat lower blue value (0.3), just what we would expect from a saturated green.

Figure 2-33 Close-up of slice line on edge transition

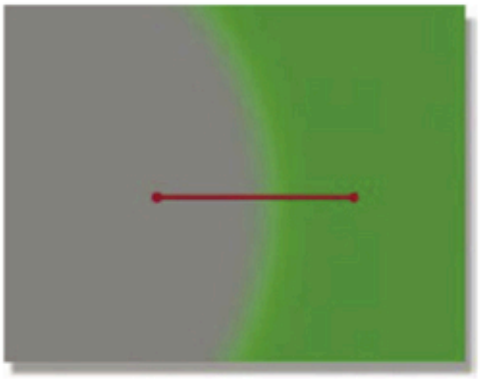
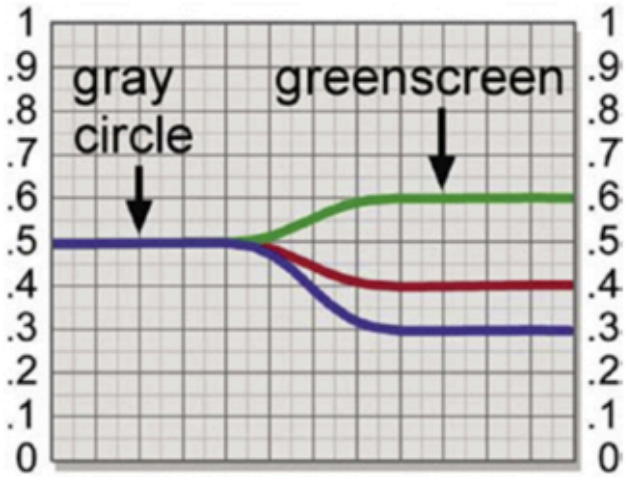


Figure 2-34 Slice graph of gray circle edge transition



To make our first simplified color difference matte, all we have to do is subtract the red channel from the green channel. Examining the slice graph in Figure 2-34, we can see that in the foreground region (the gray circle) the RGB values are all the same, so green minus red will equal 0, or black. However, in the green backing color region the green record has a value of 0.6 and the red record has a value of 0.4. In this region when we subtract green minus red we will get 0.6 – 0.4 = 0.2, not 0. In other words, we get a 0 black for the foreground region and a density of 0.2, a dark gray, for the backing region – the crude beginnings of a simple color difference

matte. Expressed mathematically, the raw matte density is

$$\text{raw matte} = G - R \tag{2-6}$$

which reads as "the raw matte density equals green minus red." The term "density" is used here instead of "brightness" simply because we are talking about the density of a matte instead of the brightness of an image. For a matte, the item of interest is its density, or how transparent it is. The exact same pixel value dropped into an RGB channel would suddenly be referred to as brightness.

The raw color difference matte that results from the green minus red math operation is shown in Figure 2-35 with another slice line. We are calling this the "raw" matte, because it is the raw matte density produced by the initial color difference operation and needs a scaling operation before becoming the final matte. In the green backing region, subtracting red from green gave a result of 0.2, which appears as the dark gray region. In the foreground region the same operation gave a result of 0, which is the black circle in the center. We get two different results from one equation because of the different pixel values in each region. In Figure 2-36, the slice graph of the raw matte shows the pixel values where the red slice line is drawn across the raw matte. The raw matte density is 0.2 in the backing region, 0 in the foreground region, and a nice gradual transition between the two regions. This gradual transition is the soft-edge matte characteristic that makes the color difference matte so effective.

Figure 2-35 Raw color difference matte with slice line

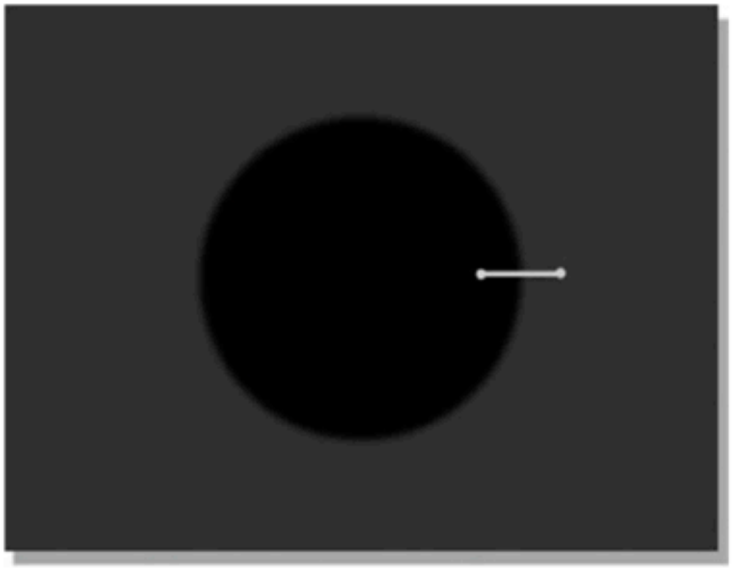
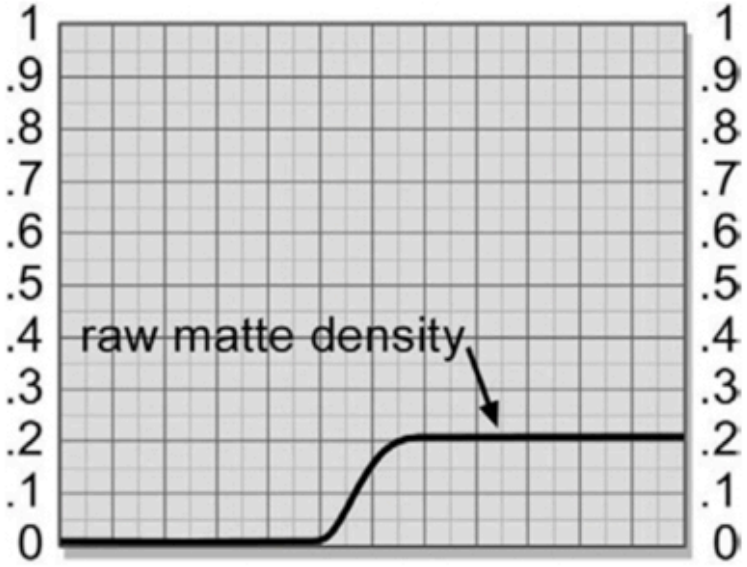
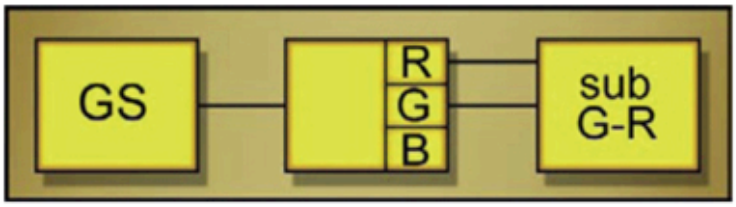


Figure 2-36 Slice graph of raw color difference matte



Ex. 2-7 Figure 2-37 is a flowgraph of the simple G–R raw color difference matte operation. From the greenscreen plate (GS), the red and green channels are connected to a subtract node. The output of the subtract node is the one-channel raw color difference matte in Figure 2-35.

Figure 2-37 Flowgraph of simple raw matte



2.6.1.4 A Slightly More Realistic Case

Real images obviously have more complex colors than just gray, so let's examine a slightly more realistic case of pulling a matte with a more complex foreground color. Our next test plate will be Figure 2-38, which is a blue circle on a greenscreen backing with a slice graph next to it in Figure 2-39. As you can see by inspecting the slice graph, the foreground element (the blue circle) has an RGB value of 0.4 0.5 0.6, and the same green backing RGB values we had before (0.4 0.6 0.3). Clearly the simple trick of subtracting the red channel from the green channel won't work in this case, because in the foreground region (the blue circle), the green record is *greater* than the red record. For the foreground region, the simple rule of green minus red (0.5 – 0.4) will result in a matte density of 0.1, a far cry from the desired 0 black.

Figure 2-38 Blue circle with slice line

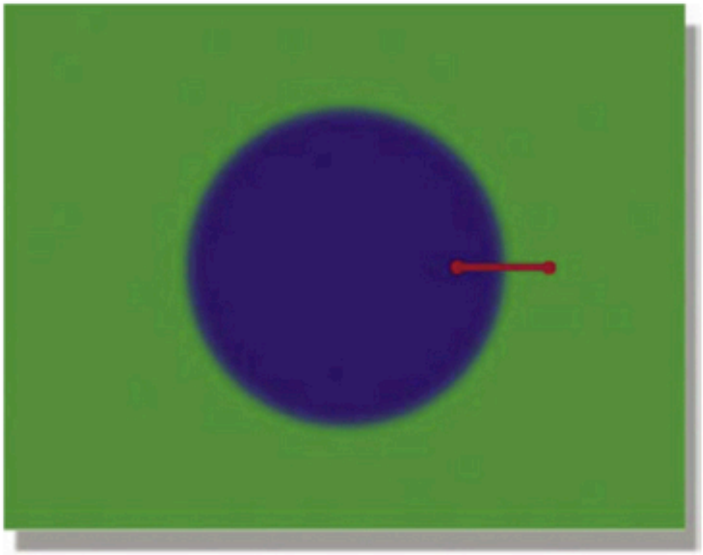
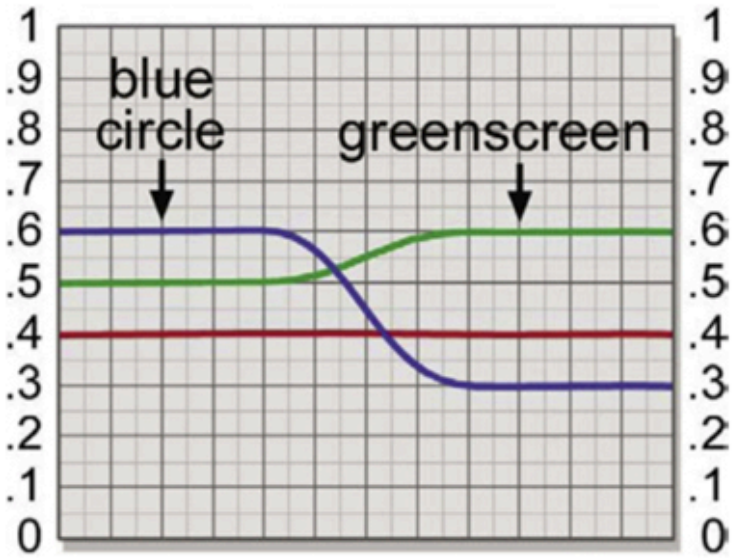


Figure 2-39 Slice graph of blue circle



If anything, from this example, we should subtract the blue record from the green record. In the foreground region, green minus blue (0.5 – 0.6) will result in –0.1, a negative number. As there are no negative pixel values allowed, this clips to 0 black in the foreground region, which is good. In the backing region, green minus blue (0.6 – 0.3) gives us a nice raw matte density of 0.3, even better than our first simple example. The problem with simply switching the rule from green minus red to green minus blue is that in the real world the foreground colors are scattered all over the color space. In one area of the foreground, red may be dominant, but in another region, blue may be dominant. One simple rule might work in one region but not in another. What is needed is a matte extraction rule that changes and *adapts* to whatever pixel value the foreground regions happens to have.

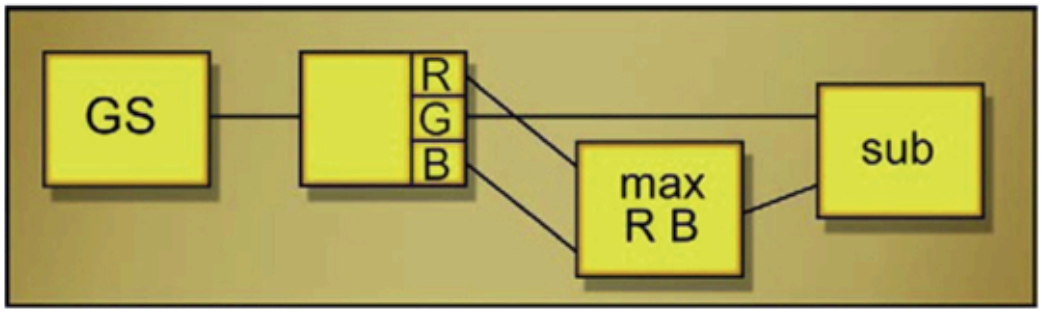
Here's a rule that adapts to those elusive scattered pixel values – the green channel minus *whichever is greater*, red or blue. Or, put in a more succinct mathematical form:

$$\text{raw matte} = G - \max(R, B) \qquad (2\text{-}7)$$

which reads as "the raw matte density equals green minus the maximum of red or blue." If we inspect the slice graph in Figure 2-39 with this rule in mind, we can see that it will switch on the fly as we move from the foreground region to the green backing region. In the foreground (blue circle) region, blue is greater than red (max(R, B)), so the raw matte will be green minus blue (0.5 – 0.6), which results in –0.1, which clips to 0 black. So far, so good – we get a nice solid black for our foreground region. Following the slice graph lines in Figure 2-39 further to the right, we see the blue pixel values dive under the red pixel values in the edge transition region, so the maximum color channel switches to the red channel as it transitions to the backing region. Here the raw matte density becomes green minus red (0.6 – 0.4), resulting in a density of 0.2. There you have it: an adaptive matte extraction algorithm that switches behavior depending on what the pixel values are.

