МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №3**

**з дисципліни «Комп’ютерна графіка»**

**на тему: «Тривимірна графіка»**

Виконав: студент гр. ПЗ1911

Сіньков Г.О.

Прийняла: ас. каф. КІТ

Нежуміра О. І.

Дніпро, 2020

**Лабораторна робота №3**

**Тема.** Тривимірна графіка.

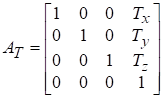
**Мета.** Вивчення методів формування 3D-зображень.

**Постановка задачі**

**Теоретичні відомості**

Афінні перетворення в просторі

Рівняння переміщення  записуються у вигляді матриці перетворень в однорідних координатах:   ,

де  – матриця переміщення.

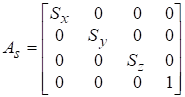
Рівняння масштабування відносно початку координат:

                                               (4.2)

де – масштабні множники.

У матричному вигляді рівняння масштабування записується як:

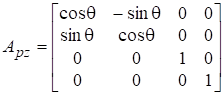
,                                               (4.3)

де  – матриця масштабування.

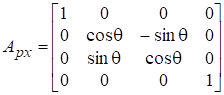
У матричному вигляді рівняння повороту записується як:

                                               (4.4)

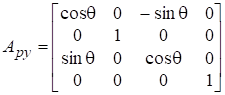
В тривимірному просторі поворот навколо осі  описує матриця повороту :

                                 (4.5)

У тривимірному просторі поворот навколо осі  описує матриця повороту :

                                 (4.6)

У тривимірному просторі поворот навколо осі  описує матриця повороту :

                                (4.7)

Якщо треба зробити одразу декілька перетворень, наприклад, два, то це можна зробити шляхом перемноження матриць  та  таким чином:

  тобто застосовується одна матриця, а потім множиться на , де .

Крива, що подаэться за допомогою В-сплайна, в загальному випадку може проходити через будь-які керуючі точки, вона є неперервною, і, крім того, неперервними є перша та друга похідні в кінцевих точках. В-сплайн описується такою формулою:

, , ,

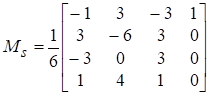
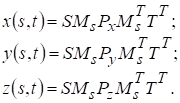


Рисунок 9. Кубічний В-сплайн, що побудовано за допомогою 9 точок

На рис. 4.9 зображено апроксімацію декількох точок за допомогою В-сплайнів.

Форми В-спланів і Без’є придатні до роботи в інтерактивному режимі, тому що їх геометричні вектори складаються лише з точок. Криву, що спочатку була задана в одній формі, можна перетворити в іншу форму, якщо записати геометричний вектор першої форми в термінах другої. Тому форму Ерміта, яка не має властивості випуклої оболонки, можна перетворити у форму Без’є, яка має таку властивість.

Куски у формі В-сплайнів подаються у вигляді

                                   (4.33)

При такому поданні поверхні досягається - неперервність. Матриця  складається з 16 керуючих точок і описує кусок, а також в загальному випадку точки, що не лежать на самому куску.

**Текст програми**

Form1.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;

namespace WindowsFormsLab3

{

public partial class Form1 : Form

{

Graphic\_3d curr;

Cylinder cyl = new Cylinder(200, 200, 40, 100, 150);

BiezSurface surf;

public Form1()

{

InitializeComponent();

listBox1.SelectedIndex = 0;

curr = cyl;

point3D[] tp = {

new point3D(100, 100, 0), new point3D(120, 100, 0), new point3D(140, 100, 0), new point3D(160, 100, 0),

new point3D(100, 150, 0), new point3D(120, 150, 0), new point3D(140, 150, 0), new point3D(160, 150, 0),

new point3D(100, 200, 0), new point3D(120, 200, 0), new point3D(140, 200, 0), new point3D(160, 200, 0),

new point3D(100, 250, 0), new point3D(120, 250, 0), new point3D(140, 250, 0), new point3D(160, 250, 0)

};

surf = new BiezSurface(tp);

comboBox1.SelectedIndex = 0;

}

private void pictureBox1\_Paint(object sender, PaintEventArgs e)

{

projType[] projs = { projType.front, projType.left, projType.above, projType.dimetry, projType.isometry };

curr.draw(e.Graphics, projs[listBox1.SelectedIndex]);

}

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

pictureBox1.Refresh();

}

private void button\_Rotate1\_Click(object sender, EventArgs e)

{

if (checkBox\_X.Checked == true)

curr.rotate(10, axe.ox);

if (checkBox\_Y.Checked == true)

curr.rotate(10, axe.oy);

if (checkBox\_Z.Checked == true)

curr.rotate(10, axe.oz);

pictureBox1.Refresh();

}

private void button\_Rotate2\_Click(object sender, EventArgs e)

{

if (checkBox\_X.Checked == true)

curr.rotate(-10, axe.ox);

if (checkBox\_Y.Checked == true)

curr.rotate(-10, axe.oy);

if (checkBox\_Z.Checked == true)

curr.rotate(-10, axe.oz);

pictureBox1.Refresh();

