МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА САПР

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Компьютерная графика»

Тема: «Формирования различных поверхностей с использованием ее ортогонального проектирования на плоскость при ее визуализации»

Вариант 3

Студенты гр. 9309 _____ Аль Сайед А.З.

_____ Серов А.В.

_____ Юшин Е.В.

Преподаватель _____ Матвеева И.В.

Санкт-Петербург

2022

Цель работы

Научиться формировать билинейную поверхность.

Задание

Сформировать поверхность Безье для различного задающего многогранника. Обеспечить поворот сформированной поверхности вокруг осей X и Y.

Теоретические положения

Поверхность Безье определяется двумерным набором контр. точек $p_{i,j}$, где i в пределах от 0 до m, а j в пределах от 0 до n. Таким образом, в этом случае, имеем m+1 рядов и n+1 столбцов контр. точек и контр. точка в i-м ряду и j-м столбце обозначается $p_{i,j}$. В итоге получается (m+1)*(n+1) контр. Точек.

Вот уравнение поверхности Безье, определяемой m+1 рядами и n+1 столбцами контр. точек:

$$\mathbf{p}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_{m,i}(u) B_{n,j}(v) \mathbf{p}_{ij}$$

где $B_{m,i}(u)$ и $B_{n,j}(v)$ - это i-ая и j-я базисные функции Безье в направлениях u и v, соответственно. Как вы помните из обсуждения кривых Безье, эти базисные функции определяются так:

$$B_{m,i}(u) = \frac{m!}{i!(m-i)!} u^i (1-u)^{m-i}$$

$$B_{n,j}(v) = \frac{n!}{j!(n-j)!} v^j (1-v)^{n-j}$$

Для специального случая бикубической поверхности Безье размера 4 x 4 уравнение имеет вид:

$$Q(u,w)=\begin{bmatrix}u^3 & u^2 & u & 1\end{bmatrix}\begin{bmatrix}-1 & 3 & -3 & 1\\ 3 & -6 & 3 & 0\\ -3 & 3 & 0 & 0\\ 1 & 0 & 0 & 0\end{bmatrix}\times$$

$$\times\begin{bmatrix}B_{0,0} & B_{0,1} & B_{0,2} & B_{0,3}\\ B_{1,0} & B_{1,1} & B_{1,2} & B_{1,3}\\ B_{2,0} & B_{2,1} & B_{2,2} & B_{2,3}\\ B_{3,0} & B_{3,1} & B_{3,2} & B_{3,3}\end{bmatrix}\begin{bmatrix}-1 & 3 & -3 & 1\\ 3 & -6 & 3 & 0\\ -3 & 3 & 0 & 0\\ 1 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}w^3\\ w^2\\ w\\ 1\end{bmatrix}.$$

Наиболее часто используются бикубические поверхности Безье (n=m=3), задающиеся шестнадцатью контрольными точками.

