Darren Sheftic

CS330 – Computer Graphics & Visualization

Prof. Battersby

February 24, 2024

Project Reflection

When deciding the scene I would recreate I took into consideration the lack of experience I have with OpenGL and the complexities I would need to consider when reconstructing a scene as I was advised to keep my scene’s objects simple for the sake of coding. For this reason I chose my workspace area consisting of a desk, ring light lamp, my laptop, and a leftover tea cup that I would regularly have at my desk. Thanks to code provided from tutorial assignments and Professor Battersby I was able to recreate objects using meshes in an easy to digest process, though due to inexperience there was still some time consuming struggling. The textures were relatively easy to source, though because I had to showcase the inside of a cup using a texture as opposed to creating a mesh that showed the cup as a hollow space that proved to be a bit tricky, especially as I had to play with the image file so that it fit correctly over the inverted cone.

One of the objectives was to convert the camera to an orthographic view from a projected view. I was able to accomplish this by setting a Boolean to false that was active within the namespace, if while the window was open and the letters O or P were pressed it could toggle the perspective from UProcessInput to switch from Orthographic and Projected View. The rest of the code for this was in URender operating under a while-false statement.

// Create perspective projection (fov, aspect ratio, near plane, far plane)

if (!perspective)

{

// P is perspective (default)

projection = glm::perspective(glm::radians(gCamera.Zoom), (GLfloat)WINDOW\_WIDTH / (GLfloat)WINDOW\_HEIGHT, 0.1f, 100.0f);

}

else

// O is ortho

projection = glm::ortho(-5.0f, 5.0f, -5.0f, 5.0f, 0.1f, 100.0f);

The rest of the camera controls were implemented by using camera.h and understanding how to implement code into UInputProcess using the starter code from ProcessKeyboard in camera.h

While much of the starter code was already modularized and organized for me, through the milestones I did have to incorporate some custom functions, such as the switching from orthographic to projected view, but other custom code included exploring the scene using W, A, S, D to move the camera forward, left, backward, and right respectively, I also had to include custom code for Q, E to navigate the camera up and down. That portion of the code was kept close to other declared and relevant camera control functions and variables.