INNOVATIONS PROJECT:
NicK Hales

A Beginners Guide To Building A Fur Network

Contents:

INNOVATIONS PROJECT:
NicK Hales

A Beginners Guide To Building A Fur Network

The Basics

The **Hypershade** is the central area when it comes to rendering *(Fig. 1)*. In the Hypershade, textures are built by combining 'Nodes', each of these provides different effects to the overall texture when linked together, and can be combined in many different ways to form a **Network**.

These networks are connected together in the Hypershade with coloured lines that symbolise the connections between attributes. Highlighting these can help to more clearly see the connections made when networks become crowded. These **Nodes** are found in the option bar on the far left hand side of the Hypershade.*(Fig. 2)*

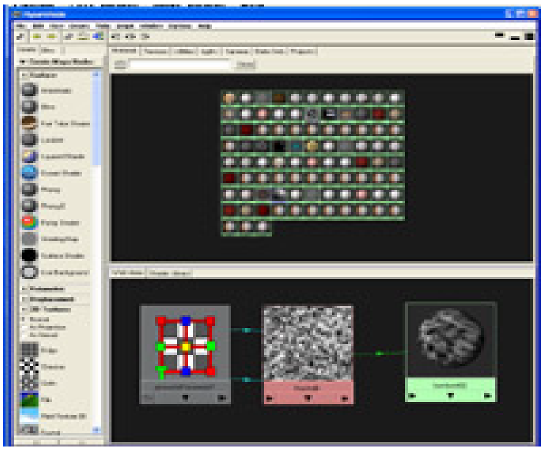


Fig.1

The Hypershade Window overview Found within Maya.

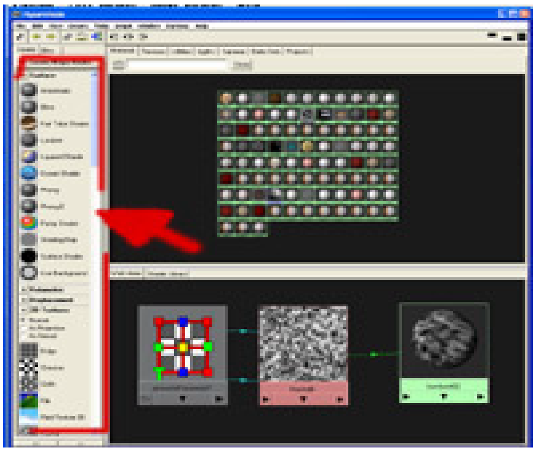


Fig.2

Where to find the Nodes in the Hypershade

Nodes in Maya are the building blocks to create any form of texture or colour for an object. They are divided into different categories dependant on their uses. The areas being looked into in these tutorials are the **Surface Nodes**, **Texture Nodes** and **Utility Nodes**.

Surface Nodes are the default *'materials'* in Maya. These all have different properties to suggest different looks for an object. These range from a **Lambert Node** that gives a flat matte look, to a **Phong Node**, which creates a shiny reflective appearance. These are the basis for any network to be created.

Texture Nodes are more sophisticated **Nodes** that add detail to the surface of an object. These are very important to the final look in a network and have the most options and variations of all the **Nodes**. Hence these are extremely useful in creating different appearances on the surface.

Utility Nodes basically determine how textures are applied and combined within networks. These can create interesting variations to textures and help add a degree of control to the final outcome.

All these will be explained in greater detail when applied in the network tutorials.

How To Connect Nodes

Here is an example of how to connect two **Nodes** together to form the most basic of networks.

- Open the Hypershade and Select a **Surface Node**, then select a **Texture Node**, both from the toolbar at the left of the screen. *(Fig. 3)*
- Drag this **Node** (using the middle mouse button) on top of the **Surface Node** and a box of connection option will appear *(Fig. 4)*.
- For this example pick the Color attribute (left mouse button) and you will find the **Nodes** now have a connecting line between them and **the Surface Node** has 'inherited' the texture as it's new colour.*(Fig.* 5*)*

These are the basic steps of how networks are created.

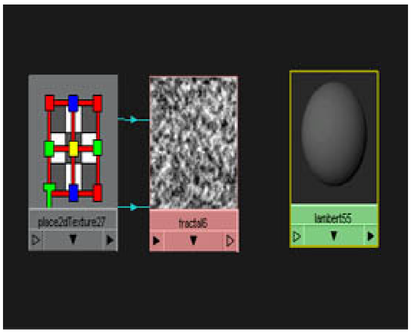


Fig. 3

Texture & Surface Nodes

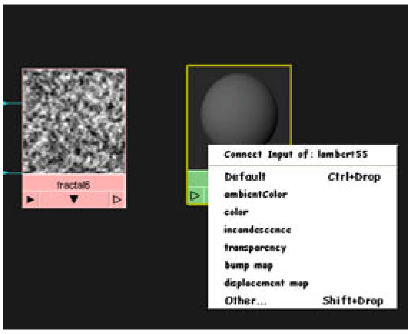


Fig. 4

Right-Click Connection Menu

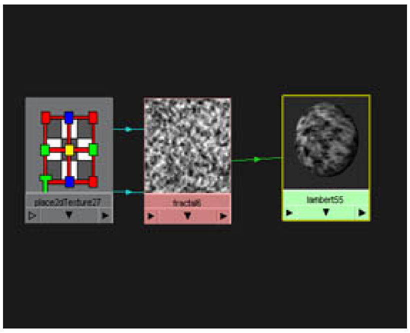


Fig. 5

Outcome: Connected Network

This works with the simple networks and default connections, such as the above. But for more complex networks a tool known as the ***Connection Editor*** must be used. This allows a greater range of connections and so helps create far more interesting results.

Connection Editor

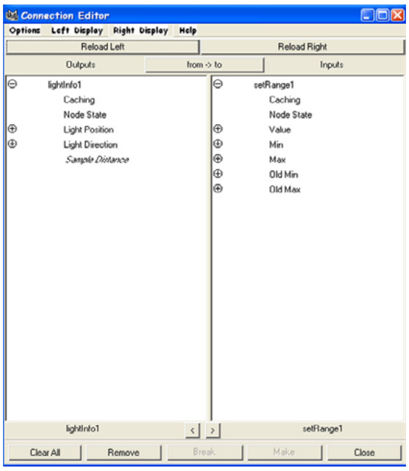
The Connection Editor is the best tool for fine tuning a network and making non-default connections.

To open the Connection Editor do **ONE** of the following:

- Click Window > General Editors > Connection Editor.
- Double click on a connection line.
- Shift & middle-mouse button drag a **Node** onto another **Node** in the Hypershade.

Once the Connection Editor is open, select the Output attribute from the left-hand side and then an Input attribute from the right hand side to connect the two. *(Fig. 6)*

When an Output attribute has been selected only the Input attributes that can be connected to this will be shown, the others will be dimmed to grey, although clicking on the **(+)** symbol by one can open a list of attributes nested inside that may be selectable.*(Fig. 7)*



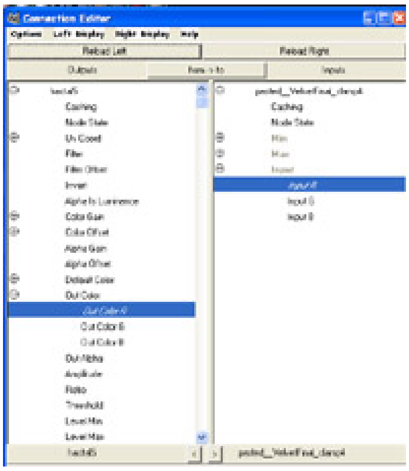


Fig. 6

The Connection Editor

Fig. 7

Possible Connections to be made

The starting point for these networks is always a **Surface Node**. These determine the type of surface the object will have.

