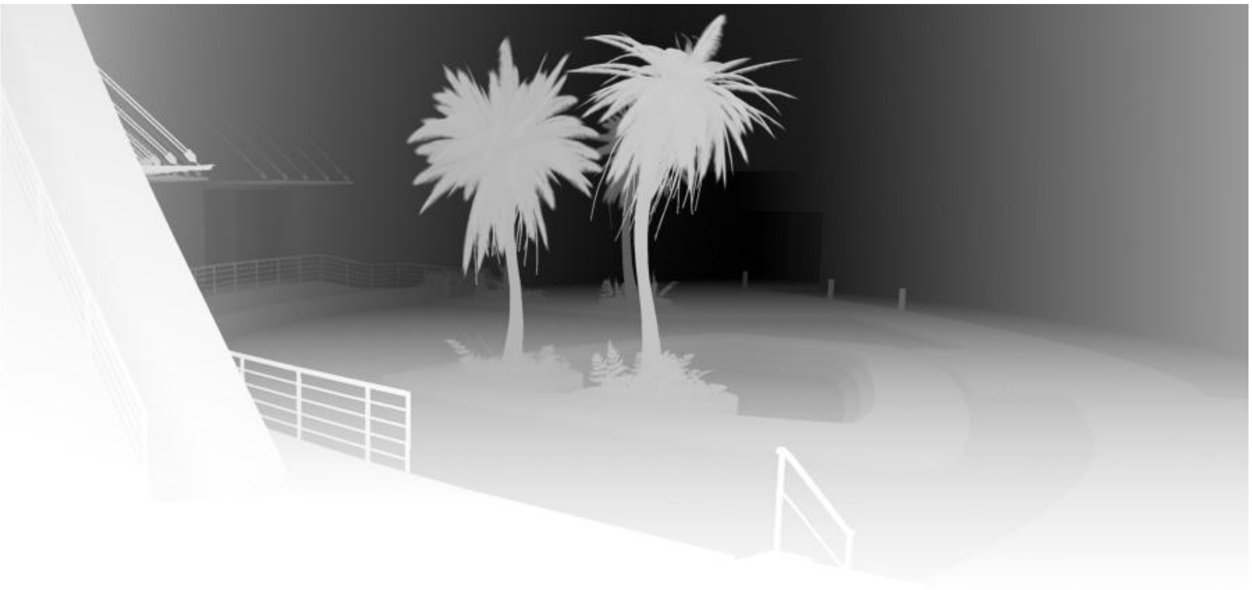
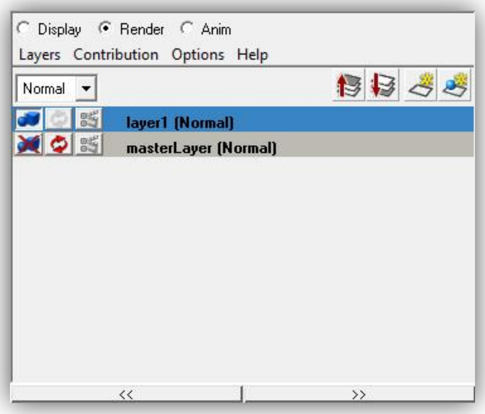
Luminance Depth DOF and Alpha



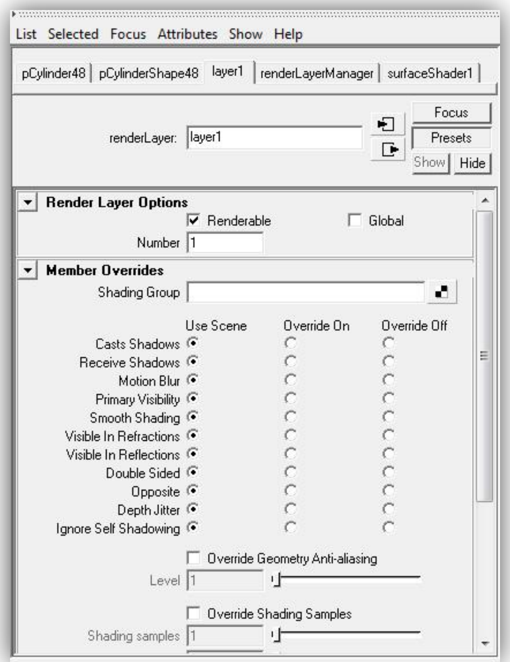
In this tutorial I will present a Depth of Field solution using Mayas luminance depth method. Also, I will present a solution to any surfaces that have an alpha and need to accurately render depth behind them.

First let's start by setting up our depth of field. If there are no transparent objects in your scene, it's as easy as adding all your objects to a new render layer (fig.1). Name the layer something appropriate, i.e. DOF_Luminance_Depth_Pass.
You do not need to add any lights to the layer as this render pass is generated by a surface shader.



layer1 (Normal)
masterLayer (Normal)

fig.1 Render Layer Options.


List Selected Focus Attributes Show Help
pCylinder48 | pCylinderShape48 | layer1 | renderLayerManager | surfaceShader1
Focus
Presets
Show
Hide

renderLayer: layer1

Render Layer Options

☑ Renderable ☐ Global

Number 1

Member Overrides

Shading Group

	Use Scene	Override On	Override Off
Casts Shadows	●	○	○
Receive Shadows	●	○	○
Motion Blur	●	○	○
Primary Visibility	●	○	○
Smooth Shading	●	○	○
Visible In Refractions	●	○	○
Visible In Reflections	●	○	○
Double Sided	●	○	○
Opposite	●	○	○
Depth Jitter	●	○	○
Ignore Self Shadowing	●	○	○

☐ Override Geometry Anti-aliasing

Level 1

☐ Override Shading Samples

Shading samples 1

fig.2 Attribute editor and Render Layer Override Options.

After you created the layer, go to your attribute editor (fig.2) and select the newly created layer and selecting Luminance Depth in the presets.
This will create a surface shader and assign it to your objects by overriding any incoming shader connections. Because this is an override, you cannot assign another shader to these objects on that layer.
Now you can render out a depth pass and use it in post to simulate depth of field.

In order to have more than one shader in your render layer, you will need to follow one of two paths. The first method is to create a luminance depth pass with the method described above, then take the same objects, assign them to a new render layer and manually assign the shading network from your hypershade. After that, you can delete the first layer.
The other way is to create the shader from scratch.
If you chose to go that route, you will need a samplerInfo node, a multiplyDivide node, a setRange node and a surface shader.

First, middle mouse drag and drop the samplerInfo onto the multiplyDivide. Chose other as the type of connection you wish to make. This will bring up your Connection Editor (fig.3).

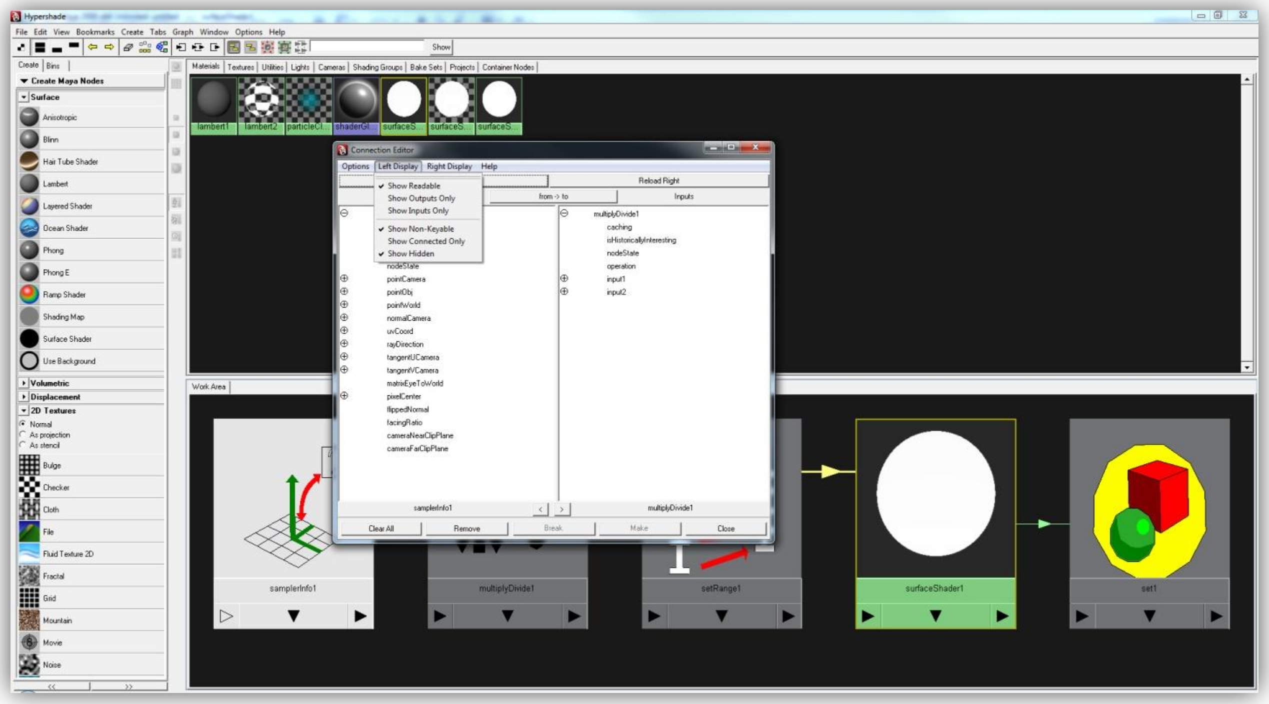


fig.3 Connection Editor

Now, follow the connections in (fig.4) and (table 1) to create the shader.

