Федеральное агентство по образованию

ФГБОУ ВО Уфимский государственный авиационный технический

университет

Кафедра ВМиК

ЛАБОРАТОРНАЯ РАБОТА №2

По дисциплине: «Инженерная и компьютерная графика»

«Программирование трехмерной графики»

Выполнил: студент группы ПРО-222

Сафронов И.А.

Проверил: Котельников В.А.

Уфа 2021

1. **Цель работы:** Ознакомиться с трехмерной графикой
2. **Задачи:** Открыть файл Glava5. Перевести примеры с VB на на С#, C++, Python, Java на выбор. (трехмерные трансформации)

Выполнить задания:

- Изобразите сферу, вписанную в куб

- Изобразите тор

- Изобразите эллипсоид вращения

- Изобразите шесть сфер разного диаметра, установленных друг на друга

1. **Ход** **работы**

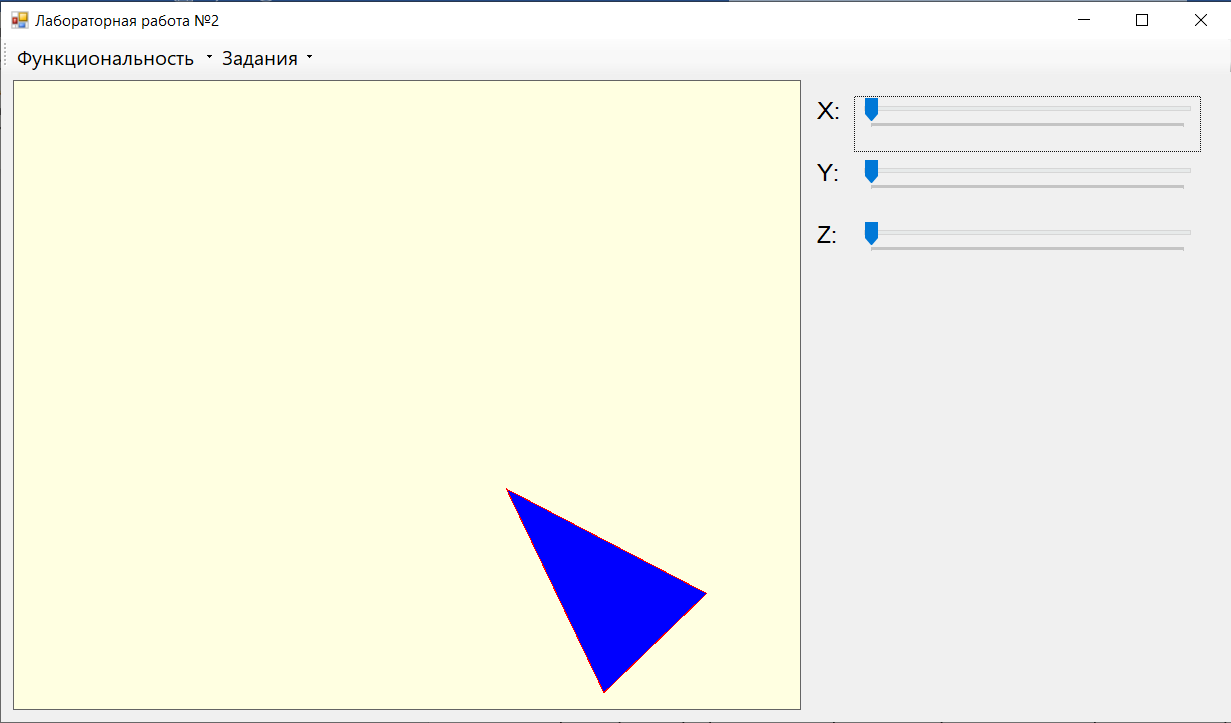


Рис. Вращение объекта относительно трех осей координат

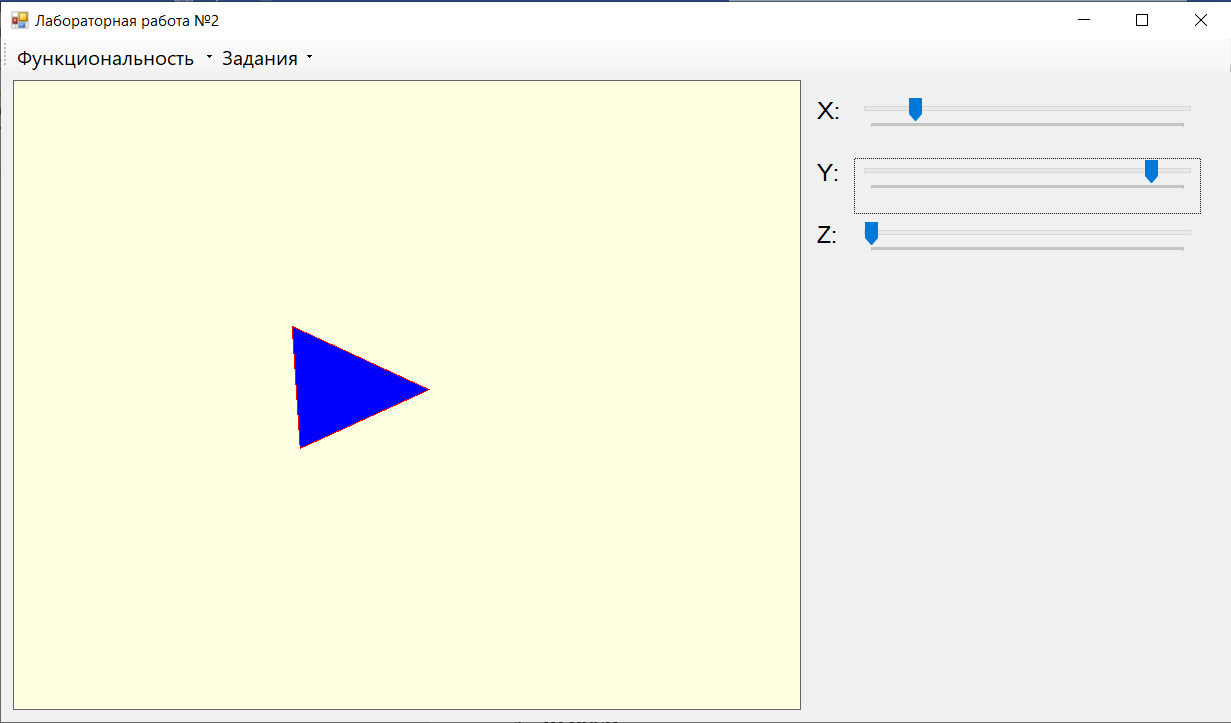


Рис. Вращение объекта относительно трех осей координат

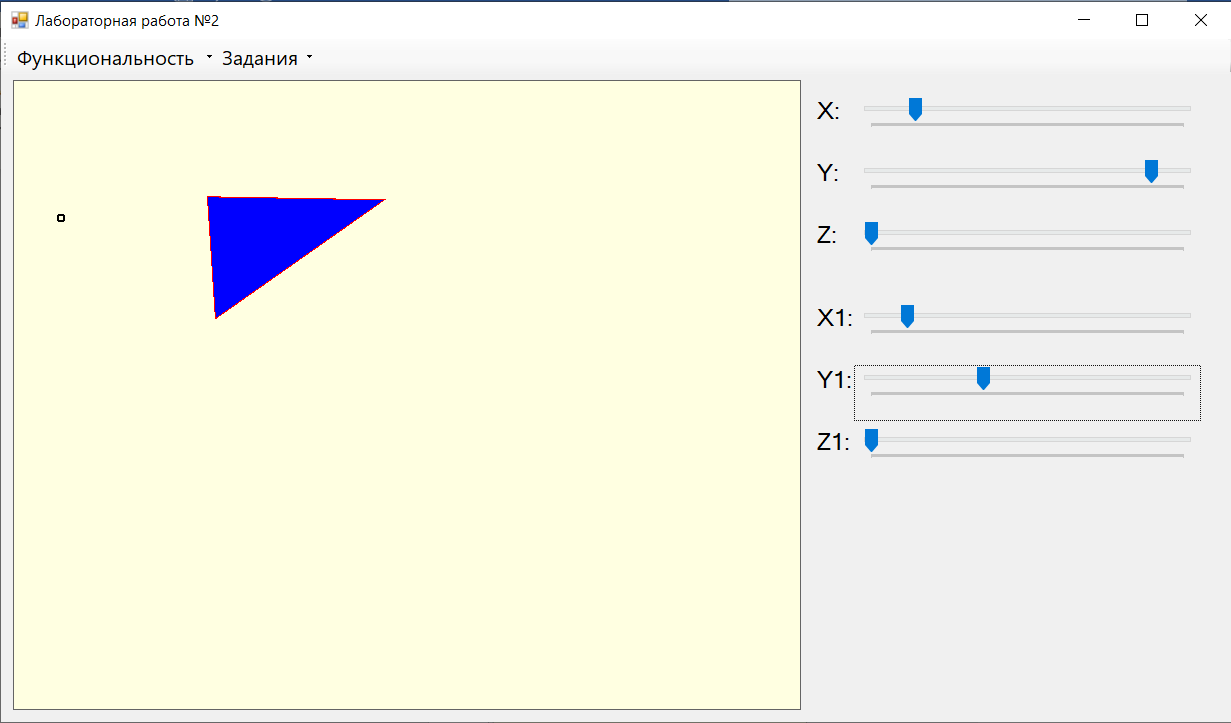


Рис. Поворот вокруг координатных осей относительно произвольной точки

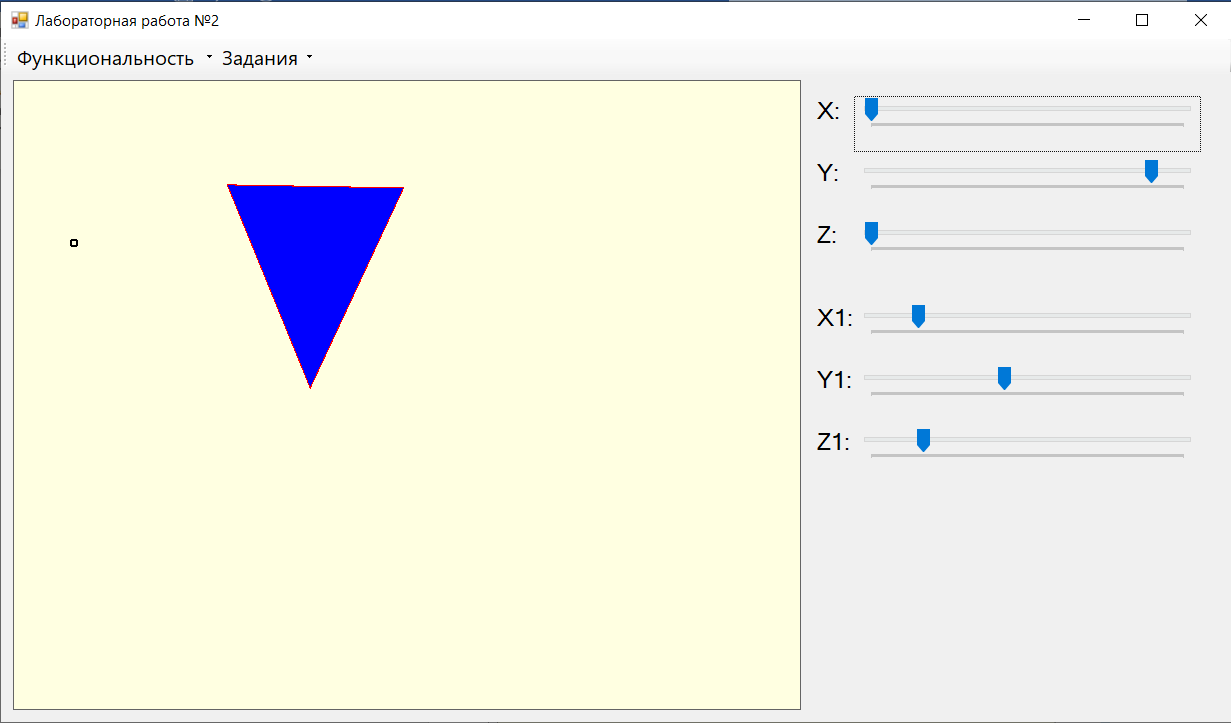


Рис. Поворот вокруг координатных осей относительно произвольной точки

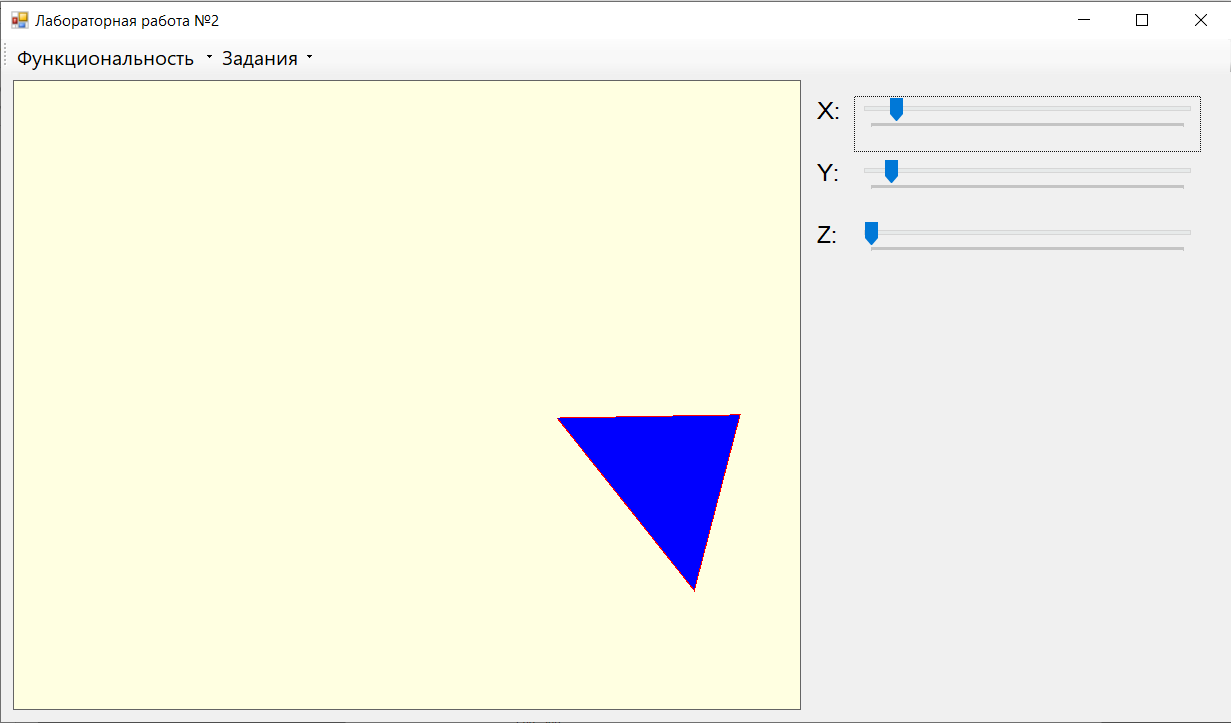


Рис. Поворот вокруг оси, проходящей через начало координат

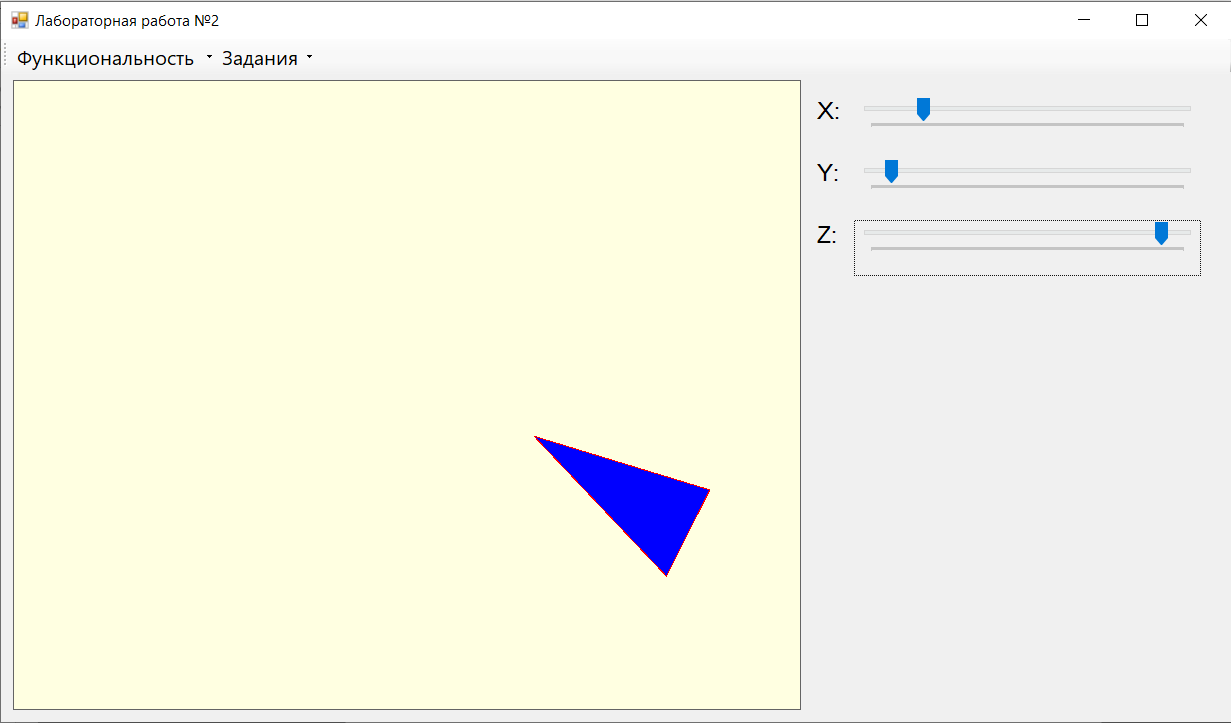


Рис. Поворот вокруг оси, проходящей через начало координат

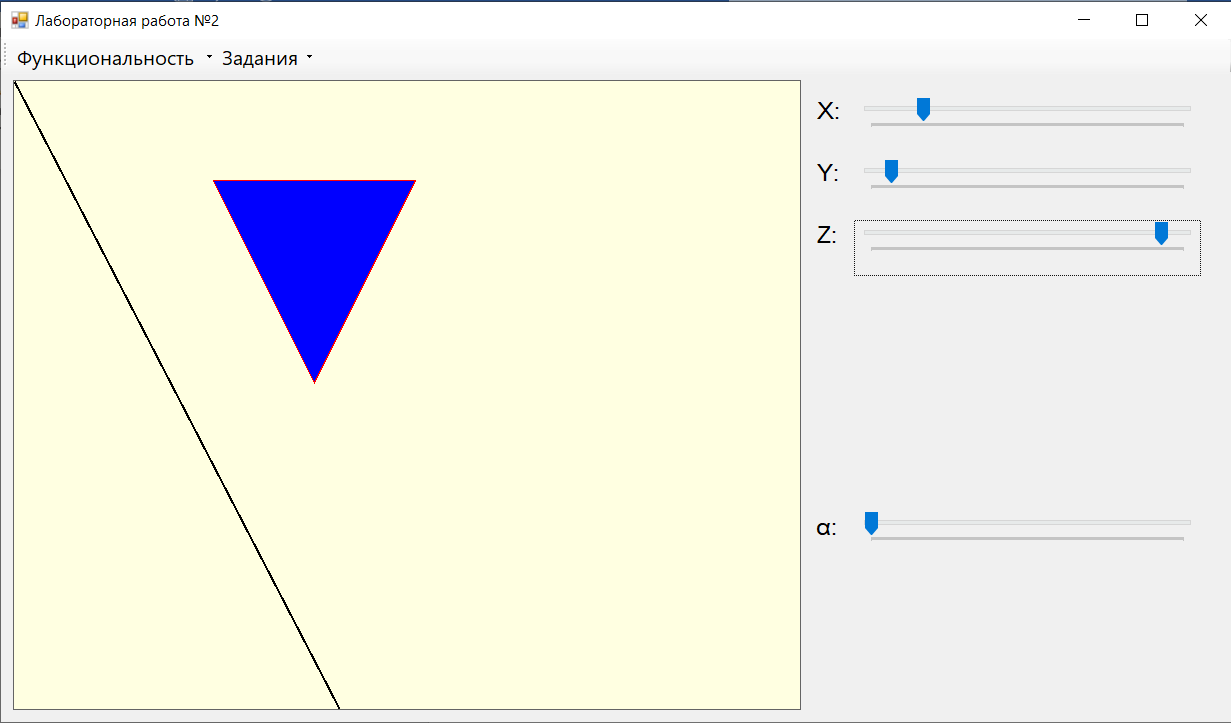


Рис. Поворот вокруг произвольной оси, проходящей через начало координат на угол α

