

**PROJECT REPROT ON**

**Designing an 3D Desktop Table Using OpenGL**

**Course Title: Computer Graphics LAB**

**Course Code: CSE-404**

|  |  |
| --- | --- |
| **Submitted By:** | **Submitted To:** |
| **Fozle Rabbbi**  **173002018**  **Ahsan Ahmend**  **173002010** | **Ahsan Habib**  **Lecturer.**  **Department of CSE**  **Green University of Bangladesh** |

**Drawing Computer Desk Room Setup Using OpenGL:**

**Introduction:**

Introduction- OpenGL provides various viewing function that helps us to develop various views of single object. It also provides different kind of transformation function. We can even rotate the object along desired location and with desired angle on the screen. So here we made a simple house shape in 3d model viewing using OpenGL in code blocks. We made it in cube shape and then we translate and scaled it for the right shape.

Overview of Project- In this project we first construct a ground floor, then we made side wall. After doing this, we a computer table which have 4 legs. Then we add four rooftop side bars, a monitor, ground floor. We add some different color for different shapes in this house. Also, we have tried to add a fan. But for short times we cannot successfully to attached this fan.

**Hardware Used-**

Processor- Intel Core i7 @3.2 GHz.

RAM- 16 GB

Software Requirements-

OS- Windows 10

Application- Code-Blocks 17.12

A Visual C++ compiler is required for compiling the source code to make the

executable file which can be directly executed.

**Designing an 3D Desktop Table**

**Source Code:**

*/\*\**

*\* Title : Build a 3D Computer Desk Room Setup in C++OpenGL(House.cpp)*

*\* Program Description: Write a C++ program to build a house using OpenGL 2D Graphics in C++.*

*\* Author : Fozle Rabbbi(173002018) and Ahsan Ahmend(173002010)*

*\* Interface : Console*

*\* Library : OpenGL*

*\* Environment : GLUT*

*\* IDE : CodeBlocks 17.12*

*\* Operating System : Windows 10*

*\*\*/*

#include<windows.h>

#include <GL/glut.h>

#include <math.h>

#include <stdlib.h>

**static** **float** da = 0, db = 0, dc = 0;

**static** **float** dx = 0, dy = 0, dz = 0;

**static** **float** ex = 0, ey = 0, ez = 30;

**static** **float** cx = 0, cy = 0, cz = 0;

**static** **float** hx = 0, hy = 1, hz = 0;

**static** **float** zx = 1, zy = 1, zz = 1;

**static** **float** degreex = 0, degreey = 0, degreez = 0;

**static** **float** theta = 0;

**static** **int** height = 640, width = 480;

**static** **int** flag = 0, flag1 = 0;

*/\* GLUT callback Handlers \*/*

**static** **void** resize(**int** width, **int** height)

{

**const** **float** ar = (**float**) width / (**float**) height;

glViewport(0, 0, width, height);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glFrustum(-ar, ar, -1.0, 1.0, 2.0, 100.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity() ;

}

**void** cube()

{

glBegin(GL\_QUADS);

glVertex3f(-1,-1, 1);

glVertex3f(1,-1, 1);

glVertex3f(1, 1, 1);

glVertex3f(-1, 1, 1);

glVertex3f(1, 1, 1);

glVertex3f(1,

-1, 1);

glVertex3f(1,-1,-1);

glVertex3f(1, 1,-1);

glVertex3f(-1, 1,-1);

glVertex3f(-1,-1,-1);

glVertex3f(-1,-1, 1);

glVertex3f(-1, 1, 1);

glVertex3f(-1, 1,-1);

glVertex3f(1, 1,-1);

glVertex3f(1,-1,-1);

glVertex3f(-1,-1,-1);

glVertex3f(-1, 1,-1);

glVertex3f(-1, 1, 1);

glVertex3f(1, 1, 1);

glVertex3f(1, 1,-1);

glVertex3f(-1,-1, 1);

glVertex3f(-1,-1,-1);

glVertex3f(1,-1,-1);

glVertex3f(1,-1, 1);

glEnd();

