Computer Graphics (UCS505)

Project on 2D Helicopter Game

Submitted By

Mohammed Sohaib Rasul 101903385

Kushav Garg 101903378

Tushar Bansal 101903386

COE15

B.E. Third Year – COE

Submitted To:

Dr. Amrita Kaur

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology

Patiala – 147001

Table of Contents

INTRODUCTION

Computer graphics is concerned with all aspects of producing images using a computer. The field began humbly 50 years ago, with the display of few lines on a cathode ray tube(CRT); now, we can create images by computer that are indistinguishable from photographs of real objects.

In this chapter we discuss about the application of computer graphics, overview of the graphic system, graphics architectures. In this project we discuss about OpenGL- open graphics library API- which is used to develop the application program which is the matterin question.

APPLICATIONS OF COMPUTER GRAPHICS

The applications of computer graphics are many and varied. However we can divide them into four major areas:

- Display of Information
- Design
- Simulation and Animation
- User interfaces

Display of Information

Classical graphics techniques arose as a medium to convey information among people. Although spoken and written languages serve this purpose, images are easier to communicate with and hence the computer graphics plays an important role.

Information visualization is becoming increasingly important in the field of security, medicine weather forecasting and mapping. Medical imaging like CT, MRI, ultrasound scans are produced by medical imaging systems but are displayed by computer graphics system.

Design

Professions such as engineering and architecture are concerned with design which is an iterative process. The power of the paradigm of humans interacting with the computer displays was recognized by Ivan Sutherland.

Simulation and Animation

Graphics system, now are capable of generating sophisticated images in real time. Hence engineers and researchers use them as simulators. The most important use has been in the training of pilots.

USER INTERFACES

User interaction with computers has become dominated by a visual paradigm that includes windows, icons, menus and pointing devices. Applications such as office suites, web browsers are accustomed to this style.

PROBLEM STATEMENT

To design and implement a helicopter game using OpenGL.

OBJECTIVES

In this project we are going to design a Helicopter game using OpenGL. This project displays a helicopter in flight with its fan and rotor stabilizer moving.
The basic idea of the game is to dodge the rectangles and the game ends when the helicopter touches one of the rectangles.
The score is calculated based on the distance covered by the helicopter before the crash.

SCOPE

It can be used for gaming and entertainment purpose. It can also be extended to the development of android application.

COMPUTER GRAPHICS CONCEPTS USED

Vertex Processing

Vertex Processing mainly involves the transformation of objects, transformation of the coordinatesystem and projection transforming. It also involves color computation for each vertex.

Clipping and Primitive Assembly

The outputof the vertex processor is given as the input to this stage. This step is mainly involved in the clipping of the primitive; Clipping is necessary since the camera film has a limited sizeand hence cannot image the whole world at once. This stage computes the clipping volumeand considers only those vertices that falls within this volume for image formation. Those vertices that are outside this volume do not appear in the image and are said to be clipped.

Rasterization

This step is necessary because the primitives that emerge out of clipper are still represented in terms of vertices. Rasterizer would further process their vertices to generate pixels in the frame buffer. The Rasterizer determines which pixels in the frame buffer are inside the polygon and which pixels fall outside.

USER DEFINED FUNCTIONS

Main Function:

void main(int argc,char **argv)
It contains calls to various functions to perform different operations.

Display Function:

void display()

This function is used to call various functions that draw different objects to be displayed at the location depending upon where the function call to the particular function is made and messages to be displayed. It also has functions to set the color and also has function that forces the content of the frame buffer onto the display (glFlush( )).

Rectangle Function:

void rectangle(int b)
void rectangle1(int d)
void rectangle2(int d)
These functions are responsible for drawing rectangles at various positions. The values b andd are responsible for deciding the exact position of the rectangles.

Reset function:

Void reset()

Used to reset the values of all the variables to default values.

Limit function:

void limit()
This function is used to calculate the score.

Helicopter Function:

void helicopter(int x, int y)

x and y represents the x andy co ordinates respectively.
This function is used to draw the helicopter at the specified position. This function contains

code to draw various parts of the helicopter such as black line, head, big wing, small wing, foot, foot line.

Crashed Helicopter Function:

void crashed_helicopter(intx, int y)

x and y represents the co-ordinates of the crashed helicopter.

