Dodge Traffic

Made By:

Md. Samiul Islam

Ayesha Sultana Orna

Nusrat Jahan

Department: Computer Science and Engineering

United International University

Introduction:

Dodge traffic is a game made with OpenGL and Glut. Here there is a main car which moves Left and Right avoiding the traffic coming towards the Car.

Features Explanation:

In the main menu there are total 4 options. Start, Select Car, High Score and Exit. If pressed “Start” the game will start, if pressed “Select Car” there is going to be options to select certain car skin. In “High Score” all the high scores are stored and if pressed “Exit” the game will be over.

Game Functionalities:

Left Mouse Button-> Select options from the Select Car screen.

A-> Move the Car Left.

D-> Move the Car Right.

Code explanation:

1.

```
12    float _move1 = 0;
13    float _move2 = 0;
14    float _move3 = 0;
15    float _move4 = 0;
16    int driverCarPos = 0;
17    float speed = 0.02;
18    int carPos = 1;
19    int carCreate1 =0,carCreate2 = 0,carCreate3 =0;
20    int carOpPos[] = {0,0,0,0,0,0,0,0};
21    float _opCar[] = {0,0,0,0,0,0,0,0};
22    int nOp = 7;
23    int screen = 0;
24    char start[] = "Start";
25    char hScore[] = "HighScore";
26    char exitGame[] = "Exit";
27    char scoreText[] = "Score:";
28    char mphText[] = "MPH:";
29    char levelText[] = "Level:";
30    char gameOverText[] = "GAME OVER!!!!!!!!";
31    char mainMenuText[] = "Main Menu";
32    char carsText[] = "Select Car";
33    int score = 0;
34    char buffer[10];
35    bool collide = false;
36    int highScore[10] = {0,0,0,0,0,0,0,0,0,0};
37    int prevOpPos =0;
38    int timer = 25;
39    int mph = 50;
40    int level = 0;
41    int carModel = 2;
42    int polLights = 1;
43    int lightzzz = 0;
```

All the global variables are declared here. Every characters. Here some of the values are initialized. Speed is pre-defined from the beginning.

2.

```
45  void Sprint( float x, float y, char *st)
46  {
47      int l,i;
48
49      l=strlen( st );
50      glColor3f(0.0,1.0,0.7);
51      glRasterPos2f( x, y); // location to start printing text
52      for( i=0; i < l; i++) // loop until i is greater then l
53      {
54          glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24, st[i]);
55      }
56  }
```

Every text which are printed in the screen are printed with this function. It takes the x, y axis and the char value and prints the text.

3.

```
57  void writeHighScore()
58  {
59      ofstream scoreFile;
60      scoreFile.open("score.txt");
61      for(int i = 0; i < 10; i++)
62      {
63          scoreFile << highScore[i] << endl;
64      }
65      scoreFile.close();
66  }
67  void readHighScore()
68  {
69      ifstream scoreFile;
70      scoreFile.open("score.txt");
71      if(scoreFile.fail())
72      {
73          cerr << "Error: Opening score File." <<endl;
74          exit(1);
75      }
76      for(int i = 0; i < 10; i++)
77      {
78          scoreFile >> highScore[i];
79      }
80      scoreFile.close();
81  }
```

High score are shown using the file read/write method.

4.

```
83    void writeCarModel()
84    {
85        ofstream carFile;
86        carFile.open("carModel.txt");
87        carFile << carModel;
88        carFile.close();
89    }
90
91    void readCarModel()
92    {
93        ifstream carFile;
94        carFile.open("carModel.txt");
95        if(carFile.fail())
96        {
97            carModel = 1;
98        }
99        carFile >> carModel;
100       carFile.close();
101   }
```

Car models are also stored using the file read/write method.

5.

```
122   void SprintScore(char ch[],int numtext,float x, float y)//counting the score
123   {
124       int len;
125       int k,i;
126       k = 0;
127       len = numtext - strlen (ch);
128       glLoadIdentity ();
129       glRasterPos2f( x , y);
130       for (i = 0; i <=numtext - 1; i++)
131       {
132           if ( i < len )
133               glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18,'0');
134           else
135           {
136               glutBitmapCharacter(GLUT_BITMAP_HELVETICA_18, ch[k]);
137               k++;
138           }
139       }
140   }
```

Scores shown in the screen in the game is done with this function.

