ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«АНИМАЦИЯ ТОЧКИ»**

**ПО ДИСЦИПЛИНЕ «ТЕОРЕТИЧЕСКАЯ МЕХАНИКА И ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ»**

**ВАРИАНТ ЗАДАНИЯ №3**

Выполнил студент группы М8О-208Б-23

Гаврилов Никита Валерьевич \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Ст. преп. каф. 802 Волков Е.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2024

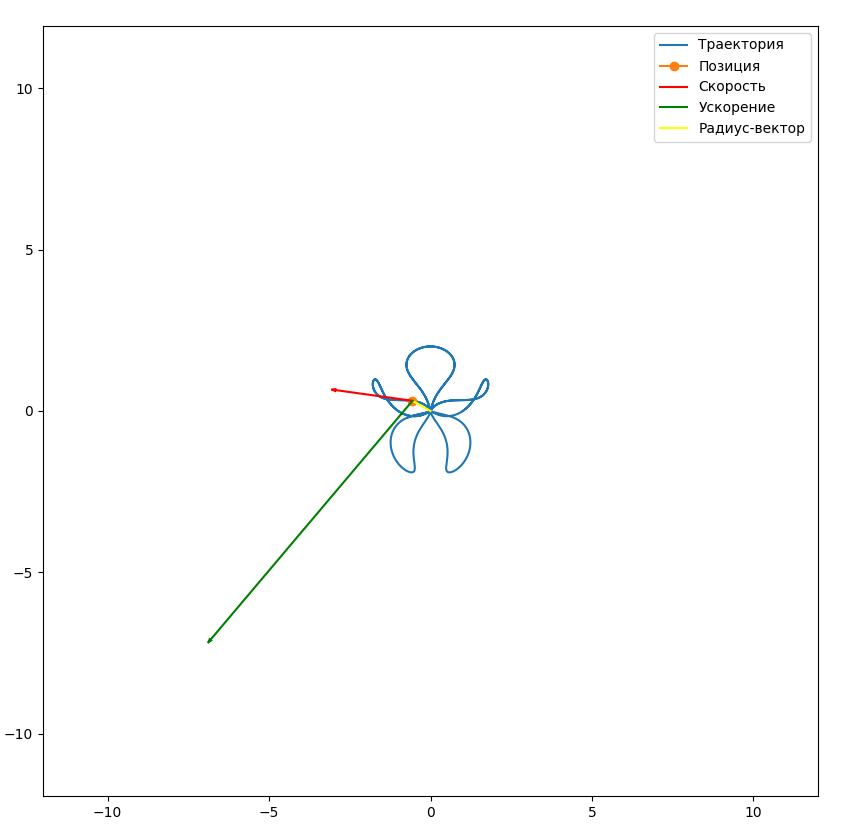
**Задание:** Построить заданную траекторию и анимацию движения точки, а также отобразить стрелки радиус-вектора, скорости и ускорения. Построить радиус кривизны траектории.

**Закон движения точки:**

r(t) = 1 + sin(5t)

φ(t) = t + 0.3 \* sin(30t)

**Рисунок получившейся физической модели:**



**Текст программы:**

*import* numpy *as* np

*import* matplotlib.pyplot *as* plt

*from* matplotlib.animation *import* FuncAnimation

*import* sympy *as* sp

*# Функция для поворота координат в 2D.*

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

*# Определение символических переменных и выражений*

t *=* sp.Symbol('t')

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

phi *=* t *+* 0.3 *\** sp.sin(8 *\** t)

*# Вычисление координат*

x *=* r *\** sp.cos(phi)

y *=* r *\** sp.sin(phi)

*# Вычисление первой производной*

Vx *=* sp.diff(x, t)

Vy *=* sp.diff(y, t)

*# Вычисление второй производной*

Ax *=* sp.diff(Vx, t)

Ay *=* sp.diff(Vy, t)

*# Преобразование символьных выражений в функции*

F\_x *=* sp.lambdify(t, x)

F\_y *=* sp.lambdify(t, y)

F\_Vx *=* sp.lambdify(t, Vx)

F\_Vy *=* sp.lambdify(t, Vy)

F\_Ax *=* sp.lambdify(t, Ax)

F\_Ay *=* sp.lambdify(t, Ay)

*# Генерация значений времени*

t\_vals *=* np.linspace(0, 10, 1001)

*# Вычисление значений координат, скорости и ускорения*

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)

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

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

*# Настройка графика*

fig *=* plt.figure(figsize*=*[10, 10])

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

ax.axis('equal')

ax.set(xlim*=*[*-*12, 12], ylim*=*[*-*12, 12])

*# Масштабные коэффициенты для стрелок скорости и ускорения*

k\_V *=* 0.5

k\_A *=* 0.5

*# Отображение траектории*

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

*# Инициализация точек и стрелок*

P, *=* ax.plot(x\_vals[0], y\_vals[0], marker*=*'o', label*=*'Позиция')

V\_line, *=* ax.plot([x\_vals[0], x\_vals[0] *+* k\_V *\** Vx\_vals[0]],

[y\_vals[0], y\_vals[0] *+* k\_V *\** Vy\_vals[0]],

'r-', label*=*'Скорость')

a *=* 0.1

b *=* 0.03

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

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

RotX\_V, RotY\_V *=* Rot2D(x\_arr, y\_arr, Alpha\_V[0])

V\_Arrow, *=* ax.plot(x\_vals[0] *+* k\_V *\** Vx\_vals[0] *+* RotX\_V,

y\_vals[0] *+* k\_V *\** Vy\_vals[0] *+* RotY\_V,

color*=*'r')

*# Добавление стрелки ускорения*

A\_line, *=* ax.plot([x\_vals[0], x\_vals[0] *+* k\_A *\** Ax\_vals[0]],

[y\_vals[0], y\_vals[0] *+* k\_A *\** Ay\_vals[0]],

'g-', label*=*'Ускорение')

RotX\_A, RotY\_A *=* Rot2D(x\_arr, y\_arr, Alpha\_A[0])

A\_Arrow, *=* ax.plot(x\_vals[0] *+* k\_A *\** Ax\_vals[0] *+* RotX\_A,

y\_vals[0] *+* k\_A *\** Ay\_vals[0] *+* RotY\_A,

color*=*'g')

R\_line, *=* ax.plot([0, x\_vals[0]], [0, y\_vals[0]], 'yellow', label*=*'Радиус-вектор')

*#Функция для анимации*

def anima(i):

*# Обновление позиции*

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

*# Обновление радиус-вектора*

R\_line.set\_data([0, float(x\_vals[i])], [0, float(y\_vals[i])])

*# Обновление стрелки скорости*

V\_line.set\_data([float(x\_vals[i]), float(x\_vals[i]) *+* k\_V *\** float(Vx\_vals[i])],

[float(y\_vals[i]), float(y\_vals[i]) *+* k\_V *\** float(Vy\_vals[i])])

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

V\_Arrow.set\_data(float(x\_vals[i]) *+* k\_V *\** float(Vx\_vals[i]) *+* RotX\_V,

float(y\_vals[i]) *+* k\_V *\** float(Vy\_vals[i]) *+* RotY\_V)

*# Обновление стрелки ускорения*

A\_line.set\_data([float(x\_vals[i]), float(x\_vals[i]) *+* k\_A *\** float(Ax\_vals[i])],

[float(y\_vals[i]), float(y\_vals[i]) *+* k\_A *\** float(Ay\_vals[i])])

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

A\_Arrow.set\_data(float(x\_vals[i]) *+* k\_A *\** float(Ax\_vals[i]) *+* RotX\_A,

float(y\_vals[i]) *+* k\_A *\** float(Ay\_vals[i]) *+* RotY\_A)

*return* [P, R\_line, V\_line, V\_Arrow, A\_line, A\_Arrow]

ax.legend()

anim *=* FuncAnimation(fig, anima, frames*=*len(t\_vals), interval*=*20, blit*=*True)

plt.show()

**Вывод:**

Я успешно выполнил лабораторную работу по теоретической механике, используя язык программирования Python и библиотеки matplotlib, numpy и sympy. В ходе выполнения задания я построил заданную траекторию движения точки и создал анимацию, демонстрирующую это движение. Для каждого момента времени я визуализировал векторы скорости, ускорения и радиус-вектора. что позволило наглядно представить динамику движения.

Эта лабораторная работа значительно углубила мое понимание темы движения точки. В результате, я не только выполнил поставленные задачи, но и получил опыт в применении программирования для решения задач теоретической механики.