Fur - The Breakdown

In the process of creating fur it is important to think about the individual components that create its distinctive feel. These have been narrowed down to three key elements, each demonstrated with a tutorial:

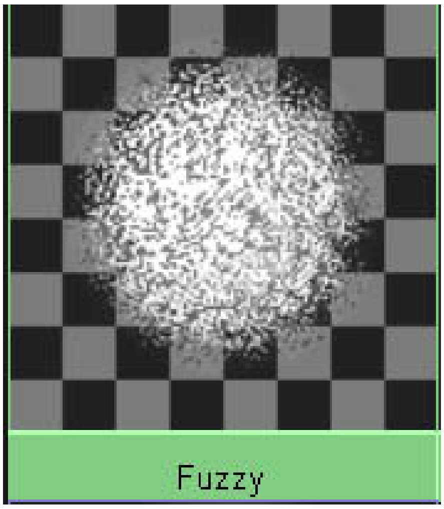
- Transparency
- Movement
- Softness

These networks can be combined or used separately to better describe the look of fur required by the user. Each of these is to be explored in turn.

TO NOTE: In the following tutorials some of the **Node's Place2D/3D Texture Nodes** (See Node Definitions) will be deleted. This is purely for visual simplicity reasons and will not effect the networks if not removed.

Transparency Network

This area of the network rather logically works through the transparency channel of the **Surface Node**. The most interesting examples of how this network can be developed are effects for objects such as artist style pointillism and a particle look. For this purpose here are the basics to create a fuzzy haze of transparency around the **Node**.



This works on a system that means the colour/texture fades around the edges, ideal on a simple sphere and for more abstract images.

Although this network works through two channels, the Bump and Transparency *(Fig. 8)* the important effect is generated in just one, the transparency channel *(Fig. 9)*, the other merely effect the general texture:

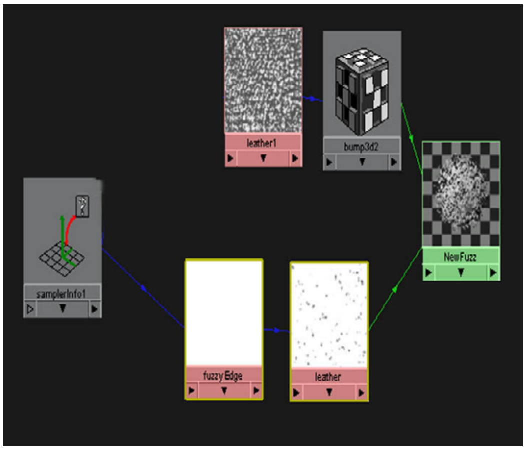


Fig. 8

The final network layout Working through the Bump and Transparency channels

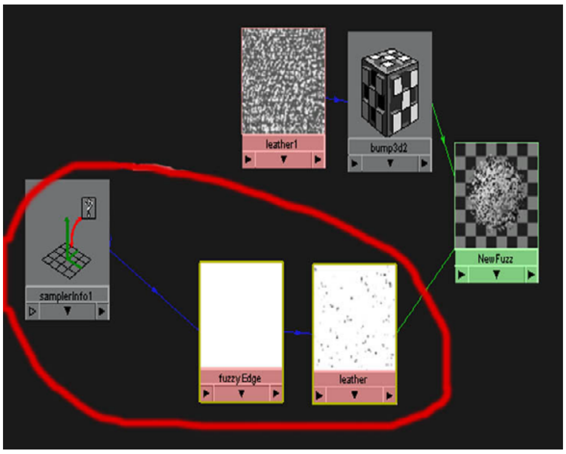


Fig. 9

The selected area needed For Fuzzy Outline only using the Transparency channel

This transparency area of the network is created with a sequence of just 3 **Nodes**, but gives a very nice effect.

The first of these **Nodes** uses the transparency channel of the **Surface Node** directly.

- In the Hypershade create a Lambert **Surface Node**.
- Next to it create a Leather **3D Texture Node** and connect it to the **Surface Node** through the transparency channel.
- Delete the Leather's attached **Place3DTexture Node**.

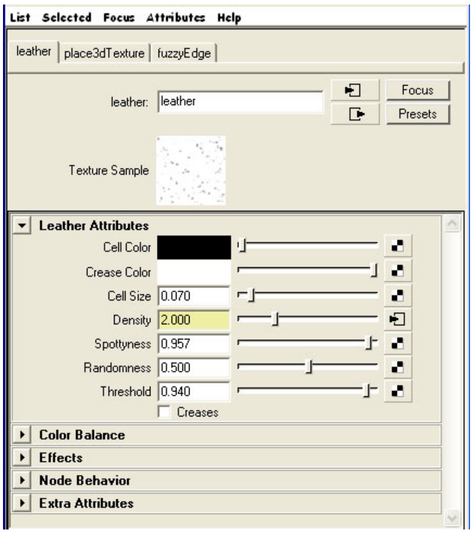
This **3D Texture Node** (See Node definitions) creates a pattern texture all over the object and so is ideal to use as a basis for a constant effect. The next step is to edit a few of the default Leather attribute values:

- In the Leather Attribute menu set the Cell Color to black and Crease Color to white. This gives the maximum contrast. (*Fig. 10)*
- Check that the Color Attributes are set to Color Gain white and Color Offset black.
- Change the Cell Size to 0.07, the Spottyness to 1 and the Threshold to 1.
- Un-tick the Creases option box and in the Effects menu tick the Inverse option box. This will mean the texture is now mostly white, and hence allow a solid visible centre to the **Surface Node** when in the network.

This **Node** creates the 'speckled effect' seen on the finished surface**,** but it is the Leather's Density channel that is used to achieve the fading edge effect.

In this example the density controls the amount of white seen through the black, and to create the fading effect a **Ramp Node (See Node definitions)** is used through this attribute.

- Create a **Ramp Node** and delete its **Place2DTexture Node**.
- Using the Connection Editor, (Shift & middle-mouse button drag the **Node** onto the other) connect the Ramp's Out Alpha to the Leather's Density channel, as shown previously. As this network is used through the transparency channel the black to white colour change acts as a true or false control for the objects visibility.
- Make sure there are only two colours on the Ramp with the upper most one set to black, and the lowest to white and position as seen in the diagram. *(Fig. 11)* This means that with a Ramp from black down to white, the amount of black to white ratio determines how solid the object will look.



leather: leather

Leather Attributes

Cell Color ■

Crease Color □

Cell Size 0.070

Density 2.000

Spottyness 0.957

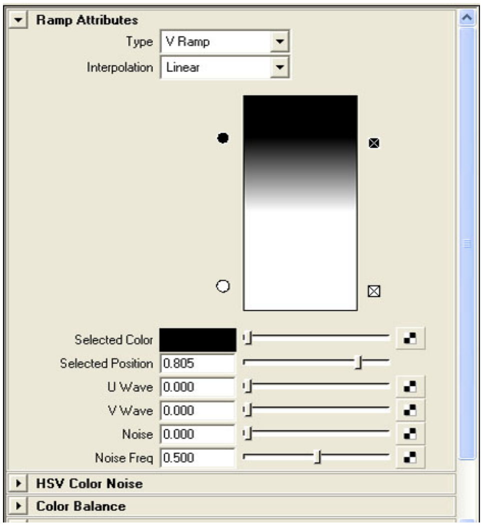
Randomness 0.500

Threshold 0.940

□ Creases

Fig. 10

The edited Texture attributes set to Black and White extremes.



Ramp Attributes

Type V Ramp

Interpolation Linear

Selected Color ■

Selected Position 0.805

U Wave 0.000

V Wave 0.000

Noise 0.000

Noise Freq 0.500

Fig. 11

The adapted Ramp Node

- To finish this effect create a **SampleInfo Node**
- Attach its FacingRatio attribute into the Ramp's V Co-ordinate attribute (Found in Uv Coord) to produce an even spread over the entire **Node**, again using the Connection Editor.

This, as an attribute of the speckled Leather texture breaks up the object to produce the fuzzy particle like edging to the surface. As less of the 'dots' the texture pattern is made up of are visible around the outside (Due to the Ramp gradient) so it appears to fade away.

This produces the effect around the edges, but to create a fuzzy look to the whole **Node** there is one more step.

