1. Introduction

Computer graphics is one of the most popular technologies in recent years and is widely used in many fields. The language of graphics, as a means of communication and communication, allows for a more vivid presentation of three-dimensional (3D) objects [1]. Therefore, this explanation will show the strong functions of computer graphics by introducing a 3D real-life scene based on OpenGL.

2. Implementation

In this explanation, the implementation of the 3D scene is divided into two parts: design and features and instructions.

2.1. Design and features

To meet the requirements of a 3D scene, the creator in this project creates an estate surrounded by roads (shown in Figure 1). Overall, the living scene can be divided into three parts. The first part is the surrounding environment. The second one is the construction of the estate. And the last one is the construction of the roads outside the estate. Apart from the surrounding environment, all the structures are created using hierarchical modelling, which means that all the complex objects are composed of basic elements.

(a) (b) (c)

Figure 1. The estate and roads

2.1.1. The surrounding environment

The idea behind the environment's design is not complicated; it is essentially a box-like object (called "skybox"). The program uses the "GL_QUADS" function and configures the coordinates of each vertex to build the basic cube frame (which is large enough to hold a large number of objects). Next, the mapping technique is employed in this cube frame. The program uses the "glBindTexture" function to bind each cube vertex to each image vertice (also named "texture") so that the image can be mapped to the surface of the cube. Then, the program needs to call the "gluLookAt" function to zoom the camera's position inside the cube to see the realistic surrounding environment in Figure 1. (a).

2.1.2. The construction of the estate

The houses, whose style is similar to European architecture, are the main part of the estate. First of all, all seven houses are constructed using a combination of cubes and cones. Instead of using the "glutSolidCube" function in the freeglut library, the project uses the "GL_QUADS" function above to construct the cubes by traversing the vertices. Applying the "GL_QUADS" function is because the "glutSolidCube" function cannot be used for mapping. The same idea is employed on the topmost square cone of the roof. To simplify the series of operations for creating 3D graphics,

constructing the cubes and the cones using "GL_QUADS" is designed as a separate function. Then, to add further realism to the scene, the pine trees in the middle of the estate are added to show the lighting and material effects using the functions called "glMaterialfv" and "glLightfv" functions (shown in Figure 1. (b) and (c)). In addition, mouse interaction animations for 'open' and 'close' are employed at the estate's gate. The purpose of this animation is to give the user a strong sense of immersion. The gates are rotated using the "glRotatef" function in combination with the "onTimer" and "mouse_input" methods. The program modifies the position variable (called "gate_rotation_angle") in the "OnTimer" method and passes the final variable value to "glRotatef" function as a parameter. Moreover, The "mouse_input" method can call "onTimer" to demonstrate the animation effect.

2.1.3. The construction of the roads

As shown in Figure 1., three tarmac roads are added to the estate's perimeter to reflect its modern feature. The surface of the road, which is designed with texture mapping technology, is just a basic cube constructed by the "GL_QUADS" function. Only the height of the cube is set very small. Two cars are on the left and right sides of the road in total. The program uses "glRotatef" to achieve the effect of turning left and right. Through keyboard interaction, these cars can automatically move. The trees next to the road are also employed with lighting and material. Different parts of the tree can be seen in different shades of green. In addition, a "for" loop is used in the tree drawing code to keep the complexity of codes as simple as possible.

2.2. Instructions

2.2.1. The keyboard interactions

First, by pressing the "w" or "W" key, the camera moves forward so that the details of the objects can be seen (shown in Figure 2.).

