Reflection

Reflecting on the development of my 3D scene, I can see a culmination of purposeful choices and careful planning. I chose to model a monitor within a virtual environment because it is an object that's both ubiquitous and technically interesting. Also, because I am mostly around my desk for work so it makes sense to create somewhat of a replica of my everyday life. Its surface properties, such as the reflective screen and matte frame, allowed me to explore and demonstrate the effects of various material and lighting setups.

I focused on creating a realistic portrayal, which involved programming functionality to simulate different lighting scenarios. The ambient and diffuse properties were adjusted to mimic an indoor setting, ensuring that the monitor's screen wasn't overly reflective, which could be distracting and unnatural.

To navigate this scene, I utilized the familiar WASD keys to move the camera forward, backward, left, and right along the X and Z axes. I programmed the QE keys to control upward and downward movement along the Y-axis. This standard control scheme allows for intuitive and efficient exploration of the 3D space.

The orbit radius of the camera was carefully calibrated to encompass the entirety of the objects. With each object added, I adjusted this radius and the camera's position, ensuring a clear view from multiple angles and highlighting the interplay of light and shadow across their surfaces.

I've also implemented nuanced camera controls that allow for subtle adjustments in the viewing angle without changing the camera's position. The mouse cursor was set up to change the pitch and yaw, giving the user the feel of a first-person perspective. The mouse scroll, an input device easily overlooked in its potential, was utilized to refine the speed of movement, offering a controlled and customizable navigation experience.

To enhance the user experience further, I integrated the functionality to switch between perspective and orthographic views. At the tap of a key, the user can shift from a 3D perspective, which offers a sense of depth and realism, to an orthographic view that presents a more structural and dimensional perspective of the objects. This dual view functionality was achieved by toggling between perspective and orthographic projection matrices while maintaining the camera's orientation, allowing for diverse visual analysis.

In my code, I've emphasized modularity by encapsulating functionality within custom functions. For instance, `SetTransformations()` is a reusable function that applies position, rotation, and scale transformations to any object, abstracting the complexities of matrix math. This not only streamlines my code, making it more readable and maintainable, but also facilitates the reuse of code for any new objects introduced to the scene.

Through these efforts, I've crafted a program that not only meets the specified requirements but also offers a platform for further experimentation and development. The flexibility and modularity of the code mean that new features or objects can be added with relative ease, making it an adaptable foundation for future projects.