Blender For The Faint Hearted
Presentation Notes

COMP 01 Basic Compositing Concepts

The following pages represent the notes from the COMP 01 presentation, as a separate file.

Slide 8: Why Composite?

If you have been using Blender for any amount of time, you will have built models, created scenes, textured models, set lighting and hit the [Render] button to get yourself a final 2D image. So if you can do all of this, why should you need to know about compositing?

There are a number of reasons why compositing is a good way forward, and the are certainly not restricted to the examples below – many of the reasons will show up in individual scenes as well as just the blanket reasons given here:

Speed and Repetitiveness

As we remark on later in this presentation, the world of cinema is not the theatre. In the theatre, everything has to come together at the same time and work perfectly for the experience to be even satisfactory, let along exceptional. And yet, this is the way in which most users of Blender (and other CG packages) try to operate. Every scene is in one place – when the scene is rendered, everything is rendered in one go. So OK, a set is not going to fall over or an actor fluff his lines, but why bother rendering parts of an image over and over again for every frame when it never changes?

In the cinema, there are multiple units. Scenes may even be filmed in completely a different order to how you eventually see them on the screen. Footage may even be projected behind an actor to hide the fact that he is doing everything in a studio and really is in no danger from that huge Bengal Tiger rushing towards him... Compositing is a little like this – different elements are rendered separately and then combined only for the final image. It may be that some parts of the scene only ever need to be rendered once and are then re-used multiple times – either because it never changes from frame to frame, or because the changes that you have been making to your principle actor never impact the rest of the scene elements.

Once configured, the entire process of rendering individual elements of a scene start to make a great deal of sense.

Post-processing

Post processing allows you to take an image and apply various additional processes to it before you use it. You may cut things out using masks, change its colour, blur it, invert it, offset it in time from the rest of the image, apply special effects, and a hundred other things.

It may be however, that you only want to perform this post-processing to certain elements of your image – by compositing different images together, you can restrict where that post-processing takes place.

Internal and External Content

Compositing allows you to decide what content actually gets into your images. You could try to put it all into a single render, but that causes a heavy burden. Instead, you can bring content into the scene only when it is needed and not before. You may take images from different render layers, or even completely separate scenes within the Blender environment. You could even bring in content from totally external sources – but only at the point at which you need it, the creation of the final image.

Slide 9: Pre-Requisites

While this presentation fits into the Blender for the Faint Hearted series, it is far from being a beginners tutorial – at least from the point of view of using the basic elements of Blender. This tutorial is about the compositor aspects of Blender, the how and the why.

As a result, this tutorial assumes that you are an existing Blender user and, even if you have not already gone through the earlier BFTFH tutorials, you are otherwise familiar with the Blender basics – most importantly, the methods of navigation both within a 3D view and around the interface.

No allowances will be made for a lack of navigation knowledge in this tutorial – if you feel that you do not have these skills, then you are probably trying to run before you can walk and should look at a one of the many familiarisation tutorials around that can give you a good foundation in Blender, including the Blender for the Faint Hearted tutorials..

Slide 11: Scenes

In many ways, you can think of a Scene as being a TV studio or a Motion Picture sound stage. In each one, a set has been built, it has its own lighting set-up and cameras, crew and cast. Often, these places are each recording their own television program or motion picture, but at other times they are being used to film different aspects of the same production – all of which will be combined later in the editing room.

Blender scenes work in exactly the same way – only in our case, we have the advantage that we don't have to rely on everything in each scene being totally unique. We work some of our own magic and have all sorts of components in one scene be 'shared' with other scenes. In many ways, our actors can be in two places at the same time.

Default Scene

Most Blender users start the application then use it, without really appreciate that they have entered a studio – or Scene. At the top of the screen is a drop-down that shows that on start-up, Blender has a single default scene already in place. Most of us don't even think about it and overlook the fact that it is there.

Additional Scenes

If we need additional scenes, we can name them and create them in the same way as any other data block in Blender. Just click in the text box and start typing to change its name; use the drop-down selector at the side to select another one or the [**`Add New`**] option to create a new one. It's that easy.

Don't overload yourself however – if you don't need more than one scene, don't create them.

In our case, we'll create a second scene purely for the purposes of demonstration..

Slide 12: New Scenes

When we ask for a new scene to be created, we are presented with four options regarding what we might want as the initial content of the scene.

Empty

Selecting empty creates a new sound-stage for us, but it is completely empty. In this state, we can do whatever we want with the scene from scratch without any baggage hanging around.

