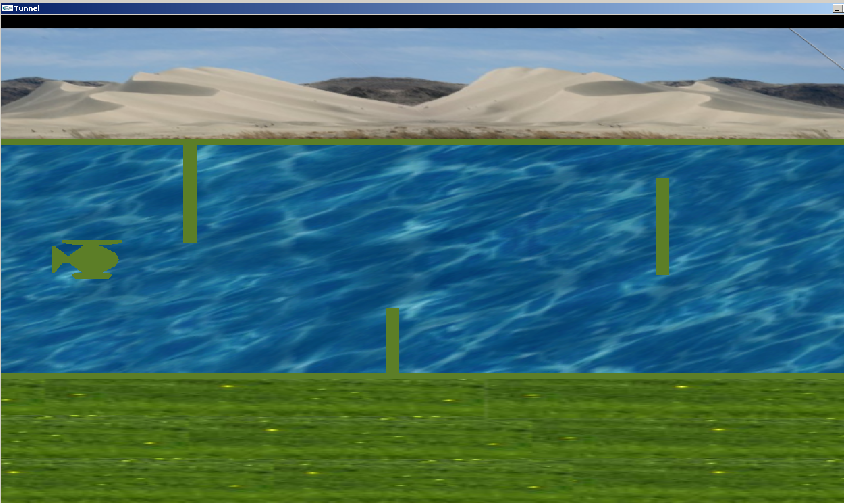
Computer Graphics   
  
Project Presentation

**Helicopter Game**



 **Group Members**   
 **Name Seat No.**  
  
 S. Abu Owais Bin Nasar B10101086  
  
 M. Umair Khan B10101067  
   
**Class:** VI Semester BS(CS)-Morning  
   
**Sec.:** B  
 

**Course Supervisor :** Miss Humera Tariq

CONTENTS

[Introduction: 2](#_Toc371626228)

[Game Screenshot 3](#_Toc371626229)

[Project Goal 3](#_Toc371626230)

[System Requirements 3](#_Toc371626231)

[Game Scenario 4](#_Toc371626232)

[Implementation Details 4](#_Toc371626233)

[Features 4](#_Toc371626234)

[World Window 5](#_Toc371626235)

[Helicopter Texture 5](#_Toc371626236)

[Moving Helicopter 5](#_Toc371626237)

[Collision Detection 6](#_Toc371626238)

[Special Effect 7](#_Toc371626239)

[Gravity 7](#_Toc371626240)

[Conclusion 8](#_Toc371626241)

[Source Code 8](#_Toc371626242)

[References 14](#_Toc371626243)

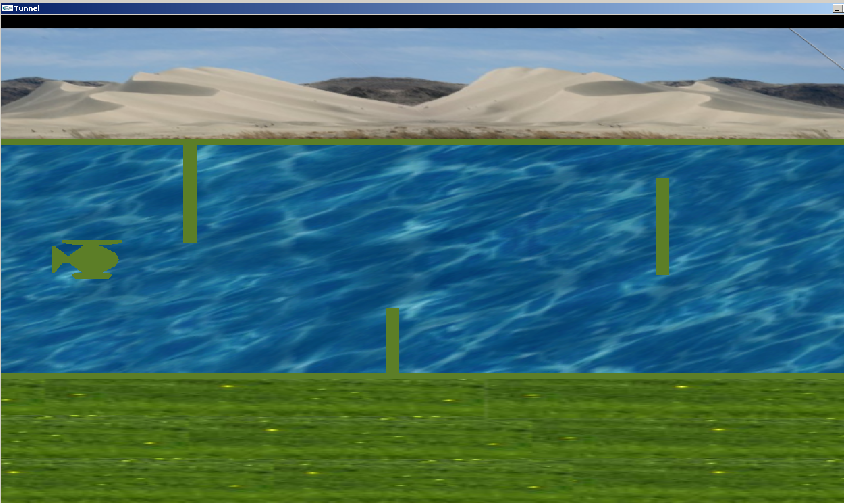
Simple Helicopter Game

# INTRODUCTION:

“Simple Helicopter Game” is compile in the Visual Studio C++ by using OPENGL.