}

**void** circle()

{

**const** **double** t = glutGet(GLUT\_ELAPSED\_TIME) / 1000.0;

**const** **double** a = t\*90.0;

glRotated(-a,0,0,1);

glBegin(GL\_POLYGON);

glColor3f(0,1,0);

**for**(**double** i = 360; i >=180; i--)

{

**double** theta = 2.0f \* 3.1416 \* i / 360;

**double** x = 2 \* cosf(theta);

**double** y = 2 \* sinf(theta);

glVertex3f(x, y, -.1);

}

glEnd();

glBegin(GL\_POLYGON);

glColor3f(1, 0, 0);

**for**(**double** i = 180; i >=0; i--)

{

**double** theta = 2.0f \* 3.1416 \* i / 360;

**double** x = 2 \* cosf(theta);

**double** y = 2 \* sinf(theta);

glVertex3f(x, y, -.1);

}

glEnd();

glBegin(GL\_POLYGON);

glColor3f(1, 0, 0);

**for**(**double** i = 0; i <= 180; i++)

{

**double** theta = 2 \* 3.1416 \* i / 360;

**double** x = 2 \* cosf(theta);

**double** y = 2 \* sinf(theta);

glVertex3f(x, y, .1);

}

glEnd();

glBegin(GL\_POLYGON);

glColor3f(0, 1, 0);

**for**(**double** i = 180; i <=360; i++)

{

**double** theta = 2 \* 3.1416 \* i / 360;

**double** x = 2 \* cosf(theta);

**double** y = 2 \* sinf(theta);

glVertex3f(x, y, .1);

}

glEnd();

glBegin(GL\_QUAD\_STRIP);

*// glColor3f(r, g, b);*

**for**(**double** i = 0; i <= 360; i++)

{

**double** theta = 2 \* 3.1416 \* i / 360;

**double** x = 2 \* cosf(theta);

**double** y = 2 \* sinf(theta);

glVertex3f(x, y, .1);

glVertex3f(x, y, -.1);

}

glEnd();

}

**void** car()

{

glPushMatrix(); *//Main shape*

glColor3d(.2,.8,0);

glScaled(5,1,2);

cube();

glPopMatrix();

glPushMatrix(); *//Upper shape*

glColor3d(0,.4,.6);

glTranslated(.1,1.7,0);

glScaled(2,.7,1.6);

cube();

glPopMatrix();

glPushMatrix(); *//front and back*

glColor3d(1,0,0);

glTranslated(.1,1.6,0);

glScaled(1.6,.6,2);

cube();

glPopMatrix();

glPushMatrix(); *// left and right*

glColor3d(1,0,0);

glTranslated(.1,1.6,0);

glScaled(2.2,.6,1.4);

cube();

glPopMatrix();

glPushMatrix();

glTranslated(-2.5,-.8,2.1);

glScaled(.4,.4,1);

circle();

glPopMatrix();

}

**void** windmill()

{

**const** **double** t = glutGet(GLUT\_ELAPSED\_TIME) / 1000.0;

**const** **double** a = t\*90.0;

glPushMatrix(); *//back left*

glColor3d(.9,.2,0);

glRotated(30,1,0,0);

glRotated(30,0,0,-1);

glScaled(.2,2,.2);

cube();

glPopMatrix();

glPushMatrix(); *// back right*

glColor3d(.0,.8,.4);

glTranslated(1.8,0,0);

glRotated(30,1,0,0);

glRotated(30,0,0,1);

glScaled(.2,2,.2);

cube();

glPopMatrix();

glPushMatrix(); *//front left*

glColor3d(.4,.2,0);

glTranslated(0,0,1.8);

glRotated(30,-1,0,0);

glRotated(30,0,0,-1);

glScaled(.2,2,.2);

cube();

glPopMatrix();

glPushMatrix(); *// front right*

glColor3d(.2,0,.8);

glTranslated(1.8,0,1.8);

glRotated(30,-1,0,0);

glRotated(30,0,0,1);

glScaled(.2,2,.2);

cube();

glPopMatrix();

glPushMatrix(); *//top box*

glColor3d(.6,.8,0);

glTranslated(.9,1.2,.9);

glScaled(.7,.7,.7);

cube();

glPopMatrix();

glPushMatrix(); *//box stick*

glColor3d(.6,.3,.1);

glTranslated(.9,1.2,1.5);

glScaled(.1,.1,.2);

cube();

glPopMatrix();

glPushMatrix(); *//fan 1*

glColor3d(.9,.1,.1);

