Mikayla-Joy Botha

CS 330 Computer Graphics and Visualization

Reflection

In the project scenario, I worked as a C++ and OpenGL 3D graphics developer for a company named Triangle and Cube Studios, which designs 3D worlds for various clients. As an employee, I was assigned to one of these company projects broken down into five milestones. I became responsible for creating code in OpenGL to create objects, apply textures, apply lighting, render, and control virtual environments relative to a virtual camera. For this project, I made a realistic and interactive 3D scene with texture, lighting, placement of objects, and camera movement features to replicate a 3D version of a 2D image. The textures I selected were based on how realistic they could depict the real world with royalty-free properties. Objects used to recreate the scene included a table, computer monitor, keyboard, books, mouse, and a pencil holder with pencils. Each of these items was subjected to the application of different textures to see effects on the top profile. Ambient lighting was set for complete illumination with a point light set as its primary light source, while the introduction of soft-colored light aids dimming harsh shadows and improving visibility. The objects were then placed using coordinates X, Y, and Z based on a reference picture, which provided a guide for object positions and sizes.

Finally, the camera control functions were included for freestyle movement throughout the scene, with WASD for movement, QE for vertical movement, and mouse pitch and yaw controls. The mouse scroll wheel dynamically controls the movement speed, and the view can be changed between orthographic and perspective, with classic views for preset alternative angles. Further, the code was modular so that the efficiency and scalability were further enhanced by functions such as LoadSceneTextures(), SetupSceneLights(), and SetTransformations() for textures, lights, and object transformations. All of these elements made the project more manageable overall. The modularity of the code, with functions such as SetShaderMaterial() and SetShaderTexture() for material and texture changes and ProcessKeyboardEvents() and Mouse\_Position\_Callback() helped to move the camera smoothly, making it easy to apply and reuse. The user can explore the 3D scene using a combination of keyboard and mouse controls. The WASD keys move the camera forward, backward, left, and right, while the QE keys allow vertical movement. The user can explore the 3D scene using a combination of keyboard and mouse controls. The WASD keys move the camera forward, backward, left, and right, while the QE keys allow vertical movement. The user can also change viewpoints by switching between perspective and orthographic modes, with the P key enabling a realistic perspective mode and the O key enabling a 2D-like orthographic mode. There are also a few other preset views: the 1 key will allow the front view, the 2 key will help the side view, the 3 key will enable the top view, and the four key will allow the perspective view.