CPT205 Computer Graphics
Assessment2—3D Modelling Project Report

Xinling Du
2363292
Information and Computing Science

November 2025

1. Introduction

With the development of computer graphics, many industries and fields require the construction of 3D scenes, such as urban planning, game development, and science fiction films. This project utilizes OpenGL to implement an exquisite interactive 3D futuristic city, employing all the graphical techniques required by the assignments. The project integrates multiple futuristic elements, including static objects as well as dynamic objects controlled by timers and dynamic functions. Additionally, through the use of texture mapping systems and lighting-material systems, the scene's futuristic and realistic feel is enhanced. Users can also interact with the city via keyboard inputs, allowing them to immerse themselves in every detail of the city.

1.1 Scene Composition

- **Static Objects:** Ground with texture mapping, Various futuristic buildings, Aerial tracks, stations, road systems, Street lamps, decorative light strips, holographic billboards
- **Dynamic Objects:** Maglev trains, Wheeled robots, Quadrotor drones , Hovering vehicles, Holographic ads (scanning animation)

2. Graphics Techniques Implementation

This project is implemented using OpenGL (Freeglut) and C++, and applies the following core graphics technologies (including all the technologies required by the assignments).

2.1 Geometric Modeling

- Basic Geometric Primitives:
  - Used `drawBox()` function to create building bodies, roads, rails, and other cubic structures

- Employed `glutSolidSphere()` to create spherical components like robot heads, joints, and vehicle lights. Utilized `gluCylinder()` to construct cylindrical structures such as robot wheels, mechanical arms, and lamp posts. Used `glutSolidCone()` to create conical tree crowns for futuristic trees.

- Complex Geometry Combinations:
  - Trains feature capsule design: cylindrical body + hemispherical ends
  - Drones consist of cubic body, cylindrical supports, and flat propeller blades
  - Building clusters are formed through combinations of cubes to create different architectural styles

2.2 Hierarchical Modeling

The project widely used hierarchical modeling techniques to manage complex object structures:

```
Robot Assembly
  - Wheel Chassis System
  - Spherical Body
  - Head (with Facial Features)
  - Mechanical Arm System
      - Shoulder Joint
      - Upper Arm
      - Elbow Joint
      - Forearm
```

Listing 1: Robot Hierarchy Structure

```
Drone Assembly
  - Cubic Body
  - Cross Support System
      - 45° Support Arm
      - -45° Support Arm
      - 135° Support Arm
      - -135° Support Arm
  - Vertical Support Columns (4)
  - Propeller System (4 groups)
      - Main Propeller Blade
      - Cross Propeller Blade
```

Listing 2: Drone Hierarchy Structure

```
Train Set
  - Head Car
      - Main Cylinder
      - Front Hemisphere
      - Windows
      - Levitation Halo
  - Middle Car
  - Tail Car
```

Listing 3: Train Hierarchy Structure

```
Building Unit
  - Main Structure
  - Glass Curtain Wall
  - Facade Lines
  - Roof Decorations
```

Listing 4: Building Hierarchy Structure

2.4 Transformations, View, and Projection

- Use glTranslatef, glRotatef, glScalef for model transformations
- Use gluLookAt for first-person camera
- Perspective projection: gluPerspective(50, aspect, 1, 5000)

2.3 Lighting and Material System

- Multiple light sources:
  - Main directional light simulating sunlight
  - Blue ambient light creating atmosphere
  - Building neon lights
  - Train-following light
  - Independent drone light sources (each drone has unique light)
  - Robot self-illumination
- Material applications:
  - Building Glass: High transparency + specular reflection
  - Metal Surfaces: High shininess + strong highlights
  - Emissive Objects: Emission material properties
  - Plastic Components: Moderate ambient and diffuse lighting
  - Drone Body: Silver-white metallic appearance
  - Propeller Blades: Dark matte material

2.5 Texture Mapping

Ground surface uses texture mapping. Techniques including:

- Custom BMP file parsing
- Convert 24-bit BGR to RGB
- Texture repetition (GL_REPEAT): Ground size 2000×2000, texture repeated 20 times for realism
- Linear filtering (GL_LINEAR)
- Fallback gray texture if loading fails

2.6 Animation System

Time-based incremental updates ensure smooth animations. Timer-based animations (glutTimerFunc) include:

- Trains: smooth movement along tracks
- Drones: floating + rotating propellers
- Robots: path movement + body tilting
- Vehicles: continuous movement, each with delay
- Holographic ads: dynamic scanning effect

3. Instructions(README)

3.1 System Requirements

- Operating System: Windows
- Dependencies: freeglut static library
- Texture File: groundTexture.bmp (place in same directory as executable)