6.

```
142  void keyboard(unsigned char key, int x, int y)
143  {
144      switch(key)
145      {
146          case 27:
147              if(screen == 2)
148                  screen = 0;
149              else if(screen == 4)
150                  screen = 0;
151          case 'e':
152
153              break;
154
155          case 'a':
156              if(carPos != 1)
157                  carPos--;
158              break;
159          case 'd':
160              if(carPos != 3)
161                  carPos++;
162              break;
163      }
164  }
```

Keyboard Functionalities are here as stated above.

7.

```
166  void mouse(int button, int state, int x, int y)
167  {
168      cout << x << " " << y << endl;
169      if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
170      {
171          if(screen == 0)
172          {
173              if(x > 275 && x < 525 && y > 180 && y < 260)
174              {
175                  screen = 1;
176                  for(int i = 1;i <= nOp;i++)
177                  {
178                      _opCar[i] = 0;
179                  }
180
181                  for(int i = 1;i <= nOp;i++)
182                  {
183                     carOpPos[i] = 0;
184                  }
185
186
187                  float diff = 0;
188                  for(int i = 1;i <= nOp;i++)
189                  {
190                      _opCar[i] += diff;
191                      diff += .7;
192                  }
193                  speed = 0.02;
194                  score = 0;
195                  mph = 50;
196                  level = 1;
197              }
198              else if(x > 275 && x < 525 && y > 300 && y < 380)
199              {
```

```
                screen = 4;
            }
            else if(x > 275 && x < 525 && y > 420 && y < 500)
            {
                readHighScore();
                screen = 2;
            }
            else if(x > 275 && x < 525 && y > 540 && y < 620)
            {
                exit(1);
            }
        }
        else if(screen == 3)
        {
            if(x > 280 && x < 517 && y > 360 && y < 440) // main menu
            {
                screen = 0;
            }
            else if(x > 280 && x < 517 && y > 480 && y < 560) // exit
            {
                exit(1);
            }
        }
        else if(screen == 4)
        {
            if(x > 160 && x < 240 && y > 370 && y < 460)
            {
                carModel = 1;
                writeCarModel();
                screen = 0;
```

```
            }
            else if(x > 360 && x < 440 && y > 370 && y < 460)
            {
                carModel = 2;
                writeCarModel();
                screen = 0;
            }
            else if(x > 560 && x < 640 && y > 370 && y < 460)
            {
                carModel = 3;
                writeCarModel();
                screen = 0;
            }
        }
    }
}
```

Here all the mouse functionalities are done as we have stated above.

