Fulldome Tutorial

Austin Durose

22/06/16

Contents

Fulldome 2
- Overview 2
- Five Camera Rig 2
- Technical Jargon 5
- Fisheye Render 6
- Fitting to a Dome 12
- VR purposes 13
- Unconventional Dome Shapes 14

Fulldome

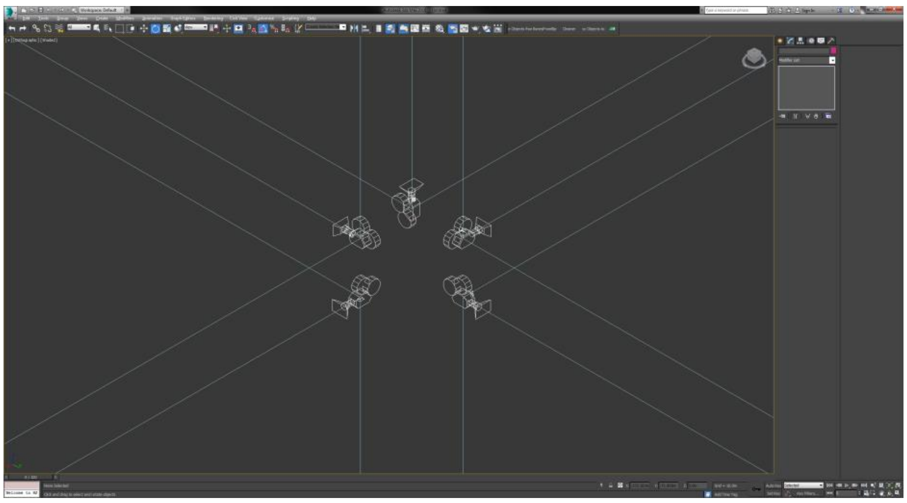
Overview

In this section I aim to explain as much as I can about fulldome. I am not a master in this field, and suggest that if you need more help on this subject, you should contact NSC Creative who have around 15 years of experience creating high detailed fulldome shows. For now, I'm going to give all of the information I have.

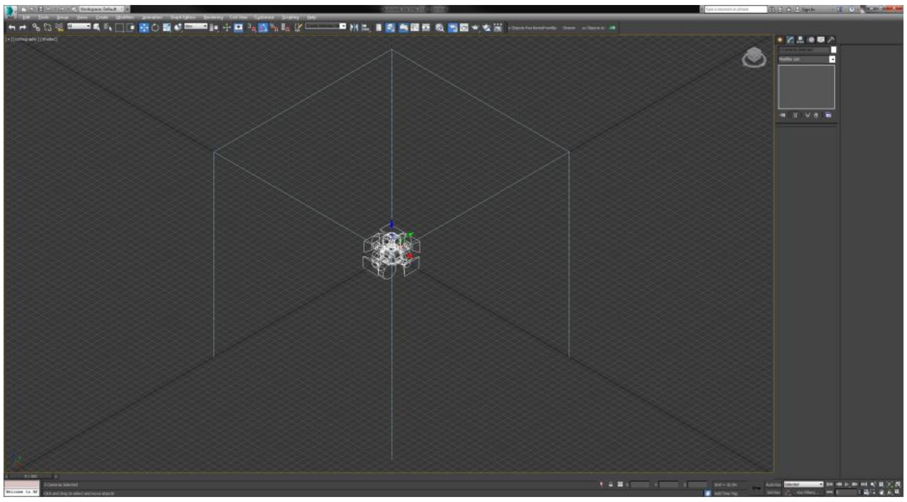
Five Camera Rig

Ok… let's have a look at this. I'm going to talk through what to do in 3Ds Max and then I can talk through some more complex stuff.

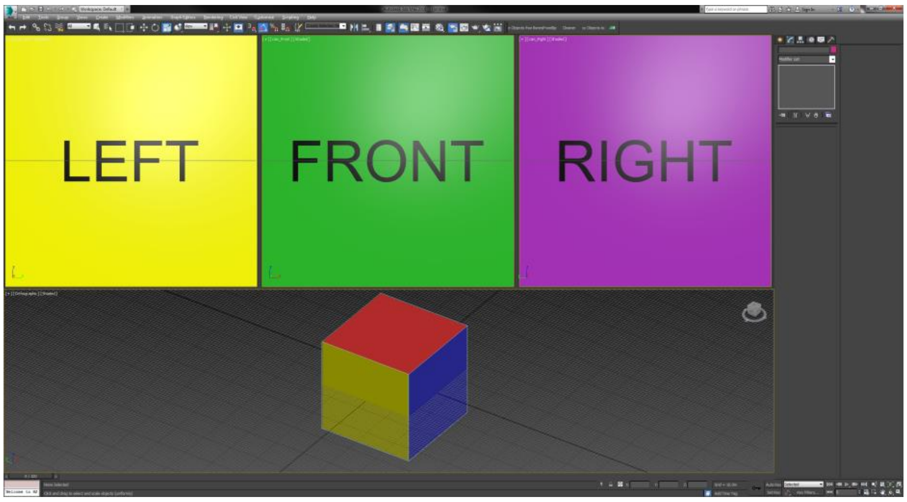
I will create five cameras and make sure that the frame size is square, I use 1024x1024 pixels. I also need each camera to have a FOV of 90 degrees, then I will rename these Up, Front, Right, Left and Back with the prefix 'cam_', then I will point them in their respective directions.



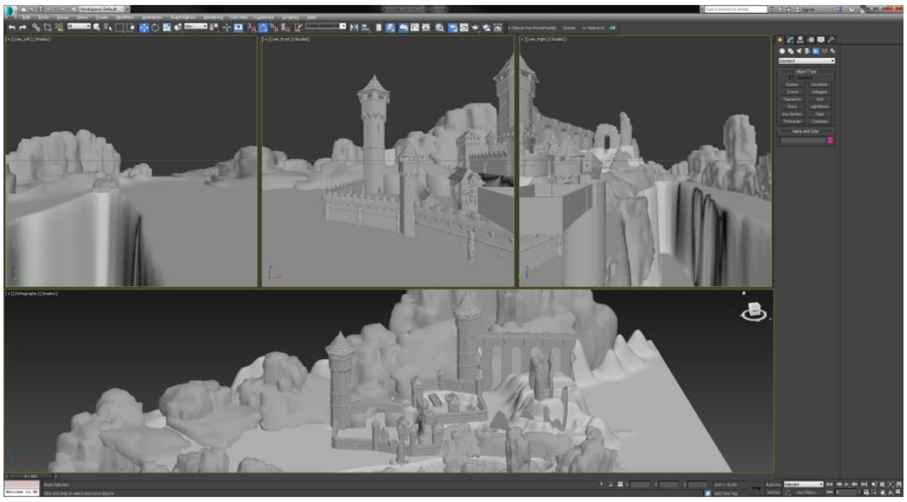
I will place these all at the origin (position 0, 0, 0), and with all of the 90 degree field of view cones, we see we have made a box sort of thing.



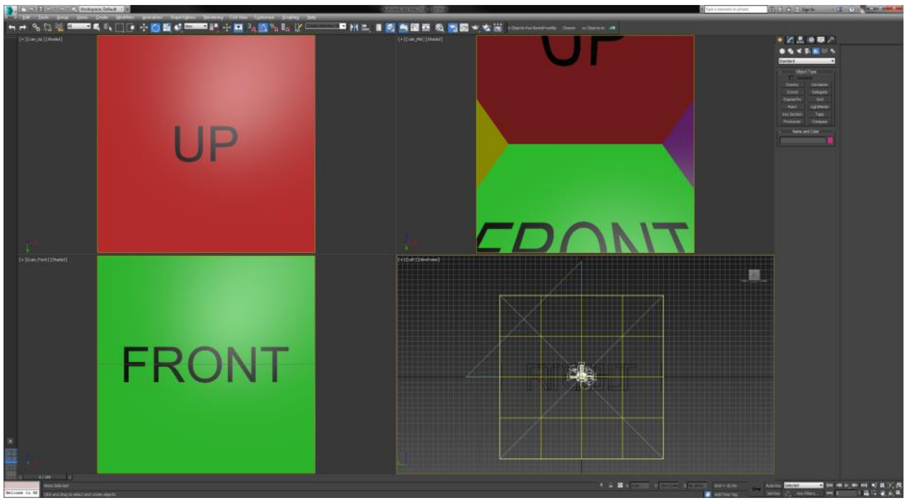
I have created a cube and coloured each side a different colour to help us understand what is going on. Here we see four view ports, the top left is from the left camera, the top middle is the front camera, the top right is the right camera.