Link Objects / ObjData

Selecting to link objects or object data results in those objects appearing in the new scene. The difference between the two is simple:

Object	–	Everything about an object is linked between the scenes. Move an object in one scene, it also moves in the other(s).
Object Data	-	Everything that makes up the structure and appearance of an object is linked between the scenes, but that is it. Where it resides in the scene (along with all of its other activity) is owned by the scene. Choose this method if you want the same object in multiple scenes but doing different things in each.

Full Copy

A full copy does exactly what it suggests – everything that was in the current scene is copied into the new one. No linking takes place, and each object in the new scene is its own, unique copy of that item.

Slide 14: Render Layers

Render layers allow different renders to take place from the same viewpoint on the same scene. Where a render layer comes into its own is in the ability to select which components of the scene will be included in that particular render. Ultimately, you can take the rendered image from multiple render layers and combine them together (composite) to create a final image.

Render layers can be configured so that a number of different aspects of the current scene are included or not included in any render:

Model Layers*

Modelling layers are represented by the four blocks of five small squares in the menu bar of a 3D view. Like a 2D application, you can turn these layers on of off to include them in the 3D view or the render. You may want to use some layers are storage, or some just to order your meshes in an organised fashion.

If you want a layer to appear in any render layer, you must have it turned on – even if a particular render layer does not use it. If you do not want a model layer to appear in a render, turn it off in the render layer, not globally.

Appearance

Any image is a combination of many different visible aspects of the Blender environment. Each textured mesh for example, has colour, specularity, reflection, refraction, transparency and more. The scene itself may be configured to use AO (ambient occlusion), ray-traceable shadows, edge enhancement, and more. In a single image render, inclusion of all of these calculations would be performed for every image if they are turned on.

For each individual render layer it is also possible to turn these aspects off only for that particular render layer. It is possible to render one layer with the basic object for example, another with only the edge enhancement, and another using only the AO pass then combine them later.

In our example, we will be restricting ourselves to only turning off some model layers.

*We have used the term model layers here purely to avoid confusion with the term Render Layers..

Slide 17: What's in an image?

Whenever anything is rendered in Blender, the final image that we get takes up the full dimensions of resolution that we selected. Select a white background in the world settings, and we will get a mesh rendered on a white background.

This background colour is a solid colour. You can think of this like putting a piece of paper onto a desk – the paper blocks out the entire rectangular area of the desk, even though we may only have a small picture drawn in the centre.

At the same time however, Blender also produces what is called an Alpha mask. Most masks are black-and-white or grey-scale (with most of the image being in the black or white range) and represent an inverted silhouette of the actual image that was rendered. If you place the alpha mask over the top of the conventional image all of the background is blotted out, leaving only the meshes showing through.

When using an alpha mask however, the background is not blotted out as such but magically removed – the background turns transparent, so using the piece of paper on the desk analogy, it is a little like the picture actually being drawn on a transparent cellophane sheet rather than white paper.

Why do we need Alpha Masks?

Most people who perform conventional rendering will never use an Alpha mask. This process provides the artist with a final image anyway, so any more information is redundant. The alpha mask comes into its own however, when we need to combine the image with something else.

Going back to our paper analogy, try putting two pieces of paper on top of each other. The top one totally obliterates what is underneath so drawing our images on two pieces of paper is pointless. What we need to do is take a hint from the 2D animation industry and draw our images on cellophane – place two cellophane images in top of each other and you can still see straight through. You can see both images at the same time.

When we combine our alpha image with the colour image, we get the same result. By using the two of these images combined, we can now place different render layers on top of each other and still see what is underneath.

Slide 19: Black-Boxes and Nodes

Black-boxes get used in any industry where some sort of process must be designed – anything from physical manufacturing to software development. Black-boxes represent a discreet operation or process. By hiding the details of the process inside the black-box was can take a step back and in an abstract fashion design the entire process from start to finish without needing to know all of the details.

Black-boxes are very simple components. They have one or more inputs, the name of some process that will take place inside it, and an output that your can send somewhere else – quite possibly into the next black-box that will do the next stage of the overall process.

Blender nodes work in exactly the same way as the black-box approach. Every node in Blender will have one or more inputs – even those nodes that are called 'Input' nodes which only seem to have an output socket that can be linked to other nodes. In this case, the Input node simply takes the input from a previously defined source over which you have no control such as the image produced by a render layer.

