Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №7 по курсу**

**«Компьютерная графика»**

Выполнил: Лисин Р.С.

Группа: М8О-306Б-20

Преподаватель: Филиппов Г.С.

Дата: \_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_

Москва, 2022

**Лабораторная работа №7**

**Построение плоских полиномиальных кривых.**

**Задание:** Написать программу, строящую полиномиальную кривую по заданным точкам. Обеспечить возможность изменения позиции точек и, при необходимости, значений касательных векторов и натяжения.

**Вариант 5:** Кривая Безье 2-й степени.

**Описание**

Для выполнения данного задания я использовал библиотеку matplotlib для Python. Эта библиотека позволяет довольно просто работать с двумерными изображениями, и в то же время она предлагает довольно много различных настроек отображения графиков. Также matplotlib автоматически масштабирует графики при изменении размера окна и позволяет увеличивать/уменьшать изображение.

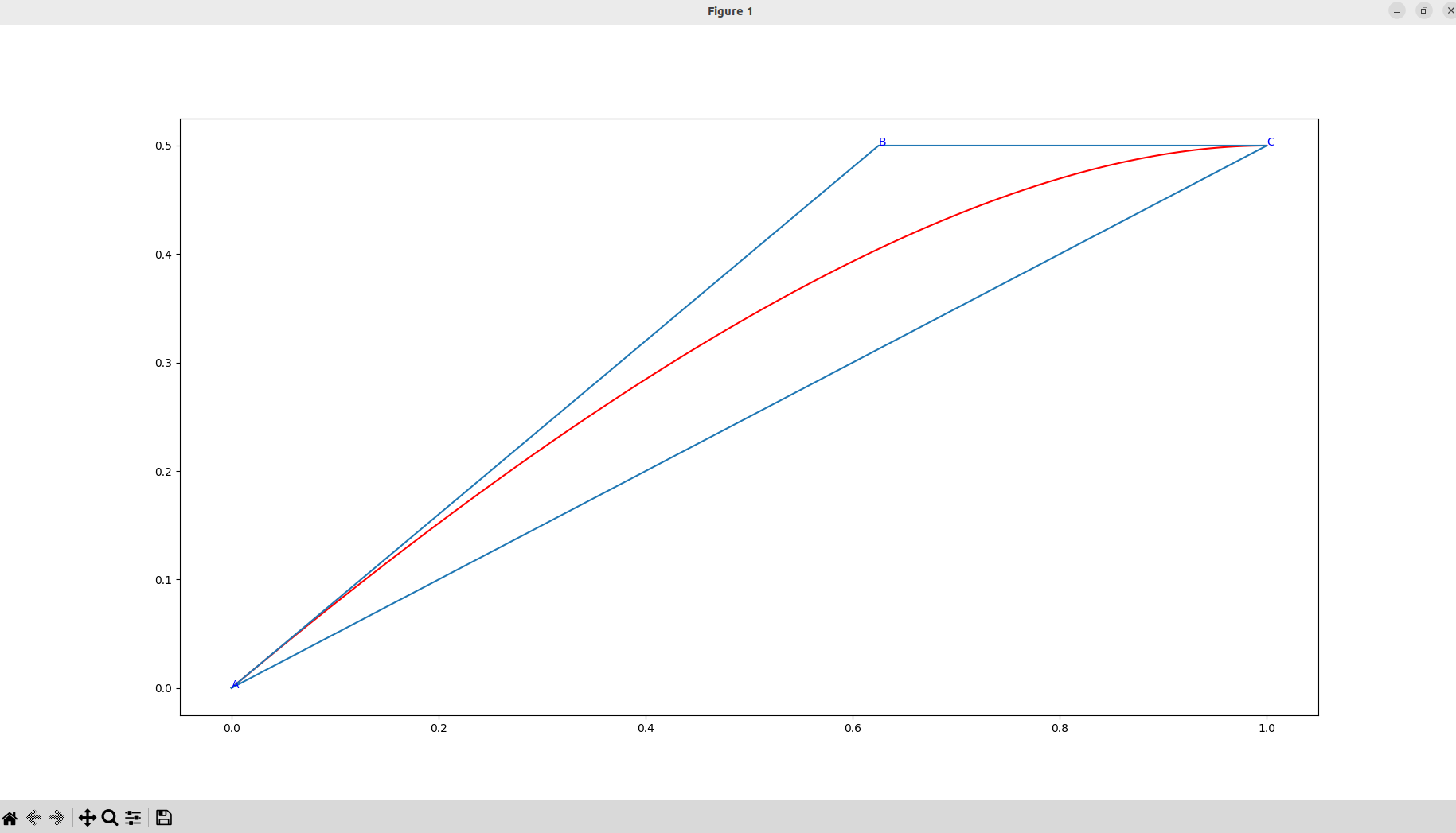
Для обработки пользовательского ввода я использовал модуль PySimpleGUI, который на самом деле является одним из самых простых для этого.

Формулу для кривой Безье я написал по определению для любого n. При задании n опорных точек функция возвращает кривую Безье n - 1 степени. Также я строю треугольник вокруг кривой, чтобы показать, что 2 его стороны будут касательными к кривой.

**Исходный код**

import PySimpleGUI as sg  
import math  
import matplotlib.pyplot as plt  
  
  
def bezier\_formula(n, t, index, def\_points):  
 return sum(math.comb(n - 1, i) \* ((1 - t) \*\* (n - 1 - i)) \* (t \*\* i) \* def\_points[i][index] for i in range(n))  
  
  
def bezier\_curve\_explicit(def\_points, speed=0.01):  
 n = len(def\_points)  
 points = []  
 for t in [\_ \* speed for \_ in range(int((1 + speed \* 2) // speed))]: # get values between 0 and 1  
 points.append([bezier\_formula(n, t, 0, def\_points), bezier\_formula(n, t, 1, def\_points)])  
 return points  
  
  
initial\_data = [  
 [0, 0],  
 [0.625, 0.5],  
 [1, 0.5],  
]  
  
  
def draw\_bezier(data=initial\_data):  
 points = bezier\_curve\_explicit(data)  
 plt.plot([point[0] for point in points], [point[1] for point in points], 'r')  
 for point, name in zip(data, ("A", "B", "C")):  
 plt.text(point[0], point[1], name, color='b')  
 plt.plot([point[0] for point in data + [data[0]]], [point[1] for point in data + [data[0]]])  
 plt.show()  
  
  
layout = [  
 [sg.Text('Points: '), sg.InputText()],  
 [sg.Button('Plot'), sg.Button('Exit'), sg.Button('Enter')]]  
  
window = sg.Window("Window", layout)  
  
while True:  
 event, values = window.read()  
 if event in (sg.WIN\_CLOSED, 'Exit'):  
 break  
 elif event == 'Plot':  
 draw\_bezier()  
 elif event == 'Enter':  
 i = 2  
 points = []  
 for point in values[0].split():  
 if i % 2 == 0:  
 points.append([float(point[1:len(point) - 1])])  
 else:  
 points[-1].append(float(point[:len(point) - 1]))  
 i += 1  
 draw\_bezier(points)  
window.close()  
  
# (1, 2) (20, 1) (15, 17)  
# (3.5, 100) (7, 1.3) (15.9090, 17)

**Результат работы**

****

**Выводы**

Выполнив эту лабораторную работу, я научился рисовать полиномиальную кривую Безье на компьютере при помощи библиотеки matplotlib. Кривые Безье достаточно часто используются в компьютерной графике и пригодятся мне в дальнейшем.