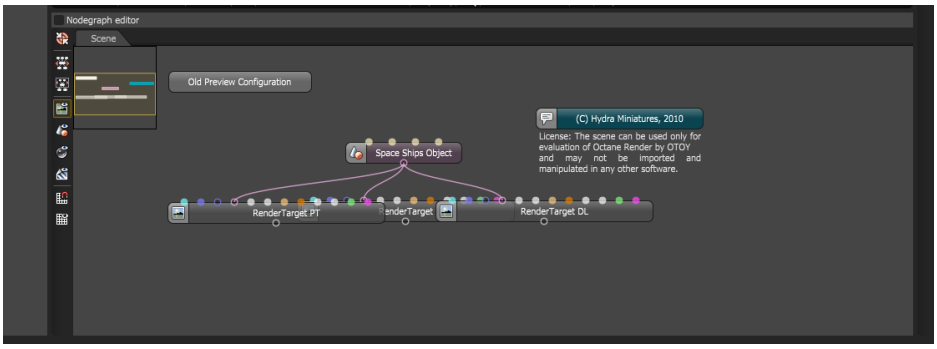
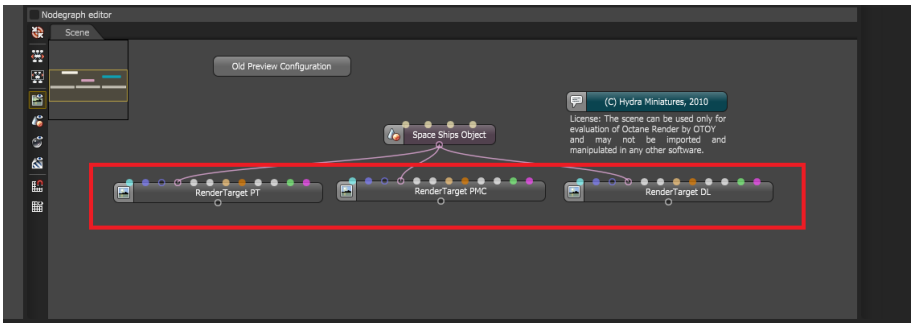
Octane- standalone – osl- camera

To begin, open the scene in which you intend to render.

Once the scene is open, we begin by looking into the Nodegraph editor.

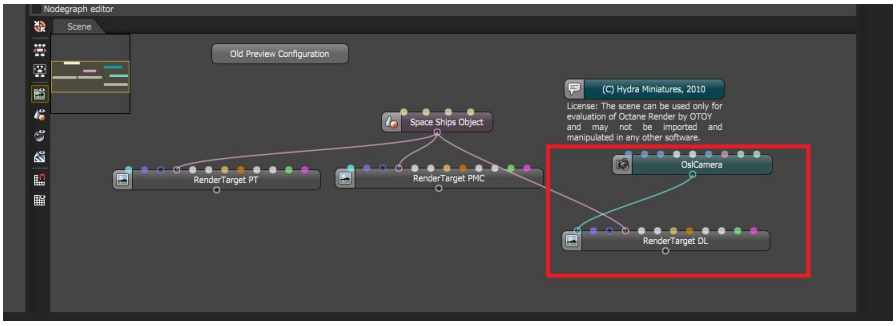


Depending on the scene this editor may be very complicated or simple, we are looking for a render target to accomplish the install of the light field camera.



In this example there are 3 Render targets it does not matter which render target we select

