UCS505 – Computer Graphics

Project Report on

TRAFFIC RACER

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Submitted to:

Kudratdeep Aulakh

BE Third Year- COE28

Submitted by:

(101803626) SHIVAM SURI

(101803628) ANSH BANGIA

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala

JUNE 2021

TABLE OF CONTENTS

PROJECT OVERVIEW

“Traffic Racer” is a 2D car game made using OpenGL. In this game, the objective is to dodge the traffic, both oncoming and ongoing.
The road is divided into two halves. The traffic on the left half moves in the direction of the player car, and the traffic on the right half moves in the opposite direction.
The player controls the car using arrow keys (left and right). The player also needs to care about fuel which gradually decreases.
For every car passed, score increases by 10. The player can get 2x score by moving to the wrong side and face oncoming traffic. The game also gives 3 lives to the player in case he collides with the traffic.

Computer Graphics Concepts Used

- Computer graphics concepts to make lines, circles, triangles, quadrilaterals, polygons have been used to draw all objects in the game such as vehicles, fuel icon, road, divider, etc.
- The concept to swap buffers has been used to achieve animation.
- 2D transformation (translation) has been used.

User Defined Functions

- main()
    - The main function is used to setup all functions and properties to be used from the glut.h library.
- ovps()
    - This function is used to initialize random values to the X coordinates of the six traffic cars.
- init()
    - This function is used to set the limits of X-axis and Y-axis.
- game()
    - This is the primary function that controls most of what is displayed in the game. First of all, it is used to further call functions that will draw road, divider, player vehicle and opponent vehicles.
    - This function is used to display current and highest scores, the current fuel and the current lives left. Game over and pause functionality is also implemented through this.
    - glutSwapBuffers() function is called through this function, which achieves animation.
- drawRoad()
    - This function is used to draw road.
- drawDivider()
    - This function is used to draw divider.
- drawVehicle()
    - This function is used to draw player vehicle.
- ellipse()
    - This is a helper function which accepts x and y coordinates, and minor and major axes lengths. It draws a semi-ellipse accordingly.
- drawOVehicle()
    - This function is used to draw the opponent vehicles and the fuel icon when fuel drops below 25%.
    - This function also controls the logic of collision between player vehicle and other vehicles and also between player vehicle and fuel icon.
- fueltank()
    - This function is used to draw fuel icon.
- ellipse_reverse()
    - This function is the same as ellipse() function, except it flips the semi-ellipse about the horizontal axis.
- drawText()
    - This function accepts a char array (string) and x and y coordinates. It displays this string at the specified location.
- drawTextNum()
    - This is similar to drawText function. It is used to display numeric values. It is used to display score.
- rectangle()
    - It is a helper function that accepts coordinates, width and height of the rectangle and draws one accordingly.

- line()
    - It is a user defined function that accepts coordinates, length and orientation (vertical or horizontal) and draws a line accordingly.
- heart()
    - This is used to draw a heart shape which is used to display the player’s lives.
- semicircle()
    - This is a helper function to draw a semi-circle at the desired location with a desired radius.
- SpecialKEy()
    - This function takes action as the left arrow key or the right arrow key is pressed. The player vehicle moves left or right accordingly.
- NormalKey()
    - This function takes action when the Escape Key is pressed to pause the game. When paused it takes action when Q key is pressed to quit the game.