*//glRotated(-a,0,0,1);*

glTranslated(.9,1.2,1.8);

glRotated(-a\*3,0,0,1);

glScaled(.8,.1,.1);

*// glRotated(-a,0,0,1);*

cube();

glPopMatrix();

glPushMatrix(); *//fan 2*

glColor3d(.9,.1,.1);

*//glRotated(-a,0,0,1);*

glTranslated(.9,1.2,1.8);

glRotated(-a\*3,0,0,1);

glScaled(.1,.8,.1);

*// glRotated(-a,0,0,1);*

cube();

glPopMatrix();

}

**static** **void** display(**void**)

{

**const** **double** t = glutGet(GLUT\_ELAPSED\_TIME) / 1000.0;

**const** **double** a = t\*90.0;

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glTranslated(dx,dy,dz);

glRotated(da,1,0,0);

glRotated(db,0,1,0);

glRotated(dc,0,0,1);

glScaled(zx,zy,zz);

*//car();*

*//windmill();*

glPushMatrix(); *//left side wall*

glColor3d(0.40,0.25,0.25);

glTranslatef(-5, 0, 0);

glScaled(.1, 3, 4);

cube();

glPopMatrix();

glPushMatrix(); *//back side wall*

glColor3d(0.40,0.25,0.25);

glTranslatef(0,0,-4);

glScaled(5,3,.1);

cube();

glPopMatrix();

glPushMatrix(); *//down floor*

glColor3d(0.0, 1.0, 1.0);

glTranslatef(0, -3, 0);

glScaled(5,0,4 );

cube();

glPopMatrix();

glPushMatrix(); *//table*

glColor3d(0.60, 0.40, 0.12);

glTranslatef(-.9, -.7, 0);

glScaled(2, 0.09, 1);

cube();

glPopMatrix();

glPushMatrix(); *//table leg back right side*

glColor3d(0.25, 0.25, 0.25); *//dark gray*

glTranslatef(1, -1.80, -0.7);

glScaled(0.1, 1, 0.1);

cube();

glPopMatrix();

glPushMatrix(); *//table leg back right side*

glColor3d(0.25, 0.25, 0.25); *//dark gray*

glTranslatef(1, -1.80, 0.7);

glScaled(0.1, 1, 0.1);

cube();

glPopMatrix();

glPushMatrix(); *// table leg front left size*

glColor3d(0.25, 0.25, 0.25);

glTranslatef(-2.8, -1.80, 0.7);

glScaled(0.1, 1, 0.1);

cube();

glPopMatrix();

glPushMatrix(); *// table leg back left size*

glColor3d(0.25, 0.25, 0.25);

glTranslatef(-2.8, -1.80, -0.7);

glScaled(0.1, 1, 0.1);

cube();

glPopMatrix();

glPushMatrix(); *//monitor*

glColor3d(1, 1, 1);

glTranslatef(-1, -.003, 0.2);

glScaled(0.9,0.5,0.04);

cube();

glPopMatrix();

glPushMatrix(); *//monitor stand*

glColor3d(.2,.3,.1);

glTranslatef(-1,-0.65,0);

glScaled(0.2,0.05,0.1);

cube();

glPopMatrix();

glPushMatrix(); *//stand stick*

glColor3d(.2,1,0);

glTranslated(-1.0, -0.65, 0);

glScaled(.05,-0.1,.3);

cube();

glPopMatrix();

glPushMatrix(); *//fan 1*

glColor3d(.9,.1,.1);