Ex. 2-8 Figure 2-40 shows the new and improved flowgraph that would create the more sophisticated raw color difference matte. The red and blue channels go to a maximum node that selects whichever channel is greater on a pixel-by-pixel basis. The subtract node now subtracts this maximum(R,B) pixel value from the green channel, which creates the raw matte. It is called the raw matte at this stage, because it is not ready to use yet. Its density, currently only 0.2, needs to be scaled up to full opacity (white, or 1.0) before it can be used in a composite, but we are not ready for that just yet. This procedure can easily be implemented in Adobe Photoshop, providing that you know that they call the maximum operation "lighten" and the minimum operation "darken."

Figure 2-40 Flowgraph of raw color difference matte



Adobe Photoshop Users – in Photoshop the maximum operation is known as the "Lighten" blending mode. You can pull an excellent color difference matte in Photoshop just like a compositing program. Exercise 2-9 will show you how.

2.6.1.5 And Now, The Real World

When we pick up a real live-action greenscreen image such as Figure 2-41 and graph a slice across the foreground to the greenscreen (Figure 2-42), we can immediately see some of the differences between real live action and our simple test graphics. First of all, we now have grain, which has the effect of degrading the edges of the matte we are trying to pull. Second, the greenscreen is unevenly lit, so the color difference matte will not be uniform across the entire backing area. Third, the foreground nonbacking colors (red and blue) cross the green backing color at different values in different locations at the edges, which results in varying widths of the transition region (edges). Fourth, the green record in the foreground object can in fact be slightly greater than the other two colors in some areas, resulting in contamination in the foreground region of the matte. Each of these issues must ultimately be dealt with to create a high-quality matte.

Figure 2-41 Real world picture

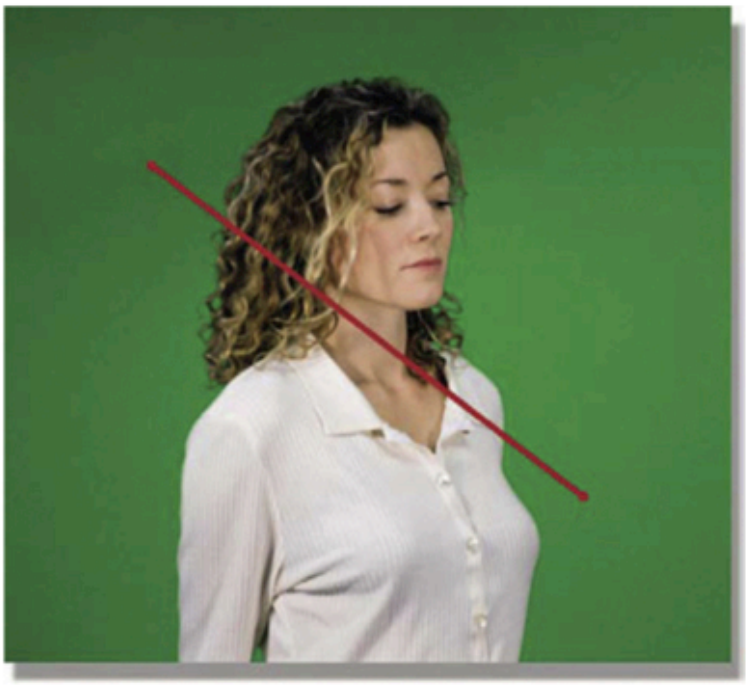
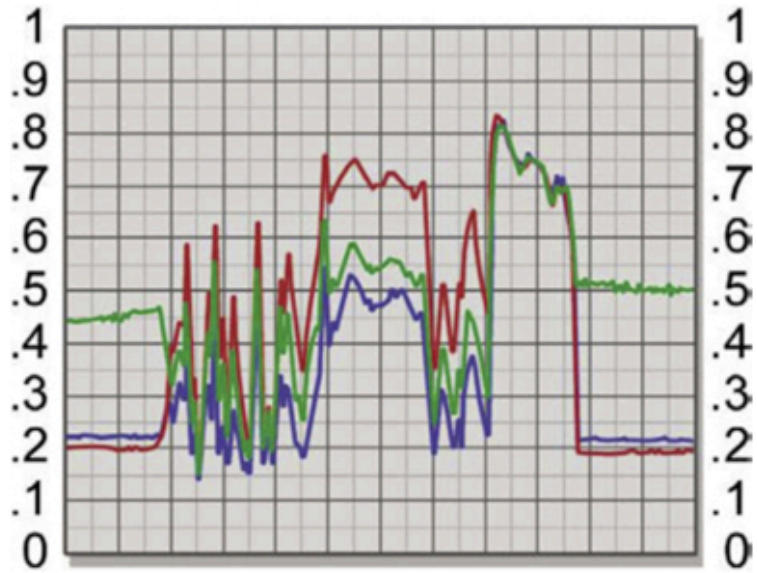


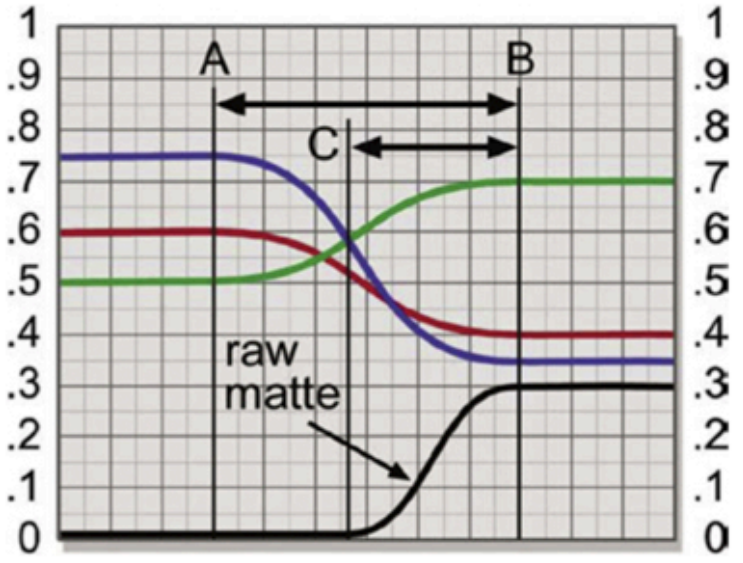
Figure 2-42 Graph of real world picture



2.6.1.6 Matte Edge Penetration

There is another real world issue to be aware of, and that is that the matte normally does not penetrate as deeply into the foreground as it theoretically should. Figure 2-43 represents a slice graph taken across the boundary between the foreground and the backing region of a greenscreen – foreground on the left, green backing on the right, with the resulting raw matte included at the bottom. The two vertical lines marked A and B mark the true full transition region between the foreground and background. The foreground values begin their roll off to become the greenscreen values at lineA and don't become the full backing color until they get to line B.

Figure 2-43 Matte edge shortfall



Theoretically, the new background we are going to composite this element over will also start blending in at line and transition to 100% background by line B. Obviously, the matte should also start at line A and end at line B. But as you can see, the raw matte does not start until much later – at line C.

The reason for this shortfall is readily apparent by inspecting the RGB values along the slice graph and reminding ourselves of the raw matte equation. Because it subtracts green from the maximum of red or blue, and green is less than both of them until it gets to line C, the raw matte density will be 0 all the way to line C. How much shortfall there is depends on the specific RGB values of the foreground and backing colors. When the foreground colors are a neutral gray, the matte does in fact penetrate all the way in and has no shortfall at all. In fact, the less saturated (more gray) a given fore-ground color is, the deeper the matte will penetrate. You might be able to figure out why this is so by inspecting the slice graph in Figure 2-33, which shows a gray foreground blending into a green backing. Just step across the graph from left to right and figure the raw matte density at a few key points and see what you get.

However, for foreground RGB values other than neutral grays, there is always a matte edge shortfall like the one in Figure 2-43, and it can easily be the true width of the transition region, sometimes even more. Further, the amount of shortfall varies all around the edge of the foreground element depending on the local RGB values that blend into the green backing. In other words, line C above shifts back and forth across the transition region depending on the local pixel values, and is everywhere wrong, except for any neutral grays in the foreground.

Ex. 2-10 The good news is that it usually does not create a serious problem (note the use of the evasive words "usually" and "serious"). The color difference matte process is very "fault-tolerant." Later, you will see how to filter the raw matte, and one of the things this filtering does is to pull the inner matte edge deeper into the foreground region to help compensate for the shortfall. In other sections, you will also see how to move this transition edge in and out, which will solve many problems.

2.6.2 Scaling the Raw Matte

When the raw matte is initially extracted, it will not have anywhere near enough density for a final matte. The criteria for the final matte is to get the backing region to 100% white and the foreground region to zero black. This is done by scaling the density of the raw matte. We have to scale the raw matte density in two directions at once – both up and down. The basic idea is to use the color curve tool to pull-up on the whites to full transparency and down on the blacks to full opacity. It is very common to have to scale some dark-gray pixels out of the black foreground region to clear it out, as there will often be some colors in the foreground where the green channel is slightly greater than both of the other two. Let's see why this is so.

Figure 2-44 illustrates a greenscreen plate with two foreground contamination regions, A and B. Area A has a green highlight in the hair and there is some green light contamination on the shoulder at area B. The rise in the green record in these two regions can be seen in the slice graph in Figure 2-45. Because the green record is noticeably greater than *both* of the other two records in these areas, it will leave a little raw matte density "residue" in the foreground region that we do not want. This residual density contamination is clearly visible in the resulting raw matte shown here in Figure 2-46. The gray residues in the foreground area are regions of partial transparency and will put "holes" in our foreground at composite time. They must, therefore, be terminated with prejudice.

Figure 2-44 Problem greenscreen plate

Figure 2-45 Slice graph over problem areas

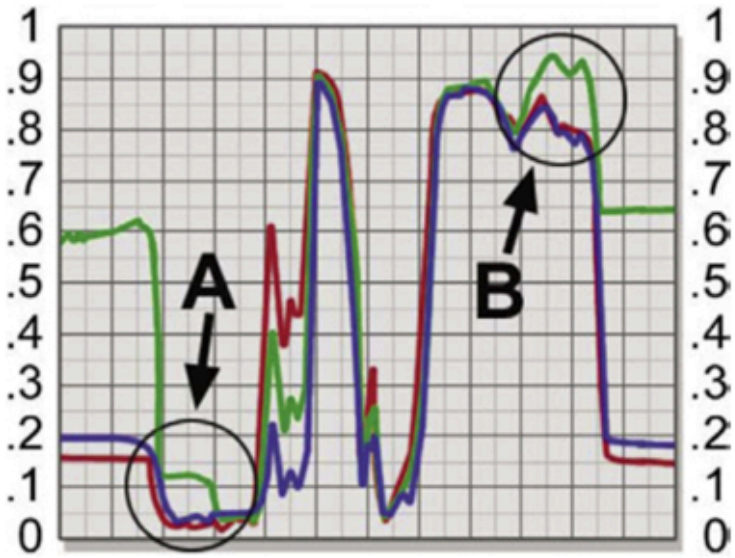


Figure 2-46 Raw matte

The first step is to scale the raw matte density up to full transparency, which is a density of 1.0 (100% white). Again, the matte can be inverted later if needed. Figure 2-46, the contaminated raw matte extracted from the problem greenscreen in Figure 2-44, shows the raw matte starting point. Move the "white end" (top) of the color curve graph to the left as shown in Figure 2-47 a bit at a time, and the raw matte density will get progressively more dense. Stop just as the density gets to a solid 1.0, as shown in Figure 2-48. Be sure to pull-up on the whites only as much as necessary *and no more* to get a solid white, because this operation hardens the matte edges, which we want to keep to an absolute minimum. In the process of scaling up the whites, the nonzero black contamination pixels in the foreground will also have gotten a bit denser, which we will deal with in a moment. We now have the backing region matte density at 100% white, and are ready to set the foreground region to zero black.

Figure 2-47 Scaling whites up

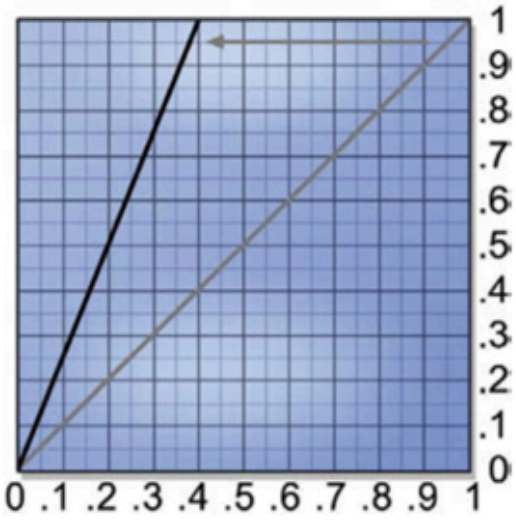


Figure 2-48 White scaled matte

To set the foreground region to complete opacity, slide the "black end" (bottom) of the color curve to the right as shown in Figure 2-49, so it will "pull-down" on the blacks. This scales the foreground contamination in the blacks down to zero and out of the picture like in Figure 2-50. This operation is also done a little at a time to find the *minimum* amount of scaling required to clear out all the pixels, because it too hardens the edges of the matte. As you pull-down on the blacks, you may find that a few of the white pixels in the backing region have darkened a bit. Go back and touch up the whites just a bit. Bounce between the whites and blacks, making small adjustments, until you have a solid matte with the least amount of scaling.

