COMSATS UNIVERSITY ISLAMABAD
Lahore Campus
Department of Computer Science

Computer Graphics (CSD 304) LAB

Week 8 Lab 2

We completed the following task in previous lab,

Task 2:

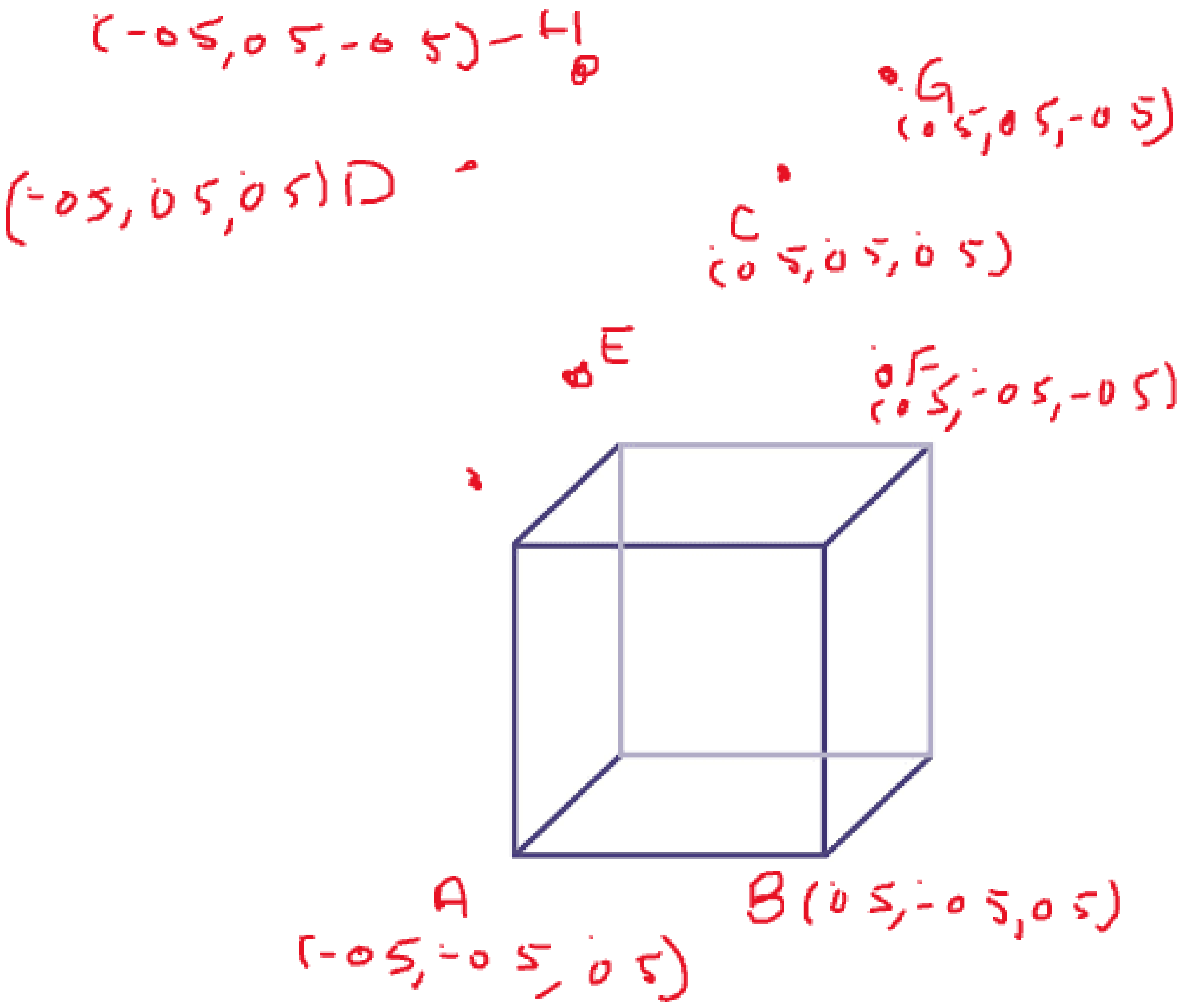
Draw a unit wire cube and control its x and y axis rotation by up and down arrow keys(x-axis rotation) , and left and right arrow keys (for y-axis rotation).

You can use code for this previous task for current lab

We wish to make our own cube , which we will construct by joining together individual square faces

Suppose we wish to make a cube whose edges along the x,y and z-axis all extend from -0.5 to 0.5

So the cube should have the vertices as labelled in the following diagram:



Vertices on front face ABCD
Vertices on Back face EFGH

Now, do the following

1. Comment out the glutWireCube API, in its place write code for steps below
2. Draw the front face with vertices ABCD (use GL_QUADS , and specify vertices in order ABCD) , give it a unique color
3. Draw right face , specify vertices in order BFGC, give it a unique color
4. Draw back face , specify vertices in order EHGF, give it a unique color
5. Draw left face , specify vertices in order ADHE, give it a unique color
6. Draw top face , specify vertices in order CGHD, give it a unique color
7. Draw bottom face , specify vertices in order AEFB, give it a unique color

