Московский авиационный институт
(Национальный исследовательский университет)

Институт: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»
Дисциплина: «Компьютерная графика»

Лабораторная работа № 1

Тема: Построение изображений 2D- кривых

Студент: Пивницкий Даниэль Сергеевич
Группа: 80-306
Преподаватель: Чернышов Л.Н.
Дата:
Оценка:

Москва, 2021

1. **Постановка задачи**

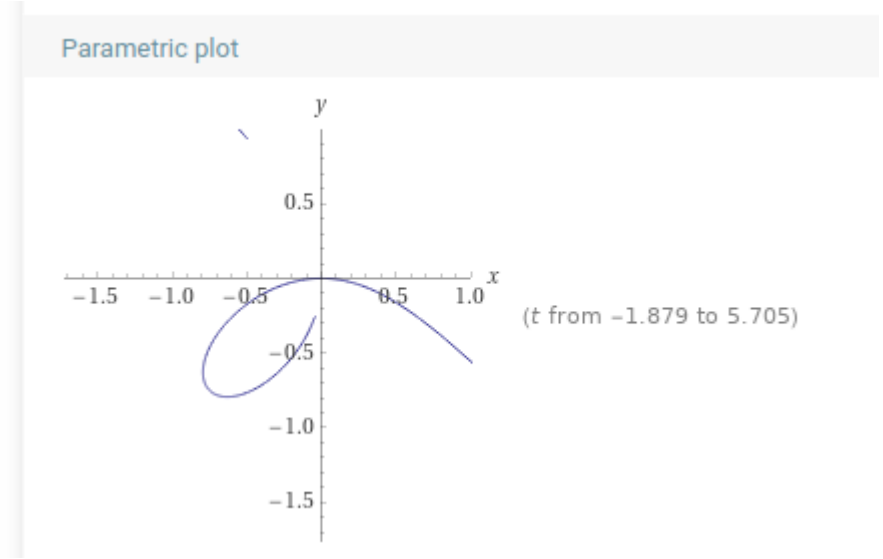
Написать и отладить программу, строящую изображение заданной замечательной кривой.
Вариант: 18
x = 3at/(1+t^3), y = 3at^2/(1+^3), -1<A<=t<= B

