COMP3811 - Computer Graphics Coursework 2

Adam Turner - ll16a6t

January 2021

1 Design

For this coursework, I have chosen to base the concept of my scene off a small bar or pub where people would gather and socialise. The scene is constructed from various object which are themselves constructed from smaller object such as cubes and cylinders. In the following section, I will explain the reasoning behind certain design choices made in the construction of the scene.

In order to make the scene feel somewhat realistic, it required object that vaguely resemble furniture you might find in social areas such as tables and chairs. These objects need to be built from smaller objects positioned correctly to achieve the desired look. The table is constructed from objects that you might expect, such as cubes and cylinders. In this case, the cube is actually a thin cuboid with equal x and z dimensions to provide a surface that resembles a table top. The cylinders, which are positioned below the cuboid, are used to create the legs of the table. As opposed to the cuboid, these cylinders have a small base but extend significantly in the vertical direction to create the look of a table leg. They are placed a uniform distance apart from each other at the four edges of the cuboid. On the other hand, the chair is built from more cylinders compared to the table. The chair contains a base similar to the table but at a smaller scale. In addition, more cylinders are used to build the back of the chair. These are similar in size and shape to the legs of the chair but are only positioned on one side of the chair rather than all four. A high number of slices has been used to construct the cylinders to make them appear as smooth as possible. This should make the cylinders appear more rounded and smooth rather than rigid.

There are other pieces of furniture that are used to build the scene too in addition to these tables and chairs. A combination of cubes are used in order to create a counter over which customers would be served. Place on top of this counter are a variety of bottles. These bottles are constructed from two main objects, a cylinder and a a cone. The cylinder has been used to make the section of the bottle that would act as the container for the drink inside whilst the cone is placed on top of the cylinder to create the look of a cap. Each bottle makes use of a different material to provide more variety in the look and feel of the scene. In addition to bottles, there are also drinking glasses in the scene too. Like the bottles, these are made from a cylinder which is slightly shorter and wider. However, the cone is instead replaced with a torus placed on the side of the cylinder that overlaps with the cylinder. This results in half of the torus being visible on the outside of the cylinder which is used to create the handle for the drinking glass.

Aside from the counter, there is one more object used in the scene to make the room feel more alive. A log fireplace constructed from cubes has been placed in the back of the scene. The cubes are used to build the outside of the fireplace and have a red material to resemble how a brick might look.

There is a gap in the middle of the fireplace where the logs would be placed. In this case, the logs are made from cylinders that use a wooden looking material. The cylinders are placed on their side to make them look more like logs compared to the other cylinders in the scene.

The lighting in the scene is positioned directly above the origin towards the roof of the scene. The light source is represented in the scene by a light fixture positioned on the ceiling constructed from a cone and a sphere. The cone is used as the light fixture itself whilst the sphere, which is placed underneath the base of the cone, acts as the light bulb. There is an option to toggle the light from on to off in the interface of the application. The light fixture also doubles as a ceiling fan with made from four cubes placed around the centre of the light. These cubes rotate around the light fixture at a steady rate. This rate is controlled by a timer widget in the main window that updates the angle every time it counts down to zero from the set time limit.

2 Implementation

For the structure of the solution, I have chosen to take a modular approach to the implementation that provides an easy way to further extend the program if necessary. In the following section, I will highlight the main points of the implementation and explain the decision making process behind them.

The main window class creates the window itself and sets up the interface of the application. The widget that creates the scene is a part of this interface but is implemented as a separate class that inherits from QGLWidget as per the coursework specification. The rest of this class defines the interface for interacting with the scene including sliders to control the viewing position and a checkbox to toggle the lighting in the scene on and off. The class itself is split into four different functions, one for initialising the widgets, one for configuring the widgets, one for arranging the position of the widgets in the interface and one for making the connections between the signals emitted from the widgets and the slots in the scene class that control the camera and lighting. This approach of splitting the creation of the widgets into different functions prevents the constructor of the class being too long and messy whilst also making it easier to read and easier to add additional widgets if needed.

The scene class implements the widget that will contain the scene we want to build. As it inherits from QGLWidget, it contains all the necessary OpenGL functions needed to setup and render a scene. There are no functions for drawing objects defined in this class. Whilst it does make use of such functions, the functions themselves are defined in a separate class to provide a clear distinction between the scene itself and the objects that make up that scene. Of course, the paintgl function contains the calls to these functions as well as the transformations that position the objects in the places they need to be. The class also contains several custom slots that are used to control various aspects of the scene such as the values of gluLookAt. There is a slot that will change the x, y and z positions of the eye matrix, centre matrix and up matrix used in the gluLookAt function. Additionally, there is a slot that controls whether GL_LIGHT0 is enabled which is the main light source used in the scene. All of these slots allow the user to interact with the scene in some way.

There are two other classes that are responsible for drawing objects using the standard glut objects as well as object created by OpenGL using polygons. The first of these classes is responsible for creating simple objects such as planes, cubes and cylinders. The second class takes these objects and uses them to make more complex objects such as chairs and tables. This separation helps to distinguish between basic shapes and objects made from basic shapes. It also allows for more objects to easily be added to either class without confusing the reader as to what each class is supposed to do. The relationship that exists between the simple objects class and the complex objects class is a composition, that is, the functions in the complex object class are made up of functions in the simple object class. As such, the complex objects class cannot exist without the simple objects class but the simple objects class could exist on its own if you needed it to. All the materials used by these objects are defined in a separate materials header file. This helps to distinguish the materials from the objects themselves, provide an easy way to add more materials and make the materials available to all classes without having to also include other unwanted functions.

3 Additional Notes

3.1 Installation Instructions

To run this project, use the make file in the main directory of the submission to generate an executable file. This executable will be called ”cw2” and can be found in the same directory as the make file. In the event the make file does not work, you can generate a new make file by using qmake with the ”cw2.pro” file.

1. Navigate to the directory containing the make file which should be the root directory of the submission
2. Run the command ”make”
3. Run the command ”./cw2”

These steps describe the process outlined in the above paragraph. In the event that the make file doesn’t work, try the following steps:

1. Run the command ”make clean”
2. Run the command ”qmake cw2.pro”
3. Run the command ”make”
4. Run the command ”./cw2”