ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

**ОТЧЕТ**

**О ВЫПЛОНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**«ДИНАМИКА СИСТЕМЫ»**

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

**ВАРИАНТ ЗАДАНИЯ 8**

Выполнил(а) студент группы М8О-203Б-22

Касумова Наида Рашидовна\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата

Проверил и принял

Авдюшкин А.Н.\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

подпись, дата

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

Москва, 2023

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.

Изображение выглядит как текст, Шрифт, белый, документ

Автоматически созданное описаниеИзображение выглядит как текст, бумага, книга, письмо

Автоматически созданное описаниеВариант 8:

Код программы:

from scipy.integrate import odeint

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.animation import FuncAnimation

t = np.linspace(0, 10, 1001)

def EqOfMovement(y, t, M, m, r, c, g):

    # y[0, 1, 2, 3] = fi, phi, fi', phi'

    # dy[0, 1, 2, 3] = fi', phi', fi'', phi''

    dy = np.zeros\_like(y)

    dy[0] = y[2]

    dy[1] = y[3]

    a11 = 1

    a12 = np.cos(y[0])

    a21 = np.cos(y[0])

    a22 = 1 + 2 \* (M / m)

    b1 = -2 \* (c / m) \* (1 - np.cos((y[0] + y[1]) / 2)) \* np.sin((y[0] + y[1]) / 2) - (g / r) \* np.sin(y[0])

    b2 = y[2]\*\*2 \* np.sin(y[0]) – 2 \* (c / m) \* (1 - np.cos((y[0] + y[1]) / 2)) \* np.sin((y[0] + y[1]) / 2)

    dy[2] = (b1 \* a22 - b2 \* a12) / (a11 \* a22 - a12 \* a21)

    dy[3] = (b2 \* a11 - b1 \* a21) / (a11 \* a22 - a12 \* a21)

    return dy

r = 1

M = 1

m = 2

c = 40

g = 9.81

fi0 = np.pi/3

phi0 = 0

dfi0 = 0

dphi0 = 0

y0 = [fi0, phi0, dfi0, dphi0]

Y = odeint(EqOfMovement, y0, t, (M, m, r, c, g))

fi = Y[:, 0]

phi = Y[:, 1]

dfi = Y[:, 2]

dphi = Y[:, 3]

ddfi = np.array([EqOfMovement(yi, ti, M, m, r, c, g)[2] for yi, ti in zip(Y, t)])

ddphi = np.array([EqOfMovement(yi, ti, M, m, r, c, g)[3] for yi, ti in zip(Y, t)])

N = m \* (g \* np.cos(fi) + r \* (dfi\*\*2 - ddphi\*\*2 \* np.sin(fi))) + 2 \* r \* c \* np.cos((fi + phi) / 2) \* (1 - np.cos((fi + phi) / 2))

fig\_for\_graphs = plt.figure(figsize = [13, 7])  # построим их графики

ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 2, 1)

ax\_for\_graphs.plot(t, fi, color = 'blue')

ax\_for\_graphs.set\_title("fi(t)")

ax\_for\_graphs.set(xlim = [0, 10])

ax\_for\_graphs.grid(True)

ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2, 2, 2)

ax\_for\_graphs.plot(t, phi, color = 'red')

ax\_for\_graphs.set\_title('phi(t)')

ax\_for\_graphs.set(xlim = [0, 10])

ax\_for\_graphs.grid(True)

ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2,2,3)

ax\_for\_graphs.plot(t,dfi,color = 'green')

ax\_for\_graphs.set\_title("fi\'(t)")

ax\_for\_graphs.set(xlim = [0,10])

ax\_for\_graphs.grid(True)

ax\_for\_graphs = fig\_for\_graphs.add\_subplot(2,2,4)

ax\_for\_graphs.plot(t,dphi,color = 'black')

ax\_for\_graphs.set\_title('phi\'(t)')

ax\_for\_graphs.set(xlim = [0,10])

ax\_for\_graphs.grid(True)

fig\_for\_N = plt.figure(figsize=[13,7])  # построим их графики

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

ax\_for\_N.plot(t, N, color = 'blue')

ax\_for\_N.set\_title("N(t)")

ax\_for\_N.set(xlim = [0, 10])