- Create another Leather **Texture Node**, again delete its **Place3Dtexture Node** and this time attach it via the **Surface Node's** Bump attribute. Simply alter the Cell and Crease colours to your desire to determine the effect wanted. This should create something similar to the finished network. *(Fig. 8)*

Visibility Network

An interesting area that runs parallel with transparency is the visibility of a texture. This network works on just this, this tutorial was discovered on the internet and has been included in these tutorials as it is a useful addition in the area concerned and produces an interesting effect. *(Fig. 12)*. This would be very useful for a texture of an object in a large scene that, for example starts in the background and ends in the foreground. While small on screen the texture details would not need to be seen and hence save on calculations and simplifying the overall look. Yet when moving to the foreground would seem to become more clearly visible and so more detailed as it got nearer, much like in real life, creating in many respects a false focus:

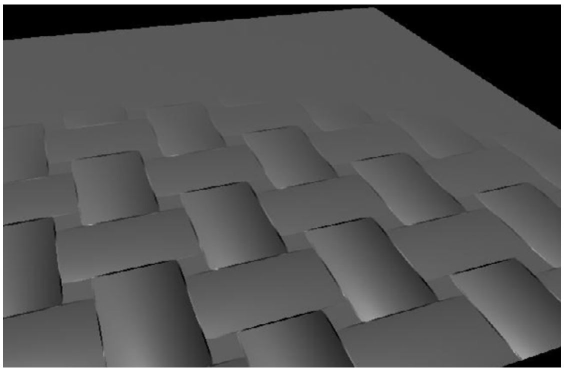


Fig. 12

Results from the Visibility Network

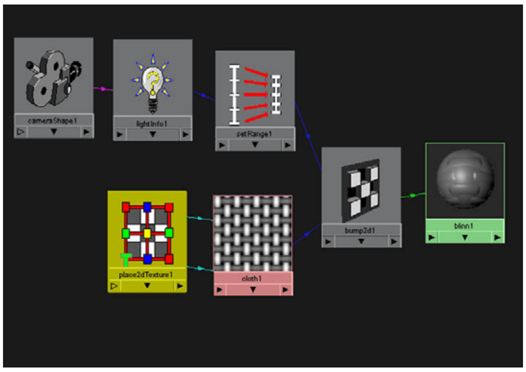


Fig. 13

The Visibility Network for this example

This type of network *(Fig. 13)* is again made up on simple principles. This network uses the **LightInfo Node** to calculate the distance of each point on the surface to the camera. This data is then used to drive the Bump value of each point, the further from the camera, the less the Bump effects the object. **SetRange Node** is used to clamp this calculated distance down to a range appropriate to the object.

Here is a quick break down using the tutorial's illustrations:

- To try this out in a simple demonstration set up a scene as shown below, needing only a default camera and for the purpose of this tutorial, a NURBS plane to be created:

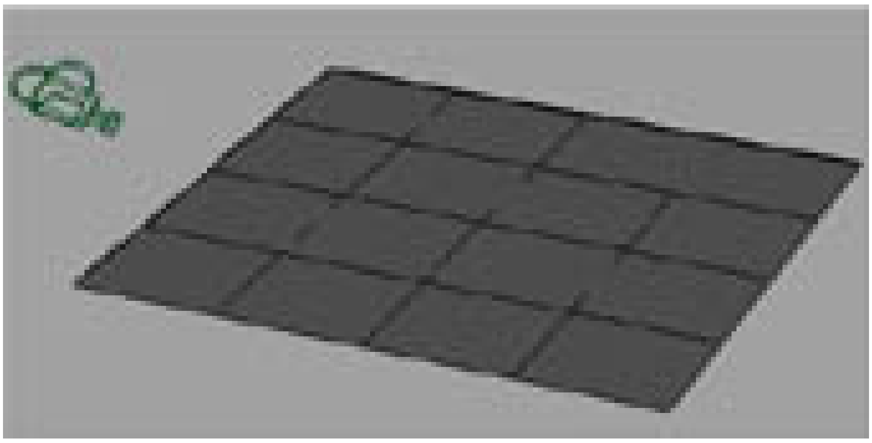


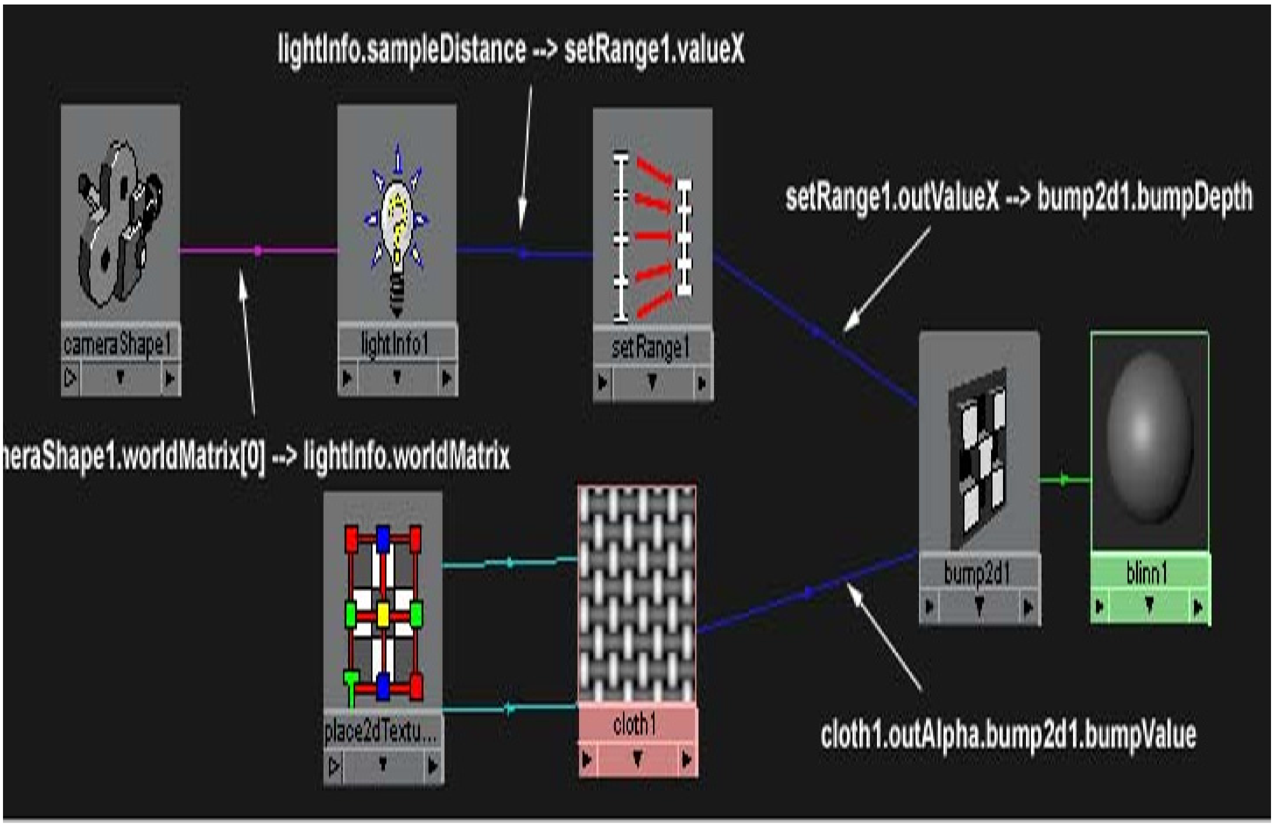
Fig. 14

The rough layout for the test scene, involving camera and plane

- Once this is set up, open the Hypershade and create the following **Nodes: LightInfo Node, SetRange Node** and a **Blinn Surface Node**, with a **Cloth Texture Node** created through it's Bump Mapping attribute by selecting the box to the right of this attribute. These should be shown in the Work Area window with the **CameraShape's Node**.

- To create the **CameraShape Node**, in the top window of the Hypershade select the Cameras tab and middle mouse button drag the **Node** into the Work Area window.