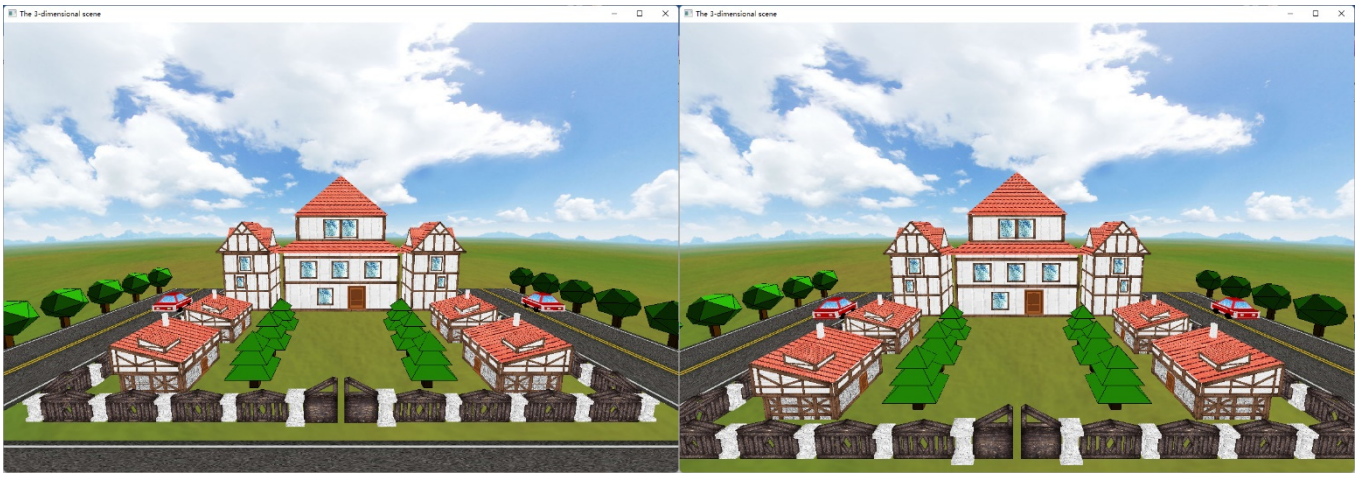

Figure 2. Zoom forward

Second, by pressing the "s" or "S" key, the camera moves backward, and more objects can be covered (shown in Figure 3.).
Third, by pressing the "a" or "A" key, the camera rotates left at a fixed angle, and the user can see the details of the object on the left (shown in Figure 4.).
Fourth, by pressing the "d" or "D" key, the camera rotates right at a fixed angle, and users can see the details of the object on the right (shown in Figure 5.).
Fifth, by pressing the "q" or "Q" key, the camera is translated to the left, and objects on the left can become the center of the picture.

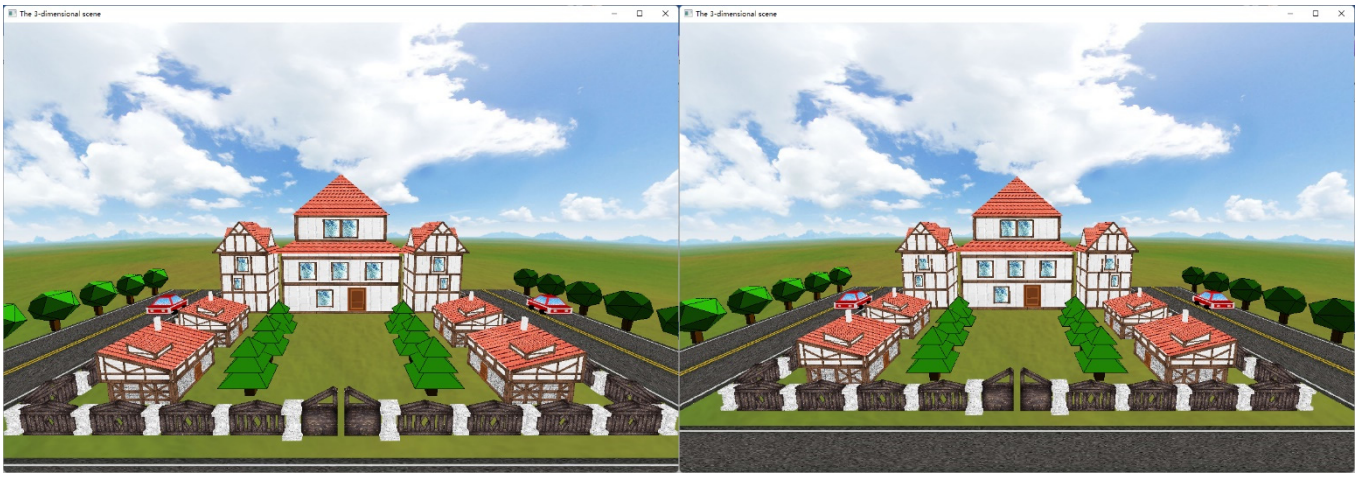

Figure 3. Zoom backward

Figure 4. Rotate left

Figure 5. Rotate right

Sixth, by pressing the "e" or "E" key, the camera is translated to the left and objects on the left become the centre of the picture.
Seventh, by pressing the "m" or "M" key, the camera is translated upwards.
Eighth, by pressing the "n" or "N" key, the camera is translated down.
Ninth, by pressing the "r" or "R" key, the camera is returned to its original starting position.
Tenth, by pressing the "t" or "T" key, the program can be exited.
At last, by pressing the "k" or "K" key, the two cars on the road can move and stop automatically (shown in Figure 6.).