---

Solution:

```
//
//  main.cpp
//  making_cube_using_quads
//
//  Created by Amaid Zia on 11/18/18.
//  Copyright © 2018 Amaid Zia. All rights reserved.
//

// #include <iostream>
// #include <OpenGL/glu.h>
// #include <openGL/gl.h>
// #include <GLUT/GLUT.h>

#include <GL/glut.h>

#include <stdlib.h>

//****************** Globals
*****************************************************
GLfloat theta = 0.0;
double rotate_y=0;
double rotate_x=0;


GLfloat vertices [8][3] = {
                                {-1.0 ,-1.0 ,-1.0 },
                                { 1.0 ,-1.0 ,-1.0 },
                                { 1.0 , 1.0 ,-1.0 },
                                {-1.0 , 1.0 ,-1.0 },
                                {-1.0 ,-1.0 , 1.0 },
                                { 1.0 ,-1.0 , 1.0 },
                                { 1.0 , 1.0 , 1.0 },
                                {-1.0 , 1.0 , 1.0 },
                            };


GLfloat normals [8][3] = {
                            {-1.0 ,-1.0 ,-1.0 },
                            { 1.0 ,-1.0 ,-1.0 },
                            { 1.0 , 1.0 ,-1.0 },
                            {-1.0 , 1.0 ,-1.0 },
                            {-1.0 ,-1.0 , 1.0 },
                            { 1.0 ,-1.0 , 1.0 },
                            { 1.0 , 1.0 , 1.0 },
                            {-1.0 , 1.0 , 1.0 },
                        };


GLfloat colors [8][3] = {
                            {0.0,0.0,0.0},
                            {0.0,0.0,1.0},
                            {0.0,1.0,0.0},
                            {0.0,1.0,1.0},
                            {1.0,0.0,0.0},
                            {1.0,0.0,1.0},
                            {1.0,1.0,0.0},
                            {0.5,0.0,0.25},
                        };


//*************************************************************************
******
//*************************************************************************
******
void draw_quad (int a,int b,int c,int d)
{
    glBegin(GL_QUADS);
        glColor3fv(colors[a]);
        // glNormal3fv(normals[a]);
        glVertex3fv(vertices[a]);

        glColor3fv(colors[b]);
        // glNormal3fv(normals[b]);
        glVertex3fv(vertices[b]);

        glColor3fv(colors[c]);
        // glNormal3fv(normals[c]);
        glVertex3fv(vertices[c]);

        glColor3fv(colors[d]);
        // glNormal3fv(normals[d]);
        glVertex3fv(vertices[d]);
    glEnd();
}
```

```
//**********************************************************************
******
void draw_color_cube ()
{
//    glColor3fv(colors[0]);
    draw_quad(0, 3, 2, 1);

//    glColor3fv(colors[1]);
    draw_quad(2, 3, 7, 6);

//    glColor3fv(colors[2]);
    draw_quad(0, 4, 7, 3);

//    glColor3fv(colors[3]);
    draw_quad(1, 2, 6, 5);

//    glColor3fv(colors[4]);
    draw_quad(4, 5, 6, 7);

//    glColor3fv(colors[5]);
    draw_quad(0, 1, 5, 4);

}

//**********************************************************************
******

void myinit() //set attributes
{
    glClearColor(1.0f,1.0f,1.0f,1.0f); //  setting background color
    glColor3f(0.0f,1.0f,0.0f); // drawing color

    // Set world coordinates
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
//    gluOrtho2D(-3,3,-3,3);
    glOrtho(-3.0, 3.0, -3.0, 3.0, -3.0, 3.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    //   glOrtho(0.0, 5.0, 0.0, 5.0, 0.0, 1.0);
}

//**********************************************************************
******
```

```
void display()
{
    // initializations of variables etc
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT); // clear the
background

    /*
     Drawing
     */

    glPushMatrix();
    glLoadIdentity();

//______________________________________

    // Rotate when user changes rotate_x and rotate_y
    glRotatef( rotate_x, 1.0, 0.0, 0.0 );
    glRotatef( rotate_y, 0.0, 1.0, 0.0 );
//    glScalef(2.0, 2.0, 2.0);
//______________________________________

    //    glRotatef(180.0, 0.0, 1.0, 0.0);

//    glRotatef(theta, 0.0, 1.0, 0.0);
//    glRotatef(30.0, 1.0, 0.0, 0.0);

//    glutWireCube(1);
    draw_color_cube();
    glPopMatrix();
    glFlush();

    glutSwapBuffers();
}

//*********************************************************************
******

//void Rotate_func (int v)
//{
//
//    theta = theta + 10.0;
//    if (theta >= 360.0)
//        theta = 0.0;
//    glutPostRedisplay();
//    glutTimerFunc(50, Rotate_func, v);
```

```
//
//}

//**************************************************************************
******

// ----------------------------------------------------------
// specialKeys() Callback Function
// ----------------------------------------------------------
void specialKeys( int key, int x, int y )
{

    //  Right arrow - increase rotation by 5 degree
    if (key == GLUT_KEY_RIGHT)
        rotate_y += 5;

    //  Left arrow - decrease rotation by 5 degree
    else if (key == GLUT_KEY_LEFT)
        rotate_y -= 5;

    else if (key == GLUT_KEY_UP)
        rotate_x += 5;

    else if (key == GLUT_KEY_DOWN)
        rotate_x -= 5;

    //  Request display update
    glutPostRedisplay();

}


//**************************************************************************
******

int main (int argc , char ** argv)
{
    glutInit(&argc,argv); // initialize GLUT
    glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB|GLUT_DEPTH); //setting
display mode

    glutInitWindowSize(500,500); // window size
    glutInitWindowPosition(0,0);  // window position

    glutCreateWindow("color_cube_modelling");
    glEnable(GL_DEPTH_TEST);


    glutDisplayFunc(display); // Register callback func
    glutSpecialFunc(specialKeys);
//    glutTimerFunc(2000, Rotate_func, 0);
    myinit(); // Set attributes

    glutMainLoop(); // enter event Loop

    return 0;
}
```