Connect these in the manner shown below using the connection editor:


lightInfo.sampleDistance --> setRange1.valueX
setRange1.outValueX --> bump2d1.bumpDepth
eraShape1.worldMatrix[0] --> lightInfo.worldMatrix
cloth1.outAlpha.bump2d1.bumpValue
cameraShape1
lightInfo1
setRange1
bump2d1
blinn1
place2dTextu...
cloth1

In case the previous diagram is not clear the connections are re-written here:

- CameraShape.worldMatrix[0] → lightInfo.worldMatrix
- LightInfo.sampleDistance → setRange.valueX
- SetRange.outValueX → Bump2d.BumpDepth
- Cloth.outAlpha → Bump2d.BumpValue

Finally set the **SetRange Node's** attributes as follows to complete the network:
Min 1, Max 0, Old Min 5 and Old Max 23.

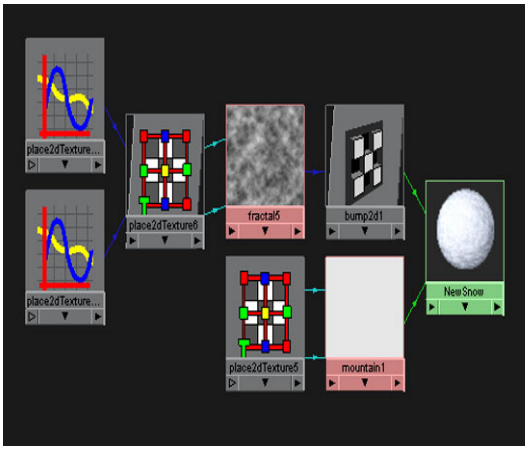
Simply IPR Render from the camera view to see the final look. *(Fig. 12)* Moving the camera closer and further away to test the effect.

Movement Network

Within fur there is always some degree of movement whether the object it lies on is stationary or in action. This network will quickly create a simple effect of this for the user.

This area of fur comes down to simply one **Node** in practice. This is the **Anim Curve Node** (See Node Definitions). The following network will show how this element is included using the example of snow, another area where this could be used effectively.

The network for this **Node** is a fairly short *(fig.15)*, as in the Fuzzy example. This makes it a good one to start with when first trying out network connections as it only consists of two **Texture Nodes**, one **2D Bump Map Node** and the crucial **Anim Curve Nodes** *(Fig. 16)* yet it creates a rather impressive final effect for such a simple network.



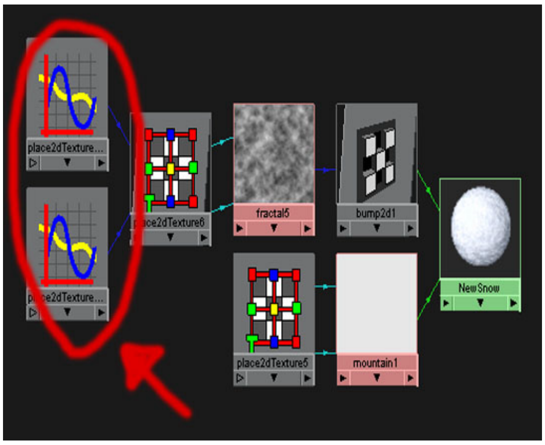


Fig. 15
An example of the Movement Network

Fig. 16
Anim Curve Nodes in the Network

These **Anim Curve Nodes** tell the texture to move around the surface over time to give a sense of movement when the object is rendered. An important note is that these **Nodes** DO NOT exist in the selection box of the Hypershade with all the other **Nodes**. They also work in a slightly different manner than **Nodes** previously mentioned, as they need to be manually keyed in the scene to be initially created in the network.

An example of how to create these is as follows:

- Create a Lambert **Surface Node.**
- Next to this create a **Fractal Texture Node**, and connect it as the Bump map of your **Surface Node.**
- Reduce the Fractal **Node's** Amplitude to around 0.15.
- Select the **Place2D Texture Node** attached to the **Fractal Node** and find its Offset channel in the attributes. This consists of two boxes, which describe the U and V Co-ordinates of the texture map.
- To create an **Anim Curve Node** for the U Co-ord. simply select the time you want the movement to start in the timeline, and type a starting value in the Offset box. Next right-click over the box and select the Set Key option from the pop up menu. *(Fig. 17).*
- Next move to another position on the timeline and again enter a value in the same box and set the key as before. This will now create a movement curve between these two values over the time.
- After these values have been created, select the **Surface Node** and with the right mouse button select and hold. In the pop up menu that appears select the Graph Network option to see an **Anim Curve Node** has been automatically created at the end of the Network. (Fig. 16)

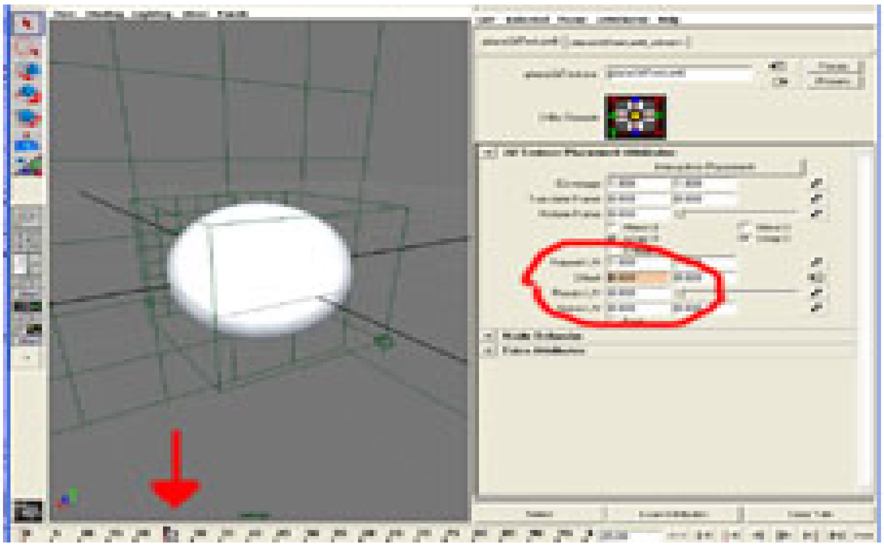


Fig. 17

The keyframe location on the timeline & the keyed attribute

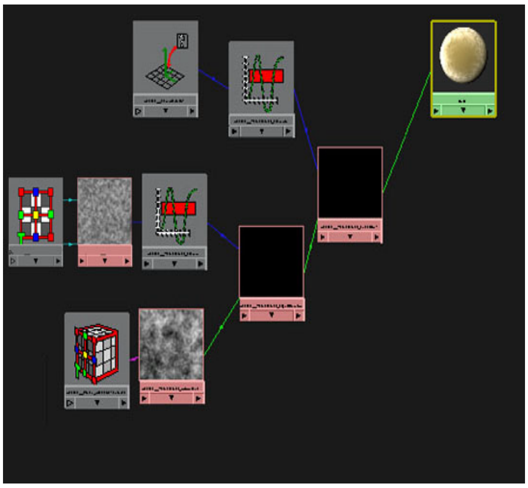
This technique is all that is needed to create general movement. Although for more subtle effects an **Anim Curve Node** is needed for both the U and V Offset channels, both created in the same manner shown. The description of each should be carefully tailored for the level of movement desired.

Softness Network: To Create A Fur Texture For Stills

In comparison to the previous tutorials, the network to create a decent effect for this texture is far more complex. It involves a few similarities to the transparency network but also includes a few **Nodes** not yet mentioned. This network is a good starting point that after being created can be easily edited to adapt for the new animation friendly effect that follows.