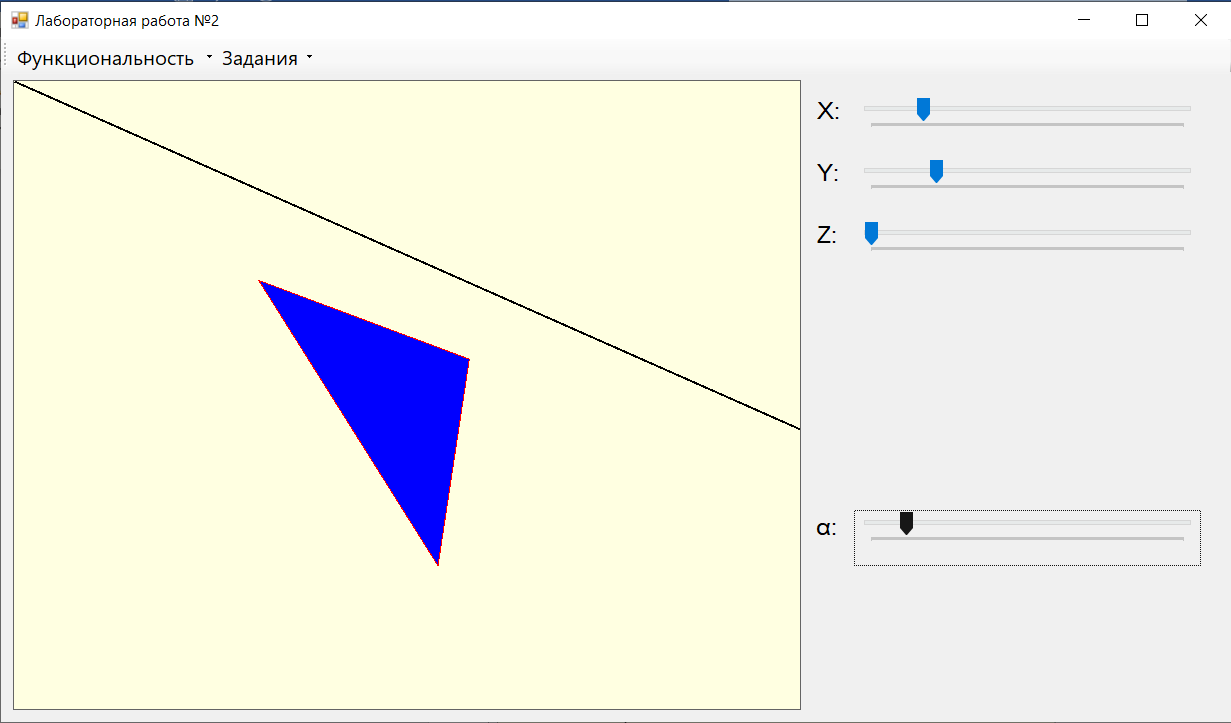


Рис. Поворот вокруг произвольной оси, проходящей через начало координат на угол α

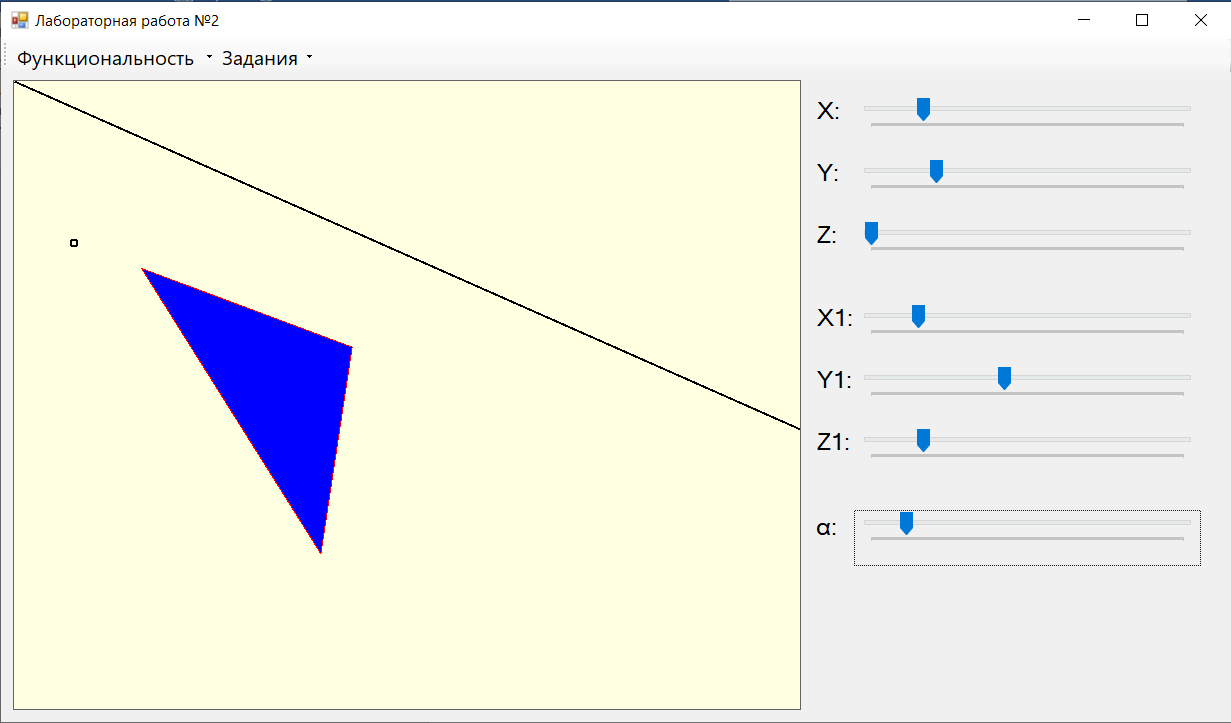


Рис. Поворот точки M (x,y,z) вокруг произвольной оси относительно произвольной точки M0(x0,y0,z0) на угол α

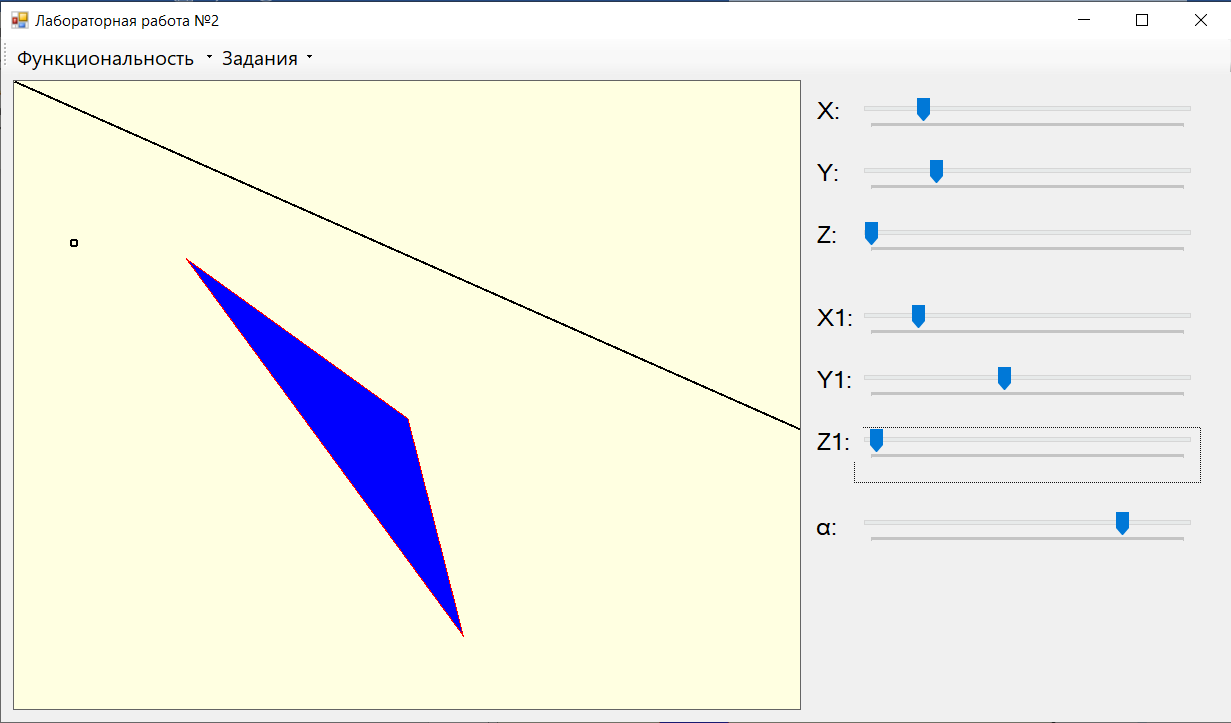


Рис. Поворот точки M (x,y,z) вокруг произвольной оси относительно произвольной точки M0(x0,y0,z0) на угол α

