American International University- Bangladesh (AIUB)
Department of Computer Science
Computer Graphics
Spring 2020-2021

Faculty: Dipta Justin Gomes

Section: A
Group:D

Project Report On

City View In Evening

Submitted By –

Kowshik Chakraborty	-	18-36200-1
Tajvir Ahmed Sejan	-	18-36151-1
Munim KS Subhan	-	18-36171-1
SOBHAN MD ABDUS	-	18-37855-2

Introduction

OpenGL provides a set of commands to render a three dimensional scene. OpenGL is a hardware- and system-independent interface. An OpenGL-application work on every platform as long as there is an installed GLUT library.

GLUT is a complete API written by Mark kilgard which allows us to create windows and render the 2D or 3D scenes. It exists for several platforms. that means that a program which uses Glut can be compiled on many platforms without (or at list with very few) changes in the code.

- Main aim of this Project is to illustrate the concepts and usage of pre-built functions in OpenGL.
- Simulation of moving car, bus, plane, some static trees and buildings, river view with a moving boat, sky with some clouds and sun are being done using computer graphics.

Background

The project is 'City View In Evening', so some buildings, moving car, bus, plane, river with a moving boat and some trees are designed with help of OpenGL functions, algorithms and translation, scaling, rotation method. Sky with some clouds and sun is also designed and colored in such a way that the whole model will give a city view during evening time.

Objective of The Project

The main objective of this project is to understand the designing concept using built-in functions, algorithms, some methods like translation, scaling and rotation, how those functions, methods and algorithms works, how the whole thing is processing and may more things.

Implementation

This program is implemented using various OpenGL functions which are shown below. Various functions used in this program:

•glutInit() : interaction between the windowing system and OPENGL is initiated.

•glutInitDisplayMode() : used when double buffering is required and depth information is required.

•glutCreateWindow() : this opens the OPENGL window and displays the title at top of the window.

•glutInitWindowSize() : specifies the size of the window.

•glutInitWindowPosition() : specifies the position of the window in screen co-ordinates .

•glutKeyboardFunc() : handles normal ascii symbols.

•glutDisplayFunc() : this handles redrawing of the window.

•glutMainLoop() : this starts the main loop, it never returns.

•glFlush() : used to flush the pipeline city View

•glBegin() : delimit the vertices of a primitive or a group of like primitives.

•glPushMatrix() : push and pop the current matrix stack.

• glVertex2f() : specify a vertex.

• glColor3f() : set the current color.

•glutPostRedisplay() : used to trigger an automatic re-drawing of the object.

•glMatrixMode() : used to set up the required mode of the matrix.

•glLoadIdentity() : used to load or initialize to the identity matrix.

•glTranslatef() : used to translate or move the rotation centre from one point to another in three dimensions.

•glScalef() : multiply the current matrix by a general scaling matrix.

System specifications

SOFTWARE REQUIREMENTS:

- CodeBlocks
- OPENGL

Source Code:

```
#include <windows.h>
#include<iostream>
#include<GL/gl.h>
#include<GL/glut.h>
#include<math.h>
#include<cstdio>
# define PI  3.14159265358979323846
using namespace std;

float _move = 0.0f;  //plane
float _move1 = 0.0f; //wave1
float _move2 = 0.0f; //wave2
float _move3 = 0.0f; //boat
float _move4 = 0.0f; //bus
float _move5 = 0.0f; //bird
float _move6 = 0.0f; //cloud1
float _move7 = 0.0f; //cloud2
float _move_car_1 = 0.0f;//car

void drawScene(){
glClear(GL_COLOR_BUFFER_BIT);
```

```
glClearColor(1.0, 1.0, 1.0, 0.0);
glLoadIdentity(); //Reset the drawing perspective
glMatrixMode(GL_MODELVIEW);

  //Sky
glPushMatrix();
glBegin(GL_POLYGON);
    glColor3f(1.000, 0.388, 0.278);
    glVertex3f(-25, 0.0, 0.0);
    glVertex3f(-25, 11.5, 0.0);
    glVertex3f(25, 11.5, 0.0);
    glVertex3f(25, 0.0, 0.0);
glEnd();
glPopMatrix();

  //sun
glPushMatrix();
  //glTranslatef(6,0,0);
glTranslatef(0.8,0.8,0.0);
  glColor3f(1.5,0.2,0.0);
glBegin(GL_POLYGON);

  for (int i=0; i<600;i++)
  {
     float pi = 3.1416;
     float A = (i*2*pi)/100;
     float r = 0.1;
     float x = r * cos(A);
     float y = r * sin(A);
     glVertex2f(x,y);
  }
```

```
glEnd();
glPopMatrix();

   //Cloud 1

glPushMatrix();
glTranslatef(_move6+0.2, 0.8, 0.0);
   glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);

for(int i=0; i<600; i++)
   {
      float pi = 3.1416;
      float A = (i*2*pi)/100;
      float r = 0.05;
      float x = r * cos(A);
      float y = r * sin(A);
      glVertex2f(x,y);
   }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move6+0.28, 0.8, 0.0);
   glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++)
   {
      float pi = 3.1416;
      float A = (i*2*pi)/100;
      float r = 0.05;
```

```
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move6+0.35, 0.8, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++)
     {
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move6+0.2, 0.75, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++)
      {
        float pi = 3.1416;
        float A = (i*2*pi)/100;
```

```
          float r = 0.05;
          float x = r * cos(A);
          float y = r * sin(A);
          glVertex2f(x,y);
        }
    glEnd();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(_move6+0.28, 0.75, 0.0);
        glColor3f(1.5, 0.4, 0.0);
    glBegin(GL_POLYGON);
    for(int i=0; i<600; i++)
        {
          float pi = 3.1416;
          float A = (i*2*pi)/100;
          float r = 0.05;
          float x = r * cos(A);
          float y = r * sin(A);
          glVertex2f(x,y);
        }
    glEnd();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(_move6+0.35, 0.75, 0.0);
        glColor3f(1.5, 0.4, 0.0);
    glBegin(GL_POLYGON);
    for(int i=0; i<600; i++)
        {
          float pi = 3.1416;
          float A = (i*2*pi)/100;
          float r = 0.05;
          float x = r * cos(A);
          float y = r * sin(A);
          glVertex2f(x,y);
        }
    glEnd();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(_move6+0.15, 0.77, 0.0);
        glColor3f(1.5, 0.4, 0.0);
    glBegin(GL_POLYGON);
    for(int i=0; i<600; i++)
        {
          float pi = 3.1416;
          float A = (i*2*pi)/100;
          float r = 0.05;
          float x = r * cos(A);
          float y = r * sin(A);
          glVertex2f(x,y);
        }
    glEnd();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(_move6+0.39, 0.77, 0.0);
        glColor3f(1.5, 0.4, 0.0);
    glBegin(GL_POLYGON);
    for(int i=0; i<600; i++)
        {
```

```
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
       }
glEnd();
glPopMatrix();

   //Cloud 2

glPushMatrix();
glTranslatef(_move7+0.6, 0.6, 0.0);
   glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
   }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.68, 0.6, 0.0);
   glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
```

```
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.75, 0.6, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.6, 0.65, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.68, 0.65, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.75, 0.65, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.55, 0.62, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
      }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move7+0.79, 0.62, 0.0);
      glColor3f(1.5, 0.4, 0.0);
glBegin(GL_POLYGON);
for(int i=0; i<600; i++){
        float pi = 3.1416;
        float A = (i*2*pi)/100;
        float r = 0.05;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
       }
   glEnd();
   glPopMatrix();


     //Road
   glPushMatrix();
   glBegin(GL_POLYGON);
       glColor3f(0.0, 0.0, 0.0);
       glVertex3f(-1.0f, -0.33f, 0.0f);
       glVertex3f(1.0f, -0.33f, 0.0f);
       glVertex3f(1.0f, 0.0f, 0.0f);
       glVertex3f(-1.0f, 0.0f, 0.0f);
   glEnd();
   glPopMatrix();

     //RoadLines
   glPushMatrix();
   glBegin(GL_POLYGON);
       glColor3f(1.0, 1.0, 1.0);