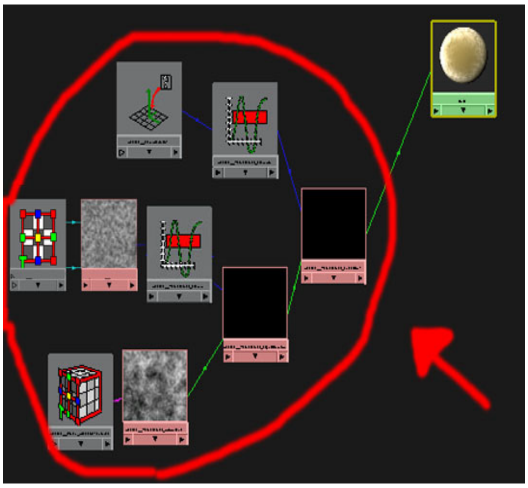
The graph network may seem a little illogical at first but when broken down in stages shows how the number of steps combine for the final look. Firstly, this element of the texture works through the Incandescence channel of the surface. This channel by default controls how much light is emitted/reflects off the surface which will result in how much of the colour is shown and how much is the reflected light colour, default being white. This is one of the most sensitive channels and hence why such a precise network is needed to obtain the correct results (*Fig*. 18). In this case the whole network is important, unlike the previous tutorials no one part can make this work.*(Fig. 19)*

Fig. 18



The Furry Network laid out

Fig. 19



The whole network needs to be created

The first step of this network uses an expansion of the transparency network. This involves a mini network of Ramps for much the same effect as the transparency, using Ramps feeding into Ramps to create more interesting patterns/effects. *(Fig. 20)*

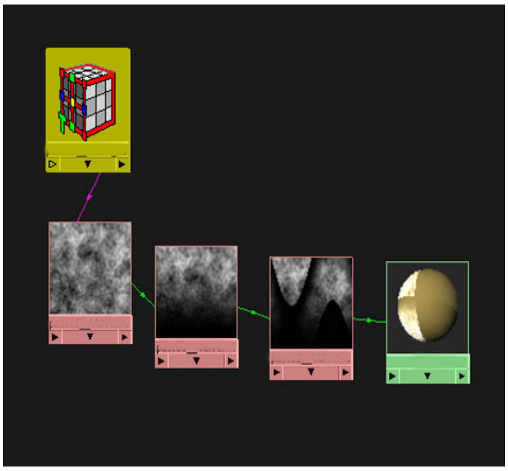
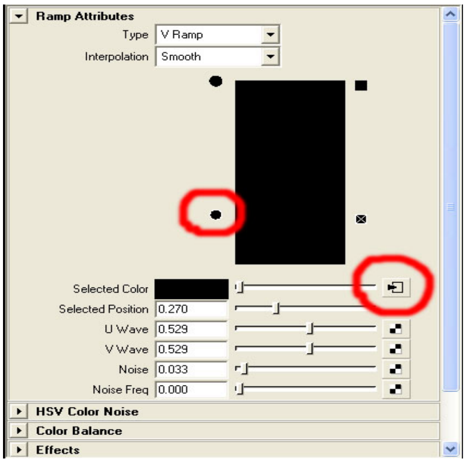


Fig. 20

The Ramp Nodes connected in the network



Ramp Attributes

Type: V Ramp

Interpolation: Smooth

Selected Color

Selected Position: 0.270

U Wave: 0.529

V Wave: 0.529

Noise: 0.033

Noise Freq: 0.000

HSV Color Noise

Color Balance

Effects

Fig. 21

The attributes of the 1st Ramp

- Create a Lambert **Surface Node.**

- Create next to this a **Ramp Node** and attach it to the Lambert via the Incandescance attribute. Delete the Ramp's **Place2DTexture Node**.

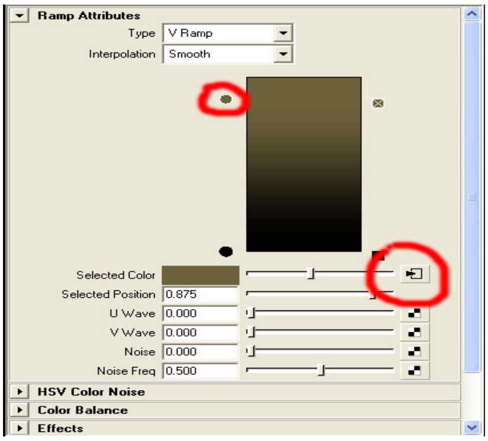
- Edit the Ramp's colours, making sure there are only 2 which needs to range from black to black to keep a solid texture. Also change the Interpolation of the Ramp to Smooth.

- Set both the U and V Wave attributes to 0.5 to adjust how the texture will eventually lie on the **Surface Node**. Finally set the Noise attribute to 0.03 and the Noise Frequency to 0. This will help create the highlights for the final effect.

- Select the lower (2nd) black colour to use as an input connection for a 2nd **Ramp Node** by selecting the option box to the right of the Color attribute and selecting a **Ramp Node** from the pop up menu.*(Fig. 21)* Again delete the Ramp's **Place2DTexture Node** for simplicity.

- This 2nd **Ramp Node** should also be set up as the previous one, with 2 colours, both set to black. This time instead of the lower colour being used as an input, use the upper one by the same method as above. *(Fig. 22)*.

- Into this create a **Brownian 3D Texture Node** to give a soft/varied look to the Ramp, (as with the transparency) and eventually to the surface itself. The values for this can vary, dependant on the colour of the object and the fur needed. For this tutorial the default attributes remained unchanged. *(Fig. 23)*



Ramp Attributes

Type: V Ramp

Interpolation: Smooth

Selected Color

Selected Position: 0.875

U Wave: 0.000

V Wave: 0.000

Noise: 0.000

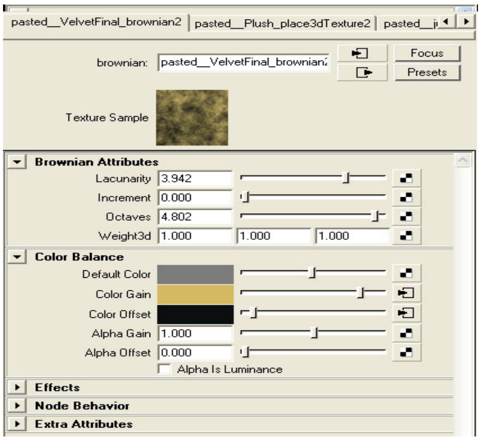
Noise Freq: 0.500

HSV Color Noise

Color Balance

Effects

Fig. 22



pasted__VelvetFinal_brownian2 | pasted__Plush_place3dTexture2 | pasted__j

brownian: pasted__VelvetFinal_brownian2

Focus

Presets

Texture Sample

Brownian Attributes

Lacunarity: 3.942

Increment: 0.000

Octaves: 4.802

Weight3d: 1.000 1.000 1.000

Color Balance

Default Color

Color Gain

Color Offset

Alpha Gain: 1.000

Alpha Offset: 0.000

☐ Alpha Is Luminance

Effects

Node Behavior

Extra Attributes

Fig. 23

The Attribute of the 2nd Ramp needing To be connected to the texture Node.

The settings of the Texture Node used for this particular effect.

The next step in this network is to add a crucial **Node** to the network, the **Fractal 2D Texture Node** (See Node Definitions). This important **Node** gives 'life' to the texture by raising the light/luminance of the texture in the desired areas using the rough fractal pattern.

- Create a **Fractal 2D Texture Node** and a **Clamp Node.**
- Connect the **Fractal 2D Texture Node's** OutColor R into the **Clamp Node's** Input R, so that this degree of luminance can be controlled and limited easily.
- The **Clamp Node's** Color attributes may need to be edited, with the R Min value remaining at 0 and the R Max value set to 1 to create the range.
- Next connect the **Clamp Node's** Output R to the 2nd **Ramp Node's** V Coord channel to take effect. *(Fig. 24)*
- The **Fractal Node's** Amplitude attribute is used to control the amount of highlights around the edges of the Fur Effect. The lower the value the softer the look.