}

private void button\_Move1\_Click(object sender, EventArgs e)

{

if (checkBox\_X.Checked == true)

curr.move(10, axe.ox);

if (checkBox\_Y.Checked == true)

curr.move(10, axe.oy);

if (checkBox\_Z.Checked == true)

curr.move(10, axe.oz);

pictureBox1.Refresh();

}

private void button\_Move2\_Click(object sender, EventArgs e)

{

if (checkBox\_X.Checked == true)

curr.move(-10, axe.ox);

if (checkBox\_Y.Checked == true)

curr.move(-10, axe.oy);

if (checkBox\_Z.Checked == true)

curr.move(-10, axe.oz);

pictureBox1.Refresh();

}

private void button\_Scale1\_Click(object sender, EventArgs e)

{

curr.scale(checkBox\_X.Checked ? 1.5 : 1, checkBox\_Y.Checked ? 1.5 : 1, checkBox\_Z.Checked ? 1.5 : 1);

pictureBox1.Refresh();

}

private void button\_Scale2\_Click(object sender, EventArgs e)

{

curr.scale(checkBox\_X.Checked ? 1 / 1.5 : 1, checkBox\_Y.Checked ? 1 / 1.5 : 1, checkBox\_Z.Checked ? 1 / 1.5 : 1);

pictureBox1.Refresh();

}

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

{

if (comboBox1.SelectedIndex == 0)

curr = cyl;

else

curr = surf;

pictureBox1.Refresh();

panel1.Visible = comboBox1.SelectedIndex != 0;

}

private void comboBox\_N\_SelectedIndexChanged(object sender, EventArgs e)

{

point3D p = surf.points[int.Parse(comboBox\_N.Text) \* 4 + int.Parse(comboBox\_M.Text)];

textBox\_x.Text = p.x.ToString();

textBox\_y.Text = p.y.ToString();

textBox\_z.Text = p.z.ToString();

}

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

{

point3D p;

p.x = double.Parse(textBox\_x.Text);

p.y = double.Parse(textBox\_y.Text);

p.z = double.Parse(textBox\_z.Text);

surf.points[int.Parse(comboBox\_N.Text) \* 4 + int.Parse(comboBox\_M.Text)] = p;

pictureBox1.Refresh();

}

}

}

Graphic 3d.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Drawing;

namespace WindowsFormsLab3

{

enum projType { above, left, front, isometry, dimetry }

enum axe { ox, oy, oz }

struct point3D

{

public double x, y, z;

public point3D(double ax, double ay, double az)

{

x = ax;

y = ay;

z = az;

}

public Point project(projType proj)

{

switch (proj)

{

case projType.above: return new Point((int)x, (int)z);

case projType.left: return new Point((int)z, (int)y);

case projType.front: return new Point((int)x, (int)y);

case projType.dimetry: return new Point(200 + (int)(x \* 0.707 - 0.707 \* z), 200 + (int)(-0.408 \* x + 0.816 \* y - 0.408 \* z));

case projType.isometry: return new Point(100 + (int)(x \* 0.935 - 0.354 \* z), 100 + (int)(-0.118 \* x + 0.943 \* y - 0.312 \* z));

default: return new Point(0, 0);

}

}

}

class Graphic\_3d

{

public virtual void draw(Graphics g, projType proj) {}

public virtual void rotate(double angle, axe ax) {}

public virtual void move(double mov, axe ax) {}

public virtual void scale(double sx, double sy, double sz) {}

}

class Cylinder: Graphic\_3d

{

point3D[] points = new point3D[10];

public Cylinder(int cx, int cz, int rad1, int rad2, int h)

{

double ang = 0;

for (int i = 0; i < 4; i++, ang += 180 / 2)

{

double rad = Math.PI \* ang / 180;

points[i] = new point3D(cx + rad1 \* Math.Sin(rad), 0, cz + rad1 \* Math.Cos(rad));

points[i + 4] = new point3D(cx + rad2 \* Math.Sin(rad), h, cz + rad2 \* Math.Cos(rad));

}

}

public override void rotate(double angle, axe ax)

{

point3D cnt = center();

double rad = angle \* Math.PI / 180;

for (int i = 0; i < 10; i++) // 10 надо может быть будет поменять значение при помтроение

{

point3D tmp = points[i];

switch (ax)

{

case axe.ox:

points[i].y = (tmp.y - cnt.y) \* Math.Cos(rad) - (tmp.z - cnt.z) \* Math.Sin(rad) + cnt.y;

points[i].z = (tmp.y - cnt.y) \* Math.Sin(rad) + (tmp.z - cnt.z) \* Math.Cos(rad) + cnt.z;

break;

case axe.oy:

points[i].x = (tmp.x - cnt.x) \* Math.Cos(rad) + (tmp.z - cnt.z) \* Math.Sin(rad) + cnt.x;

points[i].z = -(tmp.x - cnt.x) \* Math.Sin(rad) + (tmp.z - cnt.z) \* Math.Cos(rad) + cnt.z;

break;

case axe.oz:

points[i].x = (tmp.x - cnt.x) \* Math.Cos(rad) - (tmp.y - cnt.y) \* Math.Sin(rad) + cnt.x;

points[i].y = (tmp.x - cnt.x) \* Math.Sin(rad) + (tmp.y - cnt.y) \* Math.Cos(rad) + cnt.y;

break;