In this game project the user tries to fly helicopter as far as user can by controlling from mouse. Watch out the incoming rectangular shape wall and also keep flying helicopter in track, don’t crash into any of the incoming rectangular shape wall or get out from track to avoid game to be over. If user crash once, it’s game over no matter how far user have flown and user will have to start from beginning. The object of the game is to see how much distance user can cover before finally crashing.

# GAME SCREENSHOT



# PROJECT GOAL

The aim of this project is to develop a 2D game which supports basic operations which include mouse event handling, keyboard event handling, special effect, gravity, collision, transformation etc.

# SYSTEM REQUIREMENTS

Hardware requirements

* Processor : 1.5 GHz
* RAM : 256 MB
* Hard disk : At least 100mb of hard disk free space
* Monitor
* Keyboard
* Mouse

Software Requirements

* Operating system : Windows
* Video Card

# GAME SCENARIO

* By using mouse left click helicopter move upward.
* Releasing mouse left click helicopter move downward.
* User has to flying helicopter in track and save it from upcoming wall.

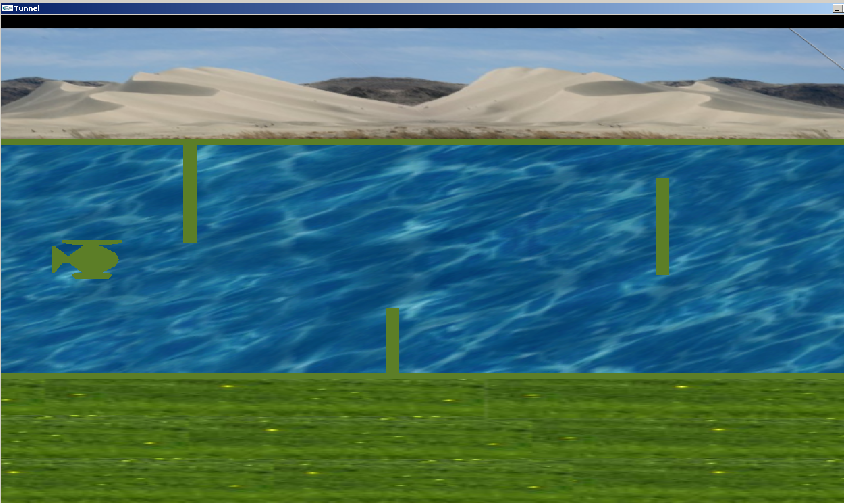
# IMPLEMENTATION DETAILS

In this project we have used cmath.h, time.h, cstdlib.h, fstream.h, iostream.h, Windows.h, glut.h,, helicopter.h, RGBA.h, textures.h, and wav.h file.

It based on object oriented programming: inheritance, constructor, the main class is helicopter.h which control all single step of helicopter.

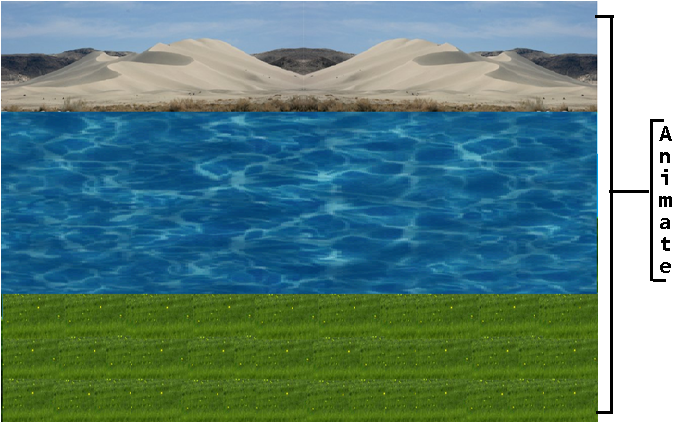
# FEATURES

* Elements of pictures Created in Computer Graphics
* Polygon drawings
* Keyboard Handler
* Mouse Handler
* Collision Detection
* Physics ( Speed , Timer )
* Gravity
* Special Effect
* Transformation



# WORLD WINDOW

The world of the game is composed of animated background which indicates the environment.

