Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дисциплина**

«Теоретическая механика и основы математического моделирования»

**Лабораторная работа №1**

Выполнил: студент группы М8О-206Б-20

Почечура Артемий Андреевич

Москва

2021

**Код программы**

import numpy as np

import matplotlib.pyplot as plt

import sympy as sp

import math as math

import matplotlib as matplotlib

from matplotlib.animation import FuncAnimation

matplotlib.use("TkAgg")

v0 = 10

R = 5

t = sp.Symbol('t')

rx = v0\*t-R\*sp.cos(v0\*t/R-math.pi/2)

ry = R + R\*sp.sin(v0\*t/R-math.pi/2)

vpx = sp.diff(rx, t)

vpy = sp.diff(ry, t)

vp = sp.sqrt(vpx\*vpx+vpy\*vpy)

wpx = sp.diff(vpx, t)

wpy = sp.diff(vpy, t)

Wt = sp.diff(vp, t)

W = sp.sqrt(wpx\*wpx+wpy\*wpy)

Wn = (abs(W\*W-Wt\*Wt))\*\*0.5

T = np.linspace(0, 20, 2001)

xn = np.zeros\_like(T)

yn = np.zeros\_like(T)

vx = np.zeros\_like(T)

vy = np.zeros\_like(T)

v = np.zeros\_like(T)

wt = np.zeros\_like(T)

wn = np.zeros\_like(T)

for i in range(len(T)):

xn[i] = sp.Subs(rx, t, T[i])

yn[i] = sp.Subs(ry, t, T[i])

vx[i] = sp.Subs(vpx, t, T[i])

vy[i] = sp.Subs(vpy, t, T[i])

v[i] = sp.Subs(vp, t, T[i])

wt[i] = sp.Subs(Wt, t, T[i])

wn[i] = sp.Subs(Wn, t, T[i])

fig = plt.figure()

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

ax.axis('equal')

ax.plot(xn, yn)

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

Phi = math.atan2(vy[0], vx[0])

VLine = ax.plot([xn[0], xn[0]+vx[0]], [yn[0], yn[0]+vy[0]], 'black')[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

V\_arrow\_x = np.array([-v[0]\*0.1, 0.0, -v[0]\*0.1], dtype=float)

V\_arrow\_y = np.array([v[0]\*0.05, 0.0, -v[0]\*0.05], dtype=float)

V\_arrow\_rotx, V\_arrow\_roty = rotate(V\_arrow\_x, V\_arrow\_y, Phi)

V\_arrow, = ax.plot(xn[0] + vx[0] + V\_arrow\_rotx, yn[0] + vy[0] + V\_arrow\_roty, color="black")

WTLine = ax.plot([xn[0], xn[0]+wt[0]\*math.cos(Phi)], [yn[0], yn[0]+wt[0]\*math.sin(Phi)], 'red')[0]

WT\_arrow\_x = np.array([-wt[0]\*0.1, 0.0, -wt[0]\*0.1], dtype=float)

WT\_arrow\_y = np.array([wt[0]\*0.05, 0.0, -wt[0]\*0.05], dtype=float)

WT\_arrow\_rotx, WT\_arrow\_roty = rotate(WT\_arrow\_x, WT\_arrow\_y, Phi)

WT\_arrow, = ax.plot(xn[0]+wt[0]\*math.cos(Phi) + WT\_arrow\_rotx, yn[0]+wt[0]\*math.sin(Phi) + WT\_arrow\_roty, color="red")

WNLine = ax.plot([xn[0], xn[0]+wn[0]\*math.cos(Phi-math.pi/2)], [yn[0], yn[0]+wn[0]\*math.sin(Phi-math.pi/2)], 'blue')[0]

WN\_arrow\_x = np.array([-wn[0]\*0.1, 0.0, -wn[0]\*0.1], dtype=float)

WN\_arrow\_y = np.array([wn[0]\*0.05, 0.0, -wn[0]\*0.05], dtype=float)

WN\_arrow\_rotx, WN\_arrow\_roty = rotate(WN\_arrow\_x, WN\_arrow\_y, Phi-math.pi/2)

WN\_arrow, = ax.plot(xn[0]+wn[0]\*math.cos(Phi-math.pi/2) + WN\_arrow\_rotx, yn[0]+wn[0]\*math.sin(Phi-math.pi/2) + WN\_arrow\_roty, color="blue")

def cha(i):

P.set\_data(xn[i], yn[i])

Phi = math.atan2(vy[i], vx[i])

VLine.set\_data([xn[i], xn[i]+vx[i]], [yn[i], yn[i]+vy[i]])

V\_arrow\_x = np.array([-v[i]\*0.1, 0.0, -v[i]\*0.1], dtype=float)

V\_arrow\_y = np.array([v[i]\*0.05, 0.0, -v[i]\*0.05], dtype=float)

V\_arrow\_rotx, V\_arrow\_roty = rotate(V\_arrow\_x, V\_arrow\_y, Phi)

V\_arrow.set\_data(xn[i] + vx[i] + V\_arrow\_rotx, yn[i] + vy[i] + V\_arrow\_roty)

WTLine.set\_data([xn[i], xn[i] + wt[i] \* math.cos(Phi)], [yn[i], yn[i] + wt[i] \* math.sin(Phi)])

WT\_arrow\_x = np.array([-wt[i]\*0.1, 0.0, -wt[i]\*0.1], dtype=float)

WT\_arrow\_y = np.array([wt[i]\*0.05, 0.0, -wt[i]\*0.05], dtype=float)

