Московский авиационный институт (национальный исследовательский университет)

Институт № 8 «Информационные технологии и прикладная математика»

**Лабораторная работа №2**

**по курсу «Теоретическая механика»**

**Анимация системы**

Выполнил студент группы М8О-201Б-21 Старцев Иван Романович

Преподаватель: Чекина Евгения Алексеевна

Оценка:

Дата: 03.11.2022

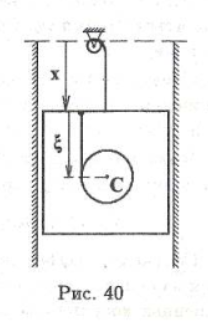
Москва, 2022

**Вариант №40**

**Задание:**

# Реализовать анимацию движения системы, отобразить с помощью графиков абсолютную скорость и абсолютное ускорение.

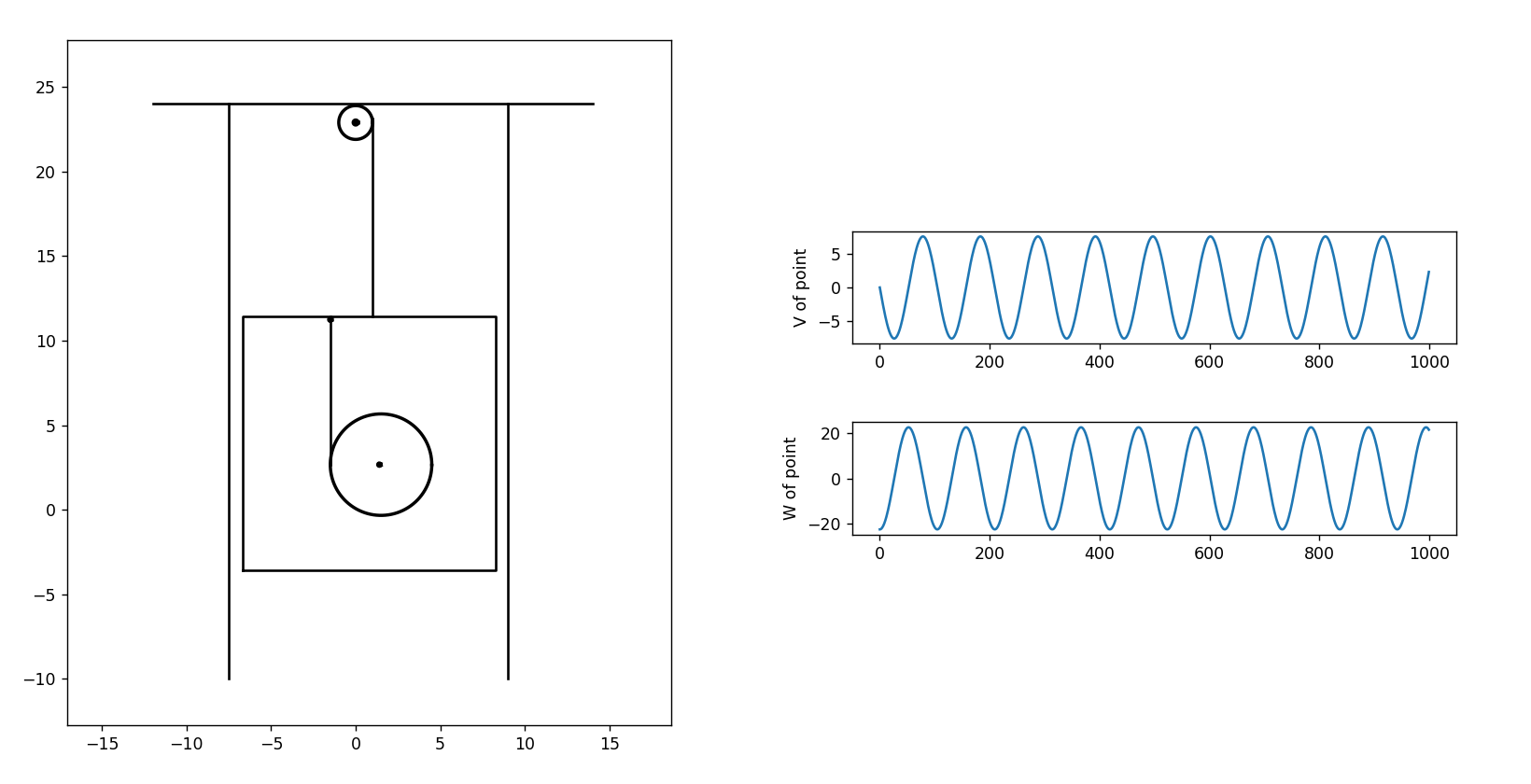
**Механическая система:**

****

**Текст программы**

import numpy as np  
import matplotlib.pyplot as plt  
from matplotlib.animation import FuncAnimation  
import sympy as sp  
import math  
  
def Square(x0, y0):  
 PX = [x0 - 7.5, x0 - 7.5, x0 + 7.5, x0 + 7.5, x0 - 7.5]  
 PY = [y0 - 7.5, y0 + 7.5, y0 + 7.5, y0 - 7.5, y0 - 7.5]  
 return PX, PY  
  
def Circle(X, Y, R):  
 CX = [X + R \* math.cos(i/100) for i in range(0, 628)]  
 CY = [Y + R \* math.sin(i/100) for i in range(0, 628)]  
 return CX, CY  
  
# [Xнач, Xкон], [Yнач, Yкон] => для линии надо сделать конечным фиксированную точку, а начало привязать к вершине блока  
def anima(i):  
 PrX, PrY = Square(XR[i], YR[i])  
 Prism.set\_data(PrX, PrY)  
 Line\_upper.set\_data([XR[i] + 0.2, 1], [YR[i] + 7.5, 23.15])  
 CX, CY = Circle(XC[i] + 0.3, 1.6 \* YC[i] - 4, 3)  
 Circle\_B.set\_data(CX, CY)  
 CBX, CBY = Circle(XC[i] - 2.7, YR[i] + 7.3, 0.05)  
 Circle\_BD.set\_data(CBX, CBY)  
 Line\_bottom.set\_data([XC[i] - 2.7, XC[i] - 2.7], [1.6 \* YC[i] - 4, YR[i] + 7.5])  
 BCX, BCY = Circle(XC[i] + 0.2, 1.6 \* YC[i] - 4, 0.05)  
 Circle\_BС.set\_data(BCX, BCY)  
 return Prism, Line\_upper, Circle\_B, Circle\_BD, Line\_bottom, Circle\_BС  
  
t = sp.Symbol('t')  
x = 4 \* sp.cos(3 \* t)  
xi = -1.5 \* sp.cos(3 \* t)  
  
Xr = x \* sp.sin(math.pi) + 0.8  
Yr = -x \* sp.cos(math.pi) + 7.5  
  
print(Xr)  
  
Xc = 1.5 \* (x \* sp.sin(math.pi) + 0.8)  
Yc = 2.5 \* (xi \* sp.cos(math.pi) + 3)  
  
V\_X = sp.diff(x, t)  
Vx = V\_X \* sp.cos(math.pi)  
Vy = -V\_X \* sp.sin(math.pi)  
  
Wx = sp.diff(Vx, t)  
Wy = sp.diff(Vy, t)  
  