This function is used to draw various parts of the crashed helicopter on the game over screen once the helicopter has touched the rectangle and the game is over.

Init Function:

void init(void)

This function is used to set up the window

Up and Down Bar Functions:

Void down_bar()

Void up_bar()

Used to display the Up and down green color bars present on the screen

Mouse function:

void mouse(int btn,int state,int x,int y)

glutMouseFunc sets the mouse callback for the *current window*. When a user presses and releases mouse buttons in the window, each press and each release generates a mouse callback.

Keyboard function:

void mykeyboard(unsigned char key, int x, int y)

glutKeyboardFunc sets the keyboard callback for the *current window*.

CODE

```
#include<stdlib.h>
#include<GL/glut.h>
#include<time.h>
#include<dos.h>
#include<stdio.h>
#include<iostream>
#include<windows.h>
int win1, win2;
void Write(float x, float y, float z, float scale, char* s)
{
    int i, l = strlen(s);
    glPushMatrix();
    glTranslatef(x, y, z);
    glScalef(scale, scale, scale);
    for (i = 0; i < l; i++)
        glutStrokeCharacter(GLUT_STROKE_ROMAN, s[i]);
    glPopMatrix();
}

void frontsheet(void)
{
    glClearColor(0, 0, 0, 1);
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0, 1.0, 0.0);
    Write(-0.67, 0.9, 1, 0.0007, (char*)"Thapar Institute of Engineering");
    Write(-0.55, 0.8, 1, 0.0006, (char*)"  and Technology, Patiala");
    glColor3f(1.0, 0.0, 0.0);
    Write(-0.45, 0.6, 0.0, 0.0007, (char*)" 2D Helicopter Game");
    glColor3f(1.0, 1.0, 0.5);
    Write(-0.4, -0.8, 0.0, 0.0006, (char*)"Press 'C' to continue");
    glColor3f(1, 1, 0.0);
    Write(-1.0, 0.1, 0.0, 0.0007, (char*)" Submitted BY:");
    glColor3f(1.0, 1.0, 1.0);
    Write(-1.0, -0.03, 0.0, 0.0006, (char*)"1. Kushav Garg: 101903378");
    Write(-1.0, -0.13, 0.0, 0.0006, (char*)"2. Tushar Bansal: 101903386");
    Write(-1.0, -0.23, 0.0, 0.0006, (char*)"3. Mohammed Sohaib Rasul:
101903385");
    glColor3f(1, 1, 0.0);
    Write(-1.0, -0.4, 0.0, 0.0007, (char*)" Submitted to: ");
    glColor3f(1.0, 1.0, 1.0);
    Write(0.15, -0.415, 0.0, 0.0006, (char*)" Dr. Amrita Kaur");
    glFlush();
```

```
}

float bspd = 0.005; // block dx value

//char name[25];

float b1x = 50.0, b1y = 0;//block 1 init position

float hm = 0.0;//copter moving dy value

int i = 0, sci = 1; float scf = 1; // for increment score score_int score_flag

char scs[20], slevel[20];
//to store score_string using itoa() and level as well
int level = 1, lflag = 1, wflag = 1; //level_flag & welcome_flag init w/1
void init(void)
{
    srand(time(0));//it generates a random no each time code is executed
    b1y = (rand() % 45) + 10;//b/w 10 to 44
    glClearColor(0.0, 0.0, 0.0, 0.0);
    glShadeModel(GL_SMOOTH);
    glLoadIdentity();
    glOrtho(0.0, 100.0, 0.0, 100.0, -1.0, .0);
}

void drawcopter()
{
    glColor3f(0.5, 1.0, 0.3);
    glRectf(10, 49.8, 19.8, 44.8);//body
    glRectf(2, 46, 10, 48);//tail
    glRectf(2, 46, 4, 51);//tail up
    glRectf(14, 49.8, 15.8, 52.2);//propeller stand
    glRectf(7, 53.6, 22.8, 52.2);//propeller*/
}

void renderBitmapString(float x, float y, float z, void* font, const char*
string)
{
    const char* c;
    glRasterPos3f(x, y, z);
    for (c = string; *c != '\0'; c++)
    {
        glutBitmapCharacter(font, *c);
    }
```