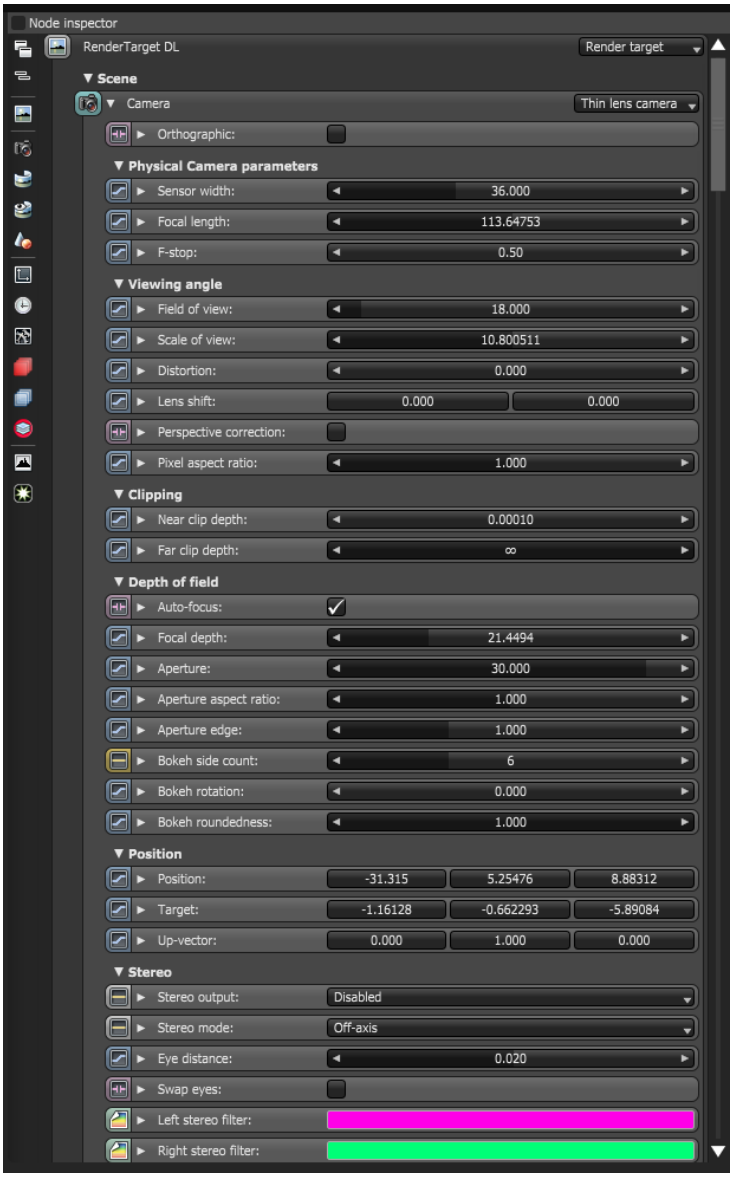
If the Render Target Has a camera node as an input, we still follow the same instructions



In the example above there is an oslCamera node, this can be any type of camera and will not effect the process.

Now that we have selected our render target the Render viewport should contain the view of the camera. Within the example scene this by default is a thin lens camera view

The node inspector (found on write side of screen by default) should now also contain many attributes to adjust the rendering Target

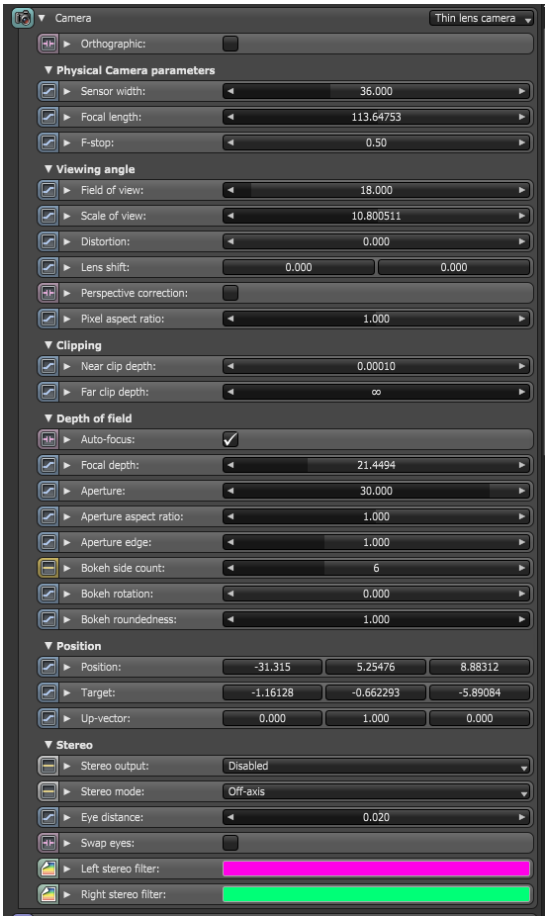


Within this list we are concerned with 2 sections.

Film settings:

Which controls the outputted Light Fields entire Resolution

The Camera:



This section will describe the actual construction of our light field.

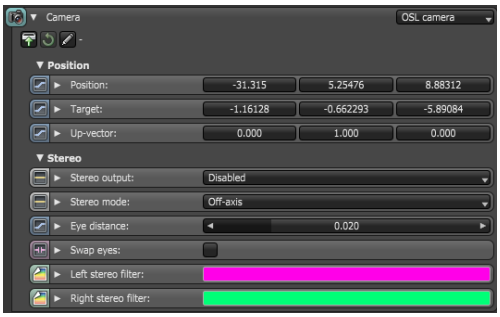
We now look at the camera section to create our light field camera:

In the top-left of the section there exists a drop-down menu with the type of camera currently in use being shown.

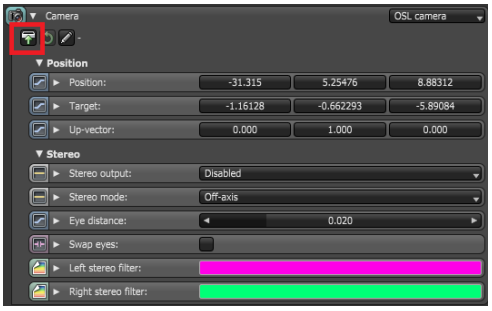
We select this box and, in the drop-down menu select osl camera.