CODE

```
#include<Windows.h>
#include<mmsystem.h>
#include <iostream>
#include<glut.h>
#include<stdlib.h>
using namespace std;

double pi = 3.14;
bool pause = false;
int life = 3;
int highscore = 0;
int a = 325;
int b = 4;
int speed = 10;
double fuel = 50;
bool over3 = false, over4 = false, over5 = false;
bool over = false;
int score = 0;
bool collide = false;
char buffer[10];
int vehicleX = 213, vehicleY = 10;
int x = 175;
int y = 4;
int ovehicleX[6], ovehicleY[6];
int divx = 250, divy = 4, movd;
int r = rand() % 256;
bool over2 = false;
int g = rand() % 256;
int bb = rand() % 256;
int flagg = 1, fuellX = -20, fuellY = 600;

void ellipse(double xx, double yy, double major, double minor)
{
        double x, y, i, R = major, r = minor;
        glBegin(GL_POLYGON);

        for (i = 0; i < (1 * pi); i += 0.001)
        {
                x = R * cos(i);
                y = r * sin(i);
                glVertex2d(x + xx, y + yy);
        }

        glEnd();
}

void ellipse_reverse(double xx, double yy, double major, double minor)
{
        double x, y, i, R = major, r = minor;
```

```
	glBegin(GL_POLYGON);

	for (i = pi; i < (2 * pi); i += 0.001)
	{
		x = R * cos(i);
		y = r * sin(i);
		glVertex2d(x + xx, y + yy);
	}

	glEnd();
}

void semicircle(double x, double y, double radius)
{
	float xx, yy, i;
	glBegin(GL_POLYGON);

	for (i = 0; i < (1 * pi); i += 0.001)
	{
		xx = radius * cos(i);
		yy = radius * sin(i);
		glVertex2d(x + xx, y + yy);
	}

	glEnd();
}

void heart(int x, int y)
{
	glBegin(GL_TRIANGLES);

	glColor3f(1, 0, 0);
	glVertex2f(x + 0, y + 5);
	glVertex2f(x - 10, y + 25);
	glVertex2f(x + 10, y + 25);

	glEnd();

	semicircle(x - 5, y + 25, 5);
	semicircle(x + 5, y + 25, 5);
}

void rectangle(double xcor, double ycor, double width, double height)
{
	glBegin(GL_QUADS);

	glVertex2f(xcor, ycor + height);
	glVertex2f(xcor, ycor);
	glVertex2f(xcor + width, ycor);
	glVertex2f(xcor + width, ycor + height);
```

```
        glEnd();
}

class Text
{
public:

        void drawText(const char ch[], int xpos, int ypos)
        {
                int numofchar = strlen(ch);
                glLoadIdentity();
                glRasterPos2f(xpos, ypos);
                for (int i = 0; i <= numofchar - 1; i++)
                {
                        glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, ch[i]);
                }
        }

        void drawTextNum(char ch[], int numtext, int xpos, int  ypos)
        {
                int len;
                int k;
                k = 0;
                len = numtext - strlen(ch);
                glLoadIdentity();
                glRasterPos2f(xpos, ypos);

                for (int i = 0; i <= numtext - 1; i++)
                {
                        if (i < len)
                                glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, '0');
                        else
                        {
                                glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18,
ch[k]);
                                k++;
                        }
                }
        }
};

void fueltank(int fuelX, int fuelY)
{
        glColor3f(0.25, 0.25, 0.25);
        glBegin(GL_QUADS);
        glVertex2i(fuelX, fuelY + 5);
        glVertex2i(fuelX + 3, fuelY - 5);
        glVertex2i(fuelX + 15, fuelY + 25);
        glVertex2i(fuelX + 12, fuelY + 35);
        glEnd();
        glColor3f(1, 1, 0);
        glBegin(GL_POLYGON);
        glVertex2i(fuelX - 15, fuelY + 30);
        glVertex2i(fuelX - 15, fuelY - 50);
        glVertex2i(fuelX + 15, fuelY - 50);
        glVertex2i(fuelX + 15, fuelY);
        glVertex2i(fuelX, fuelY + 10);
        glVertex2i(fuelX, fuelY + 30);
        glEnd();
        glColor3f(0.5, 0.5, 0.5);
        glBegin(GL_QUADS);
        glVertex2i(fuelX - 12, fuelY + 25);
        glVertex2i(fuelX - 12, fuelY + 15);
        glVertex2i(fuelX - 3, fuelY + 15);
        glVertex2i(fuelX - 3, fuelY + 25);
        glEnd();
        //Text fill;
        //glColor3f(0, 0, 0);
        //glColor3f(1, 1, 1);
        //fill.drawText("FUEL", fuelX-8, fuelY-20);
}

class Road {

public:

        void ovps()
        {
                glClearColor(0, 0, 1, 0);
                for (int i = 0; i < 6; i++)
                {
                        if (rand() % 5 == 0)
                        {
                                ovehicleX[i] = 213;
                        }
                        if (rand() % 5 == 1)
                        {
                                ovehicleX[i] = 138;
                        }
                        else if (rand() % 5 == 2)
                        {
                                ovehicleX[i] = 363;
                        }
                        else if (rand() % 5 == 3)
                        {
                                ovehicleX[i] = 288;
                        }
                        else
                        {
```

```
                        ovehicleX[i] = -300;
                }
                ovehicleY[i] = 1000 + i * 160;
        }
}

void drawRoad()
{
        glBegin(GL_QUADS);

        glColor3f(0.5, 0.5, 0.5);
        glVertex2f(100, 500);
        glVertex2f(100, 0);
        glVertex2f(400, 0);
        glVertex2f(400, 500);
        glEnd();

        glBegin(GL_QUADS);
        glColor3f(0, 1, 0);
        glVertex2f(400, 500);
        glVertex2f(400, 0);
        glVertex2f(500, 0);
        glVertex2f(500, 500);
        glEnd();

        glBegin(GL_QUADS);
        glColor3f(0, 1, 0);
        glVertex2f(0, 500);
        glVertex2f(0, 0);
        glVertex2f(100, 0);
        glVertex2f(100, 500);
        glEnd();
        glBegin(GL_QUADS);
        glColor3f(1, 1, 1);
        glVertex2f(100, 0);
        glVertex2f(100, 500);
        glVertex2f(98, 500);
        glVertex2f(98, 0);
        glEnd();

        glBegin(GL_QUADS);
        glColor3f(1, 1, 1);
        glVertex2f(400, 0);
        glVertex2f(400, 500);
        glVertex2f(402, 500);
        glVertex2f(402, 0);
        glEnd();
}

void drawDivider()
```

```
	{
		glLoadIdentity();
		glTranslatef(0, movd, 0);

		glColor3f(0, 0, 0);
		glBegin(GL_QUADS);
		glVertex2f(divx - 5, 600);
		glVertex2f(divx - 5, 0);
		glVertex2f(divx + 5, 0);
		glVertex2f(divx + 5, 600);
		glEnd();
		for (int i = 1; i <= 10; i++)
		{
			glColor3f(1, 1, 0);
			glBegin(GL_QUADS);
			glVertex2f(divx - 5, divy * 15 * i + 10);
			glVertex2f(divx - 5, divy * 15 * i - 10);
			glVertex2f(divx + 5, divy * 15 * i - 10);
			glVertex2f(divx + 5, divy * 15 * i + 10);
			glEnd();
		}

		for (int i = 1; i <= 10; i++)
		{
			glColor3f(0, 0, 0);
			glBegin(GL_QUADS);
			glVertex2f(x - 2.5, y * 15 * i + 18);
			glVertex2f(x - 2.5, y * 15 * i - 18);
			glVertex2f(x + 2.5, y * 15 * i - 18);
