Московский авиационный институт
(Национальный исследовательский университет)

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

Курсовая работа

Тема: Построение полиномиальных поверхностей.

Студент: Махмудов Орхан
Группа: О8-305
Преподаватель: Чернышов Л.Н.
Дата:
Оценка:

Москва, 2020

1. Постановка задачи

Тема: Тема: Построение плоских полиномиальных поверхностей.

Задание: Составить и отладить программу, обеспечивающую каркасную визуализацию порции поверхности заданного типа. Исходные данные готовятся самостоятельно и вводятся из файла или в панели ввода данных. Должна быть обеспечена возможность тестирования программы на различных наборах исходных данных. Программа должна обеспечивать выполнение аффинных преобразований для заданной порции поверхности, а также возможность управлять количеством изображаемых параметрических линий. Для визуализации параметрических линий поверхности разрешается использовать только функции отрисовки отрезков в экранных координатах.

Вариант №6: Линейная поверхность Кунса (границы – кубические кривые Безье 3D)

Решение задачи:

3.6. Поверхности Кунса

Возьмем векторную функцию линейчатой поверхности (3.5.1), прибавим к ней и вычтем из нее векторную функцию билинейной поверхности (3.5.2), где $\mathbf{p}_1$ и $\mathbf{p}_2$ — концевые точки направляющей кривой $\mathbf{a}(t)$, а $\mathbf{p}_3$ и $\mathbf{p}_4$ — концевые точки направляющей кривой $\mathbf{d}(w)$. В результате этих действий векторная функция не изменится, но будет иметь вид

$$\begin{aligned}\mathbf{r}(u, v) = (1-v)\mathbf{a}(u) + v\mathbf{d}(u) + (1-u)(\mathbf{p}_1(1-v) + \mathbf{p}_3 v) + \\ + u(\mathbf{p}_2(1-v) + \mathbf{p}_4 v) - (1-u)(1-v)\mathbf{p}_1 - u(1-v)\mathbf{p}_2 - \\ - (1-u)v\mathbf{p}_3 - uv\mathbf{p}_4 = (1-v)\mathbf{a}(u) + v\mathbf{d}(u) + (1-u)\mathbf{b}(v) + u\mathbf{c}(v) - \\ - (1-u)(1-v)\mathbf{p}_1 - u(1-v)\mathbf{p}_2 - (1-u)v\mathbf{p}_3 - uv\mathbf{p}_4, \quad (3.6.1)\end{aligned}$$

$$0 \leqslant u \leqslant 1, \quad 0 \leqslant v \leqslant 1.$$

В обозначениях функций $\mathbf{a}(u)$ и $\mathbf{d}(u)$ мы использовали их реальный аргумент u, а также ввели обозначения для отрезков прямых $\mathbf{b}(v) = \mathbf{p}_1(1-v) + \mathbf{p}_3 v$, $\mathbf{c}(v) = \mathbf{p}_2(1-v) + \mathbf{p}_4 v$, соединяющих концы кривых $\mathbf{a}(u)$ и $\mathbf{d}(u)$. Если допустить, что в качестве $\mathbf{b}(v)$ и $\mathbf{c}(v)$ могут использоваться произвольные кривые,

В нашем случае направляющие кривые - это кривые Безье 3-ей степени

Формулы взяты из этой статьи [Поверхность Кунса]