Note: When we select the OSL camera the render viewport may change to represent the new camera settings. This is not a problem.

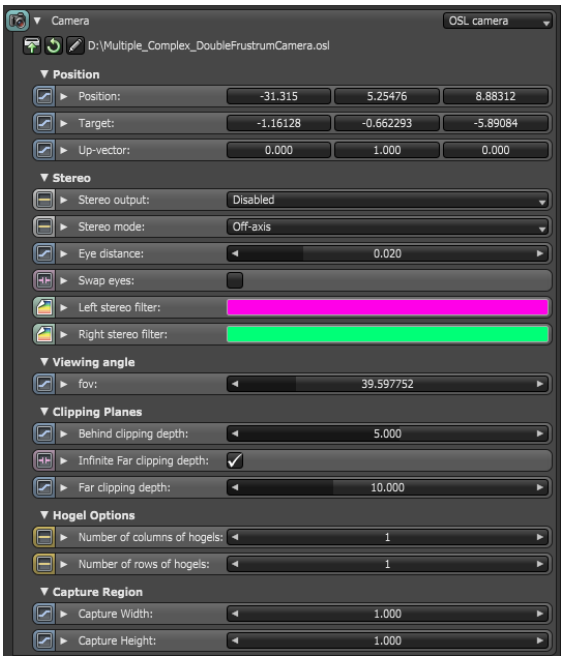
If the above has been correctly done our camera section should now contain the below options



We now click the load osl code button found on the top left of the osl camera settings.

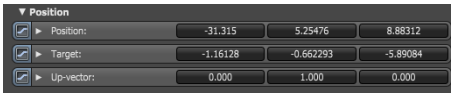


We now select the light field camera OSL file. When loaded the viewport may again adjust to the new camera and the camera section should now contain the light field specific settings



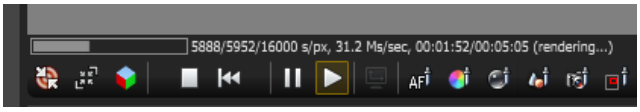
The light field camera has now successfully been installed and can be used by adjusting the settings to return an appropriate light field by specification.

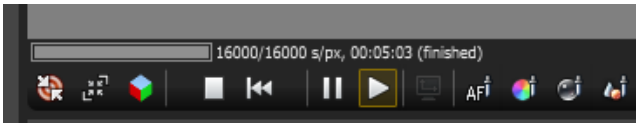
We can move the camera by moving it within the viewport or adjusting the parameters within the position tab of the camera section within the node editor.



To save the light field we simply wait until the render has completed

The time required can be found in the bottom right of the viewport





We then select from the option in the viewport to save the rendered image

Light Field Parameters

The Following parameters directly adjust the generated light fields results

Camera -> Viewing Angle->**fov**:

sets the field of view of each individual Hogel

-Can be any positive float that is less then 180

Camera -> Clipping Planes -> **Behind Clipping depth**:

sets how far behind the camera the rays from the camera will travel from

-Can be any positive float

Camera -> Clipping Planes -> **Infinite Clipping depth**:

Check box to allow an infinite clipping depth

-Can be any positive float

Camera -> Clipping Planes -> **Far clipping depth**:

If there is not an infinite clipping plane this value is used to describe how far in front of the camera Each ray from the camera should travel

-Can be any positive float

Note: within the code far clipping is added to near clipping for, so user only needs to worry about far clipping distance from the position of the camera.

Camera -> Hogel Options -> **Number of columns of hogels**:

Sets the number of hogels in the x direction to render

-Can be any positive integer

Camera -> Hogel Options -> **Number of rows of hogels**:

Sets the number of hogels in the y direction to render

-Can be any positive integer

Camera -> Capture Region -> **Capture width**:

Sets how far the 2 hogels on the opposite side of the width (left and right) should be from each other within the renderer's units.

-Can be any positive float

Camera -> Capture Region -> **Capture height**:

Sets how far the 2 hogels on the opposite side of the height (top and bottom) should be from each other within the renderer's units.

-Can be any positive float

Render settings -> Film settings -> **Resolution**:

This sets the light fields overall resolution

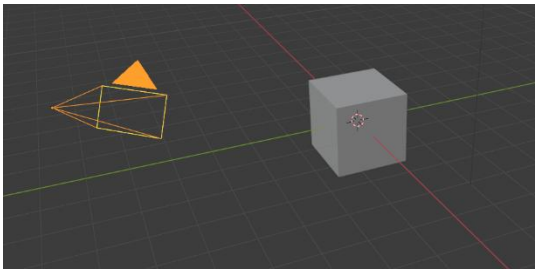
-Can be any combination of 2 positive integers

Blender – osl- camera

To begin, open the scene in which you intend to render.

