700106 Lab Book

Contents

[Week 1 – Lab 1 1](#_Toc120088731)

[Week 2 – Lab 2 8](#_Toc120088732)

[Week 3 – Lab 3 15](#_Toc120088733)

[Week 4 – Lab 4 23](#_Toc120088734)

# Week 1 – Lab 1

Date: 28th Sep 2022

## Q1. Exercise 0

### Question:

Change the “Eye” view so that you can see the top of the animated cube

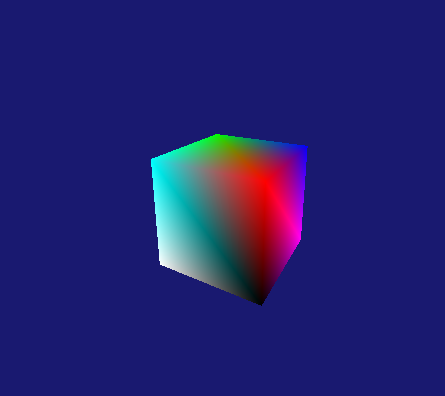
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

This was very straight-forward just had to change the y position of the “Eye” XMVECTOR

### Metadata:

Change of view

### Further information:

n/a

## Q2. Exercise 1

### Question:

Modify the vertex list in indices[] or modify the parameters in the DrawIndexed( ) to draw:

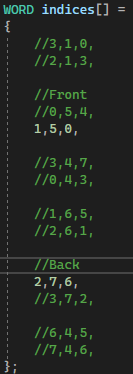
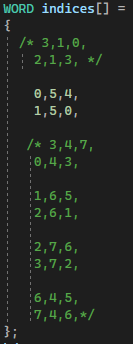
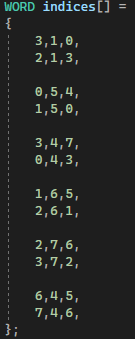
1. two triangles

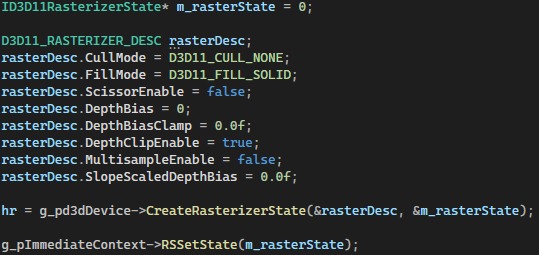
2. one face of the cube

3. the four walls of the cube

### Solution:

(1) (2) (3)

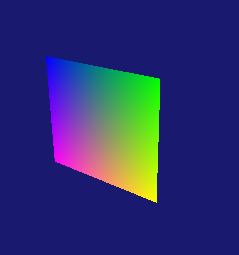
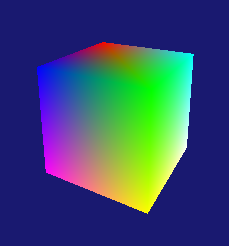


### Test data:

n/a

### Sample output:

1. 2. 3.

### Reflection:

The rasterizer seems complicated at the moment but everything else was fairly easy

### Metadata:

Faces

### Further information:

n/a

## Q3. Exercise 2

### Question:

Draw the cube as a wireframe

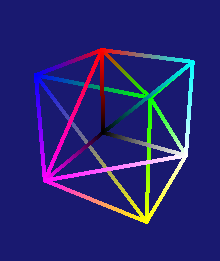
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

This was really easy to do. I only had to change half a line of code

### Metadata:

Wireframe

### Further information:

n/a

## Q4. Exercise 3

### Question:

Modify the parameter in “IASetPrimitiveTopology( )” and “indices[]” to draw:

1. A list of points corresponding to the cube’s eight vertices.

2. The 12 edges of the cube (not as a wireframe triangle mesh).

### Solution:

1.



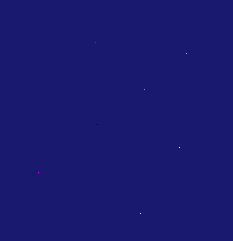
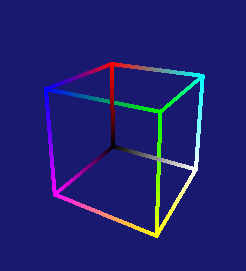


### Test data:

n/a

### Sample output:

1. 2.

### Reflection:

It took a while to figure out part 2, I was drawing too many lines for what was needed

### Metadata:

Topology

### Further information:

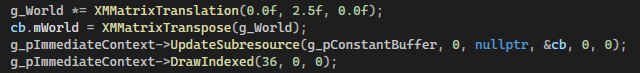
n/a

## Q5. Exercise 4

### Question:

Draw two wireframe cubes

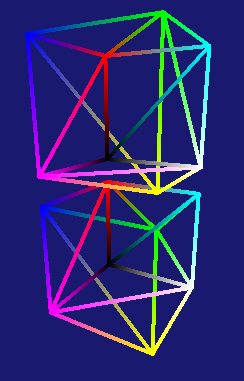
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

Figuring out where I needed to put the code was the only hard part, but at least now I know.