2. Описание программы

Используемая среда: Visual Studio Code

Используемые библиотеки: Matplotlib, Numpy, Tkinter

Язык программирования: Python

Используемые структуры данных: массивы

Ввод: Вводятся все координаты точек и количество разбиений зарисовки поверхности

Вывод: Выводится поверхность с указанной зарисовкой

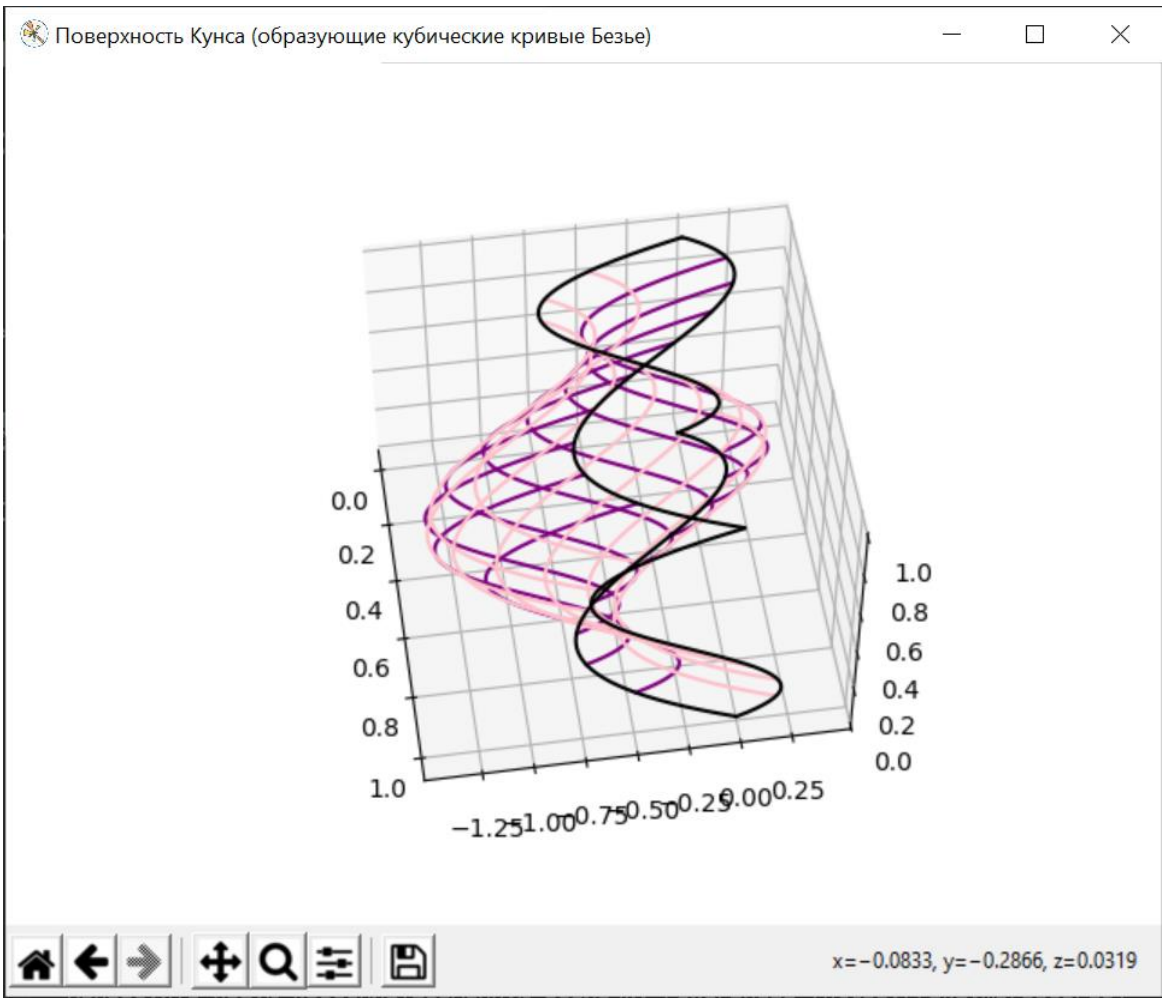
Краткая инструкция для пользователя:
Запуск программы по кнопке в среде разработки, после запуска появляется форма, в которой нужно вбить координаты точек кривой Безье и количество разбиений зарисовки поверхности (от 1 до 1000). После того, как всё введено, нужно нажать на кнопку “Запуск”, после нажатия появится окно с поверхностью.