The same goes for output nodes – they may appear only to have an input socket, but that is because the output is already defined as going some place that is not negotiable, such as the screen, a file, or the final composite render.

Most nodes however, have at least one input and one output socket. Usually, these 'sockets' can be connected together to pass an image around in some state of processing. Sometimes, it will be a visible image, sometimes an alpha image, and sometimes other types of data like numeric values or time signatures.

Node networks are created in an abstract fashion – if you want to modify the appearance of an image, you need to establish in simple terms the steps that you would need to take in order to do it. Each step will usually relate directly to a node that can do the work.

You may decide that the rendered image need to be brightened, blurred, and inverted. To do this we can:

- Take a Render Layer input node,
- Connect it to a Brightness/Contrast node,
- Connect that to a Blur node,
- Connect that to a Invert node, and finally,
- Add a Composite node to display the result.

Simple, yes? It is all about planning.

Slide 20: What can nodes do?

While we are not going to look at every node in this presentation, we can take a quick look at an overview to see the range of things that a node network can do. Always remember however, that the power of nodes is in the network – not just the individual nodes. The combination of nodes together is far more powerful than any individual node.

Input Nodes

Input nodes provide the initial image (or data) for part of the node network. It is this input that can be fed to other nodes to generate a result. Typically, at least on Render Layer input node will be present.

Output Nodes

Output nodes take the final output from the network as a whole or part of the network and send it somewhere external to the network. Typically, a Composite node will be present to generate the final image, but output nodes can also send visible images to the screen for viewing, or to files.

Colour Nodes

Colour nodes modify the visible content of an image – this might be directly or based on other inputs. This group includes Brightness/Contrast, Gamma, AlphaOver and Mix.

Vector Nodes

Vector nodes modify the vector information of the image / render and including Normalisation and Mapping nodes.

Filter Nodes

Filter nodes provide a range of filtration methods including Blur, Defocusing and Erosion.

Converter Nodes

Converter nodes directly modify part of the technical aspect of the image. Nodes in this group directly can replace alpha maps, convert colour spaces and add ramps.

Matte Nodes

The matte nodes allow the introduction of matte techniques such as chroma keying (often known as Green-Screening) and modification to luminescence.

Distort Nodes

The distort nodes allow the modification of the size and shape of images (the physical characteristics) such as flipping, cropping and scaling, translation and lens distortion.

Slide 23: The Planet

The actual size and shape of our planet is not particularly important as this presentation is really all about compositing. It might be an idea however, to follow the standard configuration that we present and then modify according to any limitations to your system rather than deciding to do something different right from the outset – if anything does go wrong, it is not always possible to tell whether the problem lies with the differences that you have introduced. Try our configuration first, all the way through, then try something of your own.

Layers

It is important to place the camera and light into layer 11 and the planet into layer 1. This is because we want the camera and light to be independent of the planet. We will ultimately want this camera and light to see the cross hairs as well, but to leave them in layer 1 means that the planet starts to intrude – the whole idea of the compositing is that the planet and the cross-hairs will appear in different render layers.

While we have turned off model layers in the two Render Layers for this purpose, it is important before rendering to ensure that all three modelling layers in use are turned on. The modelling layers at the base of each 3D view window represent a master for what cameras will see. If you turn off any of these layers, no camera is going to see your meshes. Even if the layer is turned on in the appropriate Render Layer, nothing will be visible..

Slide 25: The Composite Screen

Contrary to the complaints of many (non-) Blender users, the interface is actually extremely adaptable and easy to understand. Having come from a Lightwave environment, I find that Blender's interface is actually quite straight forward...but then I am still taking the pills.

Each of the 'screens' that are configured within Blender were created using the same set of interface configuration components – a screen just represents the same set of window types oriented in different places that best suit a particular type of operation, such as modelling, animation, or composit... er, oh. No compositing screen.

The nodes system for compositing actually came quite late in Blender's lifetime so of the 6 or so useful screens that come pre-configured a Compositing screen is not included. It is possible to just switch a window into a Node Editor, but if you are really serious about compositing longer term (you probably will be once you see the power behind it), it is probably an idea to go up and create a new screen then orient it in a good layout for composite work.

There is no point in re-inventing the wheel, so we are not going to suggest one. We have found however, that the screen layout presented in "The Essential Blender" from the Blender Foundation is as good as any, so we would recommend re-producing that one. A sample blend file with this screen already configured is available from the Blender Foundation website.

If you do decide to build this screen, it will make working with this presentation a little easier – rather than switching between button menus, much of what you will need to use is available directly from the composite screen anyway and therefore directly at hand.