2.2.2. The mouse interactions

When the user clicks the left mouse button, the estate gates are opened backward (shown in Figure 7. (a)). When clicking the right mouse button, the gate rotates to its original position (shown in Figure 7. (b)).

Figure 6. Cars move and stop

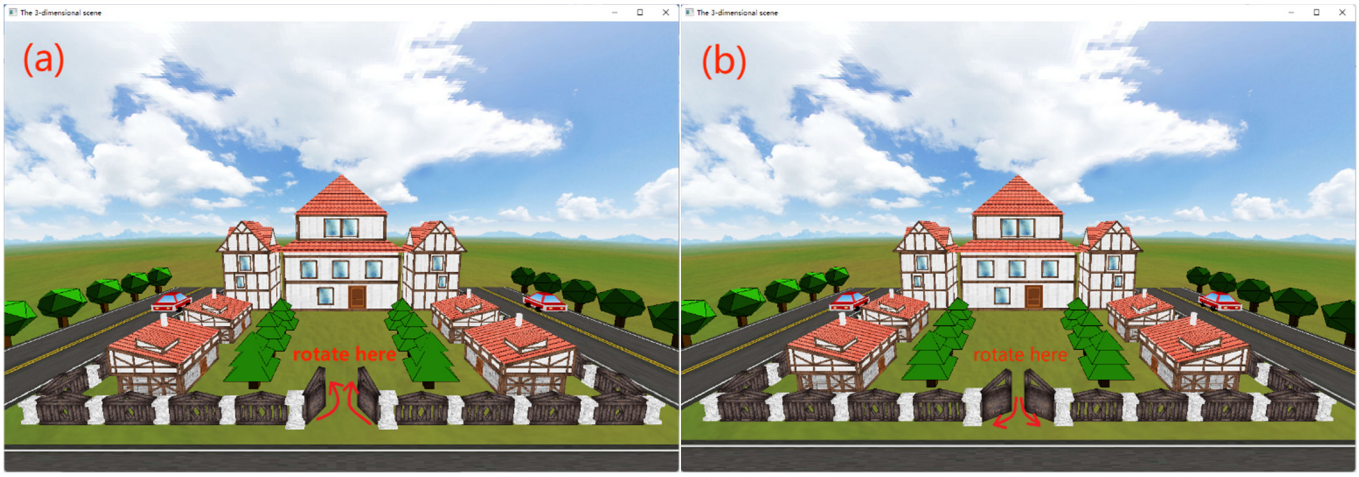

Figure 7. The rotating gate

3. Evaluation

Among all the elements, the construction of houses is one of the most creative designs. While highly vivid, the design of houses is relatively easy to implement. Visually, the project is well-finished and realistic in showing a living scene. However, this project still has much space for improvement, such as the lengthy codes. Although some basic graphics are extracted into an independent method, many methods calls still result in wasting computer resources. Additionally, this project's lighting and the material system do not work for objects with textures, so the full effect is not perfect.

4. Conclusion

In conclusion, this explanation is based on the OpenGL library to implement a beautiful 3D real-life scene. The methods of drawing 3D objects, such as geometry creation, hierarchical modelling, transformations, viewing, lighting and materials, texture mapping, animation and interactions, are applied to the project. Apart from these advantages, this project has not addressed issues such as lighting and material failure on objects and code redundancy.

5. References

[1] T. Long, 'Reconstructing the New Significance of Combining Graphic Language —— Hand-drawn Graphic Language with Computer Graphics Language', in 2021 International Conference on Computer Technology and Media Convergence Design (CTMCD), Sanya, China, Apr. 2021, pp. 1–5. doi: 10.1109/CTMCD53128.2021.00008.