ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

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

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| РАБОТА С ИСТОЧНИКОМ СВЕТА И СВОЙСТВАМИ  ИСТОЧНИКОВ И МАТЕРИАЛОВ |
| по дисциплине:  КОМПЬЮТЕРНАЯ ГРАФИКА |
|  |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

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

Санкт-Петербург 2018

**Задание на лабораторную работу**

Включить 3 источника света, задать отражающие свойства поверхностей, положение и цвет источников света. Рекомендуется цвет поверхностей сделать одинаковым, а отражающие свойства – разными. Вывести несколько объемных объектов. Источники света должны иметь различный цвет. Необходимо организовать вращение сцены, управляемое с клавиатуры (сцена с объектами объекты вращается, источники неподвижны)

**Текст программы**

#include <GL/glut.h>

#include <stdlib.h>

void init(void)

{

  GLfloat light\_ambient[] = { 1.0, 0.0, 0.0 }

  GLfloat light\_diffuse1[] = { 0.0, 5.0, 1.0

  GLfloat light\_specular1[] = { 5.0, 0.0, 1.0 };

  GLfloat light\_diffuse2[] = { 5.0, 1.0, 0.0

  GLfloat light\_specular2[] = { 0.0, 1.0, 5.0 };

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

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

double rotate\_y = 0;

double rotate\_x = 0;

float shininess = { 80.0 };

float mat\_ambient[] = { 0.2, 0.2, 0.2, 1.0 };

float mat1\_dif[] = { 0.2, 0.2, 0.2 };

float mat1\_spec[] = { 0.8, 0.8, 0.8 };

float mat2\_dif[] = { 1.0, 1.0, 1.0 };

float mat2\_spec[] = { 1.0, 1.0, 1.0 };

float mat3\_dif[] = { 0.5, 0.5, 0.5 };

float mat3\_spec[] = { 0.8, 0.8, 0.8 };

float mat4\_dif[] = { 0.5, 0.5, 0.5 };

float mat4\_spec[] = { 0.0, 0.0, 0.0 };

  glPushMatrix();

  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();

  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();

  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)

{

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

  glPushMatrix();

  glRotatef(20.0, 1.0, 0.0, 0.0);

  glRotatef(rotate\_x, 1.0, 0.0, 0.0);

  glRotatef(rotate\_y, 0.0, 1.0, 0.0);

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

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE, mat1\_dif);

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, mat1\_spec);

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

  glPushMatrix();

  glTranslatef(-0.75, 0.5, 0.0);

  glRotatef(90.0, 1.0, 0.0, 0.0);

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

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE, mat2\_dif);

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, mat2\_spec);

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

  glPushMatrix();

  glTranslatef(-0.75, -0.5, 0.0);

  glRotatef(270.0, 1.0, 0.0, 0.0);

  glutSolidCone(1.0, 2.0, 100, 100);

  glPopMatrix();

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

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE, mat3\_dif);

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, mat3\_spec);

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

  glPushMatrix();

  glTranslatef(1.5, 0.0, -1.0);

  glutSolidSphere(1.0, 100, 100);

  glPopMatrix();

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

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE, mat4\_dif);

  glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, mat4\_spec);

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

  glPushMatrix();

  glTranslatef(1.0, 2.0, -1.0);

  glRotatef(45.0, 1.0, 1.0, 0.0);

  glutSolidCube(1.0);

  glPopMatrix();

  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)

{

  switch (key) {

  case 27:

    exit(0);

    break;

  }

}

void specialKeys(int key, int x, int y) {

  if (key == GLUT\_KEY\_RIGHT)

    rotate\_y += 3;

  else if (ey == GLUT\_KEY\_LEFT)

    rotate\_y -= 3;

  else if (key == GLUT\_KEY\_UP)

    rotate\_x -= 3;

  else if (key == GLUT\_KEY\_DOWN)

    rotate\_x += 3;

  glutPostRedisplay();

}

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

{

  glutInit(&argc, argv);

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

  glutInitWindowSize(500, 500);

  glutCreateWindow("CG LR3");

  init();

  glutReshapeFunc(reshape);

  glutDisplayFunc(display);

  glutSpecialFunc(specialKeys);

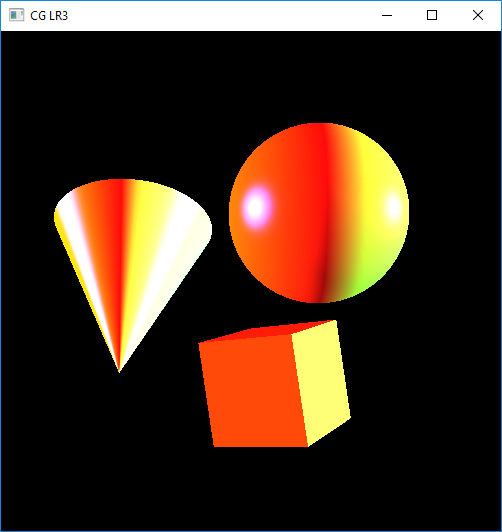
  glutKeyboardFunc(keyboard);

  glutMainLoop();

  return 0;

}

**Результат работы программы**



**Выводы**

В результате лабораторной работы была разработана программа, выводящая трёхмерную сцену с несколькими объёмными фигурами. Включены 3 источника света, заданы отражающие свойства поверхностей, положение и цвет источников света. Организовано вращение сцены, управляемое с клавиатуры, также есть возможность включать и выключать источники света с клавиатуры.