8.

```
void drawRoad()
{
    glPushMatrix();

        glColor3ub(0,255,0);
        glBegin(GL_QUADS);
            glVertex3f(-1,-1,0);
            glVertex3f(-1,1,0);
            glVertex3f(1,1,0);
            glVertex3f(1,-1,0);
        glEnd();

        glColor3ub(128,128,128);
        glBegin(GL_QUADS);
            glVertex3f(-.70,-1,0);
            glVertex3f(-.70,1,0);
            glVertex3f(.70,1,0);
            glVertex3f(.70,-1,0);
        glEnd();
            //white coloured strip left
            glColor3ub(255,255,255);
            glPushMatrix();
                glTranslatef(0,_move1,0);
                glBegin(GL_QUADS);
                    glVertex3f(-.30,1.70,0);
                    glVertex3f(-.30,1.10,0);
                    glVertex3f(-.20,1.10,0);
                    glVertex3f(-.20,1.70,0);
                glEnd();
            glPopMatrix();
            glPushMatrix();
                glTranslatef(0,_move2,0);
                glBegin(GL_QUADS);
                    glVertex3f(-.30,1,0);
                    glVertex3f(-.30,.40,0);
                    glVertex3f(-.20,.40,0);
                    glVertex3f(-.20,1,0);
                glEnd();
            glPopMatrix();

            glPushMatrix();
                glTranslatef(0,_move3,0);
                glBegin(GL_QUADS);
                    glVertex3f(-.30,.30,0);
                    glVertex3f(-.30,-.3,0);
                    glVertex3f(-.20,-.3,0);
                    glVertex3f(-.20,.30,0);
                glEnd();
            glPopMatrix();

            glPushMatrix();
                glTranslatef(0,_move4,0);
                glBegin(GL_QUADS);
                    glVertex3f(-.30,-.40,0);
                    glVertex3f(-.30,-1,0);
                    glVertex3f(-.20,-1,0);
                    glVertex3f(-.20,-.40,0);
                glEnd();
            glPopMatrix();

            //Right white
            glPushMatrix();
                glTranslatef(0,_move1,0);
```

```
311                 glBegin(GL_QUADS);
312                     glVertex3f(.30,1.70,0);
313                     glVertex3f(.30,1.10,0);
314                     glVertex3f(.20,1.10,0);
315                     glVertex3f(.20,1.70,0);
316                 glEnd();
317             glPopMatrix();
318
319             glPushMatrix();
320                 glTranslatef(0,_move2,0);
321                 glBegin(GL_QUADS);
322                     glVertex3f(.30,1,0);
323                     glVertex3f(.30,.40,0);
324                     glVertex3f(.20,.40,0);
325                     glVertex3f(.20,1,0);
326                 glEnd();
327             glPopMatrix();
328
329             glPushMatrix();
330                 glTranslatef(0,_move3,0);
331                 glBegin(GL_QUADS);
332                     glVertex3f(.30,.30,0);
333                     glVertex3f(.30,-.3,0);
334                     glVertex3f(.20,-.3,0);
335                     glVertex3f(.20,.30,0);
336                 glEnd();
337             glPopMatrix();
338
339             glPushMatrix();
340                 glTranslatef(0,_move4,0);
341                 glBegin(GL_QUADS);
342                     glVertex3f(.30,-.40,0);
343                     glVertex3f(.30,-1,0);
344                     glVertex3f(.20,-1,0);
345                     glVertex3f(.20,-.40,0);
346                 glEnd();
347             glPopMatrix();
348
349     glPopMatrix();
350 }
```

The full Road Drawing part is done over here. Main moving effect of the Road is given white stripes part. In the glTranslate() function the Y-axis part is given the move function which moves the road downwards.