### Metadata:

Twice the cubes

### Further information:

n/a

## Q6. Exercise 5

### Question:

Draw the cube as triangle strips by setting primitive topology as D3D11\_PRIMITIVE\_TOPOLOGY\_TRIANGLESTRIP

### Solution:

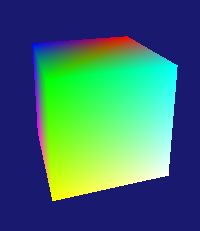




### Test data:

n/a

### Sample output:



### Reflection:

This took me way too long and I still couldn’t get it perfect. I understand triangle-strip conceptually, I just can’t visualise it clearly enough in my head to implement it perfectly.

### Metadata:

Triangle strip

### Further information:

How can I better visualise it to implement it correctly?

# Week 2 – Lab 2

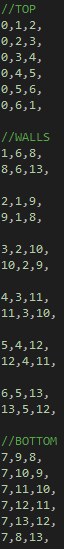
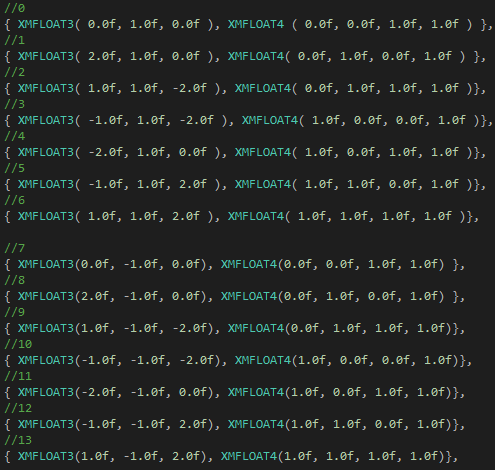
Date: 05/10/2022

## Q1. Exercise 1

### Question:

Modify the vertex list and indices to create a hexagonal cylinder

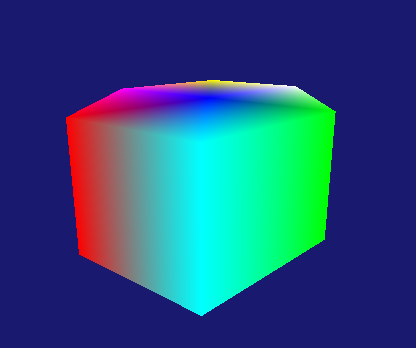
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

This took way longer the it needed because I missed a single line. I forgot to update the size of the vertex list. 

### Metadata:

Hex

### Further information:

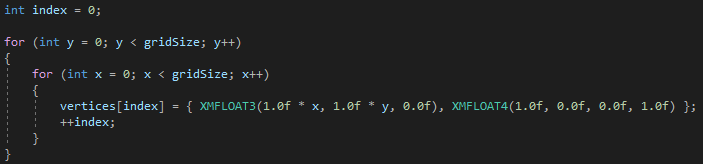
n/a

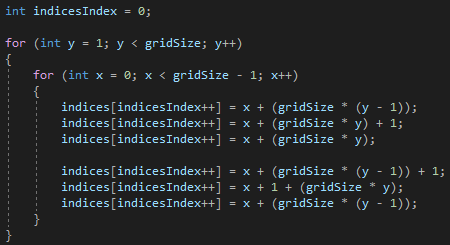
## Q2. Exercise 2

### Question:

Modify the cube vertex list in the sample to specify a flat 3D grid and display it as a wireframe

### Solution:

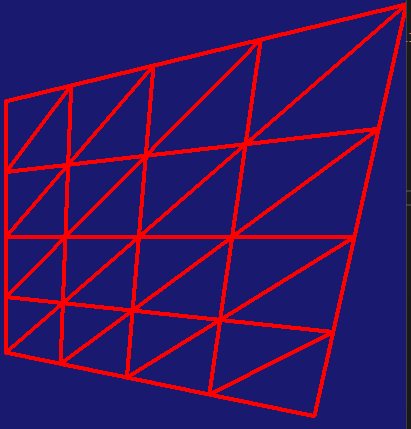
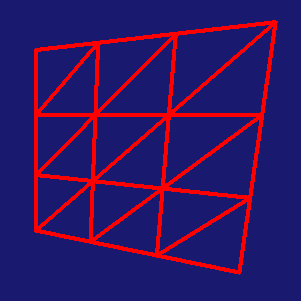




### Test data:

n/a

### Sample output:



### Reflection:

The procedural indices took forever to figure out how to do, I was over engineering the solution needed.

### Metadata:

Procedural

### Further information:

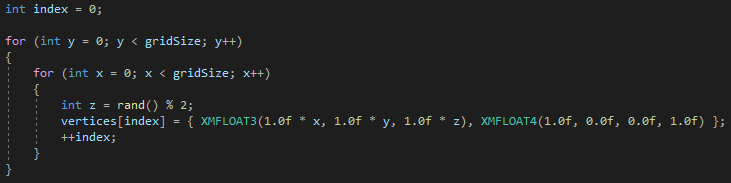
n/a

## Q3. Exercise 3

### Question:

Specify different heights at different grid points for the 3D grid you created in Exercise 6 to create a terrain triangle mesh

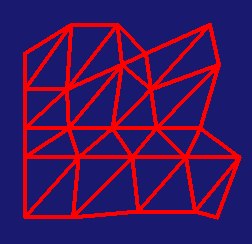
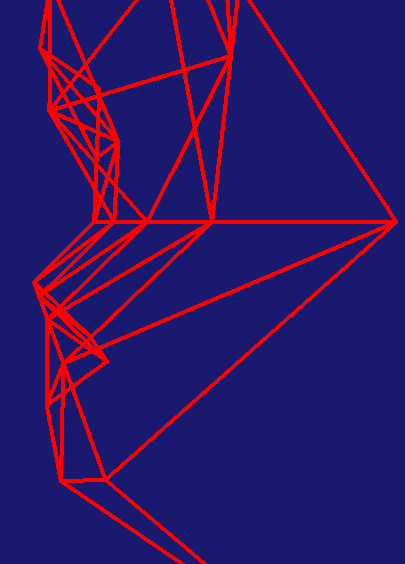
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

This was pretty straight-forward once I got the procedural indices to work. It’s a bit hard to get a good angle to show off that it has got an z axis. I probably should have done a wider range because at the moment the z axis is only ever 0.0f or 1.0f.