Figure 2-49 Color curve moved to scale down blacks

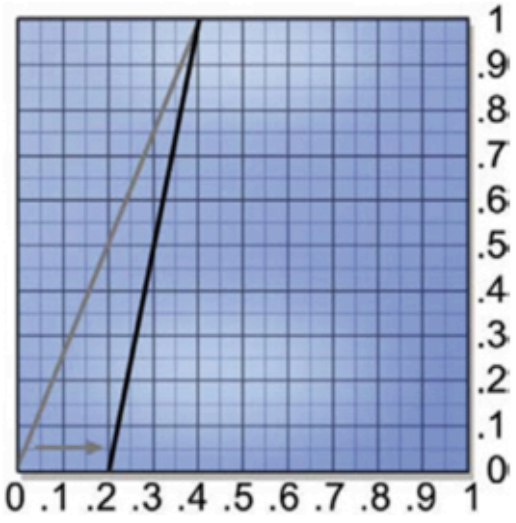
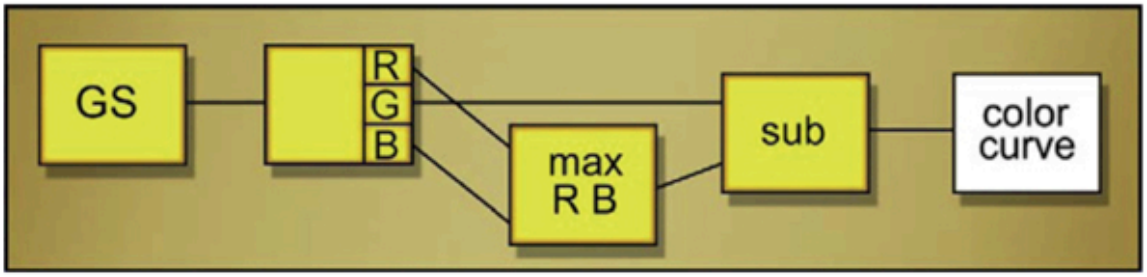


Figure 2-50 Blacks scaled down

It is normal to have to scale the whites up a lot, because we are starting with raw matte densities of perhaps 0.2 to 0.5, which will have to be scaled up by a factor of 2 to 5 in order to make a solid white matte. The blacks, however, should only need to be scaled down a little to clear the foreground to opaque. If the raw matte's foreground region has a lot of contamination in it, then it will require a lot of scaling to pull the blacks down, which is bad, because it hardens the edges too much. Such extreme cases will require heroic efforts to suppress the foreground pixels to avoid severe scaling of the entire matte.

Ex. 2-11 Figure 2-51 shows a new flowgraph with a color curve node for the scaling operation added to our growing sequence of operations.

Figure 2-51 Flowgraph with color curve scaling operation added



2.6.3 Refining the Color Difference Matte

So far, you've seen the basic technique for extracting or "pulling" a color difference matte, but when used in the real world on actual bluescreens, all kinds of problems will surface in both the foreground and backing regions, not to mention the edges. In this section, you will see how to refine the basic color difference matte to control the ever-present artifacts and finish with the best possible matte ready for compositing.

2.6.3.1 Preprocessing the Greenscreen

As a broad strategy, you can almost always improve your raw matte results with some kind of preprocessing of the greenscreen to "set it up" for a better color difference matte. By the by, all of these preprocessing techniques can also be used to help Ultimatte, as it is basically a color difference matte technique, too. What works for your own color difference mattes will work for Ultimatte, too. The basic idea is to perform some operations on the greenscreen plate *before* the matte is pulled that improves either the raw matte density in the backing region, clears holes in the foreground region, improves edge characteristics, or all of the above. There are a number of these operations, and which ones to use depends on what's wrong.

Figure 2-52 Preprocessed greenscreen

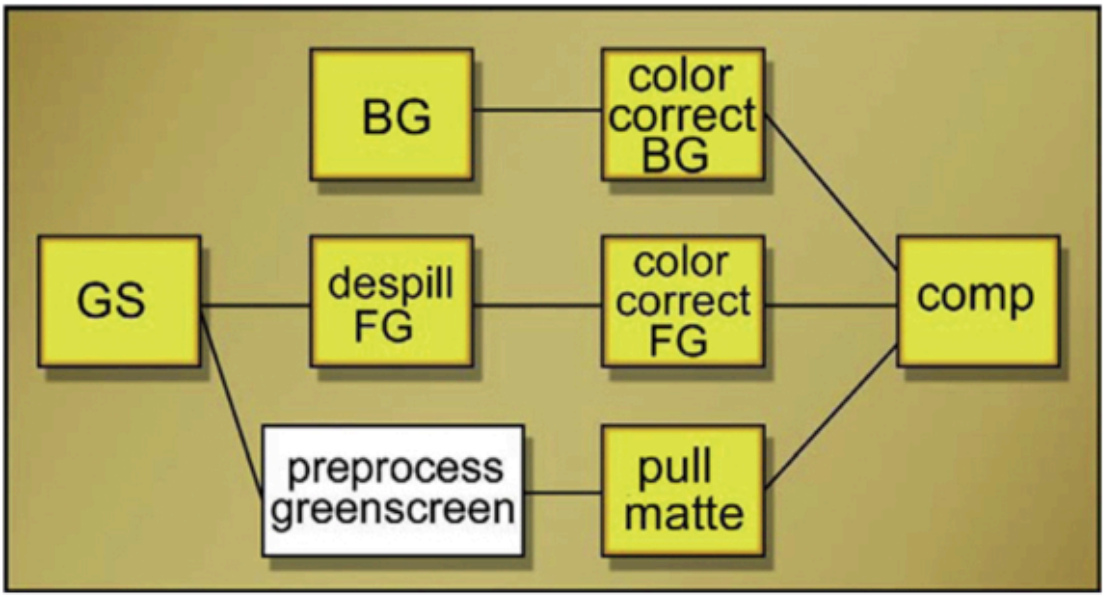


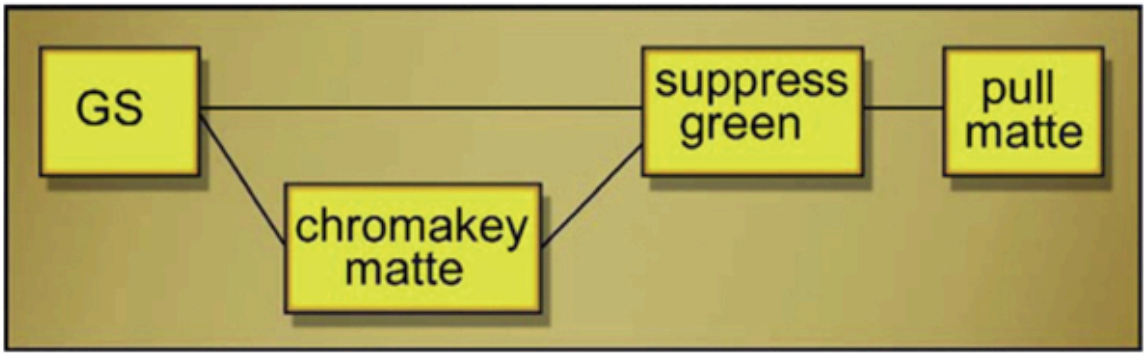
Figure 2-52 shows a flowgraph of how the preprocessing operation fits into the overall sequence of operations. The key point here is that the preprocessed greenscreen is a "mutant" version kept off to the side for the sole purpose of pulling a better matte, but not to be used in the actual composite. The original untouched greenscreen is used for the actual foreground despill, color-correction, and final composite. In this section, we will explore several preprocessing operations that can help in different circumstances. Which techniques will work on a given shot depends entirely on the actual colors present in the greenscreen plate. These, then, are some of the "heroic" rescue techniques alluded to at the beginning of this chapter. You can use one or all of them on a given greenscreen.

2.6.3.2 Local Suppression

As we know, the problem areas in the foreground region are caused by the green record climbing above the other two records in a local region. As another overarching strategy, we would like to avoid "global" solutions to the problem (such as scaling the blacks down harder) because they effect the entire matte, which can introduce problems in other areas. A surgical approach that just effects the problem area is much preferred. Ideally, we would just push the green record down a bit in the problem area, solving the problem without disturbing any other region of the greenscreen plate. This is sometimes both possible and easy to do.

Use a chroma-key tool to create a mask from the original greenscreen plate that covers the problem area. Use this chroma-key mask and a color adjusting tool to lower the green record under the mask just enough to clear (or reduce to acceptable) the problem area from the foreground. Perhaps the subject has green eyes, which will obviously have a green component higher than the red and blue in that area. Make a chroma-key mask for the eyes and suppress the green a bit. Perhaps there is a dark green kick on some shiny black shoes that also punches holes on the foreground region of the matte. Mask and suppress. A flowgraph of the general sequence of operations is shown in Figure 2-53. If the target item is too close to the backing in color, some of the backing may show up in the chroma-key matte. A traveling garbage matte may be required to isolate the target item from the backing.

Figure 2-53 Flowgraph of chroma key used for local suppression



2.6.3.3 Channel Clamping

Let's explore a more sophisticated solution to the problem greenscreen in Figure 2-44, which has the green highlight in the hair and some green spill on the shoulder. Re-examining the slice graph in Figure 2-45, the graph area marked A shows the green record rising about 0.08 above the red record, which is the result of the green highlight in the hair. The graph area marked B shows the green record has risen above the red channel due to the green spill on the shoulder. It has reached a value as high as 0.84; the average value of the backing green channel is down near 0.5. What we would like to do is shift the color channels so that the green record is at or below the red record in these two regions without disturbing any other parts of the picture. We can either move the green record down or the red record up. In this case, we can do both.

Referring to the color curve in Figure 2-54, the red curve has been adjusted to clamp the red channel *up* to 0.13 so no red pixel can fall below this value. As a result, the red channel in the hair highlight in area A has been "lifted" up to be equal to the green record, clearing the contamination out of this area of the raw matte. The resulting slice graph in Figure 2-55 shows the new relationship between the red and green channels for area A. Similarly, the problem in area B has been fixed by the same color curve in Figure 2-54 by putting a clamp on the green record that limits it to a value of 0.75. This has pulled it down below the red record, which will clear this area of contamination.

Figure 2-54 Color curve used to clamp channels

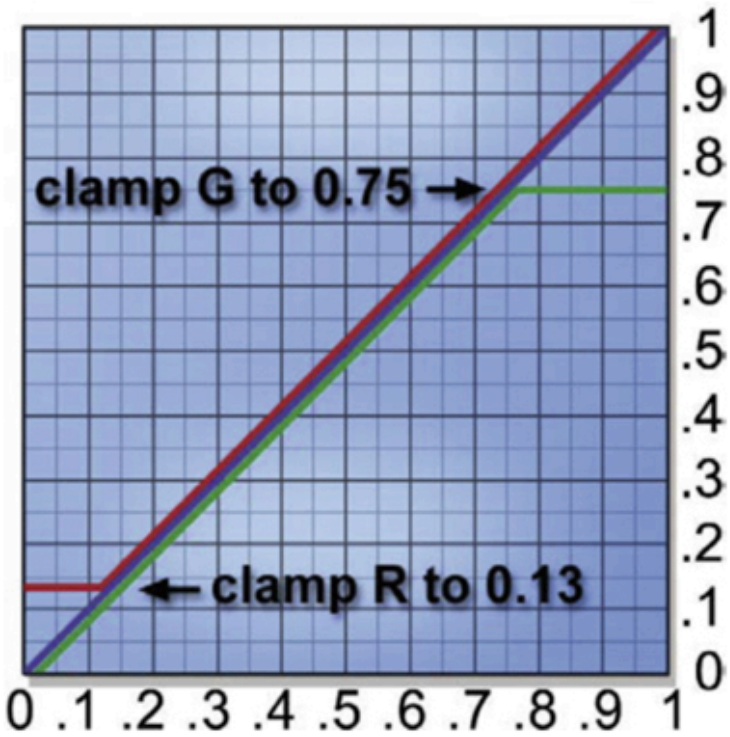
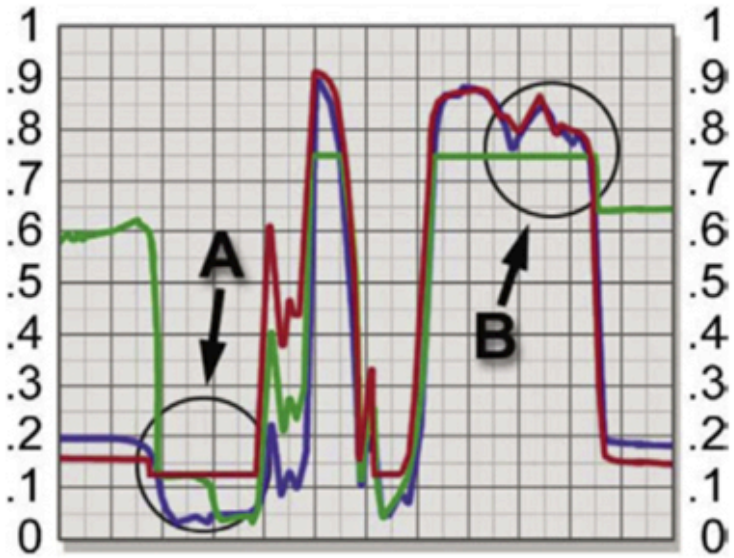


Figure 2-55 Red and green channels clamped by color curve



Again, we just need to get the green record below (or nearly below) either the red *or* blue records to clear the foreground matte. This means either lowering the green record or raising one of the other two. Either one will work. The key is to shift things as little as possible, then check the raw matte for the introduction of any new problems in other areas. Keep in mind that there could be other parts of the picture that also get caught in this "net" that could introduce other problems. Make small changes then check the raw matte.

