Ron Bal

CS-330 final project reflection

6/18/2023

I wanted to choose some objects I had nearby that I thought would give me a variety of difficulty and shapes to utilize. I felt my desk was the perfect plane to have the objects on, especially since I could quickly look at it to properly design it within my 3D space. The pencil box allowed me to use a few different rectangles, the coffee cup gave me the opportunity to work with cylinders and torus’ although, I couldn’t properly render the torus so I utilized a sphere instead. The most challenging object I chose was a vase that geometrically composed of different triangles. While this was by far the most difficult shape for me to render, it gave me a lot of satisfaction when I was able to generate it.

I was able to render the sphere and the cylinder through their respective source codes and headers. Once implemented properly, I could adjust the size and location of each object.

The rest of my objects I modeled with vertices and indices. What helped me the most was to draw out the outer bounds of my plane (desk) on some graph paper so that I could properly place each of the objects within my scene. And with the same method, I was able to effectively decipher which indices should be used to generate my objects.

A user can navigate through my scene utilizing the WASD keys. This is done by initializing the camera class within the namespace function with vector coordinates, adding a float of the x and y axis, dividing them by the window width and height respectively, by 2. The key calls are defined within the UPRocessInput function to bind the different keyboard inputs to their respective outputs. Those being, W to move forward, S to move backward, A to move left, and D to move right. Classes are later created to hold the information for the mouse position, button, and scroll callbacks. The camera function is called within the URender function and given projection requirements.

I didn’t do anything to modularize and make the program more organized as I struggled a lot and spent most of my time troubleshooting. If I had more time, I would have separated out the shape generation into its own class or source file which would lead to overall cleaner looking code and not having to scroll through hundreds of lines of code trying to navigate to different classes and functions. Much like the Cylinder and Sphere classes we were given which could be called upon in the URender function, allowing it to be manipulated and textured dependent on the VAO and VBOS they are assigned to.