МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский Авиационный Институт» (Национальный Исследовательский Университет)

Институт: №8 «Компьютерные науки и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Лабораторная работа №1

по курсу «Теоретическая механика»

Ⅲ семестр

«Анимация точки»

|  |  |
| --- | --- |
| Группа | М8О-206Б-21 |
| Студент | Чистяков К.С. |
| Преподаватель | Чекина Е.А. |
| Оценка |  |
| Дата |  |

Москва, 2023

Оглавление

[*Задание:* 3](#_Toc126758437)

[*Закон движения точки*: 3](#_Toc126758438)

[*Текст программы:* 3](#_Toc126758439)

[*Результат работы программы:* 6](#_Toc126758440)

Вариант 24

# *Задание:*

Построить заданную траекторию и анимацию движения точки, а такж**е** отобразить стрелки скорости, нормального и тангенциальногоускорения. 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

import math

# Var 24: r(t) = 1 - sin(t), phi(t) = 5t

# Функция поворота на угол альфа

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

t = sp.Symbol('t')

phi = 5\*t

# Переход из полярных координат в Декартовы координаты

x = (1 - sp.sin(t))\*sp.cos(phi)

y = (1 - sp.sin(t))\*sp.sin(phi)

# Вычисление скорости по x и y, а также общей

Vx = sp.diff(x, t)

Vy = sp.diff(y, t)

Vmod = sp.sqrt(Vx\*Vx+Vy\*Vy)

# Вычисление ускорения по x и y, а также общего

Wx = sp.diff(Vx, t)

Wy = sp.diff(Vy, t)

Wmod = sp.sqrt(Wx\*Wx+Wy\*Wy)

# Вычисление тангенсального ускорения

cosa = (Vx\*Wx + Vy\*Wy)/(Vmod \* Wmod)

Wtaux = Vx/Vmod\*Wmod\*cosa

Wtauy = Vy/Vmod\*Wmod\*cosa

Wtau = sp.diff(Vmod, t)

# Вычисление радиуса кривизны

rho = (Vmod\*Vmod)/sp.sqrt(Wmod\*Wmod-Wtau\*Wtau)

Rhox = -sp.diff(y, t)\*(sp.diff(x, t)\*\*2 + sp.diff(y, t)\*\*2)/(sp.diff(x, t)\*sp.diff(y, t, 2) - sp.diff(x, t, 2)\*sp.diff(y, t))

Rhoy = sp.diff(x, t)\*(sp.diff(x, t)\*\*2 + sp.diff(y, t)\*\*2)/(sp.diff(x, t)\*sp.diff(y, t, 2) - sp.diff(x, t, 2)\*sp.diff(y, t))

# Вычисление нормального ускорения

sina = sp.sqrt(1-cosa\*\*2)

Wnx = Rhox/rho\*Wmod\*sina

Wny = Rhoy/rho\*Wmod\*sina

# Равномерное распределение по массиву тысячи чисел от 0 до 10.

T = np.linspace(0, 10, 1000)

X\_def = sp.lambdify(t, x)

Y\_def = sp.lambdify(t, y)

VX\_def = sp.lambdify(t, Vx)

VY\_def = sp.lambdify(t, Vy)

WY\_def = sp.lambdify(t, Wny)

WX\_def = sp.lambdify(t, Wnx)

Rho\_def = sp.lambdify(t, rho)

RhoY\_def = sp.lambdify(t, Rhoy)

Phi\_def = sp.lambdify(t, phi)

WtauY\_def = sp.lambdify(t, Wtauy)

WtauX\_def = sp.lambdify(t, Wtaux)

X = X\_def(T)

Y = Y\_def(T)

VX = VX\_def(T)

VY = VY\_def(T)

WY = WY\_def(T)

WX = WX\_def(T)

Rho = Rho\_def(T)

Phi = Phi\_def(T)

WtauY = WtauY\_def(T)

WtauX = WtauX\_def(T)

# Создать окно

fig = plt.figure()

# ax1 - окно с графиком

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

ax1.axis('equal')

ax1.set\_title("Модель движения точки")

ax1.set\_xlabel('Ось абцисс')

ax1.set\_ylabel('Ось ординат')

# Построение траектории

ax1.plot(X, Y)

# Построение точки

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

# Построение векторов тангенциального ускорения, нормального ускорения, скорости, радиуса-кривизны, радиус-вектора

VLine, = ax1.plot([X[0], X[0]+VX[0]], [Y[0], X[0]+VX[0]], 'r', label = 'Вектор скорости')