3. Набор тестов

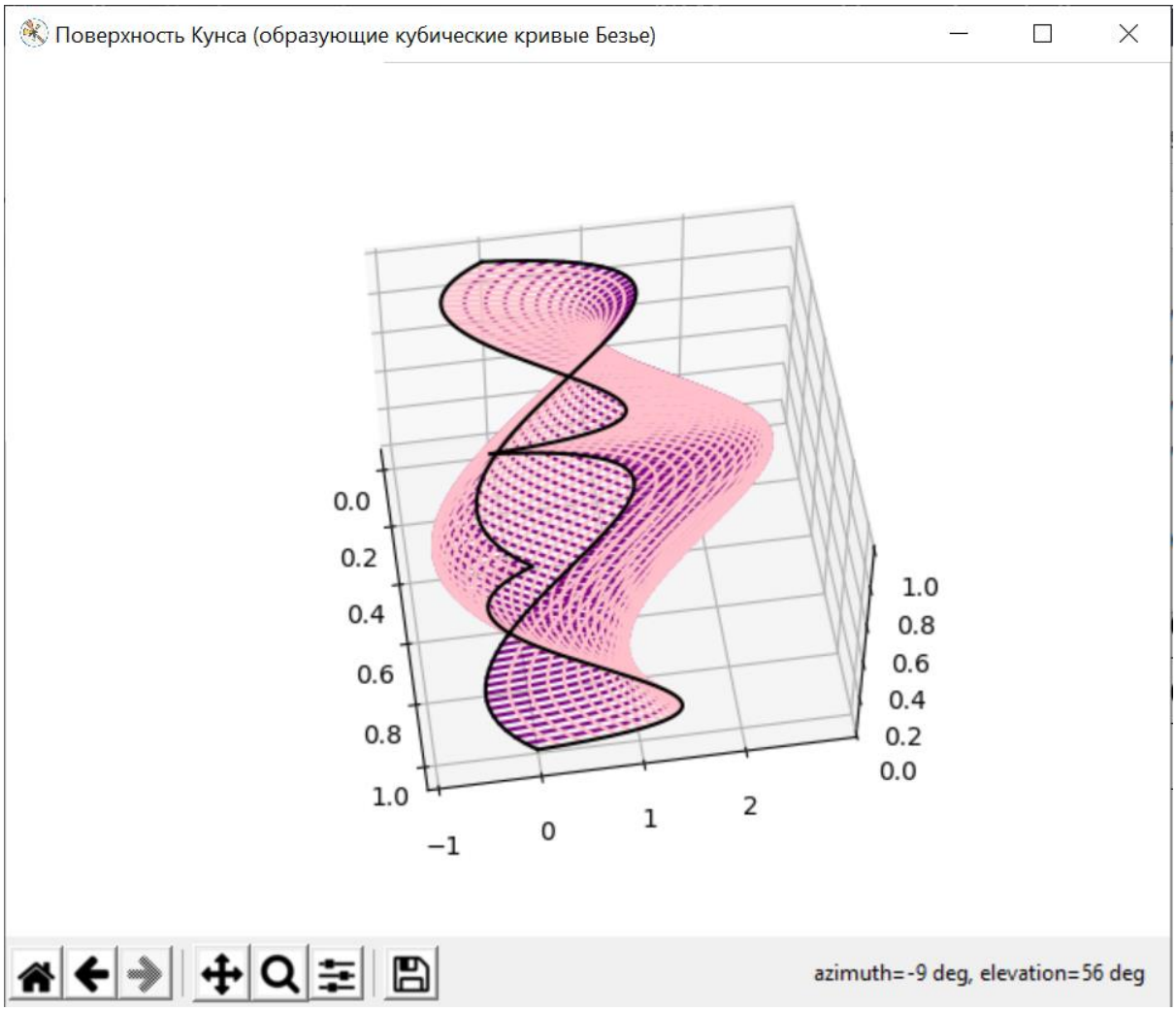
№ теста	Координаты 1 точки	Координаты 2 точки	Координаты 3 точки	Координаты 4 точки	Кол-во разбиений
1	(0,0)	(0.3,1)	(0.6, -2)	(1,0)	10
2	(0,0)	(0.1,4)	(0.7,-5)	(1,0)	50
3	(0,0)	(0.2,-16)	(0.7,10)	(1,0)	150

