Федеральное агентство по образованию РФ

Государственное образовательное учреждение высшего профессионального образования

Белгородский Государственный Технологический Университет

им. В.Г. Шухова

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №7 по дисциплине**

**«Системное моделирование»**

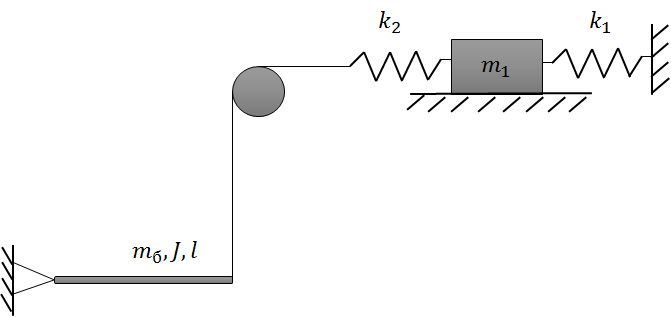
**Нелинейное представление динамических систем**

Выполнил студент группы ПВ-22

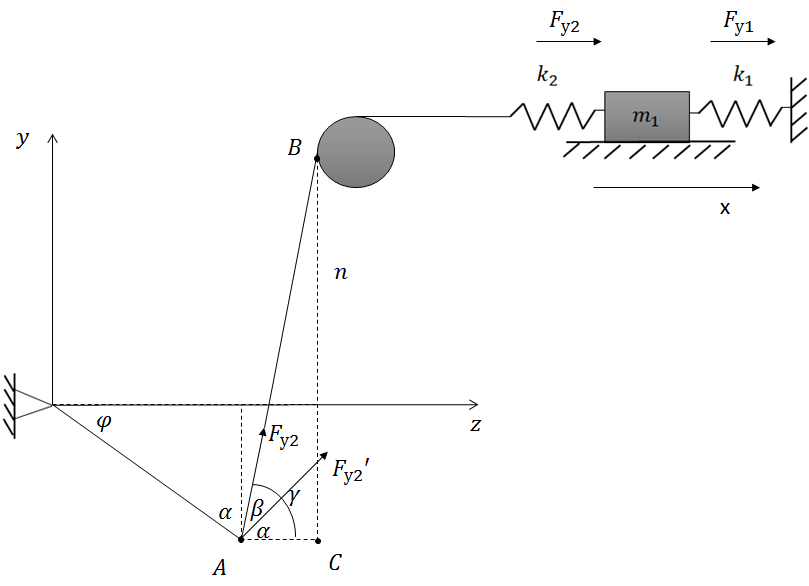
Притчин И.С.

Белгород 2017 г.

**Задание:** Расставить силы и составить дифференциальное уравнение данной нелинейной модели. Написать программу для решения этой системы дифференциальных уравнений.



Решение:

1. Степеней свободы – 2.
2. Введем координаты для каждой степени свободы - .
3. Для каждой степени свободы, для каждого тела, определяем действующие на тело по каждой координате (см. рис).