Slide 28: A Typical Node Layout

Each node in the network has an appearance which is similar to that presented here for the Render Layer node. Some elements may be removed if they are not appropriate – such as the show/hide image button of the node does not actually have the ability to show an image.

Ultimately, a complex node network will start to become overwhelming. While we will not look at this in this presentation, you may ultimately want to start grouping nodes as well as hiding elements that are not in use. You may even wish to start minimising nodes down to their title-bar and only maximise them when you need to make modifications.

Perhaps the largest problem with a large node network is that maximised nodes tend to hide the ribbons that join nodes together. As you will see with the network that we will build in this presentation, that can mean that some ribbons appear to go to one place, when in actual fact they go elsewhere – this perception can be created by the fact that ribbons cross over each other behind nodes.

Slide 33: Node Operations

In this presentation we are looking at an basic overview of the nodes and composite system and not really delving too deeply into individual nodes. Longer term however, it is an idea to create a scene and configure it just to examine the operation of individual nodes to see actual what their effects are in practice – manuals can tell you everything, but it is never quite the same as seeing the results for yourself.

A node that is typical of this in our example, is the RGB Curve. Many people have problems with RGB Curves despite their appearance in many applications from 2D bitmap and vector utilities to 3D renderers like Blender. Try adding this type of node to a scene and make a single adjustment at a time. Adding the RGB Curves node to a scene and then dragging all of the sliders all over the place isn't going to show you much.

Playing is one thing, learning is something else.

Slide 34: Alpha or not Alpha?

When examining a final image, it is easy to be caught out by this question. Is the alpha present, or not present? In some cases, it doesn't make any difference – at least visually – but in others, such as when you are feeding the image into another node it can make the difference between the operation working and not working.

Visual images, such as a full render will typically look the same regardless of whether an alpha is present or not, unless you are rendering the sky and the sky is not black (such as in our example here). A transparent area with nothing behind it is going to be displayed as a black space even if the alpha is present.

A tip is to look at a Viewer or Composite node (the initial Render Layer input node does this as well), not the final image. Because these nodes are showing an image for technical purposes, they will show a transparent area as a two-tone grey chess-board. On these nodes if the background is black, it really is black.

Does it make any difference?

Knowing whether the results of a node network contain an alpha or not makes a difference if you are then going to feed the results into another node – such as an AlphaOver node (which you will see in a short while). If the background of an image is solid, we are back to the stacked paper situation. If we are going to use the image to overlay something else, we need to start thinking about getting an Alpha Mask from somewhere.

Slide 35: Turning off the sky

Regardless of how much we hate it, the simple fact is that any computer image is made up of a set of rectangles (squares pixels in a perfect world). In order to avoid the 'jaggies' therefore – the saw-tooth edge that exists along any line which is not perfectly horizontal or perfectly vertical – we need to introduce dithering. In Blender, this effect is generated using over-sampling and turned on or off with the OSA button.

The results of this process is an optical illusion – any single pixel on the jagged line is blended with a pixel near it by creating a tonal graduation across one or more pixels. If one is white and one is black for example, a shade of grey is placed between them. The result is that, if small enough, the human eyes makes up the deficit and the line become smooth.

Unfortunately, there is an assumption being made here – that the colour of the background behind the line is always going to be constant. In a conventional render this is not a problem as even in an animation, once rendered it's rendered. The background is never going to change afterwards.

The same is not true in composite rendering.

We therefore run into a problem – if we generated a dither pattern based on black going to white and the background is suddenly swapped out with bright-red, we have pixels in the image which are trying to smooth the line into white. The result is a white haze around the line – and probably jagged, totally negating the intention in the first place.

There is no solution to this in post-work that is not drastic – a common approach is to reduce the size of the alpha mask to the extent that it is cutting off some of the outside edge of the image, and therefore degrading the image quality. The solution is to ensure that the image is rendered correctly in the first place.

Premul

The way in which we do this in Blender is to simply turn off the sky – not just the colour, but any sky in the image at all. What this really equates to is turning off any background in both the rendering colour and its effects on over-sampled pixels. To do this, we select the option [**`Premul`**] rather that [**`Sky`**] on the [**`Scenes`**] menu (F10).

[Premul] results in a background never being rendered and any visible image pixels being multiplied by the contents of the alpha channel prior to actual use (hence Pre-Multiplied Alpha, or premul) rather than just relying on colours embedded in the visible image. Dithering information stored in the visible image assumes a black background which, when combined with the alpha channel, allows a smooth transition to whatever image we place behind it.

