Report of CPT205 Assessment 2 – 3D Modelling Project

Module Code	CPT205
Module Name	Computer Graphics
Degree Programme	Information and Computing Science
Name	Yichuan Zhang
Student ID	2251668

This project showcases a carefully designed high-speed railway train running between two caves, along with railway decorations, catenary poles, slopes, and grass, all created using OpenGL and the Freeglut library.

1. Game Instructions

This section is intended to inform some basic instructions about the scene so that you can have a better view of the whole project. Following the instruction sets below, you can modify your viewing position as you want.

Keyboard interaction:

Key W: Move the camera upward in positive y axis.

Key A: Move the camera along the railroad in positive x axis.

Key S: Move the camera downward in negative y axis.

Key D: Move the camera along the railroad in negative x axis.

Mouse interaction:

Left Click: Shift the camera to the other side of the railway.

2. Specific features and design techniques

2.1. Lighting and Material

Material: Material properties such as **GL_AMBIENT**, **GL_DIFFUSE**, **GL_SPECULAR**, and **GL_SHININESS** are used with **glMaterialfv** to define how the material responds to light in terms of ambient reflection, diffuse reflection, specular highlights, and shininess. Different materials will use different parameter values for these properties to achieve their desired appearance.

Light Source Configuration: **glEnable(GL_DEPTH_TEST)** enables depth testing to handle rendering based on object distance. **glEnable(GL_LIGHTING)** activates the lighting system, and **glEnable(GL_LIGHT0)** enables the first light source. **glLightfv** is used to set various properties of the light source: **GL_AMBIENT** defines the ambient light intensity for general illumination, **GL_DIFFUSE** specifies the diffuse light intensity that interacts with surface normals, **GL_SPECULAR** sets the specular intensity for highlights, and **GL_POSITION** specifies the position or direction of the light source. It determines whether the light behaves as a point light or a directional light based on the w-component of the position.

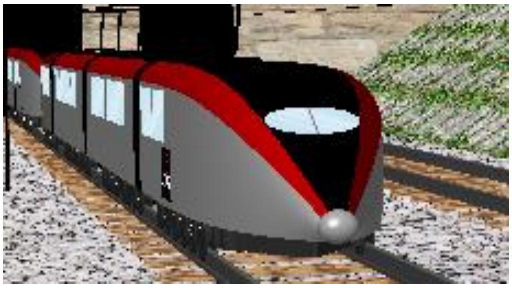

Figure 1.Train front

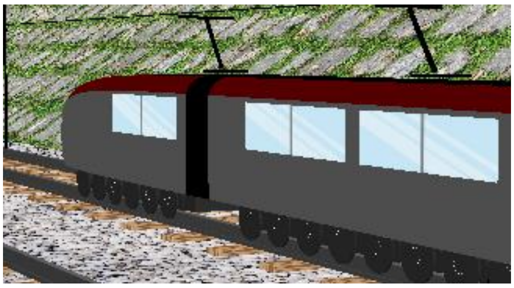

Figure 2.Train back

Here are two sample snapshots illustrating the lighting effects in the scene. The same material is applied to all parts of the train. In the snapshots, the color of the train's side varies depending on its position. Notably, the front of the train appears brighter and closer to white compared to the back.

2.2. Texture Mapping

Figure 3.Texture example

This code uses **glEnable(GL_TEXTURE_2D)** to turn on 2D texture mapping, then reads image data using **ReadImage** and sets the pixel storage alignment with **glPixelStorei(GL_UNPACK_ALIGNMENT, 1)**. Next, **glGenTextures** and **glBindTexture** assign and select texture IDs, while **glTexImage2D** uploads the texture data to the GPU. Texture filtering parameters **(GL_TEXTURE_MAG_FILTER** and **GL_TEXTURE_MIN_FILTER)** are set with **glTexParameterf**, and **glTexEnvf** adjusts how textures interact with lighting (e.g., **GL_MODULATE** or **GL_DECAL**). Finally, when rendering, **glTexCoord2f** provides texture coordinates for each vertex, and **glDisable(GL_TEXTURE_2D)** is used at the end to turn off texture mapping.