       glVertex3f(-1.0f, -0.166f, 0.0f);
       glVertex3f(-0.5f, -0.166f, 0.0f);
       glVertex3f(-0.5f, -0.13f, 0.0f);
       glVertex3f(-1.0f, -0.13f, 0.0f);
   glEnd();
   glBegin(GL_POLYGON);
       glColor3f(1.0, 1.0, 1.0);
```

```
        glVertex3f(-0.33f, -0.166f, 0.0f);
        glVertex3f(-0.166f, -0.166f, 0.0f);
        glVertex3f(-0.166f, -0.13f, 0.0f);
        glVertex3f(-0.33f, -0.13f, 0.0f);
    glEnd();
    glBegin(GL_POLYGON);
        glColor3f(1.0, 1.0, 1.0);
        glVertex3f(0.33f, -0.166f, 0.0f);
        glVertex3f(0.833f, -0.166f, 0.0f);
        glVertex3f(0.833f, -0.13f, 0.0f);
        glVertex3f(0.33f, -0.13f, 0.0f);
    glEnd();
    glBegin(GL_POLYGON);
        glColor3f(1.0, 1.0, 1.0);
        glVertex3f(0.833f, -0.166f, 0.0f);
        glVertex3f(1.0f, -0.166f, 0.0f);
        glVertex3f(1.0f, -0.13f, 0.0f);
        glVertex3f(0.833f, -0.13f, 0.0f);
    glEnd();
    glPopMatrix();

      //Lake
    glPushMatrix();
    glBegin(GL_POLYGON);
        glColor3f(0.23, 0.70, 0.81);
        glVertex3f(-1.0f, -1.0f, 0.0f);
        glVertex3f(1.0f, -1.0f, 0.0f);
        glVertex3f(1.0f, -0.33f, 0.0f);
        glVertex3f(-1.0f, -0.33f, 0.0f);
    glEnd();
    glPopMatrix();
```

```
  //Border
glPushMatrix();
glBegin(GL_POLYGON);
     glColor3f(0.2, 0.098, 0.0);
     glVertex3f(-1.0f, -0.33f, 0.0f);
     glVertex3f(1.0f, -0.33f, 0.0f);
     glVertex3f(1.0f, -0.416f, 0.0f);
     glVertex3f(-1.0f, -0.416f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
     glColor3f(0.0, 0.0, 0.0);
     glVertex3f(-1.0f, -0.35f, 0.0f);
     glVertex3f(1.0f, -0.35f, 0.0f);
     glVertex3f(1.0f, -0.33f, 0.0f);
     glVertex3f(-1.0f, -0.33f, 0.0f);
glEnd();
glPopMatrix();

////Plane////
glPushMatrix();
glTranslatef(_move1, 0.0f, 0.0f);

glBegin(GL_POLYGON);
   glColor3f(0.0,1.0, 0.0);
   glVertex3f(-1.0f, 0.5f, 0.0f);
   glVertex3f(-0.5f, 0.5f, 0.0f);
   glVertex3f(-0.366f, 0.55f, 0.0f);
   glVertex3f(-0.45f, 0.616f, 0.0f);
   glVertex3f(-0.5f, 0.66f, 0.0f);
   glVertex3f(-1.0f, 0.66f, 0.0f);
```

```
glEnd();


    ////PlaneWindowa///
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0, 0.0);
    glVertex3f(-0.966f, 0.55f, 0.0f);
    glVertex3f(-0.916f, 0.55f, 0.0f);
    glVertex3f(-0.916f, 0.63f, 0.0f);
    glVertex3f(-0.966f, 0.63f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.0, 1.0);
    glVertex3f(-0.866f, 0.55f, 0.0f);
    glVertex3f(-0.816f, 0.55f, 0.0f);
    glVertex3f(-0.816f, 0.63f, 0.0f);
    glVertex3f(-0.866f, 0.63f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,1.0, 1.0);
    glVertex3f(-0.766f, 0.55f, 0.0f);
    glVertex3f(-0.716f, 0.55f, 0.0f);
    glVertex3f(-0.716f, 0.63f, 0.0f);
    glVertex3f(-0.766f, 0.63f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 0.0);
    glVertex3f(-0.666f, 0.55f, 0.0f);
    glVertex3f(-0.616f, 0.55f, 0.0f);
    glVertex3f(-0.616f, 0.63f, 0.0f);
    glVertex3f(-0.666f, 0.63f, 0.0f);
glEnd();
```

```
glBegin(GL_POLYGON);
    glColor3f(0.5,0.8, 0.7);
    glVertex3f(-0.566f, 0.55f, 0.0f);
    glVertex3f(-0.516f, 0.55f, 0.0f);
    glVertex3f(-0.516f, 0.63f, 0.0f);
    glVertex3f(-0.566f, 0.63f, 0.0f);
glEnd();

    ////Planesides////
glBegin(GL_POLYGON);
    glColor3f(1.0,0.6, 0.1);
    glVertex3f(-0.916f, 0.66f, 0.0f);
    glVertex3f(-0.83f, 0.66f, 0.0f);
    glVertex3f(-0.883f, 0.833f, 0.0f);
    glVertex3f(-0.916f, 0.833f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0, 0.0);
    glVertex3f(-0.66f, 0.66f, 0.0f);
    glVertex3f(-0.583f, 0.66f, 0.0f);
    glVertex3f(-0.633f, 0.833f, 0.0f);
    glVertex3f(-0.7f, 0.833f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0, 0.0);
    glVertex3f(-0.66f, 0.33f, 0.0f);
    glVertex3f(-0.616f, 0.33f, 0.0f);
    glVertex3f(-0.583f, 0.516f, 0.0f);
    glVertex3f(-0.68f, 0.516f, 0.0f);
glEnd();
glPopMatrix();
```

```
    ////BuildingsBackRow////

    ////1////
glPushMatrix();
glBegin(GL_POLYGON);
    glColor3f(1.4,1.2, 0.0);
    glVertex3f(-0.66f, 0.0f, 0.0f);
    glVertex3f(-0.416f, 0.0f, 0.0f);
    glVertex3f(-0.416f, 0.25f, 0.0f);
    glVertex3f(-0.66f, 0.25f, 0.0f);
glEnd();

    ////2////
glBegin(GL_POLYGON);
    glColor3f(1.6,0.675, 0.29);
    glVertex3f(0.33f, 0.0f, 0.0f);
    glVertex3f(0.5f, 0.0f, 0.0f);
    glVertex3f(0.5f, 0.25f, 0.0f);
    glVertex3f(0.33f, 0.25f, 0.0f);
glEnd();

    ////2////
glBegin(GL_POLYGON);
    glColor3f(0.0,0.0, 0.0);
    glVertex3f(0.833f, 0.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, 0.25f, 0.0f);
    glVertex3f(0.833f, 0.25f, 0.0f);
glEnd();
```

```
    /////BackRowBuildingWindow////

    ////1////
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.65f, 0.166f, 0.0f);
    glVertex3f(-0.566f, 0.166f, 0.0f);
    glVertex3f(-0.566f, 0.216f, 0.0f);
    glVertex3f(-0.65f, 0.216f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.65f, 0.15f, 0.0f);
    glVertex3f(-0.566f, 0.15f, 0.0f);
    glVertex3f(-0.566f, 0.1f, 0.0f);
    glVertex3f(-0.65f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.65f, 0.083f, 0.0f);