9.

```
void drawDriver()
{
    if(carPos == 2)
    {
        glTranslatef(.53,0,0);
    }
    else if(carPos == 3)
    {
        glTranslatef(1.05,0,0);
    }
    else if(carPos == 1)
    {
        glTranslatef(0,0,0);
    }

    if(carModel == 1)
    {
        glPushMatrix();
            glColor3ub(255,0,0);
            glBegin(GL_POLYGON);
                glVertex3f(-.55,-.7,0);
                glVertex3f(-.6,-.75,0);
                glVertex3f(-.6,-.85,0);
                glVertex3f(-.55,-.9,0);
                glVertex3f(-.5,-.9,0);
                glVertex3f(-.45,-.85,0);
                glVertex3f(-.45,-.75,0);
                glVertex3f(-.5,-.7,0);
            glEnd();

            glColor3ub(0,0,255);
            glBegin(GL_POLYGON);
                glVertex3f(-.6,-.75,0);
                glVertex3f(-.6,-.85,0);
                glVertex3f(-.45,-.85,0);
                glVertex3f(-.45,-.75,0);
            glEnd();

            glColor3ub(0,0,255);
            glBegin(GL_POLYGON);

                glVertex3f(-.63,-.88,0);
                glVertex3f(-.63,-.9,0);
                glVertex3f(-.42,-.9,0);
                glVertex3f(-.42,-.88,0);

            glEnd();

            glColor3ub(0,0,0);
            glPointSize(10);
            glBegin(GL_POINTS);
                glVertex3f(-.6,-.75,0);
                glVertex3f(-.6,-.85,0);
                glVertex3f(-.45,-.85,0);
                glVertex3f(-.45,-.75,0);
            glEnd();

        glPopMatrix();
    }else if(carModel == 2)
    {
        glColor3ub(0,0,255);
            glBegin(GL_POLYGON);
                glVertex3f(-.6,-.7,0);
                glVertex3f(-.6,-.9,0);
                glVertex3f(-.45,-.9,0);
                glVertex3f(-.45,-.7,0);
            glEnd();

            glColor3ub(255,255,255);
            glBegin(GL_POLYGON);
                glVertex3f(-.58,-.77,0);
                glVertex3f(-.58,-.87,0);
                glVertex3f(-.47,-.87,0);
                glVertex3f(-.47,-.77,0);
            glEnd();

            glColor3ub(0,0,0);
            glPointSize(10);
```

```
            glBegin(GL_POINTS);
                glVertex3f(-.605,-.75,0);
                glVertex3f(-.605,-.85,0);
                glVertex3f(-.445,-.85,0);
                glVertex3f(-.445,-.75,0);
            glEnd();

            if(polLights == 1)
            {
                glColor3ub(255,0,0);
                glBegin(GL_POINTS);
                    glVertex3f(-.54,-.83,0);
                glEnd();

                glColor3ub(0,0,255);
                glBegin(GL_POINTS);
                    glVertex3f(-.51,-.83,0);
                glEnd();
                lightzzz++;
                if(lightzzz == 3)
                {
                    polLights = 2;
                    lightzzz = 0;
                }

            }
            else if(polLights == 2)
            {
                glColor3ub(0,0,255);
                glBegin(GL_POINTS);
                    glVertex3f(-.54,-.83,0);
                glEnd();

                glColor3ub(255,0,0);
                glBegin(GL_POINTS);
                    glVertex3f(-.51,-.83,0);
                glEnd();

                lightzzz++;
                 if(lightzzz == 3)
                 {
                     polLights = 1;
                     lightzzz = 0;
                 }
             }
        glPopMatrix();
    }
    else if(carModel == 3)
    {
        glColor3ub(255,0,0);
            glBegin(GL_POLYGON);
                glVertex3f(-.6,-.7,0);
                glVertex3f(-.6,-.9,0);
                glVertex3f(-.45,-.9,0);
                glVertex3f(-.45,-.7,0);
            glEnd();

            glColor3ub(255,255,255);
            glBegin(GL_POLYGON);
                glVertex3f(-.58,-.77,0);
                glVertex3f(-.58,-.87,0);
                glVertex3f(-.47,-.87,0);
                glVertex3f(-.47,-.77,0);
            glEnd();

            glColor3ub(0,0,0);
            glPointSize(10);
            glBegin(GL_POINTS);
                glVertex3f(-.605,-.75,0);
                glVertex3f(-.605,-.85,0);
                glVertex3f(-.445,-.85,0);
                glVertex3f(-.445,-.75,0);
            glEnd();

        glPopMatrix();
    }
}
```

Here the Driver has been drawn which we will drive in this game. As there are total 3 models all the models are drawn in this whole code snippet.

10.

```
509  int getOpCarPos(int opCarPos)
510  {
511      if(opCarPos == 0)
512      {
513
514          int i =1;
515          int pos = rand() % 3 + 1;
516          while(i)
517          {
518               if(pos == prevOpPos)
519                  pos = rand() % 3 + 1;
520               else
521                  break;
522          }
523
524          prevOpPos = pos;
525          return pos;
526      }
527      else
528          return opCarPos;
529  }
```

The traffic Positioning is done over here. We already stated that total 7 traffic will be present in the screen at a time. After that the loop is going to start over.

