**Module Seven Final Project**

Ethan Anderson

CS-330: Computer Graphics and Visualization

Professor Wabara

April 18, 2024

**Justify development choices for your 3D scene**. **Think about why you chose your selected objects.**

The reason I chose the selected objects in my scene is because the whole scene is comprised of simple shapes such as cylinders, boxes, and planes. Any more complex geometry would be difficult to model using only the provided ShapeMeshes code and would most likely require loading in models from a third-party 3d modeling software. The materials and textures in the scene were also very straightforward and included metal, wood, and glass. I believe this was a good scene to display the Blinn-Phong lighting system, since the metal, wood, and glass materials interact with the light sources very nicely.

**Explain how a user can navigate your 3D scene**. **Explain how you set up to control the virtual camera for your 3D scene using different input devices.**

There are a few ways for users to navigate the 3D scene. Most of these movements were implemented using the glfwGetKey() and glfwGetMouseButton() functions. The first is basic forward, backward, left, and right movement using the W,A,S,D keys. These functions check for a button press, for example, GLFW\_KEY\_W == GLFW\_PRESS, which checks if the W key is pressed. Next, users can move the camera straight up and down using the Q and E keys. These two inputs are implemented the same as the W,A,S,D keys. The last set of keys are the O and P keys, which change the projection matrix of the camera to switch between orthographic and perspective views.

There is also a function called Mouse\_Position\_Callback, which checks for mouse movement and orients the camera accordingly. I added a Boolean which checks for a right mouse press and release, so that the user must hold down the right mouse button to move the camera. This is more intuitive in my opinion and works similarly to some 3d modeling software’s and game engines. If this project were for a video game, the former mouse controls would be more appropriate.

The final method of input is using the Mouse\_Scroll\_Callback() function, which checks for a mouse wheel scroll. When the mouse wheel is scrolled forward the speed of the camera increases and slows down when scrolled backward. This is important for navigating the scene quickly and is very helpful for making quick adjustments to the scene.

All of the input functionality interfaces with the Camera class which includes all of the logic to translate the camera in 3d space and update speed, sensitivity, and zoom.

**Explain the custom functions in your program that you are using to make your code more modular and organized**. **Ask yourself, what does the function you developed do and how is it reusable?**

In my opinion, the most important functions in this program were all the Render() functions which render each individual object in the scene. The way I went about organizing this is by keeping all the objects with the same materials and textures in their own render functions. For example, each knob object has a gold texture and metal material, so each instance of a knob would be included in the RenderKnobs() function. Having all these objects organized helped a lot when making translations and testing different textures and materials. Also, these functions can easily be reused to create similar objects with different scales or translations.