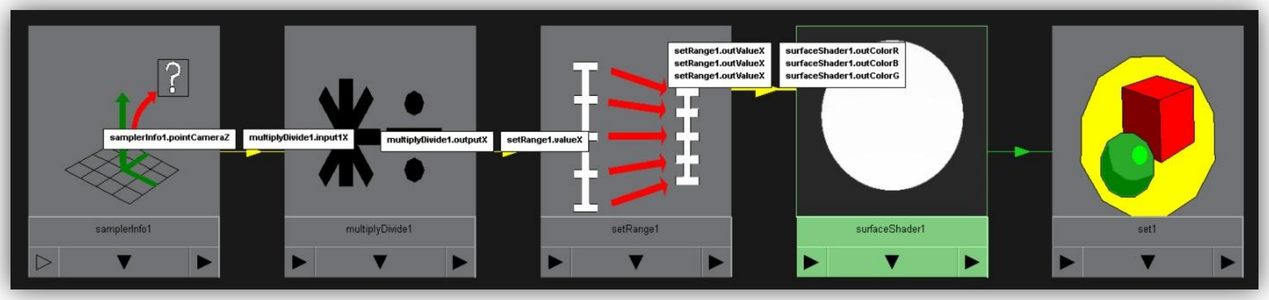


fig.4 Connections diagram

samplerInfo.pointCamera	multiplyDivide.input1X
multiplyDivide.outputX	setRange.valueX
samplerInfo.cameraNearClipPlane samplerInfo.cameraFarClipPlane	setRange.oldMinX setRange.oldMaxX
setRange.outValueX	surfaceShader.outColorR surfaceShader.outColorG surfaceShader.outColorB

table.1 Connections

You will notice that (table.1) has two more connections than the diagram. By default these connections are made so that your entire scene fits onto the DOF radius. Personally I prefer to have control over my DOF falloff and set these values manually.

Now, some problems arise when there are objects with alpha, such as foliage, hair, torn cloth etc.
If you use the luminance depth solution the way it was described above you will get something like in (fig.5).

The best way around this and the real purpose of this tutorial is to substitute the shader you want to have transparency with a Lambert. Create and assign the surface shader to everything that does not have an alpha, then create a Lambert shader and assign the setRange node outValueX to the color RGB channels.

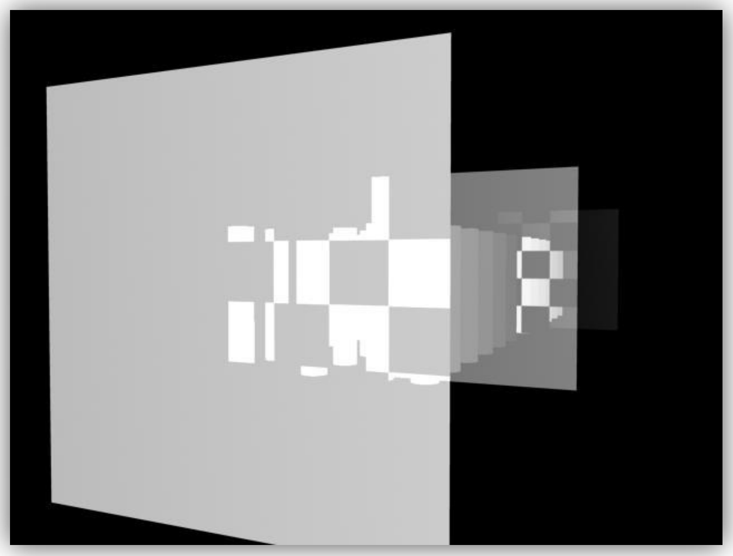


fig.5 Transparency issues with surface shaders and DOF.

As a last step you will need to set the ambient color value to white, effectively turning the Lambert to a surface shader (fig.6).

