Compositing in Blender using Nodes- a brief intro.

The compositing Nodes in Blender make it one of, if not the, most powerful freely available applications for image compositing and effects. By this I mean colour grading, keying and spill management, as well as other less obvious techniques, like artificial depth-of-field and so-on.

As there are not really any node-based apps available to the masses, there isn't any documentation on how to start to use them, except for Shake, Fusion or Nuke tutorials, and these can be hard to translate to Blender if you've had no experience with those apps. With this in mind I'll try here to give a brief intro to nodes, and a couple of examples of keying and de-spill operations.

Some of this will be VERY basic, mainly because I couldn't find any basic node compositing tutorials or info when I started out. If you know the basics already, please skip them.

The image used in this tutorial is from another tutorial about building a keyer in Shake, by Mike Seymour, posted on www.FXGuide.com.

The Ultra-Basics.

Fist you'll need an input, either a render result or an image sequence of movie file. You'll also need a viewer to see the output without rendering and a composite node to render the composite. To view the output, set a window to the UV/Image Editor, and set the pull-down menu to "Compositor". To render a composite, set the parameters as you would normally would, and select "Do Composite" from the render menu before hitting render.

Although there is as yet no system for animating the compositing system, and the play button won't render on the fly, you can see the effect on different frames by moving to the desired frame and refreshing the view by clicking on the desired viewer.

Add nodes by hitting space bar and selecting them from the add menu.

Think Like A Pixel

The key to nodes, unlike layers (eg photoshop, after-effects etc...), is to think like a pixel mathematically. A pixel can be represented by a four-dimensional vector, that is [Red, Green, Blue, Alpha]. Therefore, a half transparent orange pixel is given by: [1, 0.5, 0, 0.5], (I'm using floating point notation here for the pixel values because you can visualise the colour as a percent of the channels capacity. To convert to 0-255 notation, simply multiply by 255).

These values are the crux of compositing, as they are what operations like Add, Screen, and the rest of the channel ops use.

So if you have two solid images [1, 0.5, 0.8, 1] and [0, 0.2, 0.1, 1]. the add result would be [1, 0.7, 0.9, 1] (a purply-white), the subtract reslut would be [1, 0.3, 0.7, 0] (fully transparent if you subtract alphas, a deeper lilac other wise. If you did a lighten on the image (returns the maximum of the images on a channel-by-channel basis), you'd get [1, 0.5, 0.8, 1] (which is the first image), a darken (returns the minimum) would give you [0, 0.2, 0.1, 1] (which is the second image).

Know Your Nodes

It is most important to know what nodes you have to use, and what they do, or are capable of. Blender has at present, only a coupole of dozen nodes, but it has the most important ones, and while some operations may seem a little convulted, they are possible.

The two most imporant nodes for colour-type work you can have are the Mix node (which gives you all the channel ops like Add, subtract, Multiply, Screen, Lighten, Darken and more.....), and the RGB Curves node (which can be used as a channel seperator, clamper, filter, re-colourer and more...).

There are also other important nodes such as:

Seperate RGBA (gives you three single channeled images (values) from the RGBA records of an image).

Serperate HSVA (same as above, but giuves you the Hue, Saturation, Value and Alpha Channels).

ColourRamp (allows you to map the range of a value to different colours- works the same as the ramp in the other Blender sections).

Blur (allows you to apply different blurs to an image)

Filters (applies different filters: edge detect, soften, sharpen etc...)

Texture (applies a texture defined in the 3D section to an image, useful for garbage mattes using Blend texture)

Translate (translates an image!)

