**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Компьютерная графика»

**Курсовой проект**

Тема: Визуализацию порции поверхности

заданного типа.

Студент: Чекушкин Д.И.

Группа: 80-304

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2018

1. Постановка задачи

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 13) Поверхность вращения. Образующая – кривая Безье 3D 2-й степени

1. Решения задачи

ЯП: Python

ОС: Ubuntu 16.04

Библиотеки: numpy, matplotlib.pyplot, mpl\_toolkits.mplot3d, math

matplotlib.pyplot.plot(\*args, \*\*kwargs) - Создает график

matplotlib.pyplot.show() - Отображает окно с графиком

numpy.arange(arg1,arg2) - Cоздания последовательностей чисел

Ход работы:

* Создаем кривую Безье (на основе полиномов Бернштейна). Вычисляем значения координат кривой с высокой точностью(1000 измерений).
* Задаем 3 точки для построения первой кривой. С помощью преобразований вращаем эти три точки на заданный угол относительно заданной прямой.
* Отрисовываем получившиеся после преобразования кривые, получая поверхность вращения.

При тестировании были выявлены и исправлены незначительные ошибки, связанные с поворотом кривой относительно прямой.

1. Руководство по использованию программы

Файл test.py содержит код программы .

На вход не подаются координаты кривой Безье и прямой, относительно которой будет строиться поверхность вращения, на выход – график искомого каркаса поверхности.

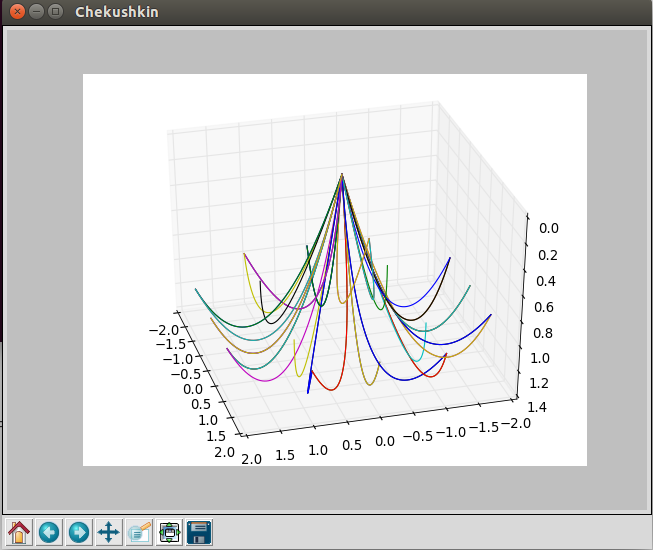


Рис 1 - Вывод программы

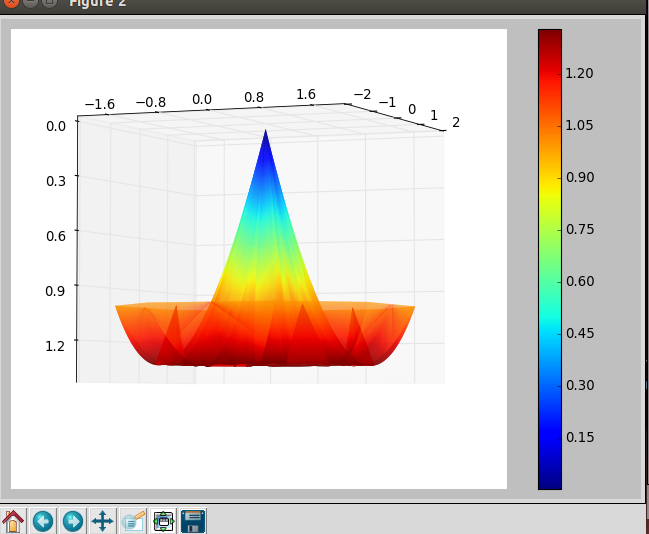


Рис 2 - Вывод программы (второй вариант)

