Новосибирский государственный технический университет

Кафедра вычислительной техники



**Пояснительная записка к**

**Расчётно-графической работе.**

**Вариант 2**

**«Программирование графики»**

**Факультет: АВТ Преподаватель:О.М.Гейнц**

**Группа: АВТ-615**

**Студент: Кузенков В.С.**

Новосибирск

2016

# Постановка задачи

По экрану движется вращающийся куб, изображаемый в виде граней. Вариант: управление вращением с клавиатуры.

# Основная идея

Требуется реализовать движение куба, а также его вращение с клавиатуры. Для этого необходимо решить следующие проблемы:

* Изобразить куб в виде граней
* Вращать с помощью клавиш стрелочек на клавиатуре
* Двигать куб

# Основные методы и решения

Основой для прорисовки куба, является графический приметив библиотеки glut проволочный куб(glutWireCube), с размером 0.3, установив размер видимой сцены единицей.

Для поворота используется функция glRotatef, при каждом нажатии на клавиши стрелок, куб поворачивается в пространстве на 5 градусов. Если вправо\влево относительно x, вверх\низ относительно y.

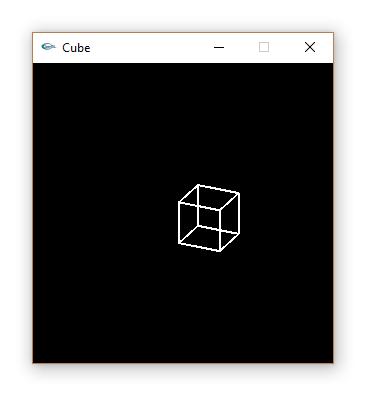
Движение куба реализовано при помощи параллельного переноса (функция glTranslatef). Так же используется функция-таймер(glutTimerFunc), в которой одним из параметров вызываем нашу функцию, задающую направление и само движение. Для этого используем флаг rotation, проверяя, если координата x середины куба больше 0.7, то двигаем куб в противоположную сторону. Для более плавного движения установил изменение пути(dx) на 0.01, а задержку функции-таймера на 75 миллисекунд. Тогда скорость равна 0.01 единица в 75мс.

# Ограничения

Размер примитивов в glut основывается на размере окна, поэтому, используя стандартные методы winAPI я убрал возможность менять размер окна.

# Тестирование

Программа работает корректно.



# Текст программы с комментариями

#include <GL/glut.h>

#include <Windows.h>

#include <iostream>

/\* По экрану движется вращающийся куб, изображаемый в виде граней. Вариант: управление вращением с клавиатуры.\*/

// ----------------------------------------------------------

// Прототипы функций

// ----------------------------------------------------------

void display();

void specialKeys();

// ----------------------------------------------------------

// Глобальные переменные

// ----------------------------------------------------------

double rotate\_y = 0;

double rotate\_x = 0;

// XZ позиция камеры

float dx = 0.0;

//направление движения куба

int rotation = 0; //0 - право, 1 - лево

// ----------------------------------------------------------

// Инициализация

// ----------------------------------------------------------

void Initialize() {

glClearColor(0.0, 0.0, 0.0, 1.0);

glMatrixMode(GL\_PROJECTION);

glEnable(GL\_LINE\_SMOOTH); //сглаживание линий

glHint(GL\_LINE\_SMOOTH\_HINT, GL\_NICEST);

glLoadIdentity();

}

// ----------------------------------------------------------

// Функция таймер для движения

// ----------------------------------------------------------

void TimerFunction(int value) {

if(int(dx \* 10) / 10.0 == -0.7)

{

rotation = 0;

}

else if (int(dx \* 10) / 10.0 == 0.7)

{

rotation = 1;

}

if (rotation == 1)

{

dx -= 0.01;

}

else if (rotation == 0) {

dx += 0.01;

}

//std::cout << int(dx \* 100) / 100.0 << ' ' << rotation << ' ';

glutPostRedisplay();

glutTimerFunc(75, TimerFunction, 1);

}

//----------------------------------------------------

// display()

// ----------------------------------------------------------

void display() {

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

glLoadIdentity();

gluLookAt(0, 0, 0, 0, 0, -1, 0, 1, 0);

//повороты вдоль ox и oy

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

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

//движение

glTranslatef(dx, 0.0, 0.0);

glutWireCube(0.3);

glFlush();

glutSwapBuffers();

}

// ----------------------------------------------------------

// specialKeys()

// ----------------------------------------------------------

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

if (key == GLUT\_KEY\_RIGHT)

rotate\_y += 5;

else if (key == GLUT\_KEY\_LEFT)

rotate\_y -= 5;

else if (key == GLUT\_KEY\_UP)

rotate\_x -= 5;

else if (key == GLUT\_KEY\_DOWN)

rotate\_x += 5;

glutPostRedisplay();

}

// ----------------------------------------------------------

// main()

// ----------------------------------------------------------

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

glutInit(&argc, argv);

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

// Создание окна

glutCreateWindow("Cube");

glutInitWindowSize(500, 500);

glutInitWindowPosition(200, 740);

//запрещаем изменять размер окна

HWND handle = FindWindow(NULL, "Cube");

SetWindowLongPtr(handle, GWL\_STYLE, (GetWindowLong(handle, GWL\_STYLE) ^ WS\_SIZEBOX) ^ WS\_MAXIMIZEBOX);

glEnable(GL\_DEPTH\_TEST);

// Вызов функций

glutDisplayFunc(display);

glutTimerFunc(10, TimerFunction, 0);

glutSpecialFunc(specialKeys);

Initialize();

glutMainLoop();

return 0;

}