МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Лабораторна робота №2

з дисципліни

Комп’ютерна графіка

Виконав:

студент групи КВ-81

Петренко М. Е.

Київ 2011

Задание: Создать анимированное трехмерное изображение, указанное в варианте, используя описанные в лабораторной работе эффекты прозрачности и смешения цветов.

*Код програми:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

// для работы с библиотекой OpenGL

using Tao.OpenGl;

// для работы с библиотекой FreeGLUT

using Tao.FreeGlut;

// для работы с элементом управления SimpleOpenGLControl

using Tao.Platform.Windows;

namespace WindowsFormsApplication4

{

public partial class Form1 : Form

{

private float rot\_1, rot\_2;

float an1 = 0;

float an2 = 0;

double a;

double col11, col22, col33, col44, col55, col66, op1;

public Form1()

{

InitializeComponent();

AnT.InitializeContexts();

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void Form1\_Load(object sender, EventArgs e)

{

// инициализация Glut

Glut.glutInit();

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

// очитка окна

Gl.glClearColor(255, 255, 255, 1);

// установка порта вывода в соотвествии с размерами элемента anT

Gl.glViewport(0, 0, AnT.Width, AnT.Height);

// настройка проекции

Gl.glMatrixMode(Gl.GL\_PROJECTION);

Gl.glLoadIdentity();

Glu.gluPerspective(45, (float)AnT.Width / (float)AnT.Height, 0.1, 200);

Gl.glMatrixMode(Gl.GL\_MODELVIEW);

Gl.glLoadIdentity();

// настройка параметров OpenGL для визуализации

Gl.glEnable(Gl.GL\_DEPTH\_TEST);

//Gl.glEnable(Gl.GL\_ALPHA\_TEST);

Gl.glEnable(Gl.GL\_BLEND);

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

// активация таймера

RenderTimer.Start();

}

private void RenderTimer\_Tick(object sender, EventArgs e)

{

{

// вызываем функцию, отвечающей за отрисовку сцены

Draw();

}

}

// функция отрисовки сцены

private void Draw()

{

// два параметра, которые мы будем использовать для непрерывного вращения сцены вокруг 2 координатных осей

rot\_1 -= 3 + an1;

rot\_2 -= 5 + an2;

// rot\_3 = (float)Math.Sin(rot\_1) + (float)Math.Cos(rot\_1) + (float)Math.Sin(rot\_1);

// очистка буфера цвета и буфера глубины

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

Gl.glClearColor(255, 255, 255, 1);

// очищение текущей матрицы

Gl.glLoadIdentity();

// установка положения камеры (наблюдателя). Как видно из кода

// дополнительно на полложение наблюдателя по оси Z влияет значение

// установленное в ползунке, доступном для пользователя.

// таким образом, при перемещении ползунка, наблюдатель будет отдалятся или приближатся к объекту наблюдения

Gl.glTranslated(0, 0, -160 + a);

// 2 поворота (углы rot\_1 и rot\_2)

Gl.glRotated(rot\_1, 40, 1, 15);

Gl.glRotated(rot\_2, rot\_1 + rot\_2, 5, 20);

//Gl.glRotated(20, -20, 0, 0);

//Буква П

//лицевая сторона

Gl.glColor4d(1.0, 0.3, 0.7, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-35.0, 25.0, 0.0);

Gl.glVertex3d(-30.0, 25.0, 0.0);

Gl.glVertex3d(-30.0, -20.0, 0.0);

Gl.glVertex3d(-35.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-30.0, 25.0, 0.0);

Gl.glVertex3d(-15.0, 25.0, 0.0);

Gl.glVertex3d(-15.0, 20.0, 0.0);

Gl.glVertex3d(-30.0, 20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-15.0, 25.0, 0.0);

Gl.glVertex3d(-10.0, 25.0, 0.0);

Gl.glVertex3d(-10.0, -20.0, 0.0);

Gl.glVertex3d(-15.0, -20.0, 0.0);

Gl.glEnd();

//задняя сторона буквы П

Gl.glColor4d(0.0, 0.3, 0.7, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-35.0, 25.0, -5.0);

Gl.glVertex3d(-30.0, 25.0, -5.0);

Gl.glVertex3d(-30.0, -20.0, -5.0);