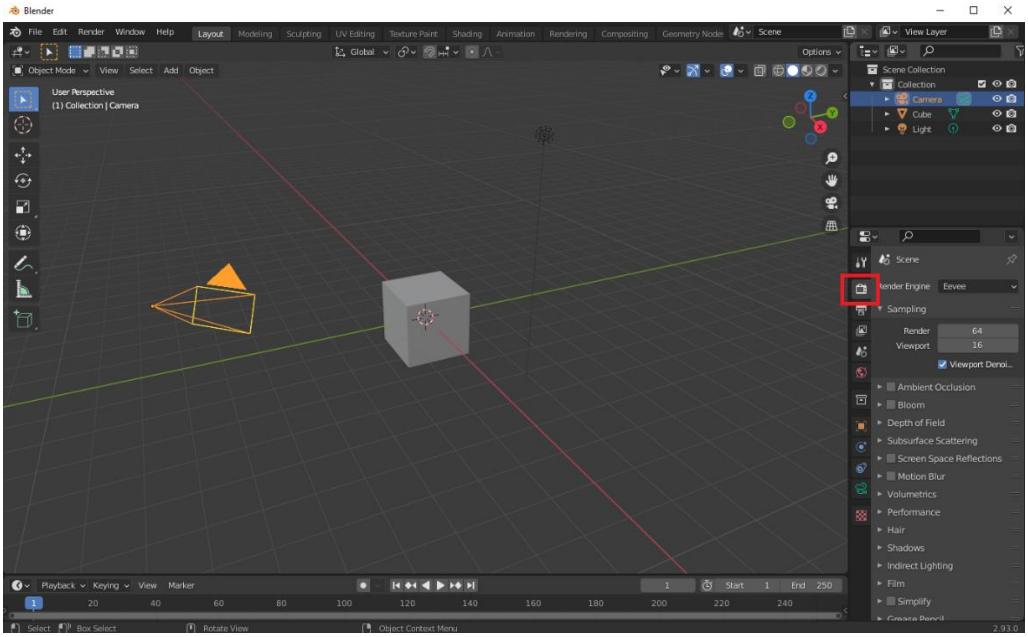
Within blender we have 2 options for rendering a light field with octane, we can use the blender render options or we can use octane's external renderer. We will begin with using the external renderer. Although selecting the render engine property is the same step for both

We now position the camera within the scene to the location we intend to render.

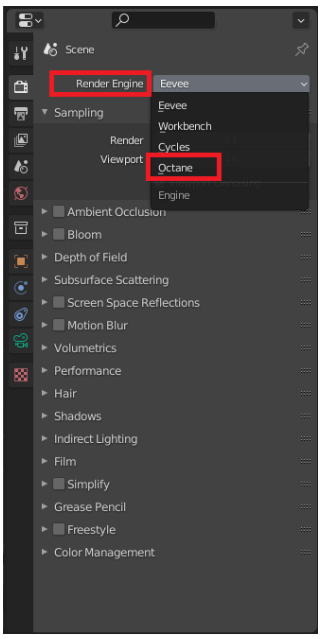


Note: the scene used is blenders default starting scene

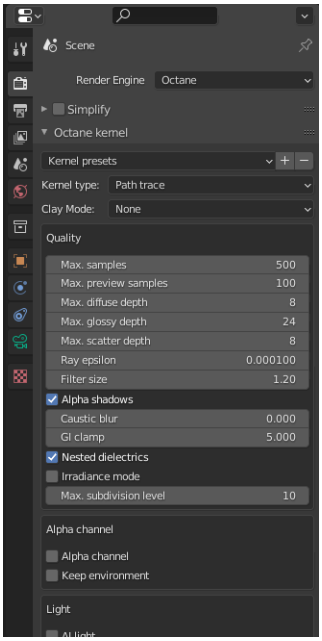
Next, we go to the Render properties tab found on the right of the screen



Within this tab Find the Render Engine property and change its value to Octane



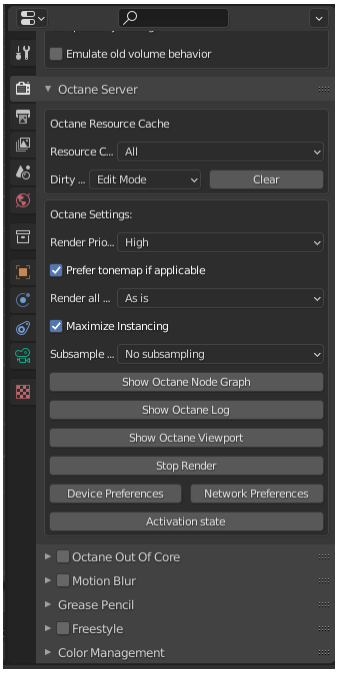
Once you select this option the Render Properties tab should change