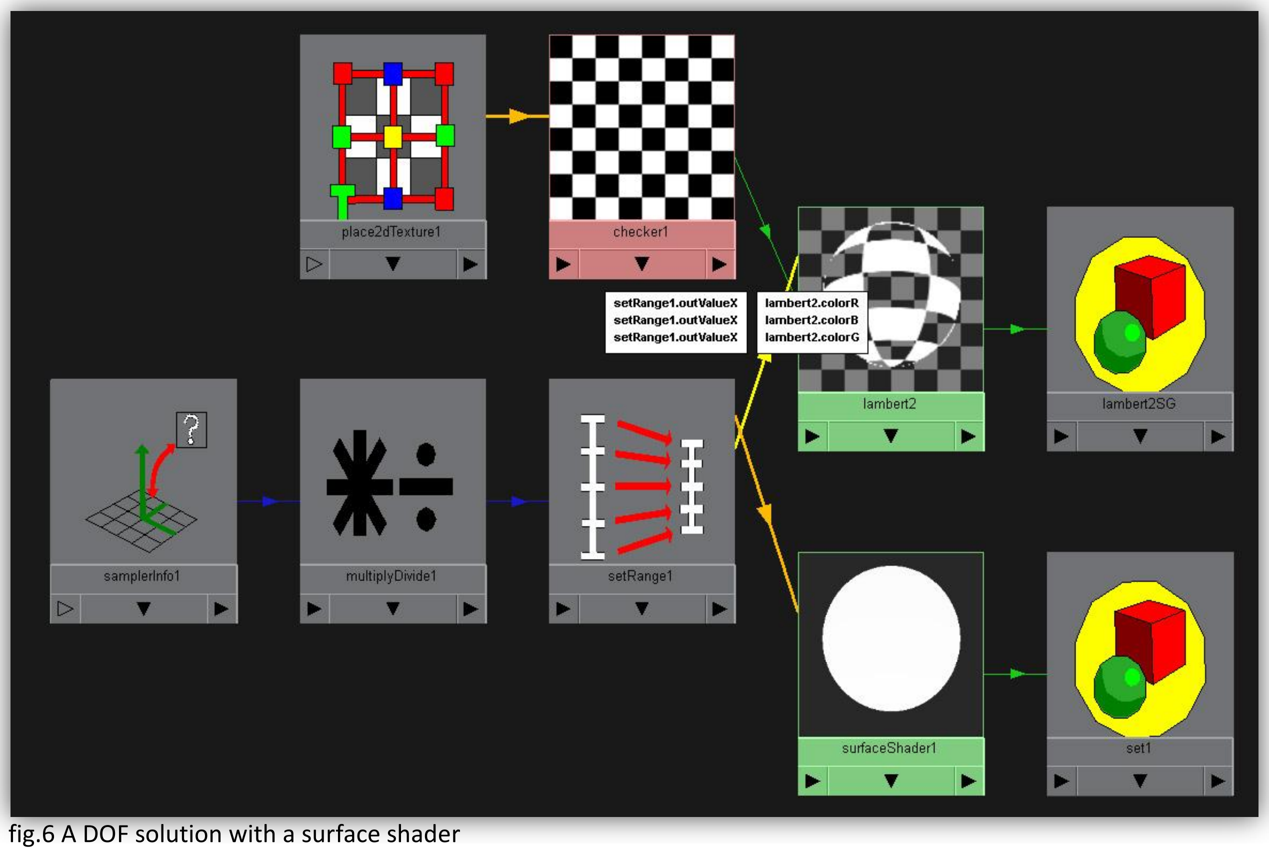


fig.6 A DOF solution with a surface shader and a Lambert to handle transparent objects.

If you apply the Lambert to the correct surfaces your render will now look like (fig.7).

As a side note, you will need to create as many Lambert shaders as you have different alpha maps.

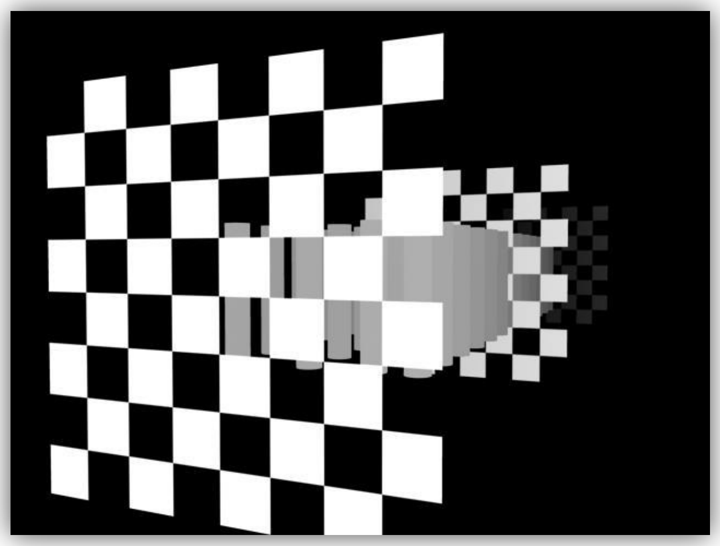


fig.7 Corrected DOF render.