2.6.3.4 Channel Shifting

Here is a nifty technique that you will sometimes be able to use – again, depending on the actual color content of the greenscreen. <T>By simply raising or "lifting" the entire green record, you can <T>increase the raw matte density, resulting in a better matte with nicer edges. Starting with the slice graph in Figure 2-56, the difference between the green channel and the maximum of the red and blue channels in the backing region is about 0.3. We can also see that the green record is well below both the red and blue records in the foreground region. If the green record were raised up by 0.1 as shown in Figure 2-57 it would increase the raw matte density from 0.3 to 0.4, a whopping 33% increase in raw matte density! As long as it is not raised up so much that it climbs above the red or blue records, no holes in the foreground will be introduced. Of course, in the real world, you will rarely be able to raise the green record this much, but every little bit helps.

Figure 2-56 Original raw matte density

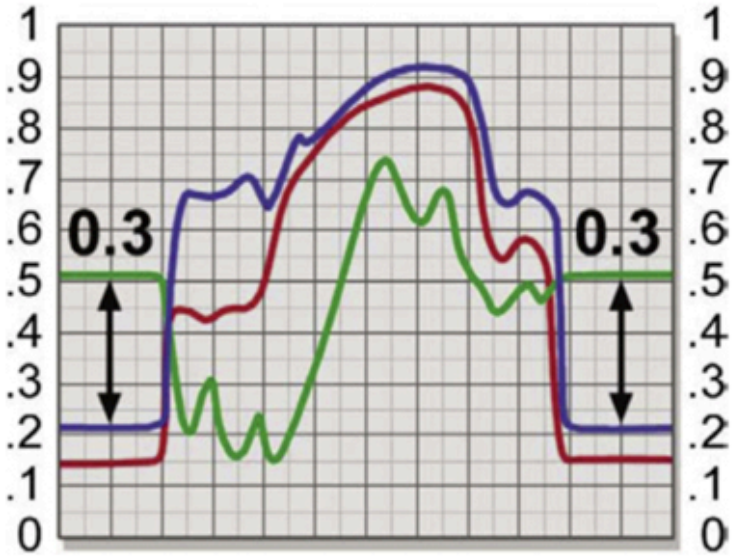
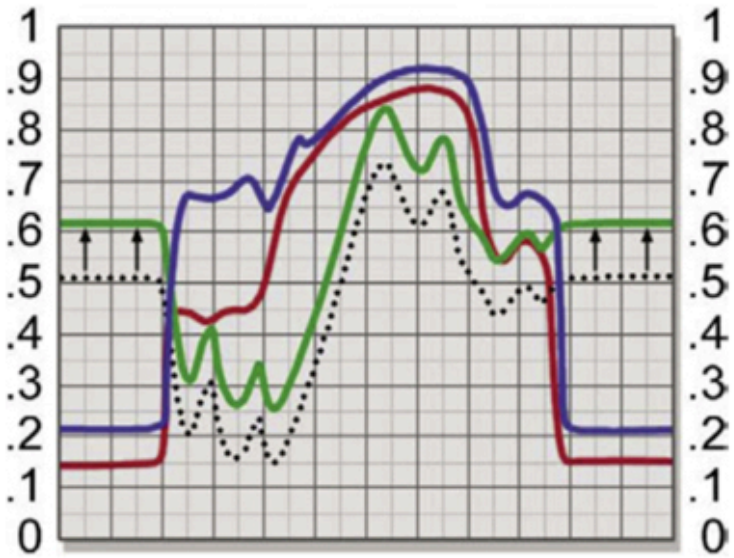


Figure 2-57 Green channel shifted to increase raw matte density



The procedure is to raise the green record a bit by adding a small value (0.1 in this example) to the green channel, then pull the raw matte again to look for any new contamination in the foreground. If all seems well, raise the green record a bit more and check again. Conversely, you can try lowering the red and blue records by subtracting constant values from them. These techniques can sometimes solve big problems in the matte extraction process, but sometimes at the expense of introducing other problems. The key is an intelligent trade-off that solves a big, tough problem while only introducing a small, soluble problem. For example, if only a few holes are introduced you can always pound them back down with local suppression techniques or perhaps with a few quick rotos.

In addition to adding or subtracting constants from the individual records to push the channels apart for better raw mattes, you can also scale the individual channels with the color curve tool. It is perfectly legal to have "creative color curves" to push the color records around in selected regions to get the best matte possible. Always keep in mind that the color difference matte extraction works on the simple *difference* between the green record and the other two records, so raising or scaling up the green can increase the difference, but so can scaling down or lowering the red or blue records. Push them around, see what happens.

2.6.3.5 Degraining

All film has grain. Even video shot with a video camera has "video noise" that acts like grain. When film is transferred to video in telecine there will still be grain. It is important to know how the grain will effect the matte extraction process and what to do about it to reduce its impact on your results. While the grain degrades all matte extraction processes, its most important impact is in bluescreen matte extractions, because that is normally the most frequent and demanding scenario.

Figure 2-58 is a close-up of a greenscreen plate showing the grain with a slice line crossing the transition region between the foreground and backing. The grain "noise" can easily be seen in its slice graph in Figure 2-59. Regardless of the matte extraction method, the first thing that is generated by a keyer or your own matte extraction procedure is the raw matte. This is the initial color difference density before it is scaled up to full density for use, and the grain has a big impact at this point.

Figure 2-58 Close-up of greenscreen grain

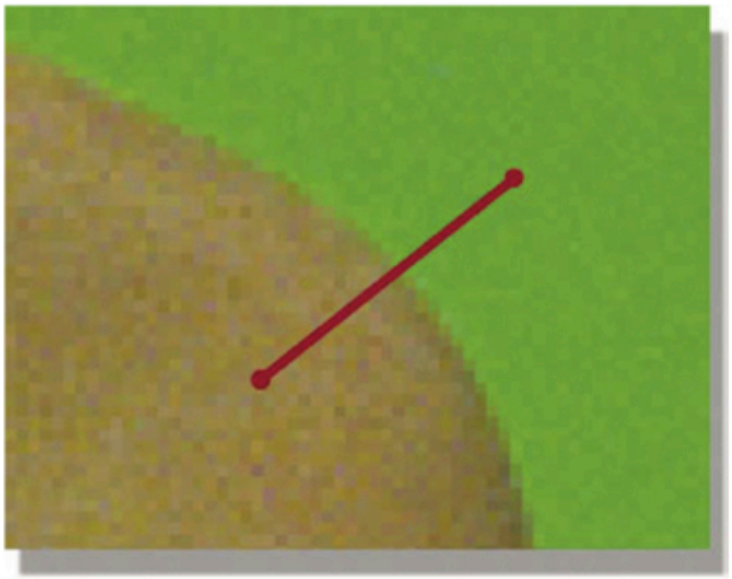


Figure 2-59 Slice graph of greenscreen noise

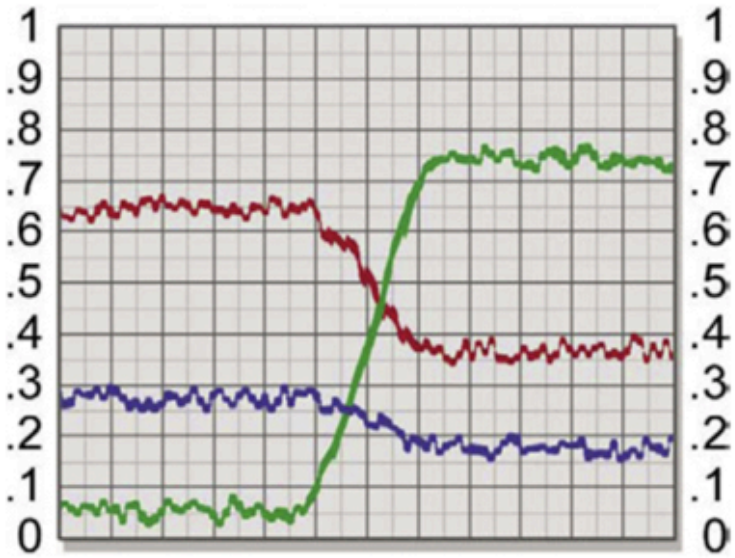
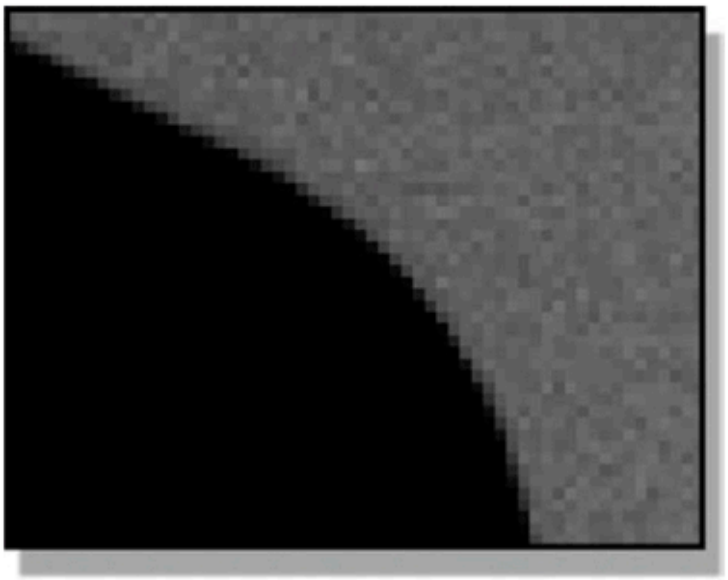


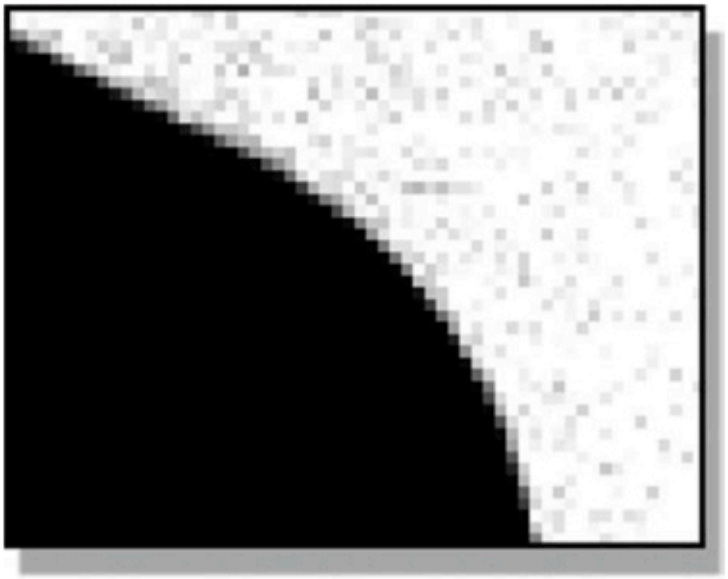
Figure 2-60 represents the raw matte created from the greenscreen closeup in Figure 2-58. As you can see, the grain from the film has become "embossed" into the raw matte as "noise." I am calling it "noise," because it is no longer really grain, but the interaction between the grains of two or more color records of the film. The good news is that the resulting raw matte "grain noise" is no greater than the noise (grain) in any one of the three color channels. In other words, the grain does not build up or accumulate in any way. Anywhere in a matte extraction process that the blue channel in incorporated in its calculations will end up embossing the blue channel's grain into the raw matte, and the blue channel has *much* more grain than the green or red (which are similar to each other). The blue grain is both larger in diameter and greater in amplitude than the red or green grains.

Figure 2-60 Raw matte



Another issue is that the raw matte's density must be scaled up from the typical value of 0.2 or 0.3 to a full density value of 1.0, which is a scale factor of 3 to 5. This will obviously also scale up the matte noise by a factor of 3 to 5. In addition to that, if the raw matte density were an average of, say, 0.2, then we would scale it up by a factor of 5 to get a density of 1.0. But the 0.2 was an *average* value, so when the matte is scaled up some of the pixels don't make it to 100% white. This leaves some "salt and pepper" contamination in the backing region, as shown in the scaled matte in Figure 2-61.

Figure 2-61 Scaled matte



We are therefore forced to scale the brightness up by *more* than the original factor of 5 to get all the noise out of the backing region. Scaling the matte up more hardens the edges, increases "edge sizzle" – chatter at the edges of the composite on moving pictures caused by this "grain noise" – and destroys edge detail. Making a degrained version of the frames to pull the matte will solve a great many of these problems.