WLine, = ax1.plot([X[0], X[0]+WX[0]], [Y[0], Y[0]+WY[0]], 'g', label = 'Вектор норм. ускорения')

WtauLine, = ax1.plot([X[0], X[0] + WtauX[0]], [Y[0], Y[0] + WtauY[0]], 'y', label='Вектор танг. ускорения')

Rholine, = ax1.plot([X[0], X[0] + VY[0]\*Rho[0]/math.sqrt(VX[0]\*\*2 + VY[0]\*\*2)], [Y[0], Y[0] - VX[0] \* Rho[0]/math.sqrt(VX[0]\*\*2 + VY[0]\*\*2)], 'b', label = 'Вектор кривизны')

RLine, = ax1.plot([0, X[0]], [0, Y[0]], 'black', label = 'Радиус-вектор')

# Построение стрелочек на концах векторах

R = math.sqrt(math.pow(X[0], 2) + math.pow(Y[0], 2))

ArrowX = np.array([-0.2\*R, 0, -0.2\*R])

ArrowY = np.array([0.1\*R, 0, -0.1\*R])

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

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

WArrowX = np.array([-0.2\*R, 0, -0.2\*R])

WArrowY = np.array([0.1\*R, 0, -0.1\*R])

RWArrowX, RWArrowY = Rot2D(WArrowX, WArrowY, math.atan2(WY[0], WX[0]))

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

ArrowRx = np.array([-0.2\*R, 0, -0.2\*R])

ArrowRy = np.array([0.1\*R, 0, -0.1\*R])

RArrowRx, RArrowRy = Rot2D(ArrowRx, ArrowRy, math.atan2(Y[0], X[0]))

RArrow, = ax1.plot(RArrowRx + X[0], RArrowRy + Y[0], 'black')

ArrowRhoX = np.array([-0.2\*R, 0, -0.2\*R])

ArrowRhoY = np.array([0.1\*R, 0, -0.1\*R])

RArrowRhox, RArrowRhoy = Rot2D(ArrowRhoX, ArrowRhoY, math.atan2(- VX[0] \* Rho[0]/math.sqrt(VX[0]\*\*2 + VY[0]\*\*2), VY[0]\*Rho[0]/math.sqrt(VX[0]\*\*2 + VY[0]\*\*2)))

ArrowRho, = ax1.plot(RArrowRhox + X[0] + VY[0] \* Rho[0]/math.sqrt(VX[0]\*\*2 + VY[0]\*\*2), RArrowRhoy + Y[0] -VX[0] \* Rho[0]/math.sqrt(VX[0]\*\*2 + VY[0]\*\*2), 'b')

WtauArrowX = np.array([-0.2 \* R, 0, -0.2 \* R])

WtauArrowY = np.array([0.1 \* R, 0, -0.1 \* R])

RWtauArrowX, RWtauArrowY = Rot2D(WtauArrowX, WtauArrowY, math.atan2(WtauY[0], WtauX[0]))

WtauArrow, = ax1.plot(RWtauArrowX + X[0] + WtauX[0], RWtauArrowY + Y[0] + WtauY[0], 'y')

# Вывод легенды на график

ax1.legend(ncol = 2, facecolor = 'oldlace', edgecolor = 'g')

ax1.set(xlim=[-5, 5], ylim=[-5, 5])

# Функция для анимации

def anima(i):

RhoX = X[i] + VY[i] \* Rho[i]/math.sqrt(VX[i]\*\*2 + VY[i]\*\*2)

RhoY = Y[i] - VX[i] \* Rho[i]/math.sqrt(VX[i]\*\*2 + VY[i]\*\*2)

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

VLine.set\_data([X[i], X[i]+VX[i]], [Y[i], Y[i]+VY[i]])

Rholine.set\_data([X[i], RhoX], [Y[i], RhoY])

WLine.set\_data([X[i],X[i]+WX[i]],[Y[i],Y[i]+WY[i]])

RLine.set\_data([0, X[i]], [0, Y[i]])

WtauLine.set\_data([X[i], X[i] + WtauX[i]], [Y[i], Y[i] + WtauY[i]])

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

RWArrowX, RWArrowY = Rot2D(WArrowX, WArrowY, math.atan2(WY[i], WX[i]))

RArrowRx, RArrowRy = Rot2D(ArrowRx, ArrowRy, math.atan2(Y[i], X[i]))

RArrowRhox, RArrowRhoy = Rot2D(ArrowRhoX, ArrowRhoY, math.atan2(-Y[i] + RhoY, -X[i] + RhoX))