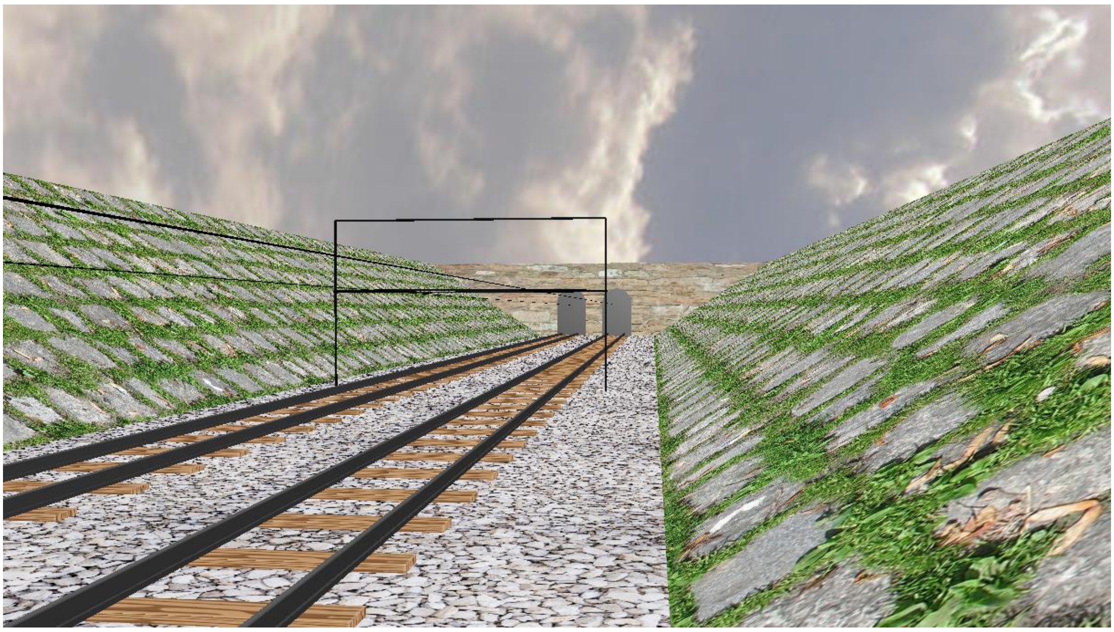

Figure 4.Texture mapping

2.4. Viewing

The **glViewport** function is called to set the view size to match the window's dimensions. Next, the matrix mode is switched to **GL_PROJECTION**, and the **glLoadIdentity()** function is used to reset the projection matrix. The **gluPerspective** function is then utilized to configure a perspective projection. Afterward, the matrix mode is switched to **GL_MODELVIEW**, resetting the model-view matrix to its initial state using **glLoadIdentity()**. Finally, the **gluLookAt** function is invoked to define the camera's viewpoint, specifying its position, the target it looks at, and its upward direction.

The camera's viewpoint always follows the **front of the train**. Users can change the camera's position through keyboard input, and the camera's forward direction is always perpendicular to the ground, pointing upward.

2.5. Hierarchical Modeling

In this project, three key examples demonstrate the scene's construction and management. The high-speed train's front is intricately formed from five **Bézier surfaces** plus an **elliptical section**, all carefully assembled at precise relative positions to create a distinctive shape. Once the train's front geometry is defined, a unified transform operation is applied to the entire train, simplifying all subsequent movements by allowing rotations, translations, or scaling to be performed from a single reference point.

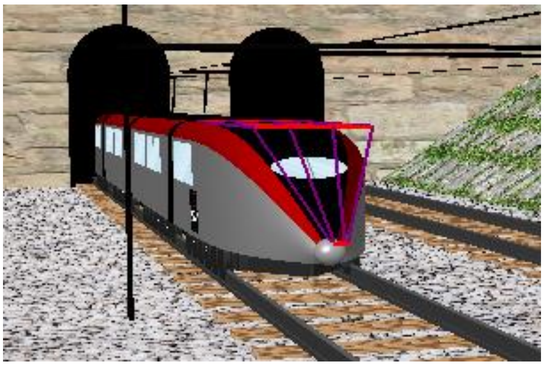

Figure 5.Train front control points(Visualized)

Both the track wood and the power pole are duplicated, enabling multiple instances to fill up the environment.

2.6. Dynamicity

glutTimerFunc is invoked in main function to call after the specified time interval with **OnTimer** as the parameter, achieving the dynamic effect. The timer function controls the movement of the high-speed railway train and the camera in this project. It ensures the fluency of the animation.

Acknowledgement:

I would like to express my gratitude to the creators and contributors of the online texture resources that significantly enhanced the visual quality of this project.

3. Screenshots