The 3D scene for this project is based on a photo of a laptop setup. This includes a lamp, a laptop, a coffee mug, and book. Each object was modeled using low-polygon shapes, such as boxes, cylinders, spheres, cones, and torus meshes. The lamp base and arm are represented using cylinders, while the lamp head is a scaled sphere. The laptop body and screen are constructed from boxes, with the screen angled backward to replicate its realistic tilt.

Textures were applied to match the real-world materials in the image. For instance, brushed gold and metal textures were used for the lamp, a wood grain texture was mapped onto the desk, and the notebook was styled with a tiled pattern. The laptop screen uses a sleek, high-resolution image to resemble a reflective display.

Navigation and Input Controls

The project includes full support for camera navigation. Users can move through the scene using the W, A, S, D keys for directional movement, with Q and E providing vertical motion. The mouse is used to control pitch and yaw for orientation, while the scroll wheel adjusts the zoom level or movement speed. Additionally, users can toggle between perspective and orthographic projection modes using the P and O keys. Camera movement and orientation were implemented following modern OpenGL practices (de Vries, n.d.).

Custom Functions and Modularity

The SceneManager class contains custom functions like SetTransformations, SetShaderTexture, and RenderScene, which let me build and place objects efficiently. Texture loading is handled using the STB image library through CreateGLTexture, which supports several objects using identifiers. The ViewManager and ShaderManager classes separate the concerns for camera and shader handling. The scene rendering techniques align with those outlined in LearnOpenGL tutorials (de Vries, n.d.).

Reflection and Future Improvements

This project captures the essence of a realistic desktop layout while meeting low-poly modeling and texturing requirements. The implementation of interactive camera controls and view projection toggling improves usability and immersion. Future improvements could include dynamic lighting enhancements using Phong shading model (Phong, 1975) or the addition of animated elements to bring more life to the scene.

References

Phong, B. T. (1975). Illumination for computer generated pictures. \*Communications of the ACM\*, 18(6), 311–317. https://doi.org/10.1145/360825.360839

de Vries, J. (n.d.). Learn OpenGL. https://learnopengl.com/