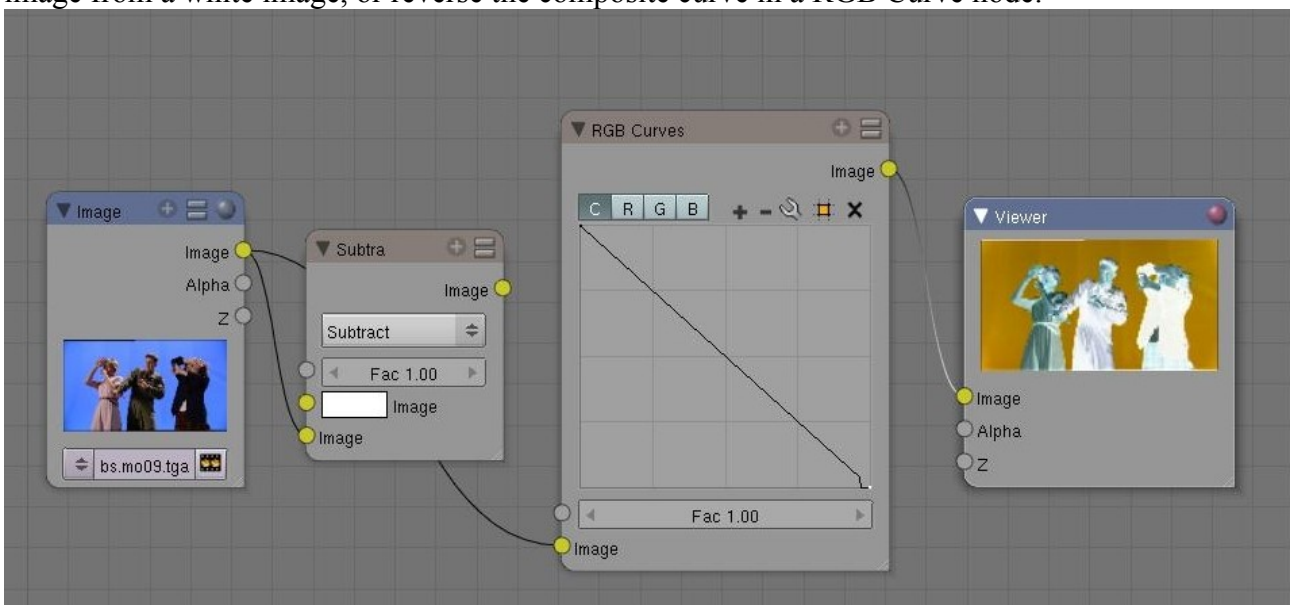
Hue-Saturation (allows alterring of the Hue and Sat records).

There are more nodes than this, are there will no doubt be more to come in future releases.
When you know what your nodes do, you can also get any tutorial for any node-based package, and translate them for blender.

Some Examples

Inverting an image

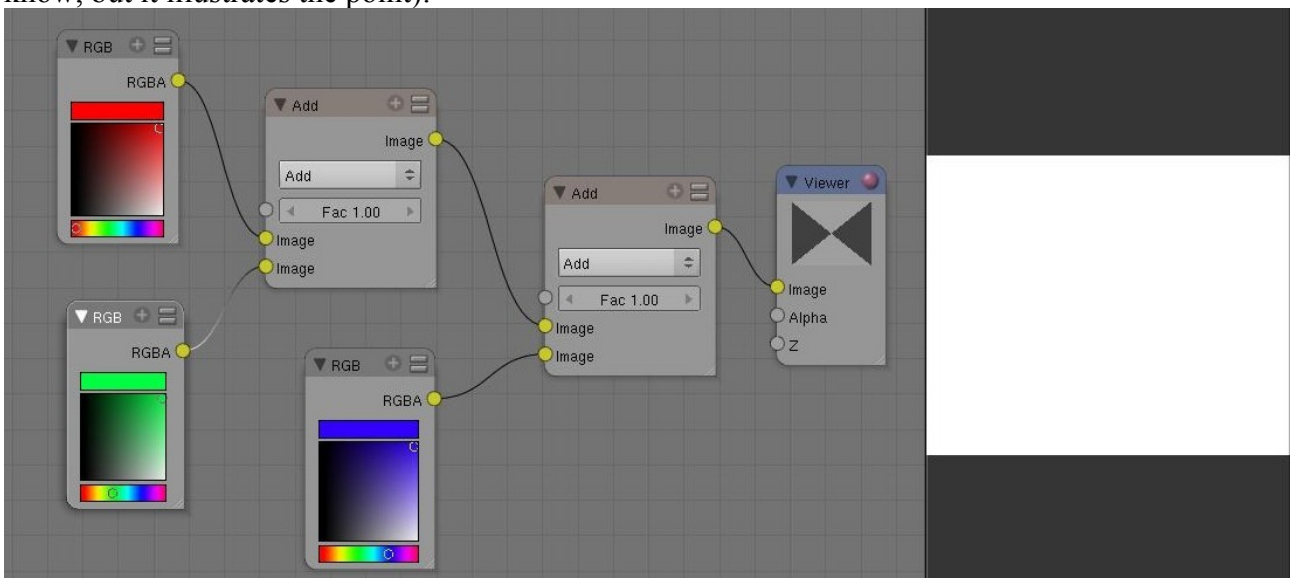
There are two ways to do this simple, but indispensible op, either use a Mix node to subtract the image from a white image, or reverse the composite curve in a RGB Curve node:



If you use the second method, you can also invert only specific channels of an image.

Contructing an RGB image from three sources.

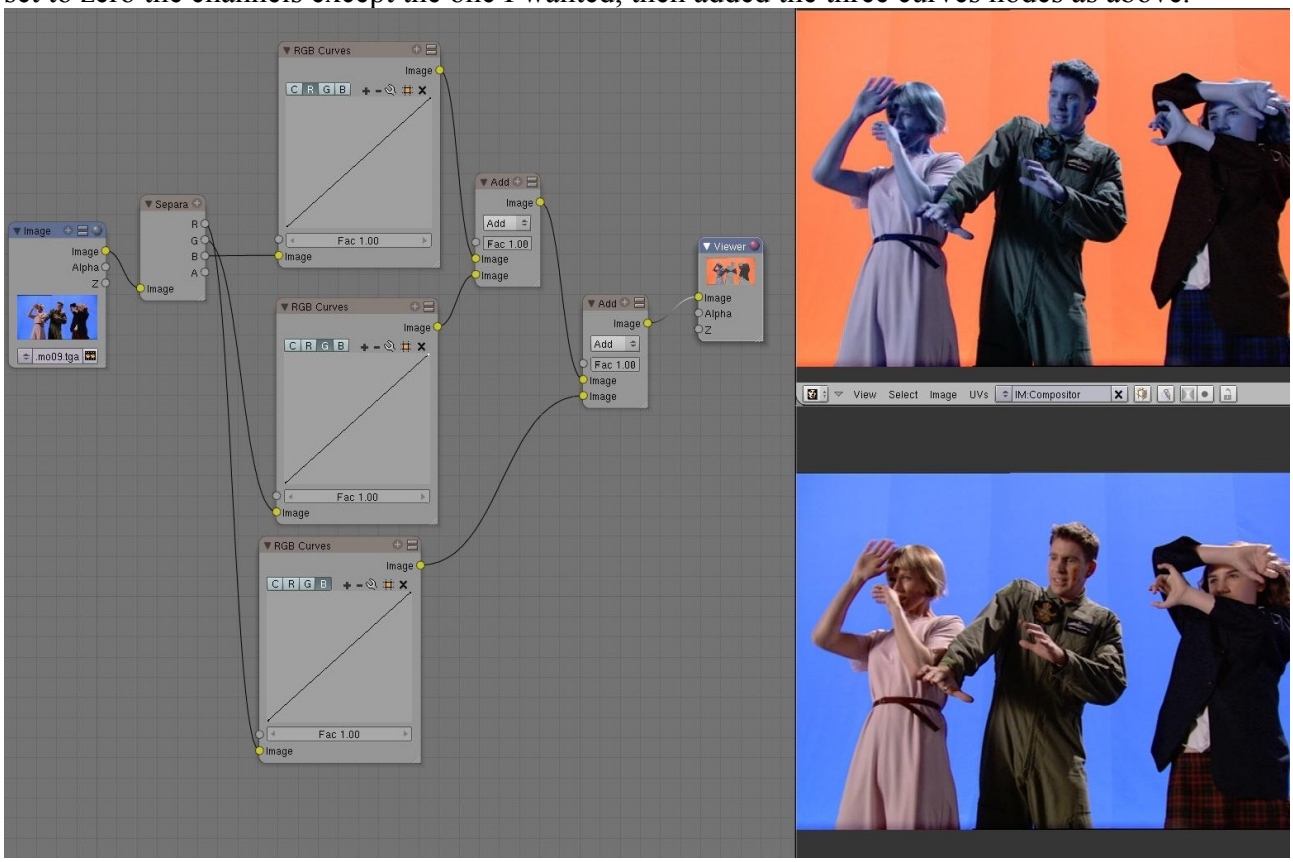
This is acheived using two mix nodes to Add the sources together. In the example below, I have added a [1, 0, 0] (red) to a [0, 1, 0] (green) to a [0, 0, 1] (blue), to get a white [1, 1, 1]. (Cool I know, but it illustrates the point):



