ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

ОТЧЕТ

О ВЫПОЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

«ДИНАМИКА СИСТЕМЫ»

ПО ДИСЦИПЛИНЕ «ОСНОВЫ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ СИСТЕМ»

ВАРИАНТ ЗАДАНИЯ №41

Выполнил(а) студент группы М8О-208Б-23

Соловьева Надежда Сергеевна ______________________________

подпись, дата

Проверил и принял

Ст. преп. каф. 802 Волков Е.В. ______________________________

подпись, дата

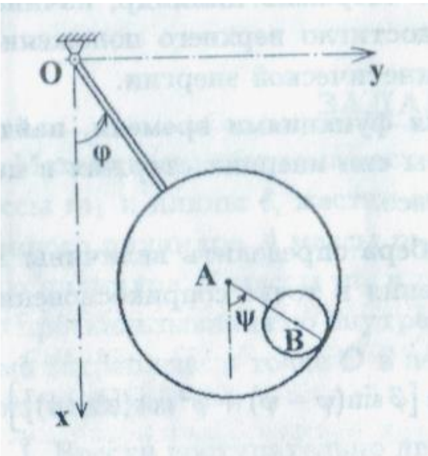
с оценкой ______________________________

Москва, 2024

Задание:

Проинтегрировать систему дифференциальных уравнений движения системы с двумя степенями свободы с помощью средств Python. Построить анимацию движения системы, а также графики законов движения системы и указанных в задании реакций для разных случаев системы.

Механическая система:



$$R_{Ox} = -[(m_1/2)\ell + (m_2+m_3)(R+\ell)]\left(\ddot\varphi\sin\varphi + \dot\varphi^2\cos\varphi\right) -$$
$$- m_3(R-r)\left(\ddot\psi\sin\psi + \dot\psi^2\cos\psi\right) - (m_1+m_2+m_3)g,$$

$$R_{Oy} = [(m_1/2)\ell + (m_2+m_3)(R+\ell)]\left(\ddot\varphi\cos\varphi - \dot\varphi^2\sin\varphi\right) +$$
$$+ m_3(R-r)\left(\ddot\psi\cos\psi - \dot\psi^2\sin\psi\right).$$

$$\{(m_1/3)\ell^2 + (m_2+m_3)(R+\ell)^2 + [m_2 + (m_3/2)]R^2\}\ddot\varphi +$$
$$+ m_3(R-r)\left\{(R+\ell)\left[\ddot\psi\cos(\varphi-\psi) + \dot\psi^2\sin(\varphi-\psi)\right] - (R/2)\ddot\psi\right\} =$$
$$= -[(m_1/2)\ell + (m_2+m_3)(R+\ell)]g\sin\varphi, \qquad g\sin\psi +$$
$$+ (R+\ell)\left[\ddot\varphi\cos(\varphi-\psi) - \dot\varphi^2\sin(\varphi-\psi)\right] - (R/2)\ddot\varphi + (3/2)(R-r)\ddot\psi = 0.$$

12. Задавая численные значения параметров и начальные условия: $m_1 = 2$ кг, $m_2 = 5$ кг, $m_3 = 3$ кг, $\ell = 1$ м, $R = 0,5$ м, $r = 0,1$ м; $t_0 = 0$, $\varphi_0 = \pi/6$, $\psi_0 = \pi/3$, $\dot\varphi_0 = 0$, $\dot\psi_0 = 0$, составить программу решения системы дифференциальных уравнений и на ЭВМ построить зависимости $\varphi(t)$, $\psi(t)$, $R_{Ox}(t)$, $R_{Oy}(t)$.

Текст программы:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
from scipy.integrate import solve_ivp

# Константы
R = 0.5  # Радиус цилиндра A
r = 0.1  # Радиус цилиндра B
l = 1.0  # Длина стержня
m1 = 2.0  # Масса стержня
m2 = 5.0  # Масса цилиндра A
m3 = 3.0  # Масса цилиндра B
g = 9.81  # Ускорение свободного падения

# Уравнения движения с использованием метода Крамера
def equations(t, y):
    phi, phi_dot, psi, psi_dot = y
```

```
    # Коэффициенты уравнений
    A11 = (m1/3)*l**2 + (m2 + m3)*(R + l)**2 + (m2 + m3/2)*R**2
    A12 = m3*(R - r)*(R + l)*np.cos(phi - psi)
    A21 = (R + l)*np.cos(phi - psi) - R/2
    A22 = (3/2)*(R - r)

    B1 = -((m1/2)*l + (m2 + m3)*(R + l)) * g * np.sin(phi) - m3*(R - r)*(R +
l)*phi_dot**2*np.sin(phi - psi)
    B2 = -g*np.sin(psi) + (R + l)*phi_dot**2*np.sin(phi - psi)

    # Метод Крамера для решения системы
    det = A11 * A22 - A12 * A21
    phi_ddot = (B1 * A22 - B2 * A12) / det
    psi_ddot = (B2 * A11 - B1 * A21) / det

    return [phi_dot, phi_ddot, psi_dot, psi_ddot]

# Начальные условия
phi0 = np.pi / 6
psi0 = np.pi / 3
phi_dot0 = 0.0
psi_dot0 = 0.0
y0 = [phi0, phi_dot0, psi0, psi_dot0]

# Временной интервал
t_span = (0, 10)
t_eval = np.linspace(*t_span, 600)

# Решение системы уравнений
sol = solve_ivp(equations, t_span, y0, t_eval=t_eval)

# Вычисление R_Ox и R_Oy
phi_ddot_values = np.gradient(sol.y[1], t_eval)
psi_ddot_values = np.gradient(sol.y[3], t_eval)

R_Ox = -((m1/2)*l + (m2 + m3)*(R + l)) * (phi_ddot_values * np.sin(sol.y[0]) + sol.y[1]**2 *
np.cos(sol.y[0])) \
      - m3*(R - r)*(psi_ddot_values * np.sin(sol.y[2]) + sol.y[3]**2 * np.cos(sol.y[2])) \
      - (m1 + m2 + m3)*g

R_Oy = ((m1/2)*l + (m2 + m3)*(R + l)) * (phi_ddot_values * np.cos(sol.y[0]) - sol.y[1]**2 *
np.sin(sol.y[0])) \
      + m3*(R - r)*(psi_ddot_values * np.cos(sol.y[2]) - sol.y[3]**2 * np.sin(sol.y[2]))

# Анимация
def update(frame):
    plt.cla()
    phi = sol.y[0, frame]
    psi = sol.y[2, frame]

    # Координаты стержня
    rod_x = [0, l * np.sin(phi)]
```

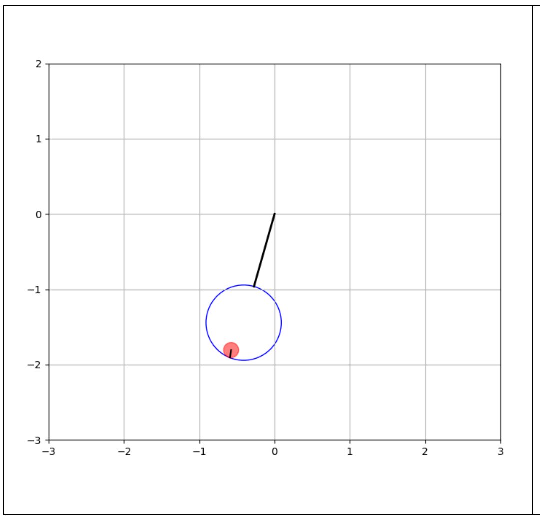
```
    rod_y = [0, -l * np.cos(phi)]

    # Координаты центра цилиндра A
    center_A_x = (l + R) * np.sin(phi)
    center_A_y = -(l + R) * np.cos(phi)

    # Координаты центра цилиндра B
    center_B_x = center_A_x + (R-r) * np.sin(psi)
    center_B_y = center_A_y - (R-r) * np.cos(psi)

    ## Обновление угла для линии-индикатора вращения цилиндра B
    # Угол вращения цилиндра B по его окружности
    theta_B = (R - r) * psi / r

    # Добавление линии-индикатора вращения для цилиндра B
    indicator_x = center_B_x + r * np.cos(theta_B)
    indicator_y = center_B_y + r * np.sin(theta_B)

    # Отрисовка
    plt.plot(rod_x, rod_y, 'k-', linewidth=2)  # Стержень

    # Отрисовка цилиндра A
    circle_A = plt.Circle((center_A_x, center_A_y), R, fill=False, color='blue')
    plt.gca().add_patch(circle_A)

    # Отрисовка цилиндра B с индикатором вращения
    circle_B = plt.Circle((center_B_x, center_B_y), r, color='red', alpha=0.5)
    plt.gca().add_patch(circle_B)
    plt.plot([center_B_x, indicator_x], [center_B_y, indicator_y], 'k-')  # Индикатор вращения

    # Установка пределов графика и соотношения сторон
    plt.xlim(-3, 3)
    plt.ylim(-3, 2)
    plt.gca().set_aspect('equal')
    plt.grid(True)

# Создание фигуры
fig = plt.figure(figsize=(8, 8))

# Создание анимации
anim = FuncAnimation(fig, update, frames=len(t_eval), interval=50, repeat=True)

plt.show()

# Построение графиков
plt.figure(figsize=(12, 8))
plt.subplot(2, 2, 1)
plt.plot(sol.t, sol.y[0], label='phi(t)')
plt.xlabel('t')
plt.ylabel('phi')
plt.grid(True)
plt.legend()
```

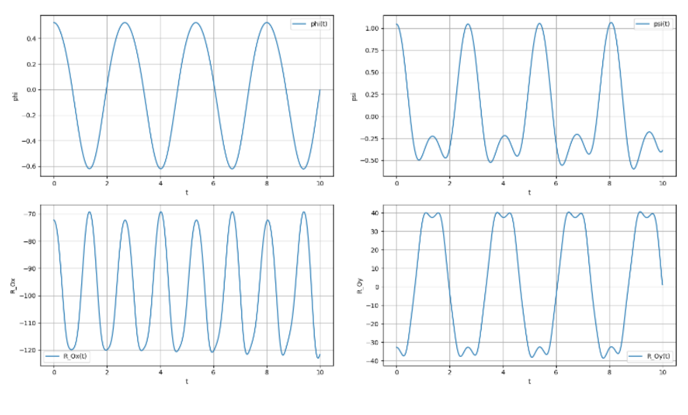
```
plt.subplot(2, 2, 2)
plt.plot(sol.t, sol.y[2], label='psi(t)')
plt.xlabel('t')
plt.ylabel('psi')
plt.grid(True)
plt.legend()

plt.subplot(2, 2, 3)
plt.plot(sol.t, R_Ox, label='R_Ox(t)')
plt.xlabel('t')
plt.ylabel('R_Ox')
plt.grid(True)
plt.legend()

plt.subplot(2, 2, 4)
plt.plot(sol.t, R_Oy, label='R_Oy(t)')
plt.xlabel('t')
plt.ylabel('R_Oy')
plt.grid(True)
plt.legend()

plt.tight_layout()
plt.show()
```

Результаты работы программы:

Случай 1

Дано: m1 = 2.0, m2 = 5.0, m3 = 3.0, R = 0.5, r = 0.1, l = 1.0, g = 9.81

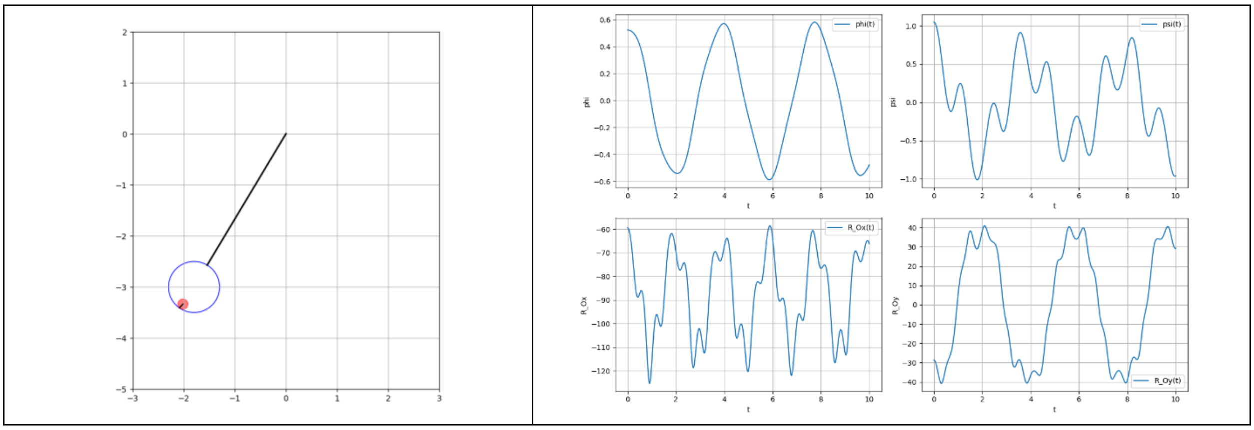




- $\phi(t)$ и $\psi(t)$ изменяются в зависимости от начальных условий и параметров масс и размеров.
- Поскольку массы m2 и m3 значительны, это приводит к более сложным колебаниям в системе.
- ROx(t) и ROy(t) показывают реакции на закрепление, которые будут значительными из-за веса системы.

Случай 2

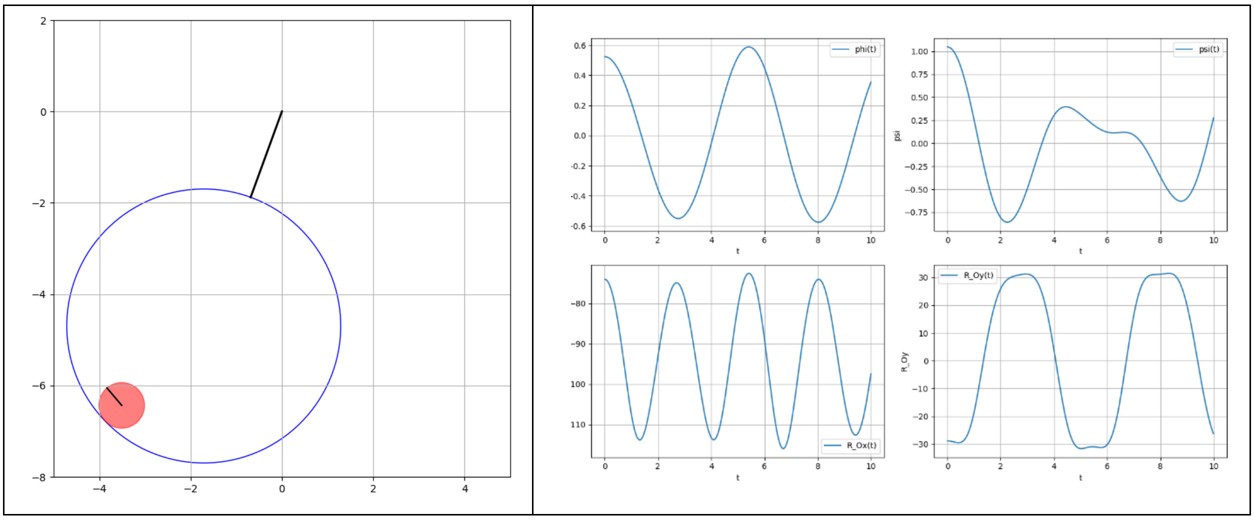
Дано: m1 = 2.0, m2 = 2.0, m3 = 5.0, R = 0.5, r = 0.1, l = 5.0, g = 9.81



- Увеличенная длина стержня l=5.0 приводит к изменению динамики ϕ(t) и ψ(t), увеличивая амплитуду колебаний.
- Большая масса m3 в сравнении с m2 может приводит к более выраженным колебаниям внутреннего цилиндра.
- Реакции ROx(t) и ROy(t) могут менее симметричны из-за разницы в распределении масс.

Случай 3

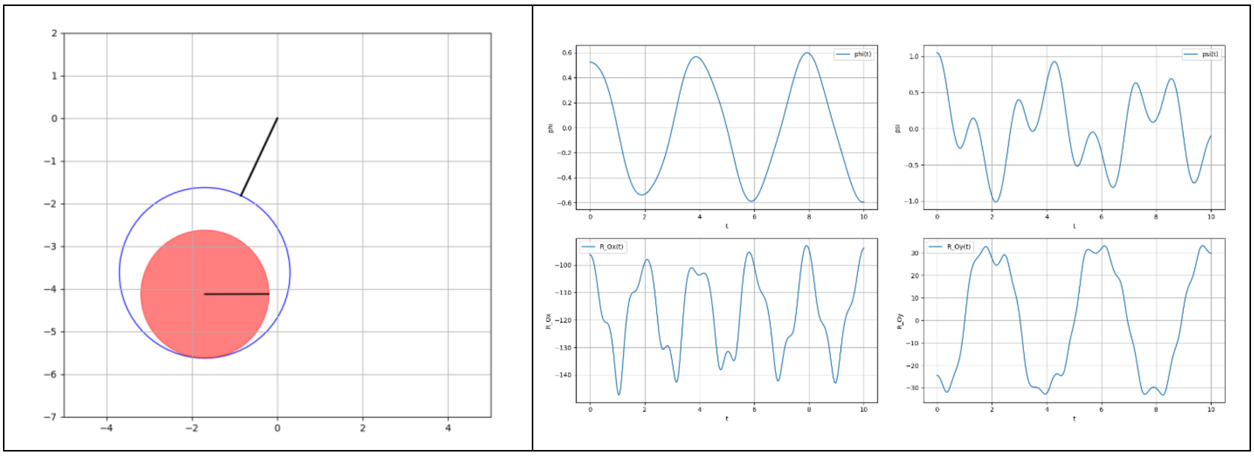
Дано: m1 = 0.5, m2 = 8.0, m3 = 1.0, R = 3.0, r = 0.5, l = 2.0, g = 9.81



- Большой радиус R=3.0 и большая масса m2 делают внешний цилиндр доминирующим в системе.
- Большой радиус внешнего цилиндра и его масса увеличивают момент инерции системы. Это приводит к тому, что система колеблется медленнее. ψ(t) могут демонстрировать движение с меньшей амплитудой за счёт значительной инерции. Внутренний цилиндр должен катиться по более крупной окружности, что увеличивает путь, который он проходит при каждом колебании, тем самым влияя на частоту и инерцию движения.
- Реакции ROx(t) и ROy(t) будут высокими из-за больших размеров и массы внешнего цилиндра.

Случай 4

Дано: m1 = 8.0, m2 = 1.0, m3 = 3.0, R = 2.0, r = 1.5, l = 2.0, g = 9.81



- Большая масса стержня m1 делает его основным фактором динамики ϕ(t).
- ψ(t) будет зависеть от взаимодействия между цилиндрами, но с меньшей амплитудой из-за меньшего m2. Радиусы внешнего и внутреннего цилиндров относительно близки по величине. Это означает, что внутренний цилиндр занимает значительную часть внутреннего пространства внешнего цилиндра, что ограничивает его движение и уменьшает амплитуду колебаний.
- Реакции ROx(t) и ROy(t) будут значительными из-за веса стержня и его влияния на систему.

Вывод:

В процессе выполнения работы я смогла реализовать интеграцию системы дифференциальных уравнений с помощью Python, построила анимацию и графики движения для различных случаев системы. Это позволило проанализировать динамическое поведение системы при различных параметрах, что отражено в представленных результатах. В целом, данные случаи показывают важность понимания и учета взаимодействия всех параметров системы для точного предсказания и анализа ее динамического поведения.