			glVertex2f(x + 2.5, y * 15 * i + 18);
			glEnd();
		}

		for (int i = 1; i <= 10; i++)
		{
			glColor3f(0, 0, 0);
			glBegin(GL_QUADS);
			glVertex2f(a - 2.5, b * 15 * i + 18);
			glVertex2f(a - 2.5, b * 15 * i - 18);
			glVertex2f(a + 2.5, b * 15 * i - 18);
			glVertex2f(a + 2.5, b * 15 * i + 18);
			glEnd();
		}

		glLoadIdentity();
	}
};

class Vehicle
{
```

```
public:

        void drawVehicle()
        {
                glBegin(GL_TRIANGLES);
                glColor3f(1, 1, 0.3);
                glVertex2f(vehicleX + 15, vehicleY + 120);
                glVertex2f(vehicleX + 22, vehicleY + 145);
                glVertex2f(vehicleX + 8, vehicleY + 145);
                glEnd();
                ellipse(vehicleX + 15, vehicleY + 145, 7, 5);

                glBegin(GL_TRIANGLES);
                glColor3f(1, 1, 0.3);
                glVertex2f(vehicleX - 15, vehicleY + 120);
                glVertex2f(vehicleX - 22, vehicleY + 145);
                glVertex2f(vehicleX - 8, vehicleY + 145);
                glEnd();
                ellipse(vehicleX - 15, vehicleY + 145, 7, 5);

                glBegin(GL_POLYGON);
                glColor3f(1, 0, 0);
                glVertex2f(vehicleX - 20, vehicleY);
                glVertex2f(vehicleX + 20, vehicleY);
                glVertex2f(vehicleX + 25, vehicleY + 30);
                glVertex2f(vehicleX + 25, vehicleY + 90);
                glVertex2f(vehicleX + 20, vehicleY + 130);
                glVertex2f(vehicleX - 20, vehicleY + 130);
                glVertex2f(vehicleX - 25, vehicleY + 90);
                glVertex2f(vehicleX - 25, vehicleY + 30);
                glEnd();

                glBegin(GL_QUADS);
                glColor3f(0, 0, 0);
                glVertex2f(vehicleX - 15, vehicleY + 80);
                glVertex2f(vehicleX + 15, vehicleY + 80);
                glVertex2f(vehicleX + 20, vehicleY + 100);
                glVertex2f(vehicleX - 20, vehicleY + 100);
                glEnd();

                glBegin(GL_QUADS);
                glColor3f(0, 0, 0);
                glVertex2f(vehicleX - 15, vehicleY + 35);
                glVertex2f(vehicleX + 15, vehicleY + 35);
                glVertex2f(vehicleX + 20, vehicleY + 25);
                glVertex2f(vehicleX - 20, vehicleY + 25);
                glEnd();

                glBegin(GL_QUADS);
                glColor3f(0, 0, 0);
```

```
            glVertex2f(vehicleX + 23, vehicleY + 30);
            glVertex2f(vehicleX + 23, vehicleY + 100);
            glVertex2f(vehicleX + 17, vehicleY + 80);
            glVertex2f(vehicleX + 17, vehicleY + 40);
            glEnd();

            glBegin(GL_QUADS);
            glColor3f(0, 0, 0);
            glVertex2f(vehicleX - 23, vehicleY + 30);
            glVertex2f(vehicleX - 23, vehicleY + 100);
            glVertex2f(vehicleX - 17, vehicleY + 80);
            glVertex2f(vehicleX - 17, vehicleY + 40);
            glEnd();
    }

    void drawOVehicle()
    {
            fueltank(fuellX, fuellY);
            for (int i = 0; i < 6; i++)
            {
                    if (ovehicleX[i] > 250)
                    {
                            glBegin(GL_TRIANGLES);
                            glColor3f(1, 1, 0.3);
                            glVertex2f(ovehicleX[i] + 15, ovehicleY[i] + 10);
                            glVertex2f(ovehicleX[i] + 22, ovehicleY[i] - 15);
                            glVertex2f(ovehicleX[i] + 8, ovehicleY[i] - 15);
                            glEnd();
                            ellipse_reverse(ovehicleX[i] + 15, ovehicleY[i] - 15, 7, 5);

                            glBegin(GL_TRIANGLES);
                            glColor3f(1, 1, 0.3);
                            glVertex2f(ovehicleX[i] - 15, ovehicleY[i] + 10);
                            glVertex2f(ovehicleX[i] - 22, ovehicleY[i] - 15);
                            glVertex2f(ovehicleX[i] - 8, ovehicleY[i] - 15);
                            glEnd();
                            ellipse_reverse(ovehicleX[i] - 15, ovehicleY[i] - 15, 7, 5);

                            glBegin(GL_POLYGON);
                            glColor3f(r / 256.0 * i + 0.4, g / 256.0 * i + 0.2, bb / 256.0 * i +
0.7);
                            glVertex2f(ovehicleX[i] - 20, ovehicleY[i]);
                            glVertex2f(ovehicleX[i] + 20, ovehicleY[i]);
                            glVertex2f(ovehicleX[i] + 25, ovehicleY[i] + 40);
                            glVertex2f(ovehicleX[i] + 25, ovehicleY[i] + 100);
                            glVertex2f(ovehicleX[i] + 20, ovehicleY[i] + 130);
                            glVertex2f(ovehicleX[i] - 20, ovehicleY[i] + 130);
                            glVertex2f(ovehicleX[i] - 25, ovehicleY[i] + 100);
                            glVertex2f(ovehicleX[i] - 25, ovehicleY[i] + 40);
                            glEnd();
```

```
		glBegin(GL_QUADS);
		glColor3f(0, 0, 0);
		glVertex2f(ovehicleX[i] - 15, ovehicleY[i] + 95);
		glVertex2f(ovehicleX[i] + 15, ovehicleY[i] + 95);
		glVertex2f(ovehicleX[i] + 20, ovehicleY[i] + 105);
		glVertex2f(ovehicleX[i] - 20, ovehicleY[i] + 105);
		glEnd();

		glBegin(GL_QUADS);
		glColor3f(0, 0, 0);
		glVertex2f(ovehicleX[i] - 15, ovehicleY[i] + 50);
		glVertex2f(ovehicleX[i] + 15, ovehicleY[i] + 50);
		glVertex2f(ovehicleX[i] + 20, ovehicleY[i] + 30);
		glVertex2f(ovehicleX[i] - 20, ovehicleY[i] + 30);
		glEnd();

		glBegin(GL_QUADS);
		glColor3f(0, 0, 0);
		glVertex2f(ovehicleX[i] + 23, ovehicleY[i] + 30);
		glVertex2f(ovehicleX[i] + 23, ovehicleY[i] + 100);
		glVertex2f(ovehicleX[i] + 17, ovehicleY[i] + 90);
		glVertex2f(ovehicleX[i] + 17, ovehicleY[i] + 50);
		glEnd();

		glBegin(GL_QUADS);
		glColor3f(0, 0, 0);
		glVertex2f(ovehicleX[i] - 23, ovehicleY[i] + 30);
		glVertex2f(ovehicleX[i] - 23, ovehicleY[i] + 100);
		glVertex2f(ovehicleX[i] - 17, ovehicleY[i] + 90);
		glVertex2f(ovehicleX[i] - 17, ovehicleY[i] + 50);
		glEnd();
	}
	else
	{
		glBegin(GL_TRIANGLES);
		glColor3f(1, 1, 0.3);
		glVertex2f(ovehicleX[i] + 15, ovehicleY[i] + 120);
		glVertex2f(ovehicleX[i] + 22, ovehicleY[i] + 145);
		glVertex2f(ovehicleX[i] + 8, ovehicleY[i] + 145);
		glEnd();
		ellipse(ovehicleX[i] + 15, ovehicleY[i] + 145, 7, 5);

		glBegin(GL_TRIANGLES);
		glColor3f(1, 1, 0.3);
		glVertex2f(ovehicleX[i] - 15, ovehicleY[i] + 120);
		glVertex2f(ovehicleX[i] - 22, ovehicleY[i] + 145);
		glVertex2f(ovehicleX[i] - 8, ovehicleY[i] + 145);
		glEnd();
		ellipse(ovehicleX[i] - 15, ovehicleY[i] + 145, 7, 5);
```

```
					glBegin(GL_POLYGON);
					glColor3f(r / 256.0 * i + 0.4, g / 256.0 * i + 0.2, bb / 256.0 * i +
0.7);
					glVertex2f(ovehicleX[i] - 20, ovehicleY[i]);
					glVertex2f(ovehicleX[i] + 20, ovehicleY[i]);
					glVertex2f(ovehicleX[i] + 25, ovehicleY[i] + 30);