### Metadata:

Terrain

### Further information:

n/a

## Q4. Exercise 4

### Question:

Use e “GeometryGenerator.cpp” , “GeometryGenerator.h” and “ShapesApp.cpp” to draw a sphere, cylinder and plane.

### Solution:

Text

Description automatically generated 

These are pretty much the same for all the shapes the only difference is the keyword changes e.g. “grid”.

### Test data:

n/a

### Sample output:

### Shape Description automatically generated A picture containing text Description automatically generated

### Reflection:

This was pretty straight forward once I had spotted the simple mistake I had made. I’m noy sure how you would draw all of them as once like in the “ShapesApp.cpp” example.

### Metadata:

Geometry generator

### Further information:

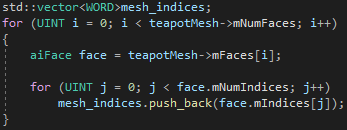
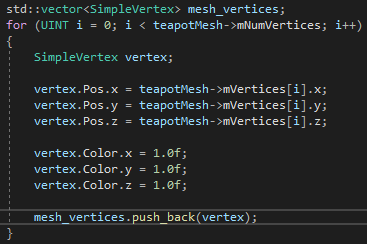
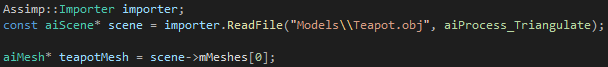
n/a

## Q5. Exercise 5 Loading graphics assets using AssImp

### Question:

Use AssImp to draw the teapot model given

### Solution:



### Test data:

n/a

### Sample output:

A picture containing kitchenware, pot

Description automatically generated

### Reflection:

Finally got this fixed nearly 10 weeks later, once the error with the includes and min had been fixed, I was having errors with the obj file becoming corrupt. I fixed this by loading the given obj into blender and then exporting it as a new obj which seemed to fix the problem but it might be because I tried doing this in the final lab code.

### Metadata:

Assimp

### Further information:

n/a

# Week 3 – Lab 3

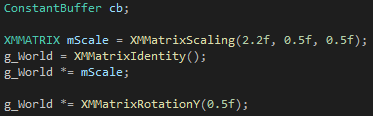
Date: 11/10/2022

## Q1. Exercise 1

### Question:

Transform the cube using scaling transformation

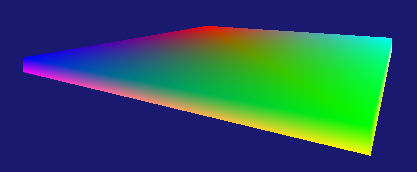
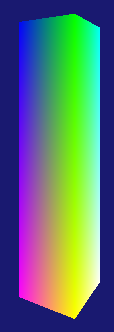
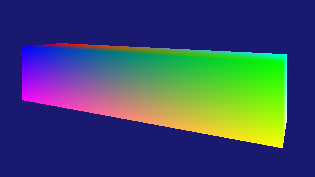
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

These were really easy to do, the only challenge was getting the proportions right to make shapes like the examples

### Metadata:

Scaling

### Further information:

n/a

## Q2. Exercise 2

### Question:

Perform scaling, translation, and rotation transformation to achieve the following effects:

1. A cube rotates by a vertical rotation axis
2. Two cubes rotate by two different rotational axes with different rotational speeds respectively

### Solution:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

### Test data:

n/a

### Sample output:

1. 2.

Shape

Description automatically generated A picture containing icon

Description automatically generated

### Reflection:

This was fairly simple to do once I realised that I needed to create a depth stencil to make the cubes go behind the pillar properly.

### Metadata:

Cube around a pillar

### Further information:

Transformation order is Scale the rotation then translation. However, different orders can be used for different effects

## Q3. Exercise 3

### Question:

scale the small rotating cube into a long-thin stick and rote the stick by a rotation axis such that the stick is always tangent to the rotation path.

### Solution:

Text

Description automatically generated

### Test data:

n/a

### Sample output:

Background pattern

Description automatically generated

### Reflection:

This is the exact same as the last exercise except the scaling is different. Have I misunderstood what one of the exercises is asking for?

### Metadata:

Tangent rotation

### Further information:

n/a

## Q4. Exercise 4

### Question:

Scale the cube into different sizes corresponding to the Sun, the Earth and the Moon respectively and then combine a set of rotation and translation transformations to animate a simple solar system

### Solution:

Text

Description automatically generatedA screenshot of a computer

Description automatically generated with medium confidenceText

Description automatically generated

### Test data:

n/a

### Sample output:

### Shape Description automatically generated

### Reflection:

The sun and the earth were easy to do but the moon I can’t quite figure out. I can get it to be the right size and have its own spin, but I can’t get it to orbit the earth. I’m not sure how to make the origin point for the moons orbit be the centre of the earth cube.