Примеры работы программы

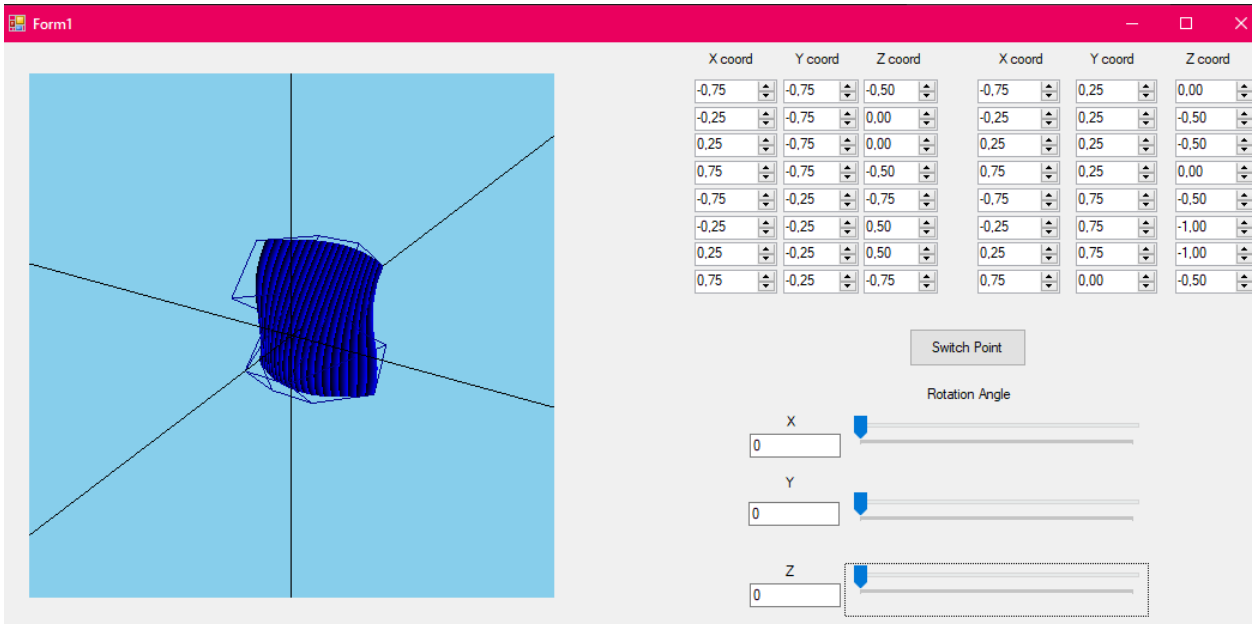

Рис. 1.

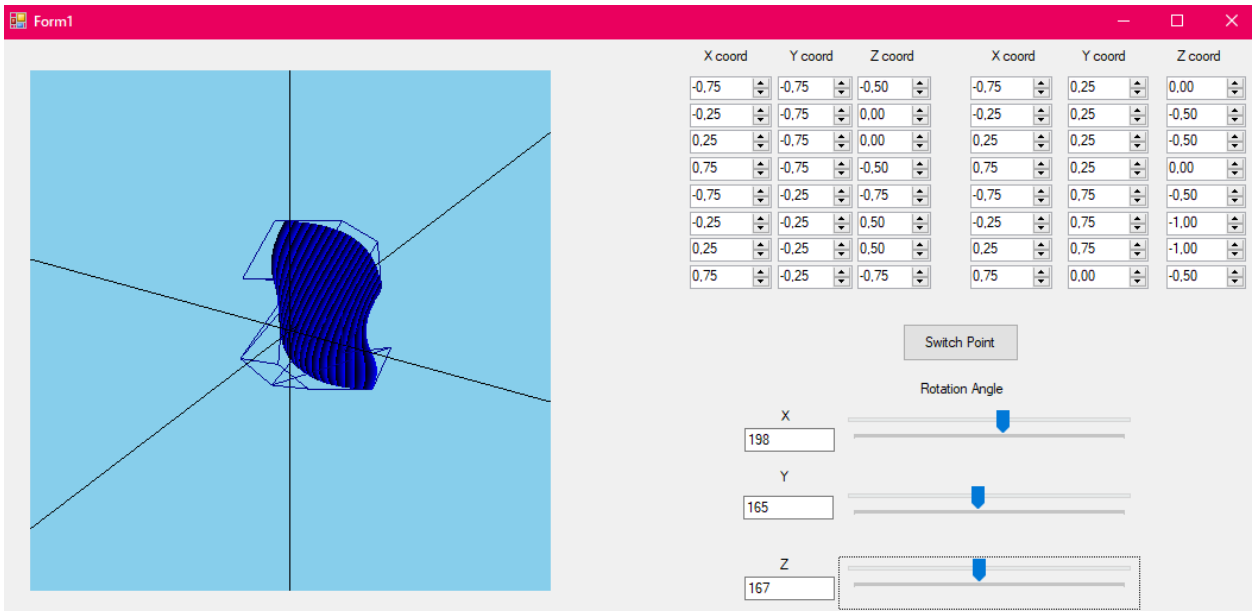

Рис. 2.