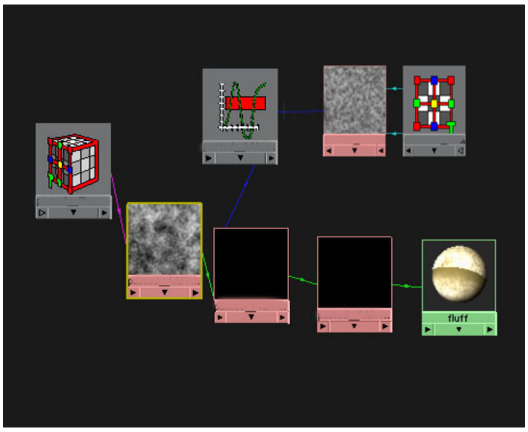


Fig. 24

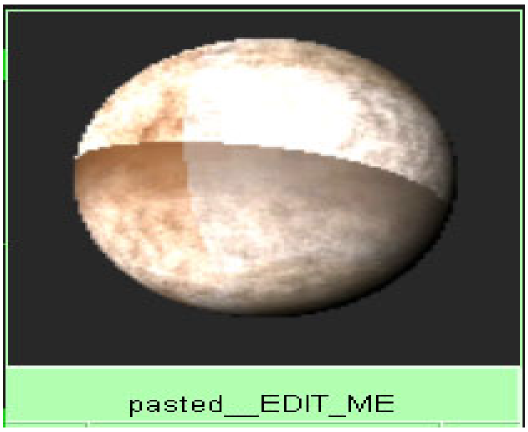


Fig. 25

The Network so far

The resulting Surface Node

Although this will give a haze effect around the edges of the surface the co-ordinates are now offset and so the texture does not 'sit right' over the surface *(Fig. 25)*. This is rectified by inserting another couple of **Nodes**, as demonstrated in the next step.

This step in the network follows similar lines as the previous. This time the **SamplerInfo Node** (See Node Definitions) is the decisive step to correcting the Uv displacement created above.

- Create a **SamplerInfo Node** and a **Clamp Node**.
- This **SamplerInfo Node's** Facing Ratio attribute should be connected as an input to the new **Clamp Node's** R Input using the Connection Editor.
- Again the **Clamp Node's** Color Attributes need to be edited to create a range for the R values. With Min set to 0 and Max set to 1.
- This **Clamp Node's** Output R attribute needs to be connected to the V coordinate of the 1st **Ramp Node** to complete the network.

All these combined in the network *(Fig. 26)* produce the desired soft effect. *(Fig. 27)*

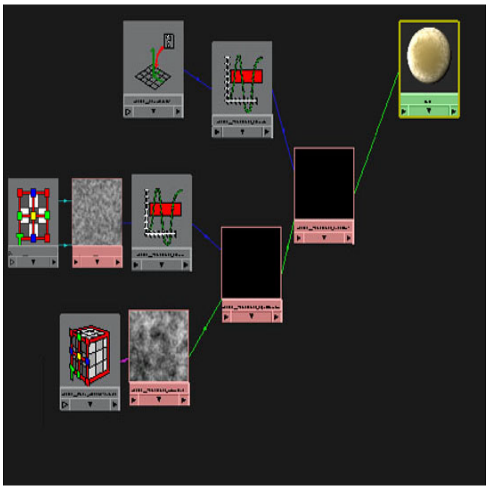


Fig. 26

The Final Network Layout

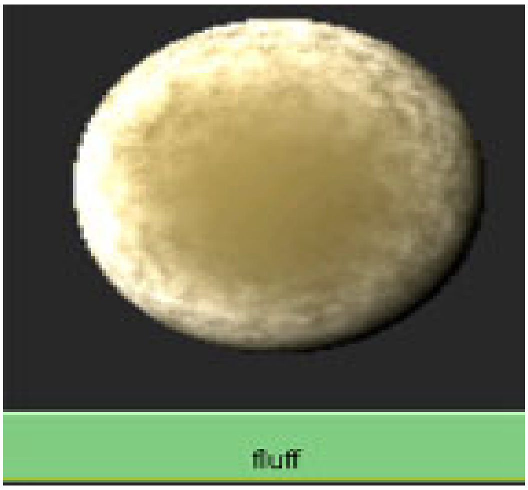


Fig. 27

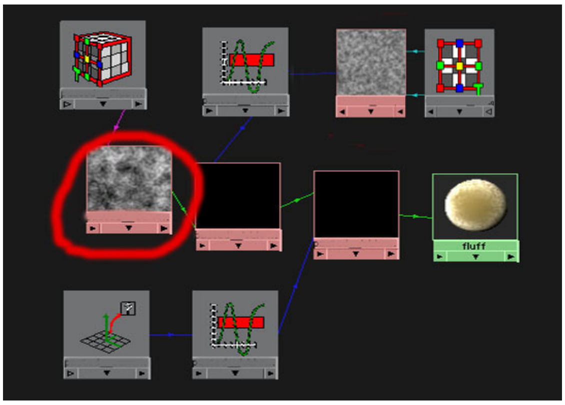
The resulting Final Surface Node

These networks, in a proverbial nutshell form the basis for a soft short fur texture, which can be taken further for more specialist needs.

Softness Network: Adapting The Fur Texture For Animation

Although the above texture *(Fig. 27)* will not work for a moving object, with a few modifications this can be resolved to create a similar look that sticks to the object, however it is manipulated.

The only area that has to be changed in the network is the **Brownian Texture Node**.*(Fig. 28)*



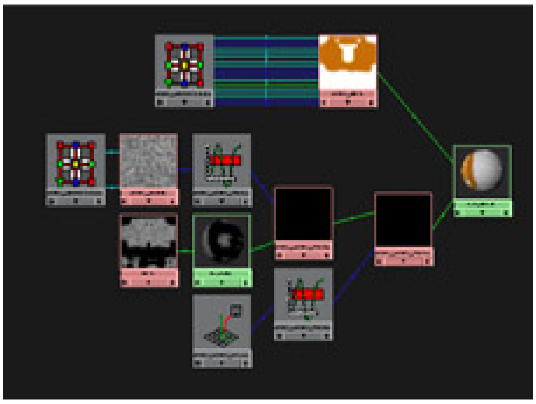


Fig. 28

The Original Static Fur Network with the area to change selected

Fig. 29

The completed New Fur Network with file input flat colour added

Step 1)

For this step leave the Hypershade and move into another important area of Maya, the UV Texture Editor. The UV Texture Editor can be opened by selecting:

Window>UV Texture Editor in the Maya menu bar.

To see what the Texture Editor provides in Maya select the object. This shows a wire-frame map of it represented inside the Editor. This is basically a map of the object's faces laid out flat in a 2D view, and shows how any colour texture created *(Fig. 30)* will sit, and cover the object *(Fig. 31)*. The area of UV mapping and the creation of the best maps possible is a large subject in itself and usually task specific. For this reason these useful sites have been included to show the basics, should there be the need for detailed maps on complex objects. On more simple or default objects this may not be needed:

This link is a step by step tutorial for applying UV texture maps to a car object and taking them through to Photoshop to create a flat colour texture. Covers all the steps needed for any complex object:

http://web.alfredstate.edu/ciat/tutorials/TexturingPolygons.htm

This link from Eliteops.com shows one way of creating a good texture map for a complex area such as a character's face:

http://www.eliteops.com/sama/uv_tutorial/index.htm

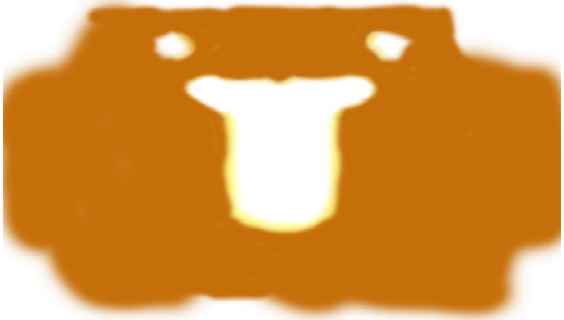


Fig. 30

A simple example of the flat colour texture created in Photoshop

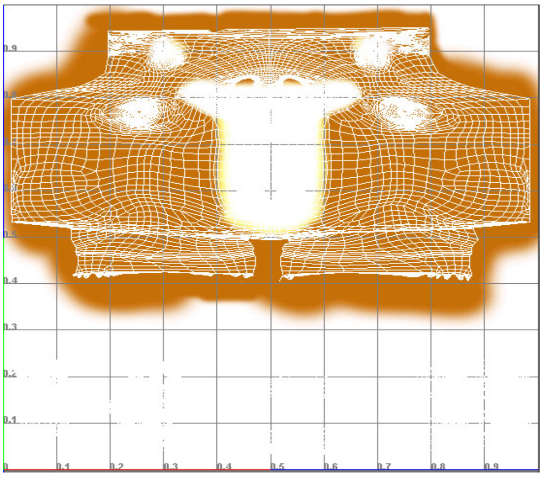


Fig. 31

A finished UV map laid over the colour texture showing how the process works

Step 2)

- After sorting out the UV map return to the Hypershade. Select your original **Surface Node** and bring up its network. (Right clicking over the **Node** and selecting Graph Network)
- Create another **Lambert Node** and position it next to the existing network in the Work Area, this will be used to form a new temporary network.
- Next delete the connection line between the **Brownian Texture Node** and the **Ramp Node** in the original network to isolate the **Brownian Node.** Do this by highlighting the connection line by selection and hitting the delete key.
- Attach this now free **Brownian Node** to the new Lambert via the Color channel. *(Fig. 32)*
- Finally for this step select the **Surface Node** of the original network, press and hold the Right Mouse Button over it and in the option box choose **Select Objects with Material**. When the selected object is highlighted press and hold the Right Mouse Button over the newly created **Surface Node** and select Assign Material To Selection so that the object now takes on the look of the new surface.