11.

```
void drawOpCar(int pos)
{
    if(pos == 1)
    {
        glPushMatrix();
            glColor3ub(255,127,39);
            glBegin(GL_QUADS);
                glVertex3f(-.6,.75,0);
                glVertex3f(-.6,.95,0);
                glVertex3f(-.45,.95,0);
                glVertex3f(-.45,.75,0);
            glEnd();

            glColor3ub(0,0,0);
            glPointSize(15);
            glBegin(GL_POINTS);
                glVertex3f(-.575,.72,0);
                glVertex3f(-.475,.72,0);
            glEnd();

            glColor3ub(255,0,255);
            glBegin(GL_QUADS);
                glVertex3f(-.575,.75,0);
                glVertex3f(-.575,.7,0);
                glVertex3f(-.475,.7,0);
                glVertex3f(-.475,.75,0);
            glEnd();
        glPopMatrix();
    }
    else if(pos == 2)
    {
        glPushMatrix();
            glColor3ub(255,127,39);
            glBegin(GL_QUADS);
                glVertex3f(-.075,.75,0);
                glVertex3f(-.075,.95,0);
                glVertex3f(.075,.95,0);
                glVertex3f(.075,.75,0);
            glEnd();

            glColor3ub(0,0,0);
            glPointSize(15);
            glBegin(GL_POINTS);
                glVertex3f(-.05,.72,0);
                glVertex3f(.05,.72,0);
            glEnd();

            glColor3ub(255,0,255);
            glBegin(GL_QUADS);
                glVertex3f(-.05,.75,0);
                glVertex3f(-.05,.7,0);
                glVertex3f(.05,.7,0);
                glVertex3f(.05,.75,0);
            glEnd();
        glPopMatrix();
    }
    else if(pos == 3)
    {
        glPushMatrix();
            glColor3ub(255,127,39);
            glBegin(GL_QUADS);
                glVertex3f(.6,.75,0);
                glVertex3f(.6,.95,0);
                glVertex3f(.45,.95,0);
                glVertex3f(.45,.75,0);
            glEnd();

            glColor3ub(0,0,0);
            glPointSize(15);
            glBegin(GL_POINTS);
                glVertex3f(.575,.72,0);
                glVertex3f(.475,.72,0);
            glEnd();

            glColor3ub(255,0,255);
            glBegin(GL_QUADS);
                glVertex3f(.575,.75,0);
                glVertex3f(.575,.7,0);
```

```
608                 glVertex3f(.475,.7,0);
609                 glVertex3f(.475,.75,0);
610             glEnd();
611         glPopMatrix();
612     }
613 }
614
615 void drawMenu()
616 {
617     glPushMatrix();
618
619         glColor3ub(0,0,0);
620         glBegin(GL_QUADS);
621             glVertex3f(-.45,.75,0);
622             glVertex3f(-.45,-.75,0);
623             glVertex3f(.45,-.75,0);
624             glVertex3f(.45,.75,0);
625         glEnd();
626
627         glColor3ub(0,0,255);
628         glBegin(GL_QUADS);
629             glVertex3f(-.4,.7,0);
630             glVertex3f(-.4,-.7,0);
631             glVertex3f(.4,-.7,0);
632             glVertex3f(.4,.7,0);
633         glEnd();
634
635         glColor3ub(0,0,0);
636
637         glBegin(GL_QUADS);
638             glVertex3f(-.32,.55,0);
639             glVertex3f(-.32,.35,0);
640             glVertex3f(.32,.35,0);
641             glVertex3f(.32,.55,0);
642         glEnd();
643
644         glBegin(GL_QUADS);
645             glVertex3f(-.32,.25,0);
```

```
646             glVertex3f(-.32,.05,0);
647             glVertex3f(.32,.05,0);
648             glVertex3f(.32,.25,0);
649         glEnd();
650
651         glBegin(GL_QUADS);
652             glVertex3f(-.32,-.25,0);
653             glVertex3f(-.32,-.05,0);
654             glVertex3f(.32,-.05,0);
655             glVertex3f(.32,-.25,0);
656         glEnd();
657
658         glBegin(GL_QUADS);
659             glVertex3f(-.32,-.55,0);
660             glVertex3f(-.32,-.35,0);
661             glVertex3f(.32,-.35,0);
662             glVertex3f(.32,-.55,0);
663         glEnd();
664
665         glColor3ub(128,128,128);
666
667         glBegin(GL_QUADS);
668             glVertex3f(-.30,.53,0);
669             glVertex3f(-.30,.37,0);
670             glVertex3f(.30,.37,0);
671             glVertex3f(.30,.53,0);
672         glEnd();
673
674         glBegin(GL_QUADS);
675             glVertex3f(-.30,.23,0);
676             glVertex3f(-.30,.07,0);
677             glVertex3f(.30,.07,0);
678             glVertex3f(.30,.23,0);
679         glEnd();
680
681         glBegin(GL_QUADS);
682             glVertex3f(-.30,-.53,0);
683             glVertex3f(-.30,-.37,0);
684             glVertex3f(.30,-.37,0);
685                 glVertex3f(.30,-.53,0);
686             glEnd();
687
688             glBegin(GL_QUADS);
689                 glVertex3f(-.30,-.23,0);
690                 glVertex3f(-.30,-.07,0);
691                 glVertex3f(.30,-.07,0);
692                 glVertex3f(.30,-.23,0);
693             glEnd();
694
695             //text
696             Sprint(-.07,.43,start);
697             Sprint(-.13,-.17,hScore);
698             Sprint(-.07,-.47,exitGame);
699             Sprint(-.13,.13,carsText);
700
701
702         glPopMatrix();
703     }
```

