МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего профессионального образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

кандидат технических наук, доцент		С.В.Щекин
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

№2

«Работа с источником света и свойствами материала поверхности»

по дисциплине: КОМПЬЮТЕРНАЯ ГРАФИКА

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. Z7431	20.12.2019	М.Д.Семочкин
	подпись, дата	инициалы, фамилия

Студ. билет 2014/1054

Санкт-Петербург

2019

Содержание

Задание

Включить источник света, задать отражающие свойства поверхностей: диффузное, зеркальное отражение, цвет поверхности.

Вывести несколько объемных объектов. Каждый из них должен иметь различные свойства поверхностей (доминирующее зеркальное или диффузное отражение, различный цвет внешних поверхностей).

Используемые инструменты разработки

IDE – Microsoft Visual C++ 2010 Express

Библиотека для работы с openGL - freeglut 3.0.0 MinGW

ОС – Windows 7 x64

Описание программы

Написанная программа рисует три сферы, каждая из которых имеет различные параметры:

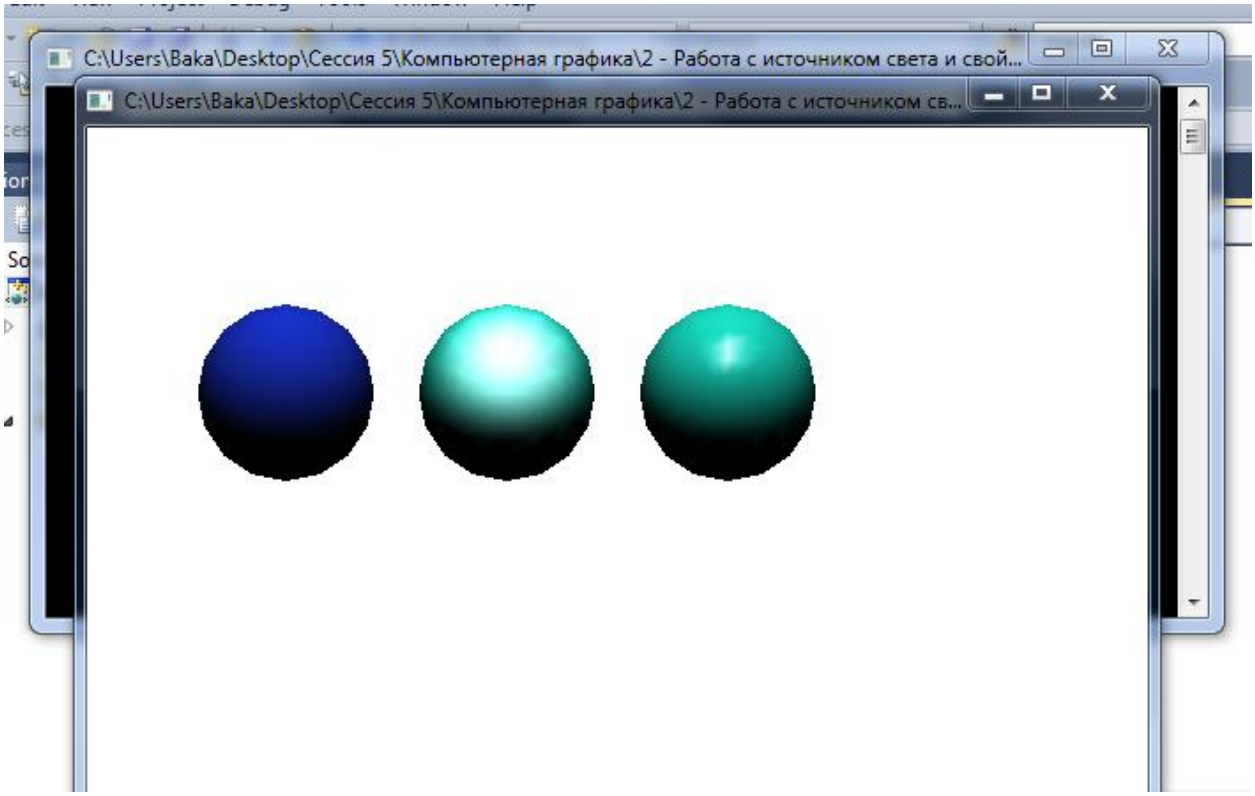
1) Только диффузное отражение, синий цвет
2) Диффузное и доминирующее зеркальное отражение, зеленый цвет
3) Диффузное и зеркальное отражение, зеленый цвет

