Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение науки и высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

**Лабораторная работа № 12**

**по дисциплине**

**«Программирование компьютерной графики»**

**Тема: «Преобразование объектов с использованием GLUT»**

Выполнил:

ст. гр. ПРИ-117

Хлызова В.Г.

Принял:

Жигалов И.Е.

Владимир, 2021

ЦЕЛЬ РАБОТЫ

Изучение способов реализации функций рисования и преобразования объектов с использованием библиотеки GLUT.

ЗАДАНИЕ

1. Ознакомиться по методическим указаниям и литературе с теоретическим материалом.

2. Выполнить действия, приведенные в разделе 12.2. При разработке программы имя проекта, создаваемого в MS Visual Studio, должно содержать фамилию студента и группу (например, Ivanov\_Ivan\_ISG\_105\_lab\_1).

3. При выполнении 12.2 изменить программу таким образом, чтобы одновременно отрисовывалось N выбранных объектов в режиме (по умолчанию), указанном для вашего варианта.

ХОД РАБОТЫ

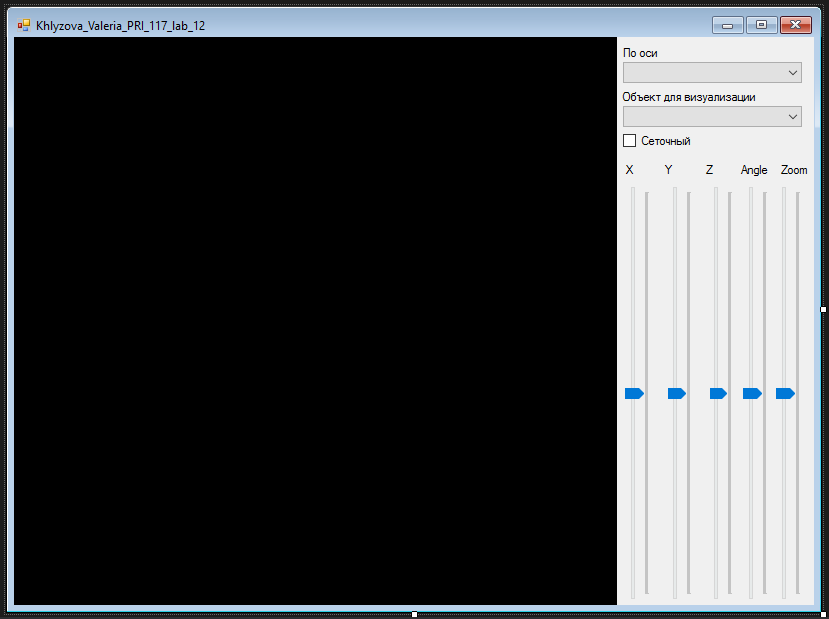


Рисунок 1 - Конструктор формы.

using System;

using System.Windows.Forms;

using Tao.FreeGlut;

using Tao.OpenGl;

namespace Khlyzova\_Valeria\_PRI\_117\_lab\_12

{

public partial class Form1 : Form

{

double a = 0, b = 0, c = -5, d = 0, zoom = 1;

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

{

Draw();

}

private void trackBar5\_Scroll(object sender, EventArgs e)

{

a = (double)trackBar5.Value / 1000.0;

label3.Text = a.ToString();

}

private void trackBar4\_Scroll(object sender, EventArgs e)

{

b = (double)trackBar4.Value / 1000.0;

label4.Text = b.ToString();

}

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

{

c = (double)trackBar1.Value / 1000.0;

label5.Text = c.ToString();

}

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

{

d = (double)trackBar2.Value;

label6.Text = d.ToString();

}

private void trackBar3\_Scroll(object sender, EventArgs e)

{

zoom = (double)trackBar3.Value / 1000.0;

label7.Text = zoom.ToString();

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

Wire = checkBox1.Checked;

}

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

{

switch (comboBox1.SelectedIndex)

{

case 0:

{

os\_x = 1;

os\_y = 0;

os\_z = 0;

break;

}

case 1:

{

os\_x = 0;

os\_y = 1;

os\_z = 0;

break;