}

}

}

public override void move(double mov, axe ax)

{

for (int i = 0; i < 10; i++)

{

switch (ax)

{

case axe.ox:

points[i].x = points[i].x + mov;

break;

case axe.oy:

points[i].y = points[i].y + mov;

break;

case axe.oz:

points[i].z = points[i].z + mov;

break;

}

}

}

public override void scale(double sx, double sy, double sz)

{

point3D cnt = center();

for (int i = 0; i < 8; i++)

{

points[i].x = (points[i].x - cnt.x) \* sx + cnt.x;

points[i].y = (points[i].y - cnt.y) \* sy + cnt.y;

points[i].z = (points[i].z - cnt.z) \* sz + cnt.z;

}

}

private point3D center()

{

point3D c = new point3D(0, 0, 0);

for (int i = 0; i < 10; i++)

{

c.x += points[i].x;

c.y += points[i].y;

c.z += points[i].z;

}

c.x /= 10;

c.y /= 10;

c.z /= 10;

return c;

}

public override void draw(Graphics g, projType proj)

{

point3D cntr = new point3D(100, 100, 0);

point3D cntrX = new point3D(200, 100, 0);

point3D cntrY = new point3D(100, 0, 0);

point3D cntrZ = new point3D(100, 100, 100);

for (int i = 0; i < 4; i++)

{

g.DrawLine(Pens.Black, points[i].project(proj), points[(i + 1) % 4].project(proj));// top

g.DrawLine(Pens.Black, points[i + 4].project(proj), points[(i + 1) % 4 + 4].project(proj)); //bottom

g.DrawLine(Pens.Black, points[i].project(proj), points[i + 4].project(proj));//edges

}

g.DrawLine(Pens.Red, cntr.project(proj), cntrX.project(proj));//Ox

g.DrawString("Ox", SystemFonts.DefaultFont, Brushes.Black, cntrX.project(proj));

g.DrawLine(Pens.Blue, cntr.project(proj), cntrY.project(proj));//Oy

g.DrawString("Oy", SystemFonts.DefaultFont, Brushes.Black, cntrY.project(proj));

g.DrawLine(Pens.Green, cntr.project(proj), cntrZ.project(proj));//Oz

g.DrawString("Oz", SystemFonts.DefaultFont, Brushes.Black, cntrZ.project(proj));

}

}

class Matrix

{

public double[,] elem = new double[4, 4];

int n, m;

public Matrix(int an, int am)

{

n = an;

m = am;

for (int i = 0; i < 4; i++)

for (int j = 0; j < 4; j++)

elem[i, j] = 0;

}

public static Matrix operator \* (Matrix m1, Matrix m2)

{

Matrix res = new Matrix(m1.n, m2.m);

for (int i = 0; i < res.m; i++) // colomns

{

for (int j = 0; j < res.n; j++) // rows

{

double s = 0;

for (int k = 0; k < m1.m; k++)

s += m1.elem[j, k] \* m2.elem[k, i];

res.elem[j, i] = s;

}

}

return res;

}

}

class BiezSurface: Graphic\_3d

{

public point3D[] points = new point3D[16];

Matrix Mb = new Matrix(4, 4);

Matrix MbT = new Matrix(4, 4);

public BiezSurface(point3D[] apoints)

{

points = apoints;

double[,] mb =

{

{ -1, 3, -3, 1 },

{ 3, -6, 3, 0 },

{ -3, 3, 0, 0 },

{ 1, 0, 0, 0 }

};

Mb.elem = mb;

for (int i = 0; i < 4; i++)

for (int j = 0; j < 4; j++)

MbT.elem[j, i] = Mb.elem[i, j];

}

public override void rotate(double angle, axe ax)

{

point3D cnt = center();

double rad = angle \* Math.PI / 180;

for (int i = 0; i < 16; i++)

{

point3D tmp = points[i];

switch (ax)

{

case axe.ox:

points[i].y = (tmp.y - cnt.y) \* Math.Cos(rad) - (tmp.z - cnt.z) \* Math.Sin(rad) + cnt.y;

points[i].z = (tmp.y - cnt.y) \* Math.Sin(rad) + (tmp.z - cnt.z) \* Math.Cos(rad) + cnt.z;

break;

case axe.oy:

points[i].x = (tmp.x - cnt.x) \* Math.Cos(rad) + (tmp.z - cnt.z) \* Math.Sin(rad) + cnt.x;

points[i].z = -(tmp.x - cnt.x) \* Math.Sin(rad) + (tmp.z - cnt.z) \* Math.Cos(rad) + cnt.z;

break;

case axe.oz:

points[i].x = (tmp.x - cnt.x) \* Math.Cos(rad) - (tmp.y - cnt.y) \* Math.Sin(rad) + cnt.x;

points[i].y = (tmp.x - cnt.x) \* Math.Sin(rad) + (tmp.y - cnt.y) \* Math.Cos(rad) + cnt.y;

break;

