ФГБОУ ВО «Чувашский государственный университет»

им. И.Н. Ульянова.

Кафедра Вычислительной техники

Дисциплина «Алгоритмические основы компьютерной графики»

Лабораторная работа № 2

**Размещение объектов на сцене.**

Выполнила студентка

Группы ИВТ-42-19

Волкова Софья

Проверил:

Степанов В.В.

Чебоксары 2021

Задание к лабораторной работе.

1. В отдельном модуле создать класс GraphicObject вышеописанной структуры для работы с трехмерным объектом.

2. Реализовать вывод четырех чайников разных цветов расположенных на осях OX и OZ с носиками, повернутыми в центр сцены, как изображено выше. Для хранения и вывода трехмерных объектов использовать ранее созданный класс GraphicObject.

3. Вывести все объекты, реализуя классический цикл программы с

анимацией:

1. Очистить буфер экрана (буферы цвета, глубины и т.д.)

2. Установить матрицу камеры

3. Для каждого объекта (в методе draw):

a. сохранит матрицу наблюдения в стеке

b. умножить матрицу наблюдения на матрицу модели

c. Вывести модель (установив так же её цвет)

d. Восстановить матрицу наблюдения из стека

4. Поменять местами передний и задний буферы

5. Перейти к шагу 1

Текст заголовочного файла модуля с классом GraphicObject :

#ifndef GRAPHICOBJECT\_H

#define GRAPHICOBJECT\_H

#include <stdio.h>

#include <windows.h>

#include <GL/gl.h>

#include <GL/glu.h>

#include "GL/freeglut.h"

#include "math.h"

#include "GLM/GLM/glm.hpp"

#include "GLM/GLM/gtx/transform.hpp"

#include "GLM/GLM/gtc/type\_ptr.hpp"

//using namespace glm;

class GraphicObject

{

private:

// Позиция и угол поворота для объекта

//GLfloat position[3];

GLfloat angle;

// Матрица модели (расположение объекта) ‐ чтоб не вычислять каждый раз

//GLfloat modelMatrix[16];

// Цвет модели

//GLfloat color[3];

glm::mat4 modelMatrix;

glm::vec3 position;

glm::vec3 color;

public:

// Конструктор

GraphicObject(void);

// Задать позицию объекта

void setPosition(float x, float y, float z);

// Задать угол поворота в градусах относительно оси OY

void setAngle(float a);

// Задать цвет модели

void setСolor(float r, float g, float b);

// Вывести объект

void draw(void);

};

#endif

Текст файла реализации модуля с классом GraphicObject:

#include "GraphicObject.h"

#include "GLM/GLM/glm.hpp"

#include "GLM/GLM/gtx/transform.hpp"

#include "GLM/GLM/gtc/type\_ptr.hpp"

//using namespace glm;

// Конструктор

GraphicObject::GraphicObject(void)

{

}

// задать позицию объекта

void GraphicObject::setPosition(float x, float y, float z)

{

position[0] = x;

position[1] = y;

position[2] = z;

}

// задать угол поворота в градусах относительно оси OY

void GraphicObject::setAngle(float a)

{

angle = a;

}

// Задать цвет модели

void GraphicObject::setСolor(float r, float g, float b)

{

color[0] = r;

color[1] = g;

color[2] = b;

}

// вывести объект

void GraphicObject::draw(void)

{

glPushMatrix();

//glColor3f(color[0], color[1], color[2]);

//glTranslatef(position[0], position[1], position[2]);

//glRotatef(angle, 0.0, 1.0, 0.0);

glColor3fv(glm::value\_ptr(color));

this->modelMatrix = glm::translate(position) \* glm::rotate(glm::radians(angle), glm::vec3(0.0, 1.0, 0.0));

glMultMatrixf(glm::value\_ptr(this->modelMatrix));

glutWireTeapot(1.0);

glPopMatrix();

}

Текст основной программы с комментариями:

#include <stdio.h>

#include <windows.h>

#include <GL/gl.h>

#include <GL/glu.h>

#include "GL/freeglut.h"

#include "GraphicObject.h"

double Colors[5][3] = { {1.0,0.0,0.0},{1.0,1.0,1.0},{0.0,0.0,1.0},{1.0,1.0,0.5},{0.0,1.0,1.0} };

// МАССИВ ОБЪЕКТОВ ДЛЯ ВЫВОДА

const int graphicObjectCount = 4;

GraphicObject graphicObjects[graphicObjectCount];

//int i = 0;

// функция вызывается каждые 20 мс

void Simulation(int value)

{

// устанавливаем признак того, что окно нуждается в перерисовке

glutPostRedisplay();

// эта же функция будет вызвана еще раз через 20 мс

glutTimerFunc(20, Simulation, 0);

}

// функция, вызываемая при изменении размеров окна

void Reshape(int w, int h)

{

// установить новую область просмотра, равную всей области окна

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

// установить матрицу проекции с правильным аспектом

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(25.0, (float)w / h, 0.2, 70.0);

};

// функция вызывается при перерисовке окна

// в том числе и принудительно, по командам glutPostRedisplay

void Display(void)

{

// отчищаем буфер цвета

glClearColor(1.0, 1.0, 0.0, 1.0);

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

// включаем тест глубины

glEnable(GL\_DEPTH\_TEST);

// устанавливаем камеру

//glMatrixMode(GL\_MODELVIEW);

//glLoadIdentity();

//gluLookAt(15, 10, 20, 0, 0, 0, 0, 1, 0);

// устанавливаем камеру

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

gluLookAt(15, 15, 7.5, 0, 0, 0, 0, 1, 0);

// выводим объекты

for (int i = 0; i < graphicObjectCount; i++) {

graphicObjects[i].draw();

};

// смена переднего и заднего буферов

glutSwapBuffers();

};

// Функция обработки нажатия клавиш

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

{

printf("Key code is %i\n", (unsigned int)key);

};

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

// инициализация библиотеки GLUT

glutInit(&argc, argv);

// инициализация дисплея (формат вывода)

glutInitDisplayMode(GLUT\_RGBA | GLUT\_DOUBLE | GLUT\_DEPTH | GLUT\_MULTISAMPLE);

// создание окна:

// 1. устанавливаем верхний левый угол окна

glutInitWindowPosition(200, 200);

// 2. устанавливаем размер окна

glutInitWindowSize(600, 600);

// 3. создаем окно

glutCreateWindow("laba2");

// устанавливаем функцию, которая будет вызываться для перерисовки окна

glutDisplayFunc(Display);

// устанавливаем функцию, которая будет вызываться при изменении размеров окна

glutReshapeFunc(Reshape);

// устанавливаем функцию которая будет вызвана через 20 мс

glutTimerFunc(20, Simulation, 0);

// устанавливаем функцию, которая будет вызываться при нажатии на клавишу

glutKeyboardFunc(KeyboardFunc);

graphicObjects[0].setPosition(4.0, 0.0, 0.0);

graphicObjects[0].setСolor(1.0, 0.0, 0.0);

graphicObjects[0].setAngle(180);

graphicObjects[1].setPosition(-4.0, 0.0, 0.0);

graphicObjects[1].setСolor(1.0, 1.0, 1.0);

graphicObjects[1].setAngle(0);

graphicObjects[2].setPosition(0.0, 0.0, 4.0);

graphicObjects[2].setСolor(0.0, 1.0, 0.0);

graphicObjects[2].setAngle(90);

graphicObjects[3].setPosition(0.0, 0.0, -4.0);

graphicObjects[3].setAngle(-90);

graphicObjects[3].setСolor(0.0, 0.0, 1.0);

// основной цикл обработки сообщений ОС

glutMainLoop();

}

Скриншот работы программы