					glVertex2f(ovehicleX[i] + 25, ovehicleY[i] + 90);
					glVertex2f(ovehicleX[i] + 20, ovehicleY[i] + 130);
					glVertex2f(ovehicleX[i] - 20, ovehicleY[i] + 130);
					glVertex2f(ovehicleX[i] - 25, ovehicleY[i] + 90);
					glVertex2f(ovehicleX[i] - 25, ovehicleY[i] + 30);
					glEnd();

					glBegin(GL_QUADS);
					glColor3f(0, 0, 0);
					glVertex2f(ovehicleX[i] - 15, ovehicleY[i] + 80);
					glVertex2f(ovehicleX[i] + 15, ovehicleY[i] + 80);
					glVertex2f(ovehicleX[i] + 20, ovehicleY[i] + 100);
					glVertex2f(ovehicleX[i] - 20, ovehicleY[i] + 100);
					glEnd();

					glBegin(GL_QUADS);
					glColor3f(0, 0, 0);
					glVertex2f(ovehicleX[i] - 15, ovehicleY[i] + 35);
					glVertex2f(ovehicleX[i] + 15, ovehicleY[i] + 35);
					glVertex2f(ovehicleX[i] + 20, ovehicleY[i] + 25);
					glVertex2f(ovehicleX[i] - 20, ovehicleY[i] + 25);
					glEnd();

					glBegin(GL_QUADS);
					glColor3f(0, 0, 0);
					glVertex2f(ovehicleX[i] + 23, ovehicleY[i] + 30);
					glVertex2f(ovehicleX[i] + 23, ovehicleY[i] + 100);
					glVertex2f(ovehicleX[i] + 17, ovehicleY[i] + 80);
					glVertex2f(ovehicleX[i] + 17, ovehicleY[i] + 40);
					glEnd();

					glBegin(GL_QUADS);
					glColor3f(0, 0, 0);
					glVertex2f(ovehicleX[i] - 23, ovehicleY[i] + 30);
					glVertex2f(ovehicleX[i] - 23, ovehicleY[i] + 100);
					glVertex2f(ovehicleX[i] - 17, ovehicleY[i] + 80);
					glVertex2f(ovehicleX[i] - 17, ovehicleY[i] + 40);
					glEnd();
				}

				if (pause == false)
				{
					if (ovehicleX[i] > 250)
```

```
                {
                        ovehicleY[i] = ovehicleY[i] - speed - 10;
                }
                else
                {
                        ovehicleY[i] = ovehicleY[i] - speed;
                }
                if (flagg == 0)
                {
                        fuellY = fuellY - (speed + 5) / 6;
                }

                fuel -= 0.01;
        }

        if (fuel < 0)
        {
                collide = true;
                over = true;
        }

        if (ovehicleY[i] > vehicleY - 60 && ovehicleY[i] < vehicleY + 120
&& ovehicleX[i] == vehicleX)
        {
                ovehicleY[i] = 1200 + i * 160;
                life--;
        }

        if (fuellY > vehicleY - 120 && fuellY < vehicleY + 200 && fuellX ==
vehicleX)
        {
                fuellX = -300;
                fuel = 50;
        }

        if (life == 0)
        {
                collide = true;
                over = true;
                fuellY = 1200;
        }

        if (((ovehicleY[i] <= 10) && (ovehicleY[i] >= 0)) && (ovehicleX[i]
!= -300))
        {
                if (vehicleX > 250)
                {
                        if (ovehicleX[i] > 250)
                        {
                                score += 20;
                }
                else
                {
                        score += 10;
                }
        }
        else
        {
                if (ovehicleX[i] > 250)
                {
                        score += 10;
                }
                else
                {
                        score += 5;
                }
        }
}

if (ovehicleY[i] < -130)
{
        if (rand() % 5 == 0)
        {
                ovehicleX[i] = 213;
        }
        else if (rand() % 5 == 1)
        {
                ovehicleX[i] = 138;
        }
        else if (rand() % 5 == 2)
        {
                ovehicleX[i] = 363;
        }
        else if (rand() % 5 == 3)
        {
                ovehicleX[i] = 288;
        }
        else
        {
                ovehicleX[i] = -300;
        }
        ovehicleY[i] = 800 + i * 200;
        for (int j = 0; j < 6; j++)
        {
                if (i != j && ovehicleX[i]==ovehicleX[j])
                {
                        if (abs(ovehicleY[i] - ovehicleY[j]) <= 140) {
                                ovehicleX[i] = -300;
                                //cout << "Handled\n";
                        }
```

```
                                                }
                                        }
                                }

                                if (fuel < 12 && flagg == 1)
                                {
                                        int g = rand();
                                        if (g % 4 == 0)
                                        {
                                                fuellX = 213;
                                        }
                                        else if (g % 4 == 1)
                                        {
                                                fuellX = 138;
                                        }
                                        else if (g % 4 == 2)
                                        {
                                                fuellX = 363;
                                        }
                                        else if (g % 4 == 3)
                                        {
                                                fuellX = 288;
                                        }
                                        flagg = 0;
                                }
                                if (fuellY < -250)
                                {
                                        flagg = 1;
                                        fuellY = 600;
                                }
                        }
                }
};

void SpecialKEy(int key, int x, int y)
{
        switch (key)
        {
        case GLUT_KEY_LEFT:
                if ((over == false) && (over2 == false) && (over3 == false) && (over4 ==
false) && (over5 == false))
                {
                        if (pause == false)
                        {
                                if (vehicleX > 138)
                                {
                                        vehicleX = vehicleX - 75;
                                }
                        }
                }
```

```
			break;
		case GLUT_KEY_RIGHT:
			if ((over == false) && (over2 == false) && (over3 == false) && (over4 ==
false) && (over5 == false))
			{
				if (pause == false)
				{
					if (vehicleX < 363)
					{
						vehicleX = vehicleX + 75;
					}
				}
			}
			break;
		}

		glutPostRedisplay();
}

void init()
{
		glMatrixMode(GL_PROJECTION);
		glLoadIdentity();
		gluOrtho2D(0, 500, 0, 500);
		glMatrixMode(GL_MODELVIEW);
}

void NormalKey(unsigned char key, int x, int y)
{
		switch (key)
		{
		case 113: // Q Key
			if (pause == true)
			{
				exit(0);
			}
			break;
		case 27: // Escape Key
			pause = !pause;
			break;
		}
}

void line(double x, double y, double l, int a)
{
		glBegin(GL_LINES);
		glVertex2f(x, y);
		if (a == 0)
		{
			glVertex2f(x + l, y);
```