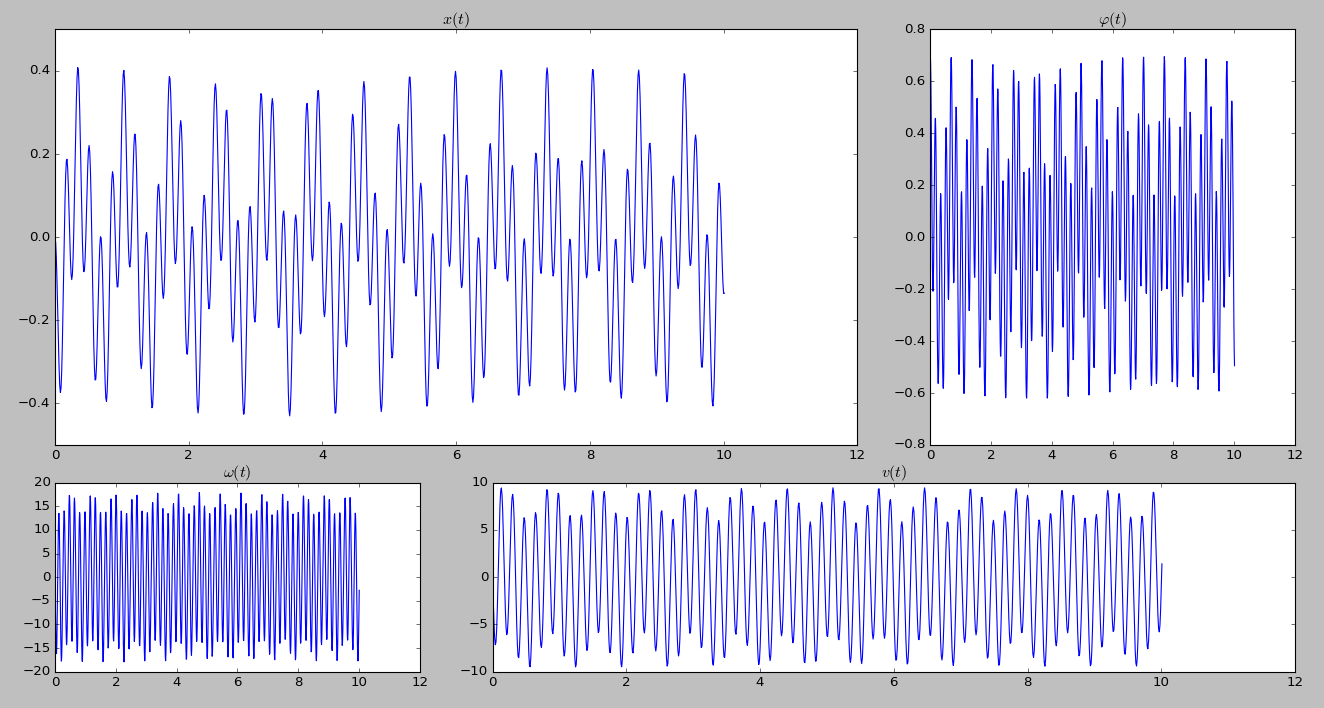
Введем вспомогательную систему координат . Оси проходит через ось подвеса балки и совпадает с положением балки, в котором она находится при недеформированной пружине. Ось тоже проходит через точку крепления и перпендикулярна оси .

Запишем систему дифференциальных уравнений:

С параметрами системы:

И начальными условиями: .

Сведем систему в форму Коши:



***Листинг программы***

|  |
| --- |
| import matplotlib.pyplot as plt  import matplotlib  import pylab  import time  from math import cos  from math import sqrt  from math import sin  from math import atan  from math import asin  from math import pi  #utf-8  k1 = 1000  k2 = 2000  l = 1  mb = 5  g = 9.8  J = 2  m1 = 5  n1 = 2  def func\_fi(w):  return w  def func\_x(v):  return v  def func\_v(x, fi):  BD = (sqrt(l\*l\*((cos(fi)-1) \*\* 2) + (-l\*sin(fi)-n1) \*\* 2))  beta = asin( (l\*sin(fi) + n1)/ BD) - (pi/2 - fi)  return (-k1\*x-k2\*(BD-n1+x))/m1  def func\_w(x, fi):  BD = (sqrt(l\*l\*((cos(fi)-1) \*\* 2) + (-l\*sin(fi)-n1) \*\* 2))  beta = asin( (l\*sin(fi) + n1)/ BD) - (pi/2 - fi)  return l\*(g\*mb\*cos(fi) - 2\*k2\*(BD-n1+x)\*cos(beta))/(2\*J)  def write\_graph\_faz(arr\_fi, arr\_w, arr\_x, arr\_v):  pylab.subplot2grid((4,4), (0,0), rowspan=2, colspan=4)  pylab.title(r'$\varphi(\omega)$')  pylab.plot(arr\_w, arr\_fi)  pylab.subplot2grid((4,4), (2,0), rowspan=2, colspan=4)  pylab.title(r'$x(v)$')  pylab.plot(arr\_v, arr\_x)  def runge\_kutte(t0, t1, fi, x, v, w, h, check, bool=False):  arr\_fi = [fi]  arr\_x = [x]  arr\_v = [v]  arr\_w = [w]  arr\_t = [0]  if bool:  plt.ion()  plt.show()  while t0 < t1:  k11 = h \* func\_fi(w)  k12 = h \* func\_x(v)  k13 = h \* func\_v(x, fi)  k14 = h \* func\_w(x, fi)  k21 = h \* func\_fi(w + 1 / 2 \* k14)  k22 = h \* func\_x(v + 1 / 2 \* k13)  k23 = h \* func\_v(x + 1 / 2 \* k12, fi + 1 / 2 \* k11)  k24 = h \* func\_w(x + 1 / 2 \* k12, fi + 1 / 2 \* k11)  k31 = h \* func\_fi(w + 1 / 2 \* k24)  k32 = h \* func\_x(v + 1 / 2 \* k23)  k33 = h \* func\_v(x + 1 / 2 \* k22, fi + 1 / 2 \* k21)  k34 = h \* func\_w(x + 1 / 2 \* k22, fi + 1 / 2 \* k21)  k41 = h \* func\_fi(w + k34)  k42 = h \* func\_x(v + k33)  k43 = h \* func\_v(x + k32, fi + k31)  k44 = h \* func\_w(x + k32, fi + k31)  fi += 1 / 6 \* (k11 + 2 \* k21 + 2 \* k31 + k41)  x += 1 / 6 \* (k12 + 2 \* k22 + 2 \* k32 + k42)  v += 1 / 6 \* (k13 + 2 \* k23 + 2 \* k33 + k43)  w += 1 / 6 \* (k14 + 2 \* k24 + 2 \* k34 + k44)  t0 += h  arr\_x.append(x)  arr\_fi.append(fi)  arr\_v.append(v)  arr\_w.append(w)  arr\_t.append(t0)  if bool:  if check == 1:  write\_graph\_faz(arr\_fi, arr\_w, arr\_x, arr\_v);  else:  pylab.subplot2grid((3,3), (0,0), colspan=2, rowspan=2)  pylab.title(r'$x(t)$')  pylab.plot(arr\_t, arr\_x)  pylab.subplot2grid((3,3), (0,2), rowspan=2)  pylab.title(r'$\varphi(t)$')  pylab.plot(arr\_t, arr\_fi)  pylab.subplot2grid((3,3), (2, 0))  pylab.title(r'$\omega(t)$')  pylab.plot(arr\_t, arr\_w)  pylab.subplot2grid((3,3), (2, 1), colspan=2)  pylab.title(r'$v(t)$')  pylab.plot(arr\_t, arr\_v)    plt.pause(0.001)  plt.clf()  return arr\_fi, arr\_x, arr\_v, arr\_w, arr\_t  if \_\_name\_\_ == "\_\_main\_\_":  fi\_t = 0.7  x\_t = 0  v\_t = 0  w\_t = 0  t0 = 0  t1 = 10  check = int(input('Вывести: 1-анимацию; 2-график: '))  bool = check == 1  check = int(input('Вывести: 1-фазовый портрет; 2-зависимость координат от времени: '))  tuple\_4 = runge\_kutte(t0, t1, fi\_t, x\_t, v\_t, w\_t, 0.001, check, bool)  if not bool:  arr\_fi = tuple\_4[0]  arr\_x = tuple\_4[1]  arr\_v = tuple\_4[2]  arr\_w = tuple\_4[3]  arr\_t = tuple\_4[4]    if check == 1:  write\_graph\_faz(arr\_fi, arr\_w, arr\_x, arr\_v);  else:  pylab.subplot2grid((3,3), (0,0), colspan=2, rowspan=2)  pylab.title(r'$x(t)$')  pylab.plot(arr\_t, arr\_x)  pylab.subplot2grid((3,3), (0,2), rowspan=2)  pylab.title(r'$\varphi(t)$')  pylab.plot(arr\_t, arr\_fi)  pylab.subplot2grid((3,3), (2, 0))  pylab.title(r'$\omega(t)$')  pylab.plot(arr\_t, arr\_w)  pylab.subplot2grid((3,3), (2, 1), colspan=2)  pylab.title(r'$v(t)$')  pylab.plot(arr\_t, arr\_v)    pylab.show() |