If grain from the blue channel is causing most of the grain grief, then it is often very effective to degrain just the blue record before pulling the matte. If a quality degrain operation is not available, then either a median filter or a gentle blur will have to do. The key is that it is done on the blue record *only*. This takes advantage of these two important film facts:

Important Film Facts

1. Most of the grain noise comes from the blue channel.
2. Most of the image detail is in the red and green channels.

By taking advantage of this fortuitous juxtaposition, the most noise reduction can be achieved with the least loss of matte detail by smoothing just the blue channel.

Ex. 2-12 The blue record grain effects bluescreens and greenscreens differently. With bluescreens, it is the primary color of the matte, so all of the matte calculations are based on it. As a result, its excessive grain is embedded into the matte edges, making bluescreen mattes "sizzle." However, the blue-screen's despill operation are based on comparisons to the red and green channels, so the despill is not badly effected. With greenscreens the blue record is part of the color difference matte calculations only in certain areas. For those areas the blue record's grain will be present, but for the majority of the matte it is not a participant. As a result, the greenscreen matte will have "quieter" edges than the bluescreen. However, the despill for a greenscreen will reference the blue channel, and the heavier blue grain often becomes "embossed" into the red or green records. Of course, this invariably happens on smooth flesh tones on the face – just what you don't need.

2.6.4 Poorly Lit Greenscreens

The entire purpose of the DP (Director of Photography) is to ensure that each scene is properly lit and correctly exposed (oh, yes – fabulous composition and exciting camera angles, too!). The entire purpose of the greenscreen is to be evenly lit, correctly exposed, and the right color – you know – green! Not cyan, not chartreuse, but green. Somehow this all too often collapses on the set and you will be cheerfully handed poorly lit, off-color greenscreen shots and expected to pull gorgeous mattes and make beautiful composites from these abused plates. As digital compositors, we are victims of our own wizardry. Digital effects have become so wondrous of late that no matter how poorly it is shot on the set, the "civilians" (nondigital types) believe that we can fix anything. The problem is that all too often we can, and as a result we are encouraging bad habits on the part of the film crew.

The standard lecture to the client is: "Poorly lit greenscreens degrade the quality of the matte that can be extracted. Extra work will be required in an attempt to restore some of the lost matte quality, but this will both raise production costs and degrade the results." Unfortunately, you may never be given the chance to deliver this lecture or supervise the greenscreen shoot. All too often, the first time you will know anything about the shot is when the digitized film frames are handed to you, and then it's too late. Now you must cope. Poorly lit greenscreens are in fact both the most difficult and common problem. There are several different things that can be wrong with the lighting, so each type of problem will be examined in turn, and its effect on the resulting matte and potential workarounds discussed. But it is not a

pretty sight.

2.6.4.1 Too Bright

In film work, it is rare for the greenscreen to be too "hot" (bright), simply because it requires either a great amount of light (which is expensive) or overexposure due to excessive aperture (a rare mistake for all but the most inept DPs). The dynamic range of film is so great that it can take an enormous amount of light and still maintain separation between the color records. Unless, of course, you are *not* working with the full dynamic range of film – which, surprisingly, most studios are not.

When film is digitized to the Cineon 10-bit log format, the full dynamic range of the film is retained. If the Cineon log image is later converted to linear, then both the whites and the blacks get clipped. When film is digitized to 8 bits linear, either on a film scanner or a telecine, the picture is again clipped in both the whites and the blacks. See Chapter 14, "Log vs. Linear," for the full story on this fascinating topic. In the meantime, suffice it to say that because the picture is clipped in the whites, an over-bright greenscreen can become clipped.

Figure 2-62 represents a greenscreen with a "hot spot" in the middle that was overexposed with the green channel's brightness greater than the white clip point, not uncommon in film-to-video transfers. The green record got clipped in the flat region in the center. The slice graph in Figure 2-63 shows what's happening. While the red and blue records peaked around point A, the green record was clipped flat. This is lost data that cannot be restored. Most importantly, note the difference between the green and red records at point A in the clipped region, and point B in the normally exposed region. Clipping the green record has reduced the green–red color difference, so the raw matte values in this region will be significantly less than in the unclipped regions. Now the matte density will be thinner in the clipped region.

Figure 2-62 Clipped greenscreen

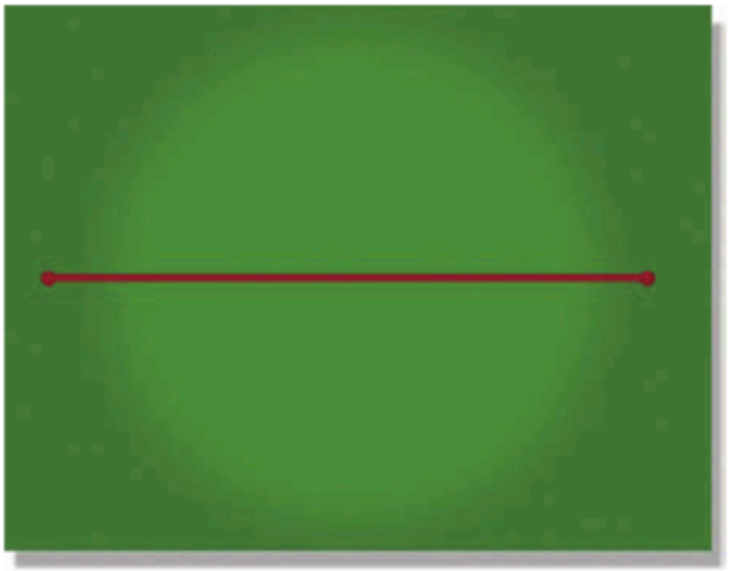
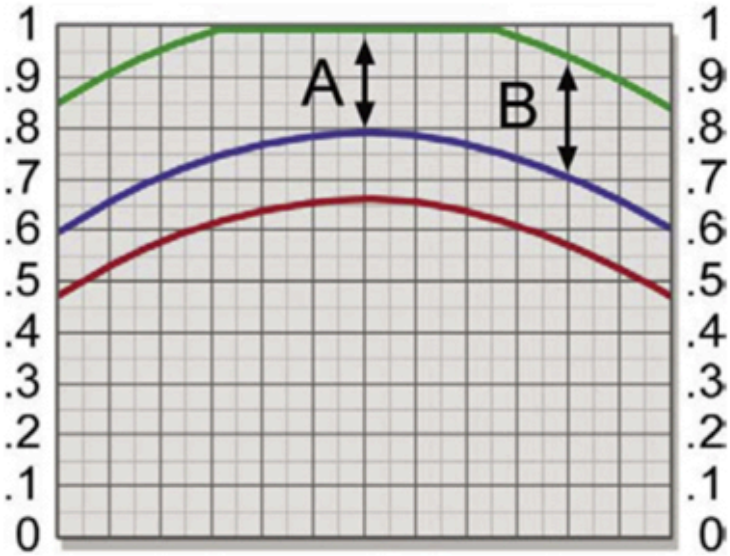


Figure 2-63 Slice graph of clipped greenscreen



What to do? The problem with this problem is that critical data has been lost: not compressed, abused, or relocated in some way that can be recovered, but gone. In the case of a Cineon log image converted to linear, the only real fix is to reconvert the original film scan, adjusting the white point to slightly above the brightest green value. This will darken the overall image somewhat, which will have to be color-corrected back later at composite time. But in a greenscreen shot, the matte's the thing, so we must have a full data set to pull a good matte. The rest can be fixed later. If the greenscreen plate was film transferred to video, then the telecine must be redone, if at all possible. If it was shot in video, then the original video is clipped and you are stuck with what you got.

Now here's a nifty trick for the film folks converting Cineon log images to linear, assuming the foreground object is not clipped along with the backing color. Keep the original clipped version to use as the foreground element in the composite, but pull the matte from the reconverted version. That way you keep the best of both. The folks working with film transferred to video with a telecine have a tougher problem. The film can be transferred to video a second time with color adjustments to lower the greenscreen values so they are no longer clipped, but unless both versions (the darker one to pull a good matte and the "overbright" one to use the foreground element for the composite) are transferred with a pin-registered telecine, the two sets of frames will have "gate weave" and will not stay registered to each other.

2.6.4.2 Too Dark

The greenscreen is too dark because it was underexposed. After vowing to purchase new batteries for the DP's light meter, we can examine the deleterious effects of underexposure on the raw color difference matte.

Figure 2-64 shows a greenscreen nicely exposed on the left and underexposed on the right with a slice line across the two regions. As you can see in the slice graph of Figure 2-65, on the correctly exposed (left) side of the greenscreen the color difference between the green and red channels will be a healthy 0.30 (0.7 – 0.4); on the underexposed side, the color difference will be an anemic 0.15 (0.35 – 0.2). In other words, the underexposed side has *half* the raw matte density of the properly exposed side! The reason is apparent from the slice graph. As the exposure gets lower, the RGB records not only get lower, but they also move *closer together*. Moving closer together reduces their difference, hence a lower raw matte density. As a result, the "thinner" raw matte will have to be scaled much harder to achieve full density.

Figure 2-64 Normal and underexposed greenscreen

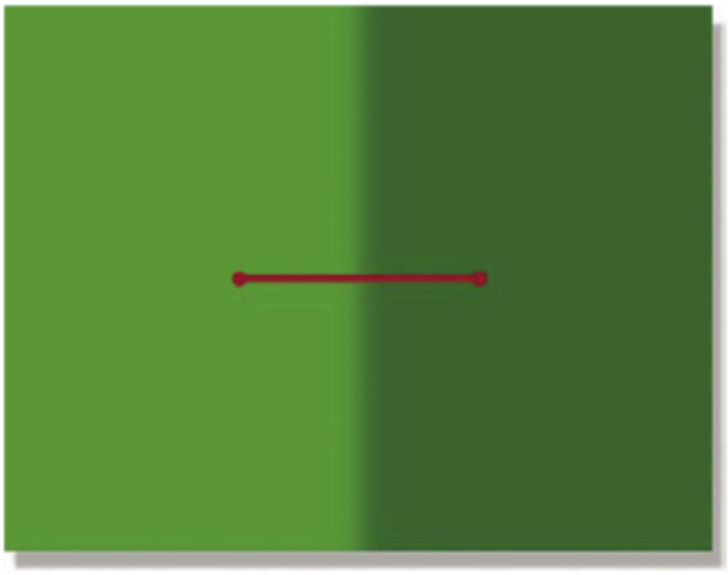
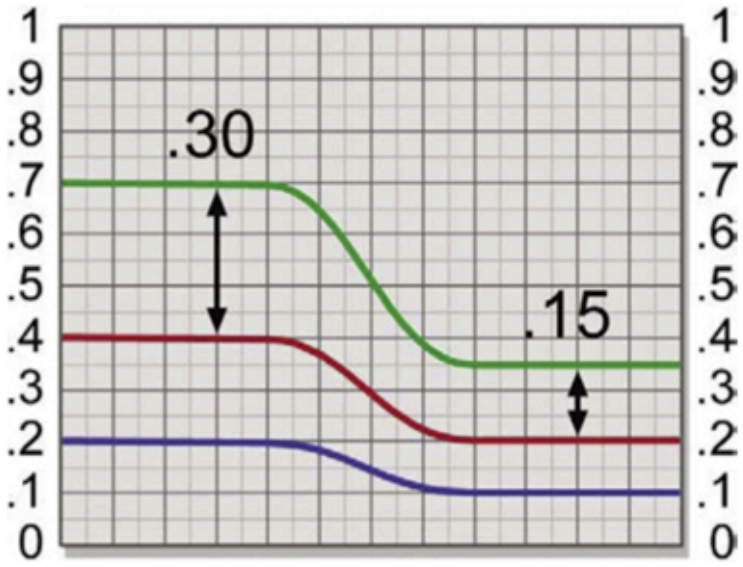


Figure 2-65 Slice graph of normal and underexposed greenscreen



Pausing for a lecture, this is a tempting situation to engage in what I call "pushing the problem around." What I mean by this is putting a fix on a problem that in fact only appears to help, but doesn't really. Here is an example of "pushing the problem around" in this case. Let's try solving our underexposed greenscreen problem by cleverly increasing the overall brightness of the greenscreen plate before pulling our raw matte. That is to say, we will scale the RGB values up by a factor of 2.0 – and voilà! The raw matte density jumped from 0.15 to 0.30, just like the properly exposed side. We can now scale the new and improved raw matte density by a modest factor of 3.3 to achieve our desired density of 1.0 and go home feeling quite clever. But are we? Let's take a closer look.

The original raw matte density was 0.15, which will need to be scaled up by a factor of 6.6 to achieve a full density of 1.0. But if we scaled the original RGB channels up by a factor of 2 to get an improved raw matte density of 0.3, then scaled that up by a factor of 3.3 to full density, we have still simply scaled the original density of 0.15 by a factor of 6.6 (2 * 3.3). We have wasted some time, some nodes, and had a fine exercise in the distributive law of multiplication, but we have not helped the situation at all. This is what I mean by "pushing the problem around," but not actually solving it. This is a chronic danger in our type of work, and understanding the principles behind what we are doing will help to avoid such fruitless gyrations. [end lecture]

Returning again to the actual problem, one thing is clear – we are going to get a pretty thin raw matte density. And this thin raw matte density will also have lots of grain in it, because the apparent grain increases in low exposure parts of the picture. We now have to scale up this grainier version of the raw matte much more than one from a properly exposed greenscreen. As a result, we get a very noisy matte with hard edges. The tragedy in this problem is that the data is simply not there for our raw matte. But wait – it's not there in the digital data you are trying to work with, but it *is* there in the film negative!