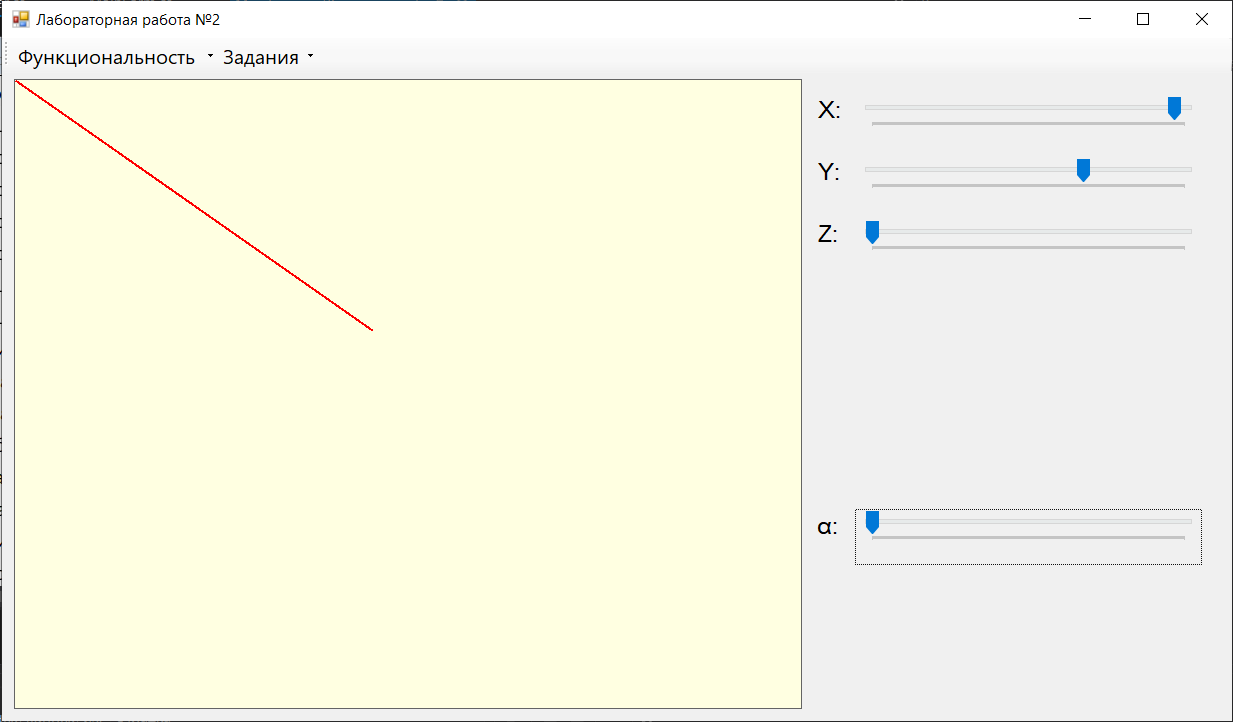


Рис. Вращение вокруг вектора, проходящего через начало координат на угол α

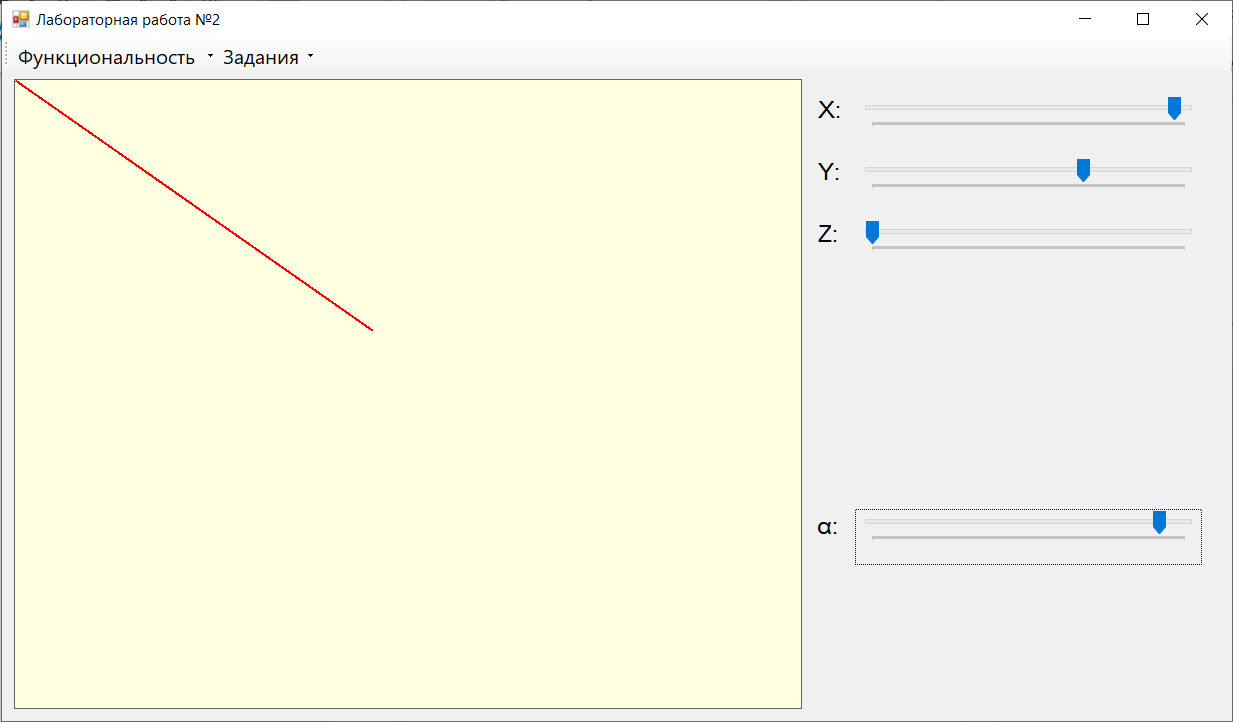


Рис. Вращение вокруг вектора, проходящего через начало координат на угол α

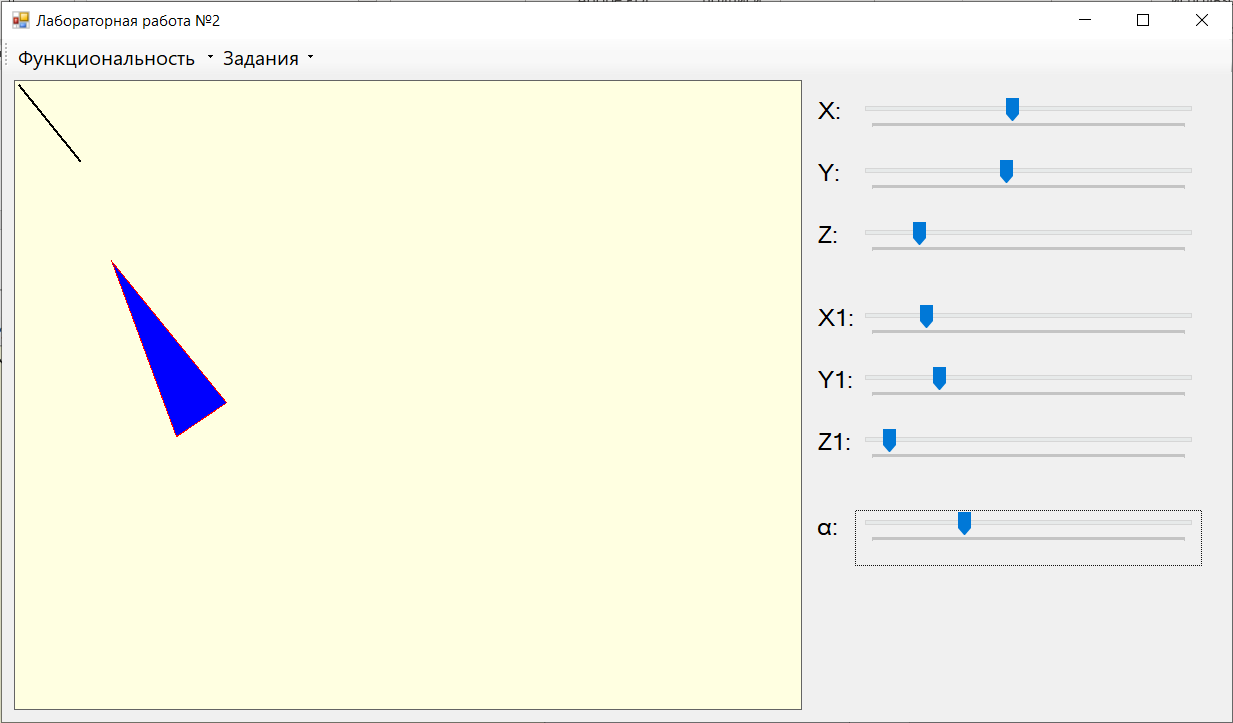


Рис. Вращение вокруг произвольной прямой на угол α

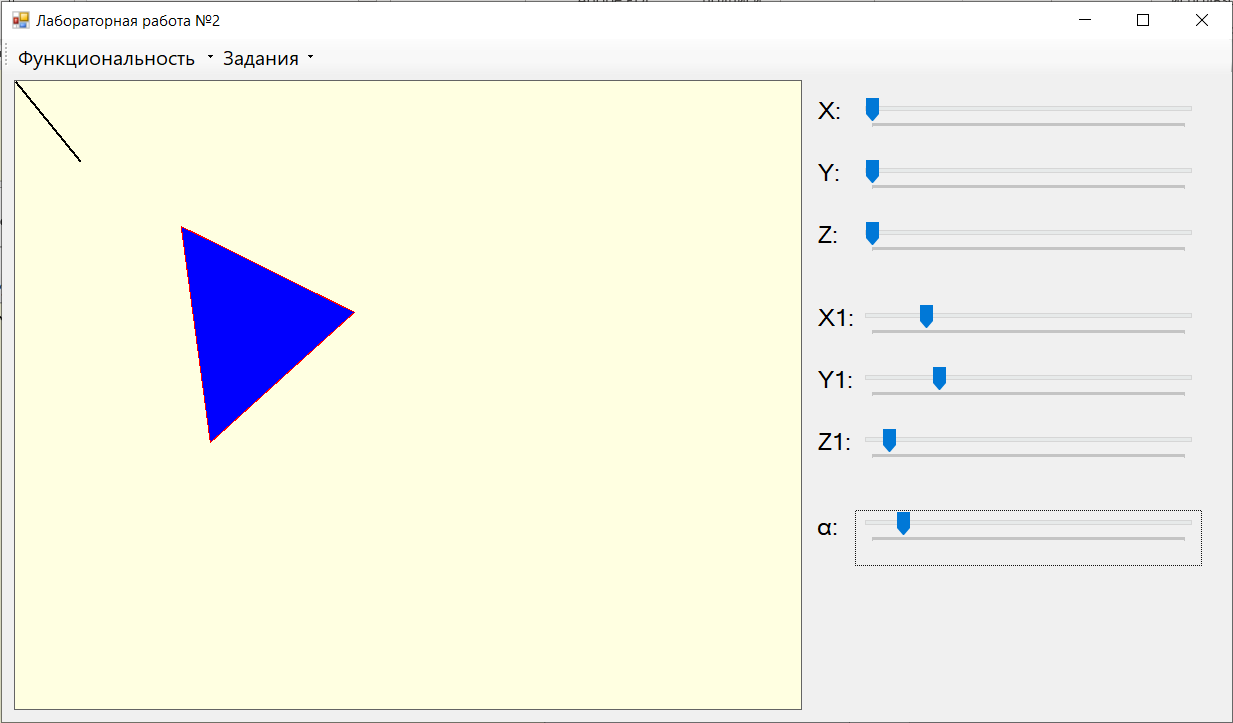


Рис. Вращение вокруг произвольной прямой на угол α

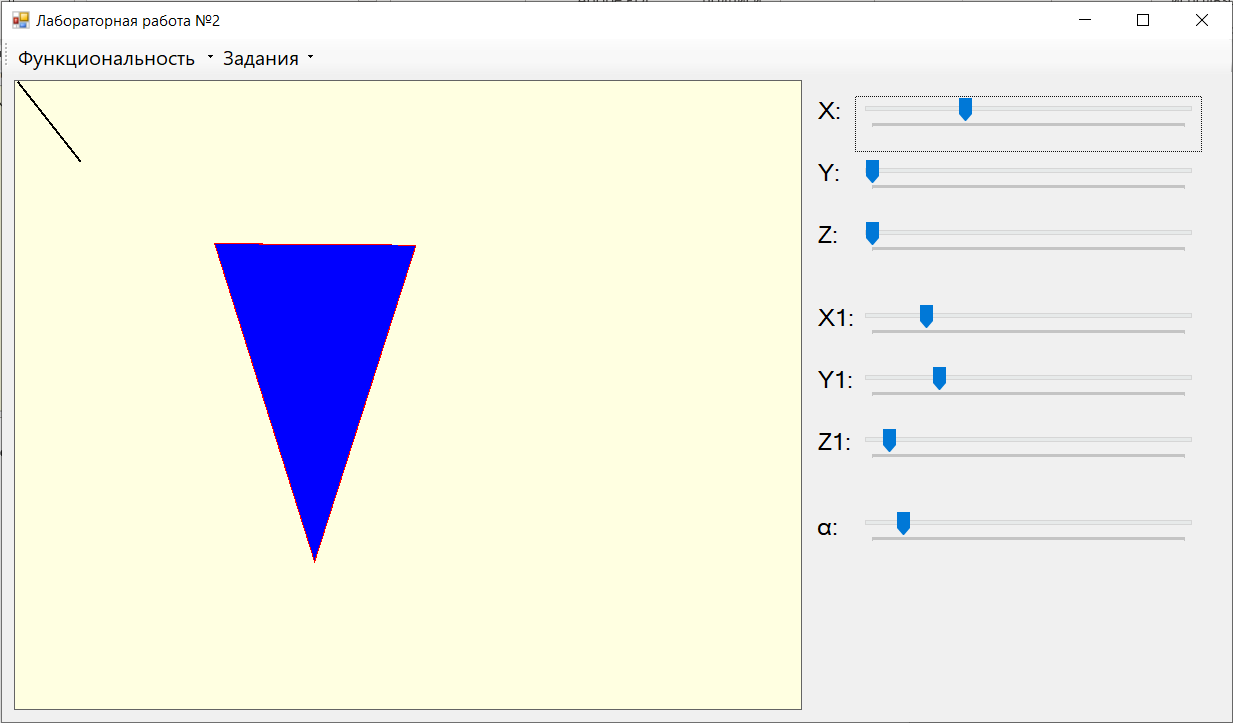


Рис. Произвольное вращение

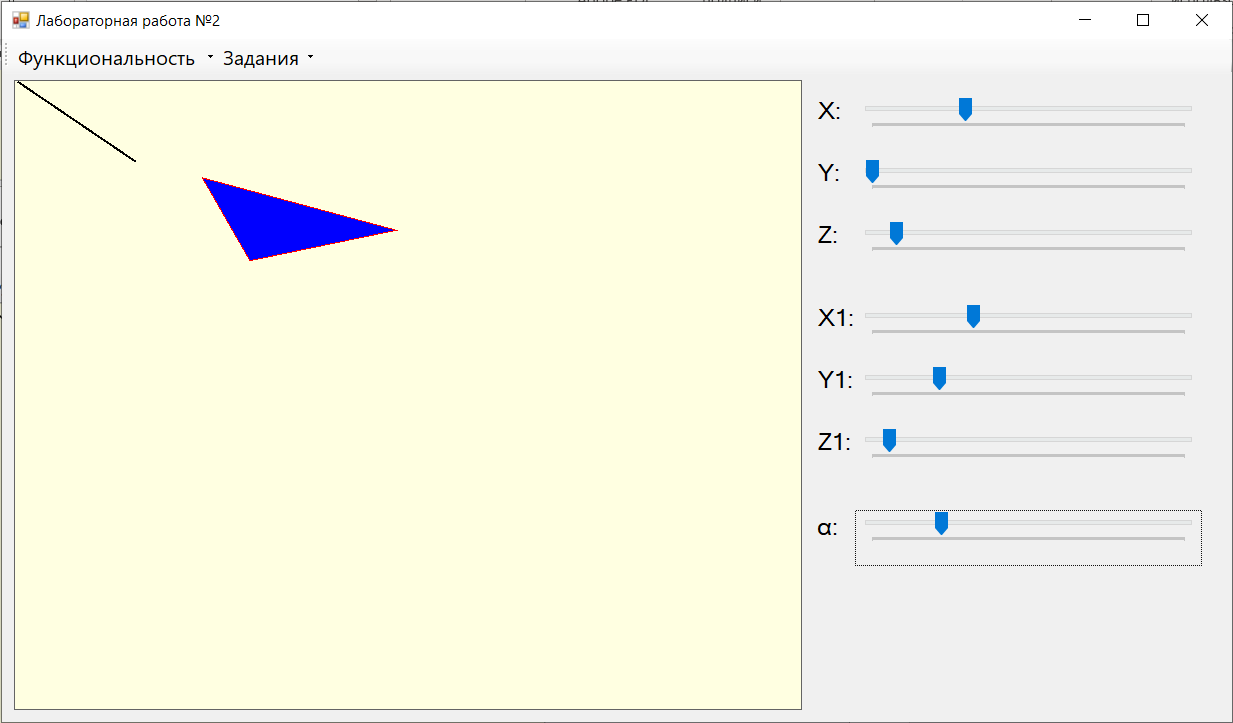


Рис. Произвольное вращение

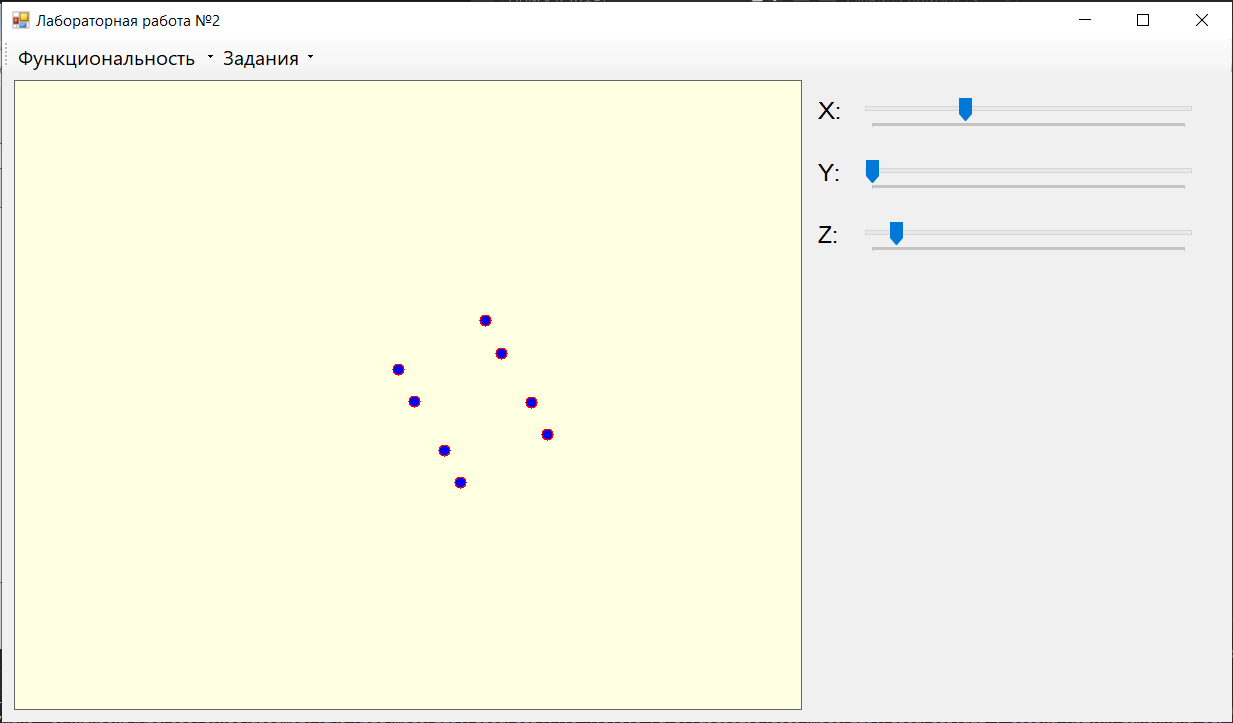


Рис. Визуализация вершин куба

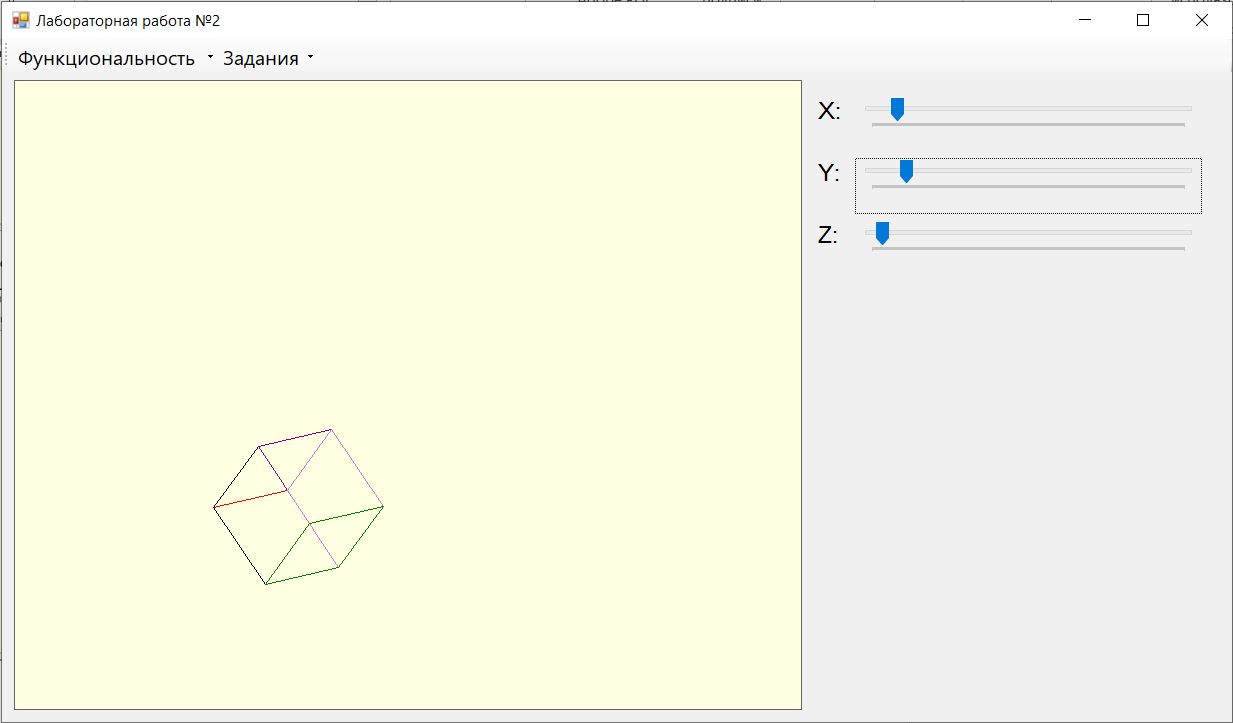


Рис. Визуализация граней куба

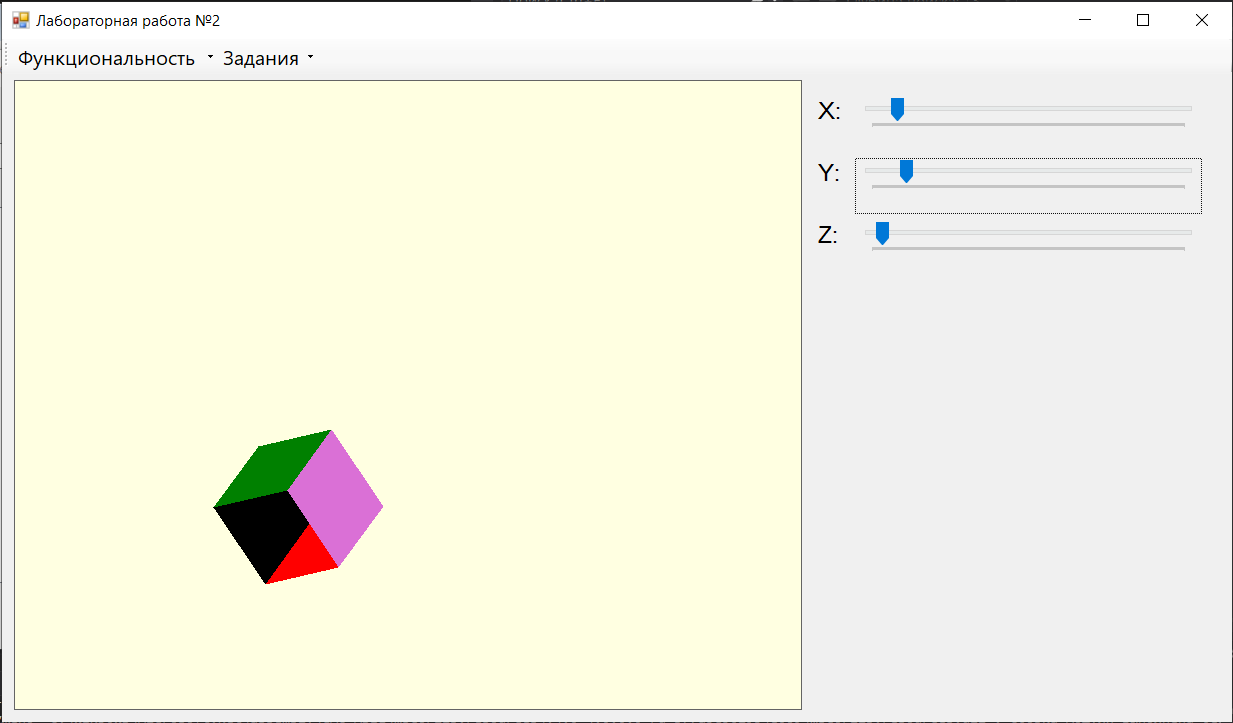


Рис. Изображение сторон куба. Удаление скрытых поверхностей

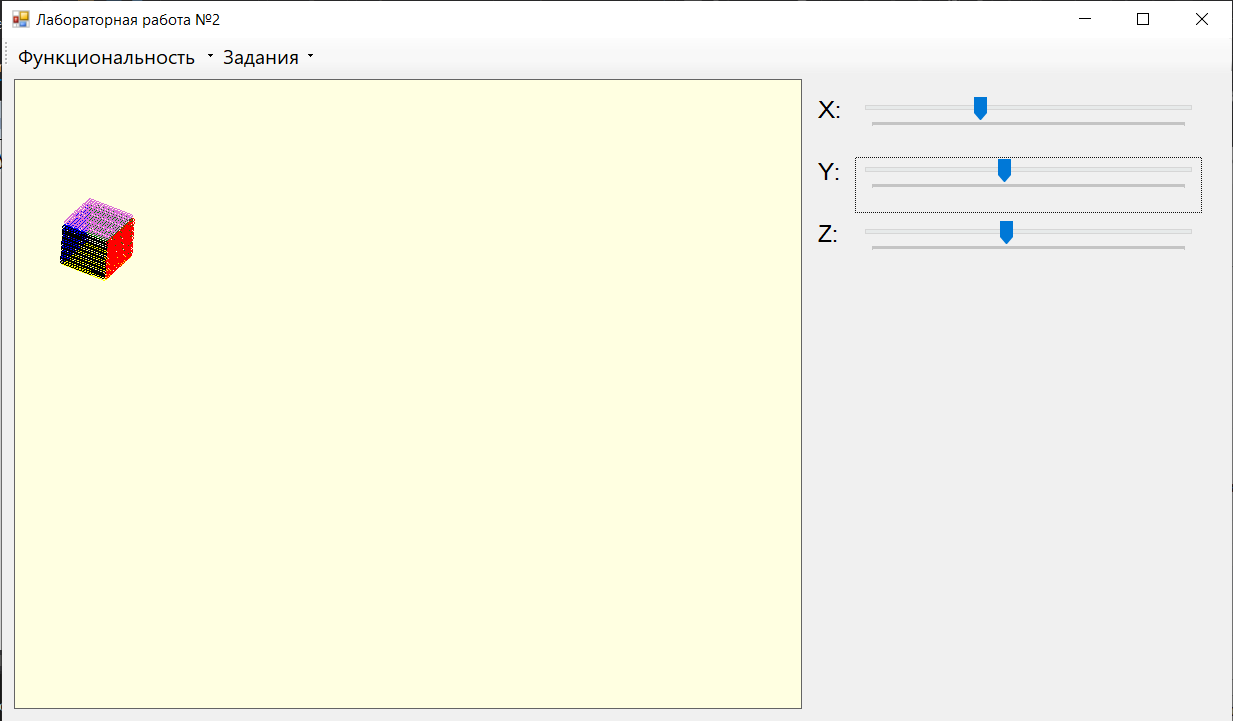


Рис. Параллелепипед, как совокупность точек

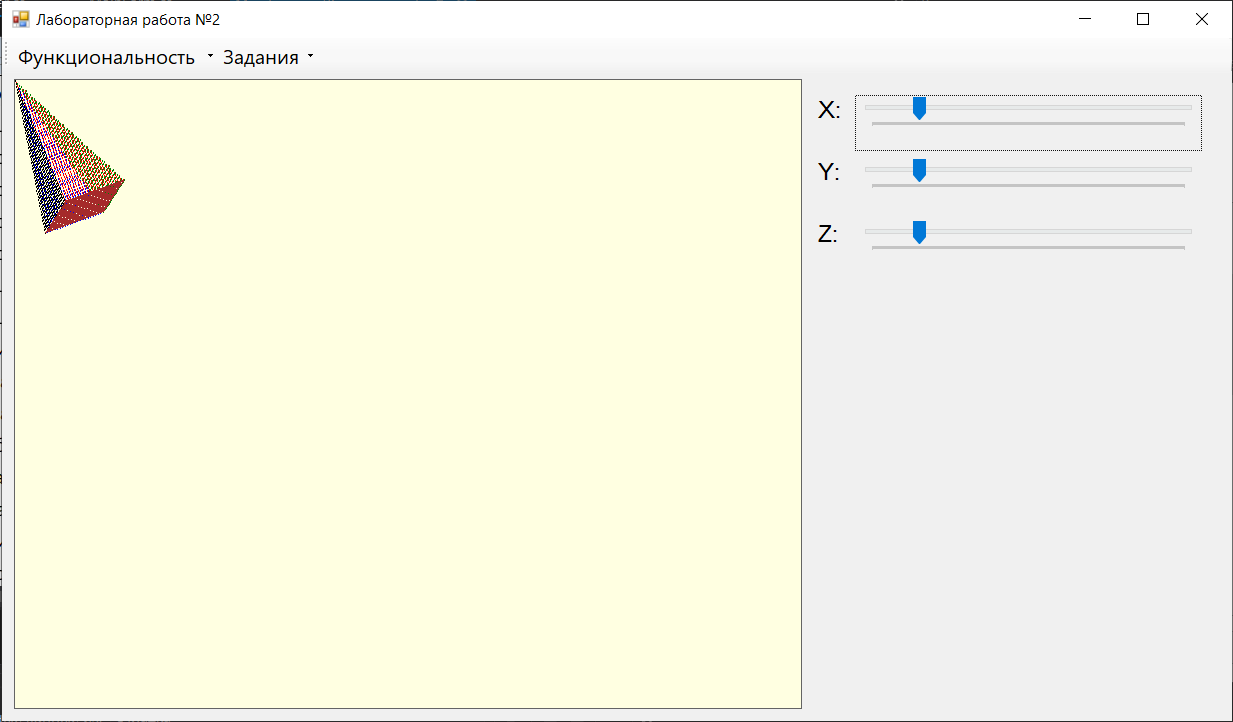


Рис. Пирамида как совокупность точек

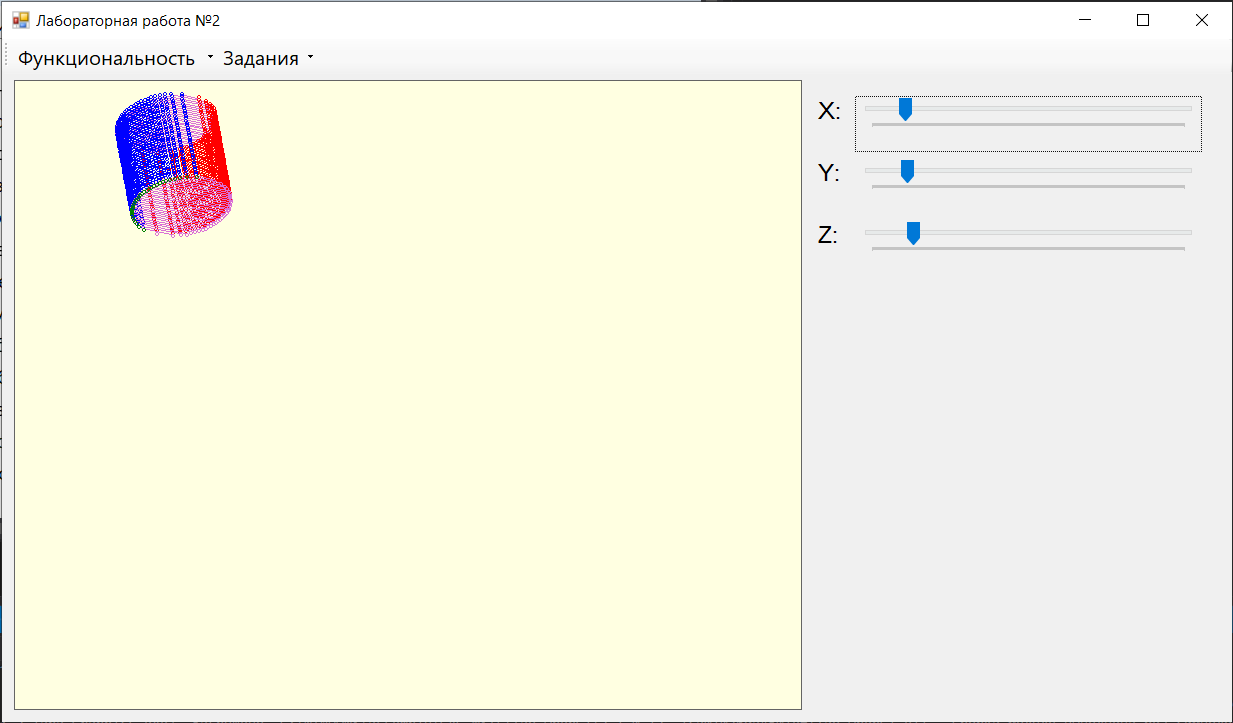


Рис. Цилиндр как совокупность точек

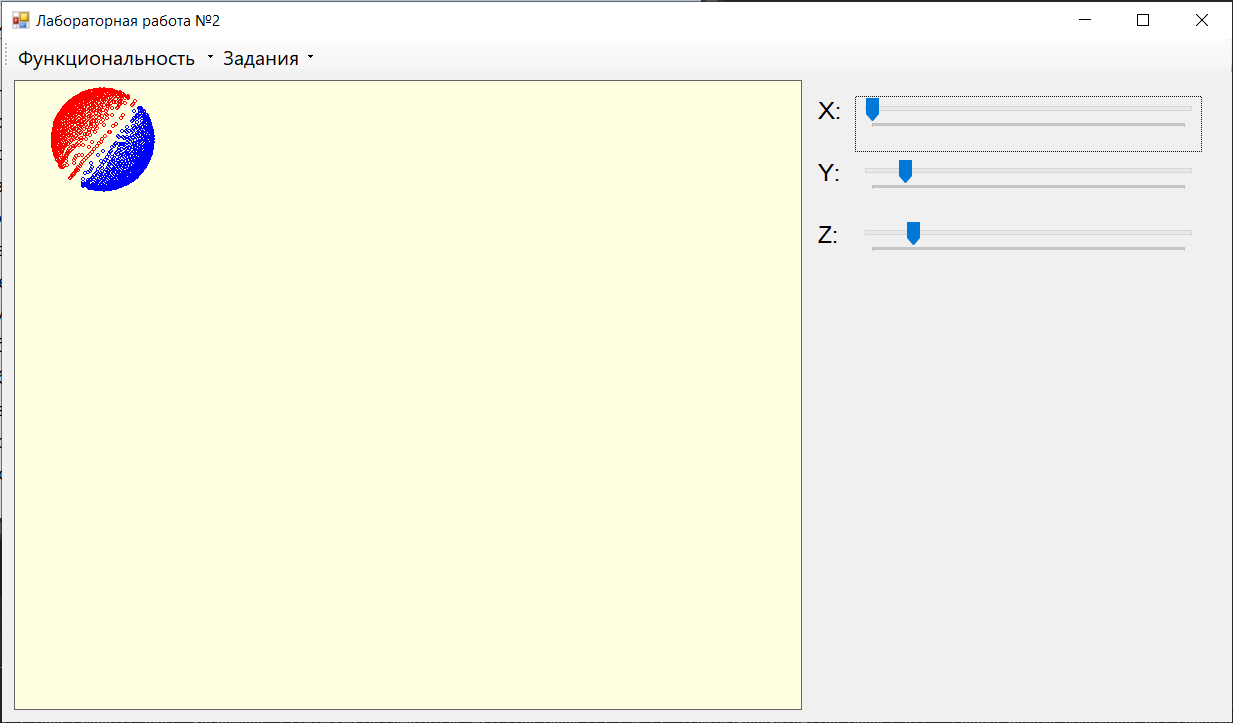


Рис. Сфера как совокупность точек

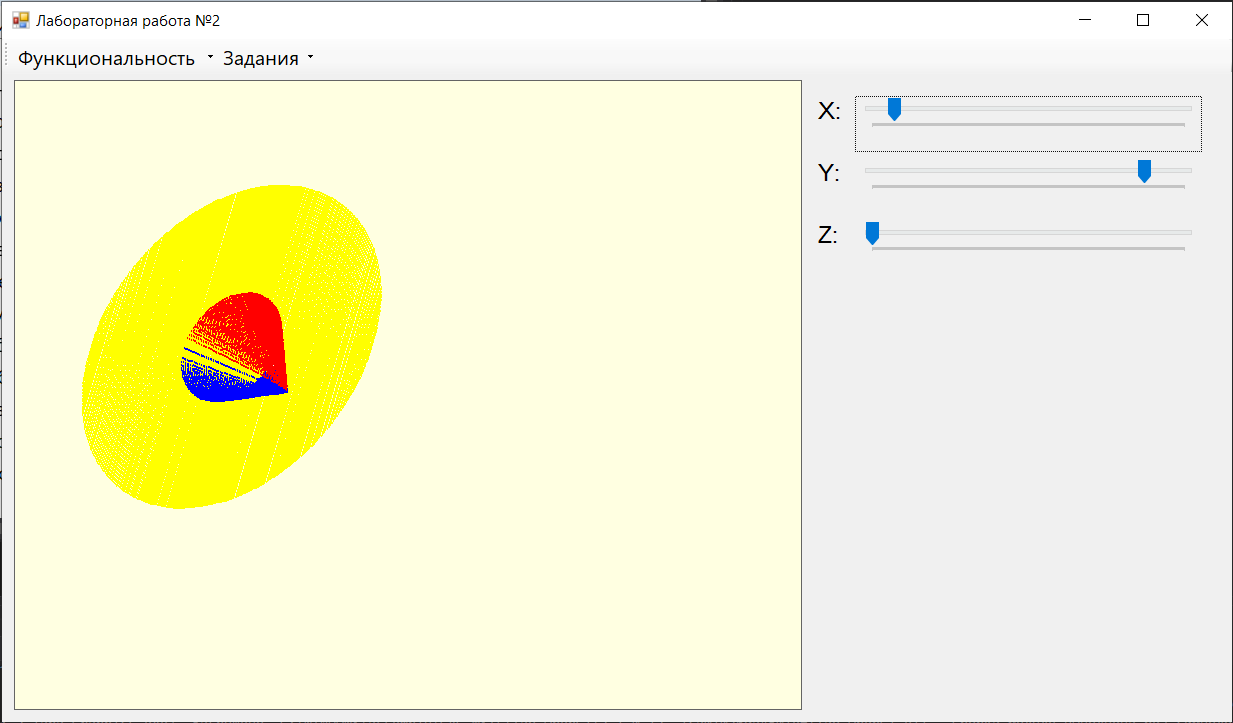


Рис. Конус как совокупность точек

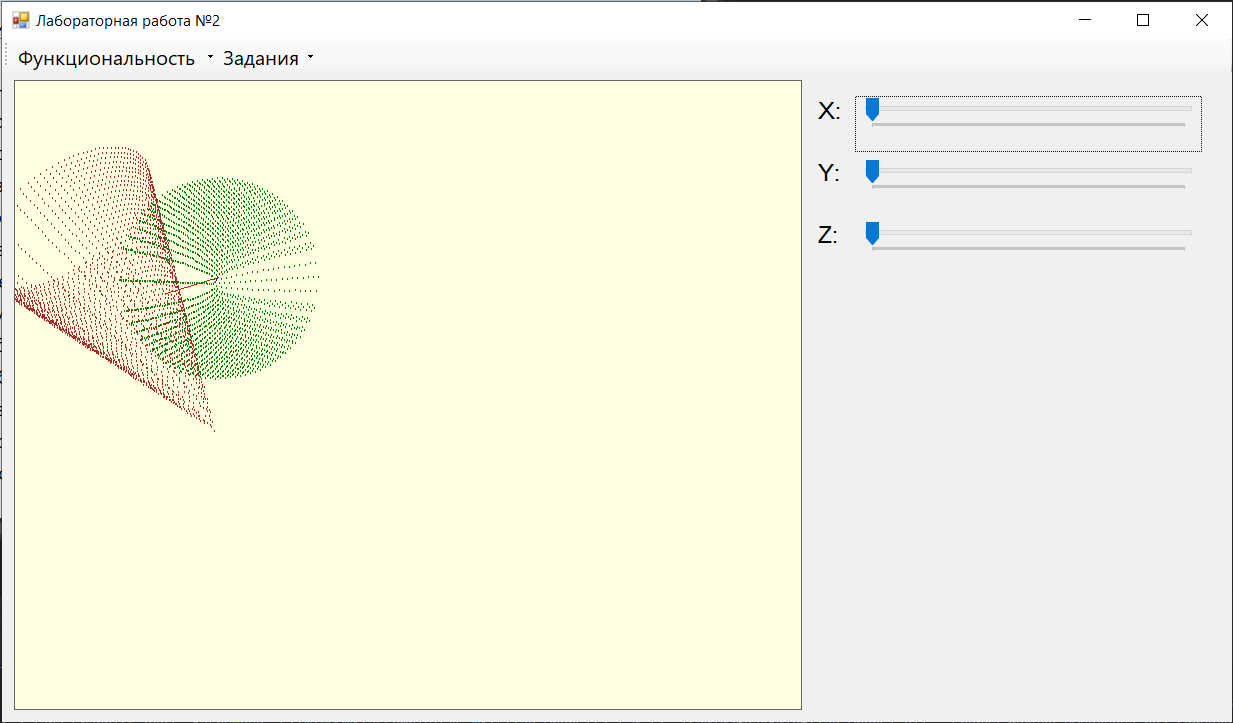


Рис. Комбинация двух конусов

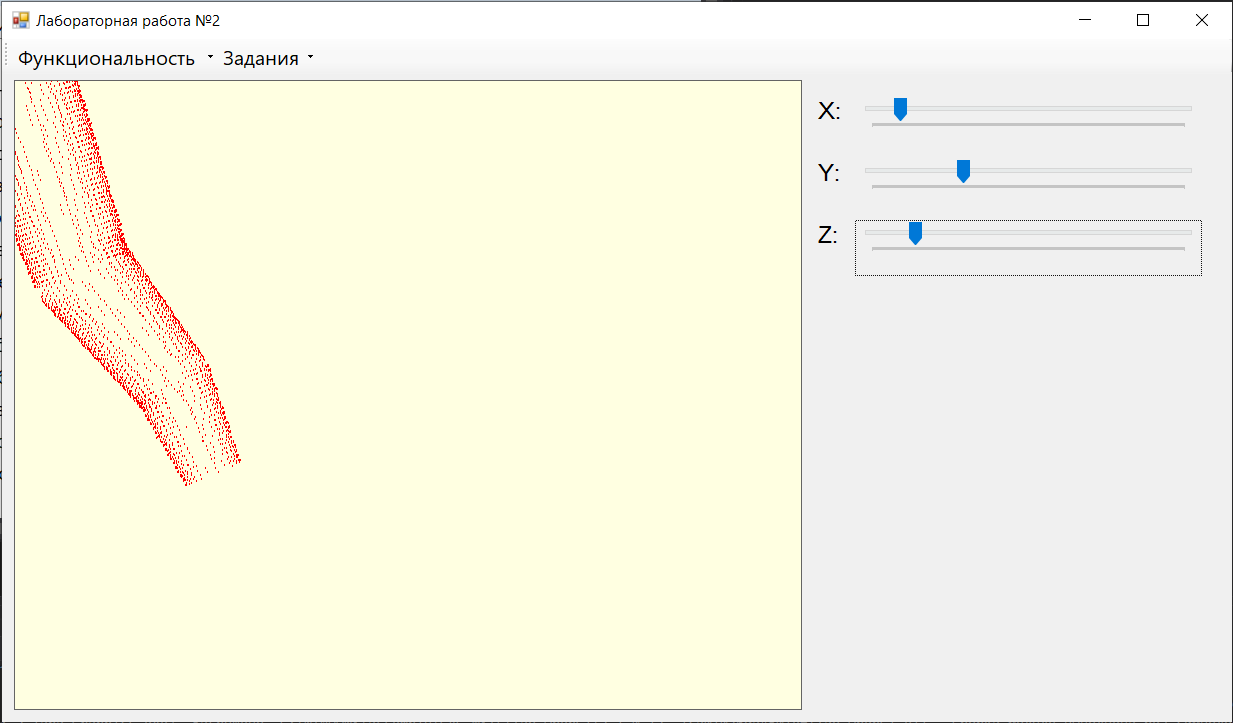


Рис. Объединение трех усеченных конусов

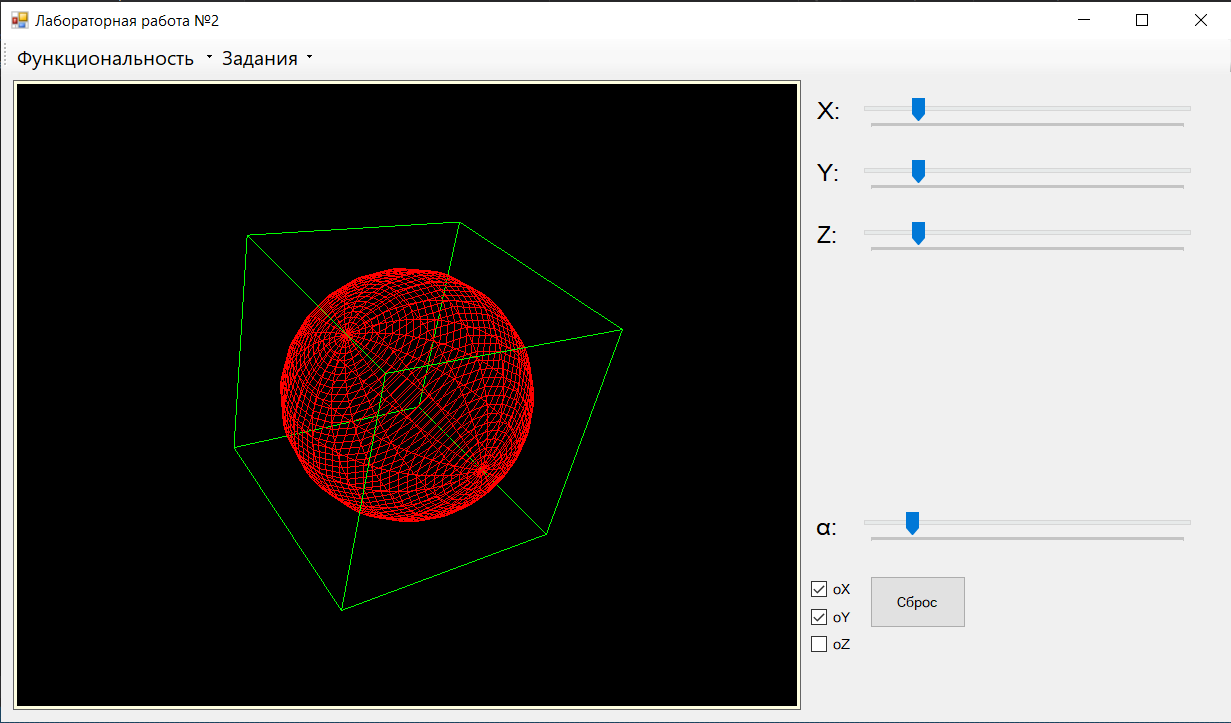


Рис. Сфера вписанная в куб

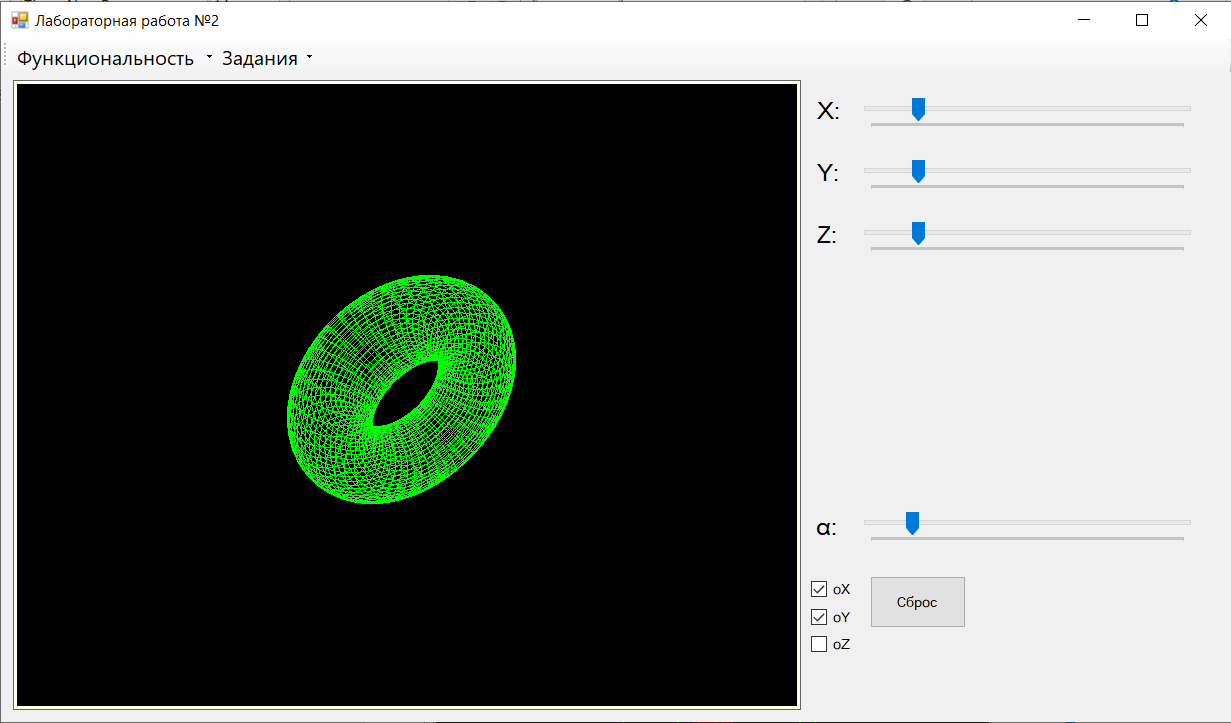


Рис. Тор

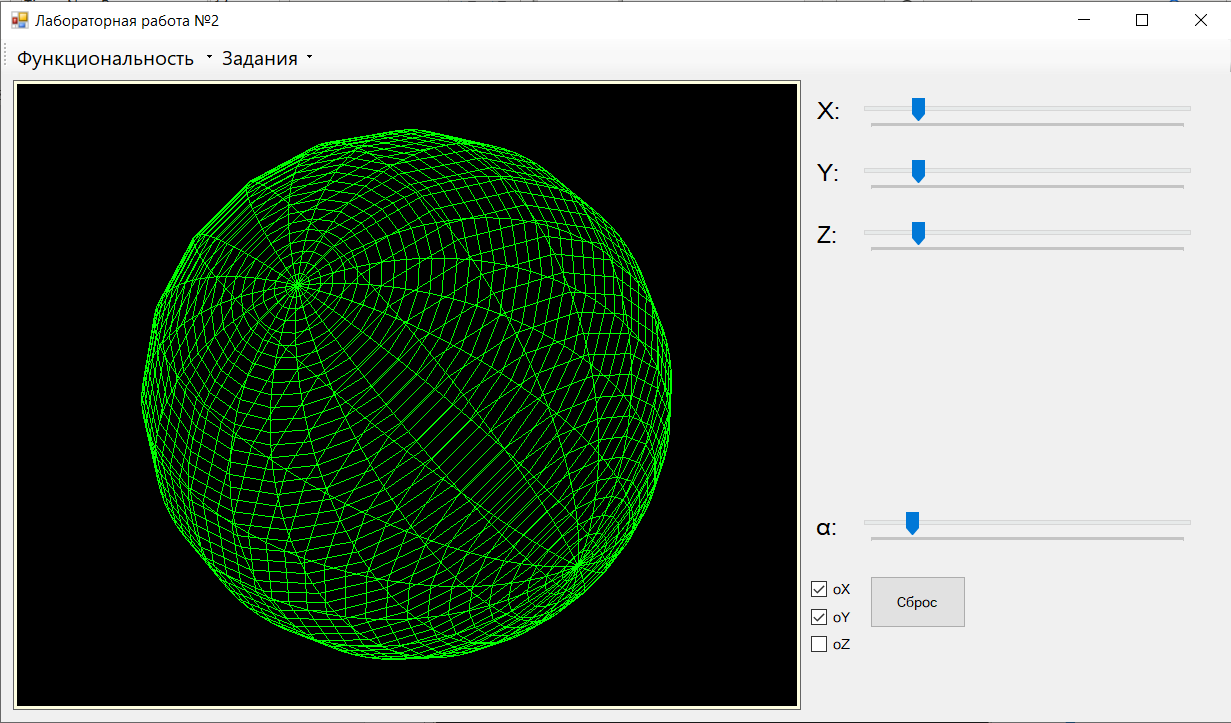


Рис. Эллипсоид вращения

Изображение выглядит как текст

Автоматически созданное описание

Рис. Эллипсоид вращения

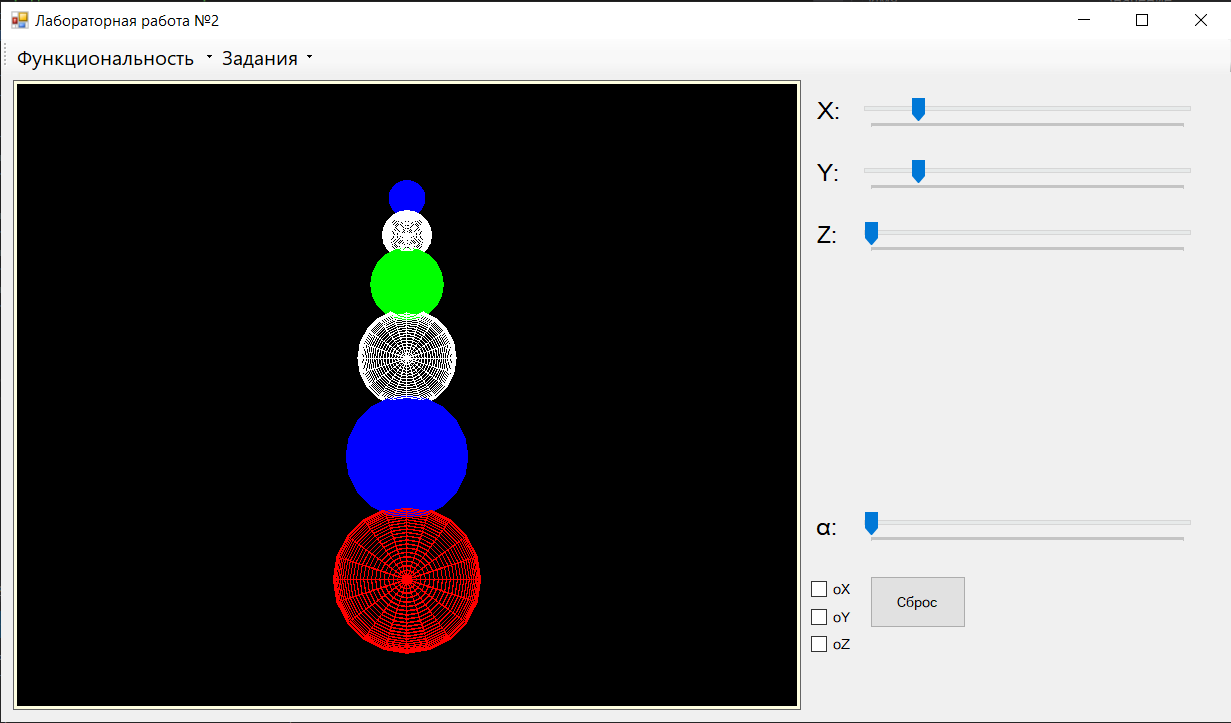


Рис. Шесть сфер разного диаметра установленных друг на друга

1. **Вывод:**

В рамках лабораторной работы я ознакомился со средствами для работы с трехмерной графикой и на практике опробовал некоторые из них в работе.

Приложение 1. Листинг программы:

**MainForm.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using Tao.OpenGl; // для работы с библиотекой OpenGL

using Tao.FreeGlut; // для работы с библиотекой FreeGLUT

using Tao.Platform.Windows;// для работы с элементом управления SimpleOpenGLControl

using Tao.DevIl;// ТЕКСТУРЫ

using OpenTK;

namespace \_2\_laba\_Graphics

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

siOGlC.InitializeContexts();

}

public int mode = 0;

Double factor = Math.PI / 180;

double X = 1;

double Y = 1;

double Z = 1;

int X0 = 0;

int Y0 = 0;

int Z0 = 0;

double angle\_alpha = 0;

int scrollAxis = 0;

Example\_1 ex1;

Example\_2 ex2;

Example\_3 ex3;

Example\_4 ex4;

Example\_5 ex5;

Example\_6 ex6;

Example\_7 ex7;

Example\_8 ex8;

Example\_9 ex9;

Example\_10 ex10;

Example\_11 ex11;

Example\_12 ex12;

Example\_13 ex13;

Example\_14 ex14;

Example\_15 ex15;

Example\_16 ex16;

Example\_17 ex17;

Example\_18 ex18;

Example\_19 ex19;

private void trackBar\_Pitch\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

scrollAxis = 0;

X = factor \* trackBar\_Pitch.Value;

panel\_drawing.Refresh();

}

private void trackBar\_Yaw\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

scrollAxis = 1;

Y = factor \* trackBar\_Yaw.Value;

panel\_drawing.Refresh();

}

private void trackBar\_Roll\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

scrollAxis = 2;

Z = factor \* trackBar\_Roll.Value;

panel\_drawing.Refresh();

}

private void trackBar\_PitchOffset\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

X0 = trackBar\_PitchOffset.Value;

panel\_drawing.Refresh();

}

private void trackBar\_YawOffSet\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

Y0 = trackBar\_YawOffSet.Value;

panel\_drawing.Refresh();

}

private void trackBar\_RollOffSet\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

Z0 = trackBar\_RollOffSet.Value;

panel\_drawing.Refresh();

}

private void trackBar\_angle\_alpha\_Scroll(object sender, EventArgs e)

{ // Событие на перемещение ползунка в trackBar

angle\_alpha = factor \* trackBar\_angle\_alpha.Value;

rotated = trackBar\_angle\_alpha.Value;

panel\_drawing.Refresh();

}

private void panel\_drawing\_Paint(object sender, PaintEventArgs e)

{

switch (mode)

{

case 1:

ex1.DrawShape(e.Graphics, scrollAxis, X, Y, Z);

break;

case 2:

ex2.DrawShape(e.Graphics, scrollAxis, X, Y, Z, X0, Y0, Z0);

break;

case 3:

ex3.DrawShape(e.Graphics, X, Y, Z);

break;

case 4:

ex4.DrawShape(e.Graphics, X, Y, Z, angle\_alpha, panel\_drawing.Width);

break;

case 5:

ex5.DrawShape(e.Graphics, X, Y, Z, angle\_alpha, panel\_drawing.Width, X0, Y0, Z0);

break;

case 6:

ex6.DrawShape(e.Graphics, X, Y, Z, angle\_alpha);

break;

case 7:

ex7.DrawShape(e.Graphics, X, Y, Z, angle\_alpha, X0, Y0, Z0);

break;

case 8:

ex8.DrawShape(e.Graphics, X, Y, Z, angle\_alpha, X0, Y0, Z0);

break;

case 9:

ex9.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 10:

ex10.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 11:

ex11.DrawShape(e.Graphics, X, Y, Z);

break;

case 12:

ex12.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 13:

ex13.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 14:

ex14.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 15:

ex15.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 16:

ex16.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 17:

ex17.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 18:

ex18.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 19:

ex19.DrawShape(e.Graphics, X, Y, Z, panel\_drawing.Width, panel\_drawing.Height);

break;

case 20:

break;

}

}

private void Example1\_Click(object sender, EventArgs e)

{ // Выбор 1 примера

ex1 = new Example\_1();

mode = 1;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example2\_Click(object sender, EventArgs e)

{ // Выбор 2 примера

ex2 = new Example\_2();

mode = 2;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = true;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example3\_Click(object sender, EventArgs e)

{ // Выбор 3 примера

ex3 = new Example\_3();

mode = 3;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example4\_Click(object sender, EventArgs e)

{ // Выбор 4 примера

ex4 = new Example\_4();

mode = 4;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = true;

panel\_drawing.Refresh();

}

private void Example5\_Click(object sender, EventArgs e)

{ // Выбор 5 примера

ex5 = new Example\_5();

mode = 5;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = true;

panel\_angle.Visible = true;

panel\_drawing.Refresh();

}

private void Example6\_Click(object sender, EventArgs e)

{ // Выбор 6 примера

ex6 = new Example\_6();

mode = 6;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = true;

panel\_drawing.Refresh();

}

private void Example7\_Click(object sender, EventArgs e)

{ // Выбор 7 примера

ex7 = new Example\_7();

mode = 7;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = true;

panel\_angle.Visible = true;

panel\_drawing.Refresh();

}

private void Example8\_Click(object sender, EventArgs e)

{ // Выбор 8 примера

ex8 = new Example\_8();

mode = 8;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = true;

panel\_angle.Visible = true;

panel\_drawing.Refresh();

}

private void Example9\_Click(object sender, EventArgs e)

{ // Выбор 9 примера

ex9 = new Example\_9();

mode = 9;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example10\_Click(object sender, EventArgs e)

{ // Выбор 10 примера

ex10 = new Example\_10();

mode = 10;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example11\_Click(object sender, EventArgs e)

{ // Выбор 11 примера

pictureBox1.Visible = false;

ex11 = new Example\_11();

mode = 11;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example12\_Click(object sender, EventArgs e)

{ // Выбор 12 примера

pictureBox1.Visible = false;

ex12 = new Example\_12();

mode = 12;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example13\_Click(object sender, EventArgs e)

{ // Выбор 13 примера

pictureBox1.Visible = false;

ex13 = new Example\_13();

mode = 13;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example14\_Click(object sender, EventArgs e)

{ // Выбор 14 примера

pictureBox1.Visible = false;

ex14 = new Example\_14();

mode = 14;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example15\_Click(object sender, EventArgs e)

{ // Выбор 15 примера

pictureBox1.Visible = false;

ex15 = new Example\_15();

mode = 15;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example16\_Click(object sender, EventArgs e)

{ // Выбор 16 примера

pictureBox1.Visible = false;

ex16 = new Example\_16();

mode = 16;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example17\_Click(object sender, EventArgs e)

{ // Выбор 17 примера

pictureBox1.Visible = false;

ex17 = new Example\_17();

mode = 17;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example18\_Click(object sender, EventArgs e)

{ // Выбор 18 примера

pictureBox1.Visible = false;

ex18 = new Example\_18();

ex18.FillClippedCone();

mode = 18;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = false;

panel\_drawing.Refresh();

}

private void Example19\_Click(object sender, EventArgs e)

{ // Выбор 19 примера

//pictureBox1.Visible = false;

//ex19 = new Example\_19();

//mode = 19;

//panel\_XYZ.Visible = true;

//panel\_XYZ1.Visible = false;

//panel\_angle.Visible = false;

//panel\_drawing.Refresh();

//siOGlC.Visible = false;

//panel\_Check\_XYZ.Visible = false;

comand = 1;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = true;

siOGlC.Visible = true;

panel\_Check\_XYZ.Visible = true;

}

private void Example20\_Click(object sender, EventArgs e)

{ // Выбор 20 примера

//\_tor = new DrawTor();

//mode = 20;

comand = 2;

//pictureBox1.Visible = true;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = true;

//panel\_drawing.Refresh();

//button1.Visible = true;

//button2.Visible = true;

siOGlC.Visible = true;

panel\_Check\_XYZ.Visible = true;

}

private void Example21\_Click(object sender, EventArgs e)

{

comand = 3;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = true;

siOGlC.Visible = true;

panel\_Check\_XYZ.Visible = true;

}

private void Example22\_Click(object sender, EventArgs e)

{

comand = 4;

panel\_XYZ.Visible = true;

panel\_XYZ1.Visible = false;

panel\_angle.Visible = true;

siOGlC.Visible = true;

panel\_Check\_XYZ.Visible = true;

}

DrawTor \_tor;

int i = 0;

int typeDraw;

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

{

typeDraw = 0;

timer1.Enabled = false;

timer1.Enabled = true;

}

private void button2\_Click(object sender, EventArgs e)

{

timer1.Enabled = false;

}

public void Draw(int type)

{

pictureBox1.Image = \_tor.DrawingTor(pictureBox1.Width, pictureBox1.Height, 300, type);

timer1.Enabled = true;

}

private void timer1\_Tick(object sender, EventArgs e)

{

Draw(typeDraw);

\_tor.Speed = i + 1;

i++;

}

// ряд вспомогательных переменных

double rotated = 0.0f; // Поворот

float color, color1, color2 = 0.0f;