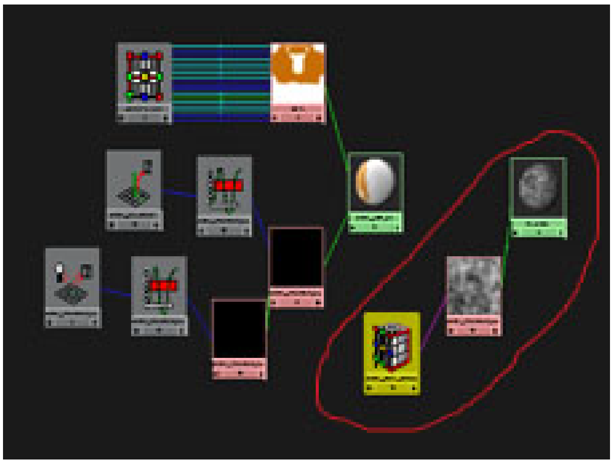


Fig. 32

The new temporary network to be assigned to the object in the Baking process

Step 3)

This step is the most important to the network working. Before attempting this, here is why. The problem in the previous network was that the texture would not move with the object, this was due to the 3D texture of the **Brownian Texture Node**.

A 2D texture wraps itself onto the object where as the 3D texture would simple project through the object in the 3D space. This would look fine in a rendered still but the object would simply move through this projection when moved. To counter this problem a method known as **Baking** must be used.

This **Baking** method in essence pre-renders the texture. This means the texture that is projected over the object is frozen at that moment into an image file and mapped evenly over the object's UV map. (This is why a good UV map is important for this stage to work effectively.) The other reasons for baking of the texture are that it helps to accelerate render time and needs less processor power when working.

To Bake the texture:

- First, in the Maya window highlight the object the texture is assigned to and Shift select the **Brownian Texture Node** itself in the Hypershade. With these highlighted, in the Hypershade menu select Edit > Convert to File Texture, selecting the option box.

- Within the options make sure only the Fill Texture Seams box is ticked. This allows the computer to artificially extend the texture boundaries where the UV maps may meet, leaving no nasty edges in the texture on the surface. The UV Range should be set to Default and the X and Y Resolutions are set to 512 or higher to make sure a detailed image is taken. The file format should be set to IFF by default.

- After this has been done select Convert and Close.

In the Hypershade, next to the Brownian **Texture Node** a new **Surface Node** and a **File Texture Node** have been created, in the shape of the object's UV map. *(Fig. 33 & 34)*

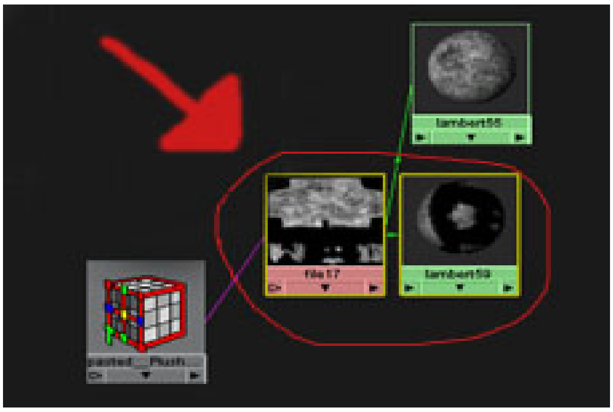


Fig. 33

The resulting network from the Baking process. Texture is 'frozen' into position Over the object's UV map

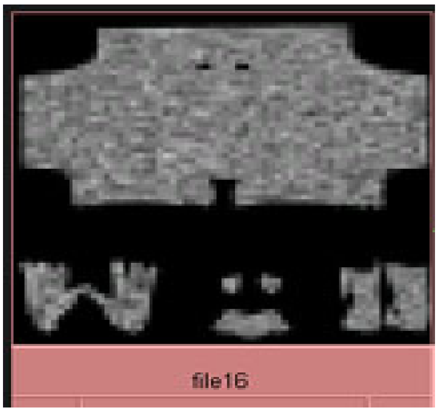


Fig. 34

A closer look at the now mapped Brownian Texture over the object

Step 4)

- The final step is to re-assign this new **Surface Node** to your original network.

- Firstly press and hold the Right Mouse Button over your new **Surface Node** and Select Objects With Material. This should select the object. (If this does not happen select the object and use the attribute editor to find out which **Surface Node** it is assigned to.) Once the object is highlighted assign it back to the **Surface Node** of the original network.

- Now select the newly created **Surface Node** and connect it back into the second **Ramp Node** in the network, where the **Brownian Node** was once connected. Do this by connecting the **Surface Node's** OutColor to the Ramp's ColorEntryList[0].Color attribute in the Connection Editor.

This restores the desired appearance of the network and mean this furry texture will now move with the chosen object. *(Fig. 35 & 36)*

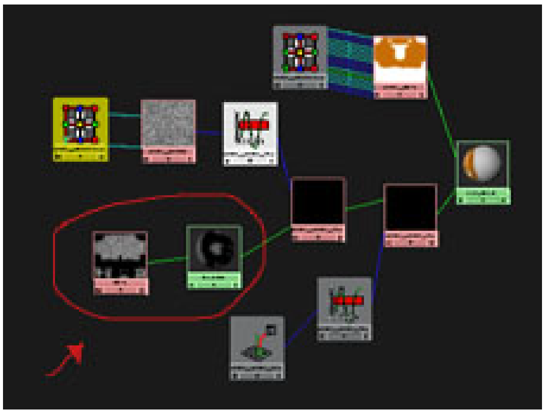


Fig. 35

Restoring the Brownian Texture to the Network using the baked Node

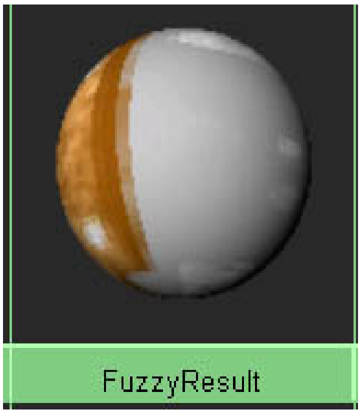
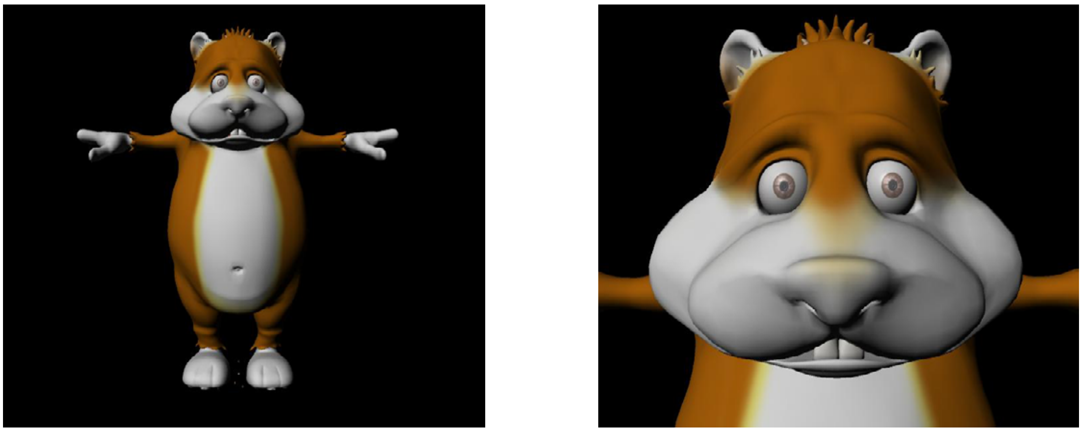