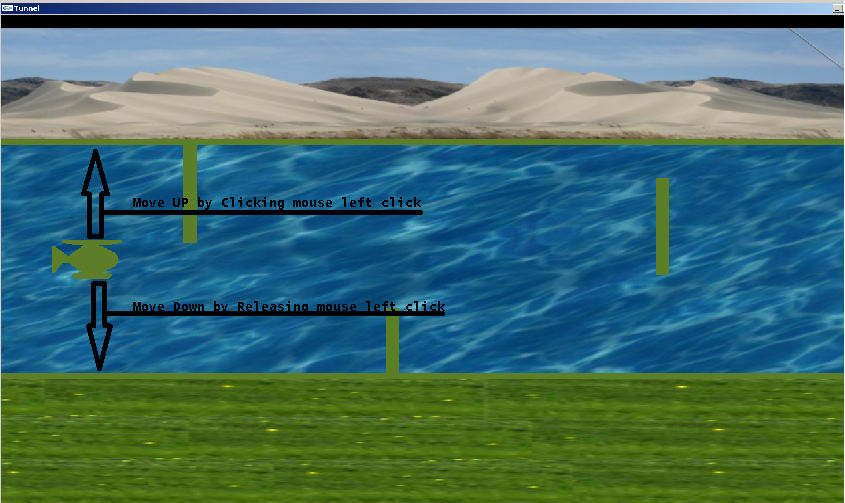


# HELICOPTER TEXTURE

The Helicopter is texture by copter.bmp image which coordinate are (100,copter.height - 20), (0, copter.height - 20), (0, copter.height + 20), and (100, copter.height + 20). In first coordinate copter.height is the initial height of helicopter which is assigned by void Initialize\_Heliopter().

# MOVING HELICOPTER

The helicopter moves horizontally with a constant speed. Helicopter moves up and down by using left mouse click. This functionality is created by using mouse handler.



The maximum of height to fly helicopter is 550 and minimum height is 200.

# COLLISION DETECTION

Collision is an instance of one moving object or person striking violently against another. When helicopter hits the objects coming from the front or get out from the track then collision is detected and the game is over.

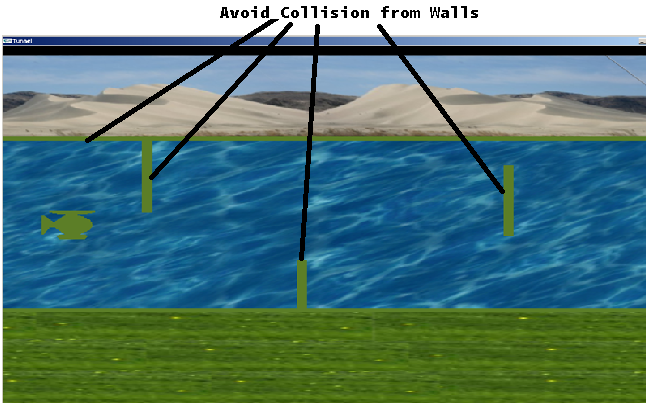
void collision() //Collide from TOP Wall

