#include <GL/glut.h>

#include<math.h>

typedef struct V

{ float x, y, z;

};

void resize(int, int);

V translate(V point, V offset); void display();

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

{

glutInit(&argc,argv); //initialize GLUT

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);//initialize display mode

glutInitWindowSize(400,400); //set display-window width & height

glutInitWindowPosition(50,50); //set display-window upper-left position

glutCreateWindow("Cube using translation"); //create display-window

glutDisplayFunc(display); //call graphics to be displayed on the window

glutReshapeFunc(resize);

glutMainLoop(); //display everything and wait

}

void resize(int w, int h)

{

glMatrixMode(GL\_MODELVIEW); //set projection paramaters

glLoadIdentity();

//change first three values to see the cube from different directions

gluLookAt(3,3,6, 0,0, 0, 0, 1, 0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity(); //reset to identity matrix

gluPerspective(45,(float)w/h, 1, 100);

glViewport(0,0,w,h);

}

V translate(V point, V offset)

{ point.x +=offset.x; point.y +=offset.y;

point.z +=offset.z;

return point;

}

void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(1.0,0.0,0.0);

glMatrixMode(GL\_MODELVIEW);

V p0 = { -0.5, -0.5, -0.5};

V amountX = {1, 0, 0}; //matrix values needed for translation

V amountY ={0, 1, 0};

V amountZ = {0, 0, 1};

V p1 = translate(p0, amountX);

V p2 = translate(p1, amountY);

V p3 = translate(p0, amountY);

glBegin(GL\_LINE\_LOOP); //draw one surface of cube

glVertex3f(p0.x, p0.y, p0.z);

glVertex3f(p1.x, p1.y, p1.z);

glVertex3f(p2.x, p2.y, p2.z);

glVertex3f(p3.x, p3.y, p3.z);

glEnd();

//calculate and draw second surface parallel to first surface using translation

V p4 = translate(p0, amountZ);

V p5 = translate(p1, amountZ);

V p6 = translate(p2, amountZ);

V p7 = translate(p3, amountZ);

glBegin(GL\_LINE\_LOOP);

glVertex3f(p4.x, p4.y, p4.z);

glVertex3f(p5.x, p5.y, p5.z);

glVertex3f(p6.x, p6.y, p6.z);

glVertex3f(p7.x, p7.y, p7.z);

glEnd();

glBegin(GL\_LINES);

glVertex3f(p0.x, p0.y, p0.z);

glVertex3f(p4.x, p4.y, p4.z);

glVertex3f(p1.x, p1.y, p1.z);

glVertex3f(p5.x, p5.y, p5.z);

glVertex3f(p2.x, p2.y, p2.z);

glVertex3f(p6.x, p6.y, p6.z);

glVertex3f(p3.x, p3.y, p3.z);

glVertex3f(p7.x, p7.y, p7.z);

glEnd();

glFinish();

}