2. **Описание программы**

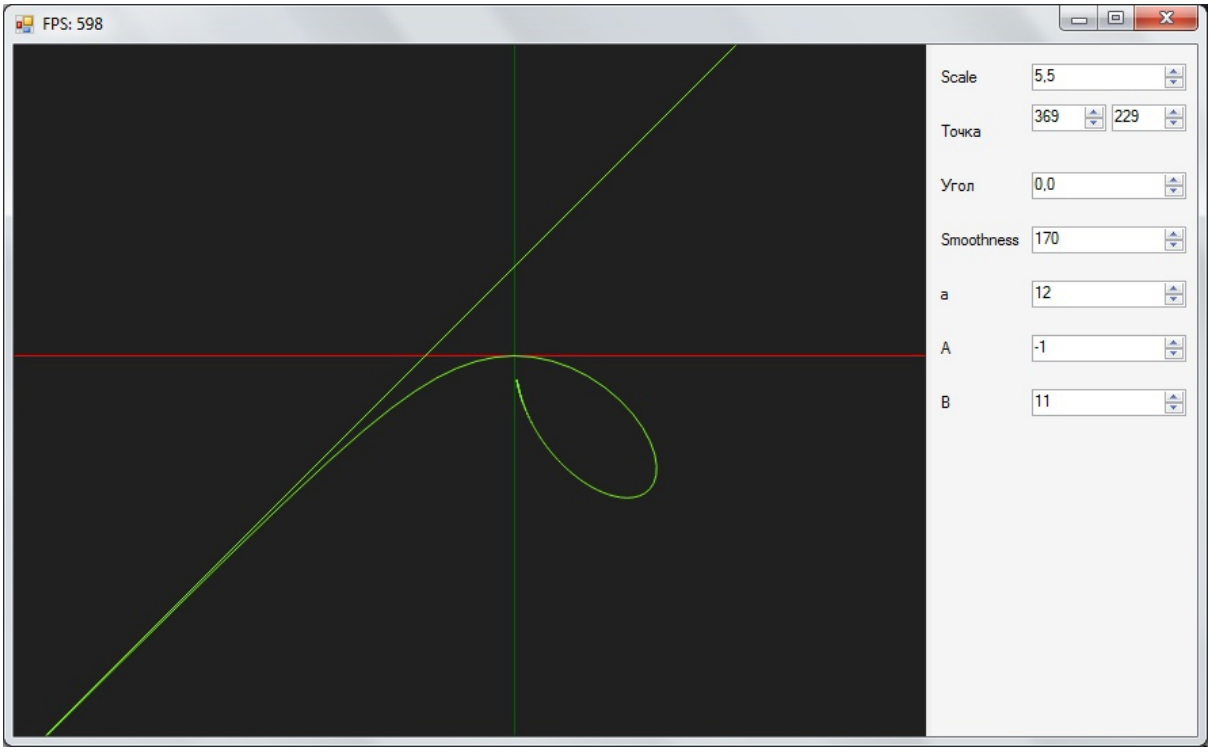
Дана функция, зависящая от нескольких параметров. В демонстрационной программе есть код отрисовки отрезка между двумя точками, и код управления значением переменной. Эти два элемента были скопированы в лабораторную работу и переделаны для выполнения задачи. Для каждой переменной заданной функции создан элемент контроля значением. Учитываются ограничения на эти значения. График функции разбит на конечное количество равных отрезков (равных по переменной ФИ, не по длине). Все эти отрезки отрисовываются в цикле. Операции переноса, вращения и масштабирования производятся перед отрисовкой. Создаваемое программой окно разбито на 2 части: одна отвечает за отрисовку графика функции, другая - за контроль переменных.

