Final Project  
Jeremia Faust  
CS-330-T2650 Comp Graphic and Visualization 21EW2  
Doctor Loay Alnaji  
December 12, 2021

**Development Choices**

The 3D scene that I chose was four different tools that sit on a plane. The tools were a claw hammer, screwdriver, tape measure, and a mallet. All these tools contained at least two primitive shapes. The primitive shapes that were used were a pyramid, cylinder, box, and a plane. For the claw hammer, the mallet, and the screwdriver I used a cylinder for the handle with a wood texture. This made them look like the tools real handle. Originally, I was going to use more solid colors but when I learned how to use textures and realized how easy they were to use I decided to use textures for everything. This made the 3D scene look more realistic. For the claw hammer head I used a cylinder for the striker and a box for the middle part and two boxes for the claw. The texture is a dark metal and a light metal so when the light hits the head the light texture shows up. Otherwise, it would use the dark texture. The mallet head is a cylinder with a rubber texture. The screwdriver shaft is a cylinder, and the head is a pyramid to look like a Phillips head screwdriver. These both have a metal texture. The screwdriver has a rotation to make it look like it is screwing down a screw. The most complex object is the tape measure. It starts with a box that has a yellow texture. The switch is a box that uses the rubber texture. The clip uses two boxes that uses the metal texture. The tape part is a plane with a tape measure numbers texture that ends with a plane that uses a metal texture that acts like the end of a tape measure. These objects sit on a plane that uses a wood texture to make it look like a workbench. There is a wall that has a lower section with tiled tile texture. The tile texture was done by changing the texture coordinates from one to three. The best way to do this was using a variable so that I did not have to change it throughout the code. The upper part of the wall is a box that uses a pegboard with tools texture.

I use three different types of light sources, a directional light , point light and a spotlight. The directional light is a soft white light that lights the entire scene. I debated on using this light source but ended up using it because it really brought the scene to life. I use two-point lights as the main light sources. I used a white ambient with a yellow diffuse and an orange spectacular. I chose this to make the light look like a fluorescent light source. I also use a spotlight attached to the camera, so it highlights whatever the user is looking at.

**Navigation**

Navigating the world starts with the basic control that have become standard in the computer world. Using the code example

if (glfwGetKey(window, GLFW\_KEY\_W) == GLFW\_PRESS)

camera.ProcessKeyboard(FORWARD, deltaTime);

we were able to set up basic controls; forward, back, left, and right uses the w, s, a, d keys respectively. The q button is setup to move the camera straight up and the e button to move the camera straight down. The p key switched the camera from perspective to orthographic. this required two if else loops in the code. We had to declare a global bool to make this possible. The first one is in the main while loop where it checks the global Bool, to see what mode the camera is currently in. The second loop is in the process input loop. This is where you toggle between the two modes by changing the global bool. I added a timeout using glfwWaitEventsTimeout(18000.7); to prevent the toggle to continue if the user holds down the p button too long. The mouse controls the direction the camera is point at. This is done with the x and y offset. There is a constraint built in to prevent the camera to spin out of control.

if (Pitch > 89.0f)

Pitch = 89.0f;

if (Pitch < -89.0f)

Pitch = -89.0f;

The camera cannot go past ninety degrees either way. The scroll is set up to increase and decrease the speed of the camera. This is done by increasing or decreasing the movement speed variable and add it to the yoffset.

**Custom Functions**

My code is set up in a few different sections but mostly with in the main of the code. the first thing I did was set up the code for the boxes, the back wall, the pyramid, and the plane. The code for the cylinders was provided and is with its own class. Originally, I had set up each object to their own vertex array object or (VAO) and vertex buffer object or (VBO) this became very unwieldly fairly quickly. Luckily, I later learned that I did not need to do so. What I did was just assign each shape to their own VAO and VBO. The next section is setting up all the textures. I used a function provided to load in the textures. Originally, I had set up each individually but found that using the loadTextures function save a lot of time and space. The textures were then attaches to the shaders.

Now we move to the main while loop that runs the program. first, we load in the processInput fuction, render the background color. The next part is to set up the lighting for the scene. This part we use the Phong shading model to set up the ambient, diffuse, and specular lighting. Now we set up the world and camera transformations. At this point we are ready to load in objects and transform them to the locations and set up their sizes and attach a texture to them. Each object is labeled so if a change is needed it is easy to find. If I had more time, I could have modularized this project further by utilizing more functions and classes.