Код программы

BezierCurvesCustom.cs

```
using System;
using System.Windows.Media.Media3D;

namespace CG_lab3
{
    public class BezierCurvesCustom
    {
        public Point3D[] initialPoints;
        Point3D[] toDrawPoints;
        float MaxStepDraw = 100f;

        public BezierCurvesCustom(Point3D[] curvePoints)
        {
            toDrawPoints = new Point3D[curvePoints.Length];
            initialPoints = curvePoints;

            initialPoints.CopyTo(toDrawPoints, 0);

            //2-3
            if (curvePoints.Length > 4) toDrawPoints[3] =
FindLineMiddle(toDrawPoints[2], toDrawPoints[3]);
            //5-6
            if (curvePoints.Length > 7) toDrawPoints[6] =
FindLineMiddle(toDrawPoints[5], toDrawPoints[6]);

        }

        public Point3D[] PointsToDraw()
        {
            Point3D[] points = new Point3D[(int)(MaxStepDraw +
1)*(toDrawPoints.Length /3)];
            int j = 0;

            for (int i = 0; i < toDrawPoints.Length - 1; i += 3)
            {
                for (float t = 0; t <= 1; t += 1.0f / MaxStepDraw)
                {
                    points[j] = PointOnCurve(t, toDrawPoints[i], toDrawPoints[i+1],
toDrawPoints[i+2], toDrawPoints[i+3]);
                    j++;
                }
            }

            return points;
        }

        Point3D PointOnCurve(double t, Point3D p1, Point3D p2, Point3D p3, Point3D
p4)
        {
            double var1 = 1 - t;
            Point3D vPoint = new Point3D(0, 0, 0);

            vPoint.X = Math.Pow(var1, 3) * p1.X + 3 * t * Math.Pow(var1, 2) * p2.X +
3 * Math.Pow(t, 2) * var1 * p3.X + Math.Pow(t, 3) * p4.X;
            vPoint.Y = Math.Pow(var1, 3) * p1.Y + 3 * t * Math.Pow(var1, 2) * p2.Y +
3 * Math.Pow(t, 2) * var1 * p3.Y + Math.Pow(t, 3) * p4.Y;
            vPoint.Z = Math.Pow(var1, 3) * p1.Z + 3 * t * Math.Pow(var1, 2) * p2.Z +
3 * Math.Pow(t, 2) * var1 * p3.Z + Math.Pow(t, 3) * p4.Z;

            return (vPoint);
        }
```

```
        public Point3D PointOnCurve4(double t)
        {
            return PointOnCurve(t, toDrawPoints[0], toDrawPoints[1],
toDrawPoints[2], toDrawPoints[3]);
        }

        Point3D FindLineMiddle(Point3D start, Point3D end) => new Point3D((start.X +
end.X) / 2, (start.Y + end.Y) / 2, (start.Z + end.Z) / 2);
    }
}
```

Rotate.cs

```
using System;
using System.Windows.Media.Media3D;

namespace CG_lab3
{
    public enum Axes
    {
        X, Y, Z
    }

    public class Rotation
    {

        public Point3D RotateArounAxis(Point3D point, Axes axis, int angleDegrees)
        {
            double radAngle = angleDegrees * Math.PI / 180;

            double[] coordsToChange = new double[2];
            switch (axis)
            {
                case Axes.X:
                    coordsToChange[0] = point.Y;
                    coordsToChange[1] = point.Z;
                    break;
                case Axes.Y:
                    coordsToChange[0] = point.X;
                    coordsToChange[1] = point.Z;
                    break;
                case Axes.Z:
                    coordsToChange[0] = point.X;
                    coordsToChange[1] = point.Y;
                    break;
            }

            double[,] muliplMatrix = new double[2, 2]
            {
                {Math.Cos(radAngle), Math.Sin(radAngle)},
                {-Math.Sin(radAngle), Math.Cos(radAngle) }
            };

            coordsToChange = Multiply(coordsToChange, muliplMatrix);

            switch (axis)
            {
                case Axes.X:
                    point.Y = coordsToChange[0];
                    point.Z = coordsToChange[1];
                    break;
                case Axes.Y:
```

```
                    point.X = coordsToChange[0];
                    point.Z = coordsToChange[1];
                    break;
                case Axes.Z:
                    point.X = coordsToChange[0];
                    point.Y = coordsToChange[1];
                    break;
            }

            return point;
        }

        private double[] Multiply(double[] first, double[,] second)
        {
            double[] answer = new double[2] { 0, 0 };

            for (int i = 0; i < answer.Length; i++)
            {
                for (int j = 0; j < first.Length; j++)
                {
                    answer[i] += first[j] * second[j, i];
                }
            }

            return answer;
        }
    }
}
```

Вывод

В ходе выполнения данной лабораторной работы мы исследовали формирование поверхности Безье на основе заданного многогранника, определяемого введенными точками