{

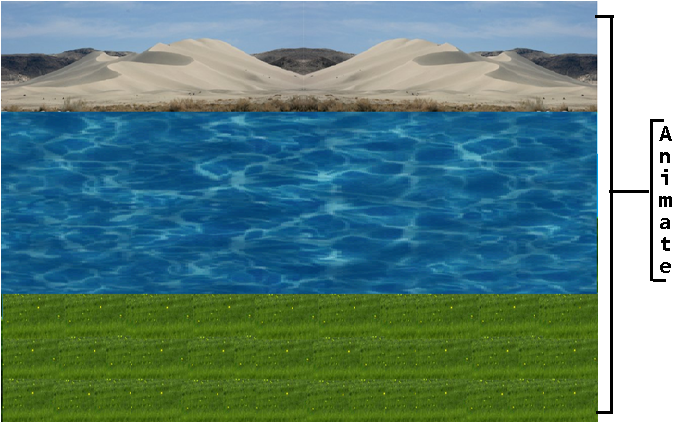
if(copter.height>550|| copter.height<250)

gameover=true;

}



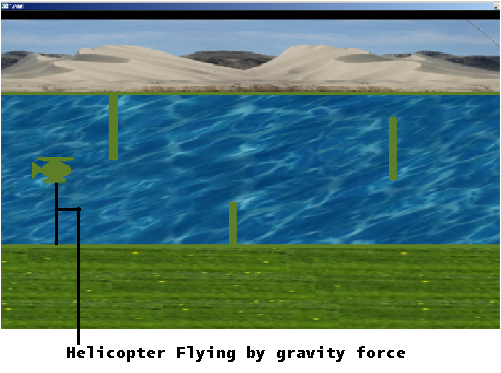
# SPECIAL EFFECT Background of helicopter is textured with images and with the help of timer its creates a special effect just like an animation.



It includes three image names as hill.bmp, water.bmp and grass.bmp. The coordinate of hill image are (1400, 730), (0,730), (0,550), and (1400, 550). The coordinate of water image are (1400, 550), (0,550), (0,200), and (1400, 200). The coordinate of grass image are (1400, 200), (0,200), (0, 0), and (1400, 0).

# GRAVITY Gravity is the force that attracts a body towards the centre of the earth, or towards any other physical body having mass

By pressing the left mouse click helicopter move against the gravity and by releasing the click gravity effects on it.



void movement()

{  
 a=downward\_force/100;

b=copter.upward\_force/100;

copter.move+=a-b;

copter.height-= copter.move;

if(copter.move>maxVel)copter.move=maxVel;

if(copter.move<-maxVel) copter.move=-maxVel;

}

The initial value of downward\_force is 300 and copter.upward is 0. By dividing downward\_force and copter.upward\_force by 100 and saving in a and b respectively. After that we subtract and add in copter.move and subtracting with copter.height because when downward\_force increase helicopter go down.

void mouse(int button, int state, int x, int y)  
{  
 if(button==GLUT\_LEFT\_BUTTON && state==GLUT\_DOWN)  
 copter.upwardThrust=2\*forceGrav;  
 else if(button==GLUT\_LEFT\_BUTTON)  
 copter.upwardThrust=0;

}

By pressing the mouse left button copter.upwardthrust increase by twice of forceGrav and helicopter fly up. By releasing mouse left button helicopter go down because copter.upwardthrust will 0 it give the effect of gravity.

# CONCLUSION

Our initial goal was to develop 2D helicopter game. In this project, we reached our goal and successfully delivered 2D game that allows users to play game in free time.

We hope to carry on working on it.

# SOURCE CODE

#include <cmath>

#include <ctime>

#include <cstdlib>

#include <fstream>

#include <iostream>

#include <Windows.h>

#include <glut.h>

#include "helicopter.h"

#include "RGBA.h"

#include "textures.h"

#include "wav.h"

using namespace std;