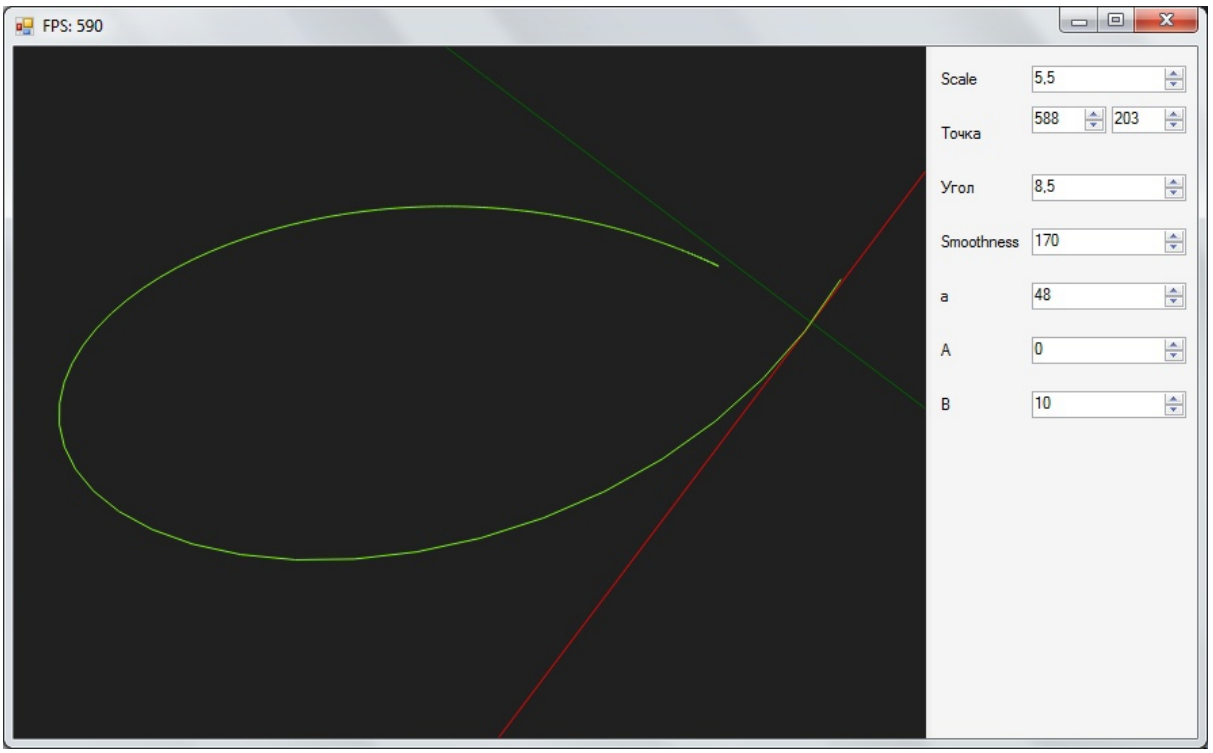
3. Набор тестов

Переменные функции.
Проверив формулу через Wolframalpha, мы получили график для сверки выполненных тестов