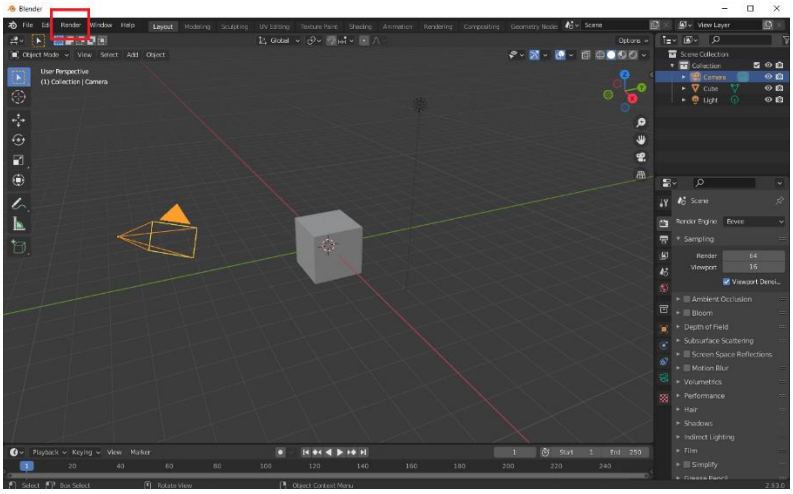
The above steps Are the same for both blender’s rendering ways

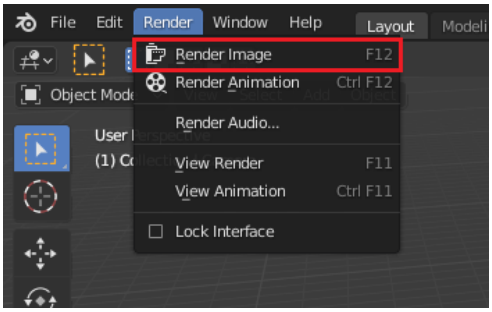
Octane's External Renderer Method:

Within this new set of properties find the Octane server sub tab

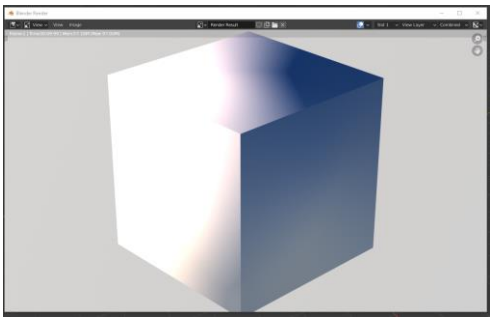


Now back to the entire blender scene we select the render option from the top left and select render Image F12 is also a default shortcut to perform this operation.



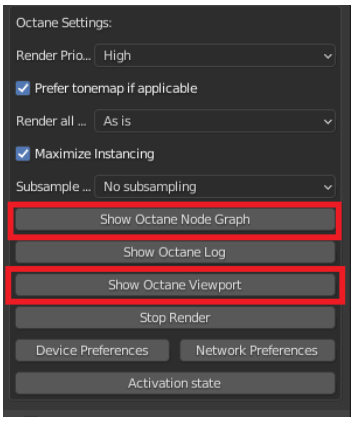


Once clicked a new window should open labeled Blender Render, we are not concerned with this window, and it can be ignored but cannot be closed.



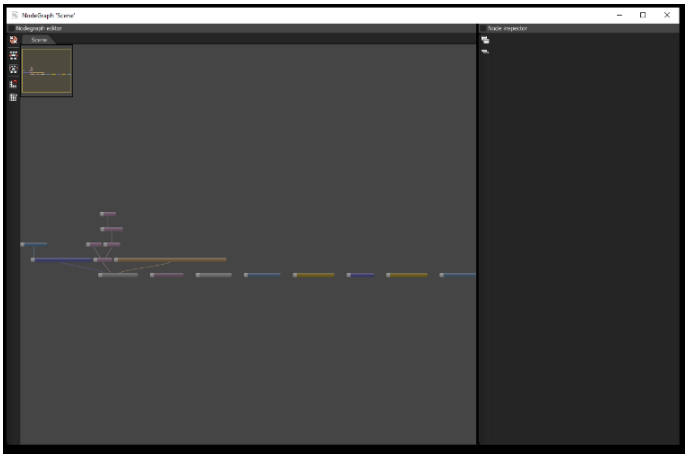
We return our attention to the Octane server tab and click both Show Octane Node Graph