int comand = 0; // Какая фигура выбрана

int COLORcomand = 2; // Цвет фигуры

int oX = 0, oY = 0, oZ = 0; // Выбор оси поворота

private void checkBox\_oX\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox\_oX.Checked)

oX = 1;

else oX = 0;

}

private void checkBox\_oY\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox\_oY.Checked)

oY = 1;

else oY = 0;

}

private void checkBox\_oZ\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox\_oZ.Checked)

oZ = 1;

else oZ = 0;

}

private void button\_sbros\_Click(object sender, EventArgs e)

{

oX = oY = oZ = 0;

X = Y = Z = 1;

trackBar\_angle\_alpha.Value = (int)(1 \* factor);

trackBar\_Pitch.Value = 57;

trackBar\_Roll.Value = 57;

trackBar\_Yaw.Value = 1;

trackBar\_PitchOffset.Value = 1;

trackBar\_RollOffSet.Value = 1;

trackBar\_YawOffSet.Value = 1;

checkBox\_oX.Checked = false;

checkBox\_oY.Checked = false;

checkBox\_oZ.Checked = false;

}

private void siOGlC\_Paint(object sender, PaintEventArgs e)

{

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

Gl.glLoadIdentity();

switch (COLORcomand)

{

case 1: color = 1.0f; color2 = 0.0f; Gl.glColor3f(color, color1, color2); break;

case 2: color1 = 5.0f; color = 0.0f; Gl.glColor3f(color, color1, color2); break;

case 3: color2 = 9.0f; color1 = 0.0f; Gl.glColor3f(color, color1, color2); break;

}

Gl.glPushMatrix();

Gl.glTranslated(0, 0, -6); Gl.glRotated(rotated, oX, oY, oZ); Gl.glScaled(X, Y, Z);

switch (comand)

{

case 1: Glut.glutWireCube(2); Gl.glFlush(); Gl.glColor3f(1.0f, 0.0f, 0.0f); Glut.glutWireSphere(1, 20, 50); Gl.glPopMatrix(); Gl.glFlush(); siOGlC.Invalidate(); break;

case 2: Glut.glutWireTorus(0.32, 0.69, 52, 52); Gl.glPopMatrix(); Gl.glFlush(); siOGlC.Invalidate(); break;

case 3: Glut.glutWireSphere(2, 20, 50); Gl.glPopMatrix(); Gl.glFlush(); siOGlC.Invalidate(); break;

case 4:

Gl.glTranslated(0, -1.5, -6); Gl.glColor3f(1.0f, 0.0f, 0.0f); Glut.glutWireSphere(0.6, 20, 50); Gl.glFlush();

Gl.glTranslated(0, 1, 0); Gl.glColor3f(0.0f, 0.0f, 1.0f); Glut.glutSolidSphere(0.5, 20, 50); Gl.glFlush();

Gl.glTranslated(0, 0.8, 0); Gl.glColor3f(1.0f, 1.0f, 1.0f); Glut.glutWireSphere(0.4, 20, 50); Gl.glFlush();

Gl.glTranslated(0, 0.6, 0); Gl.glColor3f(0.0f, 1.0f, 0.0f); Glut.glutSolidSphere(0.3, 20, 50); Gl.glFlush();

Gl.glTranslated(0, 0.4, 0); Gl.glColor3f(1.0f, 1.0f, 1.0f); Glut.glutWireSphere(0.2, 20, 50); Gl.glFlush();

Gl.glTranslated(0, 0.3, 0); Gl.glColor3f(0.0f, 0.0f, 1.0f); Glut.glutSolidSphere(0.15, 20, 50); Gl.glFlush();

Gl.glPopMatrix(); Gl.glFlush(); siOGlC.Invalidate(); break;

}

}

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

{

// инициализация Glut

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

// отчитка окна

Gl.glClearColor(0.0f, 0.0f, 0.0f, 0.0f);

// установка порта вывода в соответствии с размерами элемента siOGlC

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

// настройка проекции

Gl.glMatrixMode(Gl.GL\_PROJECTION); Gl.glLoadIdentity();

Glu.gluPerspective(45, (float)siOGlC.Width / (float)siOGlC.Height, 0.2, 200);

Gl.glMatrixMode(Gl.GL\_MODELVIEW); Gl.glLoadIdentity();

// настройка параметров OpenGL для визуализации

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

}

}

}

**Example\_1.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_1

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

Double[] x2 = new double[3];

Double[] y2 = new double[3];

Double[] z2 = new double[3];

Double[] x3 = new double[3];

Double[] y3 = new double[3];

Double[] z3 = new double[3];

PointF[] points = new PointF[3];

public Example\_1()

{ // Задаем координаты вершин треугольника

x0[0] = 500;

x0[1] = 600;

x0[2] = 700;

y0[0] = 400;

y0[1] = 600;

y0[2] = 500;

z0[0] = 0;

z0[1] = 0;

z0[2] = 0;

}

public Double RotatePitch(double x, double y, double z, double alpha, int i, double[] newX, double[] newY)

{

newX[i] = x;

newY[i] = y \* Math.Cos(alpha) - z \* Math.Sin(alpha);

return y \* Math.Sin(alpha) + z \* Math.Cos(alpha);

}

public Double RotateYaw(double x, double y, double z, double alpha, int i, double[] newX, double[] newY)

{

newX[i] = x \* Math.Cos(alpha) + z \* Math.Sin(alpha);

newY[i] = y;

return -x \* Math.Sin(alpha) + z \* Math.Cos(alpha);

}

public Double RotateRoll(double x, double y, double z, double alpha, int i, double[] newX, double[] newY)

{

newX[i] = x \* Math.Cos(alpha) - y \* Math.Sin(alpha);

newY[i] = x \* Math.Sin(alpha) + y \* Math.Cos(alpha);

return z;

}

public void DrawShape(Graphics graphic, int Axis, double factX, double factY, double factZ)

{

for (Axis = 0; Axis < 3; Axis++)

{

if (Axis == 0)

{

z1[0] = this.RotatePitch(x0[0], y0[0], z0[0], factX, 0, x1, y1);

z1[1] = this.RotatePitch(x0[1], y0[1], z0[1], factX, 1, x1, y1);

z1[2] = this.RotatePitch(x0[2], y0[2], z0[2], factX, 2, x1, y1);

}

else if (Axis == 1)

{

z2[0] = this.RotateYaw(x1[0], y1[0], z1[0], factY, 0, x2, y2);

z2[1] = this.RotateYaw(x1[1], y1[1], z1[1], factY, 1, x2, y2);

z2[2] = this.RotateYaw(x1[2], y1[2], z1[2], factY, 2, x2, y2);

}

else if (Axis == 2)

{

z3[0] = this.RotateRoll(x2[0], y2[0], z2[0], factZ, 0, x3, y3);

z3[1] = this.RotateRoll(x2[1], y2[1], z2[1], factZ, 1, x3, y3);

z3[2] = this.RotateRoll(x2[2], y2[2], z2[2], factZ, 2, x3, y3);

}

}

points[0] = new PointF(Convert.ToSingle(x3[0]), Convert.ToSingle(y3[0]));

points[1] = new PointF(Convert.ToSingle(x3[1]), Convert.ToSingle(y3[1]));

points[2] = new PointF(Convert.ToSingle(x3[2]), Convert.ToSingle(y3[2]));