This is an especially useful operation for effects, and is used widely in Keying and de-spill operations.

Mixing/ swapping channels

To swap channels of an image, or to make a channel mixer like that in Photoshop, you just need to add a few nodes to the front of the RGB constructor above. To swap the channels below, I seperated the image using seperateRGBA, then sent the RGB records through a cvurves node and set to zero the channels except the one I wanted, then added the three curves nodes as above.

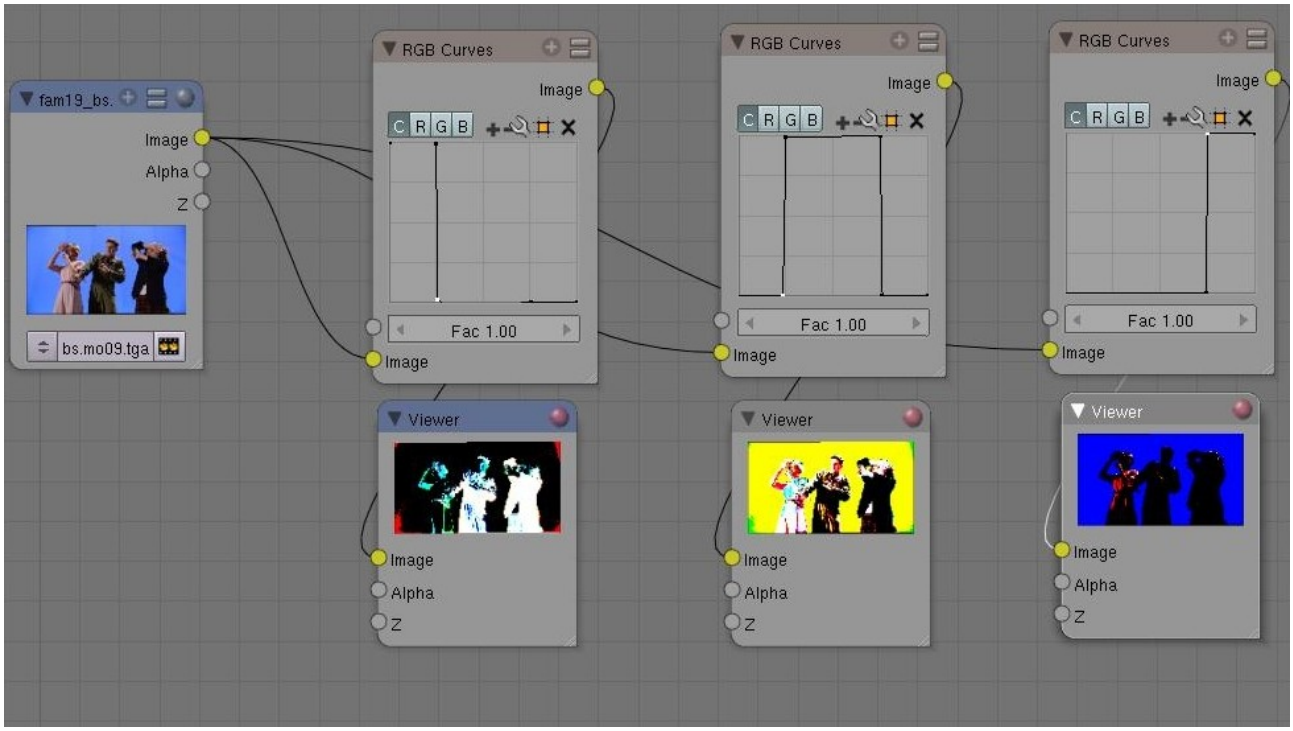


For example, I had three curves nodes, one with Red untouched, and Green and Blue set as flat (to 0), the same but with red flat and green untouched, and the same again but with blue untouched. Then, to swap red and blue, I sent the blue record to the curve node with red untouched, and red to the blue curve node. Then they were added as before.