# для графиков  
V\_Xi = sp.diff(xi, t)  
V\_X\_Xi = V\_X + V\_Xi  
W\_X\_Xi = sp.diff(V\_X\_Xi, t)  
  
T = np.linspace(0, 20, 1000)  
XR = np.zeros\_like(T)  
YR = np.zeros\_like(T)  
XC = np.zeros\_like(T)  
YC = np.zeros\_like(T)  
WX = np.zeros\_like(T)  
WY = np.zeros\_like(T)  
YX = np.zeros\_like(T)  
YC = np.zeros\_like(T)  
VX = np.zeros\_like(T)  
VY = np.zeros\_like(T)  
  
for i in np.arange(len(T)):  
 XR[i] = sp.Subs(Xr, t, T[i])  
 YR[i] = sp.Subs(Yr, t, T[i])  
 XC[i] = sp.Subs(Xc, t, T[i])  
 YC[i] = sp.Subs(Yc , 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])  
  
fig = plt.figure(figsize = (17, 10))  
  
ax1 = fig.add\_subplot(121)  
ax1.axis('equal')  
ax1.set(xlim=[XR.min() - 20, XR.max() + 20], ylim=[YR.min() - 20, YR.max() + 20])  
  
A\_R = 1  
A\_X = 0  
A\_Y = 22.9  
Circle\_A = ax1.plot(\*Circle(A\_X, A\_Y, A\_R), 'black', linewidth=2)  
  
AD\_R = 0.1  
AD\_X = 0  
AD\_Y = 22.9  
Circle\_AD = ax1.plot(\*Circle(AD\_X, AD\_Y, AD\_R), 'black', linewidth=3)  
  
B\_R = 3  
B\_X = 0  
B\_Y = 10  
Circle\_B = ax1.plot(\*Circle(B\_X, B\_Y, B\_R), 'black', linewidth=2)[0]  
  
BD\_R = 0.05  
BD\_X = -2.5  
BD\_Y = 18.9  
Circle\_BD = ax1.plot(\*Circle(BD\_X, BD\_Y, BD\_R), 'black', linewidth=3)[0]  
  
BС\_R = 0.05  
BС\_X = XR[0] + 0.2  
BС\_Y = YR[0] + 1.8  
Circle\_BС = ax1.plot(\*Circle(BС\_X, BС\_Y, BС\_R), 'black', linewidth=3)[0]  
  
upper\_line\_x = [-12, 14]  
upper\_line\_y = [24, 24]  
plt.plot(upper\_line\_x, upper\_line\_y, 'black')  
side\_line1\_x = [-7.5, -7.5]  
side\_line1\_y = [-10, 24]  
plt.plot(side\_line1\_x, side\_line1\_y, 'black')  
side\_line2\_x = [9, 9]  
side\_line2\_y = [-10, 24]  
plt.plot(side\_line2\_x, side\_line2\_y, 'black')  
  
PrX, PrY = Square(XR[0], YR[0])  
Prism = ax1.plot(PrX, PrY, 'black')[0]  
  
Line\_upper = ax1.plot([1, 1], [22.5, 19], 'black')[0]  
Line\_bottom = ax1.plot([-1.9, -1.9], [11, 18.8], 'black')[0]  
  
  
# ГРАФИКИ  
T = np.linspace(0, 20, 1000)  
VP = np.zeros\_like(T)  
WP = np.zeros\_like(T)  
l = np.zeros\_like(T)  
  
for i in np.arange(len(T)):  
 VP[i] = sp.Subs(V\_X\_Xi, t, T[i])  
 WP[i] = sp.Subs(W\_X\_Xi, t, T[i])  
 l[i] = i  
  
ax2 = fig.add\_subplot(424)  
ax2.plot(l, VP)  
ax2.set\_ylabel('V of point')  
  
ax4 = fig.add\_subplot(426)  
ax4.plot(l, WP)  
ax4.set\_ylabel('W of point')  
  
plt.subplots\_adjust(wspace = 0.3, hspace = 0.7)  
  
anim = FuncAnimation(fig, anima, frames = 1000, interval = 0.01, blit = True)  
  
plt.show()

**Результат работы программы:**

****

**Вывод:**

В ходе выполнения данной лабораторной работы были получены навыки анимации системы на языке Python.