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_2.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_2

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

Double[] x2 = new double[3];

Double[] y2 = new double[3];

Double[] z2 = new double[3];

Double[] x3 = new double[3];

Double[] y3 = new double[3];

Double[] z3 = new double[3];

int x00, y00, z00;

PointF[] points = new PointF[3];

public Example\_2()

{ // Задаем координаты вершин треугольника

x0[0] = 200;

y0[0] = 100;

x0[1] = 300;

y0[1] = 300;

x0[2] = 400;

y0[2] = 100;

}

public Double RotatePitch(double x, double y, double z, double alpha, int i, double[] newX, double[] newY)

{

newX[i] = x;

newY[i] = y00 + (y - y00) \* Math.Cos(alpha) + (z00 - z) \* Math.Sin(alpha);

return z00 + (y - y00) \* Math.Sin(alpha) + (z - z00) \* Math.Cos(alpha);

}

public Double RotateYaw(double x, double y, double z, double alpha, int i, double[] newX, double[] newY)

{

newX[i] = x00 + (x - x00) \* Math.Cos(alpha) + (z - z00) \* Math.Sin(alpha);

newY[i] = y;

return z00 + (x00 - x) \* Math.Sin(alpha) + (z - z00) \* Math.Cos(alpha);

}

public Double RotateRoll(double x, double y, double z, double alpha, int i, double[] newX, double[] newY)

{

newX[i] = x00 + (x - x00) \* Math.Cos(alpha) + (y00 - y) \* Math.Sin(alpha);

newY[i] = y00 + (x - x00) \* Math.Sin(alpha) + (y - y00) \* Math.Cos(alpha);

return z;

}

public void DrawShape(Graphics graphic, int Axis, double factX, double factY, double factZ, int Xoffset, int Yoffset, int Zoffset)

{

x00 = Xoffset;

y00 = Yoffset;

z00 = Zoffset;

for (Axis = 0; Axis < 3; Axis++)

{

if (Axis == 0)

{

z1[0] = this.RotatePitch(x0[0], y0[0], z0[0], factX, 0, x1, y1);

z1[1] = this.RotatePitch(x0[1], y0[1], z0[1], factX, 1, x1, y1);

z1[2] = this.RotatePitch(x0[2], y0[2], z0[2], factX, 2, x1, y1);

}

else if (Axis == 1)

{

z2[0] = this.RotateYaw(x1[0], y1[0], z1[0], factY, 0, x2, y2);

z2[1] = this.RotateYaw(x1[1], y1[1], z1[1], factY, 1, x2, y2);

z2[2] = this.RotateYaw(x1[2], y1[2], z1[2], factY, 2, x2, y2);

}

else if (Axis == 2)

{

z3[0] = this.RotateRoll(x2[0], y2[0], z2[0], factZ, 0, x3, y3);

z3[1] = this.RotateRoll(x2[1], y2[1], z2[1], factZ, 1, x3, y3);

z3[2] = this.RotateRoll(x2[2], y2[2], z2[2], factZ, 2, x3, y3);

}

}

points[0] = new PointF(Convert.ToSingle(x3[0]), Convert.ToSingle(y3[0]));

points[1] = new PointF(Convert.ToSingle(x3[1]), Convert.ToSingle(y3[1]));

points[2] = new PointF(Convert.ToSingle(x3[2]), Convert.ToSingle(y3[2]));

graphic.DrawEllipse(new Pen(Color.Black, 2), new Rectangle(x00, y00, 7, 7));

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_3.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_3

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

PointF[] points = new PointF[3];

Double[,] m = new double[3, 3];

public Example\_3()

{ // Задаем координаты вершин треугольника

x0[0] = 500;

y0[0] = 400;

x0[1] = 600;

y0[1] = 600;

x0[2] = 700;

y0[2] = 400;

}

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, double[] newX, double[] newY, int i)

{

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll) \* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll) \* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch) \* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch) \* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

newX[i] = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

newY[i] = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

return m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z; ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll)

{

z1[0] = RotateObject(Pitch, Yaw, Roll, x0[0], y0[0], z0[0], x1, y1, 0);

z1[1] = RotateObject(Pitch, Yaw, Roll, x0[1], y0[1], z0[1], x1, y1, 1);

z1[2] = RotateObject(Pitch, Yaw, Roll, x0[2], y0[2], z0[2], x1, y1, 2);

points[0] = new PointF(Convert.ToSingle(x1[0]), Convert.ToSingle(y1[0]));

points[1] = new PointF(Convert.ToSingle(x1[1]), Convert.ToSingle(y1[1]));

points[2] = new PointF(Convert.ToSingle(x1[2]), Convert.ToSingle(y1[2]));

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_4.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_4

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

PointF[] points = new PointF[3];

public Example\_4()

{ // Задаем координаты вершин треугольника

x0[0] = 200;

y0[0] = 100;

x0[1] = 300;

y0[1] = 300;

x0[2] = 400;

y0[2] = 100;

}

public double RotateObject(double Pitch, double Yaw, double Roll, double a, double x, double y, double z, double[] newX, double[] newY, int i)

{

double Temp;

Temp = 1.0 - Math.Cos(a);

newX[i] = x \* (Pitch \* Temp \* Pitch + Math.Cos(a)) + y \* (Yaw \* Temp \* Pitch - Math.Sin(a) \* Roll) + z \* (Roll \* Temp \* Pitch + Math.Sin(a) \* Yaw);

newY[i] = x \* (Pitch \* Temp \* Yaw + Math.Sin(a) \* Roll) + y \* (Yaw \* Temp \* Yaw + Math.Cos(a)) + z \* (Roll \* Temp \* Yaw - Math.Sin(a) \* Pitch);

return x \* (Pitch \* Temp \* Roll - Math.Sin(a) \* Yaw) + y \* (Yaw \* Temp \* Roll + Math.Sin(a) \* Pitch) + z \* (Roll \* Temp \* Roll + Math.Cos(a));

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, double Angle, int Width)

{

z1[0] = RotateObject(Pitch, Yaw, Roll, Angle, x0[0], y0[0], z0[0], x1, y1, 0);

z1[1] = RotateObject(Pitch, Yaw, Roll, Angle, x0[1], y0[1], z0[1], x1, y1, 1);

z1[2] = RotateObject(Pitch, Yaw, Roll, Angle, x0[2], y0[2], z0[2], x1, y1, 2);

points[0] = new PointF(Convert.ToSingle(x1[0]), Convert.ToSingle(y1[0]));

points[1] = new PointF(Convert.ToSingle(x1[1]), Convert.ToSingle(y1[1]));

points[2] = new PointF(Convert.ToSingle(x1[2]), Convert.ToSingle(y1[2]));

double L = 5 \* Width;

PointF[] points1 = new PointF[2];

points1[0] = new PointF(0,0);

points1[1] = new PointF(Convert.ToInt32(L \* Math.Cos(Pitch)), Convert.ToInt32(L \* Math.Cos(Yaw)));

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

graphic.DrawLines(new Pen(Color.Black, 2), points1);

}

}

}

**Example\_5.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_5

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

PointF[] points = new PointF[3];

public Example\_5()

{ // Задаем координаты вершин треугольника

x0[0] = 200;

y0[0] = 100;

x0[1] = 300;

y0[1] = 300;

x0[2] = 400;

y0[2] = 100;

}

public double RotateObject(double Angle, double Pitch, double Yaw, double Roll, double x, double y, double z, double x0, double y0, double z0, double[] newX, double[] newY, int i)

{

double temp;

temp = 1.0 - Math.Cos(Angle);

newX[i] = x0 + (x - x0) \* (Pitch \* temp \* Pitch + Math.Cos(Angle)) + (y - y0) \* (Yaw \* temp \* Pitch - Math.Sin(Angle) \* Roll) + (z - z0) \* (Roll \* temp \* Pitch + Math.Sin(Angle) \* Yaw);

newY[i] = y0 + (x - x0) \* (Pitch \* temp \* Yaw + Math.Sin(Angle) \* Roll) + (y - y0) \* (Yaw \* temp \* Yaw + Math.Cos(Angle)) + (z - z0) \* (Roll \* temp \* Yaw - Math.Sin(Angle) \* Pitch);

return z0 + (x - x0) \* (Pitch \* temp \* Roll - Math.Sin(Angle) \* Yaw) + (y - y0) \* (Yaw \* temp \* Roll + Math.Sin(Angle) \* Pitch) + (z - z0) \* (Roll \* temp \* Roll + Math.Cos(Angle));

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, double Angle, int Width, int Xoffset, int Yoffset, int Zoffset)

{

z1[0] = RotateObject(Angle, Pitch, Yaw, Roll, x0[0], y0[0], z0[0], Xoffset, Yoffset, Zoffset, x1, y1, 0);

z1[1] = RotateObject(Angle, Pitch, Yaw, Roll, x0[1], y0[1], z0[1], Xoffset, Yoffset, Zoffset, x1, y1, 1);

z1[2] = RotateObject(Angle, Pitch, Yaw, Roll, x0[2], y0[2], z0[2], Xoffset, Yoffset, Zoffset, x1, y1, 2);

points[0] = new PointF(Convert.ToSingle(x1[0]), Convert.ToSingle(y1[0]));

points[1] = new PointF(Convert.ToSingle(x1[1]), Convert.ToSingle(y1[1]));

points[2] = new PointF(Convert.ToSingle(x1[2]), Convert.ToSingle(y1[2]));

double L;

L = 5 \* Width;

PointF[] points1 = new PointF[2];

points1[0] = new PointF(0, 0);

points1[1] = new PointF(Convert.ToInt32(L \* Math.Cos(Pitch)), Convert.ToInt32(L \* Math.Cos(Yaw)));

graphic.DrawLines(new Pen(Color.Black, 2), points1);

graphic.DrawEllipse(new Pen(Color.Black, 2), new Rectangle(Xoffset, Yoffset, 7, 7));

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_6.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_6

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

PointF[] points = new PointF[3];

public Example\_6()

{ // Задаем координаты вершин треугольника

x0[0] = 600;

y0[0] = 500;

x0[1] = 700;

y0[1] = 700;

x0[2] = 800;

y0[2] = 500;

}

public double RotateObject(double Angle, double Pitch, double Yaw, double Roll, double x, double y, double z, double[] newX, double[] newY, int i)

{

double a, b, c, d, e;

a = Pitch \* x + Yaw \* y + Roll \* z;

b = Math.Pow(Pitch, 2) + Math.Pow(Yaw, 2) + Math.Pow(Roll, 2);

c = Math.Sqrt(b);

d = Math.Cos(Angle);

e = Math.Sin(Angle);

newX[i] = (Pitch \* a + (x \* (Math.Pow(Yaw, 2) + Math.Pow(Roll, 2) - Pitch \* (Yaw \* y + Roll \* z)) \* d + c \* (-Roll \* y + Yaw \* z) \* e)) / b;

newY[i] = (Yaw \* a + (y \* (Math.Pow(Pitch, 2) + Math.Pow(Roll, 2) - Yaw \* (Pitch \* x + Roll \* z)) \* d + c \* (Roll \* x - Pitch \* z) \* e)) / b;

return (Roll \* a + (z \* (Math.Pow(Pitch, 2) + Math.Pow(Yaw, 2) - Roll \* (Pitch \* x + Yaw \* y)) \* d + c \* (-Yaw \* x + Pitch \* y) \* e)) / b;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, double Angle)

{

z1[0] = RotateObject(Angle, Pitch, Yaw, Roll, x0[0], y0[0], z0[0], x1, y1, 0);

z1[1] = RotateObject(Angle, Pitch, Yaw, Roll, x0[1], y0[1], z0[1], x1, y1, 1);

z1[2] = RotateObject(Angle, Pitch, Yaw, Roll, x0[2], y0[2], z0[2], x1, y1, 2);

points[0] = new PointF(Convert.ToSingle(x1[0]), Convert.ToSingle(y1[0]));

points[1] = new PointF(Convert.ToSingle(x1[1]), Convert.ToSingle(y1[1]));

points[2] = new PointF(Convert.ToSingle(x1[2]), Convert.ToSingle(y1[2]));

PointF[] points1 = new PointF[2];

points1[0] = new PointF(0, 0);

points1[1] = new PointF(Convert.ToInt32(Pitch \* (180 / Math.PI)), Convert.ToInt32(Yaw \* (180 / Math.PI)));

graphic.DrawLines(new Pen(Color.Red, 2), points1);

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_7.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_7

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

PointF[] points = new PointF[3];

public Example\_7()

{ // Задаем координаты вершин треугольника

x0[0] = 200;

y0[0] = 100;

x0[1] = 300;

y0[1] = 300;

x0[2] = 400;

y0[2] = 100;

}

public double RotateObject(double Angle, double Xoffset, double Yoffset, double Zoffset, double Pitch, double Yaw, double Roll, double x, double y, double z, double[] newX, double[] newY, int i)

{

double L = Math.Pow(Pitch, 2) + Math.Pow(Yaw, 2) + Math.Pow(Roll, 2);

double SqrL = Math.Sqrt(L);

double Lu = Math.Pow(Yaw, 2) + Math.Pow(Roll, 2);

double Lv = Math.Pow(Pitch, 2) + Math.Pow(Roll, 2);

double Lw = Math.Pow(Pitch, 2) + Math.Pow(Yaw, 2);

newX[i] = (Xoffset \* Lu + Pitch \* (-Yoffset \* Yaw - Zoffset \* Roll + Pitch \* x + Yaw \* y + Roll \* z)

+ (-Xoffset \* Lu + Pitch \* (Yoffset \* Yaw + Zoffset \* Roll - Yaw \* y - Roll \* z) + Lu \* x) \* Math.Cos(Angle) + SqrL \* (-Zoffset \* Yaw + Yoffset \* Roll - Roll \* y + Yaw \* z) \* Math.Sin(Angle)) / L;

newY[i] = (Yoffset \* Lv + Yaw \* (-Xoffset \* Pitch - Zoffset \* Roll + Pitch \* x + Yaw \* y + Roll \* z)

+ (-Yoffset \* Lv + Yaw \* (Xoffset \* Pitch + Zoffset \* Roll - Pitch \* x - Roll \* z) + Lv \* y) \* Math.Cos(Angle) + SqrL \* (-Zoffset \* Pitch - Xoffset \* Roll + Roll \* x - Pitch \* z) \* Math.Sin(Angle)) / L;

return (Zoffset \* Lw + Roll \* (-Xoffset \* Pitch - Yoffset \* Yaw + Pitch \* x + Yaw \* y + Roll \* z)

+ (-Zoffset \* Lw + Roll \* (Xoffset \* Pitch + Yoffset \* Yaw - Pitch \* x - Yaw \* y) + Lw \* z) \* Math.Cos(Angle) + SqrL \* (-Yoffset \* Pitch + Xoffset \* Yaw - Yaw \* x + Pitch \* y) \* Math.Sin(Angle)) / L; ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, double Angle, int Xoffset, int Yoffset, int Zoffset)

{

z1[0] = RotateObject(Angle, Xoffset, Yoffset, Zoffset, Pitch, Yaw, Roll, x0[0], y0[0], z0[0], x1, y1, 0);

z1[1] = RotateObject(Angle, Xoffset, Yoffset, Zoffset, Pitch, Yaw, Roll, x0[1], y0[1], z0[1], x1, y1, 1);

z1[2] = RotateObject(Angle, Xoffset, Yoffset, Zoffset, Pitch, Yaw, Roll, x0[2], y0[2], z0[2], x1, y1, 2);

points[0] = new PointF(Convert.ToSingle(x1[0]), Convert.ToSingle(y1[0]));

points[1] = new PointF(Convert.ToSingle(x1[1]), Convert.ToSingle(y1[1]));

points[2] = new PointF(Convert.ToSingle(x1[2]), Convert.ToSingle(y1[2]));

PointF[] points1 = new PointF[2];

points1[0] = new PointF(Convert.ToInt32(Xoffset), Convert.ToInt32(Yoffset));

points1[1] = new PointF(Convert.ToInt32(Pitch), Convert.ToInt32(Yaw));

graphic.DrawLines(new Pen(Color.Black, 2), points1);

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_8.cs:**

using System;

using System.Drawing;

namespace \_2\_laba\_Graphics

{

class Example\_8

{

Double[] x0 = new double[3];

Double[] y0 = new double[3];

Double[] z0 = new double[3];

Double[] x1 = new double[3];

Double[] y1 = new double[3];

Double[] z1 = new double[3];

PointF[] points = new PointF[3];

public Example\_8()

{

x0[0] = 200;

y0[0] = 100;

x0[1] = 300;

y0[1] = 300;

x0[2] = 400;

y0[2] = 100;

}

public double RotateObject(double Angle, double Xoffset, double Yoffset, double Zoffset, double Pitch, double Yaw, double Roll, double x, double y, double z, double[] newX, double[] newY, int i)

{

double L = Math.Pow(Pitch, 2) + Math.Pow(Yaw, 2) + Math.Pow(Roll, 2);

double SqrL = Math.Sqrt(L);

double Lu = Math.Pow(Yaw, 2) + Math.Pow(Roll, 2);

double Lv = Math.Pow(Pitch, 2) + Math.Pow(Roll, 2);

double Lw = Math.Pow(Pitch, 2) + Math.Pow(Yaw, 2);

newX[i] = (Xoffset \* Lu + Pitch \* (-Yoffset \* Yaw - Zoffset \* Roll + Pitch \* x + Yaw \* y

+ Roll \* z) + ((x - Xoffset) \* Lu + Pitch \* (Yoffset \* Yaw + Zoffset \* Roll - Yaw \* y

- Roll \* z) + Lu \* x) \* Math.Cos(Angle) + SqrL \* (-Zoffset \* Yaw + Yoffset \* Roll - Roll \* y + Yaw \* z) \* Math.Sin(Angle)) / L;

newY[i] = (Yoffset \* Lv + Yaw \* (-Xoffset \* Pitch - Zoffset \* Roll + Pitch \* x + Yaw \* y + Roll \* z) + ((y - Yoffset) \* Lv + Yaw \* (Xoffset \* Pitch + Zoffset \* Roll - Pitch \* x - Roll \* z) + Lv \* y) \* Math.Cos(Angle) + SqrL \* (-Zoffset \* Pitch - Xoffset \* Roll + Roll \* x - Pitch \* z) \* Math.Sin(Angle)) / L;

return (Zoffset \* Lw + Roll \* (-Xoffset \* Pitch - Yoffset \* Yaw + Pitch \* x + Yaw \* y + Roll \* z) + ((z - Zoffset) \* Lw + Roll \* (Xoffset \* Pitch + Yoffset \* Yaw - Pitch \* x - Yaw \* y)

+ Lw \* z) \* Math.Cos(Angle) + SqrL \* (-Yoffset \* Pitch + Xoffset \* Yaw - Yaw \* x + Pitch \* y) \* Math.Sin(Angle)) / L;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, double Angle, int Xoffset, int Yoffset, int Zoffset)

{

z1[0] = RotateObject(Angle, Xoffset, Yoffset, Zoffset, Pitch, Yaw, Roll, x0[0], y0[0], z0[0], x1, y1, 0);

z1[1] = RotateObject(Angle, Xoffset, Yoffset, Zoffset, Pitch, Yaw, Roll, x0[1], y0[1], z0[1], x1, y1, 1);

z1[2] = RotateObject(Angle, Xoffset, Yoffset, Zoffset, Pitch, Yaw, Roll, x0[2], y0[2], z0[2], x1, y1, 2);

points[0] = new PointF(Convert.ToSingle(x1[0]), Convert.ToSingle(y1[0]));

points[1] = new PointF(Convert.ToSingle(x1[1]), Convert.ToSingle(y1[1]));

points[2] = new PointF(Convert.ToSingle(x1[2]), Convert.ToSingle(y1[2]));

PointF[] points1 = new PointF[2];

points1[0] = new PointF(Convert.ToInt32(Xoffset), Convert.ToInt32(Yoffset));

points1[1] = new PointF(Convert.ToInt32(Pitch), Convert.ToInt32(Yaw));

graphic.DrawLines(new Pen(Color.Black, 2), points1);

graphic.DrawPolygon(new Pen(Color.Red, 2), points);

graphic.FillPolygon(new SolidBrush(Color.Blue), points);

}

}

}

**Example\_9.cs:**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_9

{

Double[,] m = new double[3, 3];

public List <Vertex> VertexList = new List<Vertex>();

public List <Side> SideList = new List<Side>();

public Example\_9()

{

VertexList.Add(new Vertex(0, 0, 0));

VertexList.Add(new Vertex(100, 0, 0));

VertexList.Add(new Vertex(100, 0, 100));

VertexList.Add(new Vertex(0, 0, 100));

VertexList.Add(new Vertex(0, 100, 0));

