December 11, 2024

Jeanelyn Roman

CS-330 Comp Graphic and Visualization

Dr. Sherin Aly

Project 2: Reflection

**Justify development choices for your 3D scene**

I used items like a monitor, notebooks, a desk organizer with pens, a coffee mug, and a textured desk surface in my 3D scenario to create a realistic workspace. These choices create a familiar and functional environment while meeting project requirements. By using low-polygon models, I ensured the scene is both efficient and detailed enough to be visually appealing. Realistic textures were used to improve the overall quality, such as a metallic appearance for the monitor and a wood finish for the desk. To keep the code organized and reusable, I created custom functions for rendering each object, making the development process efficient and adaptable for future projects. This approach balances creativity, functionality, and technical guidelines.

**Explain how a user can navigate your 3D scene**

Users can explore my 3D scene using a fixed camera perspective that is intended to effectively and clearly display the workspace setup. Due to development limitations, the camera's position and view angle are fixed; yet, it has been set up with an appropriate field of view and orientation to catch every object in the picture. Users may move around the scene with a mouse or keyboard, zooming in, rotating the perspective, or panning across the scene for a more engaging experience if dynamic camera control was enabled

**Explain the custom functions in your program that you are using to make your code more modular and organized.**

In my program, I incorporated several custom functions to promote modularity, reusability, and maintainability. Each function has a specific role, helping to streamline development and maintain an organized codebase. I utilized render functions to manage the transformations, materials, textures, and drawing of individual objects, making it simple to add, modify, or remove objects without impacting other parts of the program. In order to ensure consistency and make modifications easier across all objects, functions like SetTransformations, SetShaderMaterial, SetShaderColor, SetShaderTexture, and SetTextureUVScale concentrate on important operations like scaling, rotation, material application, and texture mapping. To ensure that textures and materials are ready for rendering, centralized functions like LoadSceneTextures and DefineObjectMaterials also take care of their pre-preparation. To improve visual quality, light sources can be easily adjusted with the SetupSceneLights function, which is specialized to scene lighting configuration. This method guarantees that modifications, such changing the material of an object or adjusting lighting settings, can be applied effectively without causing structural disruptions.