Develop a program to demonstrate 3D transformation on basic objects

#include <stdlib.h>

#include <GL/glut.h>

GLfloat vertices[][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[][3] = {{0.0,0.0,0.0},{1.0,0.0,0.0},

{1.0,1.0,0.0}, {0.0,1.0,0.0}, {0.0,0.0,1.0},

{1.0,0.0,1.0}, {1.0,1.0,1.0}, {0.0,1.0,1.0}};

void polygon(int a, int b, int c , int d)

{

glBegin(GL\_POLYGON);

glColor3fv(colors[a]);

glVertex3fv(vertices[a]);

glColor3fv(colors[b]);

glVertex3fv(vertices[b]);

glColor3fv(colors[c]);

glVertex3fv(vertices[c]);

glColor3fv(colors[d]);

glVertex3fv(vertices[d]);

glEnd();

}

void colorcube()

{

polygon(0,3,2,1);

polygon(2,3,7,6);

polygon(0,4,7,3);

polygon(1,2,6,5);

polygon(4,5,6,7);

polygon(0,1,5,4);

}

static GLfloat theta[] = {0.0,0.0,0.0};

static GLint axis = 2;

static GLdouble viewer[]= {0.0, 0.0, 5.0}; /\* initial viewer location \*/

void display(void)

{

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

/\* Update viewer position in modelview matrix \*/

glLoadIdentity();

gluLookAt(viewer[0],viewer[1],viewer[2], 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

/\* rotate cube \*/

glRotatef(theta[0], 1.0, 0.0, 0.0);

glRotatef(theta[1], 0.0, 1.0, 0.0);

glRotatef(theta[2], 0.0, 0.0, 1.0);

colorcube(); /\* draw the rotated color cube \*/

glFlush();

glutSwapBuffers();

}

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

{

if(btn==GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN) axis = 0;

if(btn==GLUT\_MIDDLE\_BUTTON && state == GLUT\_DOWN) axis =1; if(btn==GLUT\_RIGHT\_BUTTON && state == GLUT\_DOWN) axis = 2;

theta[axis] += 4.0;

if( theta[axis] > 360.0 ) theta[axis] -= 360.0;

display();

}

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

{

/\* Use x, X, y, Y, z, and Z keys to move viewer \*/

if(key == 'x') viewer[0]-= 1.0;

if(key == 'X') viewer[0]+= 1.0;

if(key == 'y') viewer[1]-= 1.0;

if(key == 'Y') viewer[1]+= 1.0;

if(key == 'z') viewer[2]-= 1.0;

if(key == 'Z') viewer[2]+= 1.0;

display();

}

void myReshape(int w, int h)

{

glViewport(0, 0, w, h);

/\* Use a perspective view \*/

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

if(w<=h) glFrustum(-2.0, 2.0, -2.0 \* (GLfloat) h/ (GLfloat) w,

2.0\* (GLfloat) h / (GLfloat) w, 2.0, 20.0);

else glFrustum(-2.0, 2.0, -2.0 \* (GLfloat) w/ (GLfloat) h,

2.0\* (GLfloat) w / (GLfloat) h, 2.0, 20.0);

/\* Or we can use gluPerspective that is gluPerspective(45.0, w/h, -10.0, 10.0); \*/

glMatrixMode(GL\_MODELVIEW);

}

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

{

glutInit(&argc, argv);

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

glutInitWindowSize(500, 500);

glutCreateWindow("Colorcube Viewer");

glutReshapeFunc(myReshape);

glutDisplayFunc(display);

glutMouseFunc(mouse);

glutKeyboardFunc(keys);

glEnable(GL\_DEPTH\_TEST);

glutMainLoop();

}

 