Yes! The film negative has *several* times the amount of information compared to the digital file from a video telecine or the linear conversion of the original log film scan. Like the overexposed greenscreens in the previous section, we can go back to the original log data file and convert it to linear again, this time lowering the white point and adjusting the gamma to get a brighter greenscreen. This "pumped" version can be used to pull the mattes, then discarded and the foreground element from the original scan used for the actual composite. Rest assured that we are not "pushing the problem around." We are actually getting more data to work with by resampling the original log scan and recovering previously discarded data. It was on the film negative and in the 10-bit log data all the time, but was discarded when converted from log to linear. The underexposed film will still be grainier than properly exposed film, however.

Again, the folks working with film transferred to video with a telecine have a tougher problem. As mentioned previously, the film can be telecined a second time with better color-correction, but only if it is a pin-registered transfer to prevent gate weave.

2.6.4.3 Impure Greenscreens

The entire idea of how a greenscreen works is based on a pure, saturated primary color. As we have seen from the math used to pull a color difference matte, to get the maximum raw matte density the green record must be as far above the red *and* blue records as possible. This is, by definition, a saturated color. However, in the real world, the red or blue records may not be very low values. Excessive red or blue records can cause the green to be "impure" and take on a yellow tint (from high red), a cyan tint (from high blue), or a faded look (from high red *and* blue). This lack of separation of the color records dramatically reduces the quality of the matte.

Figure 2-66 shows four side-by-side greenscreen strips, with Figure 2-67 plotting the matching slice graph across all four regions. Region A is a nice saturated green, and its slice graph shows a healthy color difference value (black arrow) that would make a fine raw matte density. Region B is an unsaturated greenscreen example, and its slice graph clearly shows high red and high blue values with a resulting low raw matte density. Region C is too high in red, and since the color difference equation subtracts green from the *maximum* of the other two channels, red becomes the maximum and the resulting raw matte density is also very low. Region D simply shifts the problem over to the blue channel with similarly poor results.

Figure 2-66 Impure greenscreens

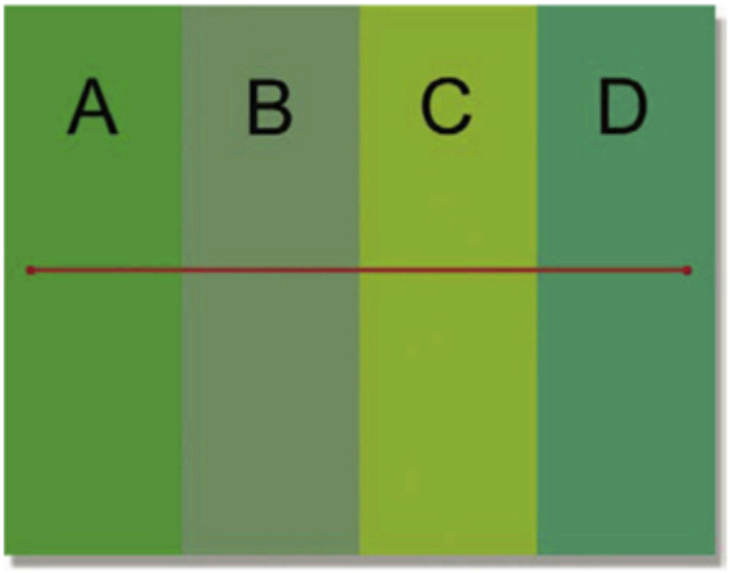
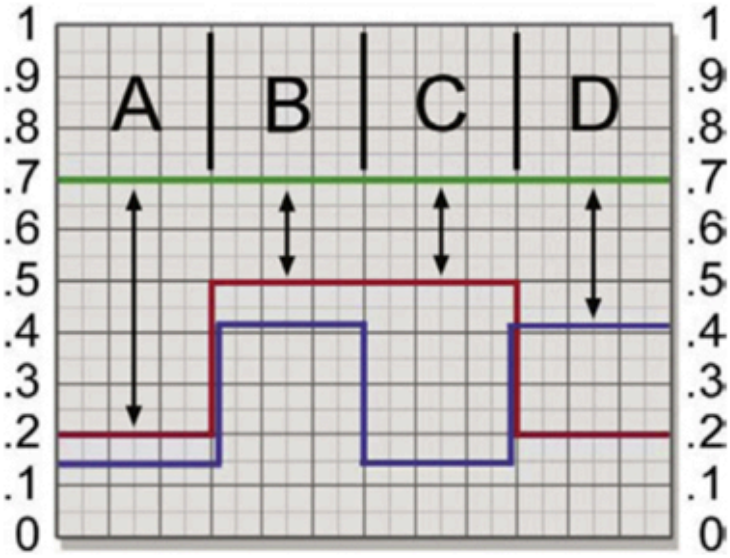


Figure 2-67 Slice graph of impure greenscreen



What can be done? Although the situation is dire, it is not hopeless. There is one possibility that may help – the channel shifting technique you learned earlier. Choosing the most offensive channel (let's say it's the red channel that is too high), you can start gently subtracting values from the red channel of the greenscreen before pulling the color difference matte. This will lower the red record overall, which will help the matte, but will also eventually start to introduce nonblack pixels into the foreground region if you go too far. Lower the red record a bit, then check the matte for foreground pixels. Lower it a bit more, check again. Some amount of nonzero pixels in the foreground is a common occurrence anyway, and is not the end of the world. Once you have found the limits of the red channel, repeat the process with the blue channel. With luck (and this depends on the actual RGB color content of the foreground) you will be able to add significantly to the overall raw matte density while only adding a little to the foreground black. If the foreground is contaminated in only one or two places, this is a candidate for some local suppression. When broad areas start to increase density, you have reached the limits of the process.

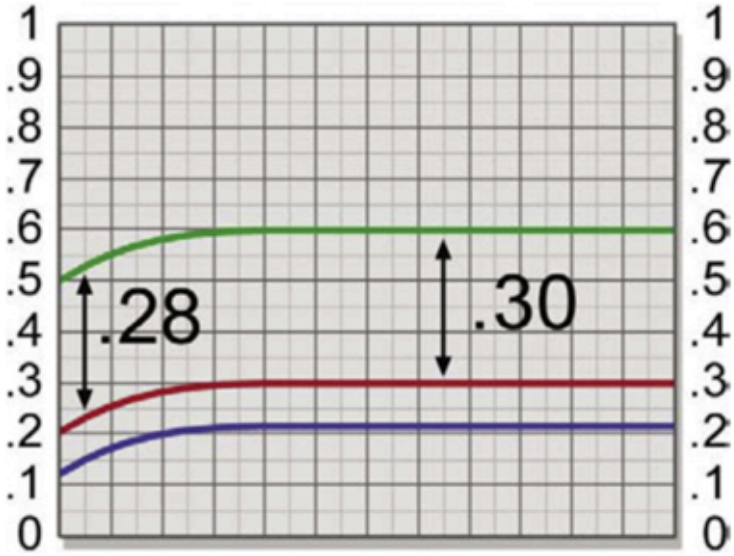
2.6.4.4 Uneven Lighting

This is both the most common and the most insidious of the bad greenscreen curses. The reason it is the most common is simply because it is very difficult to get the lighting both bright and uniform across a broad surface. The human eye is so darn adaptable that it makes everything look nice and uniformly lit, even when it is not. Film is not so accommodating. Standing on the set, looking at the greenscreen, it looks like a perfectly uniform brightness left to right, top to bottom. But when the film comes back from the lab, there is suddenly a hot spot in the center and the corners roll off to a dingy dark green, and now you have to pull a matte on this chromatic aberration. Let's see what might be done.

First of all, here's a rare bit of good news. Even though the greenscreen gets noticeably darker at, say, the left edge, when you pull your color difference matte it will be less uneven than the original film plate! "How can this be?" you marvel. This is one of the magical virtues of the color difference matte technique compared to, say, chroma key. To see how this is so, let's refer to Figure 2-68, which shows a greenscreen with a dark left edge, and Figure 2-69, which shows the slice graph of the unfortunate greenscreen.

Figure 2-68 Uneven greenscreen

Figure 2-69 Slice graph of uneven greenscreen



As can be seen in the slice graph, the greenscreen brightness drops off dramatically on the left edge. The RGB values at the left edge of the graph drop almost 20% compared to the properly lit region. But look closely at the *difference* between the red and green channels. All three channels drop together, so their *difference* does not change so dramatically. In the properly lit region the raw matte density is about 0.30, and in the dark region about 0.28, only a little less. Although the green channel dropped off dramatically, so did the red channel. In fact, they largely drop off *together*, so their difference is affected only slightly. More than any other matte extraction method, the color difference matte is fairly forgiving of uneven greenscreens. That is the end of the good news for today, however. Next, we will take a look at how the uneven greenscreen affects the matte, then what might be done about it.

Ex. 2-13 Figure 2-70 shows an unevenly lit greenscreen and foreground element, both getting darker on the left. Next to it in Figure 2-71 is the resulting uneven raw matte (before scaling) using our standard color difference matte extraction procedure **(G – max(R, B))**. A low density region can be seen along

the left edge of the raw matte that will become a partially opaque region if we don't do something about it. The slice graph of the uneven matte in Figure 2-72 shows the lower density in the backing region on the left edge (exaggerated for clarity). There is also a red and a green reference line added to the slice graph. The green line represents the desired point that would normally be pulled up to 100% white by the scaling operation. If this were done, the backing region at the left edge would remain partially opaque in the process, as it won't make it to 100% white. We can solve this problem by scaling the matte up harder from the much lower point indicated by the red line until this becomes the 100% white point. But this will clip out some of the edge detail of the matte and harden the edges, neither of which we want to do. What is needed is some way to even up the greenscreen plate.

Figure 2-70 Uneven greenscreen

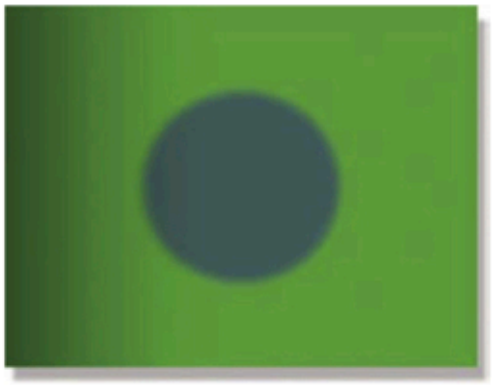


Figure 2-71 Uneven raw matte

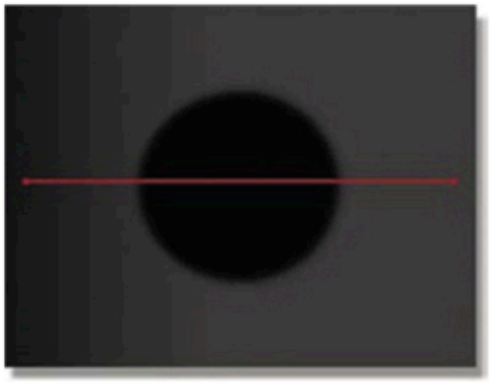
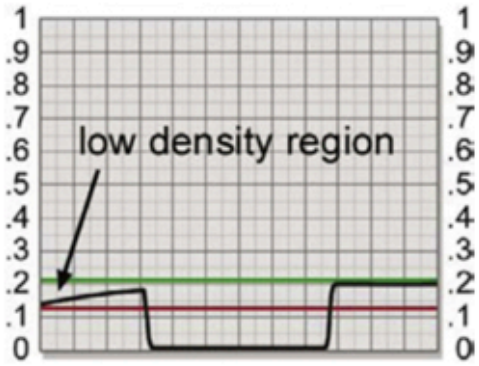


Figure 2-72 Slice graph of raw matte

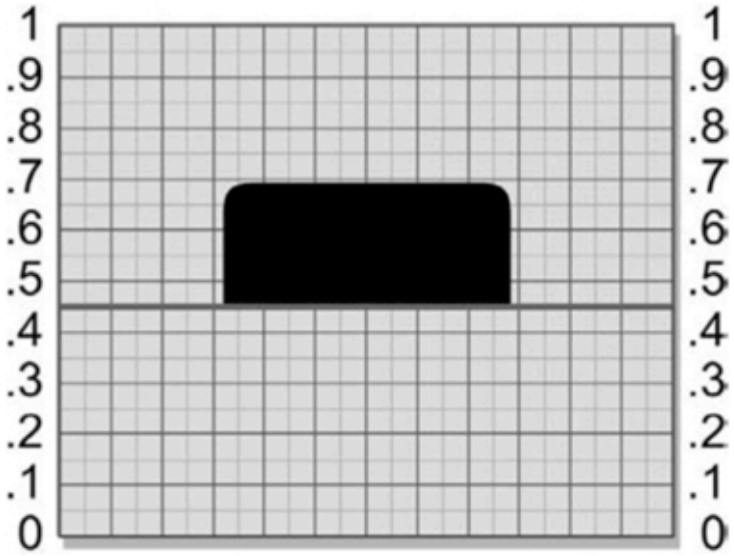


2.6.5 Screen Leveling

This is one of my favorite techniques – screen leveling. It has many more applications than just straightening out uneven greenscreens, but this is a fine place to describe the process. Imagine turning a greenscreen plate up and looking at it edge on. You might visualize the three color channels as three "planes" of colored plastic. The green backing of a uniformly lit greenscreen would be seen as a nice flat surface of green. This "edge on" visualization is very useful for understanding screen leveling, and is also the "point of view" of the slice graph.

