МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Московский Авиационный Институт
(Национальный Исследовательский Университет)

Институт №8 "Информационные технологий и прикладная математика"
Кафедра 806 "Вычислительная математика и программирование"

Лабораторная работа №3
по курсу "Теоретическая механика и основы компьютерного моделирования"
3 семестр

Студент: Леухин М. В.
Группа: М8О-206Б-20
Преподаватель: Сухов Е. А.
Подпись: ______________

Москва, 2021

Содержание

1 Теоретическая часть

Поля трубка кольцевой формы радиуса R может вращаться вокруг вертикальной оси Oz. Момент инерции трубки относительно оси Oz равен J_z. Внутри трубки без трения движется материальная точка массы m. На трубку действует момент внешних сил относительно оси Oz, равный $M_{Oz} = -c\varphi$ (φ — угол поворота трубки вокруг оси Oz, c — постоянная).

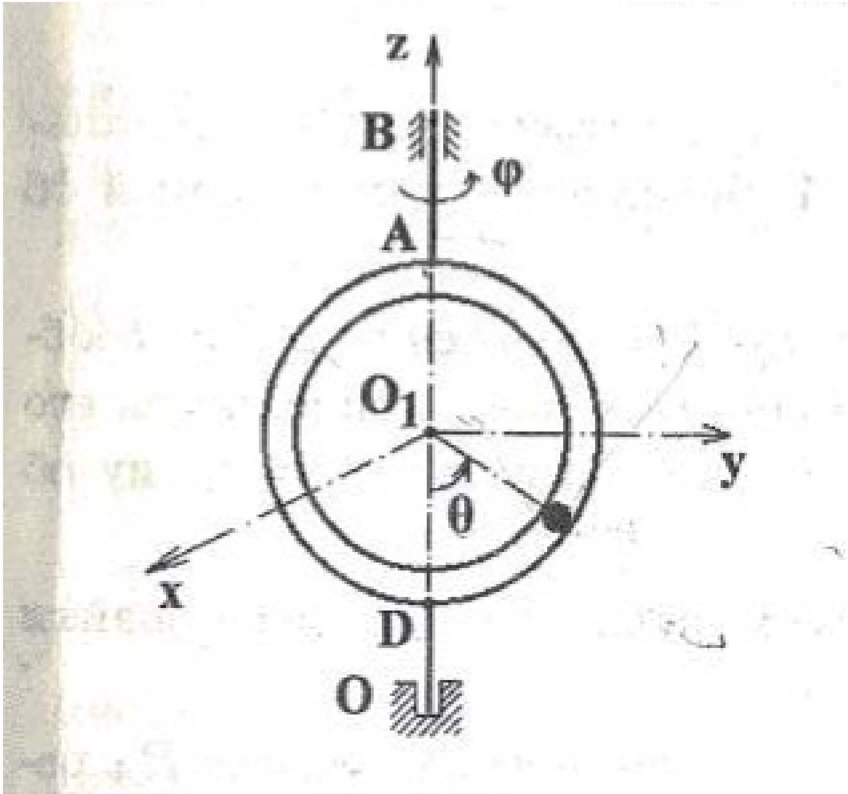

Рис. 11

Необходимо вывести дифференциальные уравнения движения системы, используя уравнения Лагранжа второго рода.

В данной системе имеем две обобщённые координаты: это углы φ и θ. Уравнение Лагранжа второго рода имеет вид:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i^n$$

где $L = T - \Pi$ есть разность кинетической и потенциальной энергий системы, а Q_i^n — непотенциальные обобщённые силы. Заметим, что на систему не действуют непотенциальные силы, а потому $Q_i^n = 0$.

Выведем уравнение для нашей системы. Сначала определим кинетическую энергию системы. Вращающаяся кольцевая трубка обладает кинетической энергией $T_1 = \frac{J_z\dot{\varphi}^2}{2}$. Вращающаяся точка обладает кинетической энергией $T_2 = \frac{mv^2}{2}$. Исходя из того, что $x = R\sin\theta\cos\varphi, y = R\sin\theta\sin\varphi, z = -R\cos\theta$, найдём скорость точки: $v = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} = R\sqrt{\dot{\theta}^2 + \sin^2\theta\dot{\varphi}^2}$. Теперь определим, какие потенциальные силы действуют на систему. На кольцевую трубку относительно оси Oz действует сила, момент которой равен $-c\varphi$, а значит потенциальная энергия будет $\Pi_1 = \frac{c\varphi^2}{2}$. На точку материальную точку действует сила тяжести. Возьмём за нулевой уровень нижнее положение точки, тогда потенциальная энергия равна $\Pi_2 = -mgR\cos\theta - mgR$. Исходя из этого, получаем:

$$L = (T_1 + T_2) - (\Pi_2 + \Pi_2)$$
$$L = \frac{J_z\dot{\varphi}^2}{2} + \frac{mR^2(\dot{\theta}^2 + \sin^2\theta\dot{\varphi}^2)}{2} - \frac{c\varphi^2}{2} + mgR\cos\theta + mgR$$

Рассмотрим сначала уравнение Лагранжа 2 рода относительно координаты φ:

$$\frac{\partial L}{\partial \dot{\varphi}} = J_z\dot{\varphi} + mR^2\sin^2\theta\dot{\varphi}$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) = J_z\ddot{\varphi} + mR^2\sin^2\theta\ddot{\varphi} + 2mR^2\sin\theta\cos\theta\dot{\varphi}\dot{\theta} =$$
$$= (J_z + mR^2\sin^2\theta)\ddot{\varphi} + mR^2\sin 2\theta\dot{\varphi}\dot{\theta}$$
$$\frac{\partial L}{\partial \varphi} = -c\varphi$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) - \frac{\partial L}{\partial \varphi} = (J_z + mR^2\sin^2\theta)\ddot{\varphi} + mR^2\sin 2\theta\dot{\varphi}\dot{\theta} + c\varphi$$

А теперь относительно θ:

$$\frac{\partial L}{\partial \dot{\theta}} = mR^2\dot{\theta}$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = mR^2\ddot{\theta}$$
$$\frac{\partial L}{\partial \theta} = mR^2\sin\theta\cos\theta\dot{\varphi}^2 - mgR\sin\theta$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) - \frac{\partial L}{\partial \varphi} = mR^2\ddot{\theta} - mR^2\sin\theta\cos\theta\dot{\varphi}^2 + mgR\sin\theta =$$
$$= mR^2(\ddot{\theta} - \sin\theta\cos\theta\dot{\varphi}^2) + mgR\sin\theta$$

Тогда дифференциальные уравнения, описывающие движение системы, имеют вид:

$$(J_z + mR^2\sin^2\theta)\ddot{\varphi} + mR^2\sin 2\theta\dot{\varphi}\dot{\theta} + c\varphi = 0$$
$$mR^2(\ddot{\theta} - \sin\theta\cos\theta\dot{\varphi}^2) + mgR\sin\theta = 0$$