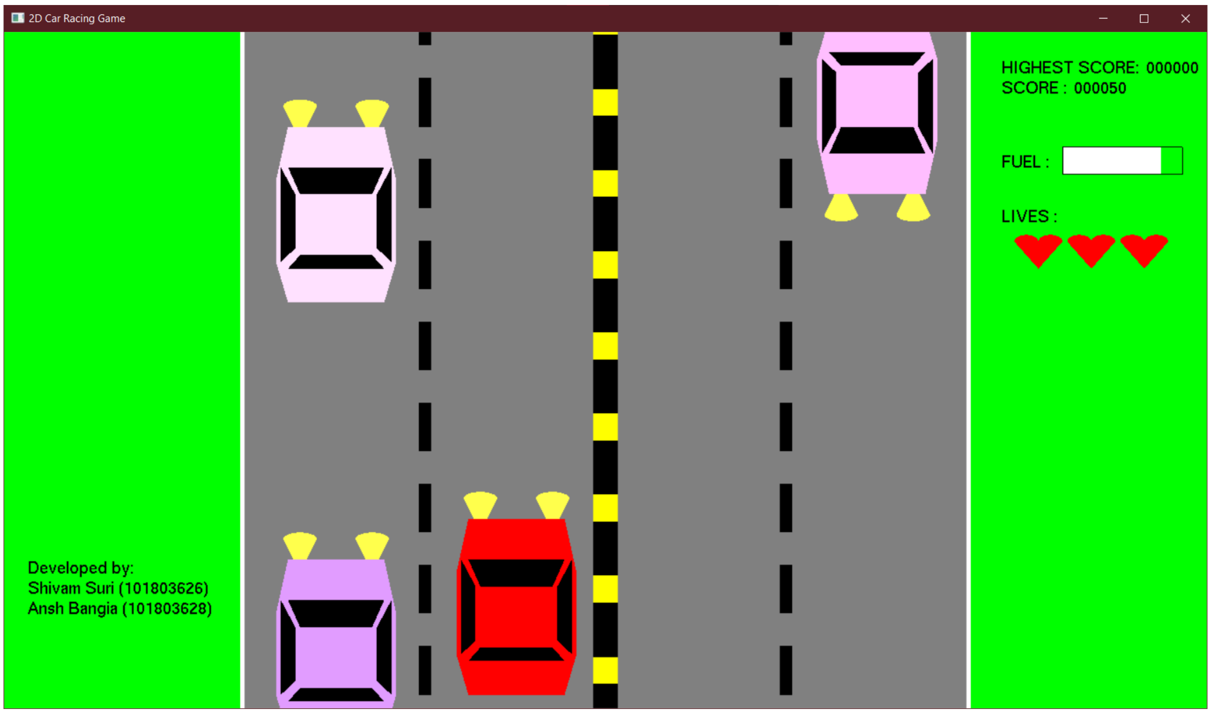
```
        }
        else
        {
                glVertex2f(x, y + l);
        }
        glEnd();
}

void game()
{
        Road a;
        Vehicle b;
        Text c;

        glClear(GL_COLOR_BUFFER_BIT);

        a.drawRoad();
        a.drawDivider();
        b.drawVehicle();
        b.drawOVehicle();

        if ((over == false) && (over2 == false) && (over3 == false) && (over4 == false) &&
(over5 == false))
        {
                if (pause == false)
                {
                        movd = movd - 16;
                        if (movd < -60)
                        {
                                movd = 0;
                        }
                }
        }

        if (pause == true)
        {
                glColor3f(0, 0, 0);
                c.drawText("PAUSE", 15, 420);
                c.drawText("Press Q to quit.", 15, 405);
        }
        if (over == true)
        {
                glColor3f(0, 0, 0);
                c.drawText("GAME OVER!", 15, 420);
                c.drawText("Game Starting in 3 . .", 15, 400);
                vehicleX = 213;
        }
        if (over2 == true)
        {
                glColor3f(0, 0, 0);
```

```
		c.drawText("Game Starting in 3 . .", 15, 400);
		glColor3f(0.5, 0.5, 0.5);
		vehicleX = -300;
	}
	if (over3 == true)
	{
		glColor3f(0, 0, 0);
		c.drawText("GAME OVER!", 15, 420);
		c.drawText("Game Starting in 3 2 .", 15, 400);
		vehicleX = 213;
	}
	if (over4 == true)
	{
		glColor3f(0, 0, 0);
		c.drawText("Game Starting in 3 2 .", 15, 400);
		glColor3f(0.5, 0.5, 0.5);
		vehicleX = -300;
	}
	if (over5 == true)
	{
		glColor3f(0, 0, 0);
		vehicleX = 213;
		c.drawText("GAME OVER!", 15, 420);
		c.drawText("Game Starting in 3 2 1", 15, 400);
	}

	glColor3f(0, 0, 0);
	c.drawText("Developed by:", 10, 100);
	c.drawText("Shivam Suri (101803626)", 10, 85);
	c.drawText("Ansh Bangia (101803628)", 10, 70);

	c.drawText("HIGHEST SCORE: ", 415, 470);
	_itoa_s(highscore, buffer, 10);
	c.drawTextNum(buffer, 6, 475, 470);
	c.drawText("SCORE :", 415, 455);
	_itoa_s(score, buffer, 10);
	c.drawTextNum(buffer, 6, 445, 455);

	if (vehicleX > 250)
	{
		glColor3f(1, 0, 0);
	}
	else
	{
		glColor3f(0, 1, 0);
	}
	c.drawText("CAUTION! WRONG SIDE!", 415, 440);
	c.drawText("2X SCORE", 415, 425);

	glColor3f(0, 0, 0);
```