And Show Octane Viewport options

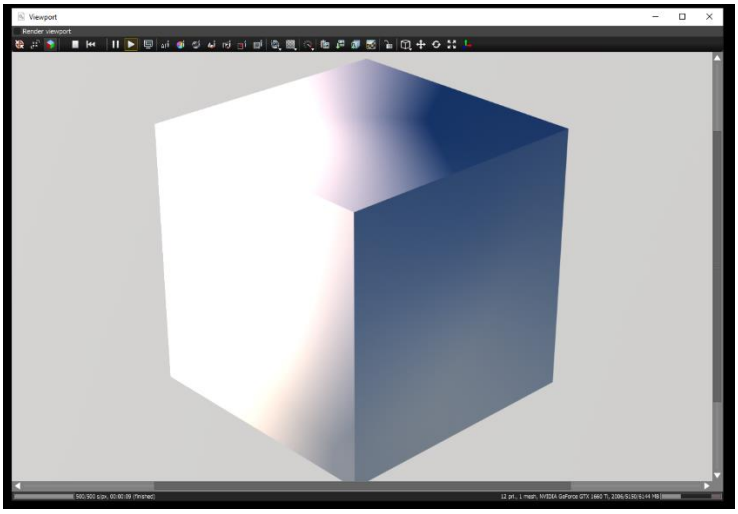


Each button should open a new window,

The show octane Node Graph should open a window named NodeGraph ‘scene_name_Here’ which then behaves in the same way as the nodeGraph editor found in the standalone renderer so the instructions above can be used to load and run the osl camera.



The Show Octane viewport opens a window labeled Viewport and from this window the scene can be viewed with the node Graphs parameters being applied.



Note: With this method you can only view the lightfield cameras - output on the Octane viewport, the blender render does not respond to changes in this method.

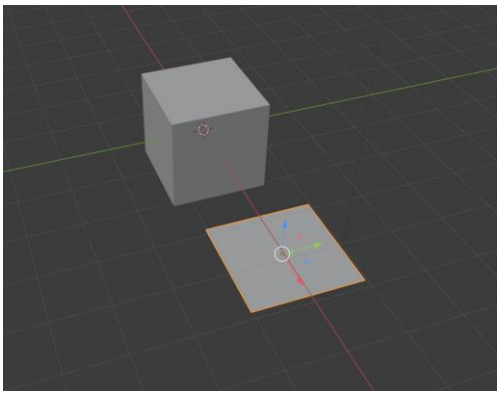
Note: You may have to wait until the viewport completes its first render before being able to change resolution of output image

Note: Moving the camera is also more complicated in this method as to move it within blender means you need to click the render image button again (or use f12 as its shortcut)

Note: The node editor changes take priority within the render so changing settings here will override blender settings in output but not change the values blender has provided

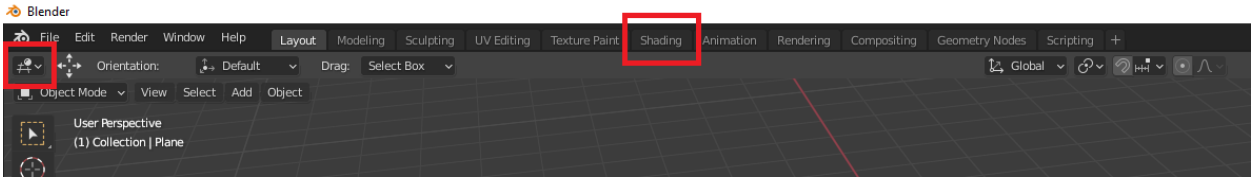
Blender Renderer method:

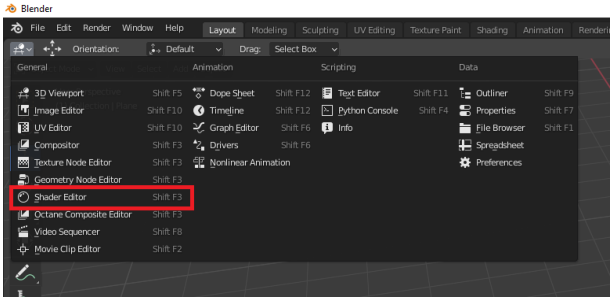
Once we have changed to the Octan renderer, we begin by adding a plane to our scene



