### Southern New Hampshire University

### 7-1 Project

Chris Wills

### Instructor Ronald Bishop

### CS-330-R3322 Comp Graphic and Visualization 24EW3

2/25/2024

I ended up deciding to recreate a scene using some simple objects that had some variations in shapes. I chose my scene and objects carefully as to ensure I don’t put too much on myself for this challenge as well. I’m glad I went with more simple objects, as I encountered multiple difficulties throughout my scene recreation. Between objects bursting, running into memory issues, and finding ways to draw objects sequentially proved to be challenging for me, however I’m proud to have overcome those roadblocks in some form or another. I initially struggled in trying to draw out my objects separately, as my torus and sphere shapes were drawn as a single object in earlier milestones.

It took me longer than I would have like to realize you can set different objects to different VAOs and VBOs in order to have them drawn sequentially in the uRender function. I initially tried to create complete separate functions in attempting to create the sphere and torus, but I encountered issues with VAO and VBO assignment as I hadn’t realized I could have varied instances of those variables. Aside from my cylinder object, I eliminated the bursting problem I ran into for most of the course. Speaking of the cylinder, I tried several refactors in order to close the top along with closing the gaps present but was unsuccessful.

In getting the torus and sphere shapes created, I had to change the draw method from arrays to elements. In doing so, I’m not certain if the textures are loading through quite right. At first glance the coloring looks to be correct, but the silver part of the AirTag doesn’t seem to have the right metallic look from what the texture has. The other object textures appear to have loaded up correctly, even in the cylinder where the triangles are present.

Setting up the controls for the camera was relatively easy compared to the rest of the scene recreation. Keeping the controls set to something PC users often use, interacting with games for example, where W is forward, D is backward, A and D keys move side to side, Q and E keys change the camera elevation makes movement within the scene natural. Utilizing the mouse to rotate the camera is another natural addition for movement.

Computer scientists began organizing around the concept of "modularization," which means to break up and organize code based on the task it executes. This made code reusable and easier to manage and debug. Large projects often involve hundreds of programmers working on millions of lines of code. In this kind of environment, it is easy to lose track of what particular code does, or to produce code that must be rewritten elsewhere (Jackson, n.d.). As mentioned before, I attempted to compartmentalize the object mesh functions to achieve a modular design. In my final project, I still kept the uCreateMesh function, but assigned each objects’ information to a separate VAO, VBO, nVertices, and nIndices variable, enabling distinct calls from the uRender function for texture binding and draw purposes. By editing the total value of these variables allowed instances in the struct GLMesh functions, one can add more shapes into the scene and assign them to those additional variables. I could definitely go back and find variables used that are defined within their functions and globalize them to help remove redundant definitions.

**References**

Jackson, G. S. (n.d.). *The Advantages of Modularization in Programming*. Techwalla. https://www.techwalla.com/articles/the-advantages-of-modularization-in-programming