To mix the channels further, you can adjust the curves that you set to 0 in the other curve nodes and you have an extremely powerful tool for colour correcting.

Shadows/Midtones/Highlights Filter

To make a device to filter between the shadows, midtones and hightlights in an image, use the colour curve as below:



The curves nodes filter, from left to right, the shadows, mids, and highlights. Now you can use these masks to perform operations on these sections of the image.

This technique can be used with a Black and White image as input to make a Luma key, to key out areas of high (or low or mid) lumiance in an image. It can also be used to make a notch filter to only target a narrow band of colours.

A Simple Colour Difference Keyer and De-spiller

And now some of the juicy stuff. Before we make a keyer, a couple of notes about keying with DV footage that I have discovered.

I prefer BlueScreen to green. Some say that green works better, because green is 'brighter' than blue, and this is true. But for this reason, I choose to use blue, as to remove green from the record makes a more noticeable hue shift (but others may disagree, and I'll still use green on occasion).

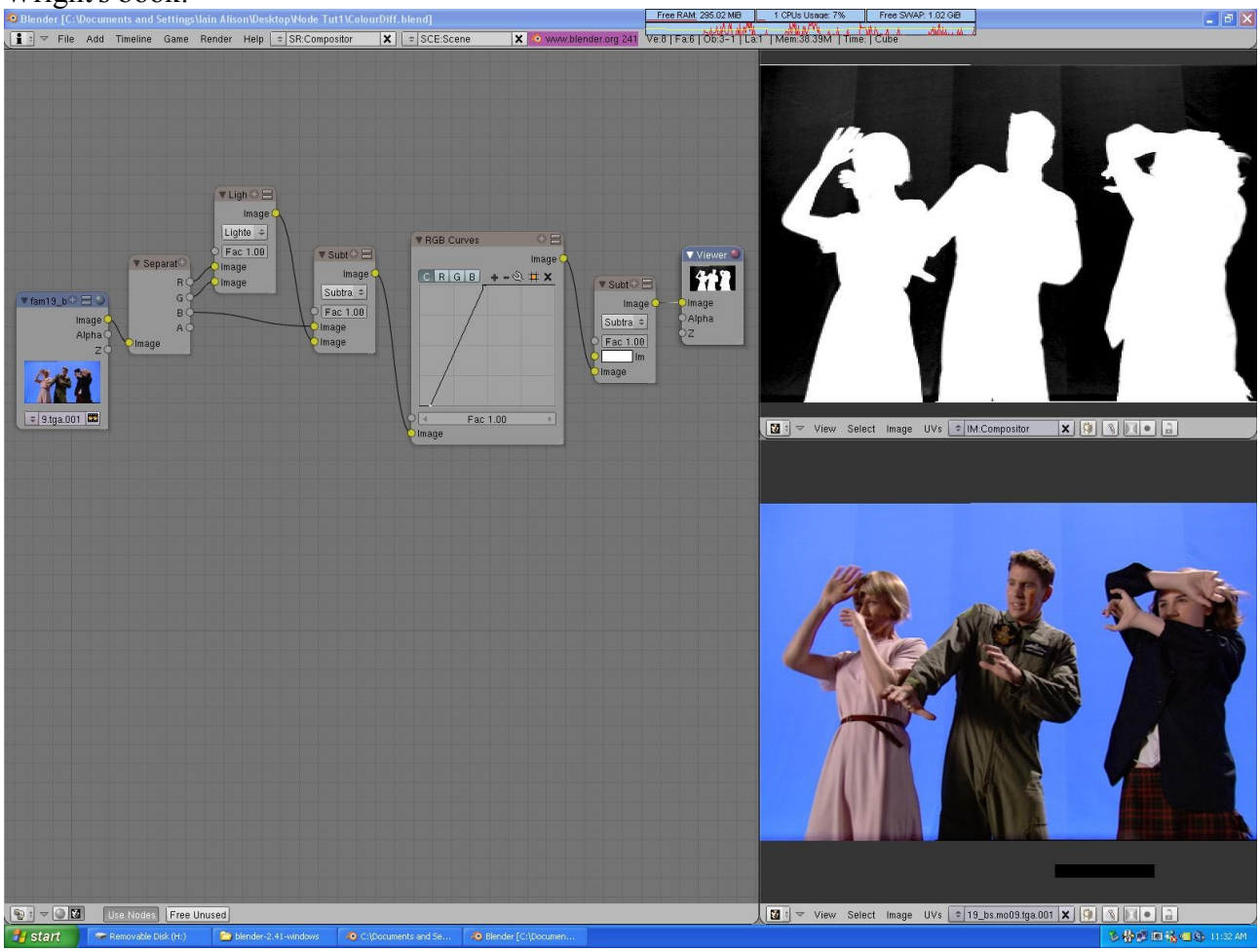
Try to light the screen as well as possible and EVENLY. This will cause less headaches when it comes to keying it out, especially when using the chroma key method.