My second attempt at getting the moon right looks like its work but its hard to tell if it actually is.

### Metadata:

Solar system

### Further information:

n/a

## Q5. Exercise 5

### Question:

Manually set the g\_View and g\_Projection and observe if you get the same effect

### Solution:

Text

Description automatically generated

Text, chat or text message

Description automatically generated

### Test data:

n/a

### Sample output:

### Shape, square Description automatically generated

### Reflection:

I seem to get same effect but this could be because I copied the matrices that get produce by the XMMatrixLookAtLH() and XMMatrixPerspectiveFovLH() and hardcoded them.

### Metadata:

Hardcoded matrices

### Further information:

n/a

# Week 4 – Lab 4

Date: 18/10/2022

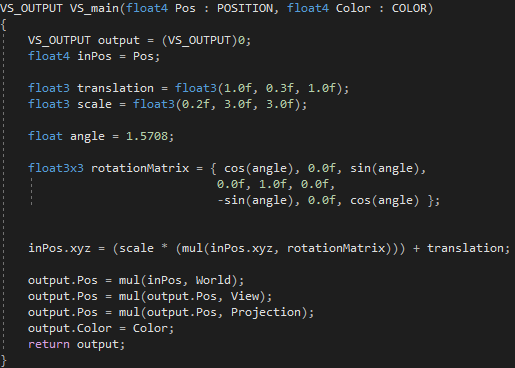
## Q1. Create your own vertex shaders

### Question:

1. Create a new vertex shader in “Tutorial04.fx” this vertex shader should translate by (1.0f, 0.3f, 1.0f) , scale by (0.2f, 3.0f, 3.0f) and rotate by the y-axis.
2. Create a new HLSL file and copy your new vertex shader into it then proceed to set it properties correctly.

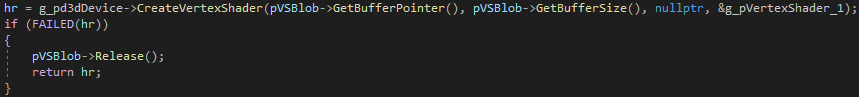
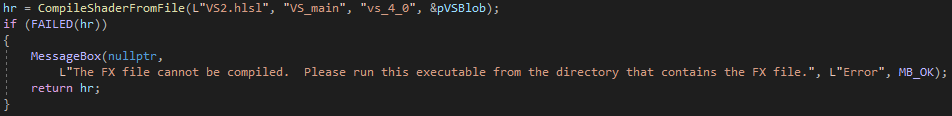
### Solution:

1.



2.



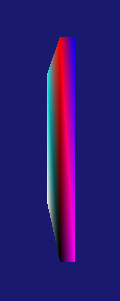


### Test data:

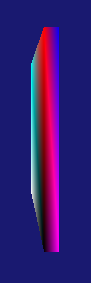
n/a

### Sample output:

1.



2.



### Reflection:

This took a while to do, I don’t really understand HLSL very well so I need to practice more. Switching from an .fx file to a .hlsl file didn’t seem to be much different.

### Metadata:

HLSL

### Further information:

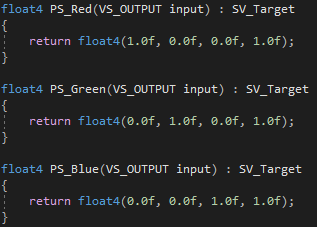
n/a

## Q2. Create your own pixel shaders

### Question:

Create pixel shaders to make 3 different coloured cubes

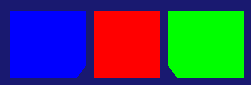
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

This was pretty simple to do. There is a lot of repetitive code to compile and create each pixel shader though, I’m sure that could improve on that by making a vertex shader and pixel shader class. I’m not sure why a chunk of the blue and green cube is missing, if I set the translations so each cube is right next to each other then it doesn’t happen