Текст программы

```
// Программа рисует три сферы с разными параметрами поверхности

#include <stdlib.h>
#include <GL/freeglut.h>


void init(void) {

   GLfloat ambient[] = { 0.0, 0.0, 0.0, 1.0 };
   GLfloat diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat specular[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat position[] = { 0.0, 3.0, 2.0, 0.0 };
   GLfloat lmodel_ambient[] = { 0.4, 0.4, 0.4, 1.0 };
   GLfloat local_view[] = { 0.0 };

   glClearColor(5.0, 5.0, 5.0, 0.0);
   glEnable(GL_DEPTH_TEST);
   glShadeModel(GL_SMOOTH);

   glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
   glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse);
   glLightfv(GL_LIGHT0, GL_POSITION, position);
   glLightModelfv(GL_LIGHT_MODEL_AMBIENT, lmodel_ambient);
   glLightModelfv(GL_LIGHT_MODEL_LOCAL_VIEWER, local_view);

   glEnable(GL_LIGHTING);
   glEnable(GL_LIGHT0);
}


void display(void) {

   GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
   GLfloat mat_ambient[] = { 0.7, 0.7, 0.7, 1.0 };
   GLfloat mat_ambient_color[] = { 0.1, 0.8, 0.2, 1.0 };
   GLfloat mat_diffuse[] = { 0.1, 0.9, 0.8, 1.0 };
   GLfloat mat_diffuse_2[] = { 0.1, 0.2, 0.8, 1.0 };
   GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat no_shininess[] = { 0.0 };
   GLfloat low_shininess[] = { 5.0 };
   GLfloat high_shininess[] = { 100.0 };
   GLfloat mat_emission[] = {0.3, 0.2, 0.2, 0.0};

   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

   // Только диффузное отражение
   glPushMatrix();
   glTranslatef (-3.75, 3.0, 0.0);
   glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
   glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse_2);
   glMaterialfv(GL_FRONT, GL_SPECULAR, no_mat);
   glMaterialfv(GL_FRONT, GL_SHININESS, no_shininess);
   glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
   glutSolidSphere(1.0, 16, 16);
   glPopMatrix();

   // Доминирующее зеркальное отражение
   glPushMatrix();
   glTranslatef (-1.25, 3.0, 0.0);
   glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
```

```
   glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
   glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
   glMaterialfv(GL_FRONT, GL_SHININESS, low_shininess);
   glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
   glutSolidSphere(1.0, 16, 16);
   glPopMatrix();

   // Диффузное и зеркальное отражение
   glPushMatrix();
   glTranslatef (1.25, 3.0, 0.0);
   glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
   glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
   glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
   glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
   glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
   glutSolidSphere(1.0, 16, 16);
   glPopMatrix();

   glFlush();

}


void reshape(int w, int h) {
   glViewport(0, 0, w, h);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   glOrtho (-6.0, 6.0, -3.0*((GLfloat)h*2)/(GLfloat)w, 3.0*((GLfloat)h*2)/(GLfloat)w, -
10.0, 10.0);
   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();
}


int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
   glutInitWindowSize (600, 600);
   glutCreateWindow(argv[0]);
   init();
   glutReshapeFunc(reshape);
   glutDisplayFunc(display);
   glutMainLoop();
   return 0;
}
```

Результат работы программы



Вывод

Во время выполнения лабораторной работы были получены навыки использования источника освещения, использования различных типов поверхности объемных фигур с помощью OpenGL при написании программ на C++.

Список использованной литературы

1. Эйнджел Э. Интерактивная компьютерная графика. Вводный курс на базе
2. OpenGL - М.: Вильямс, 2001. - 592 с.

3. Хирн, Бейкер. Компьютерная графика и стандарт Open GL – М.: Вильямс, 2005, 1168 с.
4. Роджерс Д. Алгоритмические основы машинной графики- М.: Мир, 1989. – 512 с.
5. Земсков Ю.В. Qt 4 на примерах. - СПб.: БХВ - Петербург, 2008. - 608 с. : рис. + эл. опт. диск (CD-ROM)