12_47

December 18, 2022

1 12.47

[ ]:
```
import sympy as smp
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
import matplotlib.patches as mpatches
```

[ ]:
```
t, m, M, g,R,r = smp.symbols("t m M g R r")
theta, phi = smp.symbols(r"\theta \phi",cls = smp.Function)

theta = theta(t)
theta_d = smp.diff(theta,t)
theta_dd = smp.diff(theta_d,t)

phi = phi(t)
phi_d = smp.diff(phi,t)
phi_dd = smp.diff(phi_d,t)
```

[ ]:
```
T = M*(R**2)*(theta_d**2)+(m/
  ↪2)*((phi_d**2)*((R-r)**2)+(theta_d**2)*(R**2)+2*phi_d*theta_d*(R-r)*smp.
  ↪cos(theta-phi))+(m/4)*((R*theta_d-(R-r)*phi_d)**2)
```

[ ]:
```
P = -m*g*R*smp.cos(theta)-m*g*(R-r)*smp.cos(phi)-M*g*R*smp.cos(theta)
```

[ ]:
```
L = T-P
L
```

[ ]:
$$MR^2\left(\frac{d}{dt}\theta(t)\right)^2 + MRg\cos\left(\theta(t)\right) + Rgm\cos\left(\theta(t)\right) + gm\left(R-r\right)\cos\left(\phi(t)\right) + \frac{m\left(R\frac{d}{dt}\theta(t)-\left(R-r\right)\frac{d}{dt}\phi(t)\right)^2}{4}+\frac{m\left(R^2\left(\frac{d}{dt}\theta(t)\right)^2+\left(R-r\right)^2\left(\frac{d}{dt}\phi(t)\right)^2+2\left(R-r\right)\cos\left(\phi(t)-\theta(t)\right)\frac{d}{dt}\phi(t)\frac{d}{dt}\theta(t)\right)}{2}$$

1

[ ]:
```
eq_1 = (smp.diff(smp.diff(L,theta_d),t)-smp.diff(L,theta)).simplify()
eq_1
```

[ ]:
$$2MR^2\frac{d^2}{dt^2}\theta(t) + MRg\sin(\theta(t)) + Rgm\sin(\theta(t)) + \frac{Rm\left(R\frac{d^2}{dt^2}\theta(t) - (R-r)\frac{d^2}{dt^2}\phi(t)\right)}{2} - m(R-r)\sin(\phi(t)-\theta(t))\frac{d}{dt}\phi(t)\frac{d}{dt}\theta(t)+m\left(R^2\frac{d^2}{dt^2}\theta(t) - (R-r)\left(\frac{d}{dt}\phi(t) - \frac{d}{dt}\theta(t)\right)\sin(\phi(t)-\theta(t))\frac{d}{dt}\phi(t) + (R-$$

2

[ ]:
```
eq_2 = (smp.diff(smp.diff(L,phi_d),t)-smp.diff(L,phi)).simplify()
eq_2
```

[ ]:
$$\frac{m(R-r)\left(3R\frac{d^2}{dt^2}\phi(t) - R\frac{d^2}{dt^2}\theta(t) + 2g\sin(\phi(t)) - 3r\frac{d^2}{dt^2}\phi(t) + 2\sin(\phi(t)-\theta(t))\left(\frac{d}{dt}\theta(t)\right)^2 + 2\cos(\phi(t)-\theta(t))\frac{d^2}{dt^2}\theta($$
$$2$$

1.0.1

[ ]:
```
sols = smp.solve([eq_1,eq_2],[theta_dd,phi_dd],rational = False,simplify =␣
↪False)
```

[ ]:
```
sols[theta_dd]
```

[ ]:
$$-\frac{3MRg\sin(\theta(t))}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2(\phi(t)-\theta(t))} - \frac{Rgm\sin(\phi(t))}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2mc}$$
$$\frac{3Rgm\sin(\theta(t))}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2(\phi(t)-\theta(t))} + \frac{3Rm\sin(\phi(t)-\theta(t))\left(\frac{d}{dt}\phi(t)\right)^2}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2}$$
$$\frac{Rm\sin(\phi(t)-\theta(t))\left(\frac{d}{dt}\theta(t)\right)^2}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2(\phi(t)-\theta(t))} + \frac{2gm\sin(\phi(t))\cos(\phi(t)-\theta(t))}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2}$$
$$\frac{3mr\sin(\phi(t)-\theta(t))\left(\frac{d}{dt}\phi(t)\right)^2}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2(\phi(t)-\theta(t))} + \frac{2m\sin(\phi(t)-\theta(t))\cos(\phi(t)-\theta(t))\left(\frac{d}{dt}\theta\right.}{6MR^2+4R^2m+2Rm\cos(\phi(t)-\theta(t))-2m\cos^2}$$

[ ]:
```
w_theta_d_f = smp.lambdify((g,m,M,r,R,phi,theta,phi_d,theta_d),sols[theta_dd])
theta_d_f = smp.lambdify(theta_d,theta_d)
phi_d_f = smp.lambdify(phi_d,phi_d)
w_phi_d_f = smp.lambdify((g,m,M,r,R,phi,theta,phi_d,theta_d),sols[phi_dd])
```

[ ]:
```
def system(S,t):
    theta,w_theta,phi,w_phi = S
    return [
        theta_d_f(w_theta),
        w_theta_d_f(g,m,M,r,R,phi,theta,w_phi,w_theta),
        phi_d_f(w_phi),
        w_phi_d_f(g,m,M,r,R,phi,theta,w_phi,w_theta)
    ]
```