    glVertex3f(-0.566f, 0.083f, 0.0f);
    glVertex3f(-0.566f, 0.033f, 0.0f);
    glVertex3f(-0.65f, 0.033f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.55f, 0.166f, 0.0f);
    glVertex3f(-0.5f, 0.166f, 0.0f);
    glVertex3f(-0.5f, 0.216f, 0.0f);
    glVertex3f(-0.55f, 0.216f, 0.0f);
glEnd();
```

```
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.55f, 0.15f, 0.0f);
    glVertex3f(-0.5f, 0.15f, 0.0f);
    glVertex3f(-0.5f, 0.1f, 0.0f);
    glVertex3f(-0.55f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.55f, 0.083f, 0.0f);
    glVertex3f(-0.5f, 0.083f, 0.0f);
    glVertex3f(-0.5f, 0.033f, 0.0f);
    glVertex3f(-0.55f, 0.033f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.483f, 0.166f, 0.0f);
    glVertex3f(-0.433f, 0.166f, 0.0f);
    glVertex3f(-0.433f, 0.216f, 0.0f);
    glVertex3f(-0.483f, 0.216f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.483f, 0.15f, 0.0f);
    glVertex3f(-0.433f, 0.15f, 0.0f);
    glVertex3f(-0.433f, 0.1f, 0.0f);
    glVertex3f(-0.483f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(-0.483f, 0.083f, 0.0f);
    glVertex3f(-0.433f, 0.083f, 0.0f);
    glVertex3f(-0.433f, 0.033f, 0.0f);
    glVertex3f(-0.483f, 0.033f, 0.0f);
glEnd();

    ////2////
glBegin(GL_POLYGON);
    glColor3f(1.0,0.8, 1.0);
    glVertex3f(0.366f, 0.166f, 0.0f);
    glVertex3f(0.466f, 0.166f, 0.0f);
    glVertex3f(0.466f, 0.216f, 0.0f);
    glVertex3f(0.366f, 0.216f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.8, 1.0);
    glVertex3f(0.366f, 0.15f, 0.0f);
    glVertex3f(0.466f, 0.15f, 0.0f);
    glVertex3f(0.466f, 0.1f, 0.0f);
    glVertex3f(0.366f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.8, 1.0);
    glVertex3f(0.366f, 0.083f, 0.0f);
    glVertex3f(0.466f, 0.083f, 0.0f);
    glVertex3f(0.466f, 0.033f, 0.0f);
    glVertex3f(0.366f, 0.033f, 0.0f);
glEnd();

    ////3////
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 0.4);
    glVertex3f(0.866f, 0.166f, 0.0f);
    glVertex3f(0.966f, 0.166f, 0.0f);
    glVertex3f(0.966f, 0.216f, 0.0f);
    glVertex3f(0.866f, 0.216f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 0.4);
    glVertex3f(0.866f, 0.15f, 0.0f);
    glVertex3f(0.966f, 0.15f, 0.0f);
    glVertex3f(0.966f, 0.1f, 0.0f);
    glVertex3f(0.866f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0, 0.4);
    glVertex3f(0.866f, 0.083f, 0.0f);
    glVertex3f(0.966f, 0.083f, 0.0f);
    glVertex3f(0.966f, 0.033f, 0.0f);
    glVertex3f(0.866f, 0.033f, 0.0f);
glEnd();

    //car

glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
    glColor3f(1.0,1.0,0.0);
glBegin(GL_POLYGON);
        glVertex2f(-0.85, 0.0);
        glVertex2f(-0.55, 0.0);
        glVertex2f(-0.55, 0.07);
        glVertex2f(-0.60, 0.07);
        glVertex2f(-0.65, 0.17);
        glVertex2f(-0.75, 0.17);
        glVertex2f(-0.80, 0.17);
        glVertex2f(-0.85, 0.07);
        glVertex2f(-0.85, 0.0);
glEnd();
glPopMatrix();

    glColor3f(0.0, 0.0, 0.0);
glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
glBegin(GL_POLYGON);
        glVertex2f(-0.61, 0.07);
        glVertex2f(-0.66, 0.16);
        glVertex2f(-0.79, 0.16);
        glVertex2f(-0.84, 0.07);
glEnd();
glPopMatrix();

    glColor3f(1.000, 0.0, 0.0);
glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
glBegin(GL_POLYGON);
        glVertex2f(-0.73, 0.07);
        glVertex2f(-0.72, 0.07);
        glVertex2f(-0.72, 0.16);
        glVertex2f(-0.73, 0.16);
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
glTranslatef(-0.80,0.01,0);
glScalef(0.6,1,1);
        glColor3f(0.000, 0.000, 0.000);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
        {
            float pi=3.1416;
            float A=(i*2*pi)/200;
            float r=0.04;
            float x = r * cos(A);
            float y = r * sin(A);
            glVertex2f(x,y );
        }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
glTranslatef(-0.80,0.01,0);
glScalef(0.6,1,1);
        glColor3f(1.000, 1.000, 1.000);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
        {
            float pi=3.1416;
            float A=(i*2*pi)/200;
            float r=0.01;
            float x = r * cos(A);
            float y = r * sin(A);
            glVertex2f(x,y );
        }
```

```
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
glTranslatef(-0.62,0.01,0);
glScalef(0.6,1,1);
        glColor3f(0.000, 0.000, 0.000);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
        {
            float pi=3.1416;
            float A=(i*2*pi)/200;
            float r=0.04;
            float x = r * cos(A);
            float y = r * sin(A);
            glVertex2f(x,y );
        }
glEnd();
glPopMatrix();

glPushMatrix();
glTranslatef(_move_car_1, 0.0f, 0.0f);
glTranslatef(-0.62,0.01,0);
glScalef(0.6,1,1);
        glColor3f(1.000, 1.000, 1.000);
glBegin(GL_POLYGON);
for(int i=0;i<200;i++)
        {
            float pi=3.1416;
            float A=(i*2*pi)/200;
            float r=0.01;
            float x = r * cos(A);