ax\_for\_N.grid(True)

fig = plt.figure(figsize = [6.5, 6.5])  
ax = fig.add\_subplot(1, 1, 1)  
ax.set(xlim = [-20, 20], ylim = [-20, 20])  
  
def Rot2D(X,Y,Phi):  
 RotX = X \* np.cos(Phi) – Y \* np.sin(Phi)  
 RotY = X \* np.sin(Phi) + Y \* np.cos(Phi)  
 return RotX, RotY  
  
steps = 1000  
t = np.linspace(0, 10, steps)  
  
phi = np.sin(t)  
psi = np.cos(1.2 \* t)  
R1 = 5

R2 = 4  
OsX1 = -15

OsX2 = 15  
OsY1 = OsY2 = -R1  
x\_C = 0

y\_C = 0  
  
move = R1 \* psi  
betta = np.linspace(0, 6.28, 1000)  
X\_disk1 = R1 \* np.sin(betta) + x\_C  
Y\_disk1 = R1 \* np.cos(betta) + y\_C  
X\_disk2 = R2 \* np.sin(betta) + x\_C  
Y\_disk2 = R2 \* np.cos(betta) + y\_C  
  
Ax = R1 \* np.sin(psi) + move + x\_C  
Ay = R1 \* np.cos(psi) + y\_C  
r = (R1 - R2) / 2 + R2  
Bx = r \* np.sin(phi) + move + x\_C  
By = r \* np.cos(phi) + y\_C  
  
n = 15  
b = 1/(n-2)  
sh = 0.4  
x\_P = np.zeros(n)  
y\_P = np.zeros(n)  
x\_P[0] = 0  
x\_P[n-1] = 1  
y\_P[0] = 0  
y\_P[n-1] = 0  
for i in range(n-2):  
 x\_P[i+1] = b\*(i+1) - b/2  
 y\_P[i+1] = sh\*(-1)\*\*i  
  
katet\_x = Bx[i] - Ax[i]  
katet\_y = By[i] - Ay[i]  
stretch = np.sqrt(katet\_x \*\* 2 + katet\_y \*\* 2)  
alpha = np.pi + np.arctan2(katet\_y, katet\_x)  
Rx, Ry = Rot2D(x\_P \* stretch, y\_P, alpha)  
  
Centre = ax.plot(x\_C + move[0], y\_C, 'white', marker='o', ms=10, mec="c")[0]  
Line = ax.plot([OsX1, OsX2], [OsY1, OsY2], 'black')  
Circle1 = ax.plot(X\_disk1 + move[0], Y\_disk1, color="c")[0]  
Circle2 = ax.plot(X\_disk2 + move[0], Y\_disk2, color="c")[0]  
Spr = ax.plot(Rx + Bx[0], Ry + By[0], 'red')[0]  
A = ax.plot(Ax[0], Ay[0], 'k', marker='o', ms=4)[0]  
B = ax.plot(Bx[0], Ay[0], 'k', marker='o', ms=8)[0]  
  
def Animation(i):  
 Centre.set\_data(x\_C + move[i], y\_C)  
 Circle1.set\_data(X\_disk1 + move[i], Y\_disk1)  
 Circle2.set\_data(X\_disk2 + move[i], Y\_disk2)  
 A.set\_data(Ax[i], Ay[i])  
 B.set\_data(Bx[i], By[i])  
  
 katet\_x = Bx[i] - Ax[i]  
 katet\_y = By[i] - Ay[i]  
 stretch = np.sqrt(katet\_x \*\* 2 + katet\_y \*\* 2)  
 alpha = np.pi + np.arctan2(katet\_y, katet\_x)  
 Rx, Ry = Rot2D(x\_P \* stretch, y\_P, alpha)  
  
 Spr.set\_data(Rx + Bx[i], Ry + By[i])  
 return [Circle1, Circle2, Centre, Spr, A, B]  
  
a = FuncAnimation(fig, Animation, frames=steps, interval=10)  
plt.show()

Результат:

Изображение выглядит как текст, График, линия, снимок экрана

Автоматически созданное описаниеИзображение выглядит как текст, Шрифт, снимок экрана, График

Автоматически созданное описание