Московский авиационный институт

(национальный исследовательский университет)

Институт № 8 «Информационные технологии и прикладная математика»

**Лабораторная работа №1**

**по курсу «Теоретическая механика и компьютерное моделирование»**

**Анимация точки**

Выполнил студент группы М8О-207Б-20

Чекменев Вячеслав Алексеевич

Преподаватель: Чекина Евгения Алексеевна

Оценка:

Дата:

Москва, 2021

**Вариант № «3»**

**Задание:**

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

**Закон движения точки:**

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

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

**Текст программы**

import matplotlib.pyplot as plt

import numpy as np

import math

from matplotlib.animation import FuncAnimation

import sympy as sp

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

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([X\_[j], X[j]], [Y\_[j], Y[j]])

Vline4.set\_data([X[j], X[j] + (VY[j]) \* Ro[j]/((VY[j])\*\*2 +

(VX[j])\*\*2)\*\*0.5], [Y[j], Y[j] - (VX[j]) \*

Ro[j]/((VY[j])\*\*2 + (VX[j])\*\*2)\*\*0.5])

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, Vline4,

T = np.linspace(1, 15, 1000)

t = sp.Symbol('t')

R\_ = 4

Omega = 1

#functions

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)

Wx = sp.diff(Vx, t)

Vy = sp.diff(y, t)

Wy = sp.diff(Vy, t)

W\_ = sp.sqrt(Wx \* Wx + Wy \* Wy)

W\_t = sp.diff(sp.sqrt(Vx\*\*2 + Vy\*\*2),t)

ro = (Vx\*\*2 + Vy\*\*2)/sp.sqrt((Wx \* Wx + Wy \* Wy) - sp.diff(sp.sqrt(Vx\*\*2 + Vy\*\*2), t)\*\*2)

#filling arrays with zeros

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)

W = np.zeros\_like(T)

W\_T = np.zeros\_like(T)

Ro = np.zeros\_like(T)

X\_ = [0 for i in range(1000)]

Y\_ = [0 for i in range(1000)]

#printing functions

print("Point function of coordinates:")

print(" r(t) =", x)

print(" phi(t) =", x)

print(" x(t) =", x)

print(" y(t) =", y)

print("Vx(t) =", Vx)

print("Vy(t) =", Vy)

print("Wx(t) =", Wx)

print("Wy(t) =", Wy)

print("W(t) =", W\_)

print("W\_T(t) =", W\_t)

print("Ro(t) =", ro)

#filling arrays

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])

W[i] = sp.Subs(W\_, t, T[i])

W\_T[i] = sp.Subs(W\_t, t, T[i])

Ro[i] = sp.Subs(ro, t, T[i])

#drawing

fig = plt.figure()

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

ax1.axis('equal')

ax1.set(xlim=[-R\_, R\_], ylim=[-R\_, R\_])

ax1.plot(X, Y)

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

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

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

Vline3, = ax1.plot([X\_[0], X[0]], [Y\_[0], Y[0]], 'b') # vector of radius vector

Vline4, = ax1.plot([X[0], X[0] + (VY[0]) \* Ro[0]/((VY[0])\*\*2 +

(VX[0])\*\*2)\*\*0.5], [Y[0], Y[0] - (VX[0]) \* Ro[0]/

((VY[0])\*\*2 + (VX[0])\*\*2)\*\*0.5], 'm') # vector of radius of curvature

ArrowX = np.array([-0.1 \* R\_, 0, -0.1 \* R\_]) # arrow of speed

ArrowY = np.array([0.05 \* R\_, 0, -0.05 \* R\_])

ArrowWX = np.array([-R\_, 0, -R\_]) # arrow of acceleration

ArrowWY = np.array([R\_, 0, -R\_])

ArrowRX = np.array([-0.1 \* R\_, 0, -0.1 \* R\_]) # arrow of radius vector

ArrowRY = np.array([0.05 \* R\_, 0, -0.05 \* R\_])

# drawing an arrow at the end of a vector

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(Y[0], X[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')

anim = FuncAnimation(fig, anima, frames=1000, interval=40, blit=True)

plt.show()

**Результат работы программы:**

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

**Вывод программы:**

**Функция зависимости радиус вектора точки от времени:**

**r(t)** = (1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости угла вектора точки от времени:**

**phi(t)** = (1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости координаты Х точки от времени:**

**x(t)** = (1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости координаты У точки от времени:**

**y(t)** = (1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости скорости точки по координате Х от времени:**

**Vx(t)** = -(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + 18.0\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости скорости точки по координате У от времени:**

**Vy(t)** = (1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) + 18.0\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t)