RWtauArrowX, RWtauArrowY = Rot2D(WtauArrowX, WtauArrowY, math.atan2(WtauY[i], WtauX[i]))

ArrowRho.set\_data(RArrowRhox + RhoX, RArrowRhoy + RhoY)

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

WArrow.set\_data(RWArrowX+X[i]+WX[i], RWArrowY+Y[i]+WY[i])

RArrow.set\_data(RArrowRx + X[i], RArrowRy + Y[i])

WtauArrow.set\_data(RWtauArrowX + X[i] + WtauX[i], RWtauArrowY + Y[i] + WtauY[i])

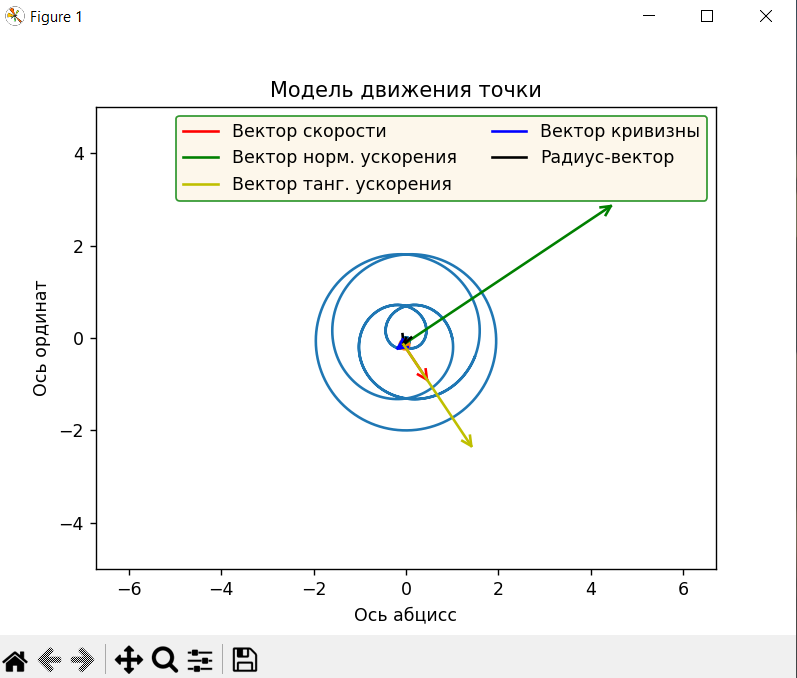
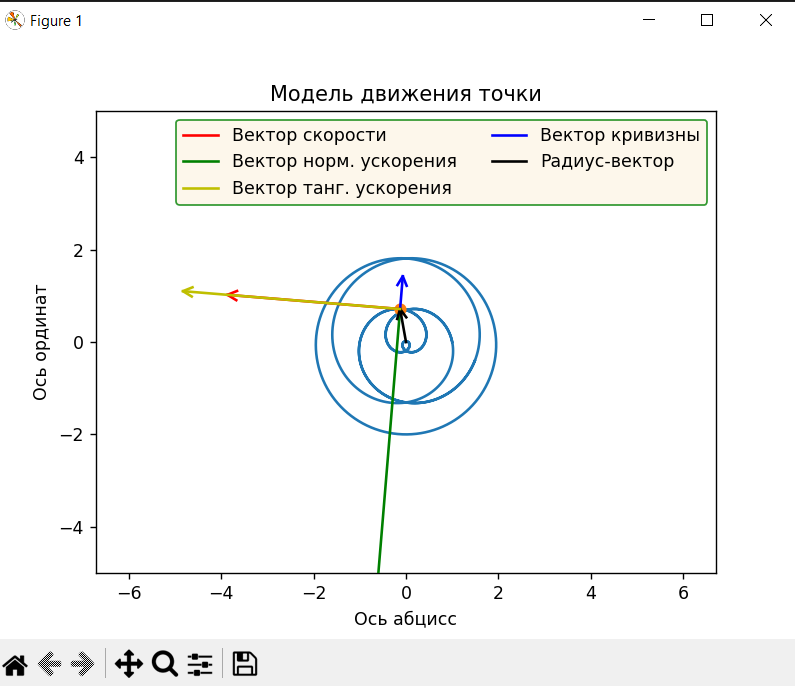
return P, VLine, Rholine, VArrow, WLine, WArrow, RLine, RArrow, ArrowRho, WtauLine, WtauArrow

# animation function

anim = FuncAnimation(fig, anima, frames=1000, interval=30, blit=True)

plt.show()

# *Результат работы программы:*

******