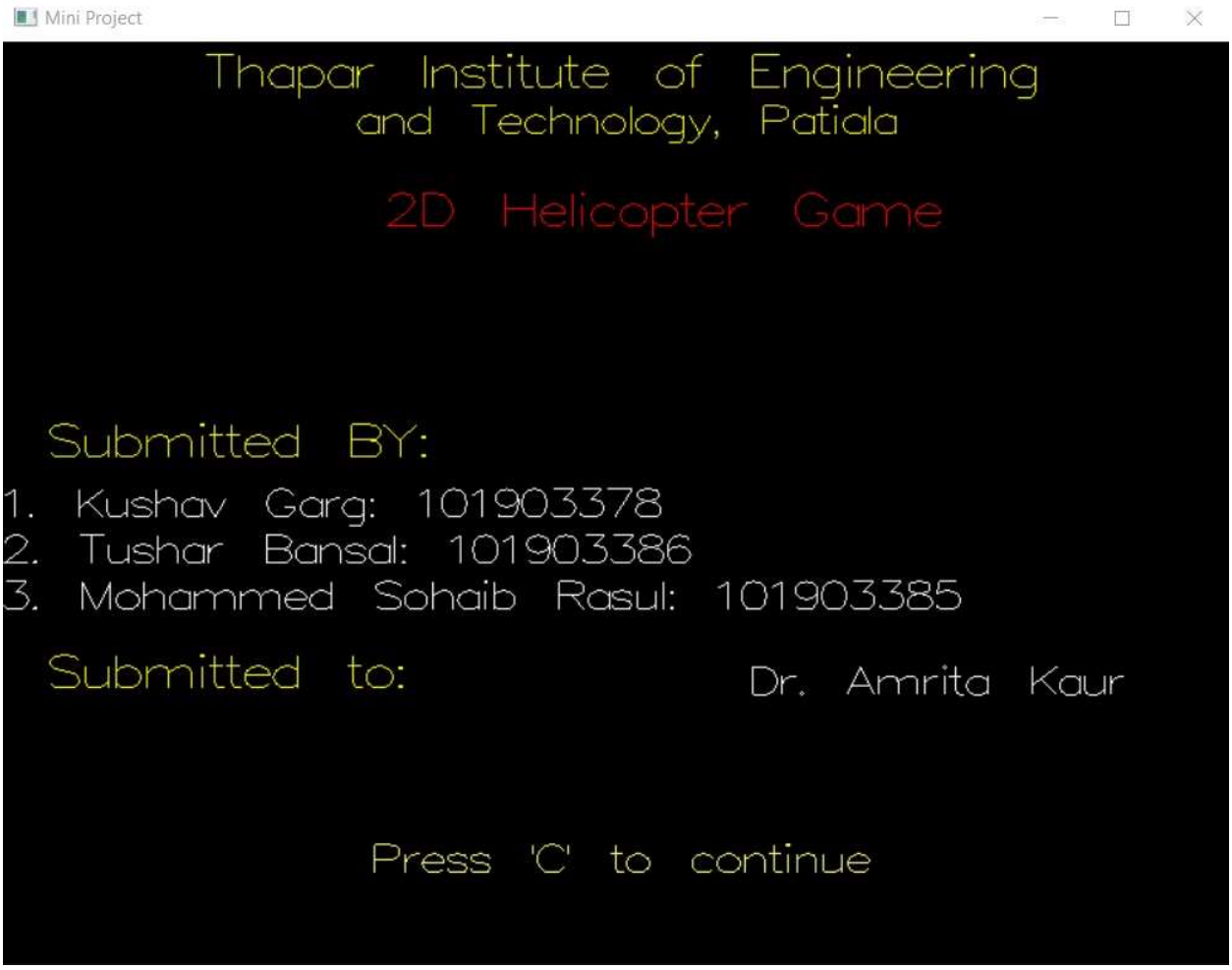
```
}

void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    //GameOver Checking
    if ((i == 3600 || i == -3600)
        //top and bottom checking

        (((int)b1x == 10 || (int)b1x == 7 || (int)b1x == 4 || (int)b1x == 1) &&
(int)b1y < 53 + (int)hm && (int)b1y + 35>53 + (int)hm)
        // propeller front checking

        (((int)b1x == 9 || (int)b1x == 3 || (int)b1x == 6) && (int)b1y < 45 +
(int)hm && (int)b1y + 35>45 + (int)hm)
        //lower body checking

        (((int)b1x == 0) && (int)b1y < 46 + (int)hm && (int)b1y + 35>46 +
(int)hm))
        // lower tail checking
    {
        glColor3f(0.0, 0.0, 1.0);
        glRectf(0.0, 0.0, 100.0, 100.0);
        glColor3f(1.0, 0.0, 0.0);
        renderBitmapString(40, 70, 0, GLUT_BITMAP_HELVETICA_18, "GAME O“ER!!!");
   ”    glColor3f(1.0, 1.0, 1.0);
        renderBitmapString(30, 58, 0, GLUT_BITMAP_TIMES_ROMAN_24, "THANKS“FOR
PLAYING THE GAME!!");
   ”    //renderBitmapString(45, 58, 0, GLUT_BITMAP_TIMES_ROMAN_24, "scored“");
   ”    renderBitmapString(70, 58, 0, GLUT_BITMAP_TIMES_ROMAN_24, scs);
        glutSwapBuffers();
        glFlush();
        printf("\nGAME“OVER\n\n");
   ”    system("pause"“;
   ”    //printf("%s\You“scored  %s", name” scs);
        printf("\n\nCl“se the console window to exit...\n");
   ”    exit(0);
        //getch();
    }

    else
    {
        //on every increase by 50 in score in each level
        if (sci % 50 == 0 && lflag == 1)
```