If you wish this information to be stored with any image file that you save (as with any Alpha information) make sure that the RGBA option is turned on rather than just the RGB.

[**`Key`**] takes this one stage further. Keyed alpha results in all of the dithering information being taken from the alpha channel and none from the visible image. In this case, colours in the visible image remain solid (no dithering information is embedded in the visible image at all).

Some external graphic applications work best with the [**`Key`**] option, but as we will be using these images internally to the Blender compositor, we will stick with [**`Premul`**]. Premul is also the default assumption for any RGBA images that are loaded into Blender from an external source.

Slide 37: Be Constructive

We have created cross-hairs using a pattern reminiscent of a 1940's or 1950's SF space-opera – though granted, we are using colour. You don't have to.

Be constructive in what you are doing in order to make this sequence your own, then continue on and create the node network that will give your cross-hairs a glow.

In this case remember that you are looking not for an atmosphere effect, but for something more electronic. The idea is that these cross-hairs are on some sort of screen, so any glow will be slight and coming from the display, not from a few hundred miles of softly illuminated oxygen-nitrogen atmosphere (though OK, we were not really going for photo-realism in our attempts either).

You can either monitor your progress using a Viewer node, or alternately temporarily plug the last node of your network (at whatever point in you're progress that you happen to be at) into the Composite node and see the results large scale. You can always plug the Planet node back into the Composite node later.

The idea here is to get comfortable with the node system. Create nodes, link them back and forth, unlink them, and feel comfortable with their use. When you are – go for the real thing, and put together a node network that you can use.

Just make sure that you leave the Planet network intact – we need to use that again later..

Slide 38: Spatial Overlap...

One interesting thing to note is that as we are generating two distinct images containing meshes that will never actually meet in the same modelling layer, we can place them anywhere we like as long as they are in the correct place within the rendered image.

The results of that, is that we can create our cross-hairs in exactly the same place as the planet and they will never disappear within it's volume. While this is a minor point, and of moderate interest, it is raising something a little more important...

Because we are going to be compositing several images, with the top most image layer always going to be visibly on top of those beneath, we have to ensure that we order our images in the correct way. It is no good for example, creating a space ship that flies behind the earth then place that image on the top layer – no matter how hard it tries, it is never going to get behind the planet.

At this simple level, make sure that the depth (or Z) order of your images are placed in a way that is going to make sense (we'll see this in a little while when we get to the stars).

If you want a ship to fly in front of the planet then fly behind it...well, that's another story and for a different presentation..

Slide 42: Using our Alpha Masks

Using our Alpha Masks

Now that we are at the point where we are combining the Planet and Cross-hair node networks, we start to see the problem that we highlighted on slide 17. If the image fills the entire resolution (i.e. the background is really black, not transparent) then no amount of merging of the two images is going to work.

By altering the Factor value of the AlphaOver node, we are really only modifying the transparency of the image – and any factor of a black image (other than 0.00 of course) is going to give us some unit grey. Set at 0.50, we are going to get 50% transparency of one image and 50% transparency of the other. Unless the colour in each pixel in both images are the same, we are always going to get a colour modified by a 50% tint of the other image.

The solution then is not the AlphaOver node by itself – we need to chop out the uninteresting parts of the top most image using our Alpha Mask. Once we have done that we can stamp the image down on the next layer without any care about the factor at all.

The easiest method of re-introducing an Alpha Mask to a visible image is using the appropriately named SetAlpha node. Make sure however, that you use the correct alpha image or it will also remove everything that was just added – in this case, we need an Alpha Image that incorporates the glow. Just using the original Alpha will remove the glow as well.

Alternatives

There are always alternatives to the methods that we suggest in this presentation. They may take a few different nodes acting together, but you will always be able to find one. Later, we will see an alternate method of cutting out a hole using the Factor socket of the AlphaOver.

The basic gist of this is simple – experiment and see what methods prove to the most efficient for the results that you need in your scene.

Slide 47: Star Glow

While it may not be easily seen in the images on the slides, the original images of the stars when incorporated into the Planet scene display quite a significant glow around each star. While this is actually quite a nice effect, and could be used quite effectively for distant star fields and nebulae, we have a problem with it in this case.

Our problem is that the glow around the stars is an unintentional by-product of the processing of the planet. What we are getting here is actually the 'atmosphere' being applied to everything in that Render Layer, which includes the stars. As a result, the glow is uncontrollable – well, that is unless you want to also change the glow of the planet itself.

