1

**Final Project Reflection**

Fellipe de Moraes

Southern New Hampshire University

CS-330-J8447 Comp Graphic and Visualization 24EW3

Professor Eisen Montalvo-Ruiz

2

The development of my project was based on realizing that OpenGL may be complex enough so that I should only stick to a environment that could be as simple as possible to create. My environment consisted of a table, an orange, a book, a coffee cup, and a coffee coaster. The requirement consisted of having 1 complex object and 3 simple objects in the scene. The book object was not simple as I later discovered in the development phase. The book consisted of different textures for the cover, the side cover, and the pages on 3 sides. The coffee cup was the other complex object. Complex objects are objects are made up of basic shapes like a pyramid, cylinder, sphere, plane, box, and a torus. The coffee cup consisted of a cylinder, a taped off cylinder for the bottom, and a torus for the handle. The book was not as simple since it consisted of one shape, a box rectangle. I initially thought I’d use a plane with a texture and place it to the box to represent the books cover, pages, or side. However, rotating the objects was a difficult task, so I came up with a simple solution: use boxes to represent a side of the book. This strategy worked out for my objective, but it didn’t render that good. I used a cylinder for the coffee coaster and a sphere for the orange. The texture implementation was straight forward, but I did have to play around with the image size to get the desired rendering on the screen. For the light source, I initially used a direction light source for the map, but I changed that to a point light source. This brought out more complexity to the map by revealing how the texture render in relation to the light position.

The project does implement a camery functionality that allows the user to move around the map. The program uses GLFW apis to determine if the keys W, E, D, Q, S, and A are pressed. If so, the user moves forward, down, right, up, back, and left respectively. These keys interact

3

with the API’s from the Camera class for this functionality to work. Another key functionality is the P key, where the camera projection is changed from perspective to ortho.

My project has several instances of code being modularized. One example is that it interacts with API’s outside of the project such as OpenGL API’s and GLFW. However, this is an assumed implementation. There are also classes that the project relies on such as the camera\_h class. This class allows users to move around the environment. There is stb\_image\_h, which allows my project to load images into the project and use as a texture to the objects. Finally, the project uses meshes.h class. This class is what allows my project to create simple and complex objects. Particularly camera\_h, this class is used at all times during the project, so its consistently meant to be reused. Likewise the main function contains all the modularized code in the project. The 2 functions that are reused consistently are: uProcessInput and URender, UProcessInput handles to user interaction with the keys and URender draws the object to the screen. Technically the object are created and re created at all times during the program execution.