Московский авиационный институт
(национальный исследовательский университет)
Институт № 8 «Информационные технологии и прикладная математика»

Лабораторная работа № 1
Анимация точки

Выполнил студент группы М8О-206Б-20
Семин Александр Витальевич
Преподаватель: Сухов Е. А.
Дата: 03.12.21
Оценка:

Москва, 2021

Задание:
Построить заданную траекторию и анимацию движения точки, а также отобразить стрелки скорости и ускорения.

Листинг программы:
```
import numpy as np
import sympy as sp
import matplotlib.pyplot as plt
import matplotlib
import math
from matplotlib.animation import FuncAnimation
matplotlib.use("TkAgg")

t = sp.Symbol('t')
R = 7.0
v0 = 10.0
phi = (v0 * t) / R
x = v0 * t - R * sp.sin(phi)
y = R - R * sp.cos(phi)
vpx = sp.diff(x, t)
vpy = sp.diff(y, t)
vp = (vpx ** 2 + vpy ** 2) ** 0.5
wpx = sp.diff(vpx, t)
wpy = sp.diff(vpy, t)
wpt = (vpx * wpx + vpy * wpy) / ((vpx ** 2 + vpy ** 2) ** 0.5)
wpn = (wpx ** 2 + wpy ** 2 - wpt ** 2) ** 0.5

T = np.linspace(0, 10, 200)
xn = np.zeros_like(T)
yn = np.zeros_like(T)
vpxn = np.zeros_like(T)
vpyn = np.zeros_like(T)
wpxn = np.zeros_like(T)
wpyn = np.zeros_like(T)
wptn = np.zeros_like(T)
wpnn = np.zeros_like(T)

for i in range(len(T)):
    xn[i] = sp.Subs(x, t, T[i])
    yn[i] = sp.Subs(y, t, T[i])
    vpxn[i] = sp.Subs(vpx, t, T[i])
    vpyn[i] = sp.Subs(vpy, t, T[i])
    wpxn[i] = sp.Subs(wpx, t, T[i])
    wpyn[i] = sp.Subs(wpy, t, T[i])
    wptn[i] = sp.Subs(wpt, t, T[i])
    wpnn[i] = sp.Subs(wpn, t, T[i])


fig = plt.figure(figsize=(v0 * T[len(T) - 1] / 10, R))
ax = fig.add_subplot(1, 1, 1)
ax.axis('equal')
ax.plot(xn, yn)
```

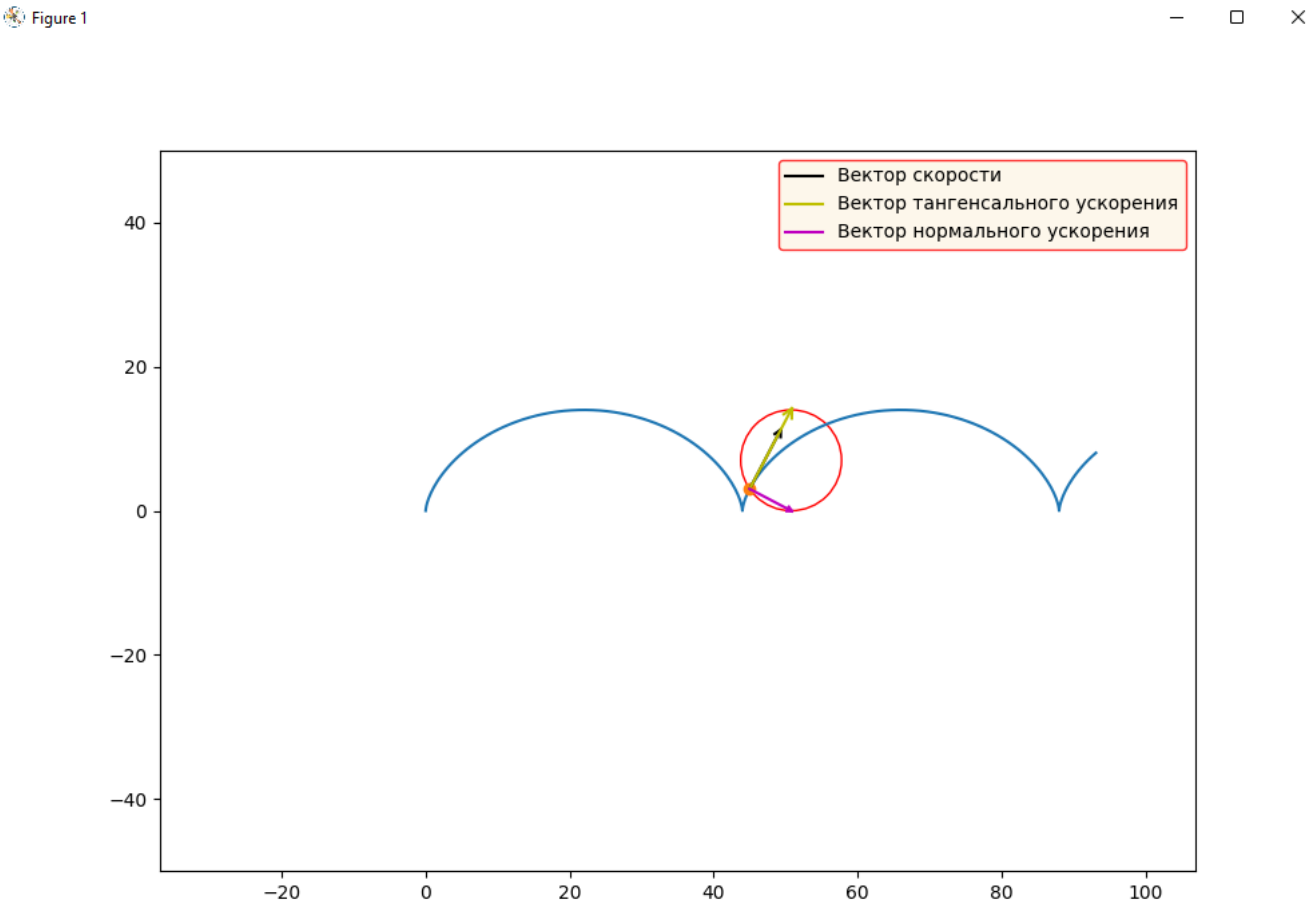
```
p = ax.plot(xn[0], yn[0], marker='o')[0]


def rotate(x, y, a):
    x_rotated = x * np.cos(a) - y * np.sin(a)
    y_rotated = x * np.sin(a) + y * np.cos(a)
    return x_rotated, y_rotated


init_angle = math.atan2(vpyn[0], vpxn[0])


'Вектор скорости'
vel, = ax.plot([xn[0], xn[0] + vpxn[0]], [yn[0], yn[0] +
vpyn[0]], color='k', label = 'Вектор скорости')
vel_mod = (vpxn[0] ** 2 + vpyn[0] ** 2) ** 0.5
vel_arrX = np.array([-vel_mod * 0.1, 0.0, -vel_mod * 0.1],
dtype=float)
vel_arrY = np.array([vel_mod * 0.05, 0.0, -vel_mod * 0.05],
dtype=float)
vel_arrow_rotx, vel_arrow_roty = rotate(vel_arrX, vel_arrY,
init_angle)
vel_arrow, = ax.plot(xn[0] + vpxn[0] + vel_arrow_rotx, yn[0] +
vpyn[0] + vel_arrow_roty, color='k')

'Вектор тангенсального ускорения'
tan, = ax.plot([xn[0], xn[0] + wptn[0] *
math.cos(init_angle)],
                [yn[0], yn[0] + wptn[0] *
math.sin(init_angle)], color='y', label = 'Вектор
тангенсального ускорения')
tac_arrX = np.array([-wptn[0] * 0.1, 0.0, -wptn[0] * 0.1],
dtype=float)
tac_arrY = np.array([wptn[0] * 0.05, 0.0, -wptn[0] * 0.05],
dtype=float)
tac_arrow_rotx, tac_arrow_roty = rotate(tac_arrX, tac_arrY,
init_angle)
tac_arrow, = ax.plot(xn[0] + wpxn[0] + tac_arrow_rotx, yn[0] +
wpyn[0] + tac_arrow_roty, color='y')

'Вектор нормального ускорения'
nor, = ax.plot([xn[0], xn[0] + wpnn[0] * math.cos(init_angle -
math.pi / 2)],
                [yn[0], yn[0] + wpnn[0] * math.sin(init_angle
- math.pi / 2)], color='m', label = 'Вектор нормального
ускорения')
nac_arrX = np.array([-wpnn[0] * 0.1, 0.0, -wpnn[0] * 0.1],
dtype=float)
nac_arrY = np.array([wpnn[0] * 0.05, 0.0, -wpnn[0] * 0.05],
dtype=float)
nac_arrow_rotx, nac_arrow_roty = rotate(nac_arrX, nac_arrY,
init_angle - math.pi / 2)
nac_arrow, = ax.plot(xn[0] + wpnn[0] * math.cos(init_angle -
math.pi / 2) + nac_arrow_rotx,
```

```
                      yn[0] + wpnn[0] * math.sin(init_angle -
math.pi / 2) + nac_arrow_roty, color='m')

circle = plt.Circle((v0 * T[0], R), R, color='r', fill=False)
ax.add_artist(circle)



# Вывод легенды на график
ax.legend(
        ncol = 1,    #  количество столбцов
          facecolor = 'oldlace',    #  цвет области
          edgecolor = 'r',    #  цвет крайней линии
         )
ax.set(xlim=[20, 50], ylim=[-50, 50])


def cha(i):
    p.set_data(xn[i], yn[i])
    rot_angle = math.atan2(vpyn[i], vpxn[i])

    'Вектор скорости'
    vel.set_data([xn[i], xn[i] + vpxn[i]], [yn[i], yn[i] +
vpyn[i]])
    vel_mod = (vpxn[i] ** 2 + vpyn[i] ** 2) ** 0.5
    vel_arrX = np.array([-vel_mod * 0.1, 0.0, -vel_mod * 0.1],
dtype=float)
    vel_arrY = np.array([vel_mod * 0.05, 0.0, -vel_mod *
0.05], dtype=float)
    vel_arrow_rotx, vel_arrow_roty = rotate(vel_arrX,
vel_arrY, rot_angle)
    vel_arrow.set_data(xn[i] + vpxn[i] + vel_arrow_rotx, yn[i]
+ vpyn[i] + vel_arrow_roty)

    'Вектор тангенсального ускорения'
    tan.set_data([xn[i], xn[i] + wptn[i] *
math.cos(rot_angle)], [yn[i], yn[i] + wptn[i] *
math.sin(rot_angle)])
    tac_arrX = np.array([-wptn[i] * 0.1, 0.0, -wptn[i] * 0.1],
dtype=float)
    tac_arrY = np.array([wptn[i] * 0.05, 0.0, -wptn[i] *
0.05], dtype=float)
    tac_arrow_rotx, tac_arrow_roty = rotate(tac_arrX,
tac_arrY, math.atan2(vpyn[i], vpxn[i]))
    tac_arrow.set_data(xn[i] + wptn[i] * math.cos(rot_angle) +
tac_arrow_rotx,
                       yn[i] + wptn[i] * math.sin(rot_angle) +
tac_arrow_roty)

    'Вектор нормального ускорения'
    nor.set_data([xn[i], xn[i] + wpnn[i] * math.cos(rot_angle
- math.pi / 2)],
                 [yn[i], yn[i] + wpnn[i] *
math.sin(rot_angle - math.pi / 2)])
```