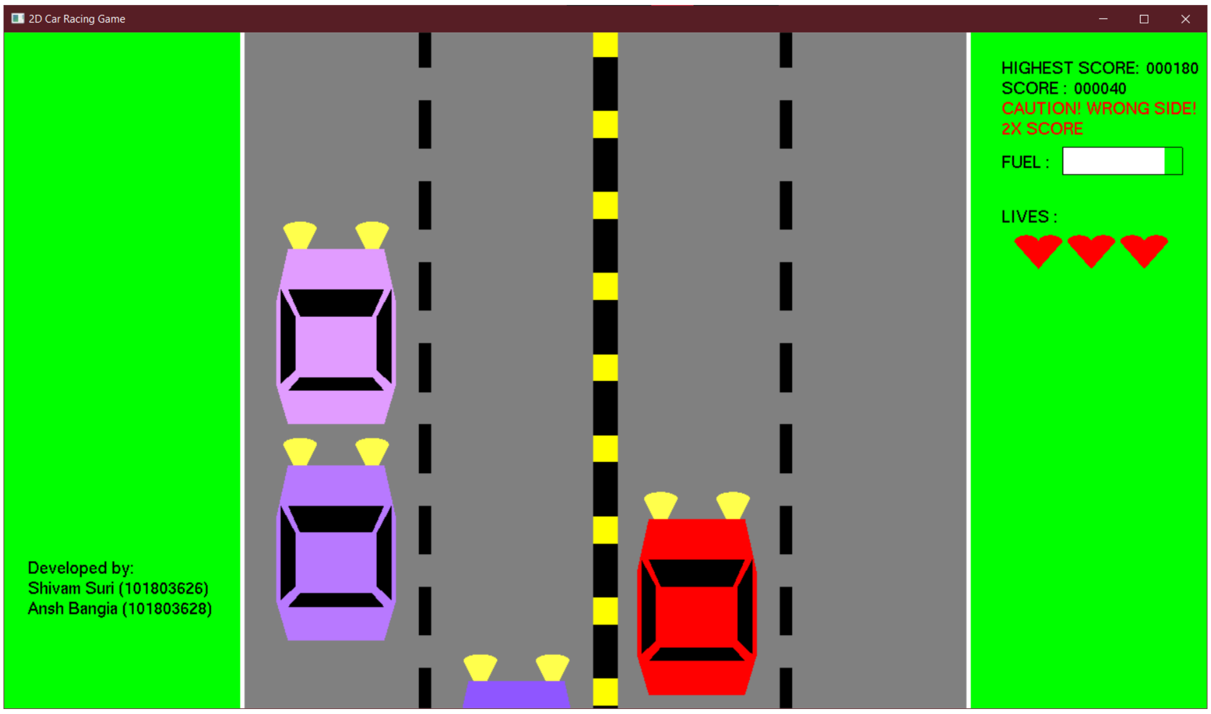
```
c.drawText("FUEL : ", 415, 400);
glColor3f(1, 1, 1);
if (fuel <= 12)
{
        glColor3f(1, 0, 0);
}
rectangle(440, 395, fuel, 20);
glColor3f(0, 0, 0);
line(440, 395, 50, 0);
line(440, 395, 20, 1);
line(440, 415, 50, 0);
line(490, 395, 20, 1);

c.drawText("LIVES : ", 415, 360);
for (int i = 0; i < life; i++)
{
        heart(430 + i * 22, 320);
}

glutSwapBuffers();

Sleep(15);

if (collide == true)
{
        life = 3;
        fuel = 50;
        vehicleX = 213;
        for (int i = 0; i < 6; i++)
        {
                ovehicleY[i] = 1000 + i * 160;
        }
        highscore = max(highscore, score);
        score = 0;
        flagg = 1;
        fuellX = -20;
        fuellY = 600;
        collide = false;
}

if (over5 == true)
{
        Sleep(500);
        over5 = false;
}
if (over4 == true)
{
        Sleep(500);
        over4 = false;
        over5 = true;
```

```
	}
	if (over3 == true)
	{
		Sleep(500);
		over3 = false;
		over4 = true;
	}
	if (over2 == true)
	{
		Sleep(500);
		over2 = false;
		over3 = true;
	}
	if (over == true)
	{
		Sleep(1000);
		over = false;
		over2 = true;
	}
}

void main(int argc, char** argv)
{
	Road f;
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
	glutInitWindowPosition(0, 0);
	glutInitWindowSize(1366, 768);
	glutCreateWindow("2D Car Racing Game");
	f.ovps();
	init();

	glutDisplayFunc(game);
	glutSpecialFunc(SpecialKEy);
	glutKeyboardFunc(NormalKey);
	glutIdleFunc(game);

	glutMainLoop();
}
```