VertexList.Add(new Vertex(100, 100, 0));

VertexList.Add(new Vertex(100, 100, 100));

VertexList.Add(new Vertex(0, 100, 100));

SideList.Add(new Side(VertexList[0], VertexList[1], VertexList[2], VertexList[3]));

SideList.Add(new Side(VertexList[4], VertexList[5], VertexList[6], VertexList[7]));

SideList.Add(new Side(VertexList[6], VertexList[5], VertexList[1], VertexList[2]));

SideList.Add(new Side(VertexList[7], VertexList[4], VertexList[0], VertexList[3]));

SideList.Add(new Side(VertexList[0], VertexList[4], VertexList[5], VertexList[1]));

SideList.Add(new Side(VertexList[7], VertexList[6], VertexList[2], VertexList[3]));

}

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll) \* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll) \* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch) \* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch) \* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int ArraySize;

ArraySize = 8;

double[] NewX = new double[ArraySize + 1], NewY = new double[ArraySize + 1], NewZ = new double[ArraySize + 1];

int i;

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

for (i = 0; i <= ArraySize - 1; i++)

{

NewZ[i] = RotateObject(Pitch, Yaw, Roll, VertexList[i].x, VertexList[i].y, VertexList[i].z, ref NewX[i], ref NewY[i]);

Rectangle MyBox = new Rectangle(Convert.ToInt32(NewX[i]), Convert.ToInt32(NewY[i]), 10, 10);

graphic.DrawEllipse(new Pen(Color.Red, 2), MyBox);

graphic.FillEllipse(new SolidBrush(Color.Blue), MyBox);

}

}

}

}

**Example\_10.cs:**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_10

{

Double[,] m = new double[3, 3];

public List<Side> Side\_List = new List<Side>();

public Example\_10()

{

Side\_List.Add(new Side(new Vertex(300, 300, 300), new Vertex(400, 300, 300), new Vertex(400, 300, 400), new Vertex(300, 300, 400)));

Side\_List.Add(new Side(new Vertex(300, 300, 400), new Vertex(300, 400, 400), new Vertex(400, 400, 400), new Vertex(400, 300, 400)));

Side\_List.Add(new Side(new Vertex(300, 300, 400), new Vertex(300, 400, 400), new Vertex(300, 400, 300), new Vertex(300, 300, 300)));

Side\_List.Add(new Side(new Vertex(300, 300, 300), new Vertex(300, 400, 300), new Vertex(400, 400, 300), new Vertex(400, 300, 300)));

Side\_List.Add(new Side(new Vertex(400, 300, 300), new Vertex(400, 300, 400), new Vertex(400, 400, 400), new Vertex(400, 400, 300)));

Side\_List.Add(new Side(new Vertex(300, 400, 400), new Vertex(300, 400, 300), new Vertex(400, 400, 300), new Vertex(400, 400, 400)));

}

public Side RotateSide(double Pitch, double Yaw, double Roll, Side MySide)

{

Vertex x1, x2, x3, x4;

x1 = MySide.x1;

x2 = MySide.x2;

x3 = MySide.x3;

x4 = MySide.x4;

Vertex NewX1, NewX2, NewX3, NewX4;

NewX1 = RotateVertex(Pitch, Yaw, Roll, x1);

NewX2 = RotateVertex(Pitch, Yaw, Roll, x2);

NewX3 = RotateVertex(Pitch, Yaw, Roll, x3);

NewX4 = RotateVertex(Pitch, Yaw, Roll, x4);

Side NewSide = new Side(NewX1, NewX2, NewX3, NewX4);

return NewSide;

}

public Vertex RotateVertex(double Pitch, double Yaw, double Roll, Vertex MyVertex)

{

int x, y, z;

int NewX, NewY, NewZ;

x = Convert.ToInt32(MyVertex.x);

y = Convert.ToInt32(MyVertex.y);

z = Convert.ToInt32(MyVertex.z);

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

NewX = Convert.ToInt32(m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z);

NewY = Convert.ToInt32(m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z);

NewZ = Convert.ToInt32(m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z);

Vertex NewVertex = new Vertex(NewX, NewY, NewZ);

return NewVertex;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int Side\_Count;

Side\_Count = Side\_List.Count;

int i;

SolidBrush[] MyBrush = new[] { new SolidBrush(Color.Blue), new SolidBrush(Color.Red), new SolidBrush(Color.Black), new SolidBrush(Color.Purple), new SolidBrush(Color.Orchid), new SolidBrush(Color.Green) };

Pen[] MyPen = new[] { new Pen(Color.Blue, 1), new Pen(Color.Red, 1), new Pen(Color.Black, 1), new Pen(Color.Purple, 1), new Pen(Color.Orchid, 1), new Pen(Color.Green, 1) };

for (i = 0; i <= Side\_Count - 1; i++)

{

Side x, newx;

x = Side\_List[i];

newx = RotateSide(Pitch, Yaw, Roll, x);

PointF[] curvePoints = new[] { new PointF(Convert.ToSingle(newx.x1.x), Convert.ToSingle(newx.x1.y)), new PointF(Convert.ToSingle(newx.x2.x), Convert.ToSingle(newx.x2.y)), new PointF(Convert.ToSingle(newx.x3.x), Convert.ToSingle(newx.x3.y)), new PointF(Convert.ToSingle(newx.x4.x), Convert.ToSingle(newx.x4.y)) };

FillMode newFillMode = FillMode.Winding;

graphic.DrawPolygon(MyPen[i], curvePoints);

}

int XStart, YStart;

XStart = Convert.ToInt32(688 / (double)2);

YStart = Convert.ToInt32(387 / (double)2);

Matrix MyMatrix = new Matrix();

MyMatrix.Translate(XStart, YStart);

graphic.Transform = MyMatrix;

}

}

}

**Example\_11.cs:**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

public class Example\_11

{

List<Side> Side\_List = new List<Side>();

Double[,] m = new double[3, 3];

public Example\_11()

{

Side\_List.Add(new Side(new Vertex(300, 300, 300), new Vertex(400, 300, 300), new Vertex(400, 300, 400), new Vertex(300, 300, 400)));

Side\_List.Add(new Side(new Vertex(300, 300, 400), new Vertex(300, 400, 400), new Vertex(400, 400, 400), new Vertex(400, 300, 400)));

Side\_List.Add(new Side(new Vertex(300, 300, 400), new Vertex(300, 400, 400), new Vertex(300, 400, 300), new Vertex(300, 300, 300)));

Side\_List.Add(new Side(new Vertex(300, 300, 300), new Vertex(300, 400, 300), new Vertex(400, 400, 300), new Vertex(400, 300, 300)));

Side\_List.Add(new Side(new Vertex(400, 300, 300), new Vertex(400, 300, 400), new Vertex(400, 400, 400), new Vertex(400, 400, 300)));

Side\_List.Add(new Side(new Vertex(300, 400, 400), new Vertex(300, 400, 300), new Vertex(400, 400, 300), new Vertex(400, 400, 400)));

}

public Vertex RotateVertex(double Pitch, double Yaw, double Roll, Vertex MyVertex)

{

int x, y, z;

int NewX, NewY, NewZ;

x = Convert.ToInt32(MyVertex.x);

y = Convert.ToInt32(MyVertex.y);

z = Convert.ToInt32(MyVertex.z);

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll) \* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll) \* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch) \* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch) \* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

NewX = System.Convert.ToInt32(m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z);

NewY = System.Convert.ToInt32(m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z);

NewZ = System.Convert.ToInt32(m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z);

Vertex NewVertex = new Vertex(NewX, NewY, NewZ);

return NewVertex;

}

public Side RotateSide(double Pitch, double Yaw, double Roll, Side MySide)

{

Vertex x1, x2, x3, x4;

x1 = MySide.x1;

x2 = MySide.x2;

x3 = MySide.x3;

x4 = MySide.x4;

Vertex NewX1, NewX2, NewX3, NewX4;

NewX1 = RotateVertex(Pitch, Yaw, Roll, x1);

NewX2 = RotateVertex(Pitch, Yaw, Roll, x2);

NewX3 = RotateVertex(Pitch, Yaw, Roll, x3);

NewX4 = RotateVertex(Pitch, Yaw, Roll, x4);

Side NewSide = new Side(NewX1, NewX2, NewX3, NewX4);

return NewSide;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll)

{

int Side\_Count;

Side\_Count = Side\_List.Count;

List<NewSide> New\_Side\_List = new List<NewSide>();

int i;

SolidBrush[] MyBrush = new[] { new SolidBrush(Color.Blue), new SolidBrush(Color.Red), new SolidBrush(Color.Black), new SolidBrush(Color.Purple), new SolidBrush(Color.Orchid), new SolidBrush(Color.Green) };

Side[] x = new Side[Side\_Count + 1], newx = new Side[Side\_Count + 1];

for (i = 0; i <= Side\_Count - 1; i++)

{

x[i] = Side\_List[i];

int a, b, c;

a = Convert.ToInt32((Side\_List[i].x1.x + Side\_List[i].x2.x + Side\_List[i].x3.x + Side\_List[i].x4.x) / (double)4);

b = Convert.ToInt32((Side\_List[i].x1.y + Side\_List[i].x2.y + Side\_List[i].x3.y + Side\_List[i].x4.y) / (double)4);

c = Convert.ToInt32((Side\_List[i].x1.z + Side\_List[i].x2.z + Side\_List[i].x3.z + Side\_List[i].x4.z) / (double)4);

Vertex V = new Vertex(a, b, c);

Vertex NewV;

NewV = RotateVertex(Pitch, Yaw, Roll, V);

int z;

z = Convert.ToInt32(NewV.z);

newx[i] = RotateSide(Pitch, Yaw, Roll, x[i]);

New\_Side\_List.Add(new NewSide(newx[i], z));

}

New\_Side\_List.Sort(new VertexComparer());

for (i = 0; i <= Side\_Count - 1; i++)

{

PointF[] curvePoints = new[] { new PointF(Convert.ToSingle(New\_Side\_List[i].x.x1.x), Convert.ToSingle(New\_Side\_List[i].x.x1.y)), new PointF(Convert.ToSingle(New\_Side\_List[i].x.x2.x), Convert.ToSingle(New\_Side\_List[i].x.x2.y)), new PointF(Convert.ToSingle(New\_Side\_List[i].x.x3.x), Convert.ToSingle(New\_Side\_List[i].x.x3.y)), new PointF(Convert.ToSingle(New\_Side\_List[i].x.x4.x), Convert.ToSingle(New\_Side\_List[i].x.x4.y)) };

FillMode newFillMode = FillMode.Winding;

graphic.FillPolygon(MyBrush[i], curvePoints, newFillMode);

}

int XStart, YStart;

XStart = Convert.ToInt32(688 / (double)2);

YStart = Convert.ToInt32(387 / (double)2);

Matrix MyMatrix = new Matrix();

MyMatrix.Translate(XStart, YStart);

graphic.Transform = MyMatrix;

}

}

}

**Example\_12.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_12

{

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

Double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x0, y0, z0; // Координаты начала отсчета

x0 = 200;

y0 = 200;

z0 = 0;

int a, b, c; // Длины сторон параллепипеда

a = 50;

b = 50;

c = 50;

int i, j, k, l;

int m; // Шаг итерации

m = 3;

Pen MyPen = new Pen(Color.Red);

Pen MyPen1 = new Pen(Color.Blue, 1);

Pen MyPen2 = new Pen(Color.Red, 1);

Pen MyPen3 = new Pen(Color.Green, 1);

Pen MyPen4 = new Pen(Color.Black, 1);

Pen MyPen5 = new Pen(Color.Yellow, 1);

Pen MyPen6 = new Pen(Color.Orchid, 1);

for (i = x0; i <= x0 + a; i += m)

{

for (j = y0; j <= y0 + b; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, i, j, z0, ref newx1, ref newy1);

Rectangle MyBox = new Rectangle(Convert.ToInt32(newx1), Convert.ToInt32(newy1), m, m);

graphic.DrawEllipse(MyPen5, MyBox);

}

}

for (i = z0 + m; i <= z0 + c - m; i += m)

{

for (j = y0; j <= y0 + b; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, x0, j, i, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, x0 + a, j, i, ref newx2, ref newy2);

Rectangle MyBox1 = new Rectangle(System.Convert.ToInt32(newx1), System.Convert.ToInt32(newy1), m, m);

Rectangle MyBox2 = new Rectangle(System.Convert.ToInt32(newx2), System.Convert.ToInt32(newy2), m, m);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

for (k = x0; k <= x0 + a; k += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, k, y0, i, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, k, y0 + b, i, ref newx2, ref newy2);

Rectangle MyBox3 = new Rectangle(System.Convert.ToInt32(newx1), System.Convert.ToInt32(newy1), m, m);

Rectangle MyBox4 = new Rectangle(System.Convert.ToInt32(newx2), System.Convert.ToInt32(newy2), m, m);

graphic.DrawEllipse(MyPen3, MyBox3);

graphic.DrawEllipse(MyPen4, MyBox4);

}

}

for (i = x0; i <= x0 + a; i += m)

{

for (j = y0; j <= y0 + b; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, i, j, z0 + c, ref newx1, ref newy1);

Rectangle MyBox = new Rectangle(Convert.ToInt32(newx1), Convert.ToInt32(newy1), m, m);

graphic.DrawEllipse(MyPen6, MyBox);

}

}

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

}

}

}

**Example\_13.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_13

{

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x0, y0, z0, h, d, m;

x0 = 0;

y0 = 0;

z0 = 0;

h = 150;

d = 70;

m = 3;

double i, j, k;

Pen MyPen1 = new Pen(Color.Blue, 1);

Pen MyPen2 = new Pen(Color.Red, 1);

Pen MyPen3 = new Pen(Color.Green, 1);

Pen MyPen4 = new Pen(Color.Black, 1);

double XMin, Xmax;

for (i = z0; i <= z0 + h; i += m)

{

XMin = -(z0 + i) \* d / (2 \* h);

Xmax = (z0 + i) \* d / (2 \* h);

for (j = XMin; j <= Xmax; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, XMin, j, i, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, Xmax, j, i, ref newx2, ref newy2);

Rectangle MyBox1 = new Rectangle(System.Convert.ToInt32(newx1), System.Convert.ToInt32(newy1), 1, 1);

Rectangle MyBox2 = new Rectangle(System.Convert.ToInt32(newx2), System.Convert.ToInt32(newy2), 1, 1);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

for (k = XMin; k <= Xmax; k += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, k, Xmax, i, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, k, XMin, i, ref newx2, ref newy2);

Rectangle MyBox3 = new Rectangle(System.Convert.ToInt32(newx1), System.Convert.ToInt32(newy1), 1, 1);

Rectangle MyBox4 = new Rectangle(System.Convert.ToInt32(newx2), System.Convert.ToInt32(newy2), 1, 1);

graphic.DrawEllipse(MyPen3, MyBox3);

graphic.DrawEllipse(MyPen4, MyBox4);

}

}

XMin = -(z0 + h) \* d / (double)(2 \* h);

Xmax = (z0 + h) \* d / (double)(2 \* h);

Pen MyPen5 = new Pen(Color.Brown, 1);

for (j = XMin; j <= Xmax; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, j, XMin, z0 + h, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, j, Xmax, z0 + h, ref newx2, ref newy2);

graphic.DrawLine(MyPen5, Convert.ToSingle(newx1), Convert.ToSingle(newy1), Convert.ToSingle(newx2), Convert.ToSingle(newy2));

}

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

}

}

}

**Example\_14.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_14

{

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x0, y0, z0, R, L, m;

x0 = 100;

y0 = 100;

z0 = 100;

R = 50;

L = 100;

m = 3; // Шаг итерации

int i, j;

for (i = x0 - R; i <= x0 + R; i += m)

{

int y1, y2;

y1 = Convert.ToInt32(y0 - Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - x0), 2)));

y2 = Convert.ToInt32(y0 + Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - x0), 2)));

double x11 = 0, y11 = 0, z11 = 0;

double x21 = 0, y21 = 0, z21 = 0;

z11 = RotateObject(Pitch, Yaw, Roll, i, y1, z0, ref x11, ref y11);

z21 = RotateObject(Pitch, Yaw, Roll, i, y2, z0, ref x21, ref y21);

Rectangle MyBox1 = new Rectangle(Convert.ToInt32(x11), Convert.ToInt32(y11), m, m);

Rectangle MyBox2 = new Rectangle(Convert.ToInt32(x21), Convert.ToInt32(y21), m, m);

Pen MyPen1 = new Pen(Color.Green, 1);

Pen MyPen2 = new Pen(Color.Orchid, 1);

graphic.DrawLine(MyPen2, Convert.ToInt32(x11), Convert.ToInt32(y11), Convert.ToInt32(x21), Convert.ToInt32(y21));

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

for (i = z0 - m; i <= z0 + L - m; i += m)

{

for (j = x0 - R; j <= x0 + R; j += m)

{

int y1, y2;

y1 = Convert.ToInt32(y0 - Math.Sqrt(Math.Pow(R, 2) - Math.Pow((j - x0), 2)));

y2 = Convert.ToInt32(y0 + Math.Sqrt(Math.Pow(R, 2) - Math.Pow((j - x0), 2)));

double x11 = 0, y11 = 0, z11 = 0;

double x21 = 0, y21 = 0, z21 = 0;

z11 = RotateObject(Pitch, Yaw, Roll, j, y1, i, ref x11, ref y11);

z21 = RotateObject(Pitch, Yaw, Roll, j, y2, i, ref x21, ref y21);

Rectangle MyBox1 = new Rectangle(Convert.ToInt32(x11), Convert.ToInt32(y11), m, m);

Rectangle MyBox2 = new Rectangle(Convert.ToInt32(x21), Convert.ToInt32(y21), m, m);

Pen MyPen1 = new Pen(Color.Blue, 1);

Pen MyPen2 = new Pen(Color.Red, 1);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

}

for (i = x0 - R; i <= x0 + R; i += m)

{

int y1, y2;

y1 = Convert.ToInt32(y0 - Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - x0), 2)));

y2 = Convert.ToInt32(y0 + Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - x0), 2)));

double x11 = 0, y11 = 0, z11 = 0;

double x21 = 0, y21 = 0, z21 = 0;

z11 = RotateObject(Pitch, Yaw, Roll, i, y1, z0 + L, ref x11, ref y11);

z21 = RotateObject(Pitch, Yaw, Roll, i, y2, z0 + L, ref x21, ref y21);

Rectangle MyBox1 = new Rectangle(Convert.ToInt32(x11), Convert.ToInt32(y11), m, m);

Rectangle MyBox2 = new Rectangle(Convert.ToInt32(x21), Convert.ToInt32(y21), m, m);

Pen MyPen1 = new Pen(Color.Green, 1);

Pen MyPen2 = new Pen(Color.Orchid, 1);

graphic.DrawLine(MyPen2, Convert.ToInt32(x11), Convert.ToInt32(y11), Convert.ToInt32(x21), Convert.ToInt32(y21));

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

}

}

}

**Example\_15.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_15

{

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x0, y0, z0, R, m;

x0 = 100;

y0 = 100;

z0 = 100;

R = 50;

m = 3;

int ZMin, Zmax;

ZMin = z0 - R;

Zmax = z0 + R;

int i, j;

Pen MyPen1 = new Pen(Color.Blue, 1);

Pen MyPen2 = new Pen(Color.Red, 1);

for (i = ZMin; i <= Zmax; i += m)

{

int XMin, XMax;

XMin = Convert.ToInt32(x0 - Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - z0), 2)));

XMax = Convert.ToInt32(x0 + Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - z0), 2)));

int SmallR;

SmallR = Convert.ToInt32(Math.Sqrt(Math.Pow(R, 2) - Math.Pow((i - z0), 2)));

for (j = XMin; j <= XMax; j += m)

{

double y1, y2;

y1 = y0 + Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((j - x0), 2));

y2 = y0 - Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((j - x0), 2));

double NewX1 = 0, NewY1 = 0, NewZ1;

double NewX2 = 0, NewY2 = 0, NewZ2;

NewZ1 = RotateObject(Pitch, Yaw, Roll, j, y1, i, ref NewX1, ref NewY1);

NewZ2 = RotateObject(Pitch, Yaw, Roll, j, y2, i, ref NewX2, ref NewY2);

Rectangle MyBox1 = new Rectangle(Convert.ToInt32(NewX1), Convert.ToInt32(NewY1), m, m);

Rectangle MyBox2 = new Rectangle(Convert.ToInt32(NewX2), Convert.ToInt32(NewY2), m, m);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

}

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

}

}

}

**Example\_16.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_16

{

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x0, y0, z0, h, R, m;

x0 = 200;

y0 = 200;

z0 = 200;

h = 100;

R = 60;

m = 1;

double i, j;

int ZMin, ZMax;