*//glRotated(-a,0,0,1);*

glTranslated(1.9,0,1.8);

glRotated(-a\*3,0,0,1);

glScaled(.8,.1,.1);

cube();

glPopMatrix();

glPushMatrix(); *//Fan Stick*

glColor3d(0.60, 0.40, 0.70);

glTranslated(2, 0, -.05);

glScaled(.8,2,.1);

cube();

glPopMatrix();

glPushMatrix(); *//box stick*

glColor3d(.6,.3,.1);

glTranslated(.9,1.2,1.5);

glScaled(.1,.1,.2);

cube();

glPopMatrix();

*//*

*// glPushMatrix();*

*// glTranslated()*

*// glPopMatrix();*

*// glPushMatrix();*

*// glColor3d(0.25, 0.25, 0.25);*

*// glTranslatef(1, 1.8, 1);*

*// glScaled(.8,-22,1);*

*// glPopMatrix();*

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity() ;

gluLookAt(ex,ey,ez,cx,cy,cz,hx,hy,hz);

glutSwapBuffers();

}

**static** **void** key(**unsigned** **char** key, **int** x, **int** y)

{

**switch** (key)

{

**case** 27 :

**case** 'q':

exit(0);

**break**;

**case** '+':

zx +=.05;

zy +=.05;

zz +=.05;

**break**;

**case** '-':

zx -=.05;

zy -=.05;

zz -=.05;

**break**;

**case** 'a':

da +=5;

**break**;

**case** 'b':

db +=5;

**break**;

**case** 'c':

dc +=5;

**break**;

**case** 'd':

da -=5;

**break**;

**case** 'e':

db -=5;

**break**;

**case** 'f':

dc -=5;

**break**;

**case** 'x':

dx +=.5;

dy = 0;

dz = 0;

**break**;

**case** 'y':

dx = 0;

dy +=.5;

dz = 0;

**break**;

**case** 'z':

dx = 0;

dy = 0;

dz +=.5;

**break**;

**case** 'i':

dx -=.5;

dy = 0;

dz = 0;

**break**;

**case** 'j':

dx = 0;

dy -=.5;

dz = 0;

**break**;

**case** 'k':

dx = 0;

dy = 0;

dz -=.5;

**break**;

}

glutPostRedisplay();

}

**static** **void** idle(**void**)

{

glutPostRedisplay();

}

*/\* Program entry point \*/*

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

{

glutInit(&argc, argv);

glutInitWindowSize(640,480);

glutInitWindowPosition(10,10);

glutInitDisplayMode(GLUT\_RGB | GLUT\_DOUBLE | GLUT\_DEPTH);

glutCreateWindow("GLUT Shapes");

glutReshapeFunc(resize);

glutDisplayFunc(display);

glutKeyboardFunc(key);

glutIdleFunc(idle);

glClearColor(1,1,1,1);

glEnable(GL\_CULL\_FACE);

glCullFace(GL\_BACK);

glEnable(GL\_DEPTH\_TEST);

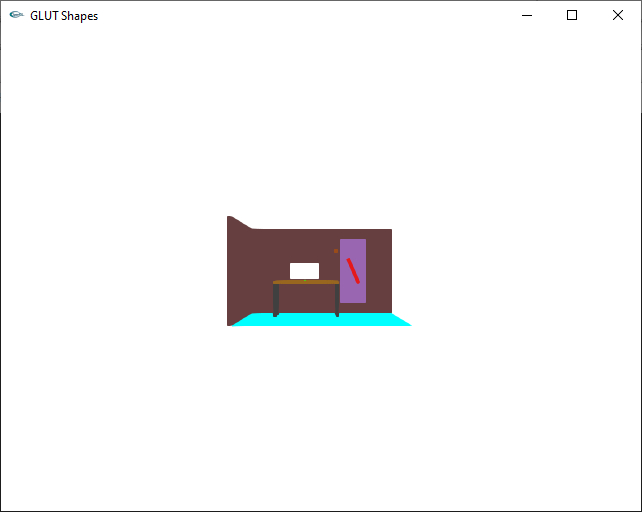
glDepthFunc(GL\_LESS);

glutMainLoop();

**return** EXIT\_SUCCESS;

}

**Output:**

****