            float y = r * sin(A);
            glVertex2f(x,y );
        }
    glEnd();
    glPopMatrix();






     ////Bus////
    glPushMatrix();
    glTranslatef(_move4, 0.0f, 0.0f);
    glBegin(GL_POLYGON);
     glColor3f(1.0,0.0, 1.0); //lower part
     glVertex3f(0.5f, -0.166f, 0.0f);
     glVertex3f(1.0f, -0.166f, 0.0f);
     glVertex3f(1.0f, 0.0f, 0.0f);
     glVertex3f(0.55f, 0.0f, 0.0f);
     glVertex3f(0.5f, -0.033f, 0.0f);
    glEnd();
    glBegin(GL_POLYGON); //headlight
     glColor3f(1.0,1.0, 0.0);
     glVertex3f(0.5f, -0.1f, 0.0f);
     glVertex3f(0.5166f, -0.1f, 0.0f);
     glVertex3f(0.5166f, -0.066f, 0.0f);
     glVertex3f(0.5f, -0.066f, 0.0f);
    glEnd();
```

```
glBegin(GL_POLYGON); //upper part
    glColor3f(1.0,1.0, 1.0);
    glVertex3f(0.55f, 0.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, 0.0f);
    glVertex3f(1.0f, 0.116f, 0.0f);
    glVertex3f(0.55f, 0.116f, 0.0f);
glEnd();

    ////BusWindows////
glBegin(GL_POLYGON);
    glColor3f(0.26,0.26, 0.26);
    glVertex3f(0.5833f, 0.016f, 0.0f);
    glVertex3f(0.633f, 0.016f, 0.0f);
    glVertex3f(0.633f, 0.1f, 0.0f);
    glVertex3f(0.5833f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.26,0.26, 0.26);
    glVertex3f(0.65f, 0.016f, 0.0f);
    glVertex3f(0.7f, 0.016f, 0.0f);
    glVertex3f(0.7f, 0.1f, 0.0f);
    glVertex3f(0.65f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.26,0.26, 0.26);
    glVertex3f(0.716f, 0.016f, 0.0f);
    glVertex3f(0.766f, 0.016f, 0.0f);
    glVertex3f(0.766f, 0.1f, 0.0f);
    glVertex3f(0.716f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
```

```
    glColor3f(0.26,0.26, 0.26);
    glVertex3f(0.783f, 0.016f, 0.0f);
    glVertex3f(0.833f, 0.016f, 0.0f);
    glVertex3f(0.833f, 0.1f, 0.0f);
    glVertex3f(0.783f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.26,0.26, 0.26);
    glVertex3f(0.85f, 0.016f, 0.0f);
    glVertex3f(0.9f, 0.016f, 0.0f);
    glVertex3f(0.9f, 0.1f, 0.0f);
    glVertex3f(0.85f, 0.1f, 0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.26,0.26, 0.26);
    glVertex3f(0.916f, 0.016f, 0.0f);
    glVertex3f(0.966f, 0.016f, 0.0f);
    glVertex3f(0.966f, 0.1f, 0.0f);
    glVertex3f(0.916f, 0.1f, 0.0f);
glEnd();
glPopMatrix();



    ////BusWheels////
glPushMatrix();
    //glTranslatef(6,0,0);
glTranslatef(_move4+0.6, -0.2, 0.0);
    glColor3f(0.0,0.0,0.0);
glBegin(GL_POLYGON);
```

```
for(int i=0;i<600;i++)
    {
        float pi=3.1416;
        float A=(i*2*pi)/100;
        float r=0.06;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y);
    }
glEnd();
glPopMatrix();
glPushMatrix();
glTranslatef(_move4+0.6, -0.2, 0.0);
    glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);

for(int i=0;i<600;i++)
    {
        float pi=3.1416;
        float A=(i*2*pi)/100;
        float r=0.03;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y );
    }
glEnd();
glPopMatrix();
glPushMatrix();
    //glTranslatef(6,0,0);
glTranslatef(_move4+0.92, -0.2, 0.0);
    glColor3f(0.0,0.0,0.0);
```

```
glBegin(GL_POLYGON);

for(int i=0;i<600;i++)
    {
        float pi=3.1416;
        float A=(i*2*pi)/100;
        float r=0.06;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y );
    }
glEnd();
glPopMatrix();
glPushMatrix();
    //glTranslatef(6,0,0);
glTranslatef(_move4+0.92, -0.2, 0.0);
    glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);

for(int i=0;i<600;i++)
    {
        float pi=3.1416;
        float A=(i*2*pi)/100;
        float r=0.03;
        float x = r * cos(A);
        float y = r * sin(A);
        glVertex2f(x,y );
    }
glEnd();
glPopMatrix();
```

```
    ////BuildingsFrontRow////

    ////1////
glPushMatrix();
glBegin(GL_POLYGON);
    glColor3f(0.0,1.0,0.0);
    glVertex3f(-1.0f, -0.33f, 0.0f);
    glVertex3f(-0.66f,-0.33f,0.0f);
    glVertex3f(-0.66f,0.33f,0.0f);
    glVertex3f(-1.0f,0.33f,0.0f);
glEnd();




    //4
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(0.5f,-0.33f,0.0f);
    glVertex3f(0.833f,-0.33f,0.0f);
    glVertex3f(0.833f,0.33f,0.0f);
    glVertex3f(0.5f,0.33f,0.0f);
glEnd();
glPopMatrix();

    //Building windows

    //1
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.966f,0.166f,0.0f);
    glVertex3f(-0.833f,0.166f,0.0f);
    glVertex3f(-0.833f,0.25f,0.0f);
    glVertex3f(-0.966f,0.25f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.7833f,0.166f,0.0f);
    glVertex3f(-0.7f,0.166f,0.0f);
    glVertex3f(-0.7f,0.25f,0.0f);
    glVertex3f(-0.7833f,0.25f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.7833f,0.0f,0.0f);
    glVertex3f(-0.7f,0.0f,0.0f);
    glVertex3f(-0.7f,0.0833f,0.0f);
    glVertex3f(-0.7833f,0.0833f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.966f,0.0f,0.0f);
    glVertex3f(-0.833f,0.0f,0.0f);
    glVertex3f(-0.833f,0.0833f,0.0f);
    glVertex3f(-0.966f,0.0833f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.966f,-0.0833f,0.0f);
    glVertex3f(-0.833f,-0.0833f,0.0f);
    glVertex3f(-0.833f,-0.166f,0.0f);
    glVertex3f(-0.966f,-0.166f,0.0f);