RGBApixmap pix[6];

float downward\_force=300, width,height,maxVel=5,a,b;

bool isStart=true, gameover=false;

int highScore=0, x=0, y=0;

float waterMovement =0.0, grassMovement = 0.0, hillMovement =0.0;

float waterTextureOffset, grassTextureOffset, hillTextureOffset;

extern const float WATER\_TEXTURE\_SIZE=10.0, GRASS\_TEXTURE\_SIZE=10.0, HILL\_TEXTURE\_SIZE=10.0;

extern const float WATER\_ALPHA=0.0, GRASS\_ALPHA=0.0, HILL\_ALPHA=0.0;

const float WATER\_TERTURE\_TIME=8.0f, HILL\_TERTURE\_TIME=8.0f, GRASS\_TERTURE\_TIME=8.0f;

const float STEP\_TIME=0.02f;

unsigned int textuer[6];

GLuint t\_grass, t\_hill, t\_water,t\_og;

Helicopter heli(0,0);

void Initialize\_Heliopter()

{

heli.height=height/2+25;

}

void DrawBoxes()

{

glColor3f(0,1,0);

glBegin(GL\_POLYGON);

glVertex2f(0,550);

glVertex2f(1300,550);

glVertex2f(1300,560);

glVertex2f(0,560);

glEnd();

glBegin(GL\_POLYGON);

glVertex2f(0,190);

glVertex2f(1300,190);

glVertex2f(1300,200);

glVertex2f(0,200);

glEnd();}

void myInit()

{

t\_hill = LoadTexture("hill.bmp");

t\_water = LoadTexture("water.bmp");

t\_grass = LoadTexture("grass.bmp");

t\_og = LoadTexture("copter.bmp");

glEnable(GL\_TEXTURE\_2D);

}

void Draw(int id)

{

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,id);

pix[4].readBMPFile("copter.bmp");

pix[4].setChromaKey(255,255,255);

glBegin(GL\_QUADS);

glTexCoord2f(0.0f ,0.0f); glVertex2i(100,heli.height - 20); // right top corner

glTexCoord2f(1.0f ,0.0f); glVertex2i(0,heli.height - 20); // left top corner

glTexCoord2f(1.0f ,1.0f); glVertex2i(0,heli.height + 20); // left bottom corner

glTexCoord2f(0.0f ,1.0f); glVertex2i(100,heli.height + 20); // right bottom corner

glEnd();

glDisable(GL\_TEXTURE\_2D);

}

void drawHill(int id)

{

glDisable(GL\_LIGHTING);

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,id);

glTexParameteri(GL\_TEXTURE\_2D,GL\_TEXTURE\_MIN\_FILTER,GL\_LINEAR);

glTexParameteri(GL\_TEXTURE\_2D,GL\_TEXTURE\_MAG\_FILTER,GL\_LINEAR);

glTexEnvi(GL\_TEXTURE\_ENV,GL\_TEXTURE\_ENV\_MODE,GL\_REPLACE);

glDisable(GL\_NORMALIZE);

glEnable(GL\_BLEND);

glBegin(GL\_QUADS);

glTexCoord2f(0.0f+sin(hillMovement/10.0),0.0f); glVertex2i(1400,730);// right top corner

glTexCoord2f(1.0f+sin(hillMovement/10.0),0.0f);glVertex2i(0,730);// left top corner

glTexCoord2f(1.0f+sin(hillMovement/10.0),1.0f);glVertex2i(0,550);// left bottom corner

glTexCoord2f(0.0f+sin(hillMovement/10.0),1.0f);glVertex2i(1400,550);// right bottom corner

glEnd();

}

void drawWater(int id)

{

glDisable(GL\_LIGHTING);

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,id);

glTexParameteri(GL\_TEXTURE\_2D,GL\_TEXTURE\_MIN\_FILTER,GL\_LINEAR);

glTexParameteri(GL\_TEXTURE\_2D,GL\_TEXTURE\_MAG\_FILTER,GL\_LINEAR);

glTexEnvi(GL\_TEXTURE\_ENV,GL\_TEXTURE\_ENV\_MODE,GL\_REPLACE);

glDisable(GL\_NORMALIZE);

glEnable(GL\_BLEND);

glBegin(GL\_QUADS);

glTexCoord2f(0.0f+sin(waterMovement/10.0),1.0f);glVertex2i(1400,550);// right top corner

glTexCoord2f(1.0f+sin(waterMovement/10.0),0.0f);glVertex2i(0,550);// left top corner

glTexCoord2f(1.0f+sin(waterMovement/10.0),1.0f);glVertex2i(0,200);// left bottom corner

glTexCoord2f(0.0f+sin(waterMovement/10.0),0.0f);glVertex2i(1400,200);// right bottom corner

glEnd();