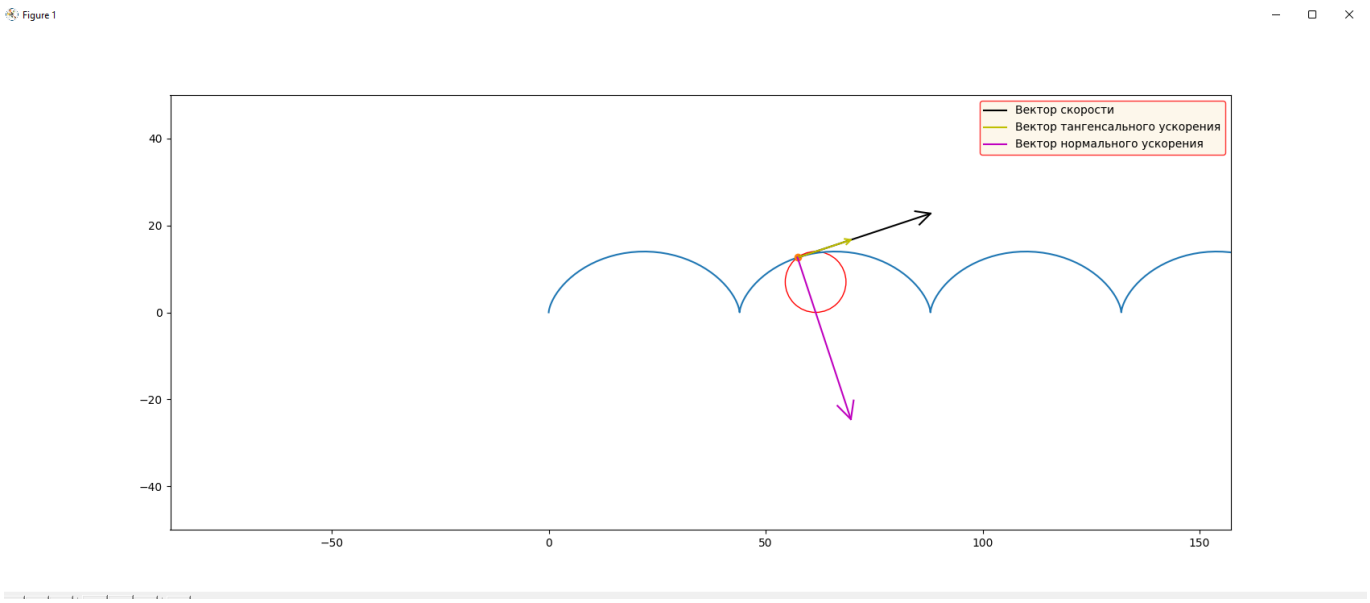
```
    nac_arrX = np.array([-wpnn[i] * 0.1, 0.0, -wpnn[i] * 0.1],
dtype=float)
    nac_arrY = np.array([wpnn[i] * 0.05, 0.0, -wpnn[i] *
0.05], dtype=float)
    nac_arrow_rotx, nac_arrow_roty = rotate(nac_arrX,
nac_arrY, math.atan2(vpyn[i], vpxn[i]) - math.pi / 2)
    nac_arrow.set_data(xn[i] + wpnn[i] * math.cos(rot_angle -
math.pi / 2) + nac_arrow_rotx,
                       yn[i] + wpnn[i] * math.sin(rot_angle -
math.pi / 2) + nac_arrow_roty)

    circle.set_center((v0 * T[i], R))


a = FuncAnimation(fig, cha, frames=len(T), interval=100)
plt.show()
```

Результат работы

a. R = 7.0, V0 = 10.0



b. R = 7.0, V0 = 17.0



Вывод

В ходе данной лабораторной работы я применил знания, полученные в начале курса для написания программы анимации точки на Python. Это мой первый опыт работы с этим языком и, как я выяснил он имеет ряд своих преимуществ. Стоит отметить, что он имеет большое количество библиотек с большим функционалом, которые позволяют решать задачи из различных сфер деятельности, что делает его очень универсальным. Например, в данной работе использованы такие библиотеки, как MatPlotLib, Sympy и Numpy.