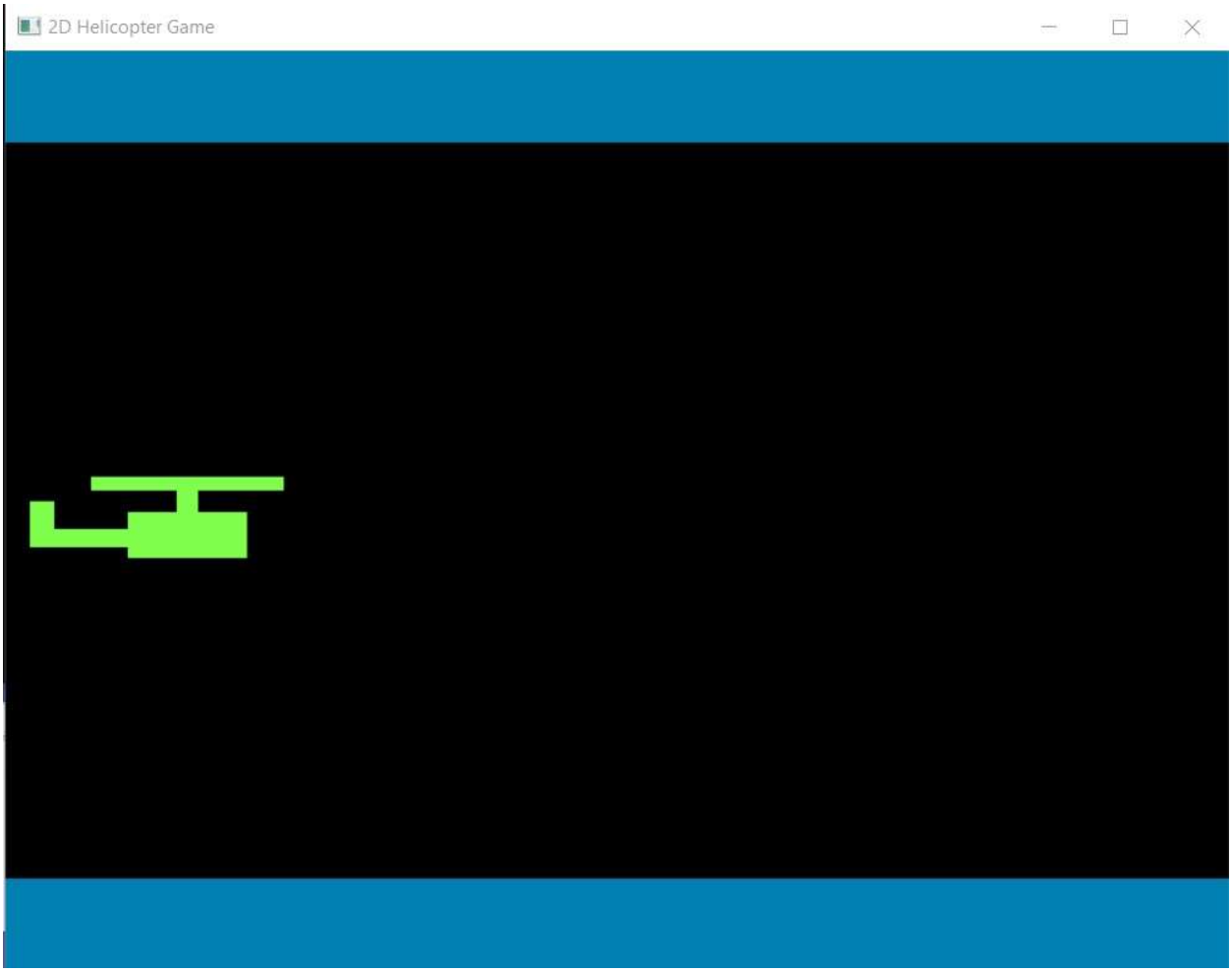
```
        {
            lflag = 0; //make level_flag=0
            level++;//increase level by 1
            //bspd += 0.001;//increase block_dx_speed by 0.01
        }
        //within every level make level_flag=1
        else if (sci % 50 != 0 && lflag != 1)
        {
            lflag = 1;
        }
        glPushMatrix();
        glColor3f(0.0, 0.5, 0.7);
        glRectf(0.0, 0.0, 100.0, 10.0);      //ceil
        glRectf(0.0, 100.0, 100.0, 90.0);    //floor

        glColor3f(0.0, 0.0, 0.0);   //score
     //  renderBitmapString(1, 3, 0, GLUT_BITMAP_TIMES_ROMAN_24, "Distan“e:");
    ”    //glColor3f(0.7,0.7,0.7);

        printf(slevel, "%d", l“ve”);   //level
     //  renderBitmapString(80, 3, 0, GLUT_BITMAP_TIMES_ROMAN_24, "Level:“);
    ”    renderBitmapString(93, 3, 0, GLUT_BITMAP_TIMES_ROMAN_24, slevel);

        scf += 0.01;              //so less as program run very fast
        sci = (int)scf;
        printf(scs, "%d", s“i)”
        //from int to char convertion to display score
        renderBitmapString(20, 3, 0, GLUT_BITMAP_TIMES_ROMAN_24, scs);
        glTranslatef(0.0, hm, 0.0);
        // hm(=dy) changes occur by mouse func
        drawcopter();
        //code for helicopter
        //if wall move towards left & get out of projection volume
        if (b1x < -10)
        {
            b1x = 50;            //total width is 50
            b1y = (rand() % 25) + 20;
            //10 for selling+10 for giving enough space
            // block bottom limit 0+20 & top limit 24+20=44
        }

        else
            b1x -= bspd;
        //within the projection volume dec its x value by block_speed
```

