3).OpenGl example

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <GL/glut.h>

/\* function declarations \*/

void

drawScene(void), setMatrix(void), initLightAndMaterial(void),

animation(void), resize(int w, int h), menu(int choice), keyboard(unsigned char c, int x, int y);

/\* global variables \*/

float ax, ay, az; /\* angles for animation \*/

GLUquadricObj \*quadObj; /\* used in drawscene \*/

static float lmodel\_twoside[] =

{GL\_TRUE};

static float lmodel\_oneside[] =

{GL\_FALSE};

int

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

{

glutInit(&argc, argv);

quadObj = gluNewQuadric(); /\* this will be used in drawScene

\*/

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

glutCreateWindow("Two-sided lighting");

ax = 10.0;

ay = -10.0;

az = 0.0;

initLightAndMaterial();

glutDisplayFunc(drawScene);

glutReshapeFunc(resize);

glutCreateMenu(menu);

glutAddMenuEntry("Motion", 3);

glutAddMenuEntry("Two-sided lighting", 1);

glutAddMenuEntry("One-sided lighting", 2);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

glutKeyboardFunc(keyboard);

glutMainLoop();

return 0; /\* ANSI C requires main to return int. \*/

}

void

drawScene(void)

{

glClearColor(0.0, 0.0, 0.0, 0.0);

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

glPushMatrix();

gluQuadricDrawStyle(quadObj, GLU\_FILL);

glColor3f(1.0, 1.0, 0.0);

glRotatef(ax, 1.0, 0.0, 0.0);

glRotatef(-ay, 0.0, 1.0, 0.0);

gluCylinder(quadObj, 2.0, 5.0, 10.0, 20, 8); /\* draw a cone \*/

glPopMatrix();

glutSwapBuffers();

}

void

setMatrix(void)

{

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-15.0, 15.0, -15.0, 15.0, -10.0, 10.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

int count = 0;

void

animation(void)

{

ax += 5.0;

ay -= 2.0;

az += 5.0;

if (ax >= 360)

ax = 0.0;

if (ay <= -360)

ay = 0.0;

if (az >= 360)

az = 0.0;

drawScene();

count++;

if (count >= 60)

glutIdleFunc(NULL);

}

/\* ARGSUSED1 \*/

void

keyboard(unsigned char c, int x, int y)

{

switch (c) {

case 27:

exit(0);

break;

default:

break;

}

}

void

menu(int choice)

{

switch (choice) {

case 3:

count = 0;

glutIdleFunc(animation);

break;

case 2:

glLightModelfv(GL\_LIGHT\_MODEL\_TWO\_SIDE, lmodel\_oneside);

glutSetWindowTitle("One-sided lighting");

glutPostRedisplay();

break;

case 1:

glLightModelfv(GL\_LIGHT\_MODEL\_TWO\_SIDE, lmodel\_twoside);

glutSetWindowTitle("Two-sided lighting");

glutPostRedisplay();

break;

}

}

void

resize(int w, int h)

{

glViewport(0, 0, w, h);

setMatrix();

}

void

initLightAndMaterial(void)

{

static float ambient[] =

{0.1, 0.1, 0.1, 1.0};

static float diffuse[] =

{0.5, 1.0, 1.0, 1.0};

static float position[] =

{90.0, 90.0, 150.0, 0.0};

static float front\_mat\_shininess[] =

{60.0};

static float front\_mat\_specular[] =

{0.2, 0.2, 0.2, 1.0};

static float front\_mat\_diffuse[] =

{0.5, 0.5, 0.28, 1.0};

static float back\_mat\_shininess[] =

{60.0};

static float back\_mat\_specular[] =

{0.5, 0.5, 0.2, 1.0};

static float back\_mat\_diffuse[] =

{1.0, 0.9, 0.2, 1.0};

static float lmodel\_ambient[] =

{1.0, 1.0, 1.0, 1.0};

setMatrix();

glEnable(GL\_DEPTH\_TEST);

glDepthFunc(GL\_LEQUAL);

glLightfv(GL\_LIGHT0, GL\_AMBIENT, ambient);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, diffuse);

glLightfv(GL\_LIGHT0, GL\_POSITION, position);

glMaterialfv(GL\_FRONT, GL\_SHININESS, front\_mat\_shininess);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, front\_mat\_specular);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, front\_mat\_diffuse);

glMaterialfv(GL\_BACK, GL\_SHININESS, back\_mat\_shininess);

glMaterialfv(GL\_BACK, GL\_SPECULAR, back\_mat\_specular);

glMaterialfv(GL\_BACK, GL\_DIFFUSE, back\_mat\_diffuse);

glLightModelfv(GL\_LIGHT\_MODEL\_AMBIENT, lmodel\_ambient);

glLightModelfv(GL\_LIGHT\_MODEL\_TWO\_SIDE, lmodel\_twoside);

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glShadeModel(GL\_SMOOTH);

}

Output:

