Московский авиационный институт

(национальный исследовательский университет)

Институт № 8 «Информационные технологии и прикладная математика»

**Лабораторная работа №1**

**по курсу «Теоретическая механика»**

**Анимация точки**

Выполнил студент группы М8О-203Б-23

Салихов Руслан Ринатович

Преподаватель: Волков Александр Владимирович

Оценка:

Дата:

Москва, 2024

**Вариант № 18**

**Задание:**

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

**Закон движения точки:**

**Текст программы**

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

import sympy as sp

t = sp.Symbol('t')

r = 1 + 1.5 \* sp.sin(12 \* t)

phi = 1.2 \* t + 0.2 \* sp.cos(12 \* t)

x = r \* sp.cos(phi)

y = r \* sp.sin(phi)

Vx = sp.diff(x, t)

Vy = sp.diff(y, t)

Ax = sp.diff(Vx, t)

Ay = sp.diff(Vy, t)

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

kappa = sp.Abs(Vx \* Ay - Vy \* Ax) / V\*\*3

rho = 1 / kappa

F\_x = sp.lambdify(t, x, modules='numpy')

F\_y = sp.lambdify(t, y, modules='numpy')

F\_Vx = sp.lambdify(t, Vx, modules='numpy')

F\_Vy = sp.lambdify(t, Vy, modules='numpy')

F\_Ax = sp.lambdify(t, Ax, modules='numpy')

F\_Ay = sp.lambdify(t, Ay, modules='numpy')

F\_rho = sp.lambdify(t, rho, modules='numpy')

t\_vals = np.linspace(0, 2 \* np.pi, 1000)

x\_vals = F\_x(t\_vals)

y\_vals = F\_y(t\_vals)

Vx\_vals = F\_Vx(t\_vals)

Vy\_vals = F\_Vy(t\_vals)

Ax\_vals = F\_Ax(t\_vals)

Ay\_vals = F\_Ay(t\_vals)

rho\_vals = F\_rho(t\_vals)

Alpha\_V = np.arctan2(Vy\_vals, Vx\_vals)

Alpha\_A = np.arctan2(Ay\_vals, Ax\_vals)

fig, ax = plt.subplots(figsize=(12, 12))

ax.axis('equal')

ax.set\_xlim(-8, 8)

ax.set\_ylim(-8, 8)

ax.grid(True)

ax.plot(x\_vals, y\_vals, label='Траектория')

P, = ax.plot([], [], 'ro', label='Точка')

V\_line, = ax.plot([], [], color='red', label='Скорость')

V\_arrow, = ax.plot([], [], color='red')

A\_line, = ax.plot([], [], color='green', label='Ускорение')

A\_arrow, = ax.plot([], [], color='green')

circle = plt.Circle((0, 0), 0, color='purple', fill=False, label='Радиус кривизны')

ax.add\_patch(circle)

a = 0.05

b = 0.025

x\_arr = np.array([-a, 0, -a])

y\_arr = np.array([b, 0, -b])

k\_V = 0.2

k\_A = 0.05

max\_rho = 6

max\_center\_dist = 6

def Rot2D(X, Y, Alpha):

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

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

return RotX, RotY

def animate(i):

P.set\_data(x\_vals[i], y\_vals[i])

V\_end\_x = x\_vals[i] + k\_V \* Vx\_vals[i]

V\_end\_y = y\_vals[i] + k\_V \* Vy\_vals[i]

V\_line.set\_data([x\_vals[i], V\_end\_x], [y\_vals[i], V\_end\_y])

RotX\_V, RotY\_V = Rot2D(x\_arr, y\_arr, Alpha\_V[i])

V\_arrow.set\_data(V\_end\_x + RotX\_V, V\_end\_y + RotY\_V)

A\_end\_x = x\_vals[i] + k\_A \* Ax\_vals[i]

A\_end\_y = y\_vals[i] + k\_A \* Ay\_vals[i]

A\_line.set\_data([x\_vals[i], A\_end\_x], [y\_vals[i], A\_end\_y])

RotX\_A, RotY\_A = Rot2D(x\_arr, y\_arr, Alpha\_A[i])

A\_arrow.set\_data(A\_end\_x + RotX\_A, A\_end\_y + RotY\_A)

V\_mag = np.sqrt(Vx\_vals[i]\*\*2 + Vy\_vals[i]\*\*2)

if V\_mag == 0:

circle.set\_visible(False)

return P, V\_line, V\_arrow, A\_line, A\_arrow, circle

Nx = -Vy\_vals[i] / V\_mag

Ny = Vx\_vals[i] / V\_mag

x\_c = x\_vals[i] + Nx \* rho\_vals[i]

y\_c = y\_vals[i] + Ny \* rho\_vals[i]

center\_dist = np.sqrt((x\_c - x\_vals[i])\*\*2 + (y\_c - y\_vals[i])\*\*2)

if rho\_vals[i] <= max\_rho and center\_dist <= max\_center\_dist:

circle.set\_visible(True)

circle.center = (x\_c, y\_c)

circle.radius = rho\_vals[i]

else:

circle.set\_visible(False)

return P, V\_line, V\_arrow, A\_line, A\_arrow, circle

ani = FuncAnimation(fig, animate, frames=len(t\_vals), interval=20, blit=True)

ax.legend()

plt.show()

**Результат работы программы:**

|  |  |
| --- | --- |
|  |  |
|  |  |