CS 330 Project Reflection

Justify

The 2D scene that I chose to turn into a 3D landscape was a picture with a monitor, laptop, pencil, ball, and battery sat upon a wooden desk. To narrow down the scene into realistic proportions I chose to depict only four of the objects, the laptop, the pencil, the ball, and the battery. After creating the desk, I first chose to implement the development of the pencil, this is due to its two very clear distinctive shapes.

Narrowing the beginning of the creation to a simple pencil that would help me develop a good foundation of how to create objects in a 3D space. I broke the pencil down into a cylinder for its stick, and a cone for its writing head. Trying to get them to seamlessly fit together did take a while to mentally figure out. In the scene I did take some artistic liberties and used fun textures to distinguish between the two pieces of the pencil.

The second object that I chose to depict was the laptop because it is essentially two long boxes. Instead of trying to completely match what my computer looked like I made it more of a generic laptop. To create the screen I used the texture of glass and the material metal to create a shiny surface. After fitting that to the black colored base the computer was complete.

The next two objects were very simple to make. I made the ball blue to match my 2D image and then matte shaded so as to not throw any specular light. The battery I made metal all around which closely shows what a battery actually looks like under its wrapping.

Explain Navigation and Functions

In order to create an accurate 3D scene moving the camera to check the orientations of the objects is essential. The way that I enabled the movement was through keyboard events and mouse movements. For the keyboard I had the general movement controlled by the keys: W,A,S,D. Where the ‘W’ key would move forward, the ‘A’ would move left, the ‘S’ would move backwards, and the ‘D’ would move the camera to the right. I also used the keyboard to change the view angle. To see the scene in an orthographic view, keys 1 through 3 could be used.

I then had two other functions to handle the camera movements that are enabled through the mouse. The function “Mouse\_Position\_Callback” checks where the mouse is in the beginning and then calculates the offset. The camera then moves when the “ProccessMouseMovement” function is called with the parameters of the x and y offsets. The other function that the mouse can control is the speed at which the camera moves in a direction, using the WASD keys. Scrolling up reduces the speed the camera moves and scrolling down on the mouse makes the camera slow. This is done with the function “Scroll\_Callback” and “ProcessMouseScroll”” in the camera.h file.

In the SceneManager file there are a total of five functions that I created to set up the scene. The first is the “LoadSceneTextures” method that finds the texture and assigns them a tag. The second function in that section is the “DefineObjectMaterials” method. These methods assign tags to textures and shadings. The “SetupSceneLights” is the second part to the shading and is the actual light in the scene. I created two different lights. I kept one closer to a white light and the other a more natural looking yellowish color. I wanted to depict an inside desk that was also close to a window. The fourth function was the “PrepareScene” method. This method calls the previous three. This method is also where we also load in the basic meshes. The plane mesh was already initiated to give us the flooring. Then we have a mesh for the cylinder, cone, box, and sphere. The last function is the “RenderScene” method. This is where the objects are actually created in the 3D space. Each object could have been separated into their own functions but I kept them in a running log using comments to separate them out.