- M -
- m -
- $start = [\theta, \dot{\theta}, \phi, \dot{\phi}]$ -

[ ]:
```
g = 9.81
M = 1
m = 5
r = 0.5
R = 2
t = np.linspace(0,20,1000)
start = [np.pi/4,0.2,np.pi/3,0.5]
ans = odeint(system,y0 = start,t = t)
```

1.0.2

[ ]:
```
center1 = (np.sin(ans.T[0])*R,-np.cos(ans.T[0])*R)
center2 = (np.sin(ans.T[2])*(R-r)+center1[0],-np.cos(ans.T[2])*(R-r)+center1[1])
```

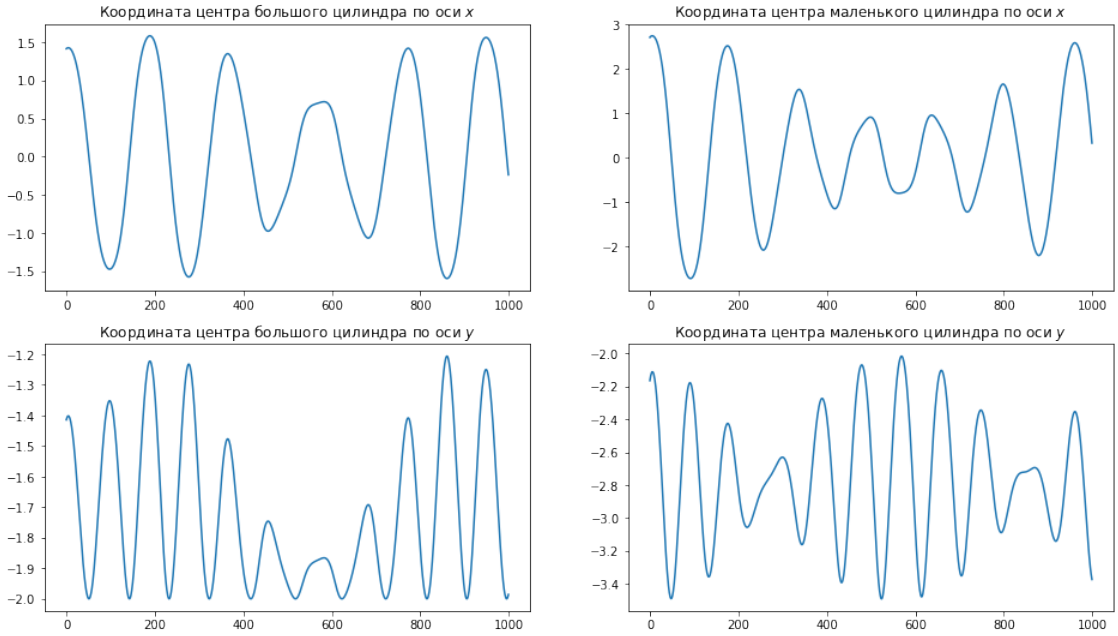
[ ]:
```
fig,ax = plt.subplots(2,2,figsize = (16,9))
ax[0,0].plot(center1[0])
ax[0,0].set_title("                          $x$")

ax[1,0].plot(center1[1])
ax[1,0].set_title("                          $y$")

ax[0,1].plot(center2[0])
ax[0,1].set_title("                           $x$")

ax[1,1].plot(center2[1])
ax[1,1].set_title("                           $y$")
```

[ ]:
```
Text(0.5, 1.0, '                          $y$')
```



[ ]:
```
c1 = plt.Circle((center1[0][0],center1[1][0]),R,fill = False,animated = True)
c2 = plt.Circle((center2[0][0],center2[1][0]),r,color = 'orange',fill =␣
 ↪True,animated = True)
massRatio = mpatches.Patch(color='red', label=r'$M/m = '+str(float(M/m))+r"$")
theta_start = mpatches.Patch(color='blue', label=r'$\theta_{0} = \pi/4'+r"$")
phi_start = mpatches.Patch(color='pink', label=r'$\phi_{0} = \pi/3'+r"$")
phi_d_start = mpatches.Patch(color='orange', label=r'$\dot{\phi} =␣
 ↪'+str(start[3])+r"$")
theta_d_start = mpatches.Patch(color='orange', label=r'$\dot{\theta} =␣
 ↪'+str(start[1])+r"$")

fig = plt.figure(figsize=(16,8))

ax = plt.axes(xlim=(-5, 5), ylim=(-4, 1))
ax.set_aspect('equal')
plt.grid()
def init():
    ax.add_patch(c1)
    ax.add_patch(c2)
    ax.add_patch(plt.Circle((0,0),0.1,color = 'black'))
    ax.
 ↪legend(handles=[phi_d_start,theta_d_start,phi_start,theta_start,massRatio],prop␣
 ↪= {'size': 16})
    return [c1,c2]

def animate(i):
```

```
    c1.set_center((center1[0][i],center1[1][i]))
    c2.set_center((center2[0][i],center2[1][i]))
    return [c1,c2]


anim = FuncAnimation(fig, animate, init_func=init,
                     frames=1000, interval=50, blit=True)
plt.show()
# anim.save('animation_4.mp4', fps=50,
#           extra_args=['-vcodec', 'h264',
#                       '-pix_fmt', 'yuv420p'])
```