George Yockachonis

2/21/25

CS-330

Professor Rodriguez

The development of the 3D arcade scene was guided by a combination of realism, functionality, and modularity. This document outlines the design choices made in terms of object selection, scene navigation, camera control, and code modularity. Each decision aimed to enhance the user experience while ensuring efficiency in rendering and interaction.

The primary objective of the 3D scene was to replicate a classic arcade environment with an interactive arcade machine. The following objects were included:

* **Arcade Machine:** The arcade machine was constructed using a combination of box meshes for the main structure and control panel. The screen was given a CRT overlay texture to simulate an old-school display.
* **Joystick and Buttons:** The joystick consists of a slim cylinder (base) and a sphere (top) to resemble a traditional arcade joystick. The buttons were created using small cylinder meshes and arranged in a realistic layout.
* **Side Panels with Custom Textures:** To enhance the visual appeal, the side panels were textured with a Pac-Man-themed design, adding authenticity to the arcade experience.
* **Lighting Effects:** The scene included three primary light sources: a warm overhead indoor light, a cool blue arcade screen glow, and an ambient light source to ensure visibility and depth.
* **Background and Environment:** A black panel was added to the back of the arcade machine to hide unnecessary details, and a textured floor and wall were included to create a complete arcade setting.

Users can navigate the 3D scene using a virtual camera system controlled through keyboard and mouse input. The navigation system was implemented as follows:

* **Keyboard Controls:** The user can move forward, backward, left, and right using the W, A, S, and D keys, similar to first-person navigation.
* **Mouse Controls:** The camera's rotation is controlled by mouse movement, allowing users to look around freely.
* **Perspective Projection:** The camera uses a perspective projection matrix, ensuring a realistic depth effect while maintaining an immersive experience.

This implementation allows smooth exploration of the arcade scene, ensuring that users can examine details from various angles.

To maintain an organized and reusable codebase, several custom functions were developed for handling transformations, textures, and materials.

Transformations: SetTransformations()

This function centralizes the object transformations, including scaling, rotation, and translation. It ensures that every object in the scene undergoes consistent matrix transformations using the GLM library.

Color Handling: SetShaderColor()

To assign flat colors to objects that do not require textures, this function sets the RGB values dynamically in the shader. This approach allows easy customization without modifying the main rendering code.

Texture Management: SetShaderTexture()

Instead of manually binding textures each time, this function assigns the correct texture based on a tag system. It finds the appropriate texture slot and passes the information to the shader, improving efficiency and organization.

Lighting Setup: setupLights()

This function initializes and configures the scene lighting. It defines positions, colors, and intensity values for multiple light sources, ensuring a consistent lighting model across the scene.

The 3D arcade scene was designed with careful consideration of object selection, user navigation, and code modularity. The combination of structured transformations, dynamic textures, and interactive lighting effects contributed to an immersive and functional arcade simulation. By implementing a modular approach, the scene can be expanded and modified with minimal effort, allowing for future enhancements and refinements.