ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

# ОТЧЕТ

**О ВЫПЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

# «АНИМАЦИЯ ТОЧКИ»

**ПО ДИСЦИПЛИНЕ «ТЕОРЕТИЧЕСКАЯ МЕХАНИКА И ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ»**

# ВАРИАНТ ЗАДАНИЯ № 3

#### Выполнила студентка группы М8О-208Б-21 Шевлякова С. С.

подпись, дата

#### Проверил и принял Зав. каф. 802, Бардин Б.С.

подпись, дата

#### с оценкой

Москва, 2022

**Вариант № 3 «Фантастическая кривая»**

**Задание:**

Построить заданную траекторию и анимацию движения точки, а также отобразить стрелки скорости, ускорения и радиус кривизны.

**Закон движения точки:**

**Текст программы:**

import matplotlib.pyplot as plt

import numpy as np

import math

from matplotlib.animation import FuncAnimation

import sympy as sp

FRAMES\_COUNT = 1000

t = sp.Symbol('t')

T = np.linspace(1, 14, FRAMES\_COUNT)

r = 1 + sp.sin(5\*t)

phi = t + 0.3\*sp.sin(30\*t)

x = r \* sp.cos(phi)

y = r \* sp.sin(phi)

Vx = sp.diff(x, t)

Vy = sp.diff(y, t)

Wx = sp.diff(Vx, t)

Wy = sp.diff(Vy, t)

V = sp.sqrt(Vx\*\*2 + Vy\*\*2)

R = np.zeros\_like(T)

PHI = np.zeros\_like(T)

X = np.zeros\_like(T)

Y = np.zeros\_like(T)

VX = np.zeros\_like(T)

VY = np.zeros\_like(T)

WX = np.zeros\_like(T)

WY = np.zeros\_like(T)

for i in np.arange(len(T)):

R[i] = sp.Subs(r, t, T[i])

PHI[i] = sp.Subs(phi, t, T[i])

X[i] = sp.Subs(x, t, T[i])

Y[i] = sp.Subs(y, t, T[i])

VX[i] = sp.Subs(Vx, t, T[i])

VY[i] = sp.Subs(Vy, t, T[i])

WX[i] = sp.Subs(Wx, t, T[i])

WY[i] = sp.Subs(Wy, t, T[i])

fig = plt.figure()

ax1 = fig.add\_subplot(1, 1, 1)

ax1.axis('equal')

ax1.set(xlim=[-2.5, 2.5], ylim=[-2.5, 2.5])

ax1.plot(X, Y, color="#e069d8")

P, = ax1.plot(X[0], Y[0], color="black", marker='o')

Vline, = ax1.plot([X[0], X[0] + VX[0]], [Y[0], Y[0] + VY[0]], 'r')

Vline2, = ax1.plot([X[0], X[0] + WX[0]], [Y[0], Y[0] + WY[0]], 'g')

Vline3, = ax1.plot([0, X[0]], [0, Y[0]], 'b')

def Rot2D(X, Y, Alpha):

RX = X \* np.cos(Alpha) - Y \* np.sin(Alpha)

RY = X \* np.sin(Alpha) + Y \* np.cos(Alpha)

return RX, RY

arrow\_size=1

ArrowX = np.array([-0.1\*arrow\_size, 0, -0.1\*arrow\_size])

ArrowY = np.array([0.05\*arrow\_size, 0, -0.05\*arrow\_size])

ArrowWX = np.array([-0.1\*arrow\_size, 0, -0.1\*arrow\_size])

ArrowWY = np.array([0.05\*arrow\_size, 0, -0.05\*arrow\_size])

ArrowRX = np.array([-0.1\*arrow\_size, 0, -0.1\*arrow\_size])

ArrowRY = np.array([0.05\*arrow\_size, 0, -0.05\*arrow\_size])

RArrowX, RArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[0], VX[0]))

RArrowWX, RArrowWY = Rot2D(ArrowWX, ArrowWY, math.atan2(WY[0], WX[0]))

RArrowRX, RArrowRY = Rot2D(ArrowRX, ArrowRY, math.atan2(X[0], Y[0]))

VArrow, = ax1.plot(RArrowX + X[0] + VX[0], RArrowY + Y[0] + VY[0], 'r')

WArrow, = ax1.plot(RArrowWX + X[0] + WX[0], RArrowY + Y[0] + WY[0], 'g')

RArrow, = ax1.plot(ArrowRX + X[0], ArrowRY + Y[0], 'b')

def anima(j):

P.set\_data(X[j], Y[j])

Vline.set\_data([X[j], X[j] + VX[j]], [Y[j], Y[j] + VY[j]])

Vline2.set\_data([X[j], X[j] + WX[j]], [Y[j], Y[j] + WY[j]])

Vline3.set\_data([0, X[j]], [0, Y[j]])

RArrowX, RArrowY = Rot2D(ArrowX, ArrowY, math.atan2(VY[j], VX[j]))

VArrow.set\_data(RArrowX + X[j] + VX[j], RArrowY + Y[j] + VY[j])

RArrowWX, RArrowWY = Rot2D(ArrowWX, ArrowWY, math.atan2(WY[j], WX[j]))

WArrow.set\_data(RArrowWX + X[j] + WX[j], RArrowWY + Y[j] + WY[j])

RArrowRX, RArrowRY = Rot2D(ArrowRX, ArrowRY, math.atan2(Y[j], X[j]))

RArrow.set\_data(RArrowRX + X[j], RArrowRY + Y[j])

return P, Vline, VArrow, Vline2, WArrow, Vline3, RArrow

anim = FuncAnimation(fig, anima, frames=FRAMES\_COUNT, interval=100, blit=True, repeat=True)

plt.grid()

plt.show()

### Результат работы программы:

|  |  |
| --- | --- |
| **Рис.1** | **Рис.2** |
| **Рис.3** | **Рис.4** |