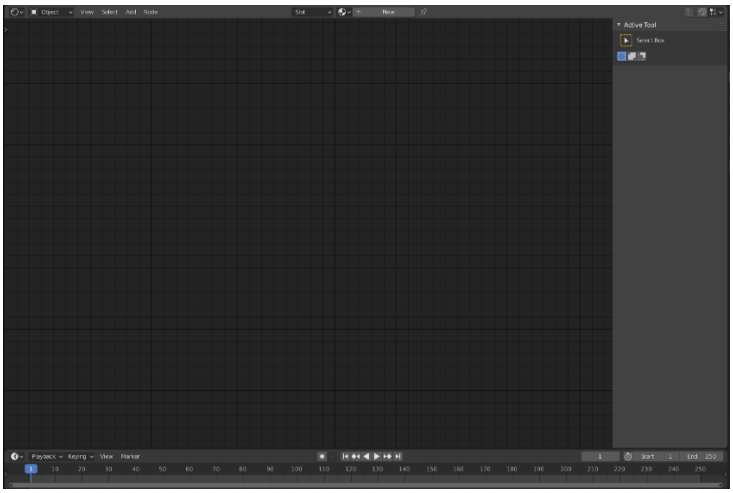
You now select that plane and move to the shader editor.

(You can either select the shader editor from the viewport or switch to the shading tab in blender or using the shortcut shift F3)



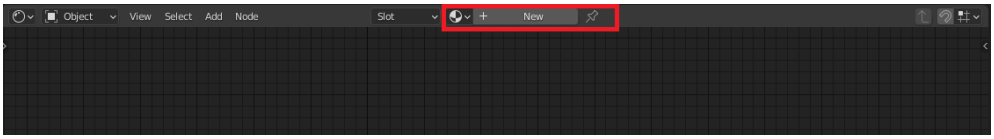


Now a view of an empty grid should appear, this is the shader editor

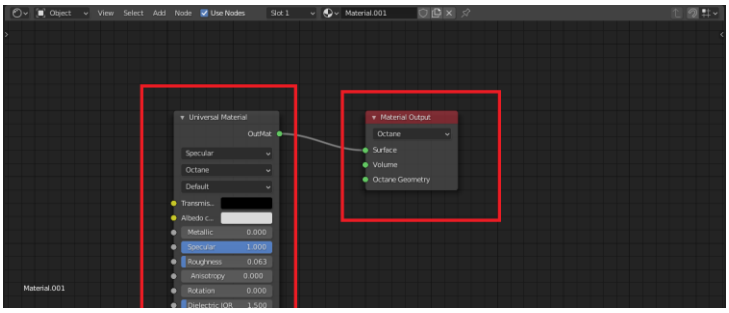


If the plane has been selected this grid should be empty.

Next, we create a new material

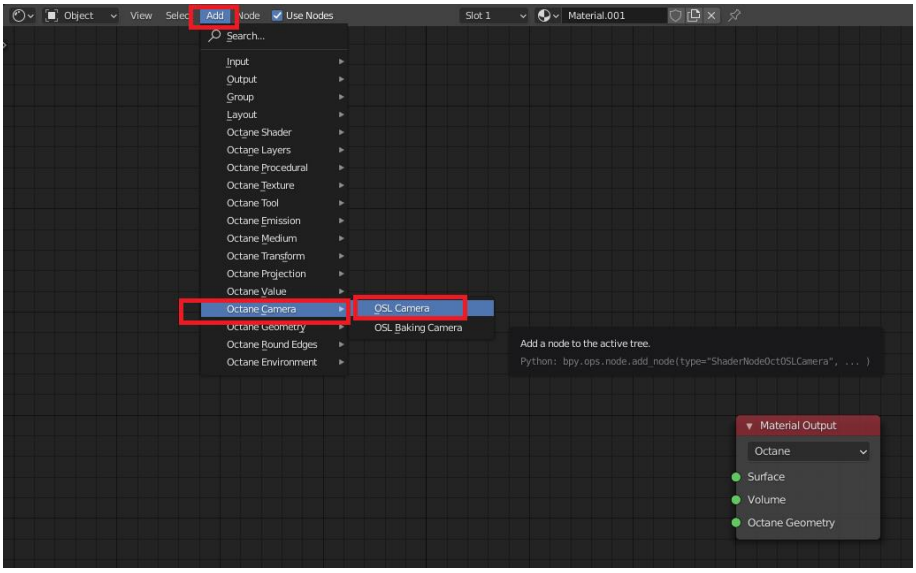


Once created the grid should contain two nodes, a Universal Material Node and a Material Output Node.