### Metadata:

Pixel shaders

### Further information:

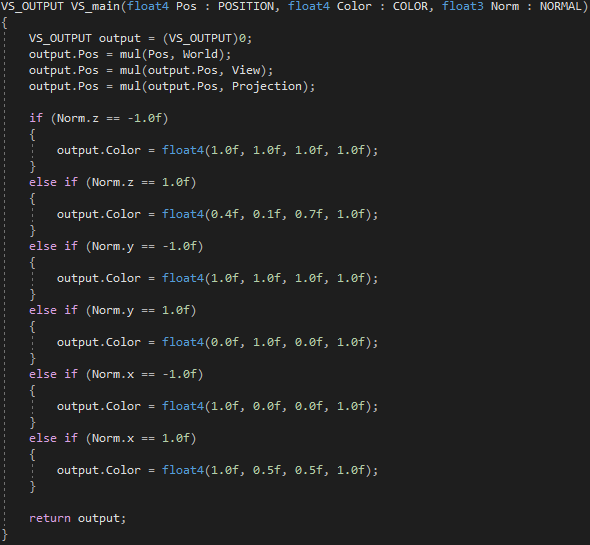
n/a

## Q3. Cornell box in vertex shaders

### Question:

Create Cornell box by using THREE different vertex shaders

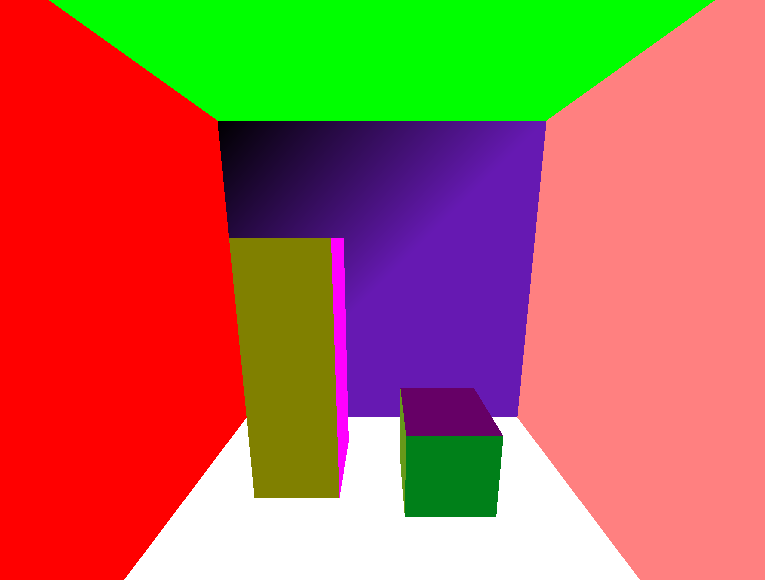
### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

It doesn’t look exactly like the example we were given but its close enough I think. For some reason the vertex shader wont colour the last vertex and I’m not sure why.

### Metadata:

Cornell

### Further information:

n/a

## Q4. Define Model-View-Projection in vertex shader (optional)

### Question:

### Solution:

### Test data:

### Sample output:

### Reflection:

As this is optional I will skip it for now as I am currently behind in the labs

### Metadata:

### Further information:

## Q5. Vertex shader point cloud (Optional)

### Question:

### Solution:

### Test data:

### Sample output:

### Reflection:

As this is optional I will skip it for now as I am currently behind in the labs

### Metadata:

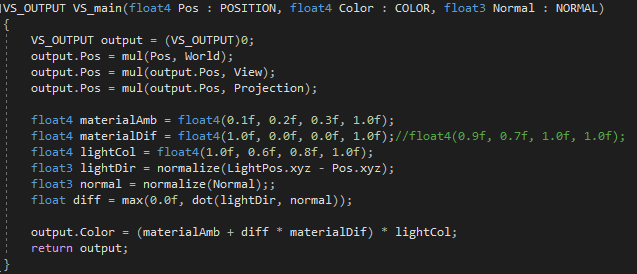
### Further information:

## Q6. Per-vertex diffuse lighting

### Question:

Create diffuse lighting in your vertex shader. Then modify the material reflection coefficient so that only red light is reflected

### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

This was pretty straight-forward to do there isn’t much to comment on. There was quite a lot of set up work to get this to work and I’m not sure if I really understand the depth-stencil texture and depth-stencil view.

### Metadata:

Light

### Further information:

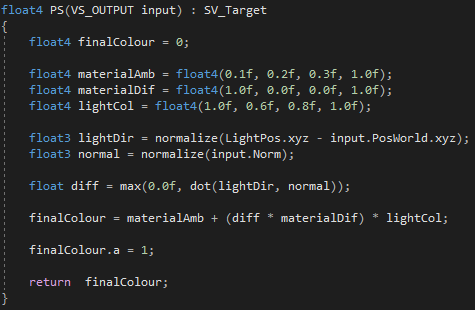
n/a

## Q7. Per-pixel diffuse lighting

### Question:

Create diffuse lighting in the pixel shader

### Solution:



### Test data:

n/a

### Sample output:



### Reflection:

Finally got this done it turns out I wasn’t passing the light position to the pixel shader. Lighting and Shader programming is still really confusing to me though

### Metadata:

Pixel lighting

### Further information:

n/a

## Q8. Per-pixel specular lighting

### Question:

Create specular lighting in the pixel shader

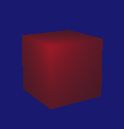
### Solution:

### 

### Test data:

n/a

### Sample output:



### Reflection:

This was fairly easy to do now that I have my pixel shader lighting working. I need to remember that with specular light that on this line:  the 30 represents shininess of the surface 1 being most shiny 200 being the least