}

}

}

public override void move(double mov, axe ax)

{

for (int i = 0; i < 16; i++)

{

switch (ax)

{

case axe.ox:

points[i].x = points[i].x + mov;

break;

case axe.oy:

points[i].y = points[i].y + mov;

break;

case axe.oz:

points[i].z = points[i].z + mov;

break;

}

}

}

public override void scale(double sx, double sy, double sz)

{

point3D cnt = center();

for (int i = 0; i < 16; i++)

{

points[i].x = (points[i].x - cnt.x) \* sx + cnt.x;

points[i].y = (points[i].y - cnt.y) \* sy + cnt.y;

points[i].z = (points[i].z - cnt.z) \* sz + cnt.z;

}

}

private point3D center()

{

point3D c = new point3D(0, 0, 0);

for (int i = 0; i < 16; i++)

{

c.x += points[i].x;

c.y += points[i].y;

c.z += points[i].z;

}

c.x /= 16;

c.y /= 16;

c.z /= 16;

return c;

}

public override void draw(Graphics g, projType proj)

{

Matrix S = new Matrix(1, 4);

Matrix T = new Matrix(4, 1);

Matrix Px = new Matrix(4, 4);

Matrix Py = new Matrix(4, 4);

Matrix Pz = new Matrix(4, 4);

for (int i = 0; i < 16; i++)

{

Px.elem[i / 4, i % 4] = points[i].x;

Py.elem[i / 4, i % 4] = points[i].y;

Pz.elem[i / 4, i % 4] = points[i].z;

}

double stp = 0.03; // 0.05

for (double t = 0; t <= 1; t += stp)

{

T.elem[0, 0] = t \* t \* t;

T.elem[1, 0] = t \* t;

T.elem[2, 0] = t;

T.elem[3, 0] = 1;

for (double s = 0; s <= 1; s += stp)

{

S.elem[0, 0] = s \* s \* s;

S.elem[0, 1] = s \* s;

S.elem[0, 2] = s;

S.elem[0, 3] = 1;

point3D p;

p.x = (S \* Mb \* Px \* MbT \* T).elem[0, 0];

p.y = (S \* Mb \* Py \* MbT \* T).elem[0, 0];

p.z = (S \* Mb \* Pz \* MbT \* T).elem[0, 0];

Point pt = p.project(proj);

g.FillEllipse(Brushes.Black, pt.X, pt.Y, 4, 4);

