Dominik Wylie video description:

In my project I used

- vertex manipulation to create the mountain and the waves.
- lighting techniques – ambient, diffuse, specular and shadow mapping.
- Motion blur as post processing.

Vertex manipulation

I used a heightmap from an image and altered the height map from a flat plane, I also created a simple texture to use on the height map. I used im gui to alter the height of the landscape

I created water with simple sin waves that I can edit the speed, frequency and amplitude.

Lighting Techniques

I created diffuse and ambient lighting, ambient lighting is the darkest a level will be and diffuse is calculated by getting the distance from the light to the mesh texture, finding the angle between the light vector and the normal vector and adding lighting to the texture at that point depending on the distance, close for bright light and dimmer for farther away. And the normal angle, brighter for closer and dimmer for steeper until it hits farther than 90 degrees where it arrives at zero diffuse light.

I also used specular lighting. Specular lighting is when we find the reflection from the camera around the surface normal and then check that against the light vector using a dot product. That lets us find the angle between our camera reflection vector and the light vector. If the angle is small it is bright and if the angle is large it's much dimmer until it is zero. We can edit the light amount by changing essentially the glossiness, in my program it's called specular power.

Motion Blur shader

I created a motion blur for my full screen filter. The blur works by taking 2 texture inputs, the Camera view and the depth values. The depth values are captured in a previous pass. the depth values are sampled from that pixel and then calculated to get the world position. Its then used to get the projecton matrix of the previous position and converted to a normalized distribution (1 to –1). The velocity is calculated by subtracting the previous position from the current position and multiplied by blur strength and then flipped so the blur is trailing. Then multiply that by the distance, the closer the pixel the more blur needed. In my shader I then rule out any completely black pixels to keep the blur from streaking over the untextured background. I then add the color buffer to the velocity and loop that over, adding more velocity every time to get a blurring effect. And take that final and divide it by the number of samples and output.

References:

Websites:

serOleg

https://seroleg.itch.io/shaders-demo

(accessed: 08/12/2023)

Used for explanation on motion blur and math's, the motion blur hlsl file is mostly unchanged from this and other sources I found. My original code is the c++ files and my implementation and integration of the motion blur in to my project and the other parts of the project.

Jeremiah

https://www.3dgep.com/texturing-lighting-directx-11/#Lighting

(accessed: 02/12/2023)

paroj

https://paroj.github.io/gltut/Illumination/Tut11%20BlinnPhong%20Model.html

(accessed: 02/12/2023)

I used this to learn specular lighting, a lot of the hlsl file is simelar to the examples I found and the same applies here as the motion blur.

Outside of this its my work from the labs and the base coursework project provided

Assets:

https://opengameart.org/users/blindman67

(accessed: 27/11/2023)

I have included this in the submission but decided to go with the heightmap from the classes as it was the best out of the assets I could find. The mountain texture was created in paint.net. I created some alternative river maps with stable diffusion to attempt to smooth them out.