Gl.glVertex3d(-35.0, -20.0, -5.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-30.0, 25.0, -5.0);

Gl.glVertex3d(-15.0, 25.0, -5.0);

Gl.glVertex3d(-15.0, 20.0, -5.0);

Gl.glVertex3d(-30.0, 20.0, -5.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-15.0, 25.0, -5.0);

Gl.glVertex3d(-10.0, 25.0, -5.0);

Gl.glVertex3d(-10.0, -20.0, -5.0);

Gl.glVertex3d(-15.0, -20.0, -5.0);

Gl.glEnd();

//обводка буквы П

Gl.glColor4d(0.8, 0.0, 0.7, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-35.0, 25.0, 0.0);

Gl.glVertex3d(-35.0, 25.0, -5.0);

Gl.glVertex3d(-35.0, -20.0, -5.0);

Gl.glVertex3d(-35.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-35.0, 25.0, 0.0);

Gl.glVertex3d(-35.0, 25.0, -5.0);

Gl.glVertex3d(-10.0, 25.0, -5.0);

Gl.glVertex3d(-10.0, 25.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-10.0, 25.0, 0.0);

Gl.glVertex3d(-10.0, 25.0, -5.0);

Gl.glVertex3d(-10.0, -20.0, -5.0);

Gl.glVertex3d(-10.0, -20.0, 0.0);

Gl.glEnd();

Gl.glColor4d(0.8, 0.3, 0.3, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-15.0, -20.0, 0.0);

Gl.glVertex3d(-15.0, -20.0, -5.0);

Gl.glVertex3d(-10.0, -20.0, -5.0);

Gl.glVertex3d(-10.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-15.0, 20.0, 0.0);

Gl.glVertex3d(-15.0, 20.0, -5.0);

Gl.glVertex3d(-15.0, -20.0, -5.0);

Gl.glVertex3d(-15.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-30.0, 20.0, 0.0);

Gl.glVertex3d(-30.0, 20.0, -5.0);

Gl.glVertex3d(-15.0, 20.0, -5.0);

Gl.glVertex3d(-15.0, 20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-30.0, 20.0, 0.0);

Gl.glVertex3d(-30.0, 20.0, -5.0);

Gl.glVertex3d(-30.0, -20.0, -5.0);

Gl.glVertex3d(-30.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(-35.0, -20.0, 0.0);

Gl.glVertex3d(-35.0, -20.0, -5.0);

Gl.glVertex3d(-30.0, -20.0, -5.0);

Gl.glVertex3d(-30.0, -20.0, 0.0);

Gl.glEnd();

Gl.glRotated(rot\_1, 10, 6, 5);

Gl.glRotated(rot\_2, 30, 10, 30);

//Буква М

//лицевая сторона

Gl.glColor4d(0.3, 0.6, 0.3, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(5.0, 25.0, 0.0);

Gl.glVertex3d(10.0, 25.0, 0.0);

Gl.glVertex3d(10.0, -20.0, 0.0);

Gl.glVertex3d(5.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(30.0, 25.0, 0.0);

Gl.glVertex3d(35.0, 25.0, 0.0);

Gl.glVertex3d(35.0, -20.0, 0.0);

Gl.glVertex3d(30.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(10.0, 25.0, 0.0);

Gl.glVertex3d(20.0, 10.0, 0.0);

Gl.glVertex3d(20.0, 3.0, 0.0);

Gl.glVertex3d(10.0, 17.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(20.0, 10.0, 0.0);

Gl.glVertex3d(30.0, 25.0, 0.0);

Gl.glVertex3d(30.0, 17.0, 0.0);

Gl.glVertex3d(20.0, 3.0, 0.0);

Gl.glEnd();

//Тыл буквы М

Gl.glColor4d(0.5, 0.5, 1.0, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(5.0, 25.0, -5.0);

Gl.glVertex3d(10.0, 25.0, -5.0);

Gl.glVertex3d(10.0, -20.0, -5.0);

Gl.glVertex3d(5.0, -20.0, -5.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(30.0, 25.0, -5.0);

Gl.glVertex3d(35.0, 25.0, -5.0);

Gl.glVertex3d(35.0, -20.0, -5.0);