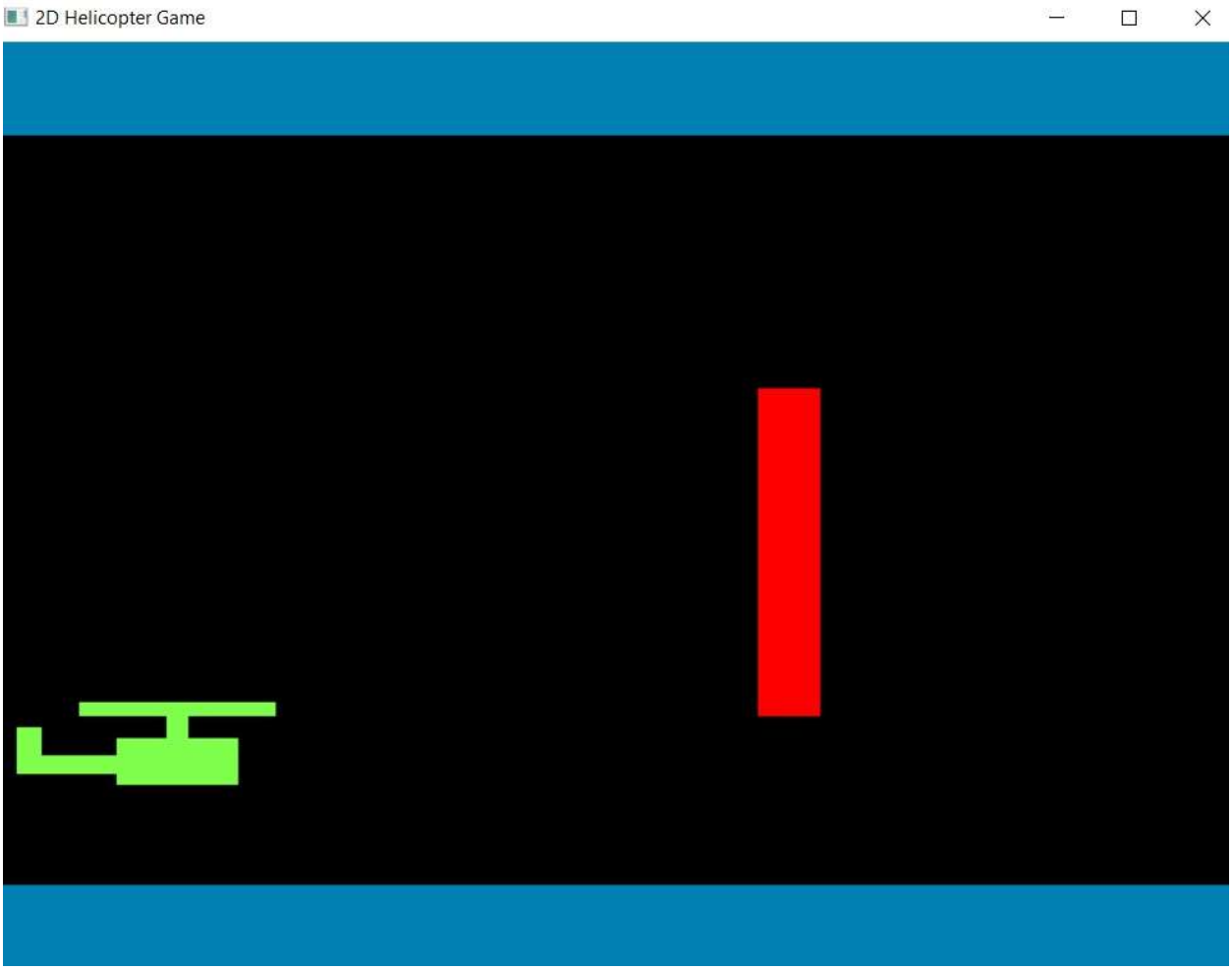
```
        glTranslatef(b1x, -hm, 0.0);

        glColor3f(1.0, 0.0, 0.0);
        glRectf(b1x, b1y, b1x + 5, b1y + 35);//block 1

        glPopMatrix();

        glutSwapBuffers();
        glFlush();
    }
}

void moveHeliU(void)
{

    hm += 0.01;
    i++;
    glutPostRedisplay();

}

void moveHeliD()
{

    hm -= 0.01;
    i--;
    glutPostRedisplay();

}
void mouse(int button, int state, int x, int y)
{
    switch (button)
    {
    case GLUT_LEFT_BUTTON:

        if (state == GLUT_DOWN)
            glutIdleFunc(moveHeliU);

        else if (state == GLUT_UP)
            glutIdleFunc(moveHeliD);
        break;
    default: break;
    }
```