}

case 2:

{

os\_x = 0;

os\_y = 0;

os\_z = 1;

break;

}

}

}

int os\_x = 1, os\_y = 0, os\_z = 0;

bool Wire = false;

public Form1()

{

InitializeComponent();

AnT.InitializeContexts();

}

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

{

Glut.glutInit();

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

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

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

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();

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

Gl.glEnable(Gl.GL\_LIGHTING);

Gl.glEnable(Gl.GL\_LIGHT0);

comboBox1.SelectedIndex = 0;

comboBox2.SelectedIndex = 0;

RenderTimer.Start();

}

private void Draw()

{

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

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

Gl.glLoadIdentity();

Gl.glPushMatrix();

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

switch (comboBox2.SelectedIndex)

{

case 0:

{

drawCircle(Wire);

break;

}

case 1:

{

drawCylinder(Wire);

break;

}

case 2:

{

drawCube(Wire);

break;

}

case 3:

{

drawCone(Wire);

break;

}

case 4:

{

drawTorus(Wire);

break;

}

}

Gl.glPopMatrix();

Gl.glFlush();

AnT.Invalidate();

}

// методы для отрисовки фигур. Вариант 15 - число объектов 6, вид объектов: Solid и Wire чередуются

private void drawTorus(bool wire)

{

if (wire)

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireTorus(0.2, 0.6, 32, 32);

} else

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireTorus(0.2, 0.6, 32, 32);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidTorus(0.2, 0.6, 32, 32);

}

}

private void drawCone(bool wire)

{

if (wire)

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCone(0.5, 1, 32, 32);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCone(0.5, 1, 32, 32);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCone(0.5, 1, 32, 32);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCone(0.5, 1, 32, 32);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCone(0.5, 1, 32, 32);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCone(0.5, 1, 32, 32);

} else

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCone(0.5, 1, 32, 32);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCone(0.5, 1, 32, 32);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCone(0.5, 1, 32, 32);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCone(0.5, 1, 32, 32);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCone(0.5, 1, 32, 32);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCone(0.5, 1, 32, 32);

}

}

private void drawCube(bool wire)

{

if (wire)

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCube(1);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCube(1);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCube(1);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCube(1);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCube(1);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCube(1);

} else

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCube(1);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCube(1);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCube(1);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCube(1);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCube(1);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCube(1);

}

}

private void drawCylinder(bool wire)

{

if (wire)

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCylinder(0.5, 1, 32, 32);

}

else

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireCylinder(0.5, 1, 32, 32);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidCylinder(0.5, 1, 32, 32);

}

}

private void drawCircle(bool wire)

{

if (wire)

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidSphere(0.5, 16, 16);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireSphere(0.5, 16, 16);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireSphere(0.5, 16, 16);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidSphere(0.5, 16, 16);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidSphere(0.5, 16, 16);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireSphere(0.5, 16, 16);

}

else

{

Gl.glTranslated(a, b, c);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireSphere(0.5, 16, 16);

Gl.glTranslated(a - 2, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidSphere(0.5, 16, 16);

Gl.glTranslated(a + 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidSphere(0.5, 16, 16);

Gl.glTranslated(a, b - 2, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireSphere(0.5, 16, 16);

Gl.glTranslated(a - 4, b, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutWireSphere(0.5, 16, 16);

Gl.glTranslated(a + 2, b + 4, c + 5);

Gl.glRotated(d, os\_x, os\_y, os\_z);

Gl.glScaled(zoom, zoom, zoom);

Glut.glutSolidSphere(0.5, 16, 16);

}

}

}

}

Результат выполнения:

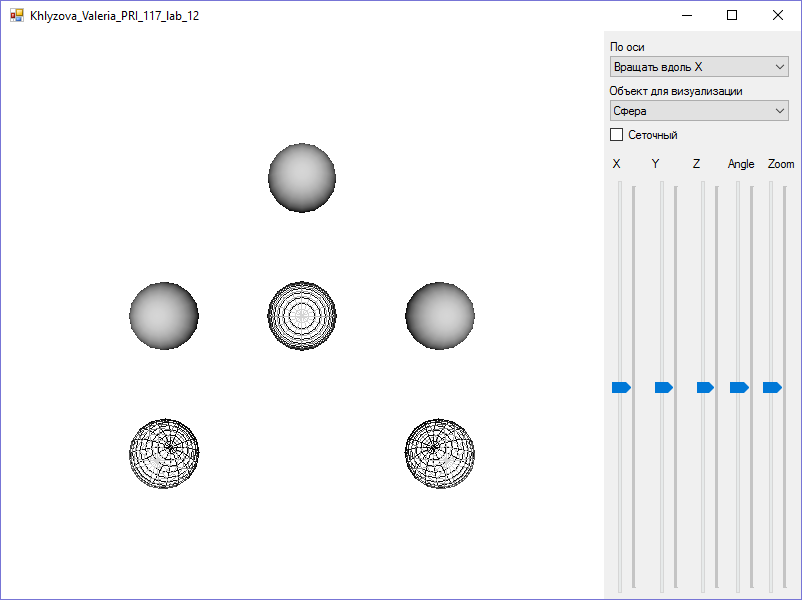


Рисунок 2 - Визуализация сферы.

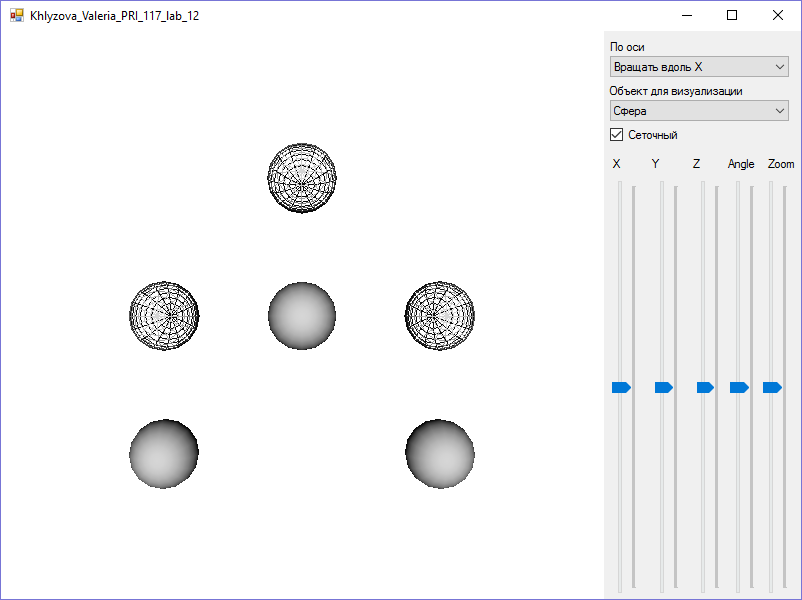


Рисунок 3 -Визуализация сферы.

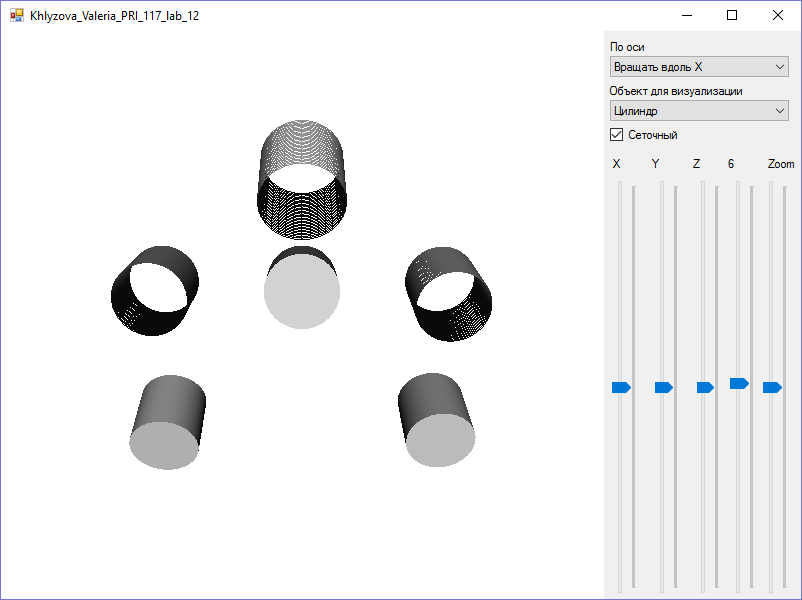


Рисунок 4 - визуализация цилиндра.