4. Результаты выполнения тестов

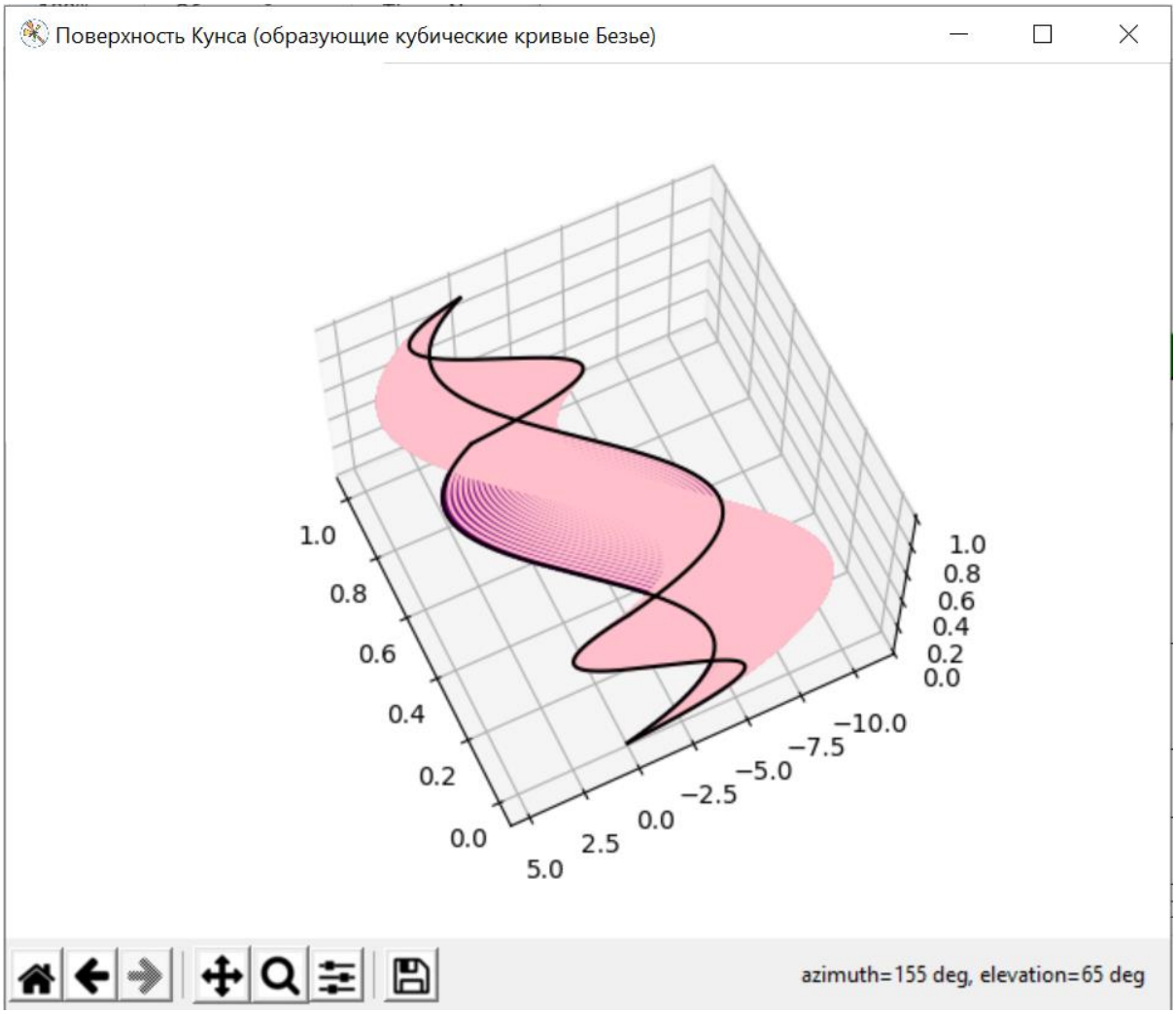
№ теста	Координаты 1 точки	Координаты 2 точки	Координаты 3 точки	Координаты 4 точки	Кол-во разбиений
1	(0,0)	(0.3,1)	(0.6, -2)	(1,0)	10



№ теста	Координаты 1 точки	Координаты 2 точки	Координаты 3 точки	Координаты 4 точки	Кол-во разбиений
2	(0,0)	(0.1,4)	(0.7,-5)	(1,0)	50



№ теста	Координаты 1 точки	Координаты 2 точки	Координаты 3 точки	Координаты 4 точки	Кол-во разбиений
3	(0,0)	(0.2,-16)	(0.7,10)	(1,0)	150