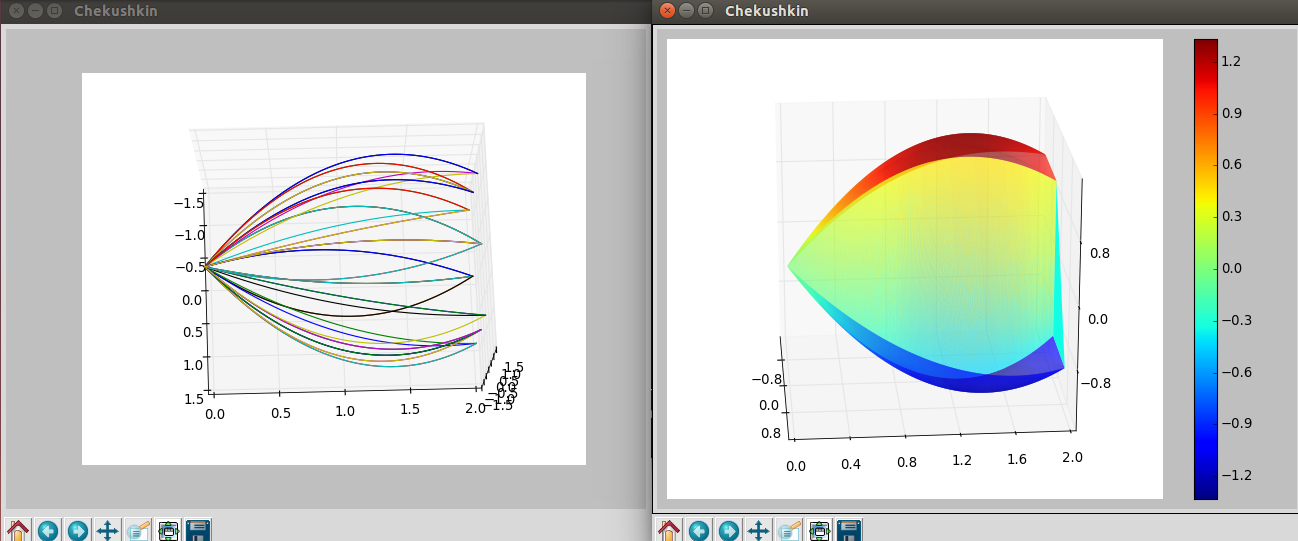


Рис 3 - Вращение относительно другой оси

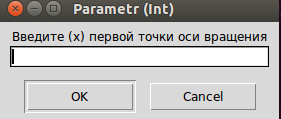


Рис 4 - Диалоговое окно ввода данных

1. Листинг программы

import sys

import matplotlib

import matplotlib.pyplot as plt

from matplotlib.ticker import MaxNLocator

from matplotlib import cm

from mpl\_toolkits.mplot3d import Axes3D

from numpy.random import randn

from scipy import array, newaxis

###############################

try:

# for Python2

import Tkinter as tk

import tkSimpleDialog as tksd

except:

# for Python3

import tkinter as tk

import tkinter.simpledialog as tksd

global ar1x, ar1y, ar1z, ar2x, ar2y, ar2z

global bc1x, bc1y, bc1z, bc2x, bc2y, bc2z, bc3x, bc3y, bc3z

root = tk.Tk()

ar1x = tksd.askinteger("Parametr (Int)", "Введите (х) первой точки оси вращения", parent=root, minvalue=0)

ar1y = tksd.askinteger("Parametr (Int)", "Введите (y) первой точки оси вращения", parent=root, minvalue=0)

ar1z = tksd.askinteger("Parametr (Int)", "Введите (z) первой точки оси вращения", parent=root, minvalue=0)

ar2x = tksd.askinteger("Parametr (Int)", "Введите (х) второй точки оси вращения", parent=root, minvalue=0)

ar2y = tksd.askinteger("Parametr (Int)", "Введите (y) второй точки оси вращения", parent=root, minvalue=0)

ar2z = tksd.askinteger("Parametr (Int)", "Введите (z) второй точки оси вращения", parent=root, minvalue=0)

bc1x=tksd.askinteger("Parametr (Int)", "Введите (х) первой точки кривой Безье", parent=root, minvalue=0)

bc1y=tksd.askinteger("Parametr (Int)", "Введите (y) первой точки кривой Безье", parent=root, minvalue=0)

bc1z=tksd.askinteger("Parametr (Int)", "Введите (z) первой точки кривой Безье", parent=root, minvalue=0)

bc2x=tksd.askinteger("Parametr (Int)", "Введите (x) второй точки кривой Безье", parent=root, minvalue=0)

bc2y=tksd.askinteger("Parametr (Int)", "Введите (y) второй точки кривой Безье", parent=root, minvalue=0)

bc2z=tksd.askinteger("Parametr (Int)", "Введите (z) второй точки кривой Безье", parent=root, minvalue=0)

bc3x=tksd.askinteger("Parametr (Int)", "Введите (x) третьей точки кривой Безье", parent=root, minvalue=0)

bc3y=tksd.askinteger("Parametr (Int)", "Введите (y) третьей точки кривой Безье", parent=root, minvalue=0)

bc3z=tksd.askinteger("Parametr (Int)", "Введите (z) третьей точки кривой Безье", parent=root, minvalue=0)