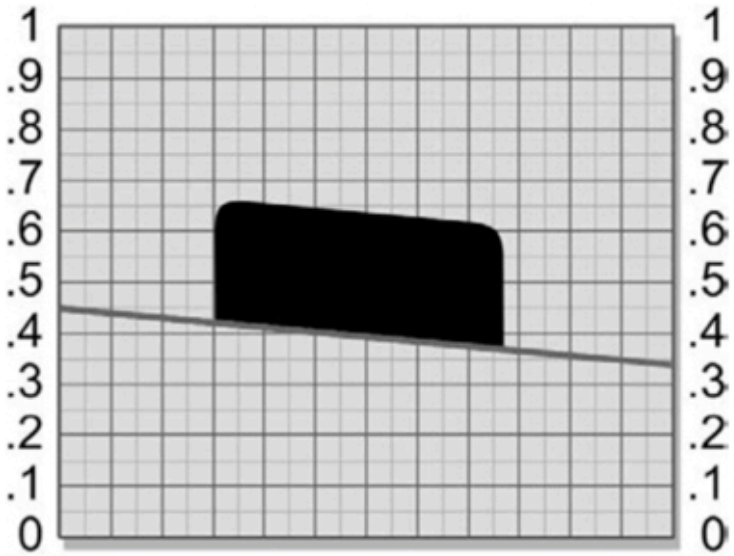
Here is a conceptual model that will help you to visualize how screen leveling works. Figure 2-73 shows a perfectly flat "greenscreen" surface with a "foreground object" sitting on it. It would be trivial to cleanly separate the foreground object from the greenscreen surface with a single threshold number – 0.45 in this case. Anything below 0.45 is the greenscreen, and anything above it is the foreground object. This threshold number represents the point that is scaled up to 1.0 to give the raw matte full transparency in the backing region.

Figure 2-73 Perfectly flat surface



However, in the real world, the greenscreen is never perfectly flat. It is invariably lit unevenly with a "slope" to it like Figure 2-74 (getting darker towards the right in this example). As a result, the raw difference matte will not have a uniform value across the screen, in spite of the color difference matte's tolerance to uneven lighting. Now try and separate the foreground object from the greenscreen with a single threshold value. It cannot be done. Whatever threshold you select will either leave some pieces of the foreground object behind, include pieces of the greenscreen, or both.

Figure 2-74 Uneven surface



But what if we could "straighten up" the greenscreen, making it flat? In Figure 2-75, a "slope" is shown at the bottom of the graph with the sloped surface as a compensating gradient that starts at 0 on the left and goes to 0.1 on the right. If this gradient is summed with (added to) the greenscreen plate, it will raise the low end of the backing and the foreground object together to restore a "flat" surface like the one in Figure 2-76, which can now be separated with a single threshold value. If the color difference matte is extracted from the "leveled" greenscreen, the raw difference matte will now be much more uniform. You cannot use this technique with the raw matte after it is pulled. The reason is that the gradient you add to level out the raw matte would also be added to the zero black region of the foreground, introducing a partial transparency there. It must be done to the greenscreen plate *before* pulling the matte.

Figure 2-75 Compensating gradient

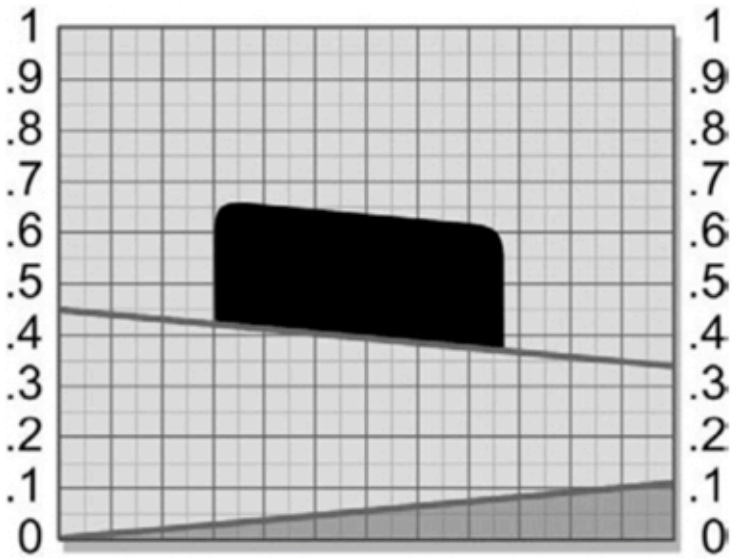
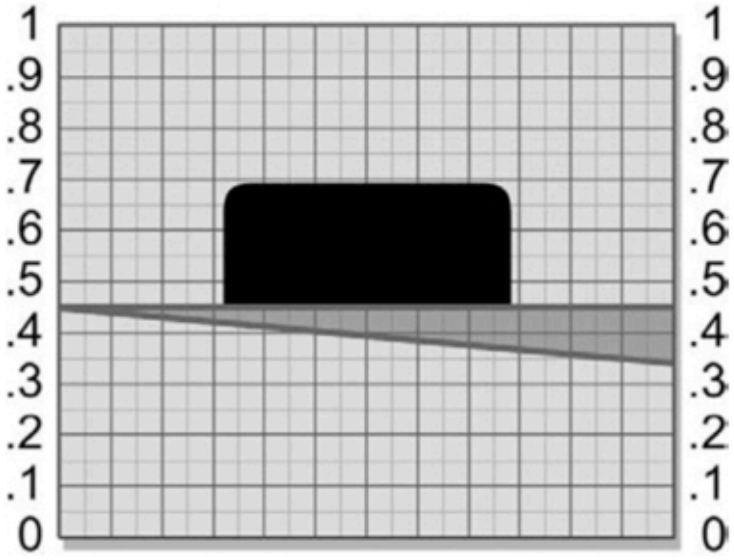


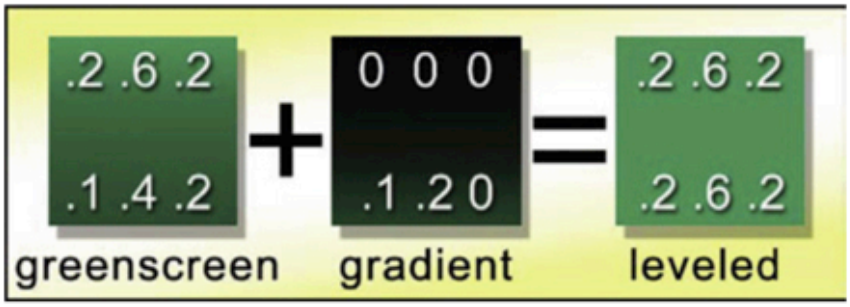
Figure 2-76 Corrected surface



To apply this to a simplified test case, let's say that we have a greenscreen plate that gets dark from top to bottom. The procedure is to measure the RGB values at the top and bottom, subtract the two sets of numbers, and this will tell us how much the RGB values drop going down the screen. We would then make a "countergradient" that is zero black at the top and whatever value the difference was at the bottom. We would then add the countergradient plate to the greenscreen plate to level it out.

An example is shown in Figure 2-77. The leftmost square represents a greenscreen plate with RGB values at the top of **0.2 0.6 0.2**, and at the bottom **0.1 0.4 0.2**. Subtracting the two sets of RGB numbers, we find that the bottom is RGB **0.1 0.2 0.0** darker than the top. We now create a countergradient that is black (**0 0 0**) at the top and **0.1 0.2 0.0** at the bottom. Note that the difference between the RGB values will not be the same for all channels, so it is not a gray gradient but a three-channel colored one with different values in each channel. When we add these two plates together, the new greenscreen will have equal values from top to bottom – it will now be "level." We can now pull a much cleaner matte. Because the compensating gradient was green-dominant and was also added to the foreground region, it will slightly raise the green record in the foreground region. This will sometimes add a bit of contamination to the foreground, depending on its color composition, but it is well worth the gains of the improved backing uniformity. We have traded the large problem of an uneven greenscreen backing for the much smaller problem of a little possible foreground contamination.

Figure 2-77 Leveling a greenscreen



Ex. 2-14 Meanwhile, back in the real world, you will not have unevenly lit greenscreens that vary neatly from top to bottom like the little previous example. They will vary from top to bottom and left to right and be bright in the center and have a dark spot just to the left of the shoulder of the actor, making a much more complex irregularly "warped" greenscreen. Your software package may or may not have tools suitable for creating these complex gradients. The key is to do what you can to level the screen as much as possible, knowing that whatever you can do will help and that it does not have to be perfect – just better. When it comes to the raw matte density, every little bit helps, because the raw matte is going to be scaled up by many times, so even a 0.05 improvement could mean a 10% or 20% improvement in the final results.

2.6.6 Screen Correction

One of the most difficult problems to solve for greenscreen matte extraction are defects in the backing material. There can be seams, folds, tears, patches, tape, and other irregularities in the backing material that deviate wildly from the pure, uniform green they are supposed to be. These backing defects become matte defects and are particularly troublesome when the foreground object crosses in front of them, which they invariably do. Figure 2-78 illustrates a greenscreen with a big fat fold across the backing material and Figure 2-79 shows the resulting matte with a matching defect in the backing region from the fold. Scaling up the whites in the matte will clear the defect from the backing region of course, but at the expense of severely hardening the edges of the entire matte.

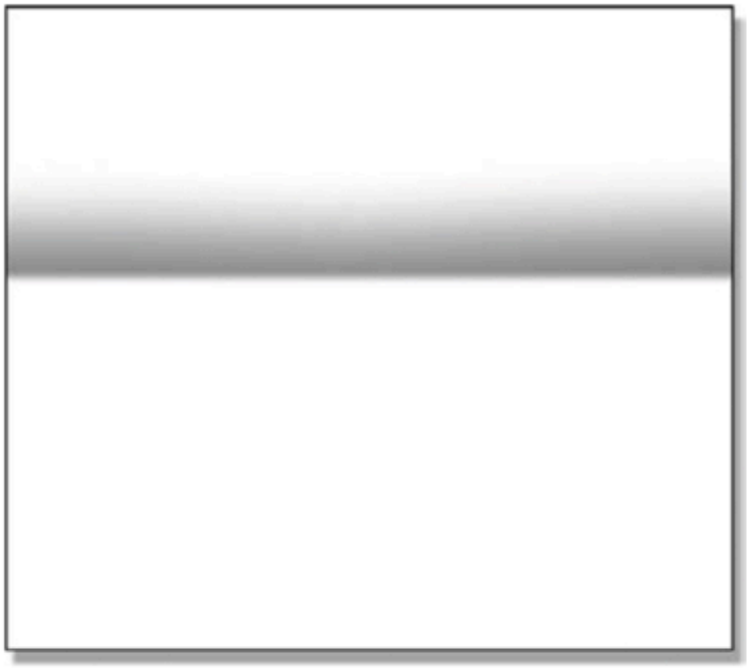
Figure 2-78 Original greenscreen

Figure 2-79 Resulting defective matte

Ultimatte devised an almost magical solution to this problem that they call *screen correction*. The basic idea is that in addition to the original greenscreen, a clean pass of the backing material is shot *without* the fore-ground object that is used as a reference to correct the defects in the original greenscreen. An example of this clean greenscreen is shown in Figure 2-80. When a matte is extracted from the clean greenscreen (Figure 2-81), it will have all of the defects of the original greenscreen, including those that were hidden by the foreground object. The clean greenscreen and its matte are used to create a "defect map" of the original greenscreen backing material that is then used to fill in its "potholes" to make it a uniform green. Beyond its use with Ultimatte, you can also perform your own screen correction to be used with other keyers or for pulling your own mattes.

Figure 2-80 Clean greenscreen

Figure 2-81 Clean greenscreen matte



2.6.6.1 Screen Correction with Ultimatte

Ultimatte cheerfully advises you to simply connect your clean greenscreen plate to the screen correction input of the Ultimatte node. All you need is a clean greenscreen plate, of course. Meanwhile, back in the real world, you will almost never be given a clean plate. There wasn't the time, or the money, or no one thought of it. Furthermore, it would only have been helpful for locked-off camera shots, which most directors today prefer to avoid. If the camera is moving, the only way to shoot a matching clean pass of the greenscreen is with a motion control rig. Yeah – that's going to happen.

So why even bring up the subject of screen correction and extol its virtues only to say that you probably can't use it? A couple of reasons. First, now that you know about it, you may be able to request that clean plates of the greenscreen be shot for your next job. You never know – it could happen. The second reason for bringing it up is that it is entirely possible to create your own clean greenscreens for screen correction, and it is well worth the hassle for particularly troublesome shots.

To create your own clean greenscreen plate, you may be able to select one frame of the original greenscreen shot and paint out the foreground object. Alternatively, you might be able to cut pieces from several plates and paste them together. If the camera is moving, then the situation is more complicated. An oversized clean plate will have to be created that covers the entire exposed area of the greenscreen over the course of the shot, then motion-tracked to make a match move with the camera. Lens distortions limit this approach, because the defects in the static clean plate can drift relative to the moving frames. Obviously, these heroic measures are only worth the time and money when you have some very serious problems to solve on a real important shot.

