Pravishna Nand

CS 330

8/18/2024

7-1 Final Project Submission

**Reflection: Development Choices**

In designing my 3D scene, I selected objects that not only served to meet the technical requirements but also created a coherent and visually engaging environment. I opted for simple geometric shapes such as cylinders, toruses, and planes to construct objects like mugs and books, which are familiar and easily recognizable to the user. These choices allowed me to focus on the core aspects of 3D rendering and lighting without the complexity of more intricate models, making the scene manageable and efficient to develop. Furthermore, these objects were selected because they demonstrate key concepts such as scaling, rotation, and positioning, which are fundamental to understanding 3D space. I ensured the functionality required by the project—such as accurate object placement, realistic shading, and smooth camera control—was implemented through careful programming and testing. This approach provided a balance between technical challenge and the ability to produce a polished final product.

**Reflection: Navigation**

Navigation within the 3D scene was designed to be intuitive, leveraging standard input devices such as the keyboard and mouse for camera movement and scene interaction. I implemented a virtual camera that can be controlled via WASD keys for movement, with the mouse used to adjust the camera's orientation, allowing us to explore the scene from different angles. This setup mimics first-person navigation found in many 3D applications and games, providing a familiar experience. The camera was programmed to move smoothly and responsively, ensuring that navigation feels natural and immersive. Additionally, I incorporated constraints to prevent the camera from moving through objects, maintaining the scene's realism and preventing any disorienting experiences. This approach to camera control ensures that users can fully appreciate the 3D scene while maintaining a sense of spatial awareness.

**Reflection: Custom Functions**

To enhance the modularity and organization of my code, I developed several custom functions that handle repetitive tasks such as setting transformations, applying shaders, and drawing meshes. For instance, the SetTransformations() function standardizes how objects are scaled, rotated, and positioned within the scene, making it easy to apply consistent transformations across different objects with minimal code duplication. This function is highly reusable, as it can be called for any object within the scene, simply by passing different parameters. Similarly, the SetShaderColor() and SetShaderMaterial() functions abstract away the complexities of shader management, allowing me to apply different visual effects to objects with a single line of code. By encapsulating these operations within functions, I not only improved the readability of the code but also made it easier to manage and update, as any changes to these processes only need to be made in one place. This modular approach ensures that the codebase remains clean, maintainable, and scalable as the project evolves.