Gl.glVertex3d(30.0, -20.0, -5.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(10.0, 25.0, -5.0);

Gl.glVertex3d(20.0, 10.0, -5.0);

Gl.glVertex3d(20.0, 3.0, -5.0);

Gl.glVertex3d(10.0, 17.0, -5.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(20.0, 10.0, -5.0);

Gl.glVertex3d(30.0, 25.0, -5.0);

Gl.glVertex3d(30.0, 17.0, -5.0);

Gl.glVertex3d(20.0, 3.0, -5.0);

Gl.glEnd();

//обводка буквы М

Gl.glColor4d(0.1, 1.0, 0.5, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(5.0, 25.0, 0.0);

Gl.glVertex3d(5.0, 25.0, -5.0);

Gl.glVertex3d(5.0, -20.0, -5.0);

Gl.glVertex3d(5.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(5.0, 25.0, 0.0);

Gl.glVertex3d(5.0, 25.0, -5.0);

Gl.glVertex3d(10.0, 25.0, -5.0);

Gl.glVertex3d(10.0, 25.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(10.0, 25.0, 0.0);

Gl.glVertex3d(10.0, 25.0, -5.0);

Gl.glVertex3d(20.0, 10.0, -5.0);

Gl.glVertex3d(20.0, 10.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(20.0, 10.0, 0.0);

Gl.glVertex3d(20.0, 10.0, -5.0);

Gl.glVertex3d(30.0, 25.0, -5.0);

Gl.glVertex3d(30.0, 25.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(30.0, 25.0, 0.0);

Gl.glVertex3d(30.0, 25.0, -5.0);

Gl.glVertex3d(35.0, 25.0, -5.0);

Gl.glVertex3d(35.0, 25.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(35.0, 25.0, 0.0);

Gl.glVertex3d(35.0, 25.0, -5.0);

Gl.glVertex3d(35.0, -20.0, -5.0);

Gl.glVertex3d(35.0, -20.0, 0.0);

Gl.glEnd();

Gl.glColor4d(0.6, 0.9, 0.0, 1 + op1);

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(30.0, -20.0, 0.0);

Gl.glVertex3d(30.0, -20.0, -5.0);

Gl.glVertex3d(35.0, -20.0, -5.0);

Gl.glVertex3d(35.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(30.0, 17.0, 0.0);

Gl.glVertex3d(30.0, 17.0, -5.0);

Gl.glVertex3d(30.0, -20.0, -5.0);

Gl.glVertex3d(30.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(30.0, 17.0, 0.0);

Gl.glVertex3d(30.0, 17.0, -5.0);

Gl.glVertex3d(20.0, 3.0, -5.0);

Gl.glVertex3d(20.0, 3.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(20.0, 3.0, 0.0);

Gl.glVertex3d(20.0, 3.0, -5.0);

Gl.glVertex3d(10.0, 17.0, -5.0);

Gl.glVertex3d(10.0, 17.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(10.0, 17.0, 0.0);

Gl.glVertex3d(10.0, 17.0, -5.0);

Gl.glVertex3d(10.0, -20.0, -5.0);

Gl.glVertex3d(10.0, -20.0, 0.0);

Gl.glEnd();

Gl.glBegin(Gl.GL\_QUADS);

Gl.glVertex3d(5.0, -20.0, 0.0);

Gl.glVertex3d(5.0, -20.0, -5.0);

Gl.glVertex3d(10.0, -20.0, -5.0);

Gl.glVertex3d(10.0, -20.0, 0.0);

Gl.glEnd();

// возвращаем сохраненную матрицу

Gl.glPopMatrix();

// завершаем рисование

Gl.glFlush();

// обновляем элемент AnT

AnT.Invalidate();

}

private void zoom\_Scroll(object sender, EventArgs e)

{

a = (double)zoom.Value;

label1.Text = a.ToString();

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

an1 = (float)trackBar1.Value;

angle1.Text = an1.ToString();

}

private void trackBar2\_Scroll(object sender, EventArgs e)

{

an2 = (float)trackBar2.Value;

angle2.Text = an2.ToString();

}

private void Op\_Scroll(object sender, EventArgs e)

{

op1 = (double)Op.Value / 100 - 1;

label3.Text = Op.Value.ToString();

}

}

}