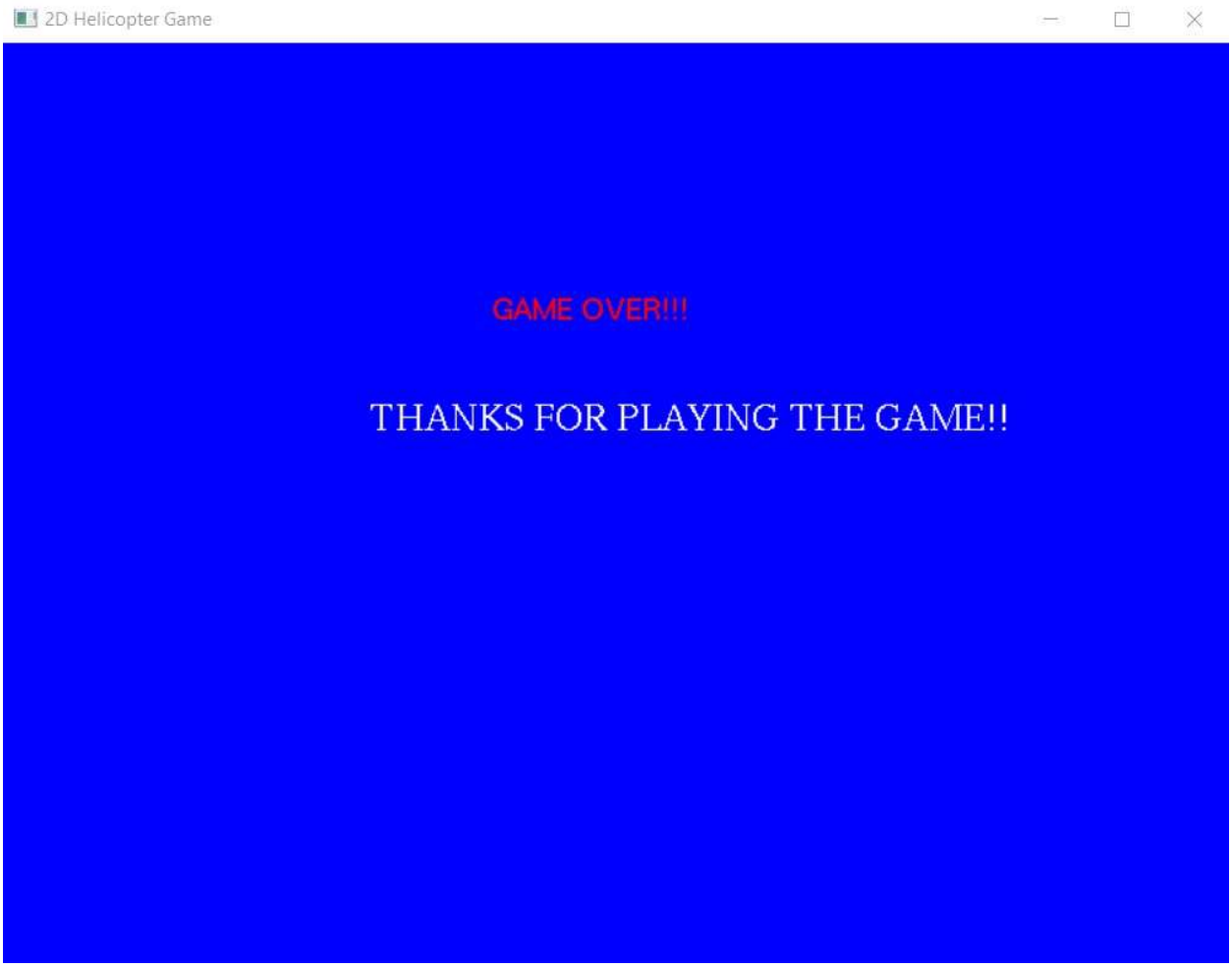
```
}
void keys(unsigned char key, int x, int y)
{
    if (key == 'w') gl‘t’dleFunc(moveHeliU);
    if (key == 's') gl‘t’dleFunc(moveHeliD);
}
void keyboards(unsigned char key, int x4, int y4)
{
    if (key == 'c' || ‘e’ == 'C')
  ‘ ’
        glutDestroyWindow(win1);
        glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
        win2 = glutCreateWindow("2D Helicopter Game");
   ”    glClearColor(0.0, 0.0, 0.0, 0.0);
        glFlush();
        glutDisplayFunc(display);
        gluOrtho2D(-1000, 1000, 0, 1000);
        init();
        glutMouseFunc(mouse);
        glutKeyboardFunc(keys);
    }
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(800, 600);
    glutInitWindowPosition(200, 200);
    win1 = glutCreateWindow("Mini Project");
    glFlush();
    glutDisplayFunc(frontsheet);
    glutKeyboardFunc(keyboards);
    glutMainLoop();
    return 0;

}
```

OUTPUT/ SCREENSHOTS




2D Helicopter Game


2D Helicopter Game


2D Helicopter Game
GAME OVER!!!
THANKS FOR PLAYING THE GAME!!