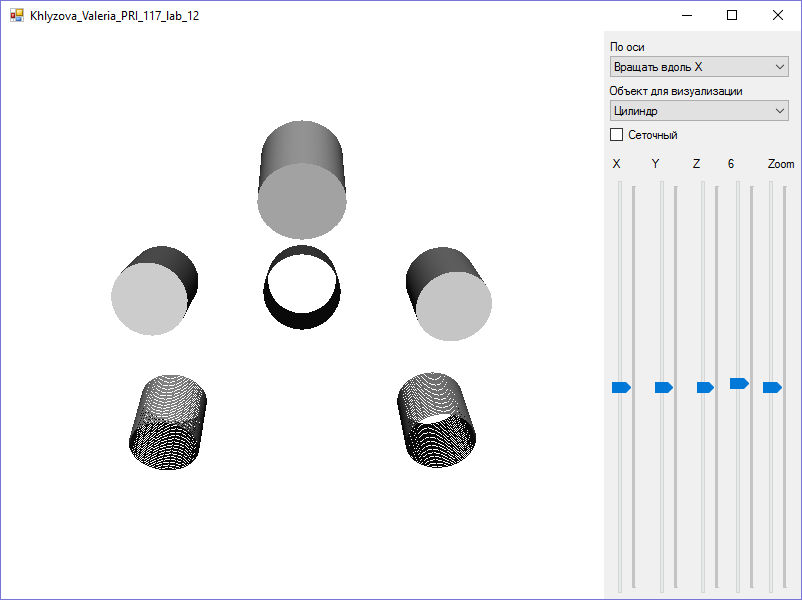


Рисунок 5 - Визуализация цилиндра.

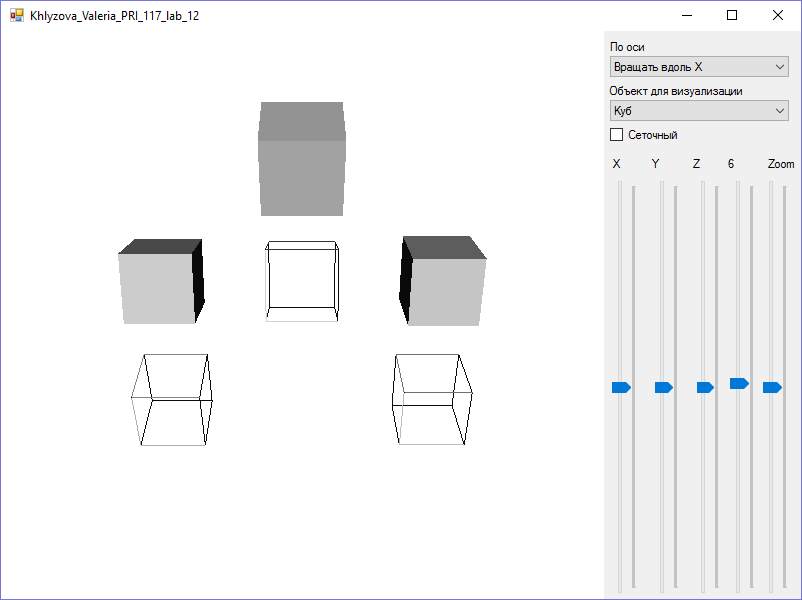


Рисунок 6 - Визуализация куба.