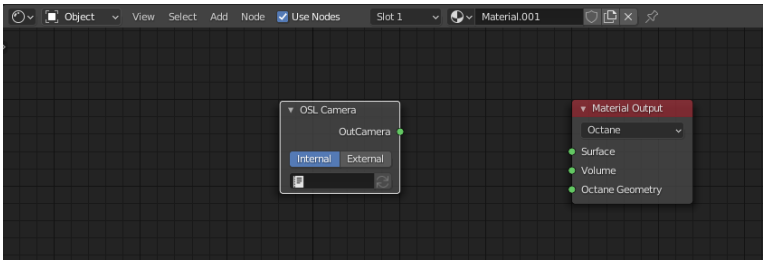


The universal node is not needed and can be deleted, leaving just the Material output Node

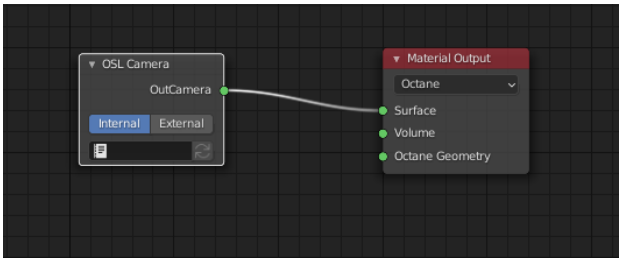
Next add the osl-Camera shader node. To do this click the add option in the list and go to the octane camera section and select osl-Camera



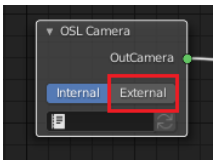
The graph should now contain the Osl- Camera Node and the Material output Node

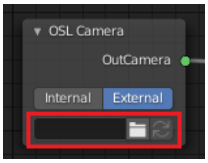


Now connect the OutCamera output on the osl camera to the Material Output's Surface input

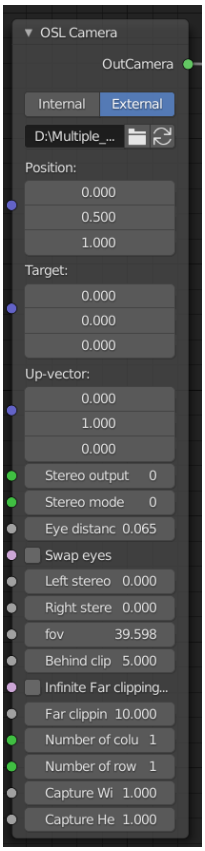


Next within the OSL Camera node click the external script option and then within the text box select the osl script to be used





Once you select the Camera osl file the Osl Camera Node should update



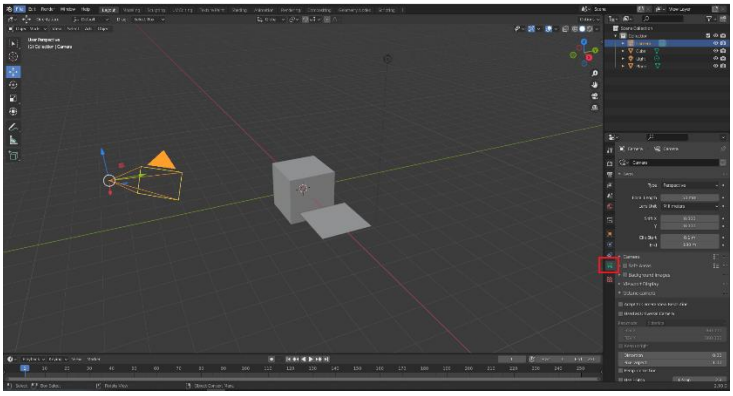
The parameters found here can be used to modify the light field as stated in the light field parameters section

Note: These settings can also be modified within the material tab of the plane within blenders GUI.

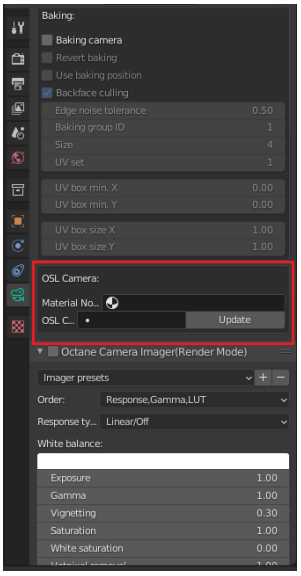
Now that the light field shader has been set up, to see the results a camera will need to be connected to it.

First select the camera you would like to use the shader on.

Once the camera is selected go to the Camera properties tab found on the right of the scene

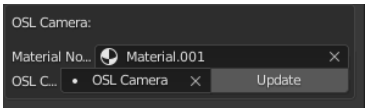


Within the Camera Properties tab find the OSL Camera section



Now Set the Material Node graph to the Material which we created above. Then set the OSL Camera node to the OSL Camera in the drop-down menu then click update

Once completed the OSL Camera section should look like this



Blender will now operate under the osl camera perspective and create the osl Camera's effect when rendering within blender.

Note: The use of the Plane is the simplest solution to allow the camera to render, this can be a problem as the plane is not part of the scene. The simplest solution to this problem is to shrink the plane to be very tiny and hide it in the scene in a way that the camera does not see it.

Note: We simply need the osl Camera node to be within the scene and rendered, As such more complex solutions to use the osl-camera exist without adding objects to the scene the method shown above is simply the simplest.