```

```
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.7833f,-0.0833f,0.0f);
    glVertex3f(-0.7f,-0.0833f,0.0f);
    glVertex3f(-0.7f,-0.166f,0.0f);
    glVertex3f(-0.7833f,-0.166f,0.0f);
glEnd();



    //4
glBegin(GL_POLYGON);
    glColor3f(1.0,0.25,0.0);
    glVertex3f(0.583f,0.166f,0.0f);
    glVertex3f(0.65f,0.166f,0.0f);
    glVertex3f(0.65f,0.25f,0.0f);
    glVertex3f(0.583f,0.25f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.25,0.0);
    glVertex3f(0.683f,0.166f,0.0f);
    glVertex3f(0.75f,0.166f,0.0f);
    glVertex3f(0.75f,0.25f,0.0f);
    glVertex3f(0.683f,0.25f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.25,0.0);
    glVertex3f(0.583f,0.083f,0.0f);
    glVertex3f(0.65f,0.083f,0.0f);
    glVertex3f(0.65f,0.0f,0.0f);
    glVertex3f(0.583f,0.0f,0.0f);
```

```
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.25,0.0);
    glVertex3f(0.683f,0.083f,0.0f);
    glVertex3f(0.75f,0.083f,0.0f);
    glVertex3f(0.75f,0.0f,0.0f);
    glVertex3f(0.683f,0.0f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.25,0.0);
    glVertex3f(0.583f,-0.083f,0.0f);
    glVertex3f(0.65f,-0.083f,0.0f);
    glVertex3f(0.65f,-0.166f,0.0f);
    glVertex3f(0.583f,-0.166f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,0.25,0.0);
    glVertex3f(0.683f,-0.083f,0.0f);
    glVertex3f(0.75f,-0.083f,0.0f);
    glVertex3f(0.75f,-0.166f,0.0f);
    glVertex3f(0.683f,-0.166f,0.0f);
glEnd();


    //Trees
glPushMatrix();
glBegin(GL_POLYGON);
    glColor3f(0.6,0.298,0.0);
    glVertex3f(-0.6f,-0.33f,0.0f);
    glVertex3f(-0.566f,-0.33f,0.0f);
    glVertex3f(-0.566f,0.166f,0.0f);
    glVertex3f(-0.6f,0.166f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.6,0.298,0.0);
    glVertex3f(-0.1f,-0.33f,0.0f);
    glVertex3f(-0.066f,-0.33f,0.0f);
    glVertex3f(-0.066f,0.166f,0.0f);
    glVertex3f(-0.1f,0.166f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.6,0.298,0.0);
    glVertex3f(0.4f,-0.33f,0.0f);
    glVertex3f(0.433f,-0.33f,0.0f);
    glVertex3f(0.433f,0.166f,0.0f);
    glVertex3f(0.4f,0.166f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.6,0.298,0.0);
    glVertex3f(0.9f,-0.33f,0.0f);
    glVertex3f(0.933f,-0.33f,0.0f);
    glVertex3f(0.933f,0.166f,0.0f);
    glVertex3f(0.9f,0.166f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.4,0.0);
    glVertex3f(-0.65f,-0.166f,0.0f);
    glVertex3f(-0.516f,-0.166f,0.0f);
    glVertex3f(-0.583f,-0.0f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.4,0.0);
    glVertex3f(-0.15f,-0.166f,0.0f);
    glVertex3f(-0.016f,-0.166f,0.0f);
    glVertex3f(-0.0833f,-0.0f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.4,0.0);
    glVertex3f(0.35f,-0.166f,0.0f);
    glVertex3f(0.483f,-0.166f,0.0f);
    glVertex3f(0.4166f,0.0f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.4,0.0);
    glVertex3f(0.85f,-0.166f,0.0f);
    glVertex3f(0.983f,-0.166f,0.0f);
    glVertex3f(0.9166f,0.0f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.4,0.8,0.0);
    glVertex3f(-0.65f,-0.05f,0.0f);
    glVertex3f(-0.516f,-0.05f,0.0f);
    glVertex3f(-0.583f,0.33f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.4,0.8,0.0);
    glVertex3f(-0.15f,-0.05f,0.0f);
    glVertex3f(-0.016f,-0.05f,0.0f);
    glVertex3f(-0.0833f,0.33f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.4,0.8,0.0);
    glVertex3f(0.35f,-0.05f,0.0f);
    glVertex3f(0.483f,-0.05f,0.0f);
    glVertex3f(0.4166f,0.33f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.4,0.8,0.0);
    glVertex3f(0.85f,-0.05f,0.0f);
    glVertex3f(0.983f,-0.05f,0.0f);
    glVertex3f(0.9166f,0.33f,0.0f);
glEnd();
glPopMatrix();

    //Boat
glPushMatrix();
glTranslatef(_move3,0.0f,0.0f);
glBegin(GL_POLYGON);
    glColor3f(0.4,0.0,1.0);
    glVertex3f(-0.833f,-0.66f,0.0f);
    glVertex3f(-0.33f,-0.66f,0.0f);
    glVertex3f(-0.25f,-0.583f,0.0f);
    glVertex3f(-0.916f,-0.583f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0,0.0);
    glVertex3f(-0.833f,-0.583f,0.0f);
    glVertex3f(-0.33f,-0.583f,0.0f);
    glVertex3f(-0.416f,-0.5f,0.0f);
    glVertex3f(-0.75f,-0.5f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.0,1.0);
    glVertex3f(-0.66f,-0.5f,0.0f);
    glVertex3f(-0.5f,-0.5f,0.0f);
    glVertex3f(-0.583f,-0.33f,0.0f);
glEnd();
glBegin(GL_LINES);
    glColor3f(0.0,0.0,0.0);
    glVertex3f(-0.583f,-0.33f,0.0f);
    glVertex3f(-0.583f,-0.166f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(1.0,1.0,1.0);
    glVertex3f(-0.583f,-0.283f,0.0f);
    glVertex3f(-0.55f,-0.25f,0.0f);
    glVertex3f(-0.583f,-0.2166f,0.0f);
glEnd();

    //Boat Windows
glBegin(GL_POLYGON);
    glColor3f(0.0,1.0,0.0);
    glVertex3f(-0.75f,-0.566f,0.0f);
    glVertex3f(-0.7f,-0.566f,0.0f);
    glVertex3f(-0.7f,-0.516f,0.0f);
    glVertex3f(-0.75f,-0.516f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,1.0,1.0);
    glVertex3f(-0.633f,-0.566f,0.0f);
    glVertex3f(-0.583f,-0.566f,0.0f);
    glVertex3f(-0.583f,-0.516f,0.0f);
    glVertex3f(-0.633f,-0.516f,0.0f);
glEnd();
glBegin(GL_POLYGON);
```