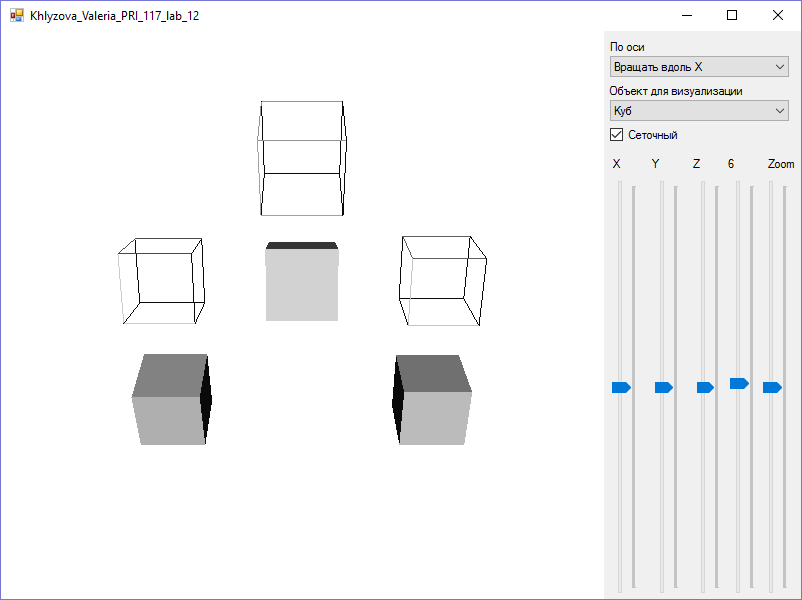


Рисунок 7 - визуализация куба.

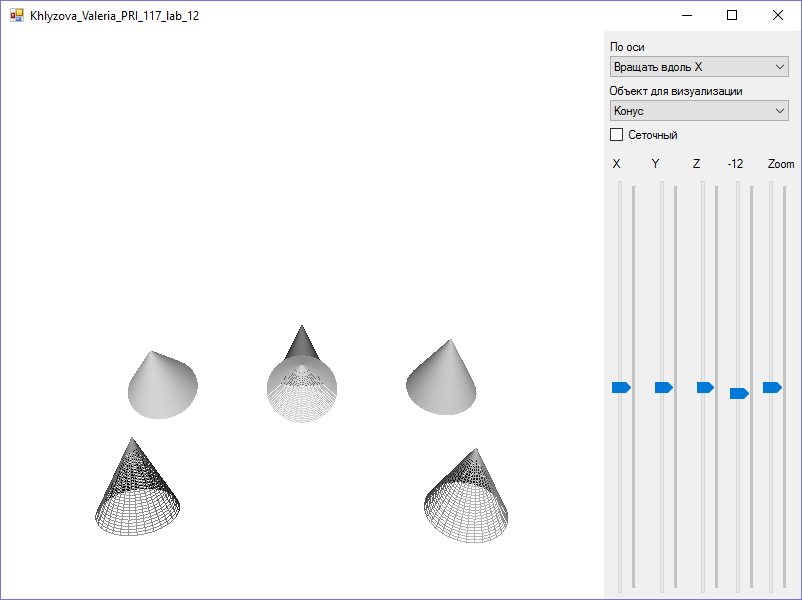


Рисунок 8 - Визуализация конуса.