}

}

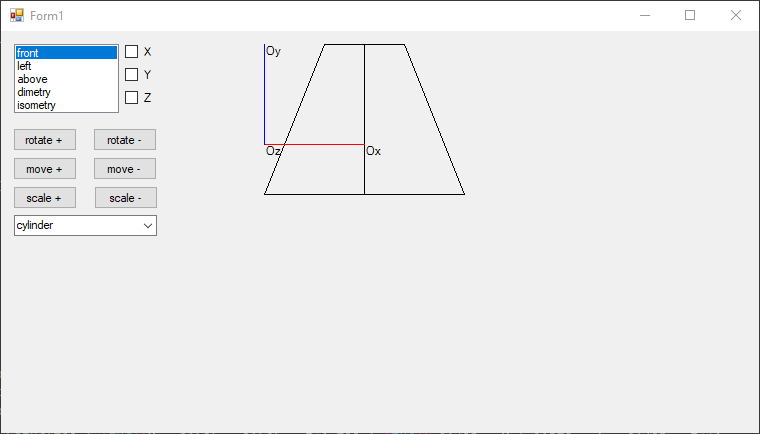
}

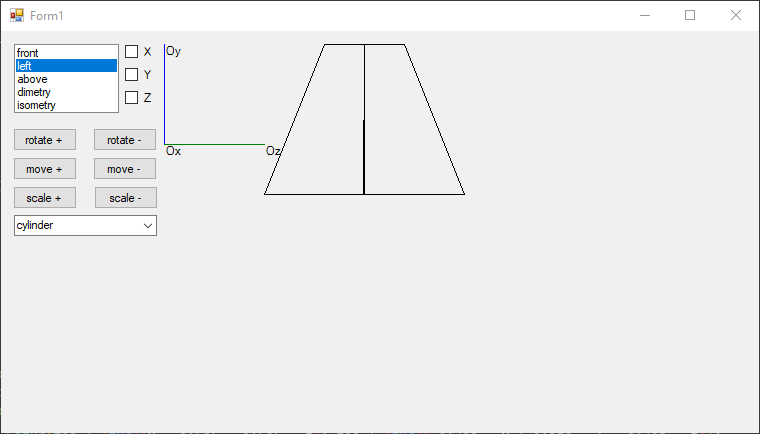
}

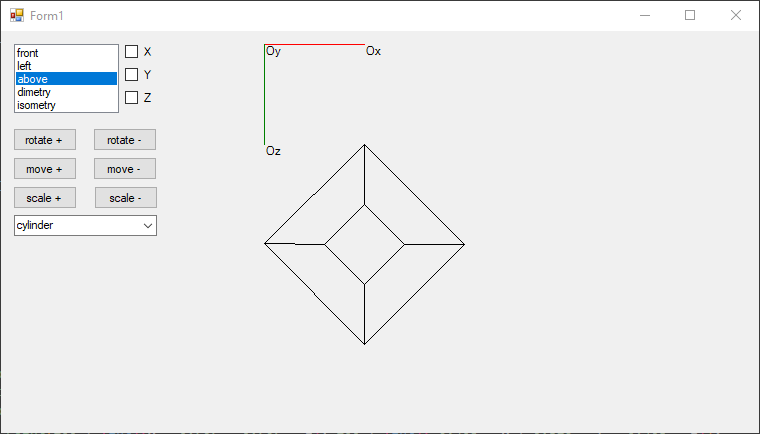
}

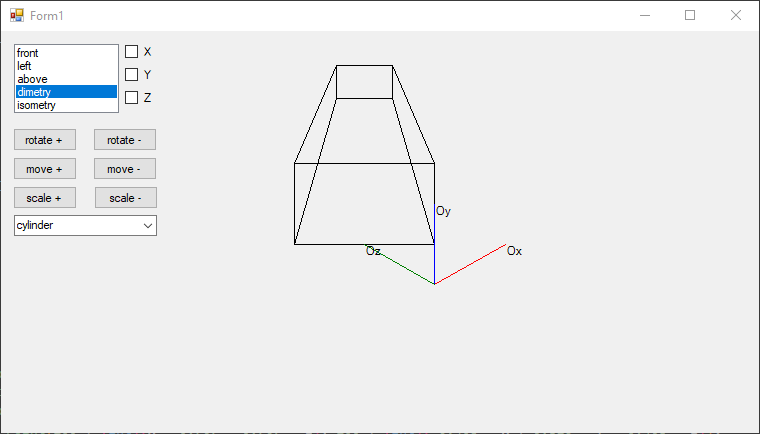
**Демонстрація роботи**

Рис. 1-5 – Різна проекція фігури









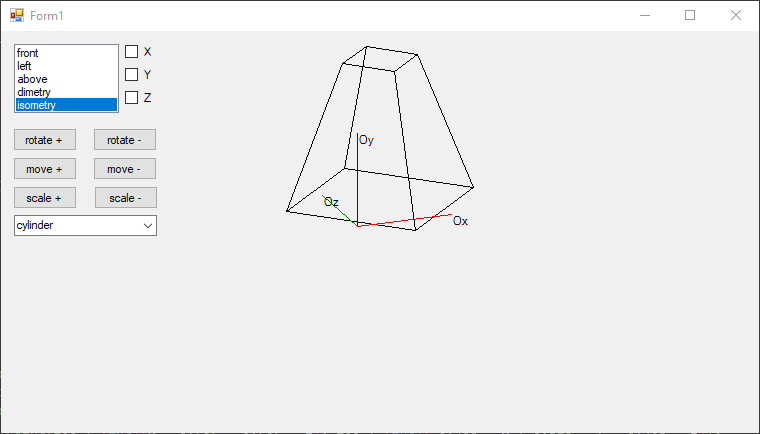
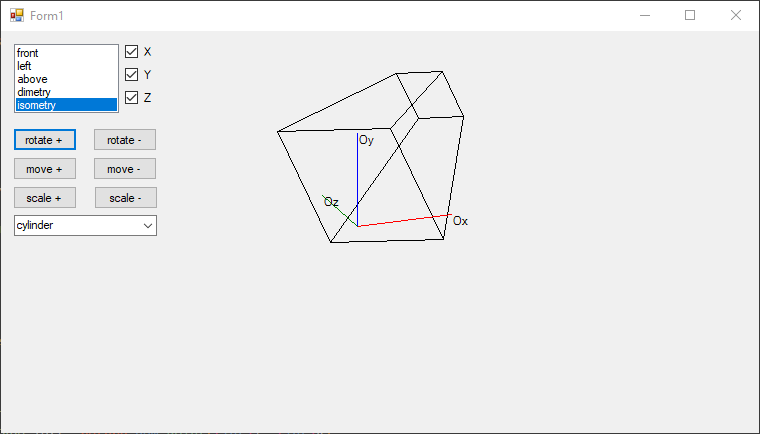