ZMin = z0;

ZMax = z0 + h;

int XMin, XMax;

int SmallR;

SmallR = Convert.ToInt32((R / (double)h) \* (h + z0));

XMin = x0 - SmallR;

XMax = x0 + SmallR;

for (j = XMin; j <= XMax; j += m)

{

int YMin, YMax, x, z;

x = Convert.ToInt32(j);

z = z0;

YMin = Convert.ToInt32(y0 - Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x - x0), 2)));

YMax = Convert.ToInt32(y0 + Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x - x0), 2)));

double NewX1 = 0, NewY1 = 0, NewZ1, Newx2 = 0, NewY2 = 0, NewZ2;

NewZ1 = RotateObject(Pitch, Yaw, Roll, x, YMin, z, ref NewX1, ref NewY1);

NewZ2 = RotateObject(Pitch, Yaw, Roll, x, YMax, z, ref Newx2, ref NewY2);

Pen MyPen1 = new Pen(Color.Yellow, 1);

Pen MyPen2 = new Pen(Color.Blue, 1);

Rectangle MyBox1 = new Rectangle(System.Convert.ToInt32(NewX1), System.Convert.ToInt32(NewY1), 1, 1);

Rectangle MyBox2 = new Rectangle(System.Convert.ToInt32(Newx2), System.Convert.ToInt32(NewY2), 1, 1);

graphic.DrawLine(MyPen1, Convert.ToInt32(NewX1), Convert.ToInt32(NewY1), Convert.ToInt32(Newx2), Convert.ToInt32(NewY2));

}

for (i = ZMin; i <= ZMax; i += m)

{

SmallR = Convert.ToInt32((R / (double)h) \* (h + z0 - i));

XMin = x0 - SmallR;

XMax = x0 + SmallR;

for (j = XMin; j <= XMax; j += m)

{

int YMin, YMax, x, z;

x = Convert.ToInt32(j);

z = Convert.ToInt32(i);

YMin = Convert.ToInt32(y0 - Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x - x0), 2)));

YMax = Convert.ToInt32(y0 + Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x - x0), 2)));

double NewX1 = 0, NewY1 = 0, NewZ1, Newx2 = 0, NewY2 = 0, NewZ2;

NewZ1 = RotateObject(Pitch, Yaw, Roll, x, YMin, z, ref NewX1, ref NewY1);

NewZ2 = RotateObject(Pitch, Yaw, Roll, x, YMax, z, ref Newx2, ref NewY2);

Pen MyPen1 = new Pen(Color.Red, 1);

Pen MyPen2 = new Pen(Color.Blue, 1);

Rectangle MyBox1 = new Rectangle(System.Convert.ToInt32(NewX1), System.Convert.ToInt32(NewY1), 1, 1);

Rectangle MyBox2 = new Rectangle(System.Convert.ToInt32(Newx2), System.Convert.ToInt32(NewY2), 1, 1);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

}

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

}

}

}

**Example\_17.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

using System.Collections.Generic;

namespace \_2\_laba\_Graphics

{

class Example\_17

{

List<Vertex> MyList = new List<Vertex>();

public int a, b, c, H, d, x0, y0, z0, R, m, L;

public Example\_17()

{

x0 = 200;

y0 = 200;

z0 = 0;

R = 100;

m = 5;

L = 200;

a = 50;

b = 100;

c = 150;

H = 300;

d = 150;

double i, j;

int ZMin, ZMax;

ZMin = z0;

ZMax = z0 + H;

for (i = ZMin; i <= ZMax; i += m)

{

int XMin, XMax;

int SmallR;

SmallR = Convert.ToInt32((R / (double)H) \* (H + z0 - i));

XMin = x0 - SmallR;

XMax = x0 + SmallR;

for (j = XMin; j <= XMax; j += m)

{

int YMin, YMax, x, z;

x = Convert.ToInt32(j);

z = Convert.ToInt32(i);

YMin = Convert.ToInt32(y0 - Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x - x0), 2)));

YMax = Convert.ToInt32(y0 + Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x - x0), 2)));

MyList.Add(new Vertex(j, YMin, i));

MyList.Add(new Vertex(j, YMax, i));

}

}

}

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public int RotateAboutAlphaBeta(double alpha, double beta, int x0, int y0, int z0, ref int x, ref int y)

{

int z;

x = Convert.ToInt32(Math.Cos(alpha) \* x0 + Math.Sin(alpha) \* Math.Sin(-beta) \* y0 + Math.Sin(alpha) \* Math.Cos(-beta) \* z0);

y = Convert.ToInt32(Math.Cos(-beta) \* y0 - Math.Sin(-beta) \* z0);

z = Convert.ToInt32(-Math.Sin(alpha) \* x0 + Math.Cos(alpha) \* Math.Sin(-beta) \* y0 + Math.Cos(alpha) \* Math.Cos(-beta) \* z0);

return z;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x1, y1, z1, x2, y2, z2;

double newx0 = 0, newy0 = 0, newz0, newx1 = 0, newy1 = 0, newz1, newx2 = 0, newy2 = 0, newz2;

x1 = x0;

y1 = y0;

z1 = z0 + H;

double alpha, beta;

alpha = Math.PI / 4;

beta = Math.PI / 4;

x2 = Convert.ToInt32(H \* Math.Sin(alpha) \* Math.Cos(beta));

y2 = Convert.ToInt32(H \* Math.Sin(beta));

z2 = Convert.ToInt32(H \* Math.Cos(alpha) \* Math.Cos(beta));

newz0 = RotateObject(Pitch, Yaw, Roll, x0, y0, z0, ref newx0, ref newy0);

newz1 = RotateObject(Pitch, Yaw, Roll, x1, y1, z1, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, x2, y2, z2, ref newx2, ref newy2);

Pen MyPen1 = new Pen(Color.Blue, 1);

Pen MyPen2 = new Pen(Color.Red, 1);

Point Point0 = new Point(Convert.ToInt32(newx0), Convert.ToInt32(newy0));

Point Point1 = new Point(Convert.ToInt32(newx1), Convert.ToInt32(newy1));

Point Point2 = new Point(Convert.ToInt32(newx2), Convert.ToInt32(newy2));

graphic.DrawLine(MyPen1, Point0, Point1);

graphic.DrawLine(MyPen2, Point0, Point2);

int ListCount;

ListCount = MyList.Count - 1;

int i;

Pen MyPen = new Pen(Color.Green, 1);

Pen MyPen3 = new Pen(Color.Brown, 1);

for (i = 0; i <= ListCount; i++)

{

int x, y, z;

double newx = 0, newy = 0, newz;

x = Convert.ToInt32(MyList[i].x);

y = Convert.ToInt32(MyList[i].y);

z = Convert.ToInt32(MyList[i].z);

int myx = 0, myy = 0, myz;

double newmyx = 0, newmyy = 0, newmyz;

myz = RotateAboutAlphaBeta(alpha, beta, x, y, z, ref myx, ref myy);

newmyz = RotateObject(Pitch, Yaw, Roll, myx, myy, myz, ref newmyx, ref newmyy);

newz = RotateObject(Pitch, Yaw, Roll, x, y, z, ref newx, ref newy);

Rectangle MyBox1 = new Rectangle(Convert.ToInt32(newx), Convert.ToInt32(newy), 1, 1);

Rectangle MyBox2 = new Rectangle(Convert.ToInt32(newmyx), Convert.ToInt32(newmyy), 1, 1);

graphic.DrawEllipse(MyPen, MyBox1);

graphic.DrawEllipse(MyPen3, MyBox2);

}

int xstart, ystart;

xstart = Convert.ToInt32(Width / (double)2);

ystart = Convert.ToInt32(Height / (double)2);

Matrix MyMatrix = new Matrix();

MyMatrix.Translate(xstart, ystart);

graphic.Transform = MyMatrix;

}

}

}

**Example\_18.cs:**

using System;

using System.Collections.Generic;

using System.Collections;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_18

{

public List<Vertex> GList = new List<Vertex>();

public Example\_18()

{

}

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public int RotateAboutAlphaBeta(double alpha, double beta, int x0, int y0, int z0, ref int x, ref int y)

{

int z;

x = Convert.ToInt32(Math.Cos(alpha) \* x0 + Math.Sin(alpha) \* Math.Sin(-beta) \* y0 + Math.Sin(alpha) \* Math.Cos(-beta) \* z0);

y = Convert.ToInt32(Math.Cos(-beta) \* y0 - Math.Sin(-beta) \* z0);

z = Convert.ToInt32(-Math.Sin(alpha) \* x0 + Math.Cos(alpha) \* Math.Sin(-beta) \* y0 + Math.Cos(alpha) \* Math.Cos(-beta) \* z0);

return z;

}

public ArrayList GetClippedCone(int Length, int RBig, int RSmall, int m)

{

ArrayList MyList = new ArrayList();

double i, j;

int ZMin, ZMax;

ZMin = 0;

ZMax = Length;

for (i = ZMin; i <= ZMax; i += m)

{

int XMin, XMax;

int SmallR;

SmallR = Convert.ToInt32((RBig - i \* (RBig - RSmall) / Length));

XMin = -SmallR;

XMax = SmallR;

for (j = XMin; j <= XMax; j += m)

{

int YMin, YMax, x, z;

x = Convert.ToInt32(j);

z = Convert.ToInt32(i);

YMin = Convert.ToInt32(-Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x), 2)));

YMax = Convert.ToInt32(Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((x), 2)));

MyList.Add(new Vertex(j, YMin, i));

MyList.Add(new Vertex(j, YMax, i));

}

}

return MyList;

}

public void FillClippedCone()

{

int Segments\_Count;

Segments\_Count = 3;

int[] x = new int[Segments\_Count + 1], y = new int[Segments\_Count + 1], z = new int[Segments\_Count + 1];

double[] alpha = new double[Segments\_Count + 1], beta = new double[Segments\_Count + 1], Length = new double[Segments\_Count + 1];

int[] RBig = new int[Segments\_Count + 1], RSmall = new int[Segments\_Count + 1];

int m;

m = 5;

RBig[0] = 60;

RSmall[0] = 50;

RBig[1] = 50;

RSmall[1] = 40;

RBig[2] = 40;

RSmall[2] = 30;

double factor;

factor = Math.PI / 180;

alpha[0] = factor \* 15;

beta[0] = factor \* 25;

alpha[1] = factor \* 30;

beta[1] = factor \* 45;

alpha[2] = factor \* 15;

beta[2] = factor \* 30;

Length[0] = 200;

Length[1] = 150;

Length[2] = 100;

x[0] = 0;

y[0] = 0;

z[0] = 0;

int i, j;

for (i = 1; i <= Segments\_Count; i++)

{

x[i] = x[i - 1] + System.Convert.ToInt32(Length[i - 1] \* Math.Cos(beta[i - 1]) \* Math.Sin(alpha[i - 1]));

y[i] = y[i - 1] + System.Convert.ToInt32(Length[i - 1] \* Math.Sin(beta[i - 1]));

z[i] = z[i - 1] + System.Convert.ToInt32(Length[i - 1] \* Math.Cos(beta[i - 1]) \* Math.Cos(alpha[i - 1]));

ArrayList MyList = new ArrayList();

MyList = GetClippedCone(Convert.ToInt32(Length[i - 1]), RBig[i - 1], RSmall[i - 1], m);

int MyListCount;

MyListCount = MyList.Count - 1;

for (j = 0; j <= MyListCount; j++)

{

int x1 = 0, y1 = 0, z1 = 0, newx = 0, newy = 0, newz;

int p = 0;

foreach (Vertex vert in MyList)

{

if (p == j)

{

x1 = Convert.ToInt32(vert.x);

y1 = Convert.ToInt32(vert.y);

z1 = Convert.ToInt32(vert.z);

}

p++;

}

newz = RotateAboutAlphaBeta(alpha[i - 1], beta[i - 1], x1, y1, z1, ref newx, ref newy);

int nx, ny, nz;

nx = newx + x[i - 1];

ny = newy + y[i - 1];

nz = newz + z[i - 1];

GList.Add(new Vertex(nx, ny, nz));

}

}

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int GListCount;

GListCount = GList.Count - 1;

int i;

Pen MyPen = new Pen(Color.Red, 1);

for (i = 0; i <= GListCount; i++)

{

int x = 0, y = 0, z = 0;

double newx = 0, newy = 0, newz;

int p = 0;

foreach (Vertex vert in GList)

{

if (p == i)

{

x = Convert.ToInt32(vert.x);

y = Convert.ToInt32(vert.y);

z = Convert.ToInt32(vert.z);

}

p++;

}

newz = RotateObject(Pitch, Yaw, Roll, x, y, z, ref newx, ref newy);

Rectangle MyBox = new Rectangle(Convert.ToInt32(newx), Convert.ToInt32(newy), 1, 1);

graphic.DrawEllipse(MyPen, MyBox);

}

int xstart, ystart;

xstart = Convert.ToInt32(Width / (double)2);

ystart = Convert.ToInt32(Height / (double)2);

Matrix MyMatrix = new Matrix();

MyMatrix.Translate(xstart, ystart);

graphic.Transform = MyMatrix;

}

}

}

**Example\_19.cs:**

using System;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace \_2\_laba\_Graphics

{

class Example\_19

{

public double RotateObject(double Pitch, double Yaw, double Roll, double x, double y, double z, ref double NewX, ref double NewY)

{

Double[,] m = new double[3, 3];

m[0, 0] = Math.Cos(Yaw) \* Math.Cos(Roll);

m[0, 1] = -Math.Cos(Yaw) \* Math.Sin(Roll);

m[0, 2] = -Math.Sin(Yaw);

m[1, 0] = Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Roll)

\* Math.Cos(Pitch);

m[1, 1] = -Math.Sin(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Cos(Roll)

\* Math.Cos(Pitch);

m[1, 2] = Math.Cos(Yaw);

m[2, 0] = -Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Cos(Roll) + Math.Sin(Pitch)

\* Math.Sin(Roll);

m[2, 1] = Math.Cos(Pitch) \* Math.Sin(Yaw) \* Math.Sin(Roll) + Math.Sin(Pitch)

\* Math.Cos(Roll);

m[2, 2] = Math.Cos(Yaw) \* Math.Cos(Pitch);

double NewZ;

NewX = m[0, 0] \* x + m[1, 0] \* y + m[2, 0] \* z;

NewY = m[0, 1] \* x + m[1, 1] \* y + m[2, 1] \* z;

NewZ = m[0, 2] \* x + m[1, 2] \* y + m[2, 2] \* z;

return NewZ;

}

public void DrawShape(Graphics graphic, double Pitch, double Yaw, double Roll, int Width, int Height)

{

int x0, y0, z0; // Координаты параллелпипеда

x0 = 200;

y0 = 200;

z0 = 0;

int a, b, c; // Длины сторон параллепипеда

a = 100;

b = 100;

c = 100;

int i, j, k, l;

int m; // Шаг итерации для параллепипеда

m = 7;

int x01, y01, z01, R1, m1; // Координаты круга

x01 = 250;

y01 = 250;

z01 = 50;

R1 = 50;

m1 = 1; // Шаг итерации для круга

int ZMin, Zmax;

ZMin = z01 - R1;

Zmax = z01 + R1;

Pen MyPen = new Pen(Color.Red);

Pen MyPen1 = new Pen(Color.Blue, 1);

Pen MyPen2 = new Pen(Color.Red, 1);

Pen MyPen3 = new Pen(Color.Green, 1);

Pen MyPen4 = new Pen(Color.Black, 1);

Pen MyPen5 = new Pen(Color.Yellow, 1);

Pen MyPen6 = new Pen(Color.Orchid, 1);

for (i = ZMin; i <= Zmax; i += m1)

{ // Сфера

int XMin, XMax;

XMin = Convert.ToInt32(x01 - Math.Sqrt(Math.Pow(R1, 2) - Math.Pow((i - z01), 2)));

XMax = Convert.ToInt32(x01 + Math.Sqrt(Math.Pow(R1, 2) - Math.Pow((i - z01), 2)));

int SmallR;

SmallR = Convert.ToInt32(Math.Sqrt(Math.Pow(R1, 2) - Math.Pow((i - z01), 2)));

for (j = XMin; j <= XMax; j += m1)

{

double y1, y2;

y1 = y01 + Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((j - x01), 2));

y2 = y01 - Math.Sqrt(Math.Pow(SmallR, 2) - Math.Pow((j - x01), 2));

double NewX1 = 0, NewY1 = 0, NewZ1;

double NewX2 = 0, NewY2 = 0, NewZ2;

NewZ1 = RotateObject(Pitch, Yaw, Roll, j, y1, i, ref NewX1, ref NewY1);

NewZ2 = RotateObject(Pitch, Yaw, Roll, j, y2, i, ref NewX2, ref NewY2);

Rectangle MyBox1 = new Rectangle(Convert.ToInt32(NewX1), Convert.ToInt32(NewY1), m1, m1);

Rectangle MyBox2 = new Rectangle(Convert.ToInt32(NewX2), Convert.ToInt32(NewY2), m1, m1);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

}

for (i = x0; i <= x0 + a; i += m)

{

for (j = y0; j <= y0 + b; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, i, j, z0, ref newx1, ref newy1);

Rectangle MyBox = new Rectangle(Convert.ToInt32(newx1), Convert.ToInt32(newy1), m, m);

graphic.DrawEllipse(MyPen5, MyBox);

}

}

for (i = z0 + m; i <= z0 + c - m; i += m)

{

for (j = y0; j <= y0 + b; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, x0, j, i, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, x0 + a, j, i, ref newx2, ref newy2);

Rectangle MyBox1 = new Rectangle(System.Convert.ToInt32(newx1), System.Convert.ToInt32(newy1), m, m);

Rectangle MyBox2 = new Rectangle(System.Convert.ToInt32(newx2), System.Convert.ToInt32(newy2), m, m);

graphic.DrawEllipse(MyPen1, MyBox1);

graphic.DrawEllipse(MyPen2, MyBox2);

}

for (k = x0; k <= x0 + a; k += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

double newx2 = 0, newy2 = 0, newz2 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, k, y0, i, ref newx1, ref newy1);

newz2 = RotateObject(Pitch, Yaw, Roll, k, y0 + b, i, ref newx2, ref newy2);

Rectangle MyBox3 = new Rectangle(System.Convert.ToInt32(newx1), System.Convert.ToInt32(newy1), m, m);

Rectangle MyBox4 = new Rectangle(System.Convert.ToInt32(newx2), System.Convert.ToInt32(newy2), m, m);

graphic.DrawEllipse(MyPen3, MyBox3);

graphic.DrawEllipse(MyPen4, MyBox4);

}

}

for (i = x0; i <= x0 + a; i += m)

{

for (j = y0; j <= y0 + b; j += m)

{

double newx1 = 0, newy1 = 0, newz1 = 0;

newz1 = RotateObject(Pitch, Yaw, Roll, i, j, z0 + c, ref newx1, ref newy1);

Rectangle MyBox = new Rectangle(Convert.ToInt32(newx1), Convert.ToInt32(newy1), m, m);

graphic.DrawEllipse(MyPen6, MyBox);

}

}

Matrix myMatrix = new Matrix();

myMatrix.Translate(Width / (float)2, Height / (float)2, MatrixOrder.Append);

graphic.Transform = myMatrix;

}

}

}

**NewSide.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2\_laba\_Graphics

{

public class NewSide

{

public Side x;

public double z;

public NewSide(Side x, double z)

{

this.x = x;

this.z = z;

}

}

}

**Side.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2\_laba\_Graphics

{

public class Side

{

public Vertex x1;

public Vertex x2;

public Vertex x3;

public Vertex x4;

public Side(Vertex x1, Vertex x2, Vertex x3, Vertex x4)

{

this.x1 = x1;

this.x2 = x2;

this.x3 = x3;

this.x4 = x4;

}

}

}

**Vertex.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2\_laba\_Graphics

{

public class Vertex

{

public double x;

public double y;

public double z;

public Vertex(double x, double y, double z)

{

this.x = x;

this.y = y;

this.z = z;

}

}

}

**VertexComparer.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2\_laba\_Graphics

{

public class VertexComparer : IComparer<NewSide>

{

public int Compare(NewSide o1, NewSide o2)

{

NewSide z1, z2;

try

{

z1 = (NewSide)o1;

z2 = (NewSide)o2;

}

catch (Exception e)

{

throw (e);

}

if (z1.z < z2.z) return 1; // В данном случае производится сортировка по убыванию, в противном(случае);

if (z1.z > z2.z) return -1; // В данном случае производится сортировка по убыванию, в(противном);

else return 0;

}

}

}

Приложение 2. Скриншот коммитов

Изображение выглядит как текст

Автоматически созданное описание

Рис. Скриншот коммитов

Приложение 3. Ссылка на GitHub

<https://github.com/MenikUG/2_laba_Graphics>