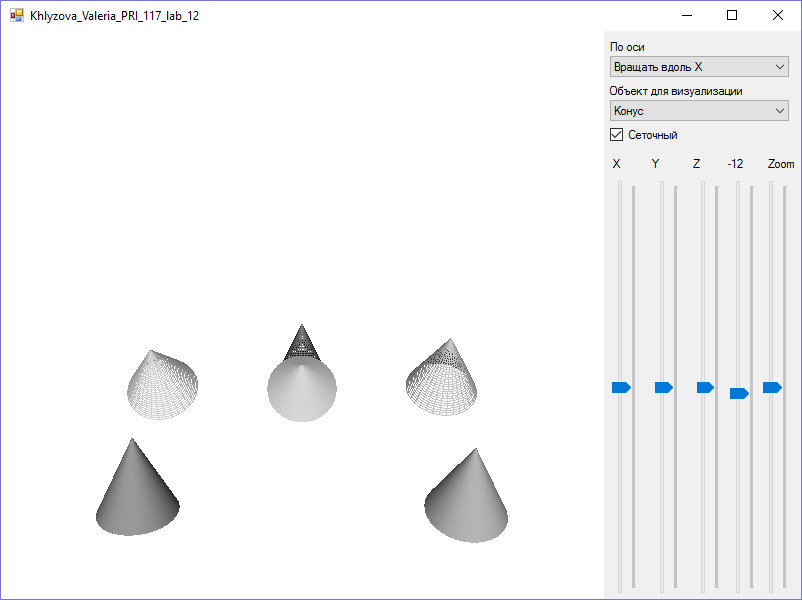


Рисунок 9 - Визуализация конуса.

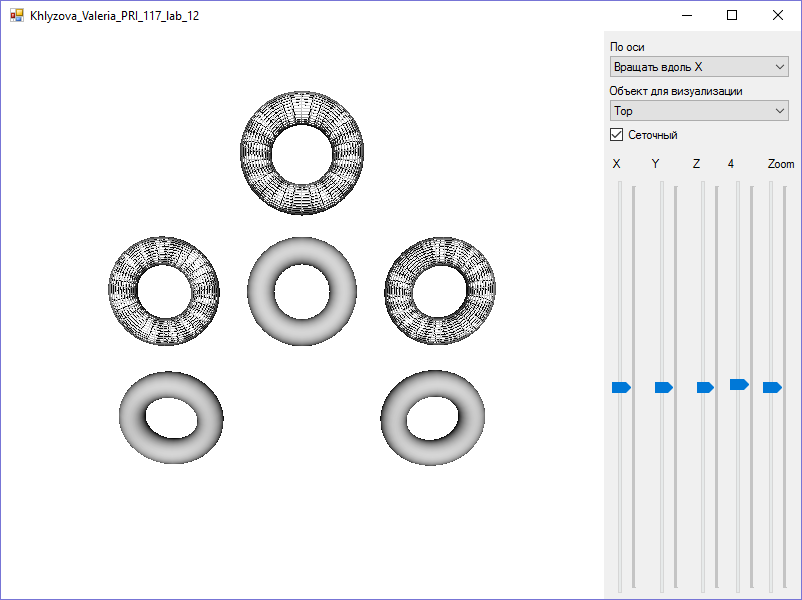


Рисунок 10 - Визуализация тора.

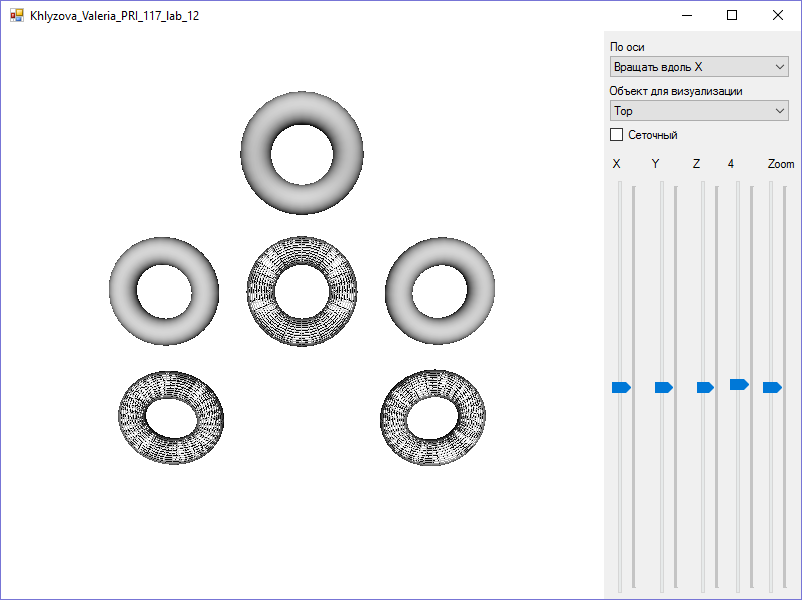


Рисунок 11 - Визуализация тора.

ВЫВОД

В ходе выполнения лабораторной работы были изучены способы реализации функций рисования и преобразования объектов с использованием библиотеки GLUT.