Try to capture your footage with the Avid DV codec (this is a quicktime codec for use with Avid

products. You can get Avid Free from the Avid website, and capture with that, export as an avi, convert to .TGA sequence with virtualDub or whatever you want. The reason for this is that the Avid codec uses a superior chroma interpolation technique when capturing, and can get past some of the 4:2:0 (PAL) or 4:1:1:(NTSC) problems of DV. If not doing this, use a box blur of about 2-4 pixels bfore pulling your matte, and this should help (if you don't know about the sampling rates of DV footage, check it out on the web).

Also, in production, mattes are practically NEVER pulled in one go, you might make a garbage matte for the bulk of the Background, then several different Chroma/Luma/Colour Difference keys on the principal mattes, which can be combined using different techniques. This is worth bearing in mind when that matte just won't look right. A good solution for me (I use Windows primarily), is to use Wax (from www.debugmode.com) for the garbage matting and even rotoscoping (it has some great bezier mask tools), and then export it into Blender and use the nodes to make a flexible keyer (Wax's keyer is pretty poor, and there are no colour tools).

That all said, here is a simple colour difference keyer, which is a technique derived from Steve Wright's book:



This is different to a Chroma Key, which uses the hue and saturation records to make a matte. The Colour Difference method is superior for most applications because it is more flexible. It uses the difference of the colour records to obtain a key (but you guessed that, didn't you?!).

A walkthrough:
The first node seperates the RGB channels, I then use a Mix Node to get the maximum of the red and green channels (using the lighten option), and then I subtract the result of that from the image's Blue record using another Mix node. This gives a grey-ish matte, which is backwards (darkest in the parts that need to be opaque):

We then use a curve node to scale up the background to white and the foreground to black, this requires some experimentation sometimes. All that then is needed is to invert the result, and we get the matte image shown at the top (the original image is shown below).

A couple of things, this is NOT an optimised matte. It is only for demonstratoin purposes and has several troublesome areas, and bad edges. Also, this is high quality footage, with a totally saturated Blue backgrouns, so it makes for an easy example! It is also shot in HD I think. Also, to use a green screen, swap the blue for green in the above description.

