The selection of objects and the overall scene setup were driven by the desire to create a visually appealing and interactive environment that demonstrates various OpenGL features. The scene comprises multiple geometric shapes, including cubes, spheres, and cylinders, strategically placed to create a complex structure resembling a staircase with railings and lighting elements. This choice was motivated by the opportunity to showcase different texturing and lighting effects on various surfaces, as well as to present a challenge for navigation and interaction within a 3D space.

Textures were carefully selected and applied to different objects to create a more immersive and detailed environment. For example, wood and carpet textures were used to simulate realistic surfaces for the staircase and floor, respectively. This choice adds to the scene's realism and visual richness, making it more engaging for the user.

User navigation within the 3D scene is facilitated through keyboard and mouse input, controlling a virtual camera. The camera's movement is managed by the processInput function, allowing the user to move forward, backward, left, right, up, and down within the scene using the W, A, S, D, Q, and E keys, respectively. This setup provides a familiar and intuitive control scheme for users, similar to standard video game navigation controls.

Mouse input is utilized to control the camera's orientation. Moving the mouse changes the camera's direction of view, enabling the user to look around the scene freely. This interaction is managed by the mouse\_callback function, which adjusts the camera's yaw and pitch based on the mouse movement. Additionally, the scroll\_callback function allows users to zoom in and out by scrolling the mouse wheel, offering a convenient method to adjust the view's distance.

Several custom functions were developed to maintain code modularity and organization. These include:

framebuffer\_size\_callback: This function adjusts the viewport size whenever the window size changes, ensuring that the scene scales correctly on different screen sizes and aspect ratios.

mouse\_callback and scroll\_callback: These functions handle mouse movements and scrolling, updating the camera's orientation and zoom level, respectively. They are essential for creating a dynamic and interactive camera system.

loadTexture: A utility function to load textures from files, enhancing the scene's visual details. It supports different texture formats and applies mipmapping for improved texture quality at various distances.

These custom functions contribute significantly to the code's organization, making it easier to manage and extend. Each function encapsulates specific functionality, following the principle of single responsibility and making the codebase more maintainable and reusable.

In conclusion, the development of this 3D scene involved thoughtful choices regarding object selection, lighting, and texturing to create an engaging and realistic environment. User navigation was carefully designed to be intuitive and flexible, using keyboard and mouse inputs to control a virtual camera. Custom functions played a crucial role in ensuring code modularity and organization, facilitating easier maintenance and potential future enhancements. This reflection not only justifies the development decisions made but also highlights the importance of user interaction and code structure in creating immersive 3D scenes.