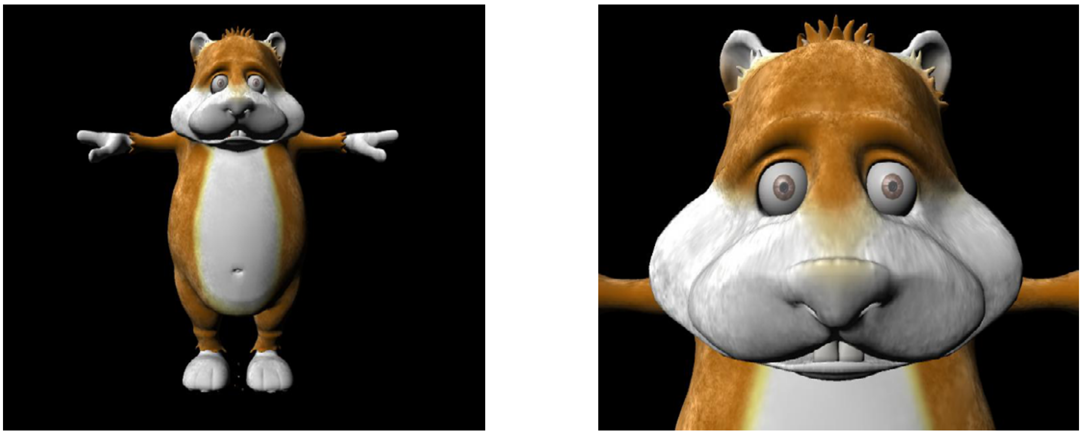
Fig. 36

An example of the look of the finished Surface Node

Here are a few shots with a complex character to demonstrate the difference this network can make when combined with a simple flat colour:



Character with only the flat basic colour texture

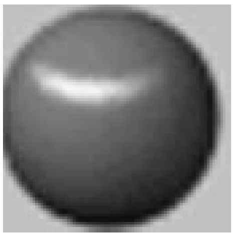


Character with the new network on top of the colour texture

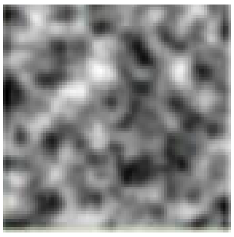
Important Note

- The final note is that this network is object specific and will only work with the UV map of the object used (See Step 3). To use this network on another object it would simply be a process of repeating this Step with the new UV map.

NODE DEFINITIONS

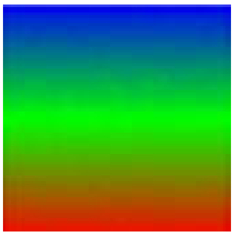


Surface Node These materials represent the types of surfaces which textures can be mapped onto. These have attributes such as shininess, matte, reflectivity etc. which vary between the different materials.

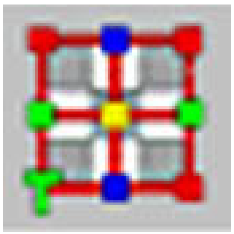


2D & 3D Texture Nodes These texture Nodes are the defaults in the Hypershade (Or allow a file textures to be used) These define how the surface will appear when rendered. These each have a set of attributes that can be used to create variations and can be easily edited.

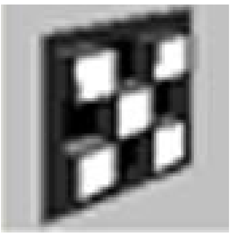
- 2D Textures wrap round the object like they are painted on flat.
- 3D Textures project through objects to add a sense of depth to the object.



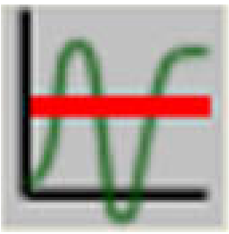
Ramp Node This Node simply creates a gradient through a series of colours the user defines. The default ramp is blue/green/red. This Node can be used to create different types of effects like stripes and patterns over a surface or as a 2D background. Feeding other Texture Nodes into the colour channels will give more unusual and complex patterns that can be used to great effect. Editing the U and V wave attributes allow control of a sine wave through the texture in the two directions. This ranges from 0-1 with 0 meaning no wave and 1 creating the maximum wave.



Place 2D Texture Node This Node defines a texture 'frame'. A rectangular area on the surface in which the UV co-ordinated of the object lie. This allows control of the position, size and angle of the texture to be placed on the object. This can also be done using the UV texture editor within Maya if a more hands-on approach is required.



Bump 2D Node Converts 2D textures into Bump maps. The two main attributes of this Node are the **Bump Value** that contains the source used for the Bumps; and the **Bump Depth** which controls how high the Bumps will appear on the surface, the larger the value the Bumpier the surface when rendered.



Clamp Node is a utility Node that can be used to keep a value (attribute) within a specified range. The usual example of this is for colour. Input values that are outside the range are "**clamped**" back to the allowed range.

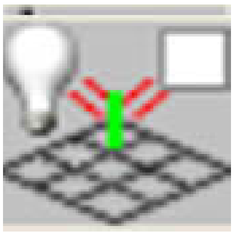
Eg. Setting MinRed to 0.3 and MaxRed to 0.6, these outputs would result from these inputs:

InputRed:	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
OutputRed:	0.3	0.3	0.3	0.4	0.5	0.6	0.6	0.6	0.6

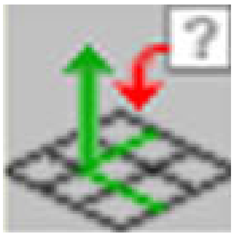
Light Info Node can be used to obtain information about the position of a light relative to a texture. As each point on the texture is being shaded, this tells you the precise distance from the light to the point being shaded. It also returns other information that can be useful when creating a shading network.

To make this work, you must connect the ‘World Matrix’ attribute of the light source to the ‘World Matrix’ attribute of the Light Info Node. Once that is done, the Light Info utility Node is ready to use. Using this information as the input to other shading Nodes produces interesting effects.

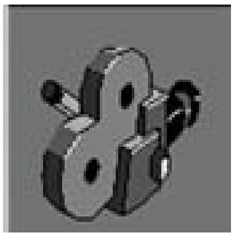
Eg. Connect the ‘**Sample Distance**’ attribute to one of the colour channels of a shader. That way, the colour of the shader will change depending on how far it is from the light.



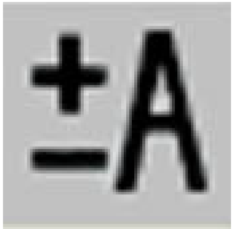
SurfaceLuminance Node calculates the luminance (brightness) of a point on a surface as it is being rendered. This takes into account all light sources shining on the object, and the angle at which they shine on the object. This allows interesting shaders to be made that change based on the light in the environment.



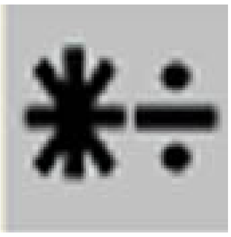
Sampler Node provides information about each point on a surface as it is being sampled or calculated for rendering. This can be useful for orientation and relative camera location. It is also used for 2 sided textures using its Flip Normal attribute.



Camera Node This can be used to automatically calculate the near and far clipping planes to enclose all objects within the camera’s view, this helps by removing depth precision problems.



Plus Minus Average Node This Node does what the name suggests. It adds, subtracts or averages the two values connected to its inputs and returns an output of the two.



Multiply Divide Node This Node works on the same logic as the above Node. It allows the multiplication or division of two inputs and outputs the result.