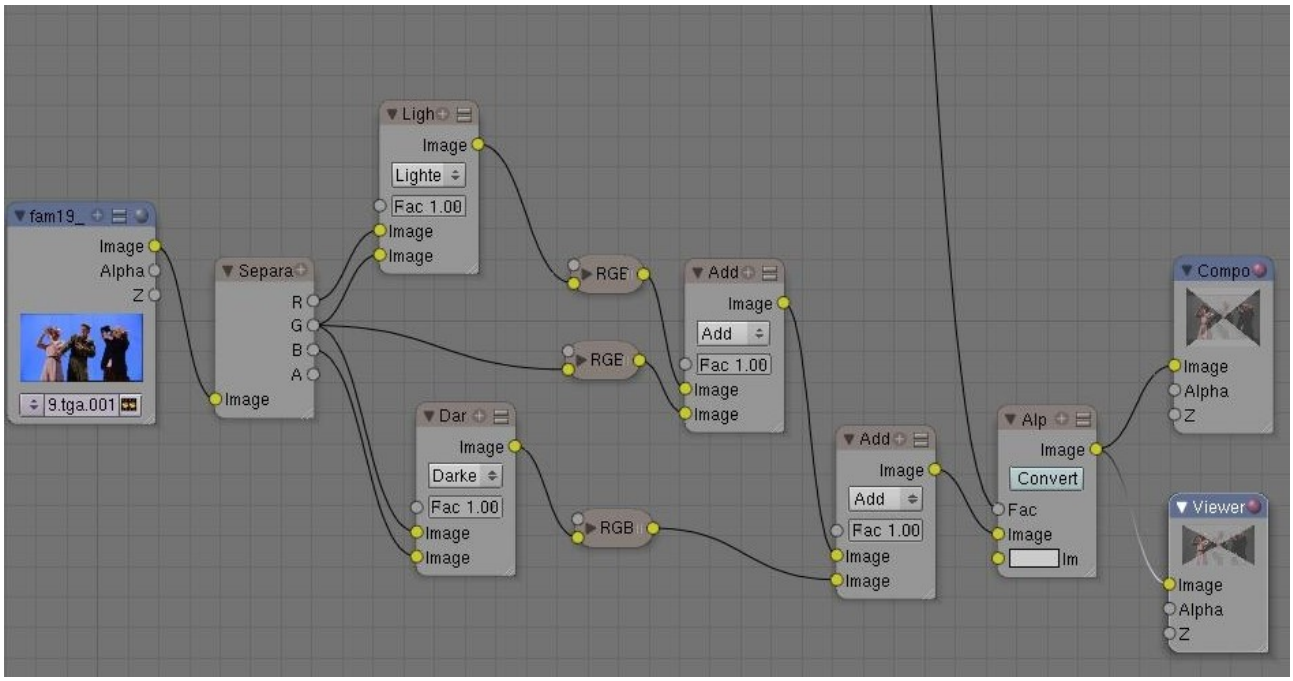
When I composite the image onto an 80% grey background using an Alpha over node, this is the result (notice the ugly blue fringing):

Now we need to look at a despill operation to remove that ugliness.

This is a technique from a tutorial on FXGuide by Mike Seymour (which is where this shot comes from), but there are many other techniques. Basically, we need to pretreat the foreground plate to 'tone-down' the blue around the edges, but retain the colour of the principals in the plate.
Remember the matte is already pulled, so we can'tr screw that uyp by treating thwe imae for spill.

This technique constructs an RGB image of the foreground plate, with the Red made up of the max of tge red and green, the green as is, and the blue the min of the green and blue. This helps to supress the blue in the transition pixels. You could also leave the red as it is, as it makes little difference in this case.

Here it is:



The line coming in is the output from the curves node of the colour Diff Key.
Here is the new composite:

Notice how the fringing is alot better now.

There are MANY ways to despill an image, and this is just one of them. Others include limiting the blue by the average of red and green and so forth.

But this one works here.

References.

Digital Compositing for Film and Video, Steve Wright. Published by Focal press.

Excellent book, loan it, buy it, read it, have it for breakfast. It will teach more about compositing than probably any other single resource avaiable.

Building a Keyer in Shake, a tutorial by Mike Seymour
http://www.fxguide.com/fxtips-283.html
A great tutorial. Its for Shake, but its the principals that count. This is where the plate I used came from too.

Thanks for reading this, I hoped it helped you somehow.
If you have any questions or suggestions, feel free to email me at
iaina@dodo.com.au, or PM Deep_Thought at the Blender Artists forum.

Thanks Again

Deep Thought.

HAPPY BLENDING