**7-1 Submit Your Project**

Matthew J. Rearick

Department of Computer Science

CS-330-H7036 Comp Graphic and Visualization 23EW2

Prof. Bishop, Ronald

12/10/2023

**7-1 Submit Your Project**

In developing my 3D scene, I made deliberate choices reflecting real-life objects I frequently encounter, aiming to replicate familiarity within the virtual environment. I utilized programming techniques to control the virtual camera and establish user interaction through various input devices to achieve the required functionality. The selection of objects within the 3D scene was based on their familiarity and relevance to real-world scenarios, facilitating an immersive user experience. Additionally, employing a Mesh base class enabled me to create diverse primitives with ease, fostering modularity and reusability within the codebase.

User navigation within the 3D scene was meticulously designed to offer an immersive and intuitive experience. Incorporating a multifaceted control scheme, I enabled users to traverse the virtual environment effortlessly. Utilizing "WASD" keys facilitated fluid movement through the scene, allowing users to explore different areas seamlessly. In addition to lateral movement controls, the "QE" keys were strategically implemented to regulate vertical motion, allowing users to ascend or descend within the environment effortlessly. The integration of mouse controls for orientation adjustment offered a natural and dynamic way for users to survey their surroundings, enhancing their sense of presence within the simulated space. The mouse wheel functionality was employed to modify movement speed to give users an adjustable pace during navigation. This feature empowered users to customize their experience, allowing leisurely exploration and swift traversal through the 3D scene.

Creating custom functions and methods, such as clamping U and V of textures on mesh objects, enhanced the complexity of textures and enabled the individual application of texture areas to primitive meshes. By connecting the UV coordinate to a sub-region of a texture, I created more compact and complex textures for my complex meshes, rather than needing to utilize multiple textures for individual primitive meshes of the complex mesh. The translateMeshPreVAO() method significantly contributed to creating complex meshes and was required to use UV clamping efficiently. This method allowed me to transform a primitive mesh's vertex buffer before sending it to the GPU, allowing me to transform multiple primitives and combine their vertex buffers into a single vertex buffer to be used by the complex mesh. This removed the need to manage several individual primitive meshes when doing later transformations. They were now all a single complex mesh and would be moved together as a single mesh. I then used a single texture for the single vertex buffer by initializing the primitives with the sub-regions of the texture while creating the vertex buffer.

Several custom functions, such as VertexMode ENUM for mesh building and rendering modes, elevated the code's modularity. This allowed quick switching between mesh attributes for testing and final rendering, promoting efficiency during development. Additionally, the CubeLightMesh class optimized the creation of point lights by reusing the CubeMesh structure and incorporating lighting variables. This streamlined generating point lights for the scene while providing visual representations of light sources. I could then add the cube lights to my scene meshes to render the cube, the light meshes to add their light to the scene, or both to render the cube and account for their light.

As the CS-330 course progressed, iterative refinement of classes led to a somewhat interconnected code structure. Acknowledging this, I would restructure the Mesh and Shader Classes if I were to start over on this project again. Specifically, I would aim to dynamically add attributes during vertex buffer creation, eliminating the need for separate code segments based on VertexMode. The recent implementation of this streamlined approach for SphereMesh and TorusMesh showcased significant code reduction and enhanced adaptability.

The development process of the 3D scene involved intentional object selection, user-centric navigation controls, and strategic code structuring. Reflecting on the iterative nature of code evolution, I've identified areas for fundamental restructuring, paving the way for a more cohesive and adaptable codebase. This reflective journey underscores the significance of user experience in 3D environments and emphasizes the value of continuously refining coding practices for enhanced efficiency and maintainability.