Рис. 6-7 – Повертання фігури по 3 осям



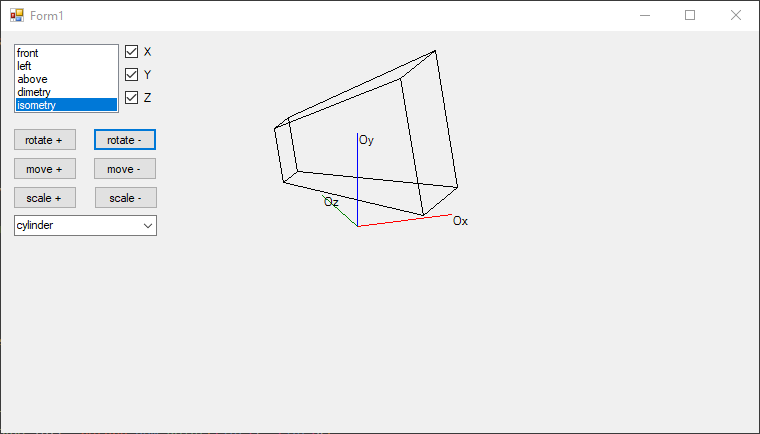
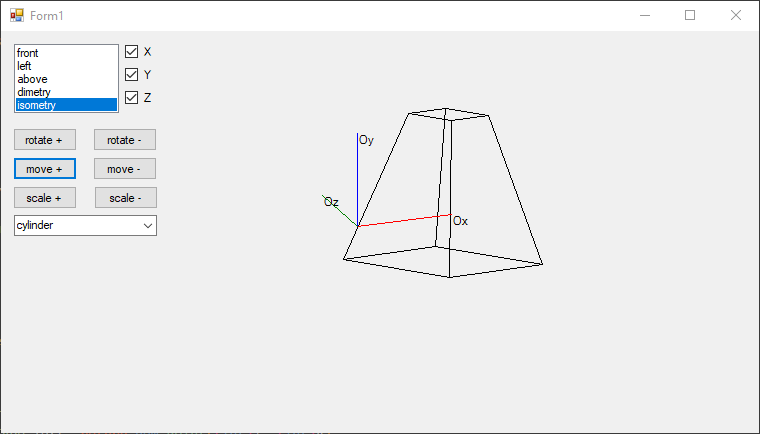


Рис. 8-9 – Переміщення фігури по 3 осям



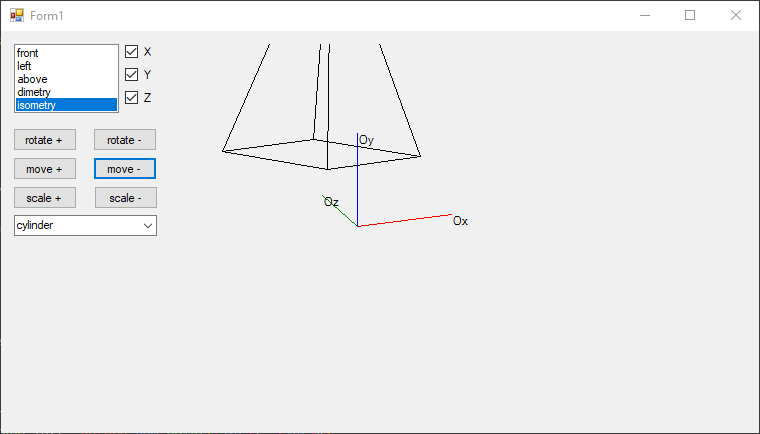
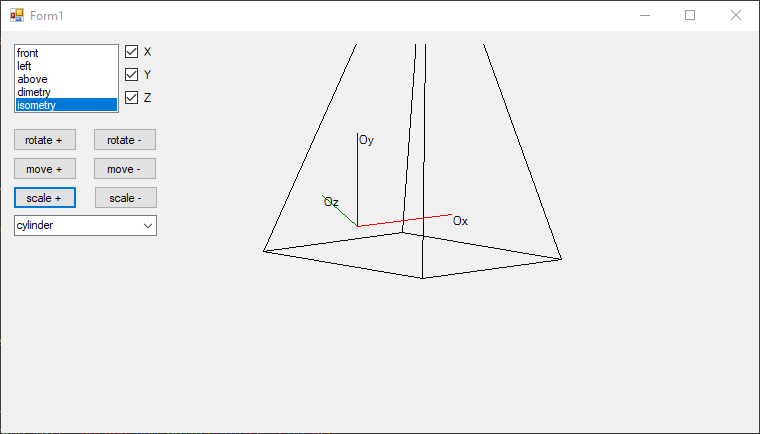