2 Листинг программы

```
import numpy as np
import sympy as sp
import math
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
from matplotlib.patches import Circle
from scipy.integrate import odeint
import mpl_toolkits.mplot3d.art3d as art3d

def odesys(y, t, g, m, J, R, c):
    dy = np.zeros(4)
    dy[0] = y[2]
    dy[1] = y[3]

    a11 = J + m * R ** 2 * np.sin(y[1]) ** 2
    a12 = 0
    a21 = 0
    a22 = R

    b1 = -c * y[0] - m * R ** 2 * y[2] * y[3] * np.sin(2 * y[1])
```

```python
    b2 = -g * np.sin(2 * y[1]) + R * y[2] ** 2 * np.sin(y[1]) * np.cos(y[1])

    dy[2] = (b1 * a22 - b2 * a12) / (a11 * a22 - a12 * a21)
    dy[3] = (b2 * a11 - b1 * a21) / (a11 * a22 - a12 * a21)
    return dy

tn = np.linspace(0, 10, 2000)
xn = np.zeros_like(tn)
yn = np.zeros_like(tn)
zn = np.zeros_like(tn)

g = 9.81
m = 1
J = 0.5
R = 0.5
c = 2

phi0 = -0.3
theta0 = 0.6
dphi0 = -0.2
dtheta0 = 0.3
y0 = [phi0, theta0, dphi0, dtheta0]

Y = odeint(odesys, y0, tn, (g, m, J, R, c))

phi = Y[:, 0]
theta = Y[:, 1]

for i in range(len(tn)):
    xn[i] = R * np.sin(theta[i]) * np.cos(phi[i])
    yn[i] = R * np.sin(theta[i]) * np.sin(phi[i])
    zn[i] = - R * np.cos(theta[i])

fig = plt.figure()
ax = fig.add_subplot(projection="3d")
ax.set(xlim=[-1, 1], ylim=[-1, 1], zlim=[-1, 1])

fig2 = plt.figure()

ax1 = fig2.add_subplot(2, 2, 1)
ax1.plot(Y[:, 0])
ax1.set_title("$\\varphi$")

ax2 = fig2.add_subplot(2, 2, 2)
ax2.plot(Y[:, 1])
ax2.set_title("$\\theta$")

ax3 = fig2.add_subplot(2, 2, 3)
ax3.plot(Y[:, 2])
ax3.set_title("$\\varphi'$")

ax4 = fig2.add_subplot(2, 2, 4)
```

```
ax4.plot(Y[:, 3])
ax4.set_title("$\\theta'$")


def plot_vector(fig, orig, v, color='blue'):
    ax = fig.gca(projection='3d')
    orig = np.array(orig)
    v = np.array(v)
    ax.quiver(orig[0], orig[1], orig[2], v[0], v[1], v[2], color=color)
    ax.set_xlim(0, 10)
    ax.set_ylim(0, 10)
    ax.set_zlim(0, 10)
    ax = fig.gca(projection='3d')
    return fig


def rotation_matrix(d):
    sin_angle = np.linalg.norm(d)
    if sin_angle == 0: return np.identity(3)
    d /= sin_angle
    eye = np.eye(3)
    ddt = np.outer(d, d)
    skew = np.array([[0, d[2], -d[1]],
                     [-d[2], 0, d[0]],
                     [d[1], -d[0], 0]], dtype=np.float64)
    M = ddt + np.sqrt(1 - sin_angle ** 2) * (eye - ddt) + sin_angle * skew
    return M


def pathpatch_2d_to_3d(pathpatch, z, normal):
    if type(normal) is str: # Translate strings to normal vectors
        index = "xyz".index(normal)
        normal = np.roll((1.0, 0, 0), index)

    normal /= np.linalg.norm(normal)  # Make sure the vector is normalised
    path = pathpatch.get_path()  # Get the path and the associated transform
    trans = pathpatch.get_patch_transform()

    path = trans.transform_path(path)  # Apply the transform

    pathpatch.__class__ = art3d.PathPatch3D  # Change the class
    pathpatch._code3d = path.codes  # Copy the codes
    pathpatch._facecolor3d = pathpatch.get_facecolor  # Get the face color

    verts = path.vertices  # Get the vertices in 2D

    d = np.cross(normal, (0, 0, 1))  # Obtain the rotation vector
    M = rotation_matrix(d)  # Get the rotation matrix

```

```
    pathpatch._segment3d = np.array([np.dot(M, (x, y, 0)) + (0, 0, z)
        for x, y in verts])


def pathpatch_translate(pathpatch, delta):
    pathpatch._segment3d += delta


def plot_plane(ax, point, normal, size=10, color='y'):
    p = Circle((0, 0), size, facecolor=color, alpha=.2)
    ax.add_patch(p)
    pathpatch_2d_to_3d(p, z=0, normal=normal)
    pathpatch_translate(p, (point[0], point[1], point[2]))


def update(i):
    point.set_data_3d(xn[i], yn[i], zn[i])
    global cr
    cr.remove()
    cr = Circle((0, 0), R)
    cr.set_alpha(0.4)
    ax.add_patch(cr)
    pathpatch_2d_to_3d(cr, z=0, normal=[yn[i], -xn[i], 0])
    return point


point = ax.plot(xn[0], yn[0], zn[0], marker=".", color="black")[0]
cr = Circle((0, 0), R)
cr.set_alpha(0.4)
ax.add_patch(cr)
pathpatch_2d_to_3d(cr, z=0, normal=[0, yn[0], 0])

a = FuncAnimation(fig, update, frames=len(tn), interval=10)
plt.show()
```

3 Результат работы программы

$$m = 1, J_z = 0.5, R = 0.5, c = 2, \varphi_0 = -0.3, \theta_0 = 0.6, \dot{\varphi}_0 = -0.2, \dot{\theta}_0 = 0.3$$