Here every Traffic has been drawn. And the way they are going to move has been shown in the previous part. This is the drawing part of the traffic.

12.

```
615  void drawMenu()
616  {
617      glPushMatrix();
618
619          glColor3ub(0,0,0);
620          glBegin(GL_QUADS);
621              glVertex3f(-.45,.75,0);
622              glVertex3f(-.45,-.75,0);
623              glVertex3f(.45,-.75,0);
624              glVertex3f(.45,.75,0);
625          glEnd();
626
627          glColor3ub(0,0,255);
628          glBegin(GL_QUADS);
629              glVertex3f(-.4,.7,0);
630              glVertex3f(-.4,-.7,0);
631              glVertex3f(.4,-.7,0);
632              glVertex3f(.4,.7,0);
633          glEnd();
634
635          glColor3ub(0,0,0);
636
637          glBegin(GL_QUADS);
638              glVertex3f(-.32,.55,0);
639              glVertex3f(-.32,.35,0);
640              glVertex3f(.32,.35,0);
641              glVertex3f(.32,.55,0);
642          glEnd();
643
644          glBegin(GL_QUADS);
645              glVertex3f(-.32,.25,0);
646              glVertex3f(-.32,.05,0);
647              glVertex3f(.32,.05,0);
648              glVertex3f(.32,.25,0);
649          glEnd();
650
651          glBegin(GL_QUADS);
652              glVertex3f(-.32,-.25,0);
```

```
653             glVertex3f(-.32,-.05,0);
654             glVertex3f(.32,-.05,0);
655             glVertex3f(.32,-.25,0);
656         glEnd();
657
658         glBegin(GL_QUADS);
659             glVertex3f(-.32,-.55,0);
660             glVertex3f(-.32,-.35,0);
661             glVertex3f(.32,-.35,0);
662             glVertex3f(.32,-.55,0);
663         glEnd();
664
665         glColor3ub(128,128,128);
666
667         glBegin(GL_QUADS);
668             glVertex3f(-.30,.53,0);
669             glVertex3f(-.30,.37,0);
670             glVertex3f(.30,.37,0);
671             glVertex3f(.30,.53,0);
672         glEnd();
673
674         glBegin(GL_QUADS);
675             glVertex3f(-.30,.23,0);
676             glVertex3f(-.30,.07,0);
677             glVertex3f(.30,.07,0);
678             glVertex3f(.30,.23,0);
679         glEnd();
680
681         glBegin(GL_QUADS);
682             glVertex3f(-.30,-.53,0);
683             glVertex3f(-.30,-.37,0);
684             glVertex3f(.30,-.37,0);
685             glVertex3f(.30,-.53,0);
686         glEnd();
687
688         glBegin(GL_QUADS);
689             glVertex3f(-.30,-.23,0);
690             glVertex3f(-.30,-.07,0);
691             glVertex3f(.30,-.07,0);
692              glVertex3f(.30,-.23,0);
693         glEnd();
694
695         //text
696         Sprint(-.07,.43,start);
697         Sprint(-.13,-.17,hScore);
698         Sprint(-.07,-.47,exitGame);
699         Sprint(-.13,.13,carsText);
700
701
702     glPopMatrix();
703 }
```

Here in this part the whole menu has been drawn. And as text are printed so we can see the Sprint function used over here.