Рис. 10-11 – Масштабування фігури по 3 осям



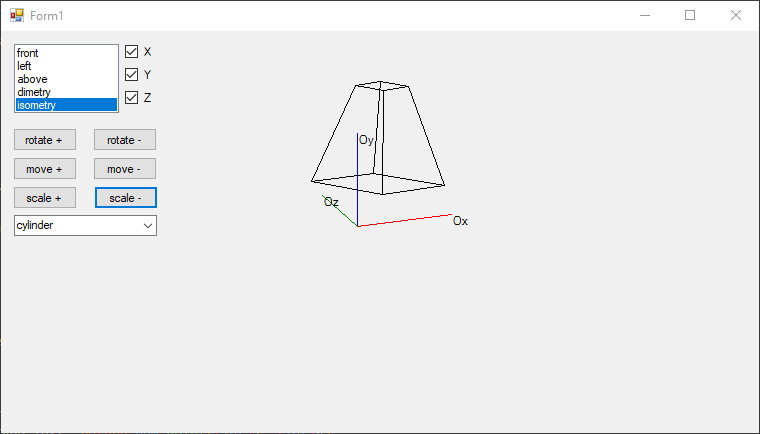
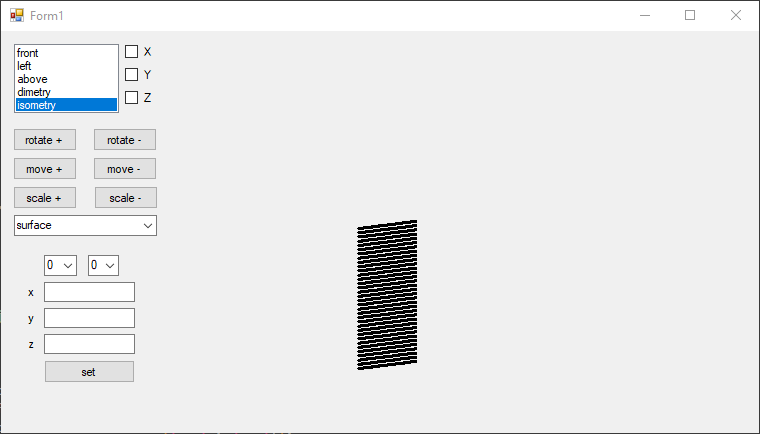


Рис. 12-13 – поверхня Без’є



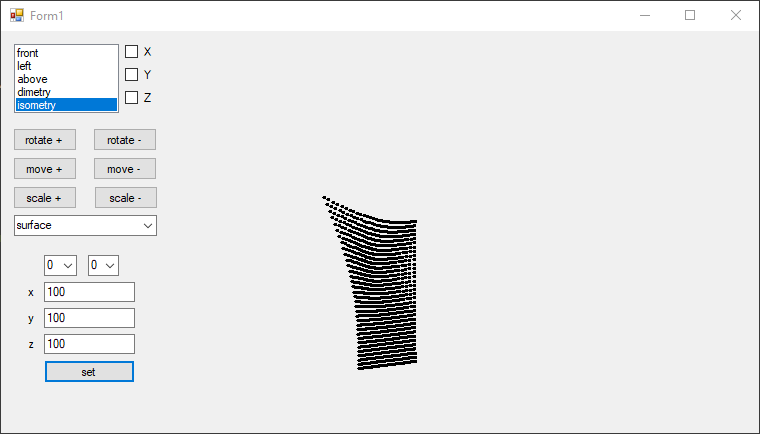
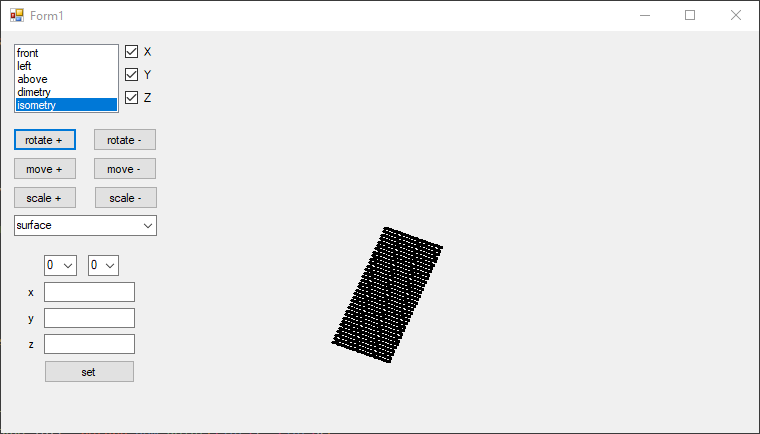


Рис. 14-15 – Повертання площини по 3 осям



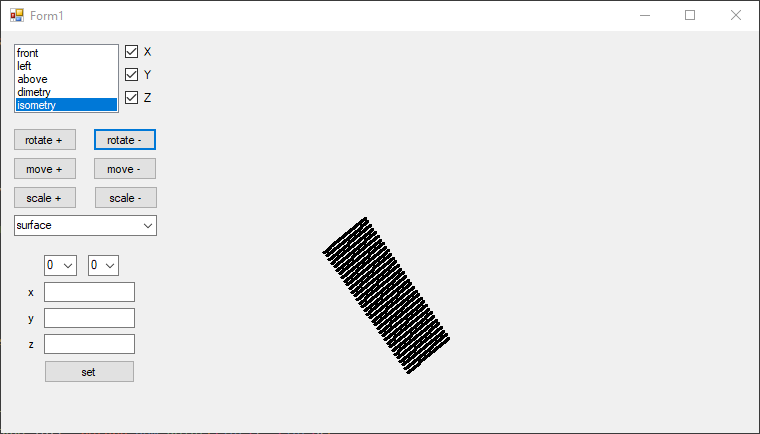
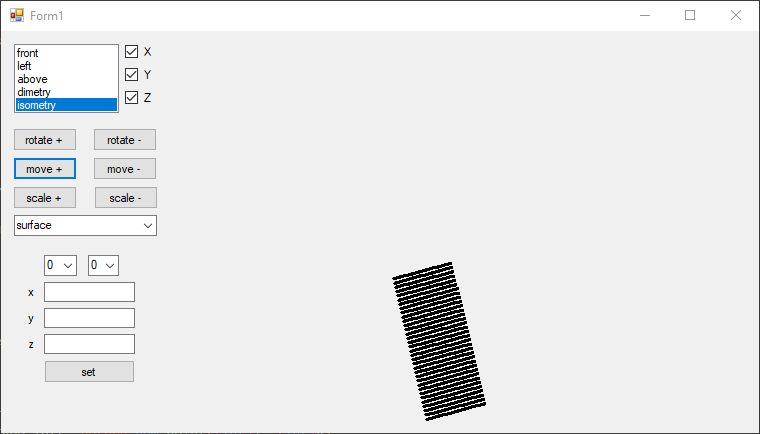


