Andres Trujillo

02/23/2025

SNHU CS330

Professor Omar Aaziz

**Design Decisions**

In developing my 3D scene, I selected objects from my workspace that I realistically use every day and that I find within my environment. The scene includes a desk, a MacBook Air, a Sony alarm clock, a set of AirPods, and a dinosaur. These objects were chosen to create a balance between real-world items and a rubber duck tool to speak aloud my thoughts if I am having an issue. The desk serves as a foundational object that grounds the scene, while the MacBook Air and Sony alarm clock add to the realism of a workspace. The AirPods add a modern touch, and the dinosaur provides a unique, playful contrast to the otherwise standard setting of a collegiate student’s desktop.

**Navigation and Camera Control**

Users can navigate the 3D scene using a combination of keyboard and mouse inputs. The camera setup enables movement along multiple axes, allowing the user to explore the scene from different angles. Keyboard controls handle forward, backward, and lateral movements, while the mouse enables rotational adjustments for a more natural navigation experience. Additionally, the camera’s position is updated in real-time based on user input, ensuring fluid motion throughout the scene.

To implement camera control, I utilized a perspective projection matrix to simulate a real-world view. The camera is positioned with a view matrix that dynamically updates based on input, allowing for intuitive movement and rotation. This design decision ensures that the user can fully interact with the environment without limitations.

**Modular Functions for Organization**

To enhance code organization and reusability, I used several custom functions. These functions included transformation setters, texture setters, and custom modular object drawing functions. For example, the SetTransformations function centralizes scaling, rotation, and positioning logic, reducing redundant code across multiple objects. This makes it easier to adjust transformations without affecting other parts of the program.

Another essential function is SetShaderTexture, which applies textures to different objects based on predefined parameters. This modular approach ensures that texture management remains consistent throughout the scene. Additionally, object-specific functions such as DrawAirpods and DrawDinosaur allow for the structured placement of each object without cluttering the main rendering logic.

By modularizing my drawing functions, I achieved a more maintainable and scalable codebase. Future additions to the scene can be implemented seamlessly by following the existing function structure, ensuring efficiency and readability in the development process.

To ensure smooth functionality, I implemented transformation functions that handle scaling, rotation, and positioning. These functions allow for efficient modifications without having to rewrite transformation logic for each object. The scene follows structured programming principles, making it easier to maintain and extendable and scalable codebase. Future additions to the scene can be implemented seamlessly by following the existing function structure, ensuring efficiency and readability in the development process.