13.

```
704  void drawHighScore()
705  {
706      glPushMatrix();
707          glColor3ub(0,0,0);
708          glBegin(GL_QUADS);
709              glVertex3f(-.45,.65,0);
710              glVertex3f(-.45,-.65,0);
711              glVertex3f(.45,-.65,0);
712              glVertex3f(.45,.65,0);
713          glEnd();
714
715          glColor3ub(0,0,255);
716          glBegin(GL_QUADS);
717              glVertex3f(-.4,.6,0);
718              glVertex3f(-.4,-.6,0);
719              glVertex3f(.4,-.6,0);
720              glVertex3f(.4,.6,0);
721          glEnd();
722
723          Sprint(-.13,.5,hScore);
724          float scPos = .4;
725          for(int i = 0; i < 10; i++)
726          {
727              if(highScore[i] < 10)
728                  highScore[i] = 0;
729              itoa(highScore[i], buffer, 10);
730              SprintScore(buffer,6,-.08,scPos);
731              scPos -= .1;
732          }
733      glPopMatrix();
734  }
```

High score is written as follows in the High Score Tab.

14.

```
736     void drawScoreBoard()
737     {
738         glColor3ub(0,0,0);
739             glBegin(GL_QUADS);
740                 glVertex3f(.70,.9,0);
741                 glVertex3f(.70,.4,0);
742                 glVertex3f(1,.4,0);
743                 glVertex3f(1,.9,0);
744             glEnd();
745
746             glColor3ub(0,0,255);
747             glBegin(GL_QUADS);
748                 glVertex3f(.72,.88,0);
749                 glVertex3f(.72,.42,0);
750                 glVertex3f(.98,.42,0);
751                 glVertex3f(.98,.88,0);
752             glEnd();
753
754         Sprint(.78,.8,scoreText);
755         itoa (score, buffer, 10);
756         SprintScore(buffer, 6, .78,.75);
757
758         Sprint(.78,.65,levelText);
759         itoa (level, buffer, 10);
760         SprintScore(buffer, 2, .83,.60);
761
762         Sprint(.78,.5,mphText);
763         itoa (mph, buffer, 10);
764         SprintScore(buffer, 3, .81,.45);
765     }
```

Score Board which is always running in the game showing score, speed and Level is drawn using this function.

15.

```
767  void drawGameOver()
768  {
769      drawRoad();
770
771      glPushMatrix();
772
773          glColor3ub(0,0,0);
774          glBegin(GL_QUADS);
775              glVertex3f(-.45,.65,0);
776              glVertex3f(-.45,-.65,0);
777              glVertex3f(.45,-.65,0);
778              glVertex3f(.45,.65,0);
779          glEnd();
780
781          glColor3ub(0,0,255);
782          glBegin(GL_QUADS);
783              glVertex3f(-.4,.6,0);
784              glVertex3f(-.4,-.6,0);
785              glVertex3f(.4,-.6,0);
786              glVertex3f(.4,.6,0);
787          glEnd();
788
789          glColor3ub(0,0,0);
790
791          glBegin(GL_QUADS);
792              glVertex3f(-.31,.11,0);
793              glVertex3f(-.31,-.11,0);
794              glVertex3f(.31,-.11,0);
795              glVertex3f(.31,.11,0);
796          glEnd();
797
798          glBegin(GL_QUADS);
799              glVertex3f(-.31,-.41,0);
800              glVertex3f(-.31,-.19,0);
801              glVertex3f(.31,-.19,0);
802              glVertex3f(.31,-.41,0);
803          glEnd();
804
805             glColor3ub(128,128,128);
806
807             glBegin(GL_QUADS);
808                 glVertex3f(-.3,.1,0);
809                 glVertex3f(-.3,-.1,0);
810                 glVertex3f(.3,-.1,0);
811                 glVertex3f(.3,.1,0);
812             glEnd();
813
814             glBegin(GL_QUADS);
815                 glVertex3f(-.3,-.4,0);
816                 glVertex3f(-.3,-.2,0);
817                 glVertex3f(.3,-.2,0);
818                 glVertex3f(.3,-.4,0);
819             glEnd();
820
821        glPopMatrix();
822             Sprint(-.35,.3,gameOverText);
823             Sprint(-.13,-.02,mainMenuText);
824             Sprint(-.07,-.32,exitGame);
825
826  }
```

Here the gameover part is done. Mainly after colliding with the traffic two options will appear showing Main Menu or Exit.