3.2 Keyboard Interaction Control

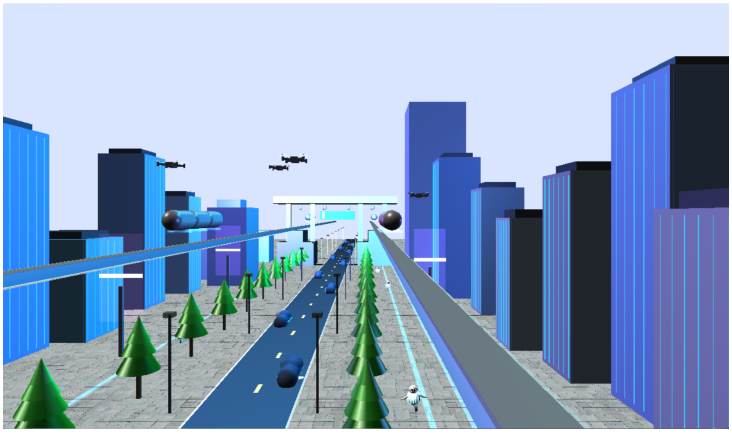
Users can use the keyboard to adjust the viewing angle and camera position to conduct first-person exploration of the city.

Key	Function
W	Move camera forward
S	Move camera backward
A	Move camera left
D	Move camera right
Space	Move camera up
Ctrl	Move camera down
R	Reset camera view
ESC	Exit program
↑	Increase pitch (look up)
↓	Decrease pitch (look down)
←	Decrease yaw (turn left)
→	Increase yaw (turn right)

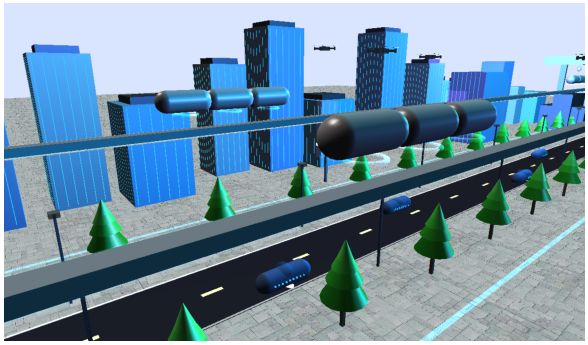
Table 1: Keyboard controls for camera and view adjustment

4. Program Screenshots

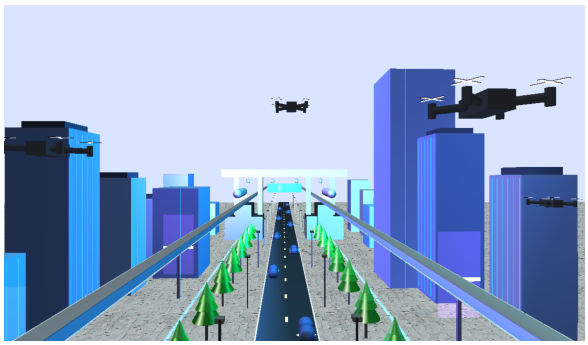
The following are typical screenshot examples from project execution



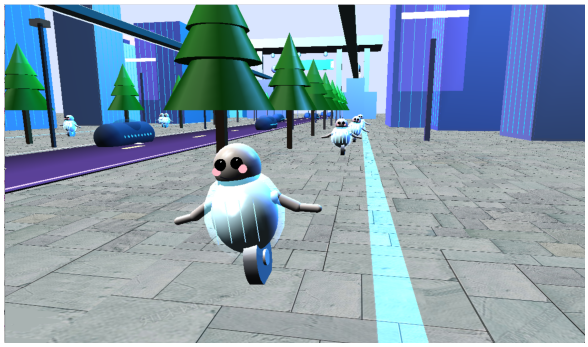
(a) City overview



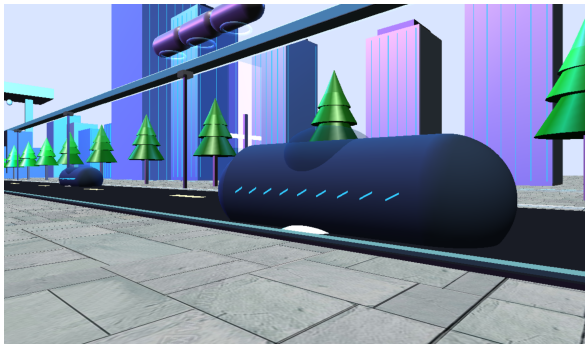
(b) Maglev train



(c) Drone flying



(d) Robot patrolling



(e) Vehicle traffic flow

Figure 1: Typical program execution screenshots. (a) City overview, (b) Maglev train system details, (c) Drone flying, (d) Robot patrolling, (e) Vehicle traffic flow.

5. Conclusion

This project successfully implemented the core technologies covered in the computer graphics course. In summary, this project demonstrates a deep understanding of computer graphics concepts and innovative capabilities, establishing a solid foundation for future endeavors in computer graphics and game development.