}

void drawGrass(int id)

{

glDisable(GL\_LIGHTING);

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,id);

glTexParameteri(GL\_TEXTURE\_2D,GL\_TEXTURE\_MIN\_FILTER,GL\_LINEAR);

glTexParameteri(GL\_TEXTURE\_2D,GL\_TEXTURE\_MAG\_FILTER,GL\_LINEAR);

glTexEnvi(GL\_TEXTURE\_ENV,GL\_TEXTURE\_ENV\_MODE,GL\_REPLACE);

glDisable(GL\_NORMALIZE);

glEnable(GL\_BLEND);

glBegin(GL\_QUADS);

glTexCoord2f(0.0f+sin(grassMovement/10.0),0.0f);glVertex2i(1400,200); // right top corner

glTexCoord2f(1.0f+sin(grassMovement/10.0),0.0f);glVertex2i(0,200);// left top corner

glTexCoord2f(1.0f+sin(grassMovement/10.0),1.0f);glVertex2i(0,0);// left bottom corner

glTexCoord2f(0.0f+sin(grassMovement/10.0),1.0f);glVertex2i(1400,0);// right bottom corner

glEnd();

}

void animate(void)

{

waterMovement-=0.001;

hillMovement-=0.001;

grassMovement-=0.001;

}

void Start()

{

if(isStart){

Initialize\_Heliopter();

isStart=false;

}

}

void movement()

{

a=downward\_force/100;

b=heli.upward\_force/100;

heli.move+=a-b;

heli.height-=heli.move;

if(heli.move>maxVel)heli.move=maxVel;

if(heli.move<-maxVel)heli.move=-maxVel;

}

void collision()

{

if(heli.height>525|| heli.height<230)

{

gameover=true;

}

}

void display(void)

{

drawHill(t\_hill);

drawWater(t\_water);

drawGrass(t\_grass);

Start();

DrawBoxes();

movement();

Draw(t\_og);

collision();

glFlush();

if(!gameover)

glutPostRedisplay();

}

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

{

if(button==GLUT\_LEFT\_BUTTON && state==GLUT\_DOWN)

heli.upward\_force=2\*downward\_force;

else if(button==GLUT\_LEFT\_BUTTON)

heli.upward\_force=0;

}

void wall()

{

heli.draw\_wall();

}

void mytimer(int t)

{

wall();

glutTimerFunc(50,mytimer,1);

}

void reshape (int w, int h)

{

glViewport (0, 0, (GLsizei) w, (GLsizei) h);

glMatrixMode (GL\_PROJECTION);

glLoadIdentity ();

width=w;

height=h;

gluOrtho2D (0.0, (GLdouble) w, 0.0, (GLdouble) h);

}

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

{

switch(key){

case 27:

exit(0);

break;

}

}

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode (GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize (1300, 750);

glutInitWindowPosition (0, 0);

glutCreateWindow ("Tunnel");

glutFullScreen();

glutTimerFunc(50,mytimer,1);

glutDisplayFunc(display);

glutReshapeFunc(reshape);

glutIdleFunc(animate);

glutMouseFunc(mouse);

glutKeyboardFunc(keyboard);

PlayBackgroundSound("heli-running2.wav");

myInit();

glutMainLoop();

return 0;

}

# REFERENCES

<https://sites.google.com/site/humeratariquok/home>

<https://stackoverflow.com>

<http://web.ics.purdue.edu/~marsh7/helicopter/helicopter.htm>