16.

```
854   void update(int value)
855   {
856       //road white
857        _move4 -= speed;
858        _move3 -= speed;
859        _move2 -= speed;
860        _move1 -= speed;
861
862       if(_move4 < -.6)
863           _move4 = 2.2;
864       else if(_move3 < -1.3)
865           _move3 = 1.5;
866       else if(_move2 < -2)
867           _move2 = .8;
868       else if(_move1 < -2.7)
869           _move1 = .1;
870
871       //car
872       for(int i = 1; i <= nOp; i++)
873       {
874           if(carOpPos[i] > 0)
875               _opCar[i] -= speed;
876
877           if(nOp == i)
878           {
879               if(_opCar[nOp] < -3.1)
880               {
881                   float diff =0;
882                   for(int i = 1; i <= nOp; i++)
883                   {
884                       carOpPos[i] = 0;
885                       _opCar[i] = diff;
886                       diff += .7;
887                   }
888               }
889           }
890       }
891
```

```
892     for(int i =1; i <= nOp ; i++)
893     {
894         if( -1.40 >_opCar[i] && _opCar[i] > -1.8 && carPos == carOpPos[i])
895         {
896             if(collide)
897                 collide = false;
898             else
899                 collide = true;
900         }
901     }
902
903     if(collide)
904     {
905         if(screen == 1)
906         {
907             screen = 3;
908             if(score > highScore[9])
909                 {
910                     highScore[9] = score;
911                     sortHighScore();
912                     writeHighScore();
913                 }
914             score = 0;
915         }
916         collide = false;
917     }
918
919     score++;
920
921     if(score > 500 && score < 1000)
922     {
923          speed = 0.03;
924          level = 2;
925          mph = 60;
926     }
927     else if(score > 1000 && score < 1500)
928     {
929          speed = 0.04;
930            level = 3;
931           mph = 70;
932      }
933      else if(score > 1500 && score < 2000)
934      {
935          speed = 0.06;
936          level = 4;
937          mph = 80;
938      }
939      else if(speed > 2000 && score < 2500)
940      {
941          speed = 0.08;
942          level = 5;
943          mph = 90;
944      }
945      else if (speed > 2500)
946      {
947          speed = 0.1;
948          level = 6;
949          mph = 100;
950      }
951
952      glutPostRedisplay();
953      glutTimerFunc(timer,update,0);
954 }
```

Here main updating part is done. If Collides with a traffic what will happen and Speed is increased after a certain distance is covered.

17.

```
956   void display()
957   {
958       glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
959       glColor3b(0,0,0);
960       glLoadIdentity();
961       glMatrixMode(GL_MODELVIEW);
962
963       if(screen == 0)
964       {
965           glPushMatrix();
966               drawRoad();
967               drawMenu();
968           glPopMatrix();
969       }
970       else if(screen == 1)
971       {
972           glPushMatrix();
973               drawRoad();
974               drawDriver();
975           glPopMatrix();
976
977           for(int i = 1;i <= nOp;i++)
978           {
979               glPushMatrix();
980                   glTranslated(0,_opCar[i],0);
981                   carOpPos[i] = getOpCarPos(carOpPos[i]);
982                   drawOpCar(carOpPos[i]);
983               glPopMatrix();
984           }
985           drawScoreBoard();
986       }
987       else if(screen == 2)
988       {
989           drawRoad();
990           drawHighScore();
991       }
992       else if(screen == 3)
993       {
994           drawGameOver();
995       }
996       else if(screen == 4)
997       {
998           drawRoad();
999           drawChooseCars();
1000      }
```

Everything we can see in the screen are being merged over here in this Display function.

18.

```
1007    int main (int argc, char** argv)
1008    {
1009
1010        readHighScore();
1011        readCarModel();
1012        glutInit(&argc, argv);
1013        glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
1014        glutInitWindowSize(800,800);
1015        glutCreateWindow("Dodge Traffic");
1016        glutDisplayFunc(display);
1017        glutKeyboardFunc(keyboard);
1018        glutMouseFunc(mouse);
1019        glutTimerFunc(25, update,0);
1020        glutMainLoop();
1021
1022        return 0;
1023    }
```

This is the main function which defines the screen size and other loop stuff needed.

Conclusion:

This is the total description of the project. All the screenshots of the code snippets are added with explanation at the end of every part for better understanding.