Рис. 16-17 – Переміщення площини по 3 осям



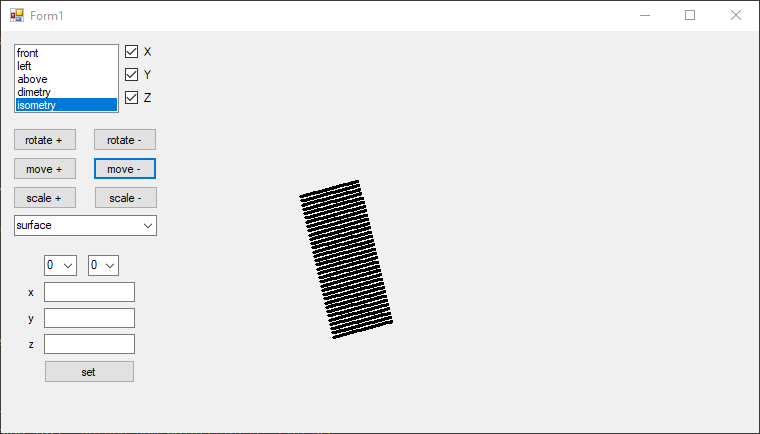
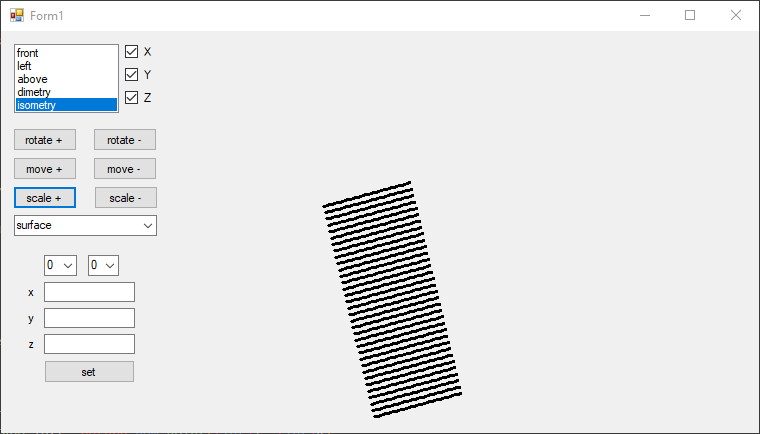
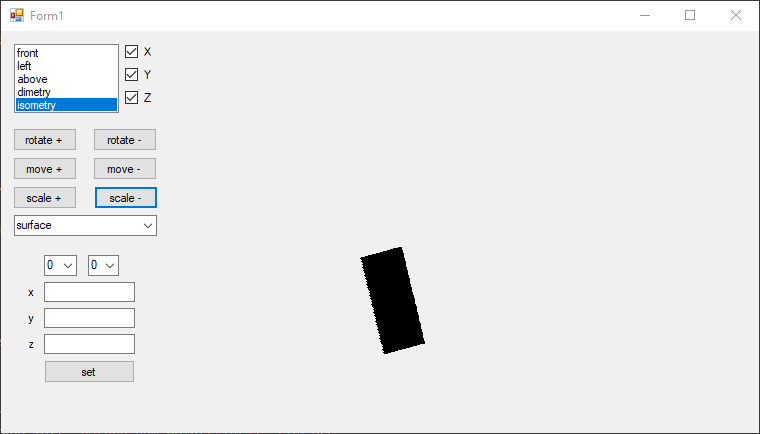


Рис. 18-19 – Масштабування площини по 3 осям





**Висновок**: В ході лабораторної роботи ми знайомились з формування 3D-зображень. Для розробки програми, яка б відповідала поставленим вимогам (афінні перетворення кожної фігури, поверхні; зображення фігури при заданому виді проектування (ізометрія чи диметрія); зображення проекцій фігур на площини XOY, XOZ, YOZ; зображення вісей з підписами), ми ознайомилися з прокцінюванням, видами та способами проеціювання. Також розібралися за афінними перетвореннями у просторі та їх особливостями у реалізації. Окремим пунктом була розробка поверхні: однієї з 3 параметричних кубічних поверхнось та операціями над ними. У нашому ж варіанті ми розібрали форму B-сплайнів, яка є дуже схожою з формою Без’є у своїх реалізації.