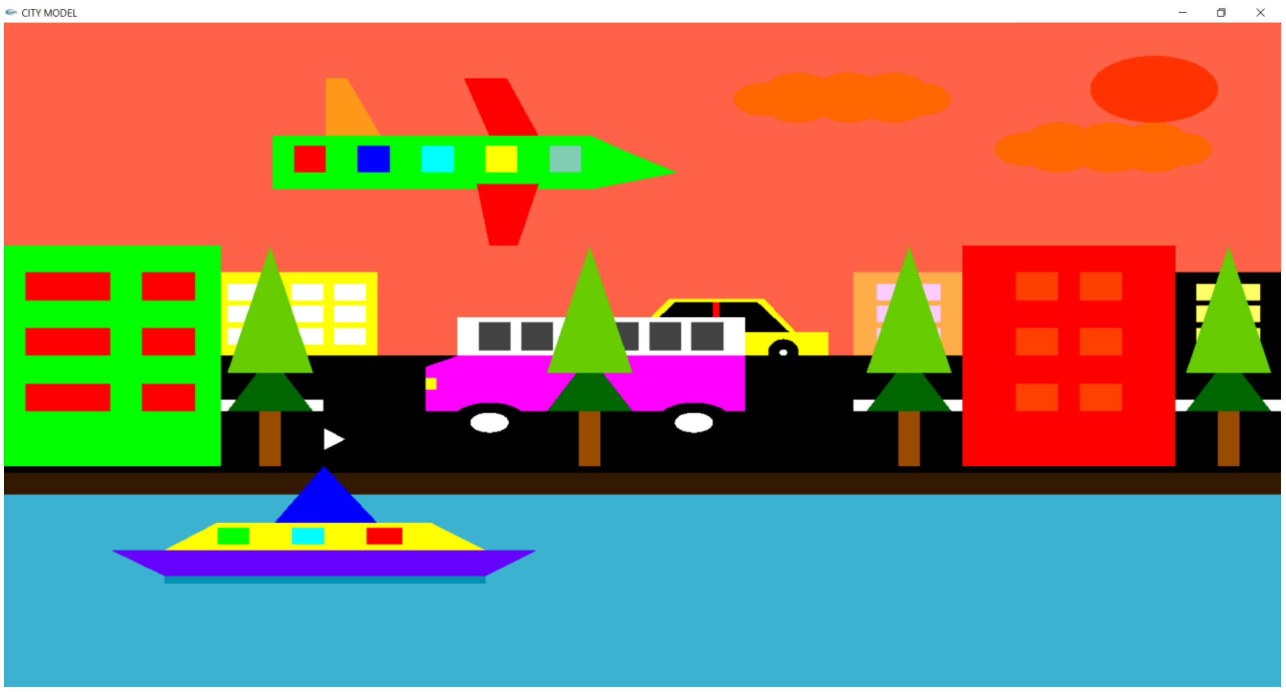
```
    glColor3f(1.0,0.0,0.0);
    glVertex3f(-0.516f,-0.566f,0.0f);
    glVertex3f(-0.46f,-0.566f,0.0f);
    glVertex3f(-0.46f,-0.516f,0.0f);
    glVertex3f(-0.516f,-0.516f,0.0f);
glEnd();
glBegin(GL_POLYGON);
    glColor3f(0.0,0.56,0.698);
    glVertex3f(-0.833f,-0.6833f,0.0f);
    glVertex3f(-0.33f,-0.6833f,0.0f);
    glVertex3f(-0.33f,-0.66f,0.0f);
    glVertex3f(-0.833f,-0.66f,0.0f);
glEnd();

glPopMatrix();

glutSwapBuffers();
}



void update(int value){

    _move +=0.0001f;
if(_move-1.5 > 1.0)
    {
        _move = -1.0;
    }


    _move1 +=0.005f;
```

```
if(_move1-1.966 > 1.0)
    {
        _move1 = -1.0;
    }



    _move2 +=0.01f;
if(_move2-1.663 > 1.0)
    {
        _move2 = -1.0;
    }

    _move3 +=0.001f;
if(_move3-1.663 > 1.0)
    {
        _move3 = -1.0;
    }

    _move_car_1 +=0.01f;
if(_move_car_1-1.663 > 1.0)
    {
        _move_car_1 = -1.0;
    }

    _move4 -=0.01f;
if(_move4+1.5 < -1.0)
    {
        _move4 = 1.0;
    }
    _move5 -=0.1f;
if(_move5+1.966 < -1.0)
```

```
    {
        _move5 = 1.0;
    }
    _move6 +=0.0005f;
if(_move6-1.39 > 1.0)
    {
        _move6 = -1.4;
    }

    _move7 +=0.0006f;
    if(_move7-0.79>1.0)
    {
        _move7 = -1.4;
    }
glutPostRedisplay();



glutTimerFunc(25, update, 0);
}

int main(int argc, char** argv)
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
glutInitWindowSize(1200,1200);
glutCreateWindow("CITY MODEL");
glutDisplayFunc(drawScene);
glutTimerFunc(25, update, 0);
glutMainLoop();
    return 0;
}
```