#Чекушкин М8О-304Б

# Вариант 13) Поверхность вращения. Образующая – кривая Безье 3D 2-й степени

def bernstein\_poly(i, n, t):

return comb(n, i) \* (t\*\*(n - i)) \* (1 - t)\*\*i

def bezier\_curve(points, nTimes=1000):

nPoints = len(points)

xPoints = np.array([p[0] for p in points])

yPoints = np.array([p[1] for p in points])

zPoints = np.array([p[2] for p in points])

t = np.linspace(0.0, 1.0, nTimes)

polynomial\_array = np.array(

[bernstein\_poly(i, nPoints - 1, t) for i in range(0, nPoints)])

xvals = np.dot(xPoints, polynomial\_array)

yvals = np.dot(yPoints, polynomial\_array)

zvals = np.dot(zPoints, polynomial\_array)

return xvals, yvals, zvals

from math import pi ,sin, cos

def R(theta, u):

return [[cos(theta) + u[0]\*\*2 \* (1-cos(theta)),

u[0] \* u[1] \* (1-cos(theta)) - u[2] \* sin(theta),

u[0] \* u[2] \* (1 - cos(theta)) + u[1] \* sin(theta)],

[u[0] \* u[1] \* (1-cos(theta)) + u[2] \* sin(theta),

cos(theta) + u[1]\*\*2 \* (1-cos(theta)),

u[1] \* u[2] \* (1 - cos(theta)) - u[0] \* sin(theta)],

[u[0] \* u[2] \* (1-cos(theta)) - u[1] \* sin(theta),

u[1] \* u[2] \* (1-cos(theta)) + u[0] \* sin(theta),

cos(theta) + u[2]\*\*2 \* (1-cos(theta))]]

def Rotate(pointToRotate, point1, point2, theta):

u= []

squaredSum = 0

for i,f in zip(point1, point2):

u.append(f-i)

squaredSum += (f-i) \*\*2

u = [i/squaredSum for i in u]

r = R(theta, u)

rotated = []

for i in range(3):

rotated.append(round(sum([r[j][i] \* pointToRotate[j] for j in range(3)])))

return rotated

if \_\_name\_\_ == "\_\_main\_\_":

nPoints = 3

points = [[bc1x,bc1y,bc1z],[bc2x,bc2y,bc2z],[bc3x,bc3y,bc3z]]

xpoints = [p[0] for p in points]

ypoints = [p[1] for p in points]

zpoints = [p[2] for p in points]

xvals, yvals, zvals = bezier\_curve(points, nTimes=1000)

fig = plt.figure()

ax = fig.gca(projection='3d')

ax.plot(xvals, yvals, zvals, label='bezier')

#ax.plot(xpoints, ypoints, zpoints, "ro")

# 1 1 2

# 1 2 2

# 0 0 0

# 0 0 1

p1=[ar1x,ar1y,ar1z]

p2=[ar2x,ar2y,ar2z]

radiane = 0

angle = pi/12

xtvals=xvals

ytvals=yvals

ztvals=zvals

while angle <= 2\*pi:

pp1 = Rotate(points[0], p1, p2, angle)

pp2 = Rotate(points[1], p1, p2, angle)

pp3 = Rotate(points[2], p1, p2, angle)

npoints=[pp1,pp2,pp3]

xnvals, ynvals, znvals = bezier\_curve(npoints, nTimes=1000)

xtvals = np.append( xtvals , xnvals )

ytvals = np.append( ytvals , ynvals )

ztvals = np.append( ztvals , znvals )

ax.plot(xnvals, ynvals, znvals, label='bezier')

print(angle)

angle= angle + pi/24

plt.gcf().canvas.set\_window\_title("Chekushkin")

#plt.show()

#myavi-lib

#####################################

#pts = mlab.points3d(xtvals, ytvals, ztvals, ztvals)

#mesh = mlab.pipeline.delaunay2d(pts)

#pts.remove()

#surf = mlab.pipeline.surface(mesh)

#mlab.xlabel("x")

#mlab.ylabel("y")

#mlab.zlabel("z")

#mlab.show()

######################################

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

surf = ax.plot\_trisurf(xtvals, ytvals, ztvals, cmap=cm.jet, linewidth=0)

fig.colorbar(surf)

ax.xaxis.set\_major\_locator(MaxNLocator(5))

ax.yaxis.set\_major\_locator(MaxNLocator(6))

ax.zaxis.set\_major\_locator(MaxNLocator(5))

fig.tight\_layout()

plt.gcf().canvas.set\_window\_title("Chekushkin")

plt.show()

pylab.show()

Выводы: благодаря проделанной работе мне удалось освоить создание порций поверхностей в Python

Список литературы:

1. Описание кривой Безье. Алгоритм построения [Электронный ресурс] Url: https://ru.wikipedia.org/wiki/Кривая\_Безье#Квадратичные\_кривые
2. Полиномы Бернштейна и составные кривых Безье [Электронный ресурс] Url: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=zvmmf&paperid=378&option\_lang=rus
3. Статья “По следу кривой (второе путешествие “за кулисы” компьютерной графики)” [Электронный ресурс] Url: https://www.osp.ru/news/articles/1999/0826/13031905