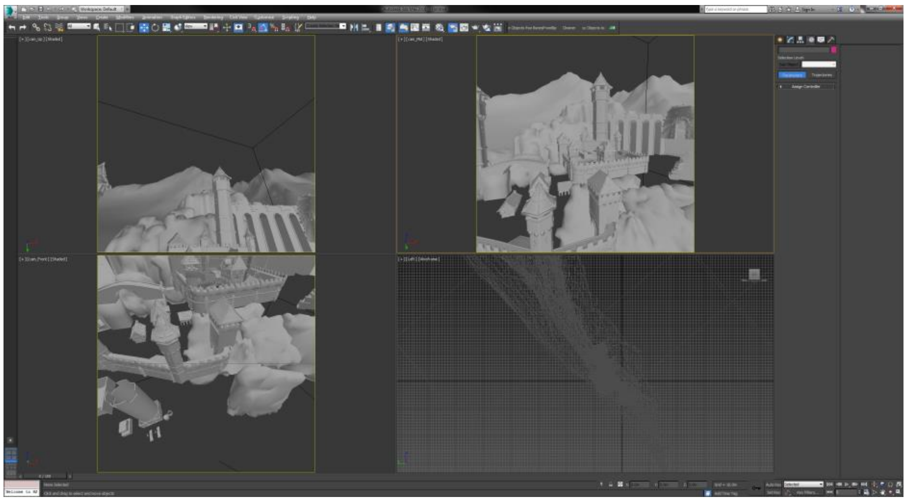
If I was to xRef an environment into the scene, we can see how these cameras will give us a panoramic view.



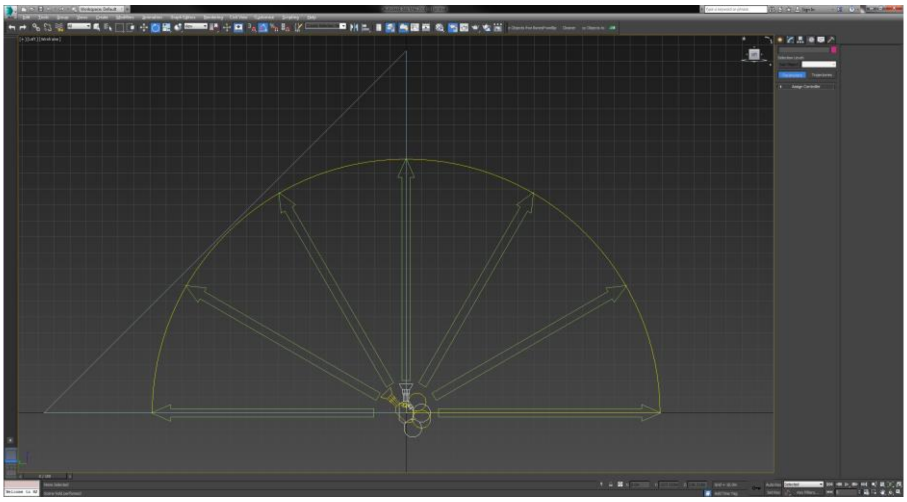
I have duplicated the cam_Front and rotated it upward on the X axis 45 degrees. I will rename this cam_mid.



In this image, the left two viewports show us views of the Up and Front camera, the upper right viewport shows us the view from the Mid camera. As you can see, the mid view cuts off half of the front and half of the up planes. We can also see from the orthographic view that the bottom of the mid camera frame is plush with the XY gridline.



Here I have added the scene to show how the cam_Mid helps us frame the scene we would want the audience to see.

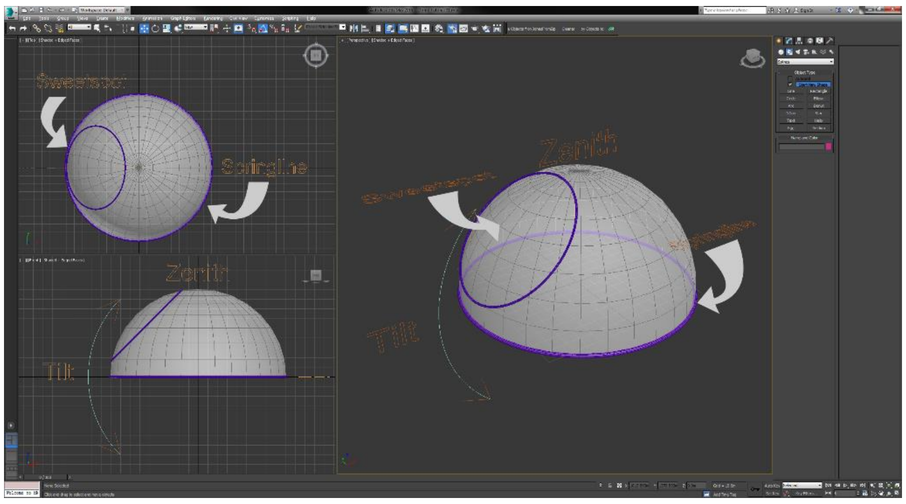


We will be rendering a 180 degree fisheye from the cam_Up. When this is the case, the extent of the frames will be plush with the XY gridline, therefore we can use the cam_Mid to help us frame what the audience will be predominantly viewing in a uni-directional (everyone sat in seats facing forward) dome.

The camera sends out rays to capture everything it can see, therefore we can think of the camera as being the dead centre at the bottom of the dome.

Technical Jargon

The four main definitions I want to give are ‘Springline’, ‘Zenith’, and ‘Sweetspot’ with some talk about the degrees of a dome and omni/uni-directional domes and some other bits..



Springline: the Springline is the very edge of the dome, where the screen ends. This is typically the circumference of the dome, though some domes may be shallower, and tends to be a complete circle.

Zenith: the Zenith of the dome is the apex of the dome, the very top at a perpendicular angle to the springline. When rendering in fisheye, the Zenith would be dead centre of the frame – the cam_Up will be showing what we see here.

Tilt: most domes have a tilt angle. The dome in the image has a tilt of 0 degrees, though most will have a tilt forward of around 15 degrees.

Sweetspot: depending on the type of the dome and tilt of the dome, the sweetspot will be where the audience will face. In a uni-directional dome, the sweetspot will be at the front of the dome. This is typically 45 degrees from the springline, the easiest place for the audience to view, and therefore where we want to put most of our action.

Circumference: this is the number of degrees relating to the springline, the degrees from start to end of the screen. In the example above, it is 360 degrees – a full circle – though some domes cut off the back of the screen, therefore giving a lesser number (half a fulldome would be 180 degrees).

Field of View: this is the angle of screen from springline to springline via the zenith, and is set from the centre if the dome was a complete sphere. Above us, the Field of View is 180 degrees, but if it was 90 degrees, the bottom half of the dome would be cut off. If it was 270 degrees, another quarter of the sphere, extended beyond the current springline, would be visible.

Uni-Directional Dome: this is an auditorium style dome, where all of the seats point towards one part of the screen (sweetspot).

Omni-Directional Dome: in this dome, there is no one direction. People are free to stand and move around the dome, or sit in seats that are inward or outward facing. There is no sweetspot as the audience will be looking all around the dome.

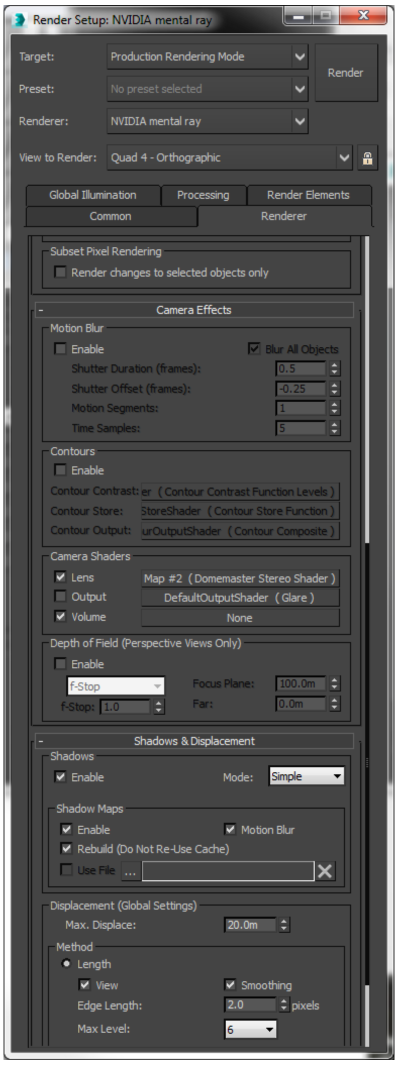
Bounce light: Because the image is being projected all over the dome, the light shone on the back (if a uni-directional dome) will bleach out the image projected on the front of the dome. It is usually best to use a vignette at the top of the dome plate to avoid this.

Fisheye Render

To render a fulldome plate, we would want to render a fisheye image. I could render from my five views and then sew these images together using a programme like Glom, or we could download the Domemaster Stereo Shader shown here:

http://www.andrewhazelden.com/blog/2012/04/domemaster3d-stereoscopic-shader-for-autodesk-maya/

This shader needs to be inserted in the 'Lens' slot in the Render Setup>Renderer panel. You will need to change the Renderer to Mental Ray. We can instance this lens material into the Material Editor by dragging it into an empty slot.



Render Setup: NVIDIA mental ray

Target: Production Rendering Mode
Preset: No preset selected
Renderer: NVIDIA mental ray
View to Render: Quad 4 - Orthographic

Subset Pixel Rendering
☐ Render changes to selected objects only

Camera Effects

Motion Blur
☐ Enable ☑ Blur All Objects
Shutter Duration (frames): 0.5
Shutter Offset (frames): -0.25
Motion Segments: 1
Time Samples: 5

Contours
☐ Enable
Contour Contrast: er (Contour Contrast Function Levels)
Contour Store: StoreShader (Contour Store Function)
Contour Output: urOutputShader (Contour Composite)

Camera Shaders
☑ Lens Map #2 (Domemaster Stereo Shader)
☐ Output DefaultOutputShader (Glare)
☑ Volume None

Depth of Field (Perspective Views Only)
☐ Enable
f-Stop Focus Plane: 100.0m
f-Stop: 1.0 Far: 0.0m

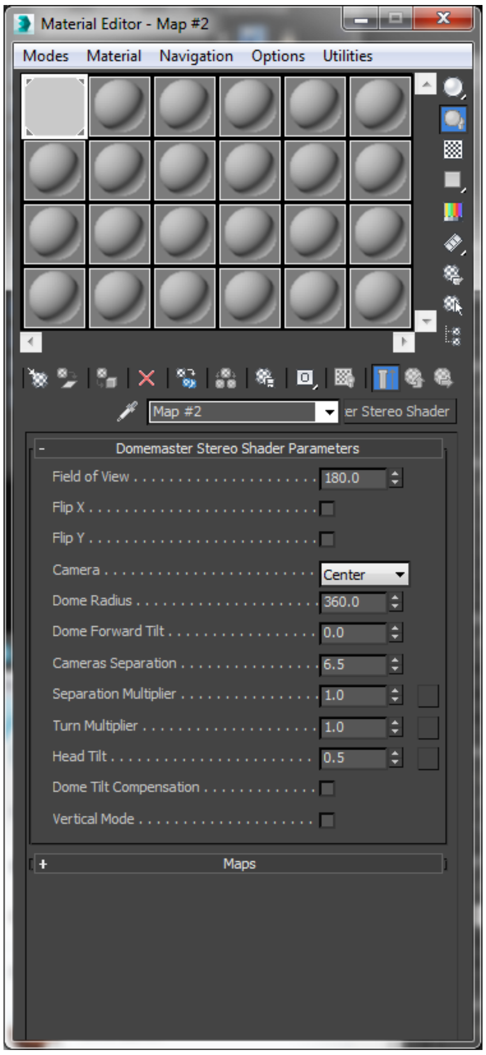
Shadows & Displacement

Shadows
☑ Enable Mode: Simple

Shadow Maps
☑ Enable ☑ Motion Blur
☑ Rebuild (Do Not Re-Use Cache)
☐ Use File ...

Displacement (Global Settings)
Max. Displace: 20.0m

Method
● Length
☑ View ☑ Smoothing
Edge Length: 2.0 pixels
Max Level: 6



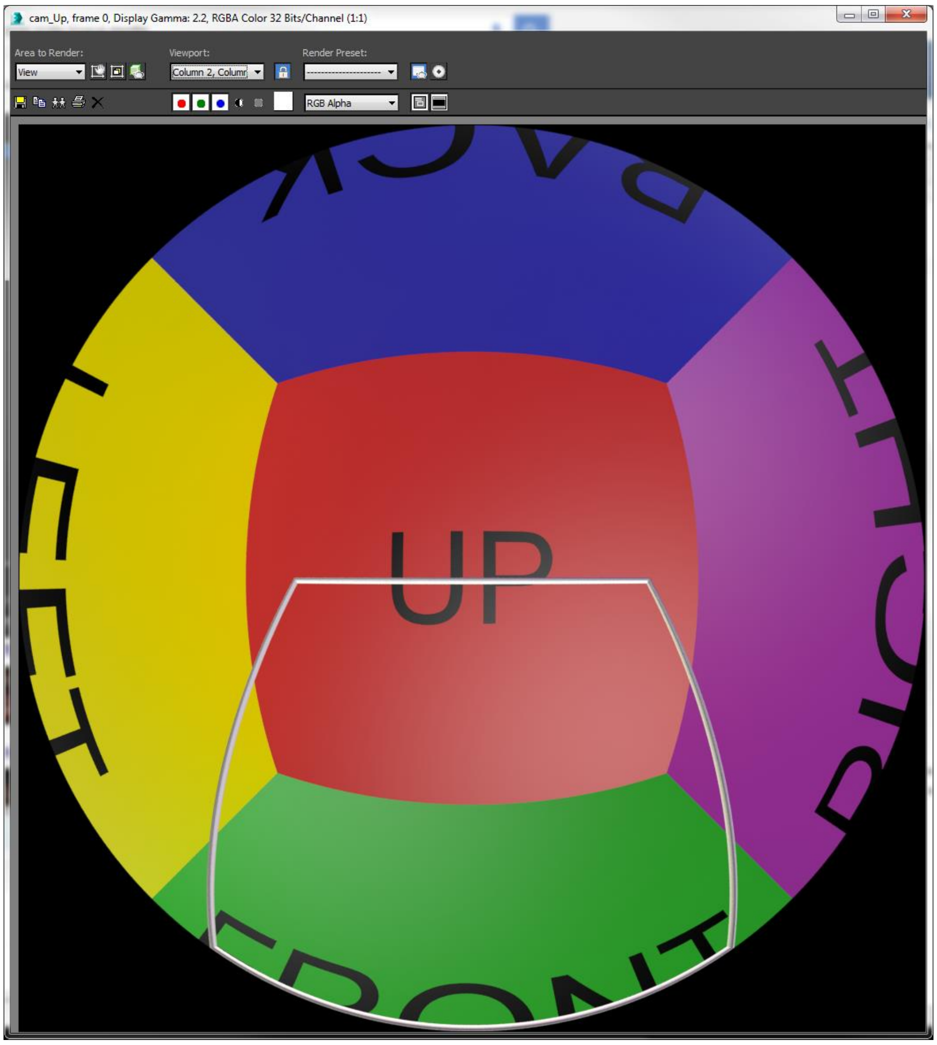
Material Editor - Map #2

Map #2 er Stereo Shader

Domemaster Stereo Shader Parameters

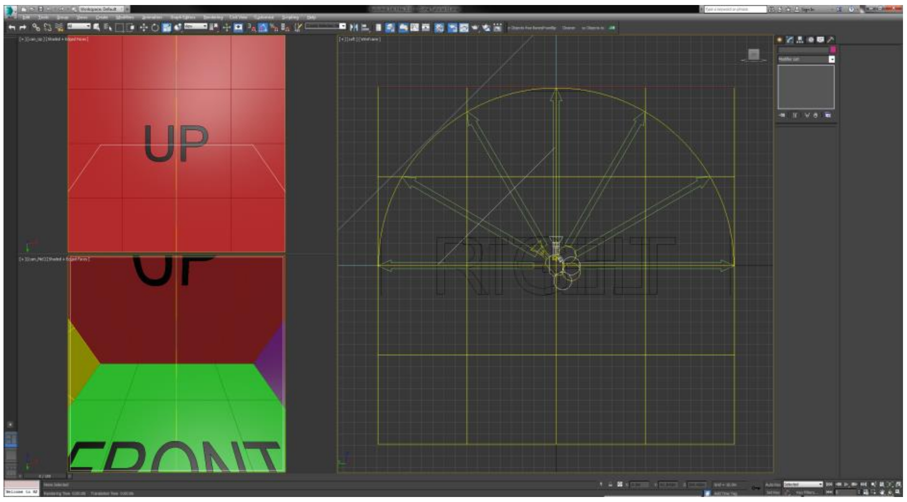
Field of View 180.0
Flip X ☐
Flip Y ☐
Camera Center
Dome Radius 360.0
Dome Forward Tilt 0.0
Cameras Separation 6.5
Separation Multiplier 1.0
Turn Multiplier 1.0
Head Tilt 0.5
Dome Tilt Compensation ☐
Vertical Mode ☐

In the Lens Shader, we can see that we can amend some of the attributes mentioned above. Keeping them as they are, cam_Up would render a fisheye place that would fit on a standard dome.

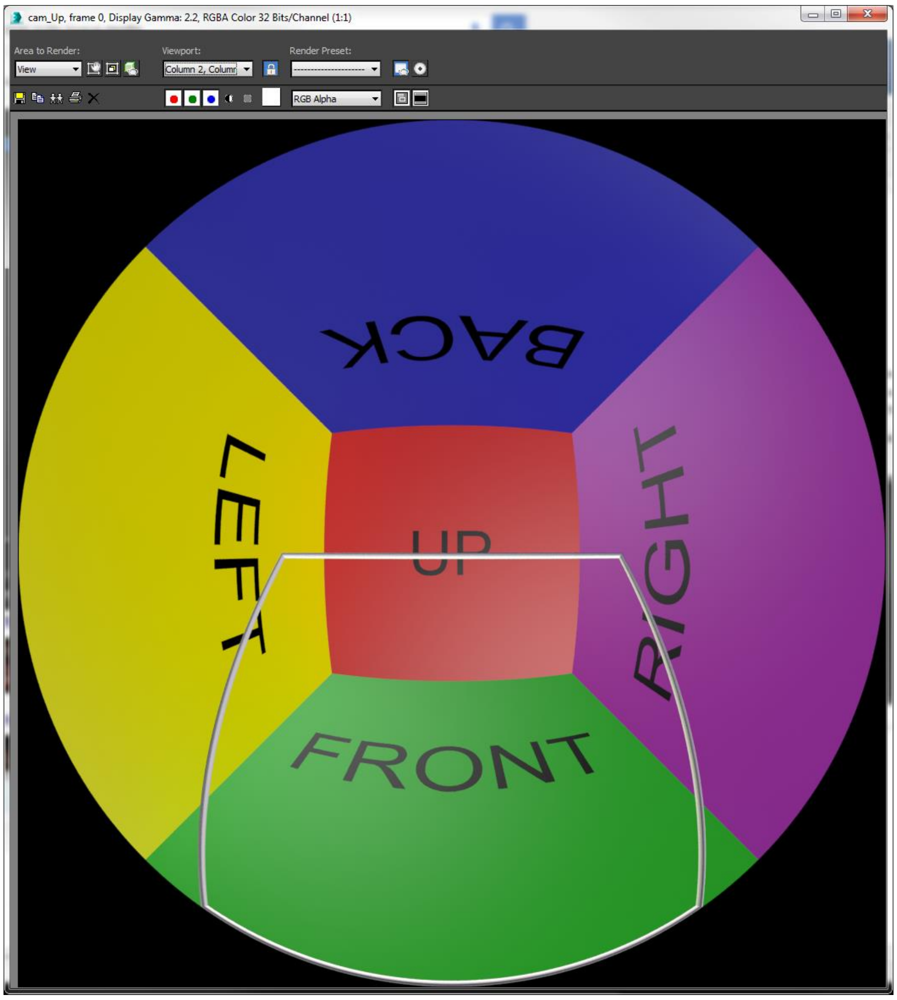

cam_Up, frame 0, Display Gamma: 2.2, RGBA Color 32 Bits/Channel (1:1)

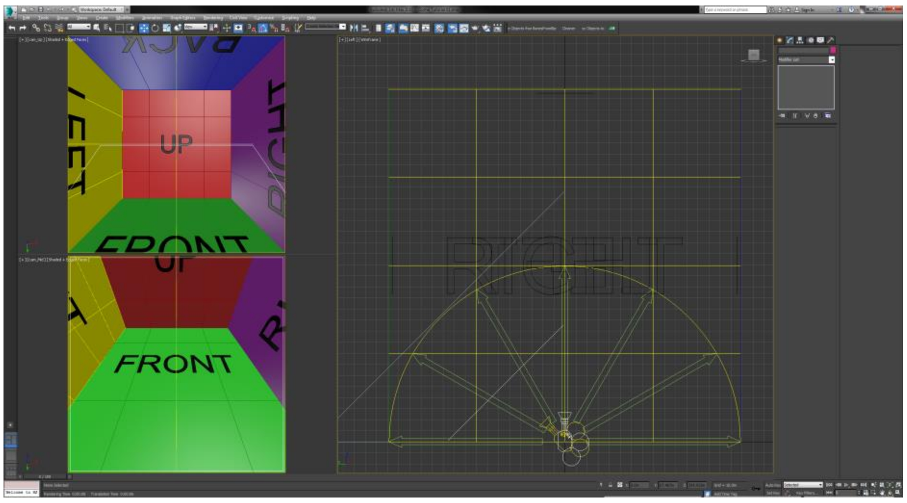
As we see, the render is a distorted, circular view. If we consider this to be a top view of a dome, the circumference would be the springline, and 'Up' would be at the zenith. What we see from the cam_Mid will be our sweetspot as outlined by the white square in the lower half of the plate.

As you can see, we only render the top half of the 'Front', 'Back', 'Left' and 'Right planes, this is because our camera is centre to the cube, and the springline cuts through them.

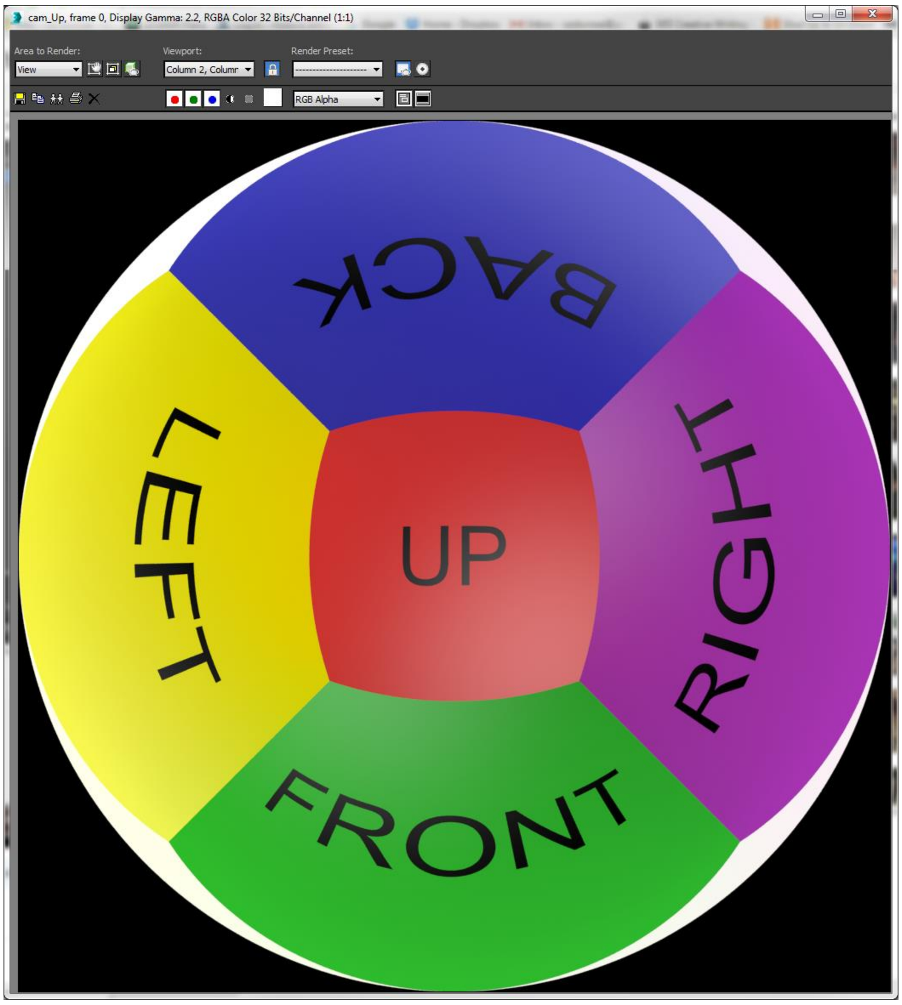


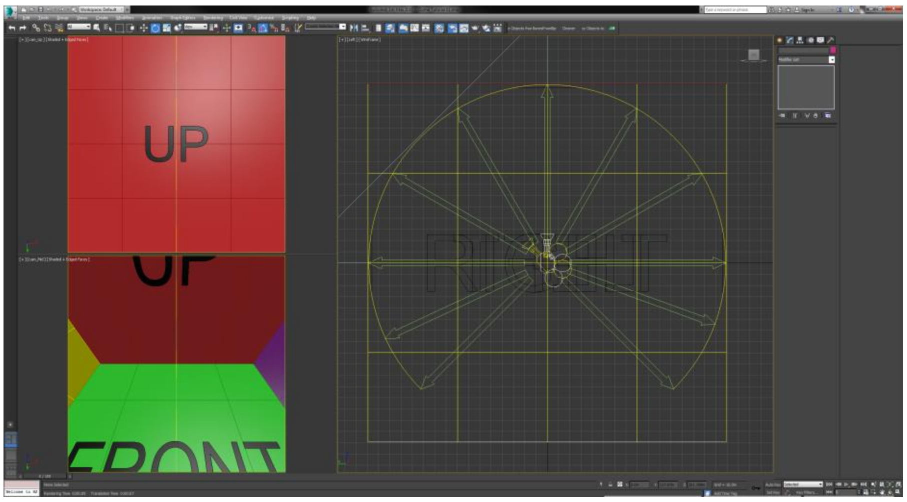
If, however, I was to render again but with my camera at the base of the cube, we would get a completely different perspective.





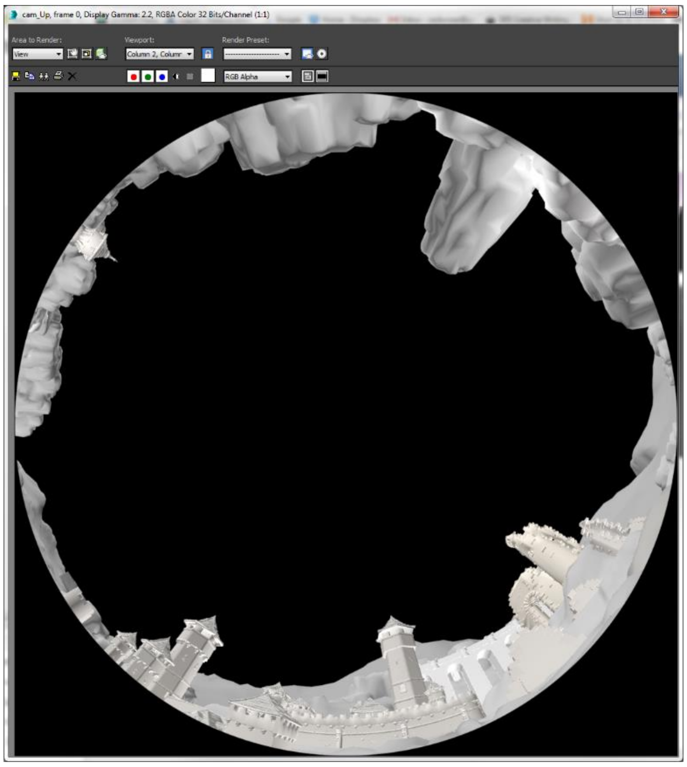
If I was to go back to the centre position and change the Field of View in the lens shader to 270 degrees, we over render beyond the typical dome.



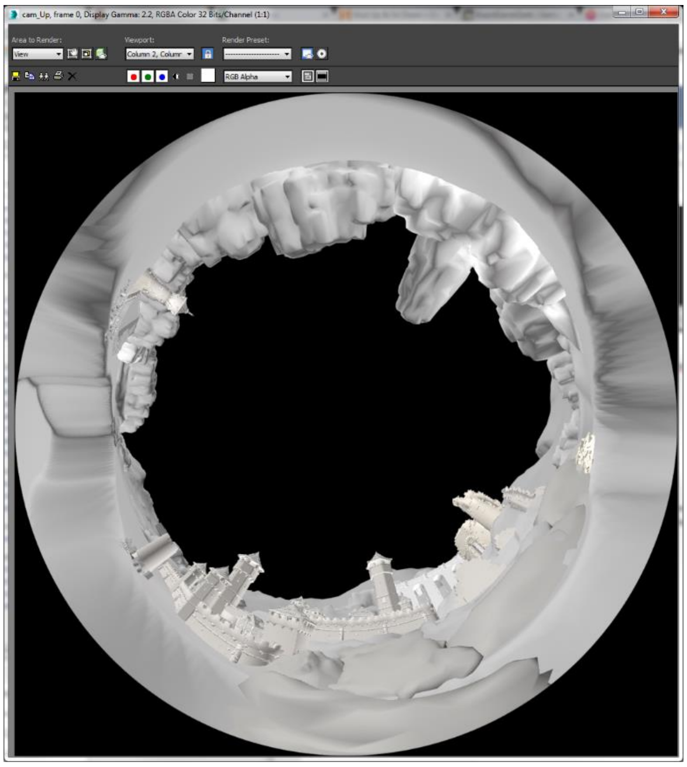


I will reset the camera to FOV:180, and render the scene shown before.

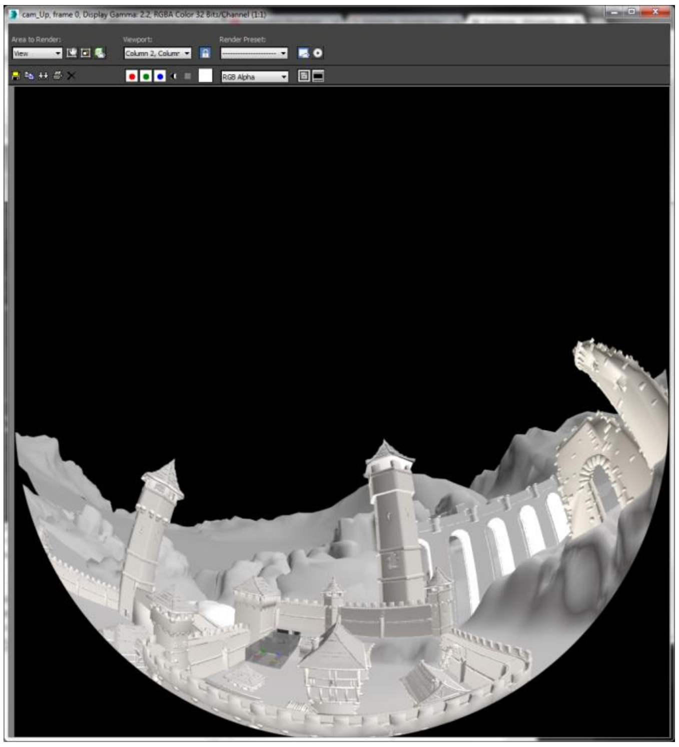
With a straightforward camera, we can see how boring the scene is. We get the sense that there is stuff all around us, but it is stuck to the springline. This is because we haven't considered the cam_Mid to be our focus.

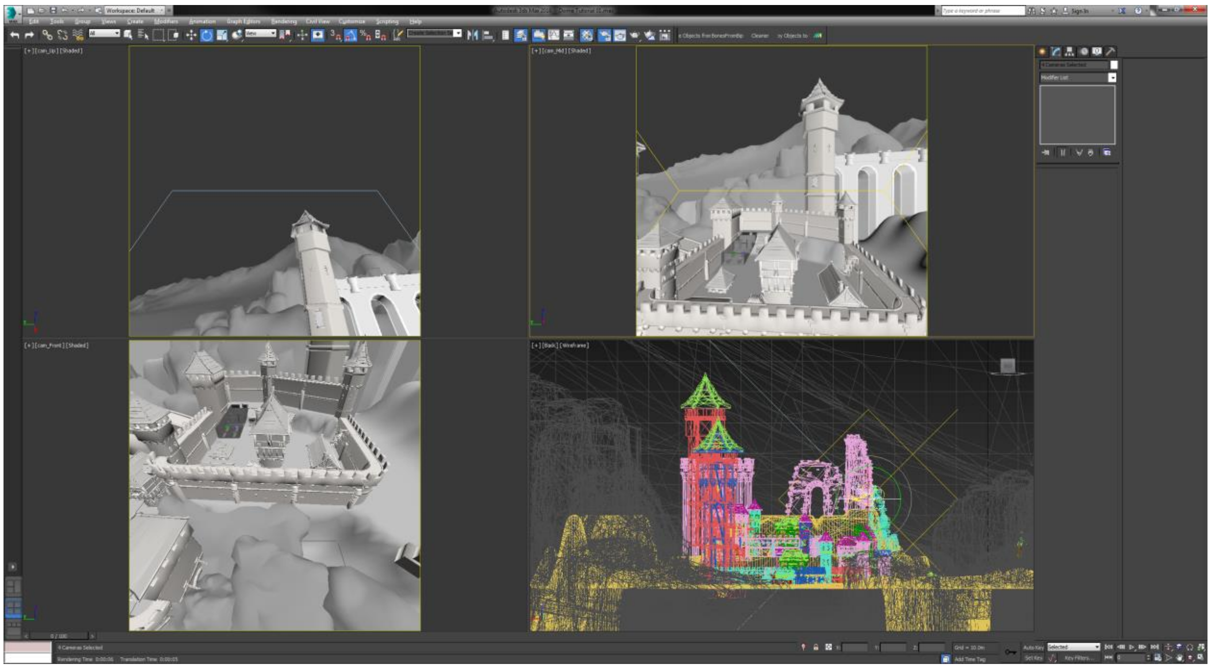

cam_Up, frame 0, Display Gamma: 2.2, RGBA Color 32 Bits/Channel (1:1)
Area to Render:
View
Viewport:
Column 2, Column
Render Preset:
RGB Alpha

However, if I pushed the FOV to 270, I could get a decent render for an omni-directional dome, having some points of interest all around us.


cam_Up, frame 0, Display Gamma: 2.2, RGBA Color 32 Bits/Channel (1:1)
Area to Render:
View
Viewport:
Column 2, Column
Render Preset:
RGB Alpha

Resetting to FOV:180, I will properly frame the cam_Mid and get a more interesting frame. Nothing would be happening behind us (top of the render), but all of the action would be happening in front of us. This is perfect for uni-directional domes.

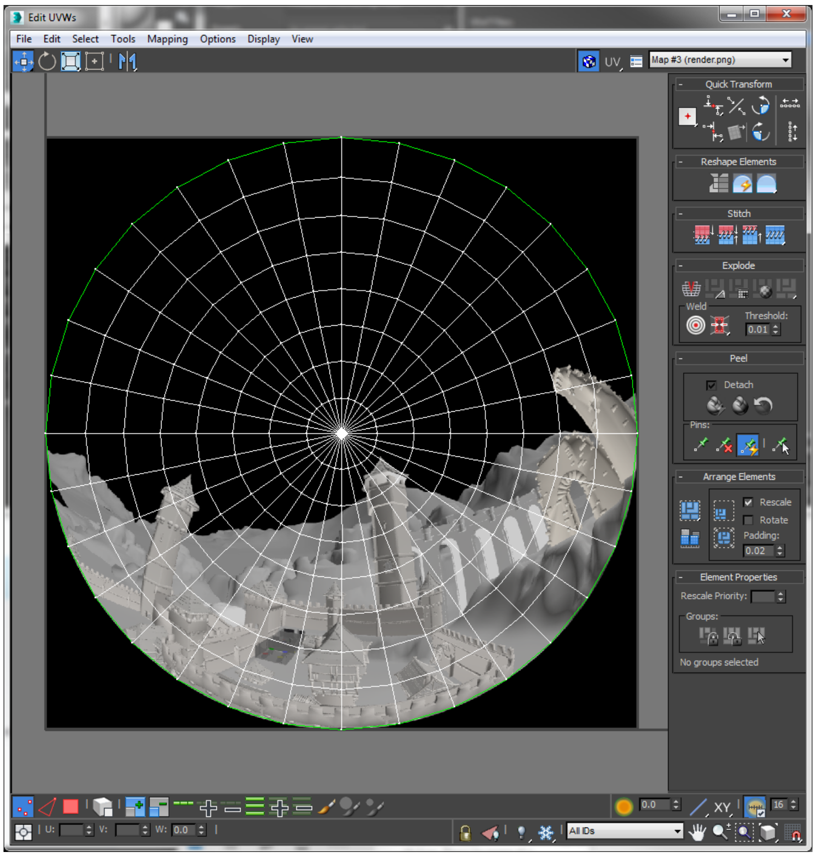

cam_Up, frame 0, Display Gamma: 2.2, RGBA Color 32 Bits/Channel (1:1)
Area to Render:
View
Viewport:
Column 2, Column
Render Preset:
RGB Alpha



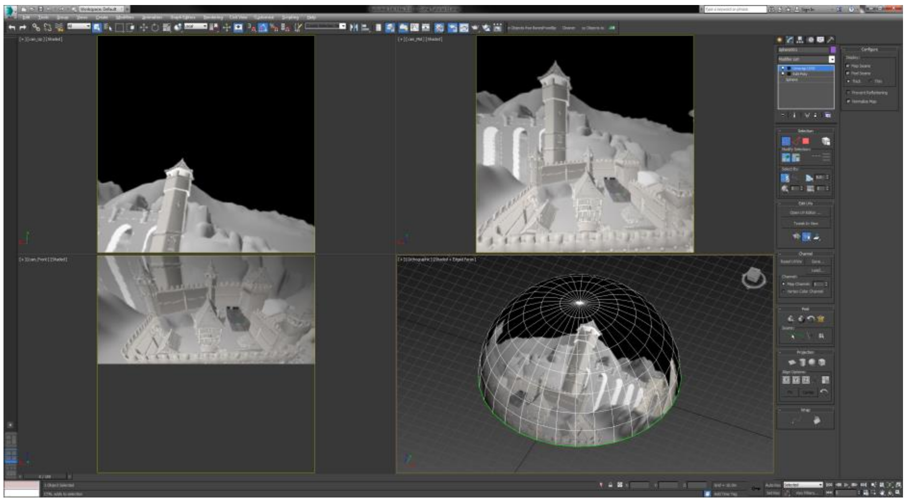
Fitting to a Dome

So an ordinary dome would take a standard fisheye plate and split it up into different sections that would then be projected from round 5 or 6 projectors set around the bottom of the dome. This is usually done by the dome's own software, and so is nothing we would have to worry about, but to show how the fisheye frame works on a dome surface, I will simply texture a hemisphere.

Taking our hemisphere, I will planar unwrap from above, and then relax the vertices whilst keeping the boundary edges static, creating an equally placed web of edges.



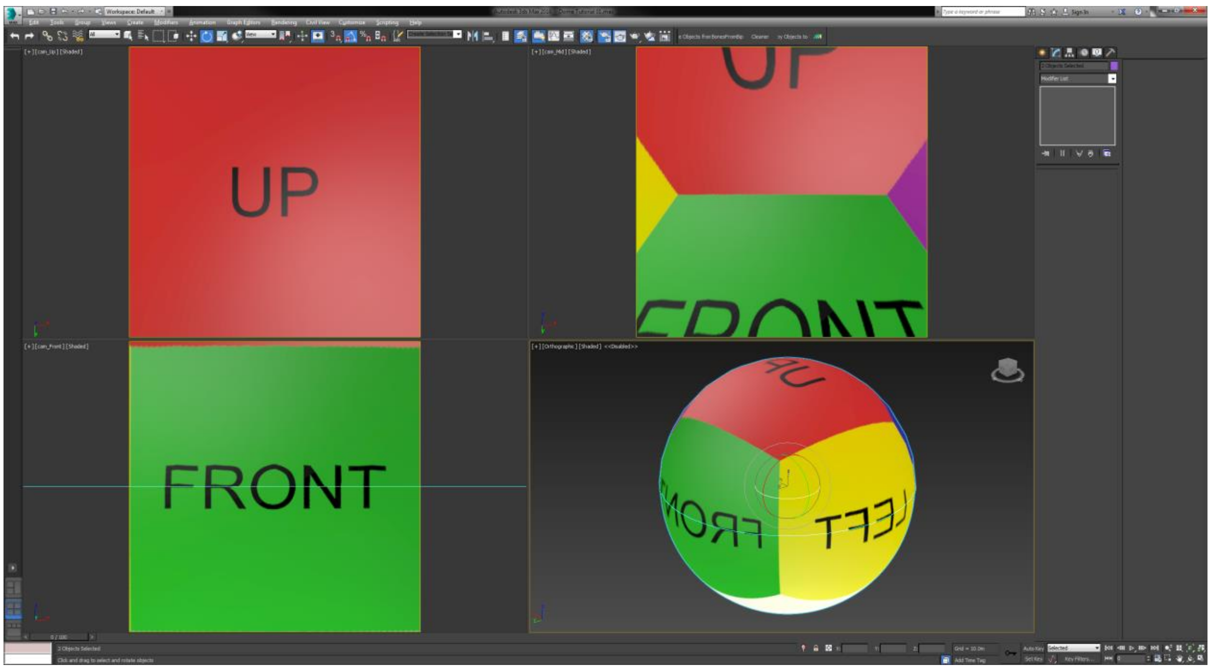
We add the rendered image and can see that the springline of our dome fits the circumference of our rendered image. When the image is applied to the dome, we see very little difference between a scene with all of the geometry in there, and a scene with a simple dome render.



VR purposes

This process could be used not only for Dome purposes, but also for VR purposes. If we took two fisheye images of a scene – a top and a bottom, and placed them on two hemispheres arranged into a sphere, we could create a 360 x 360 degree experience for pre-rendered images.

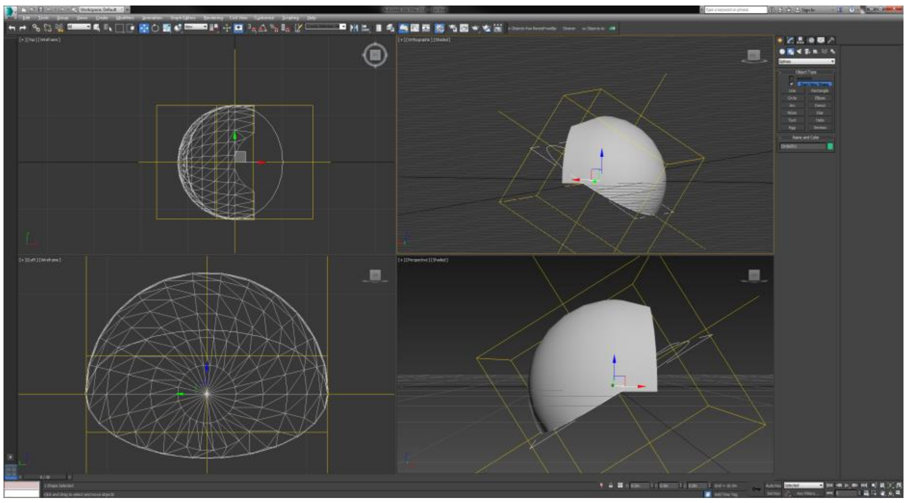
This is key – this would only be for pre-rendered animation or media, there would be no way to incorporate interactivity into this method of VR, but is a great and very optimised way of showing extremely high quality models in a VR setting.



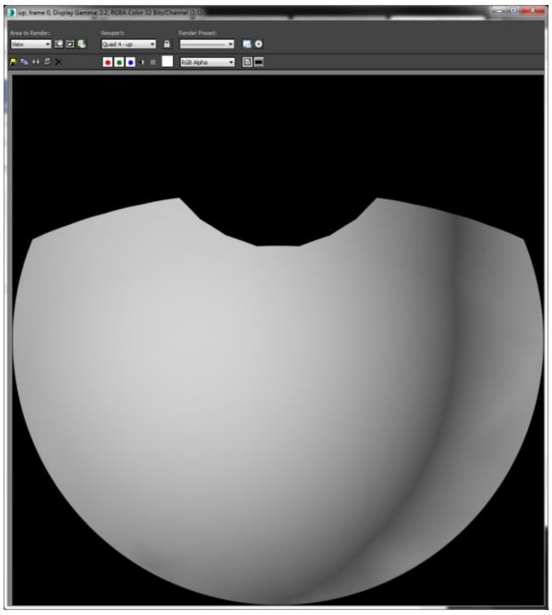
This example shows the possibility of this process in work, though there may need to be some more effort put into the UV map.

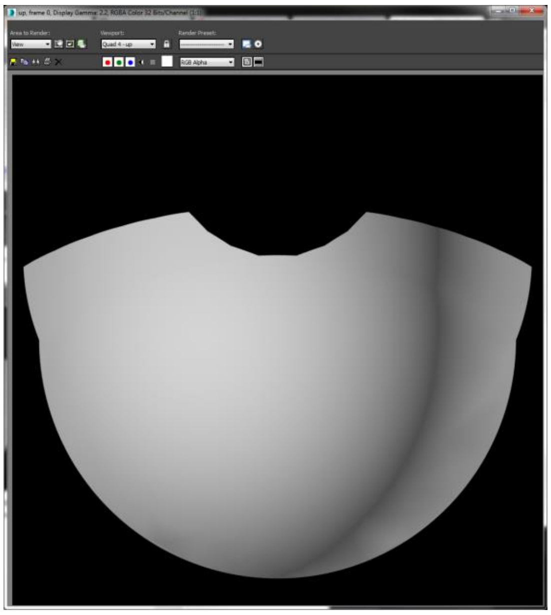
Unconventional Dome Shapes

Here I have an unconventional Dome shape. It is a 12 meter radius dome with a -32 degree tilt and an open back half (180 degree springline) but it also has an additional bit on the end. These jutting bits mean that I would have to overrender the domes FOV to compensate. The fact that the jutting bit comes out straight is nothing I can worry about at the moment.



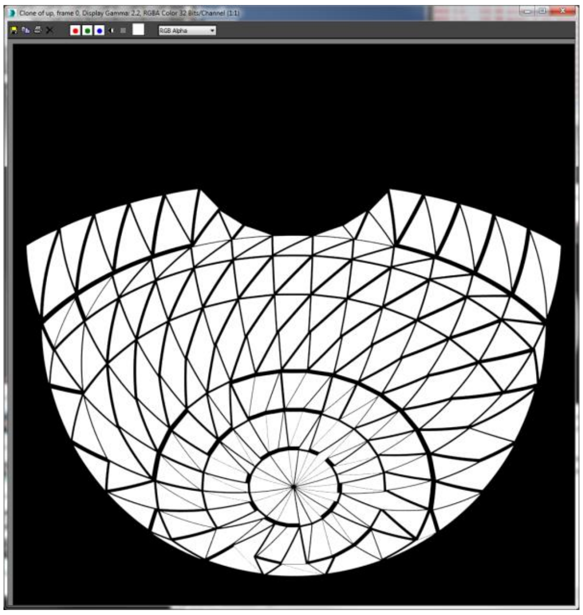
I will create a fisheye camera to fit the shape and try to estimate how much FOV we will need to compensate for the added sections.



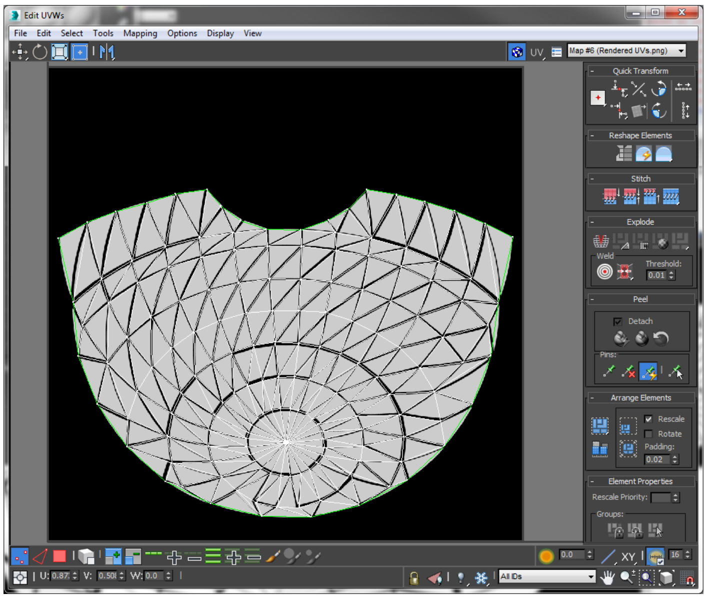


This is our normal FOV:180 fisheye image from the centre of this dome. We see that the jutting parts are out of frame. But by over-rendering with an FOV of 200, I can render something that fits the whole frame.

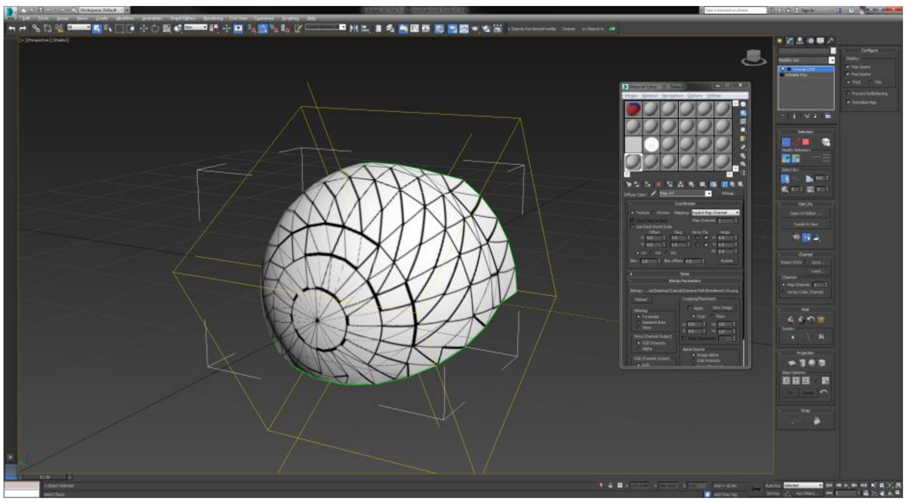
I will create a black and white material with a stepped, boxed gradient material to apply to the dome object. When rendered out with the material gradient per face, we will render out something akin to the wireframe.


Clone of up, frame 0, Display Gamma: 2.2, RGBA Color 32 Bits/Channel (1:1)
RGB Alpha

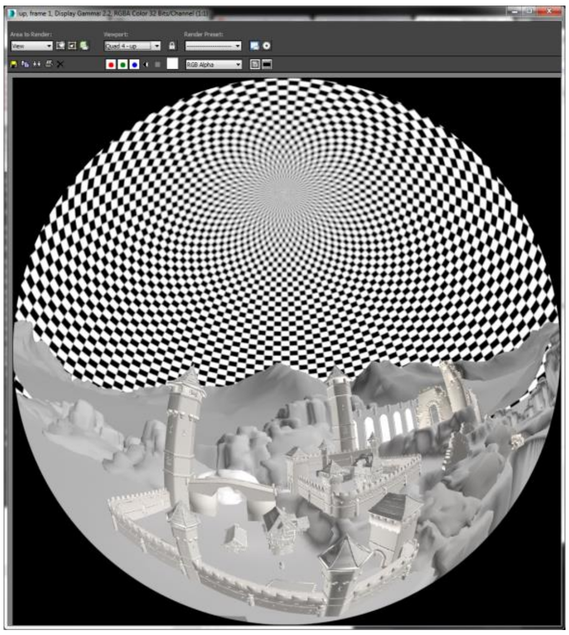
We can use this to manually fit our domes UVs to. This is close enough.

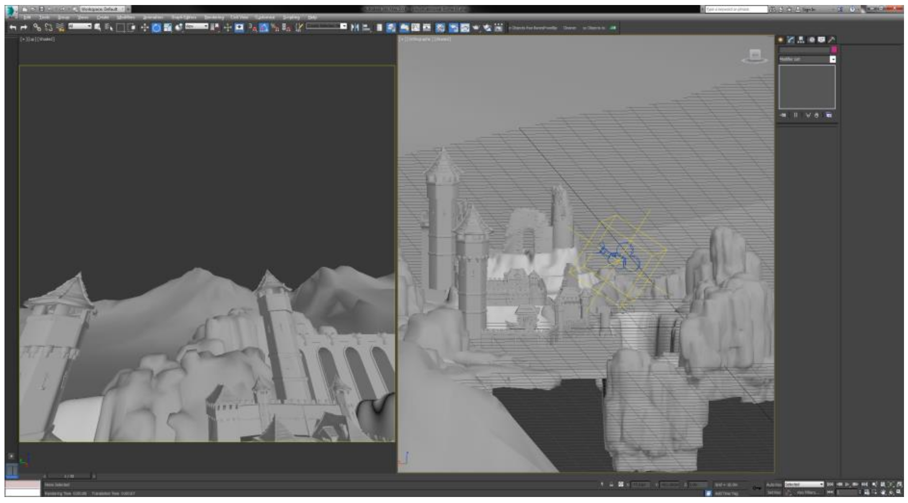


So now if we apply the render to the dome, we should have something that fits it’s edges.

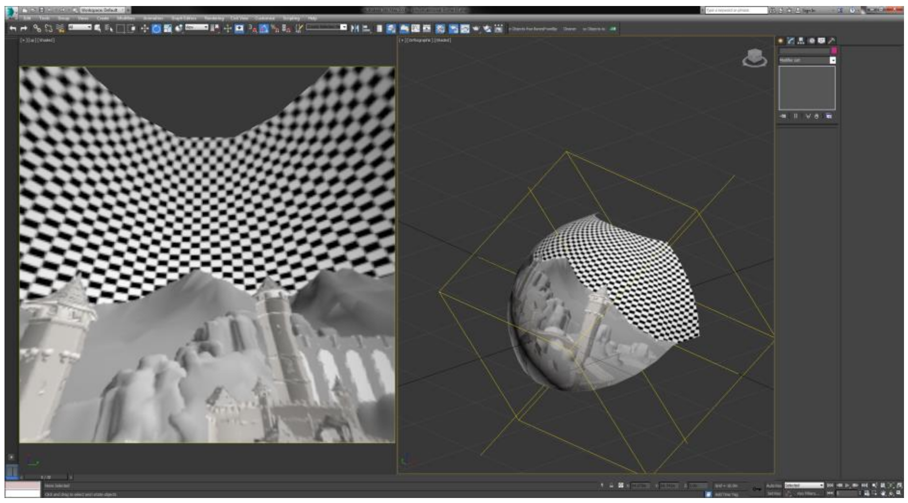


So with this in mind, we can now make a render of our scene with the FOV:200. I have added a 50x50 checker map to the environment to see how much stretching we would have on the dome.

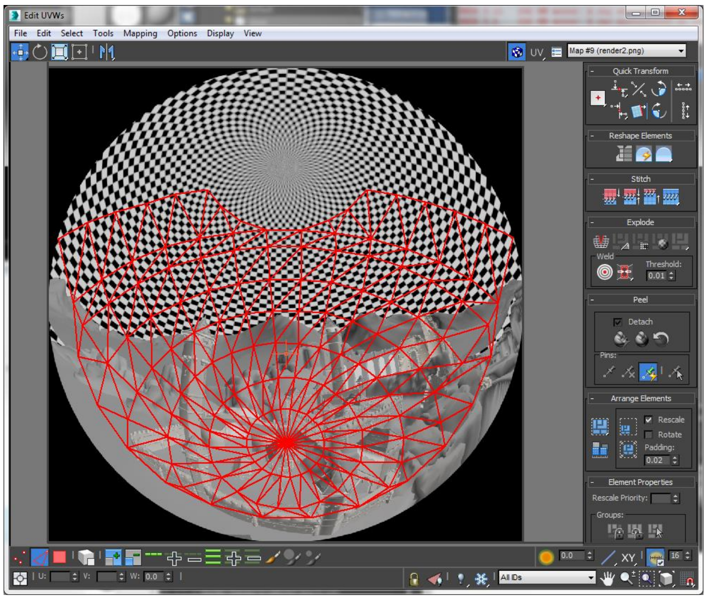




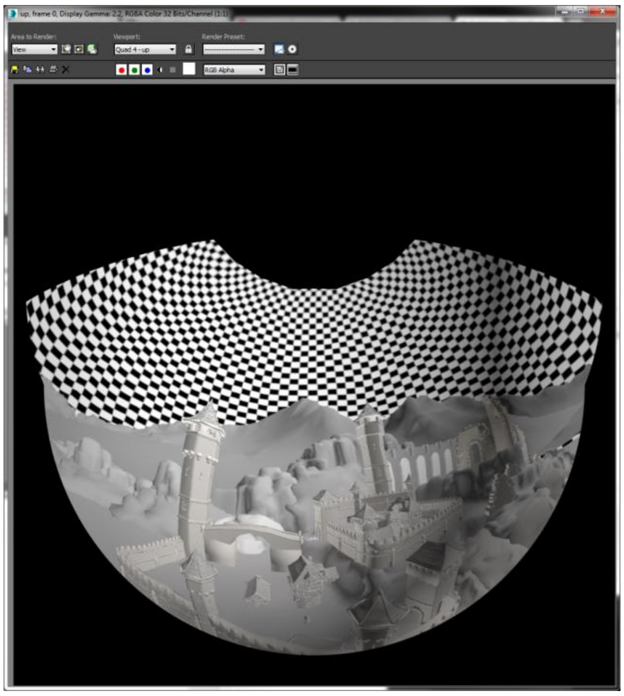
Now hiding the set and applying this render to the dome we can see that in the cam_Up view, the render fits the setting geometry perfectly.



In the UV editor window, we can see that the only pieces we are seeing are outlined in red, and so there is a lot being over rendered, which will add a lot of time to the rendering unless we created an inverse of the dome object to act as a render mask.



Re-rendering the dome object only, we can see that this has worked quite nicely.



And looking around the dome object we can confirm that this would be good enough.