2.6.6.2 Doing Your Own Screen Correction

Assume that, by hook or by crook, you have come to posses a clean greenscreen plate and are ready to use it to screen correct a defect-riddled greenscreen shot. This is a nontrivial procedure, so in addition to the step-by-step procedure here, we also have the pictographic flowchart in Figure 2-82 and the flowgraph in Figure 2-83 to help keep us on track. The flowgraph shows which operations to use, and the pictographic flowchart shows the progression of images that you will see. This procedure will ultimately create a new version of the original greenscreen called the *corrected greenscreen*, which has all of the backing defects removed. A clean matte is then pulled from the corrected greenscreen using the keyer of your choice or your own matte extraction methods. The corrected greenscreen is made by first creating a *correction frame*, which is an RGB image that holds all of the pixels required to fill in those pesky potholes in the original greenscreen. The correction frame is then summed with the original greenscreen to make it smooth and uniform. Let's see how it is done, again, referring to Figures 2-82 and 2-83.

Figure 2-82 Pictographic flowchart of screen correction procedures

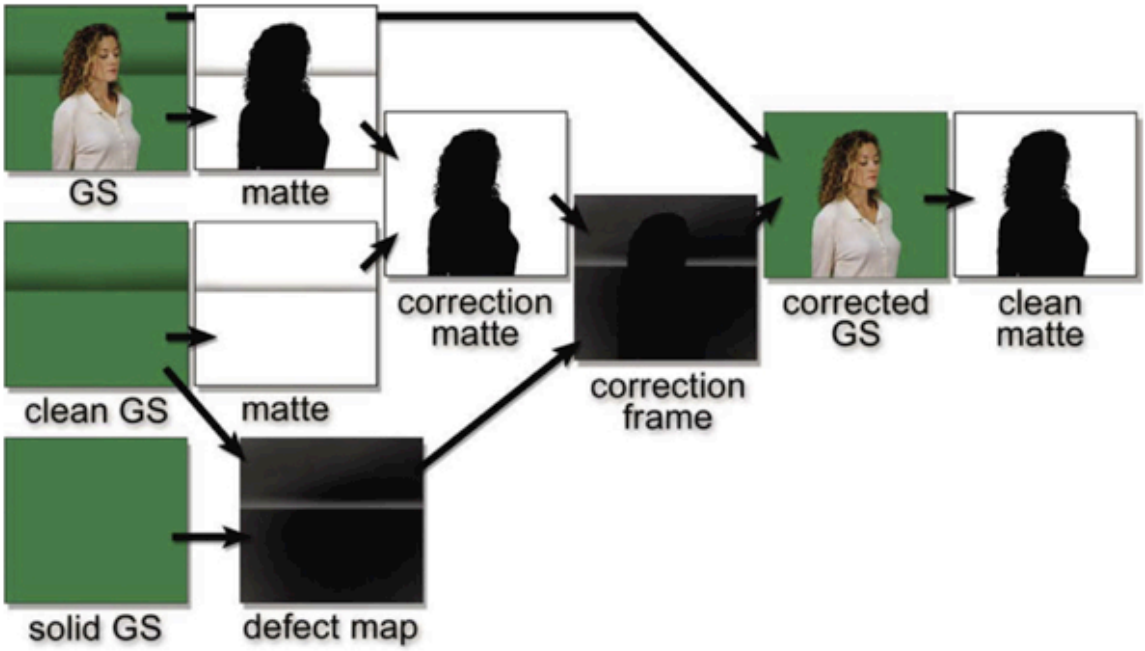
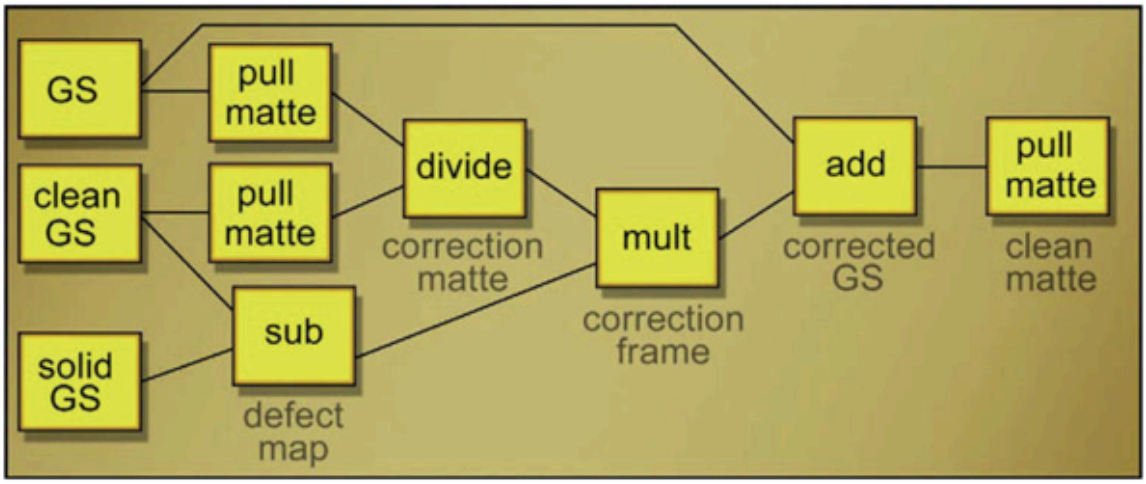


Figure 2-83 Flowgraph of screen correction procedure



Step 1 – GS matte: pull a matte from the original greenscreen plate using your best method. Of course, it will have the defects from the original greenscreen. Invert the matte if necessary to get a black on white matte as shown.

Step 2 – Clean GS matte: Pull a matte from the clean greenscreen plate using *exactly* the same methods and settings as the original greenscreen matte. This matte must be identical to the GS matte everywhere, with the only difference being the presence of the foreground object. This matte must also be a black on white version.

Step 3 – Solid GS: This is just a solid color node filled with the original backing color and represents a perfectly uniform greenscreen. The pixels in the original backing will eventually all be brought up to this level. The RGB values of the original backing will vary all over, so be sure to set the solid GS to the brightest (maximum) RGB values found in the original greenscreen.

Step 4 – Defect map: Subtract the clean GS from the solid GS to make the defect map. It represents the difference between the actual backing and a uniform one, and is normally quite dark. Before it can be used, however, the foreground object's region must be cleared to zero black.

Step 5 – Correction matte: Use a math node to divide the original GS matte by the clean GS matte. Their matching defects will magically cancel out leaving a uniform backing region while the foreground object will stay zero black. Again, both input mattes must be black over white for this step to work.

Step 6 – Correction frame: Multiply the correction matte by the defect map to make the correction frame. What this operation does is "punch out" the foreground object from the defect map by scaling it to zero. This correction frame will be used to fill in the defects in the original greenscreen.

Step 7 – Corrected GS: Simply add the correction frame from step 6 to the original greenscreen to create the corrected greenscreen. Now use any keyer or the matte extraction method of your choice on the corrected greenscreen to pull a clean matte.

Ex. 2-15 Although a keyer must obviously be given the corrected greenscreen so it can pull its own matte, if you are pulling the matte yourself, you might be able to just use the correction matte created at step 5. The correction matte is a very good matte with edge characteristics very similar to the clean matte that you would get from the final corrected greenscreen.

Adobe Photoshop Users – there is no divide operation in Photoshop, so you are forever barred from using this technique. However, you don't need to, because you can always just paint out any defects – an option not really available when compositing 500 frames.

2.7 ADOBE AFTER EFFECTS MATTE

We saw earlier that the classical color difference matte generates a raw matte by finding the difference between the backing color and the maximum of the other two color channels. Adobe has developed a rather different method of generating a raw matte that is very simple and, frankly, works much better than it should. It only uses the blue and red channels; the classical method uses all three. For a bluescreen, the blue channel is inverted, and then the blacks are scaled down to zero. This result is then screened with the red channel to create the raw matte, which is then scaled to make a solid white core and zero black backing (the screen operation is described in Chapter 6). Let's walk through an example and see why it works.

From the bluescreen shown in Figure 2-84, the blue channel is isolated in Figure 2-85. Because the backing region of a bluescreen has a very high blue level in the blue channel, the backing region surrounding the foreground object is bright, while the foreground itself is dark. This is reversed when the blue channel is inverted as shown in Figure 2-86. The backing region is now dark and the foreground is light, which is starting to look more matte-like, but we need the backing region to be zero black and the foreground region to be 100% white. The next step is to scale the blacks down to zero, which clears the backing region, as shown in Figure 2-87. However, there are seriously dark regions in the face area of the matte that must be filled in.

Figure 2-84 Bluescreen

Figure 2-85 Blue channel

Figure 2-86 Inverted blue channel

Figure 2-87 Black level scaled to zero

The red channel is used to fill in these dark regions, because there is normally high red content in things like skin and hair. Also, the red channel should be fairly dark in the backing region, because that is the bluescreen area. Compare the red channel in Figure 2-88 with the inverted and scaled blue channel in Figure 2-87. The red channel is light in the face exactly where the blue channel is dark. We can use the red channel to fill in this dark region by screening it with the inverted blue channel in Figure 2-87. The results of the screening operation produce the Adobe raw color difference matte shown in Figure 2-89. The face and other regions of the foreground have now picked up some very helpful brightness from the red channel. Of course, the backing region also picked up some brightness from the red channel's backing region, but this we can fix.

Figure 2-88 Red channel

Figure 2-89 Adobe raw color difference matte

The Adobe raw matte now simply needs to be scaled to a solid black and white to get the results shown in Figure 2-90. As when scaling the classic raw color difference matte, the scaling operation must be done very carefully to avoid over-scaling the matte, which will harden the edges. Be sure to use the matte monitor described at the beginning of Chapter 3 when you are scaling any of your raw mattes, so you can see exactly where you are in the scaling process.

Figure 2-90 Adobe scaled matte

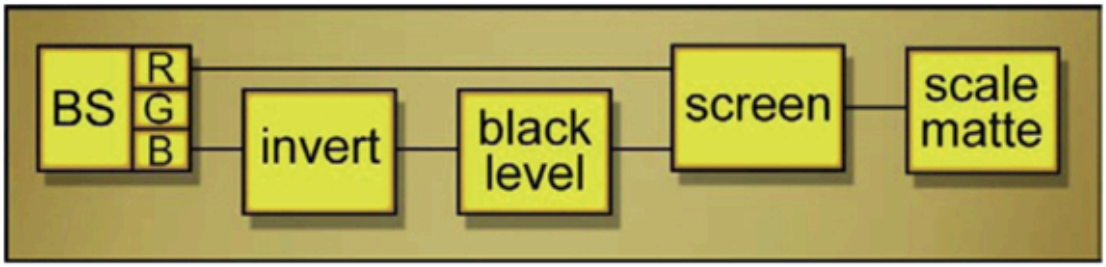
By way of comparison, the classic color difference method was used to pull the raw matte shown in Figure 2-91 from the same bluescreen. Compare it to the Adobe raw matte in Figure 2-90 and you can see that they produced utterly different results. However, once it is scaled to solid black and white in Figure 2-92, the classic scaled matte is incredibly similar to the Adobe scaled matte in Figure 2-90. But they are not identical.

Figure 2-91 Classic raw color difference matte

Figure 2-92 Classic scaled matte

A flowgraph of the Adobe color difference matte technique is shown in Figure 2-93. The first node, labeled BS, represents the original bluescreen.

Figure 2-93 Flowgraph of the Adobe color difference matte



The blue channel goes to an invert node that correlates to Figure 2-86. The black level is adjusted with the next node to scale its backing region down to zero, then the results are screened with the red channel from the original bluescreen image. The screen node produces the Adobe raw matte shown in Figure 2-89. The last node is simply to scale the matte to establish the zero black and 100% white levels of the finished matte shown in Figure 2-90.

Here are some variations on the Adobe color difference matte that you might find helpful.

Use the green channel too. After screening the red channel with the inverted blue channel, just lower the blacks to zero again, then screen in the green channel. Now scale the resulting raw matte to zero black and 100% white.

Use color curves on the blue and red channels to shift their midpoints up or down to clear out problem areas or expand and contract edges.

The red channel can be added instead of screened. In some cases it might be better.

If the foreground object had a high green content and low red content (such as green plants), use the green channel instead of the red.

Use any or all of the preprocessing techniques described in Chapter 2 to improve the original bluescreen for better matte extraction.

Of course, this technique is easily adapted to pulling mattes on green-screens. Just use the green and red channels instead of the blue and red channels.

We now have two powerful color difference matte extraction methods, the Adobe and the classical. So which is better? The truth is neither. As you saw in the introduction to Chapter 2, the color difference process is a clever cheat that is fundamentally flawed and riddled with artifacts. The only question is whether the inevitable artifacts will create visually noticeable problems in a given composite. Depending on the image content of a given shot, either method could prove superior.

Ex. 2-16 Of course, you need not choose just one method. Perhaps the Adobe method gives more detail in the hair, but the classical method has better solidity in the core of the matte. So combine them. Use the Adobe method as the outer matte and the classical method to create a solid core matte. Perhaps one method works better on one part of the foreground and the other does better on the rest of the foreground. Roto a matte to isolate that one part of the foreground, then use it to combine the two mattes. Be mindful of the edge where they meet, however, lest we see your seam.