**Функция зависимости ускорения точки по координате Х от времени:**

**Wx(t)** = (5.76\*(0.520833333333333 - sin(12\*t))\*\*2)\*(-1.5\*sin(12\*t) - 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 36.0\*(1.25 - 2.4\*sin(12\*t))\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 28.8\*(-1.5\*sin(12\*t) - 1)\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 216.0\*sin(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости ускорения точки по координате У от времени:**

**Wy(t)** = -5.76\*(0.520833333333333 - sin(12\*t))\*\*2\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + 36.0\*(1.25 - 2.4\*sin(12\*t))\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 28.8\*(1.5\*sin(12\*t) + 1)\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 216.0\*sin(12\*t)\*sin(1.25\*t + 0.2\*cos(12\*t))

**Функция зависимости ускорения точки от времени:**

**W(t)** = 216.0\*sqrt((0.0266666666666667\*(0.520833333333333 - sin(12\*t))\*\*2\*(-1.5\*sin(12\*t) - 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 0.166666666666667\*(1.25 - 2.4\*sin(12\*t))\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 0.133333333333333\*(-1.5\*sin(12\*t) - 1)\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - sin(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*\*2 + (-0.0266666666666667\*(0.520833333333333 - sin(12\*t))\*\*2\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + 0.166666666666667\*(1.25 - 2.4\*sin(12\*t))\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 0.133333333333333\*(1.5\*sin(12\*t) + 1)\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - sin(12\*t)\*sin(1.25\*t + 0.2\*cos(12\*t)))\*\*2)

**Функция зависимости тангенциального ускорения точки от времени:**

**W\_T(t)** = 18.0\*((-0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*(2\*(5.76\*(0.520833333333333 - sin(12\*t))\*\*2)\*(-0.0833333333333333\*sin(12\*t) - 0.0555555555555556)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 4.0\*(1.25 - 2.4\*sin(12\*t))\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 57.6\*(-0.0833333333333333\*sin(12\*t) - 0.0555555555555556)\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 24\*sin(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))/2 + (0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) + sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t))\*(-2\*5.76\*(0.520833333333333 - sin(12\*t))\*\*2\*(0.0833333333333333\*sin(12\*t) + 0.0555555555555556)\*sin(1.25\*t + 0.2\*cos(12\*t)) + 4.0\*(1.25 - 2.4\*sin(12\*t))\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 57.6\*(0.0833333333333333\*sin(12\*t) + 0.0555555555555556)\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 24\*sin(12\*t)\*sin(1.25\*t + 0.2\*cos(12\*t)))/2)/sqrt((-0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*\*2 + (0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) + sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t))\*\*2)

**Функция зависимости радиуса кривизны точки от времени:**

**Ro(t)** = 0.00462962962962963\*(324.0\*(-0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*\*2 + 324.0\*(0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) + sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t))\*\*2)/sqrt(-0.00694444444444444\*((-0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*(2\*(5.76\*(0.520833333333333 - sin(12\*t))\*\*2)\*(-0.0833333333333333\*sin(12\*t) - 0.0555555555555556)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 4.0\*(1.25 - 2.4\*sin(12\*t))\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 57.6\*(-0.0833333333333333\*sin(12\*t) - 0.0555555555555556)\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 24\*sin(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))/2 + (0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) + sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t))\*(-2\*5.76\*(0.520833333333333 - sin(12\*t))\*\*2\*(0.0833333333333333\*sin(12\*t) + 0.0555555555555556)\*sin(1.25\*t + 0.2\*cos(12\*t)) + 4.0\*(1.25 - 2.4\*sin(12\*t))\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 57.6\*(0.0833333333333333\*sin(12\*t) + 0.0555555555555556)\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 24\*sin(12\*t)\*sin(1.25\*t + 0.2\*cos(12\*t)))/2)\*\*2/((-0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*\*2 + (0.0555555555555556\*(1.25 - 2.4\*sin(12\*t))\*(1.5\*sin(12\*t) + 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) + sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t))\*\*2) + (0.0266666666666667\*(0.520833333333333 - sin(12\*t))\*\*2\*(-1.5\*sin(12\*t) - 1)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 0.166666666666667\*(1.25 - 2.4\*sin(12\*t))\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - 0.133333333333333\*(-1.5\*sin(12\*t) - 1)\*sin(1.25\*t + 0.2\*cos(12\*t))\*cos(12\*t) - sin(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)))\*\*2 + (-0.0266666666666667\*(0.520833333333333 - sin(12\*t))\*\*2\*(1.5\*sin(12\*t) + 1)\*sin(1.25\*t + 0.2\*cos(12\*t)) + 0.166666666666667\*(1.25 - 2.4\*sin(12\*t))\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - 0.133333333333333\*(1.5\*sin(12\*t) + 1)\*cos(12\*t)\*cos(1.25\*t + 0.2\*cos(12\*t)) - sin(12\*t)\*sin(1.25\*t + 0.2\*cos(12\*t)))\*\*2)