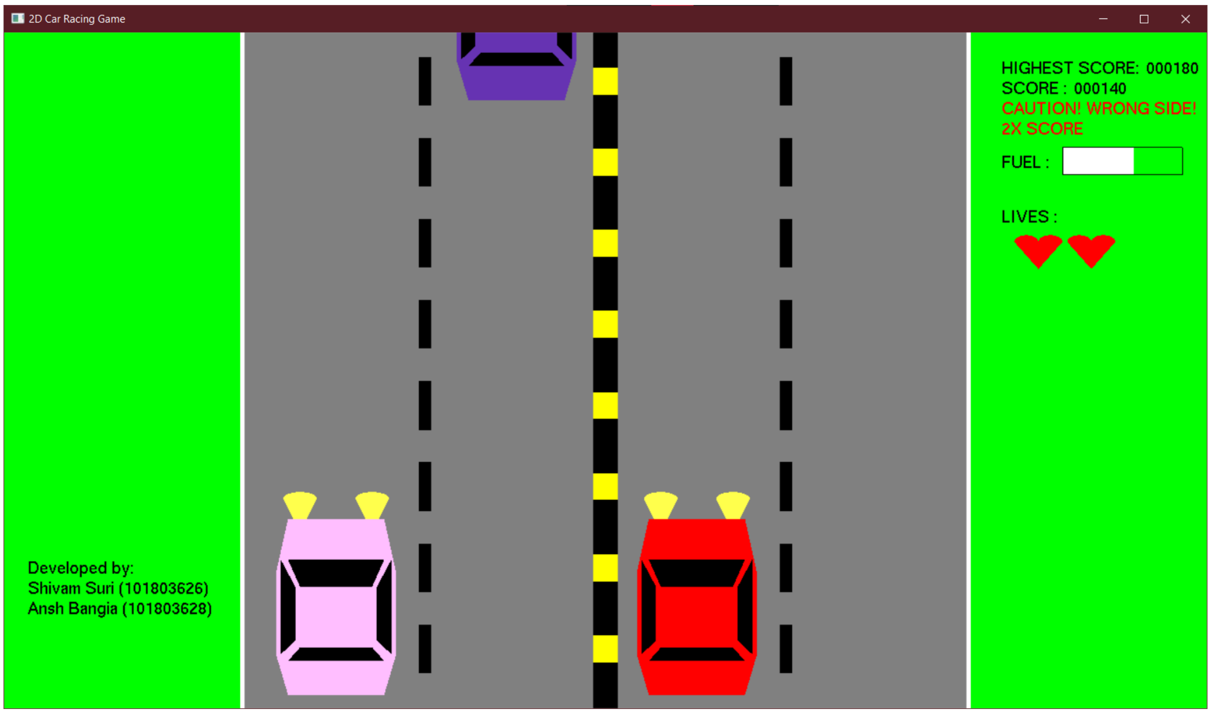
GAMEPLAY SCREENSHOTS



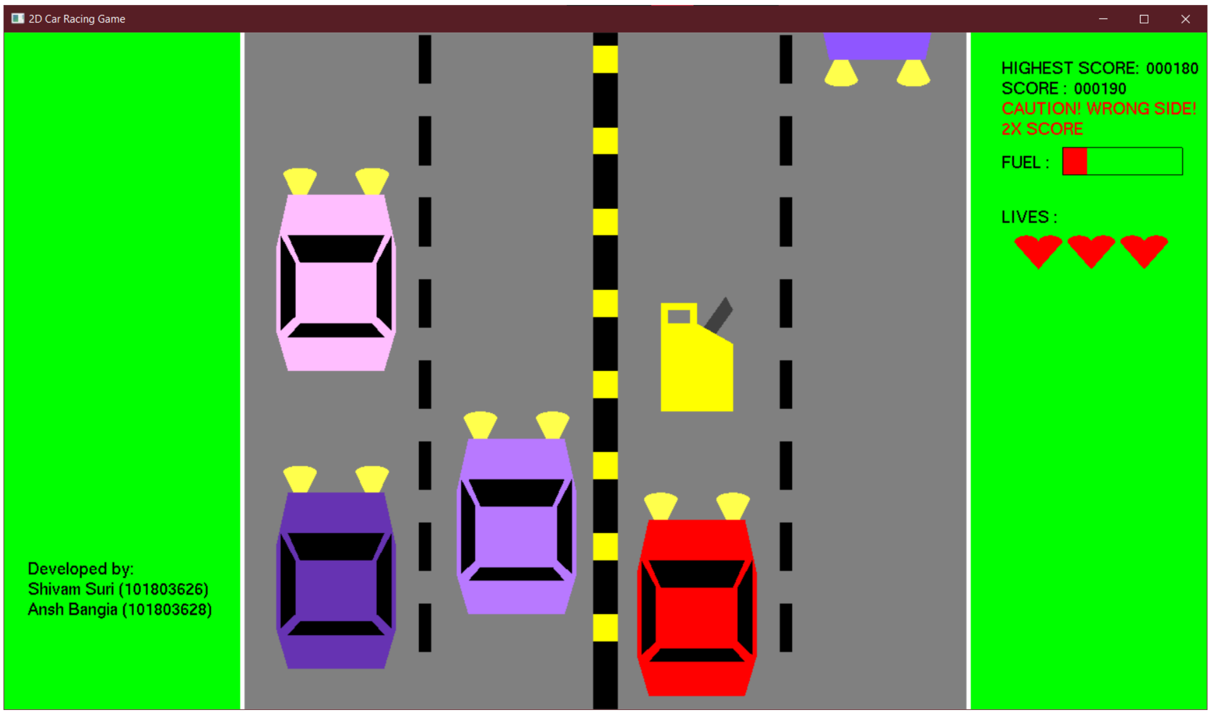
This is how the game starts. Default start location is to the left of the divider. For every car you pass, you get 10 points.