Output of The Program

Evening View



Significant of The Project

This project will help to understand the graphics designing using some special methods and algorithms. For example, here some buildings, river, car bus and other things are designed using some basic concept of computer graphics. It will help for those people in future who want to work on graphics designing. They can learn basic designing using these kinds of project concept.

Conclusion

The project ‘City View In Evening’ clearly demonstrates the simulation of evening city view using OpenGL. Finally, we conclude that this program clearly illustrates the city view during evening using OpenGL and has been completed successfully and is ready to be demonstrated.

Out of the many features availablethe project demonstrates some popular and commonly used features of OpenGL such as Rendering, Transformation, Rotation, Lighting, Scaling etc. These graphic functions will also work in tandem with various rules involved in tie game. Since this project works on dynamic values, it can be used for real time computation

The project enabled with the-level OpenGL complexity and the project demonstrates the scope of OpenGL platform as a premier developing launch pad. Hence, it has indeed been useful in developing ma, games. OpenGL in own right is good for low cost and simple game development. It serves as an important stepping stone for venturing into other fields of Computer Graphics design and applications.

References

We have obtained information from many resources to design and implement our project successively. We have acquired most of the knowledge from related websites.

The following are some of the resources:

Text books:

- Foley, van Dam, Feiner, Hughes, Computer Graphics: principles and practice Addison Wesley, Second Edition.
- Schaum's Outline of Theory & Problems of Computer Graphics.
- Peter Shirley Steve Marschner, “Fundamental of computer graphics”, Third Edition.

Web References:

- http://google.com
- http://opengl.org
- http://www.wikipedia.org