ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| асп. |  |  |  | Д.А. Смолиенко |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №4 |
| РАБОТА С БУФЕРОМ ГЛУБИНЫ, ПРОЗРАЧНОСТЬЮ |
| по дисциплине:  КОМПЬЮТЕРНАЯ ГРАФИКА |
|  |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4631 |  |  |  | С.А. Гришин |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2018

**Задание на лабораторную работу**

Вывести несколько пересекающихся объектов, как минимум один из них должен быть прозрачным. Включить 3 источника света, задать свойства поверхностей и источников. Организовать раздельное вращение источников света вокруг неподвижной сцены, управляемое с клавиатуры или мышью.

**Текст программы**

#include <GL/glut.h>

#include <stdlib.h>

void myCubes()

{

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_AMBIENT, mat\_ambient);

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE, mat\_dif);

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, mat\_spec);

glMaterialf(GL\_FRONT\_AND\_BACK, GL\_SHININESS, shininess);

glPushMatrix();

glutSolidCube(1);

glTranslatef(-0.5, -0.5, -0.5);

glutSolidCube(1);

glPopMatrix();

}

void light(void)

{

GLfloat light\_ambient[] = { 0.0, 0.0, 0.0 };

GLfloat light\_diffuse1[] = { 1.0, 1.0, 1.0 };

GLfloat light\_specular1[] = { 1.0, 0.0, 1.0 };

GLfloat light\_diffuse2[] = { 1.0, 1.0, 1.0 };

GLfloat light\_specular2[] = { 0.0, 1.0, 1.0 };

GLfloat light\_diffuse3[] = { 1.0, 1.0, 0.0 };

GLfloat light\_specular3[] = { 1.0, 1.0, 0.0 };

GLfloat light\_position1[] = { 100.0, 0.0, 0.0, 0.0 };

GLfloat light\_position2[] = { -100.0, 0.0, 0.0, 0.0 };

GLfloat light\_position3[] = { 100.0, 100.0, 100.0, 0.0 };

glPushMatrix();

glRotatef(x1\_angle, 1.0, 0.0, 0.0);

glRotatef(y1\_angle, 0.0, 1.0, 0.0);

glLightfv(GL\_LIGHT0, GL\_AMBIENT, light\_ambient);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light\_diffuse1);

glLightfv(GL\_LIGHT0, GL\_SPECULAR, light\_specular1);

glLightfv(GL\_LIGHT0, GL\_POSITION, light\_position1);

glPopMatrix();

glPushMatrix();

glRotatef(x2\_angle, 1.0, 0.0, 0.0);

glRotatef(y2\_angle, 0.0, 1.0, 0.0);

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light\_ambient);

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light\_diffuse2);

glLightfv(GL\_LIGHT1, GL\_SPECULAR, light\_specular2);

glLightfv(GL\_LIGHT1, GL\_POSITION, light\_position2);

glPopMatrix();

glPushMatrix();

glRotatef(x3\_angle, 1.0, 0.0, 0.0);

glRotatef(y3\_angle, 0.0, 1.0, 0.0);

glLightfv(GL\_LIGHT2, GL\_AMBIENT, light\_ambient);

glLightfv(GL\_LIGHT2, GL\_DIFFUSE, light\_diffuse3);

glLightfv(GL\_LIGHT2, GL\_SPECULAR, light\_specular3);

glLightfv(GL\_LIGHT2, GL\_POSITION, light\_position3);

glPopMatrix();

glEnable(GL\_DEPTH\_TEST);

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glEnable(GL\_LIGHT1);

glEnable(GL\_LIGHT2);

}

void display(void)

{

light();

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

glEnable(GL\_BLEND);

glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA);

glPushMatrix();

glRotatef(x\_angle, 1.0, 0.0, 0.0);

glRotatef(y\_angle, 0.0, 1.0, 0.0);

glRotatef(45.0, 1.0, 1.0, 1.0);

myCubes();

glPopMatrix();

glFlush();

}

void reshape(int w, int h)

{

glViewport(0, 0, (GLsizei)w, (GLsizei)h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(40.0, (GLfloat)w / h, 1, 100.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

gluLookAt(0.0f, 0.0f, 9.0f, 0.0f, 0.0f, 0.0f, 0.0f, 100.0f, 0.0f);

}

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

{

if (active\_light == 0) {

switch (key) {

case 'd':

y1\_angle += h;

break;

}

}

switch (key) {

case '1':

active\_light = 0;

break;

case '2':

active\_light = 1;

break;

case '3':

active\_light = 2;

break;

}

glutPostRedisplay();

}

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

{

glutInit(&argc, argv);

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

glutInitWindowSize(600, 600);

glutCreateWindow("LR4");

glutReshapeFunc(reshape);

glutDisplayFunc(display);

glutSpecialFunc(specialKeys);

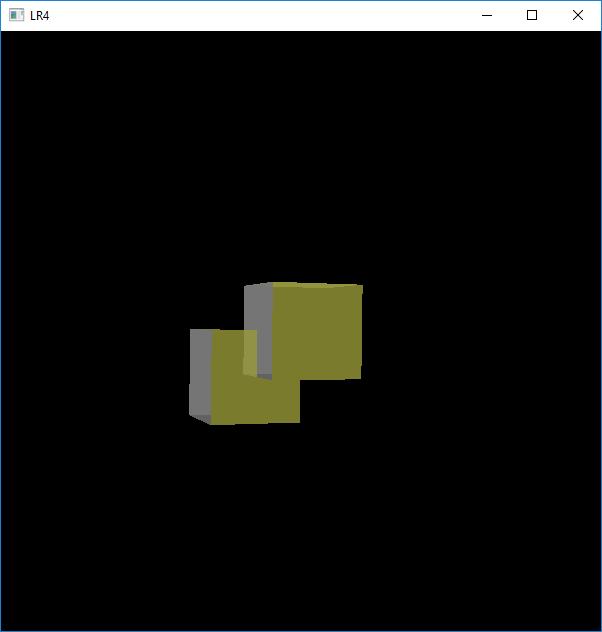
glutKeyboardFunc(keyboard);

glutMainLoop();

return 0;

}

**Результат работы программы**



**Выводы**

В результате лабораторной работы разработано приложение выводящее несколько пересекающихся объектов. Включены 3 источника света, заданы свойства поверхностей и источников. Организовано раздельное вращение источников света вокруг неподвижной сцены, управляемое с клавиатуры.