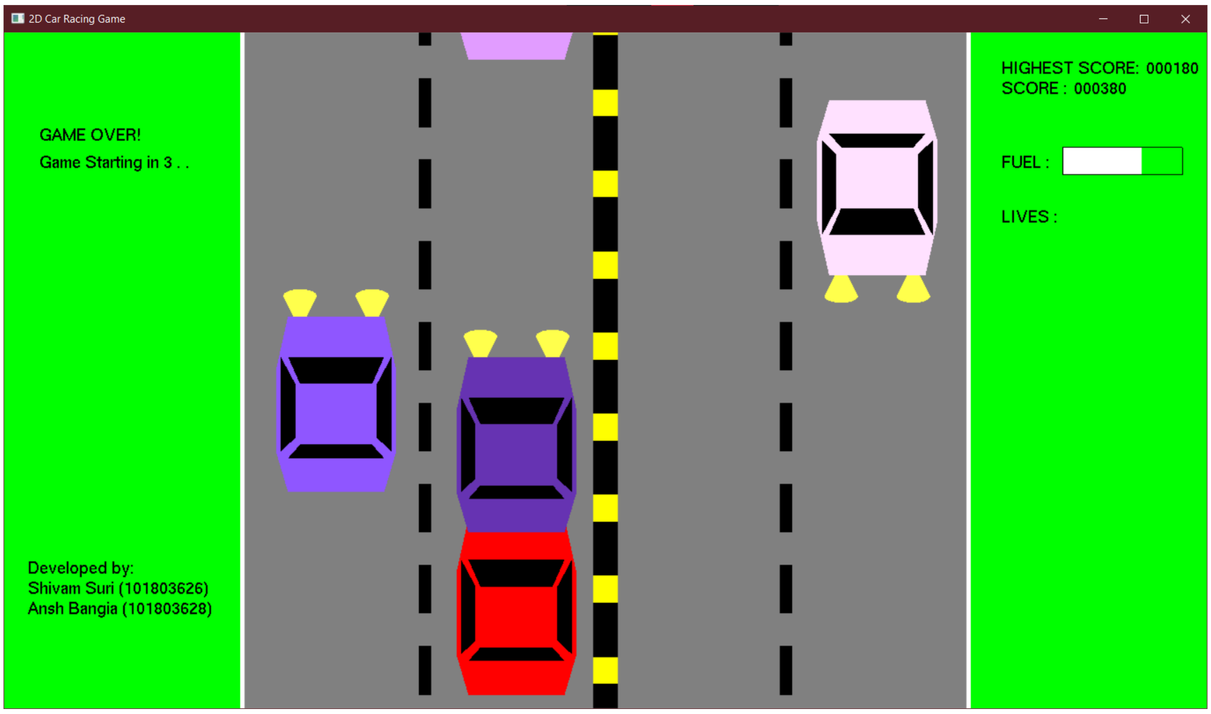
If you drive on the wrong side, you get a warning. You will also score double point i.e. 20 for every car you pass. Notice how the fuel decreases as the game progresses.



If you collide with a car, you will lose a life. Notice one less heart under “LIVES”.



If the fuel becomes less than 25%, the meter becomes red and fuel icon appears amongst the traffic. You can collect the icon to refuel your car and continue with the game.



GAME OVER!
Game Starting in 3 . .

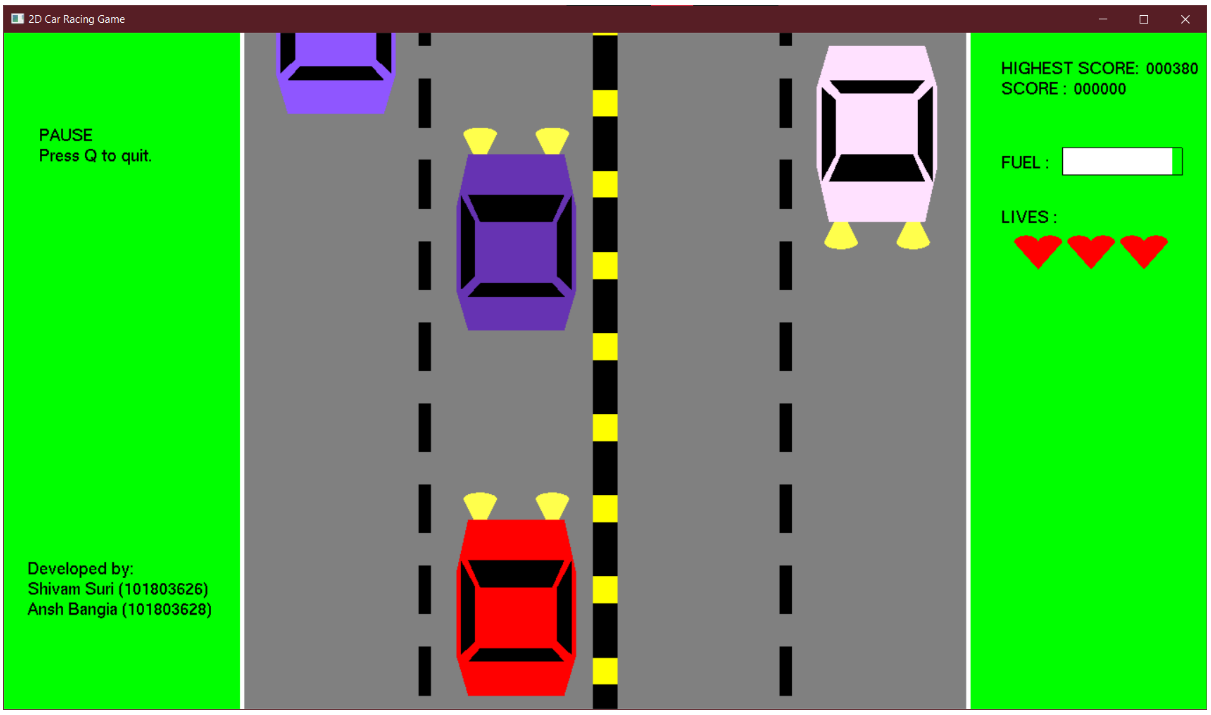
Developed by:
Shivam Suri (101803626)
Ansh Bangia (101803628)

HIGHEST SCORE: 000180
SCORE : 000380

FUEL :

LIVES :

If your lives reduce to zero or your fuel is exhausted, the GAME OVER screen appears, and the game restarts from 0 score after a countdown from 3.



PAUSE
Press Q to quit.

Developed by:
Shivam Suri (101803626)
Ansh Bangia (101803628)

HIGHEST SCORE: 000380
SCORE : 000000

FUEL :

LIVES :

This is the pause screen which shows when you press Escape Key. The traffic stops and you are not allowed to move your vehicle left or right. When paused, you may press Q to quit. Notice the HIGH SCORE is updated if you break your record.