5. Листинг программы

```
#Махмудов Орхан группа М8О-305Б-18
#Поверхность Кунса (образующие кубические кривые Безье)
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from tkinter import Tk, Label, Button, Entry
def bezier (x1,y1,x2,y2,x3,y3,x4,y4):
    t = np.arange(0, 1, 0.001)
    x = np.zeros(len(t))
    y = np.zeros(len(t))
    for i in range (len(t)):
        x[i] =  (1 - t[i])**3 * t[i] * x1 + 3 * (1 - t[i])**2 * t[i] *
x2 + 3 * (1 - t[i]) * t[i]**2 * x3 + t[i]**3 * x4
        y[i] =  (1 - t[i])**3 * t[i] * y1 + 3 * (1 - t[i])**2 * t[i] *
y2 + 3 * (1 - t[i]) * t[i]**2 * y3 + t[i]**3 * y4
    return x, y
def clicked():
    x1 = float(txt1.get())
    y1 = float(txt2.get())
    x2 = float(txt3.get())
    y2 = float(txt4.get())
    x3 = float(txt5.get())
    y3 = float(txt6.get())
    x4 = float(txt7.get())
    y4 = float(txt8.get())
    n = int(txt9.get())
    t = np.arange(0, 1, 0.001)
    z0 = np.zeros(len(t))
    z1 = np.ones(len(t))
    x_a, y_a = bezier(x1,y1,x2,y2,x3,y3,x4,y4)
    fig = plt.figure("Поверхность Кунса (образующие кубические кривые
Безье)")
    ax = fig.gca(projection='3d')
    i = 0
    while i < (len(t) - (1000//n)):
        i = i + (1000//n)
        pr = z0 + x_a[i]
        yt = y_a + y_a[i]
        ax.plot(pr, yt, x_a, color = 'purple')
    i = 0
    while i < (len(t) - (1000//n)):
        i = i + (1000//n)
        pr = z0 + x_a[i]
        yt = y_a + y_a[i]
        ax.plot(x_a, yt, pr, color = 'pink')
    ax.plot(x_a,y_a,z0, color = 'black')
    ax.plot(x_a,y_a,z1, color = 'black')
    ax.plot(z0,y_a,x_a, color = 'black')
    ax.plot(z1,y_a,x_a, color = 'black')
    plt.show()
window = Tk()
window.title("Поверхность Кунса (образующие кубические кривые Безье)")
```

```
window.geometry("400x200")
lbl = Label(window, text = "Введите координаты точек!")
lbl.place(x = 1, y = 1)
lb2 = Label(window, text = "Координаты первой точки        x1:")
lb2.place(x = 1, y = 25)
txt1 = Entry(window, width = 4)
txt1.place(x = 190, y = 25)
txt1.insert(0,"0")
txt1.Text = Entry(textvariable = "1")
lb3 = Label(window, text = "     y1:")
lb3.place(x = 220, y = 25)
txt2 = Entry(window, width = 4)
txt2.place(x = 255, y = 25)
txt2.insert(0,"0")
lb4 = Label(window, text = "Координаты второй точки        x2:")
lb4.place(x = 1, y = 50)
txt3 = Entry(window, width = 4)
txt3.place(x = 190, y = 50)
txt3.insert(0,"0.3")
lb5 = Label(window, text = "     y2:")
lb5.place(x = 220, y = 50)
txt4 = Entry(window, width = 4)
txt4.place(x = 255, y = 50)
txt4.insert(0,"1")
lb6 = Label(window, text = "Координаты третьей точки       x3:")
lb6.place(x = 1, y = 75)
txt5 = Entry(window, width = 4)
txt5.place(x = 190, y = 75)
txt5.insert(0,"0.6")
lb7 = Label(window, text = "     y3:")
lb7.place(x = 220, y = 75)
txt6 = Entry(window, width = 4)
txt6.place(x = 255, y = 75)
txt6.insert(0,"-2")
lb8 = Label(window, text = "Координаты четвёртой точки  x4:")
lb8.place(x = 1, y = 100)
txt7 = Entry(window, width = 4)
txt7.place(x = 190, y = 100)
txt7.insert(0,"1")
lb9 = Label(window, text = "     y4:")
lb9.place(x = 220, y = 100)
txt8 = Entry(window, width = 4)
txt8.place(x = 255, y = 100)
txt8.insert(0,"0")
lb10 = Label(window, text = "Введите количество разбиений от 1 до 1000:
")
lb10.place(x = 1, y = 150)
txt9 = Entry(window, width = 4)
txt9.place(x = 260, y = 150)
btn = Button(window, text = "Запуск", bg = "green", fg = "white", command
= clicked)
btn.place(x = 300, y = 150)
window.mainloop()
```

Вывод

В ходе данной лабораторной работы были приобретены навыки по работе с полиномиальными поверхностями. Строить график поверхностей по заданному разбиению и изменять график в зависимости от изменений параметров.

Список литературы

1. Работа с библиотекой Matplotlib [Электронный ресурс] URL: https://matplotlib.org/

   (дата обращения: 12.12.2020)

2. Создание графического интерфейса Tkinter [Электронный ресурс] URL: https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter

   (дата обращения: 12.12.2020)