4. Результаты выполнения тестов





5. Листинг программы

```
using System;
using System.Linq;
using System.Drawing;
using System.Drawing.Imaging;
using System.Windows.Forms;
using System.ComponentModel;
using System.Collections.Generic;
using CGLabPlatform;

public abstract class CGLab01 : GFXApplicationTemplate<CGLab01>
{
    #region Инициализация

        [STAThread] static void Main() { RunApplication(); }

     protected override void OnMainWindowLoad(object
```

```
sender, EventArgs args)
    {
        base.RenderDevice.BufferBackCol = 0x20;
           base.RenderDevice.MouseMoveWithLeftBtnDown
+= (s, e) =>
                                    FirstPoint  +=  new
DVector2(e.MovDeltaX, e.MovDeltaY);

         RenderDevice.HotkeyRegister(Keys.PageUp, (s,
e) => ++Length);
           RenderDevice.HotkeyRegister(Keys.PageDown,
(s, e) => --Length);

        MainWindow.Shown += (s, e) =>
        {

        };
    }

    DVector2 func(double t, double a, double anglo)
    {
        double x = 3*a*t/(1+t*t*t);
        double y = x * t;
        x = x * Length;
        y = y * Length;
           double  rx  =  -  x  *  Math.Cos(anglo)  -  y  *
Math.Sin(anglo);
            double  ry  =  y  *  Math.Cos(anglo)  +  x  *
Math.Sin(anglo);
        DVector2 rez = new DVector2(rx, ry);
        return rez;
    }

    #endregion
```

```
    #region Свойства

      [DisplayNumericProperty(Default: 1, Increment:
0.1, Name: "Scale", Minimum: 0.1)]
    public double Length
    {
        get { return Get<double>(); }
        set { Set<double>(value); }
    }

    [DisplayNumericProperty(
        Default: new[] { 300d, 200d },
        Increment: 1,
        Name: "Òî÷êà",
        Minimum: -1000
    )]
     public abstract DVector2 FirstPoint { get; set;
}

      [DisplayNumericProperty(Default: 0, Increment:
0.1, Name: "Óãîë")]
    public double Angle
    {
        get { return Get<double>(); }
        set
        {
            while (value < 0) value += 360;
            while (value >= 360) value -= 360;
            Set<double>(value);
        }
    }

     [DisplayNumericProperty(Default: 10, Increment:
2, Name: "Smoothness")]
    public double kn
```

```
    {
        get { return Get<double>(); }
        set
        {
            if (value < 1)
            {
                Set<double>(1);
            }
            else
            {
                Set<double>(value);
            }
        }
    }

      [DisplayNumericProperty(Default: 10, Increment:
1, Name: "a")]
    public double ka
    {
        get { return Get<double>(); }
        set { Set<double>(value); }
    }

      [DisplayNumericProperty(Default: 0, Increment:
1, Name: "A")]
    public double kaa
    {
        get { return Get<double>(); }
        set
        {
            if (value <= -1)
            {
                Set<double>(-1.000001);
            }
            else
```

```
            {
                Set<double>(value);
            }
        }
    }

     [DisplayNumericProperty(Default: 15, Increment:
1, Name: "B")]
    public double kbb
    {
        get { return Get<double>(); }
        set
        {
            if (value < kaa)
            {
                Set<double>(kaa);
            }
            else
            {
                Set<double>(value);
            }
        }
    }

    #endregion



    protected override void OnDeviceUpdate(object s,
GDIDeviceUpdateArgs e)
    {//do stuff here
        DVector2 SecondPoint, PastPoint;
        SecondPoint.X = Math.Cos(Angle) * 500;
        SecondPoint.Y = Math.Sin(Angle) * 500;
               e.Surface.DrawLine(Color.Red.ToArgb(),
```

```
FirstPoint + SecondPoint, FirstPoint - SecondPoint);
        SecondPoint.X = -Math.Sin(Angle) * 500;
        SecondPoint.Y = Math.Cos(Angle) * 500;
         e.Surface.DrawLine(Color.DarkGreen.ToArgb(),
FirstPoint + SecondPoint, FirstPoint - SecondPoint);
        double inc = (kbb - kaa) / kn;
        for (double ii = kaa; ii <= kbb; ii += inc)
        {
             PastPoint = FirstPoint + func(ii - inc,
ka, Angle);
             SecondPoint = FirstPoint + func(ii, ka,
Angle);

e.Surface.DrawLine(Color.LawnGreen.ToArgb(),
PastPoint, SecondPoint);
        }
    }


        private void AddControll(Control ctrl, int
heigh)
    {
                                    var    layout    =
ValueStorage.Controls[0].Controls[0]              as
TableLayoutPanel;

        layout.SuspendLayout();
        ctrl.Dock = DockStyle.Fill;
        layout.Parent.Height += heigh;
         layout.RowStyles.Insert(layout.RowCount - 1,
new RowStyle(SizeType.Absolute, heigh));
         layout.Controls.Add(ctrl, 0, layout.RowCount
- 1);
        layout.SetColumnSpan(ctrl, 2);
        layout.RowCount++;
```

```
        layout.ResumeLayout(true);
    }

    private void AddControl(Control ctrl, int heigh,
string InsertBeforeProperty)
    {
                                    var    layout    =
ValueStorage.Controls[0].Controls[0]              as
TableLayoutPanel;
        layout.SuspendLayout();
        ctrl.Dock = DockStyle.Fill;
        layout.Parent.Height += heigh;
                                var    beforectrl    =
ValueStorage.GetControlForProperty(InsertBeforePrope
rty);
                                 var    position    =
layout.GetPositionFromControl(beforectrl).Row + 1;
          for (int r = layout.RowCount; position <=
r--;)
        {
               for (int c = layout.ColumnCount; 0 !=
c--;)
            {
                                       var  control  =
layout.GetControlFromPosition(c, r);
                                if  (control  !=  null)
layout.SetRow(control, r + 1);
            }
        }
           layout.RowStyles.Insert(position - 1, new
RowStyle(SizeType.Absolute, heigh));
        layout.Controls.Add(ctrl, 0, position - 1);
        layout.SetColumnSpan(ctrl, 2);
        layout.RowCount++;
        layout.ResumeLayout(true);
```

```
        }
}
```

6. Литература

1.Материалы данные преподавателем Docs: https://docs.google.com/document/d/1btdhORu6SZRn5kI5W5lEE7Tur8-Fc-qY7snxN0_V4ho/edit

2 Справочник по C# URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/

3.Wolframalpha URL: https://www.wolframalpha.com/