The solution comes right back to the whole idea of splitting the scene up in the first instance – if we place each element into a location that we can control separately (another render layer in the case of the cross-hair, another scene in the case of the stars) then we have full control over all of the post-process that takes place to that element of the final composition.

Slide 48: Star Settings

Star settings in Blender have been called useless (no names) because of their unbelievable nature. In reality however, one of the biggest problems with Blender stars is that users of Blender use them straight out of the box without configuration, and without post-processing them to increase their realism.

The basic default settings of Blender stars tend too be too big, and too dispersed – people are used to seeing stars from Earth, and stars simply do not look like that from Earth. To start with, reduce the both the star size and dispersion.

Next stars tend not to be white – in fact, a lot of objects that you see in the sky may not even be stars at all. The colours of the stars in the sky however, are very subtle – introduce a small degree of colour variation, almost so that it can not be noticed, and things start to look better.

At the end of the day however, even this may not be enough. In our case, it is ideal as we are looking at a stylised 40/50's SF appearance, but for a photo-real environment you will probably want to do some more post work on a star layer. Try adding some nebulous areas and misting and some very good SF scenes can be achieved.

Slide 49: Co-Ordination

While we are using the same camera for our planet and cross-hairs, any camera movement (or object movement) is going to be co-ordinated within that scene. The same is not true however, of the Stars scene. At present, our camera is static – which for now is not a problem as a star-field is a star-field no matter which direction the camera is pointing.

Once you start to introduce animation into the equation however, some form of co-ordination is required. At the very least, camera angles need to be duplicated between the two scenes so that the viewer does not start to see entire planets moving about the sky – the viewer's frame of reference is the stars, so any planet movement is going to look extremely odd. As long as the planet and the stars move in co-ordination, then it is clear to the viewer that it is the camera that is moving.

The simplest method of doing this if we were creating the scene from scratch is to ensure that the camera is linked between scenes. Any movement in on scene will therefore automatically be duplicated in the other. In our current situation, any camera movement would need to be manually duplicated.

Parallax

An interesting technique that I have used is to set up two star scenes and two unique cameras. One scene I regard as the 'distant' stars, while the other I regard as the 'near' stars. By configuring one camera to have a slightly delayed movement in any direction (a dampening), an interesting parallax effect can be achieved.

I'd advise using this technique sparing however – viewers vomiting over their screens is probably not the best way to make friends.

Slide 54: What's with the Planet Alpha?

As we now have a third image layer, we have needed to cut out the planet so that the stars can shine through. We have implemented this in exactly the same way that we cut out the cross-hairs and yet we still have a problems – stars are shining through the edges of the planet.

This actually raises two questions, not one:

- How is this happening?
- Why didn't it happen to the cross-hairs if the process is the same?

We can answer the last question first: it did. The nature of the image and degree of blur involved however, means that the effect is far less noticeable. In fact, it can probably not really be noticed at all.

The answer to the first question we've already hinted at: the blur.

We started off in both cases with an alpha image consisting of a white on black circle with nice sharp edges. We grew it a little using Dilate, then hit it with a Blur node to a value of 30 in both X and Y. Blur however, does not just go in one direction – it has blurred the image on either side of the sharp edge so areas that were pure black in one direction have become various shades of grey, while the white in the other direction have... also become various shades of grey.

While a simple black/white image states full transparency or none at all, grey shades are varying degrees of transparency along the full/none scale. It is this transparency in the mask therefore, which is allowing the stars to shine through around the edges of the planet.

There are two solutions to this – we could try and merge the blurred alpha image with the original alpha image. This would give us an alpha mask that is solid where the planet lies, but blurred as it moves into space. This is a little complicated however, so let's give it a miss – there is a far easier way.

The other method is to add an AlphaOver image to the Stars node network and feed it the image of the stars and the original alpha of the planet as the Factor. Where previously we have used the absolute factor of 0.00 or 1.00 to blend the images together, in this case we are using the image. The node will process the two images pixel-for-pixel – if the factor image (the original Planet Alpha) is black, the stars get through. If the pixel is white, the stars don't get through – only transparency*.

Result – a nice clean hole in the stars.

** An area to note here is that for nodes that have multiple image input sockets, the order in which you connect the image inputs is important. For example: AlphaOver nodes where two images are connected one way with a factor of 1.00 will produce the same result as the two images connected the opposite way around with a factor of 0.00.*