WT\_arrow\_rotx, WT\_arrow\_roty = rotate(WT\_arrow\_x, WT\_arrow\_y, Phi)

WT\_arrow.set\_data(xn[i] + wt[i] \* math.cos(Phi) + WT\_arrow\_rotx, yn[i] + wt[i] \* math.sin(Phi) + WT\_arrow\_roty)

WNLine.set\_data([xn[i], xn[i] + wn[i] \* math.cos(Phi-math.pi/2)], [yn[i], yn[i] + wn[i] \* math.sin(Phi-math.pi/2)])

WN\_arrow\_x = np.array([-wn[i]\*0.1, 0.0, -wn[i]\*0.1], dtype=float)

WN\_arrow\_y = np.array([wn[i]\*0.05, 0.0, -wn[i]\*0.05], dtype=float)

WN\_arrow\_rotx, WN\_arrow\_roty = rotate(WN\_arrow\_x, WN\_arrow\_y, Phi - math.pi / 2)

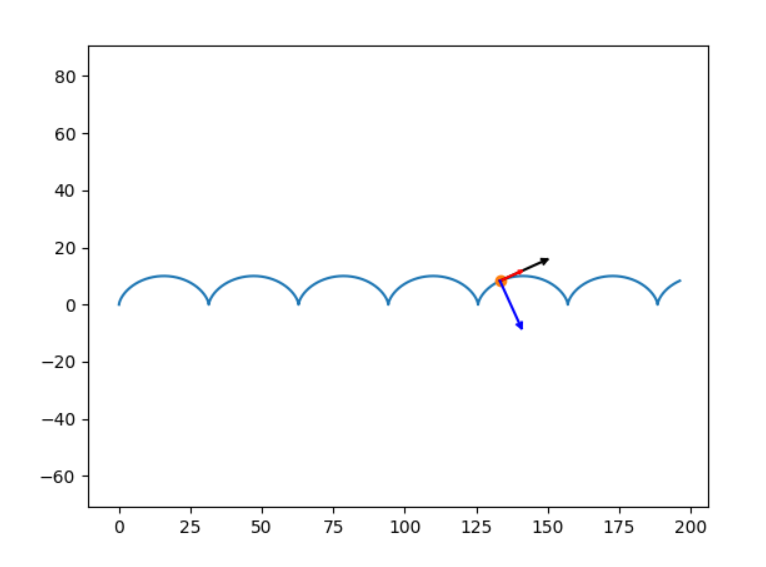
WN\_arrow.set\_data(xn[i] + wn[i] \* math.cos(Phi - math.pi / 2) + WN\_arrow\_rotx, yn[i] + wn[i] \* math.sin(Phi - math.pi / 2) + WN\_arrow\_roty)

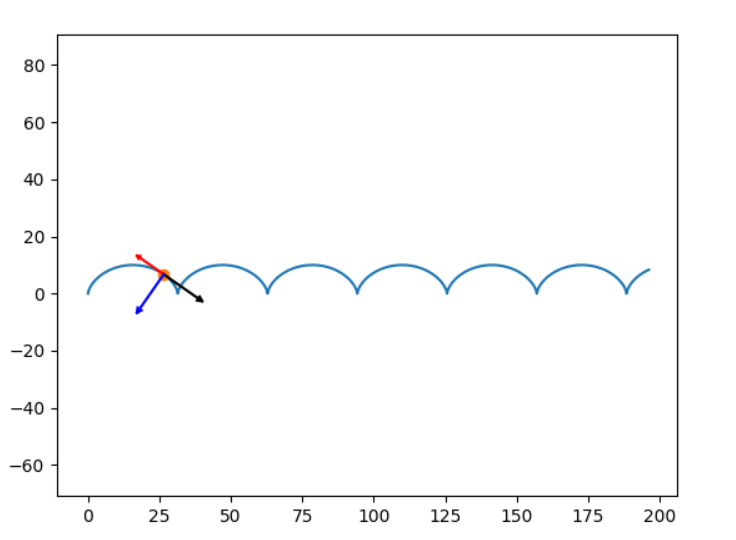
return [P]

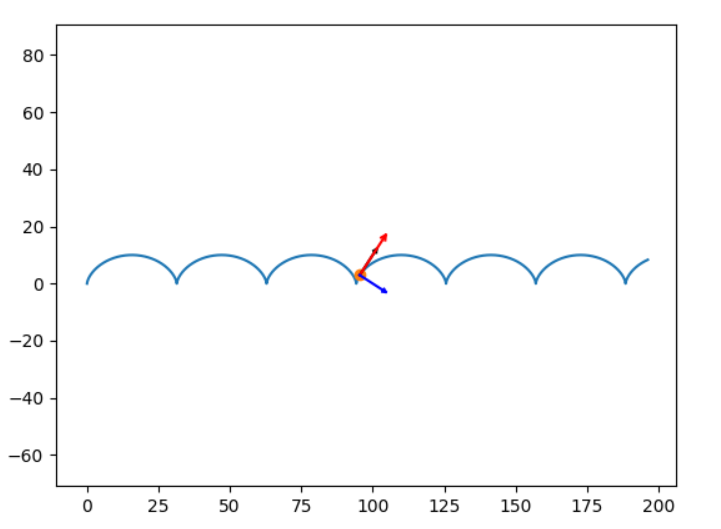
a = FuncAnimation(fig, cha, frames=len(T), interval=10)

plt.show()

**Скриншоты работы программы**

****

****

****