### Metadata:

Specular Pixel

### Further information:

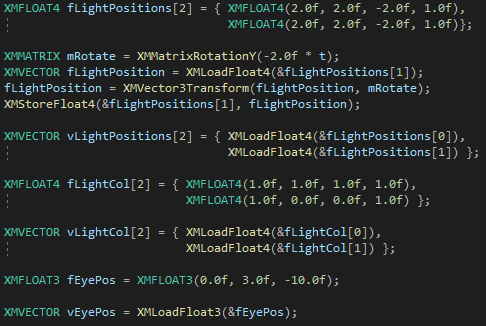
n/a

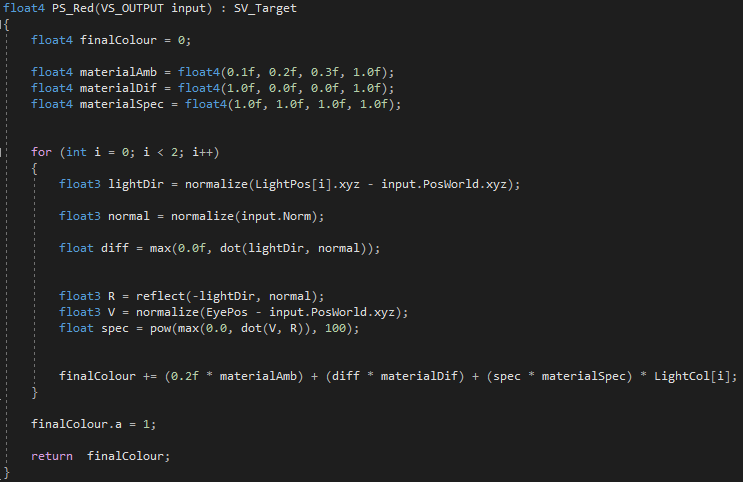
## Q9. Multiple materials and light sources

### Question:

Draw three cube objects with three different surface material properties. Illuminate these cubes with two light sources: one is a static white light, the other is a red light rotating dynamically by the y-axis.

### Solution:

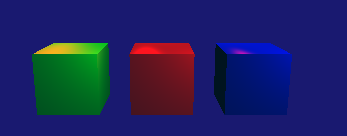




### Test data:

n/a

### Sample output:



### Reflection:

I’m not completely confident that I have done this correctly. Based on what I know and by looking at tutorial06 I think I have successfully done it.

### Metadata:

n/a

### Further information:

Can you confirm if I have implemented this correctly? and if I haven’t can you point out where I went wrong please. For some reason I’m struggling to understand lighting fully.

## Week 5 – Lab 5

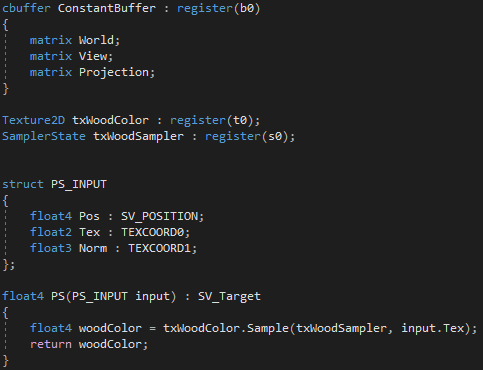
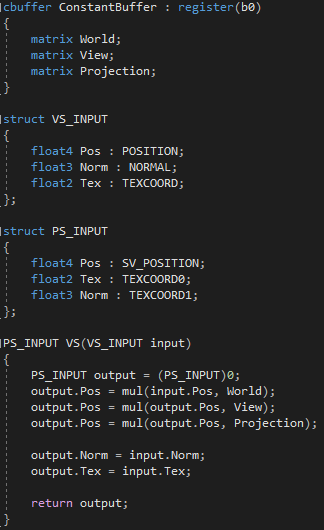
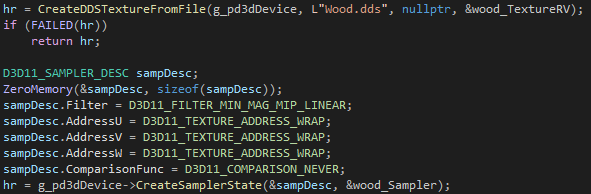
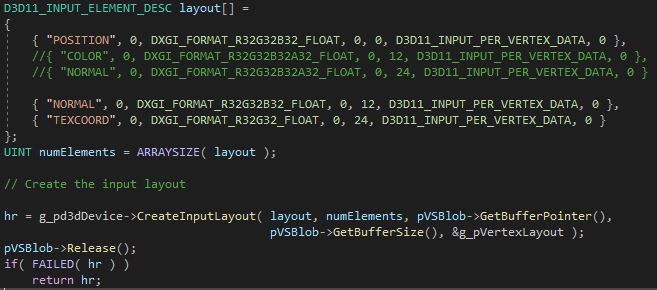
Date: 26/10/2022

### Q1. A wooden box

#### Question:

Give your cube a wood texture

#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was pretty straight-forward the only issue I had was that the layout creation was failing but I eventually found out I had just misspelled one of the matching variables in the vertex shader.

#### Metadata:

Wooden cube

#### Further information:

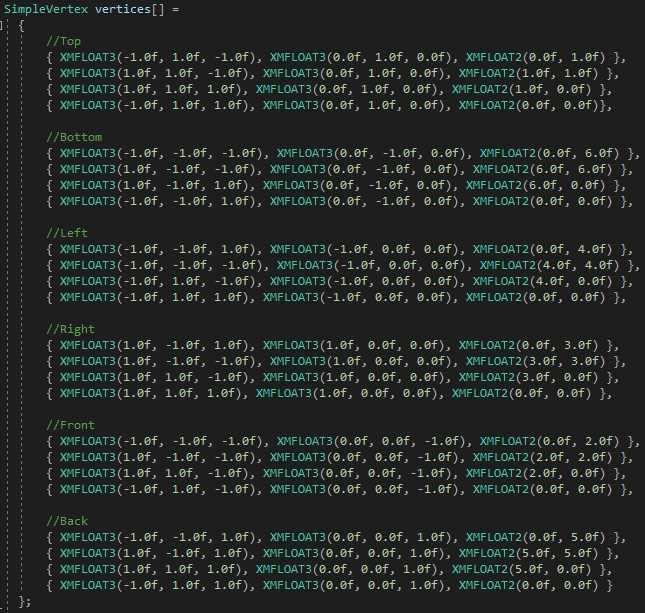
n/a

### Q2. Texture wrapping mode

#### Question:

Create a texture-mapped cube using the coin texture

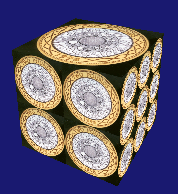
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was simple but I don’t really understand the other types of texture addresses or when I might want to use them

