CS-330 Design Decisions Reflection

1. **Justify development choices for your 3D scene**. Think about why you chose your selected objects. Also consider how you were able to program for the required functionality.

The selected objects that I chose can be seen in SceneManager.cpp under the PrepareScene() heading and we see PlaneMesh, BoxMesh, CylinderMesh, and SphereMesh. The LoadCylinderMesh is no longer used and can be deleted; originally it was supposed to be the cables that hold up the shipping container, but they were too difficult to position so I removed them. The box mesh is used many times – for the main body of the shipping container, for the front of the container, and three instances of the box mesh for the container spreader.

The sphere is used for the skybox and the plane is used for the ground. Trees were also going to be added as planes, but I do not know how to apply opacity to only part of the object texture. The programming was simple as I looked at the project’s code and adjusted or inserted similar code.

1. **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.

The user can use WSAD to control the camera to move around the scene. W to move forward, S to move backward, A to move left, D to move right, Q to move up, and E to move down. The scrollbar can be used to increase or decrease the camera movement speed. Also, the mouse cursor can pivot the camera to look slightly left, right, down, and up. The way this works is in the ViewManager.cpp file under the ProcessKeyboardEvents() heading, such as GLFW\_KEY\_E which takes the camera y position and decrements it so that the camera moves down. Additionally, the user can press O to enter orthographic mode and P to enter perspective mode.

1. **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?

Functions exist in the code such as RenderScene and PrepareScene. Functions keep the code clean and separate from other functions and can be called as needed from the main code. If it wasn’t for functions, then we’d have to write hundreds of lines of code in MainCode.cpp just to run it all and that’d be a mess. It’d be like walking through a grocery store where all the products are put just anywhere and follow no specific organization.

Instead, all that the main function needs to do is call RenderScene() and it refreshes the scene as needed. When RenderScene is called, the function in SceneManager.cpp transforms and draws the basic 3D shapes such as the skybox sphere. From MainCode.cpp, this function is called in a while loop so that the scene is continuously rendered and re-rendered until the window is closed.