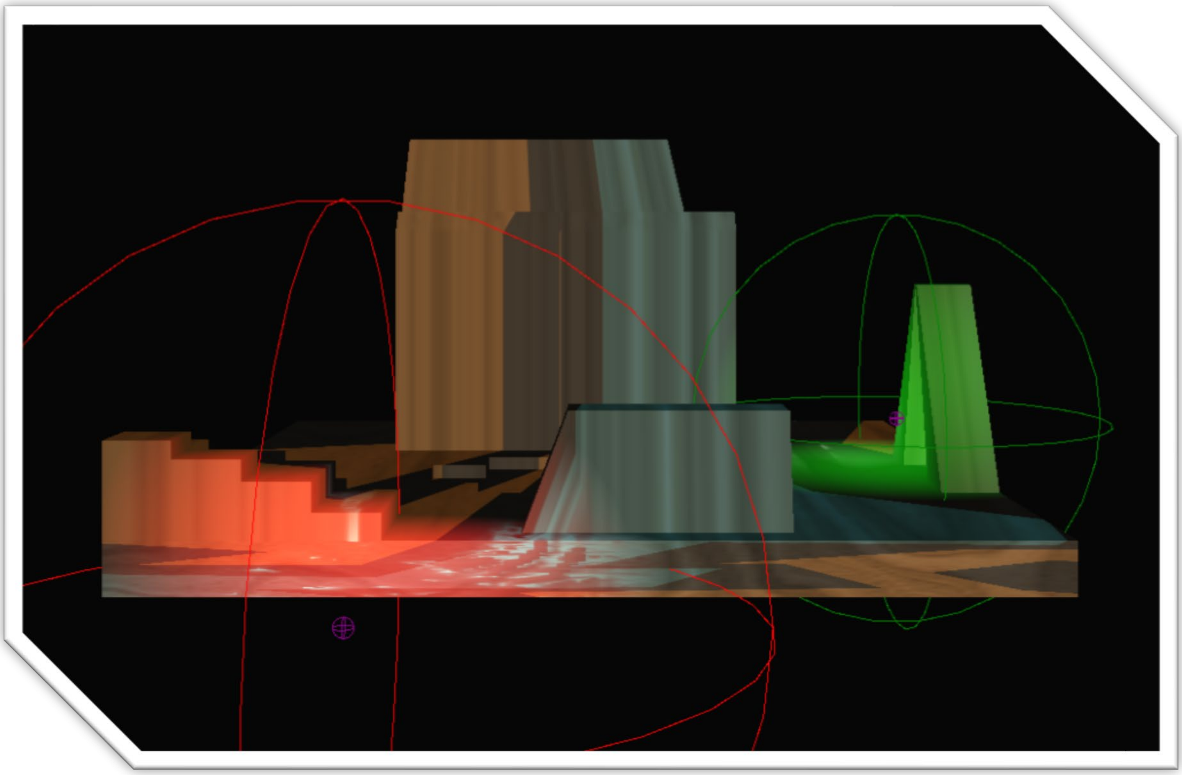
2020



3D Graphics Project

PROJECT DOCUMENTATION

RONAN CASSIDY – S00177754

Link: https://github.com/S00177754/3DGraphics302Project

Effect: Mixed Light Effect

Effect Overview:
This effect calculates a directional light alongside two-point lights using specular lighting, the effect also supports two diffuse textures and a single normal map. All lights share a common ambient and diffuse colour as they are affecting the same object.

Effect Variables:

- *#define PL_AMT 2* – This definition allows for the arrays used to be declared with a size of two when building the effect in the Monogame pipeline.

- *Float3 PointLightPositions[];* - This array is used to set the positions of the point lights; its size is set by PL_AMT as mentioned above.

- *Float PointLightAttenuations[]* - This array is used to set the attenuations of the point lights; its size is set by PL_AMT as mentioned above.

- *Float3 PointLightColors[]* – This array is used to set the colours of the point lights, as there is no default handling for an array of C# Colour variables to be passed into the effect, I wrote an extension method that converted the array of colours into an array of Vector3 which can be passed in.

- *Float PointLightFallOff* – This variable is used in the calculations of the point light attenuation and represents the inverse square law, so it is set by default to a value of 2.

- *Float3 DirectionalLightColor* – This variable allows for the colour of the directional light to be set, it uses a float3 as HLSL does not have a colour variable like C#.

- *Float3 DirectionalLightDirection* – This variable allows for the direction in which the directional light is facing to be changed, by default it is set to be facing in the same direction as the camera to illustrate the effect.

- *Float3 DiffuseColor* – This variable allows for the colour of the object to be set, to ensure the texture colour remains the same, we are using Color.White.

- *Float3 AmbientColor* – This variable is responsible for the colour of the object when there is no light source, a default lighting amount as per se.

- *Texture2D DiffuseTextureOne* – This variable holds the first diffuse texture to be applied to the model.

- *Texture2D DiffuseTextureTwo* – This variable holds the second diffuse texture to be applied to the model.

- *Texture2D NormalTexture* – This variable holds the normal texture to be applied to the model, this is used when calculating the specular light as a bump map.

- *Float3 CameraPosition* – This variable is updated with the position of the camera in the update method of our model class and is used for calculating the specular light in terms of the angle of the camera view to the point.

- *Float3 SpecularColor* – This variable allows for the color of the specular light to be set from C#, this is used in the specular calculations.

- *Float SpecularPower – This variable represents the strength of the specular lighting.*

Vertex Shader

Vertex Shader Function: VertexShaderOutput MainVS(in VertexShaderInput input)

Method Overview:

The input for the pixel shader is calculated in this function, we calculate the position of the vertex in regard to the world projection matrix, we also calculate the normal of the vertex here. Other calculations preformed here are setting the UV co-ordinate and the view direction of the camera for use in calculating the specular lighting later on in the pixel shader.

Pixel Shader

Pixel Shader Function: float4 MainPS(VertexShaderOutput input)

Method Overview:

The final colour of the vertex is calculated here and is then passed on as a float4 variable, the operations carried out in the method are broken up into individual methods for modularity and readability. The first thing carried out is the calculations for the normal mapping which will then be passed into the other methods to calculate the lighting values.

From here we are going onto our calculate diffuse method which take sin the UV value for the vertex and returns float3, this method applies the diffuse textures using sampling as well as the diffuse colour and is then passed back out and set as float3 FinalColor.
Next is our directional light calculations, here we add to FinalColor after calculating how the directional light would affect the object's colour as well as carry out specular lighting calculations on the object.

We then perform two additions of the create point lights methods as we have two point lights, this method calculates the attenuation and specular of the point lights and applies them to the FinalColor. The FinalColor is then passed out of the effect.