#### Metadata:

Texture addresses

#### Further information:

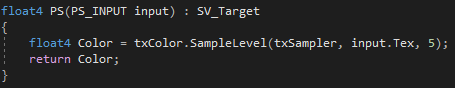
n/a

### Q3. Mipmapping

#### Question:

Create mipmaps of a loaded texture and use HLSL sampleLevel( ) to map the cube with different level of mipmaps

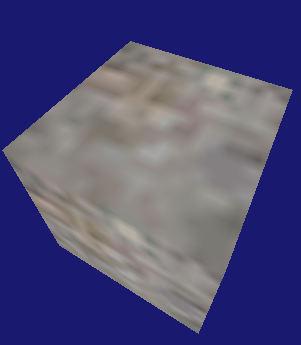
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This took way too long to do; I couldn’t get it to work but then after trying it in tutorial 7 and comparing the code I realised I had not set somethings in the sampler description properly.

#### Metadata:

Mipmaps

#### Further information:

n/a

### Q4. Texture filtering techniques

#### Question:

Scale the cube along the view direction to create a long rectangular object. Using different filtering techniques to deal with the minification and magnification issues on texture mapping and observe how visual quality of the rendered image is being changed.

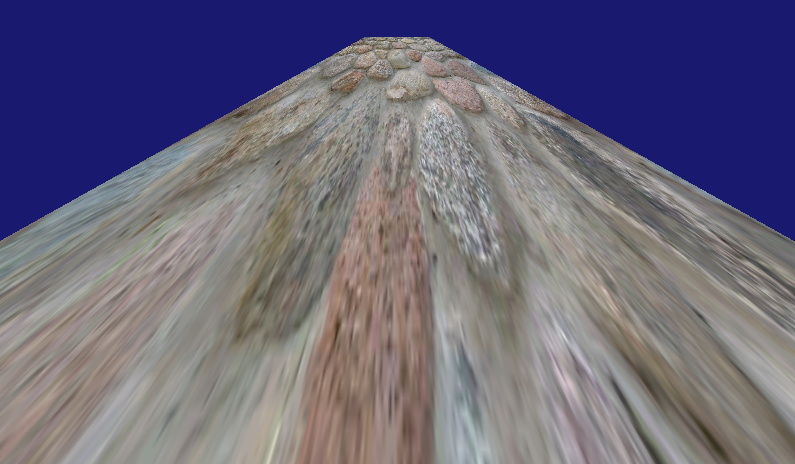
#### Solution:

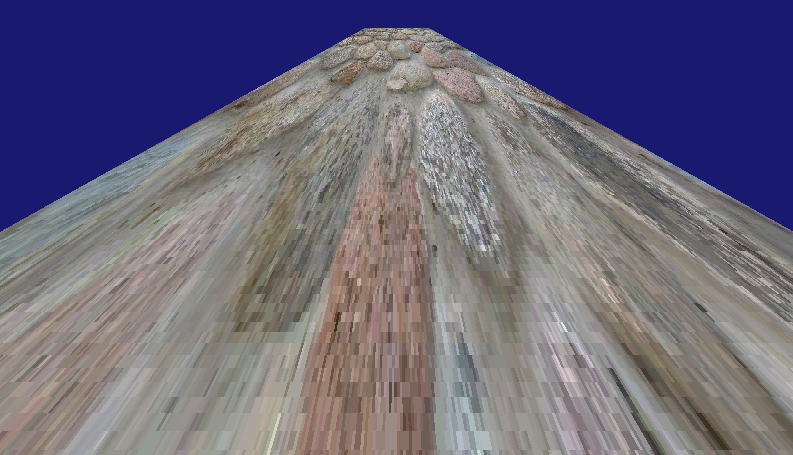
1. 
2. 
3. 
4. 

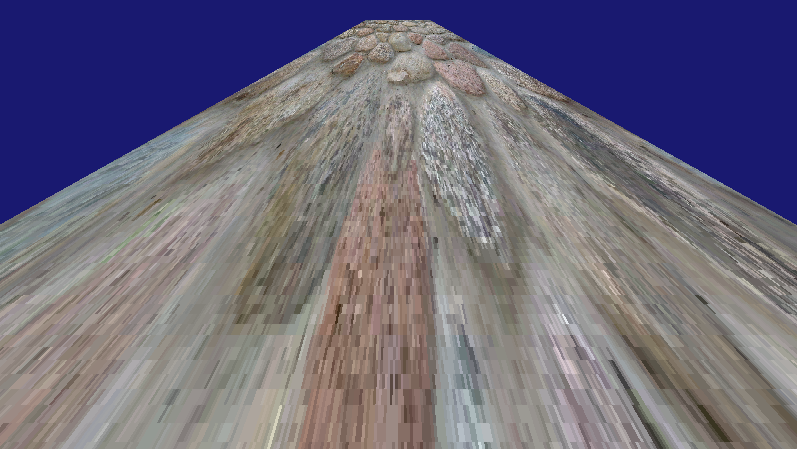
#### Test data:

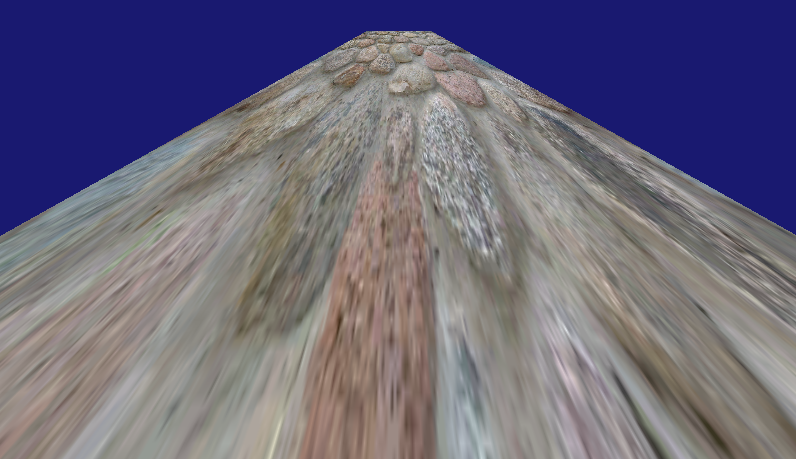
n/a

#### Sample output:









#### Reflection:

2 and 3 seem to reduce the quality of the texture and make it much more grainy. 1 and 4 seem to be pretty similar, 1 might be a little bit sharper. I’m not sure on the use cases for each filter I guess it just depends on the effect you are trying to achieve

#### Metadata:

Filters

#### Further information:

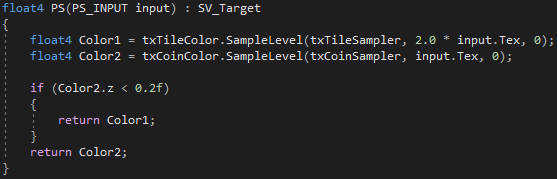
n/a

### Q5. Multiple texturing

#### Question:

Create a cube that uses the tile with the coin texture layered on top.

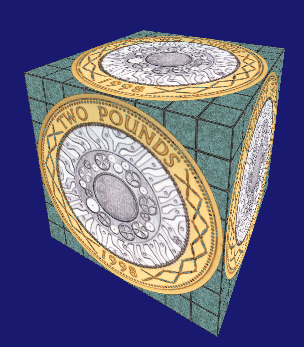
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This took a lot longer than it should of. I was trying to test the alpha of the texture by looking in the 4th float of a float4 but since the texture is 2-D the alpha is actually stored in the 3rd float of a float4. I think that’s right. Please correct me if I am wrong because I’m not full sure how I managed to achieve the desired effect.

#### Metadata:

Multiple Textures

#### Further information:

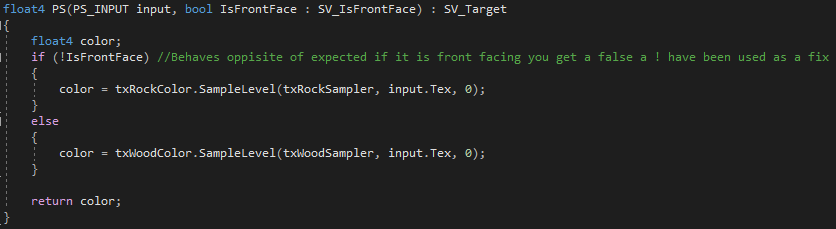
n/a

### Q6. An open box

#### Question:

Create an open box with a wood texture on the inside and a rock texture on the outside

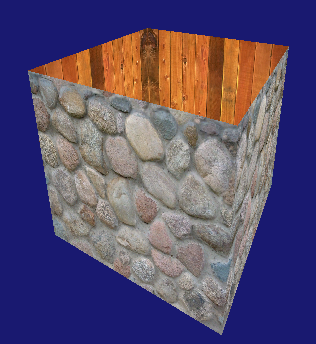
#### Solution:



#### Test data:

n/a

#### Sample output:



#### Reflection:

This was easy to do when I found out about the “IsFrontFace” semantic that DirectX has built in. Although for some strange reason the semantic returns a bool the opposite of what you’d expect. E.g. if the vertex is front facing it returns false when you’d expect it to be true

#### Metadata:

Stone and wood box

#### Further information:

Why does the “IsFrontFace” semantic work opposite to what makes sense?

## Week 6 – Lab 6

Date: 02/11/2022

### Q1. Normal mapped bumpy cube

#### Question:

Implement normal bump mapping

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information:

### Q2. Height map-based bumpy cube

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information:

### Q3. Procedural mapping

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information:

### Q4. Parallax mapping-based bump effect

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information:

## Week 7 – Lab 7

Date: 09/11/2022

### Q1.

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information:

## Week 8 – Lab 8

Date: 16/11/2022

### Q1.

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information:

## Week 9 – Lab 9

Date: 23/11/2022

### Q1.

#### Question:

#### Solution:

#### Test data:

#### Sample output:

#### Reflection:

#### Metadata:

#### Further information: