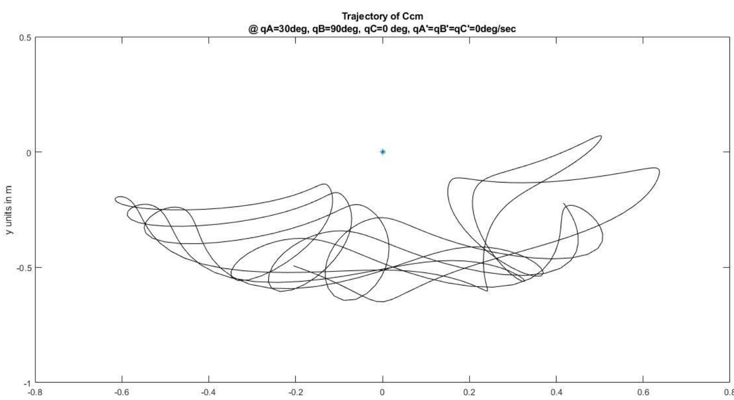
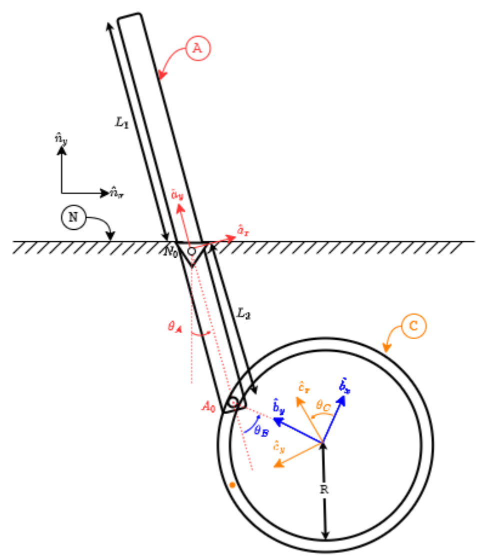
Chaotic Double Pendulum

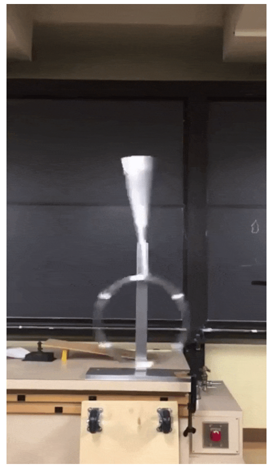
Anish Mokkarala & Nico Carballal



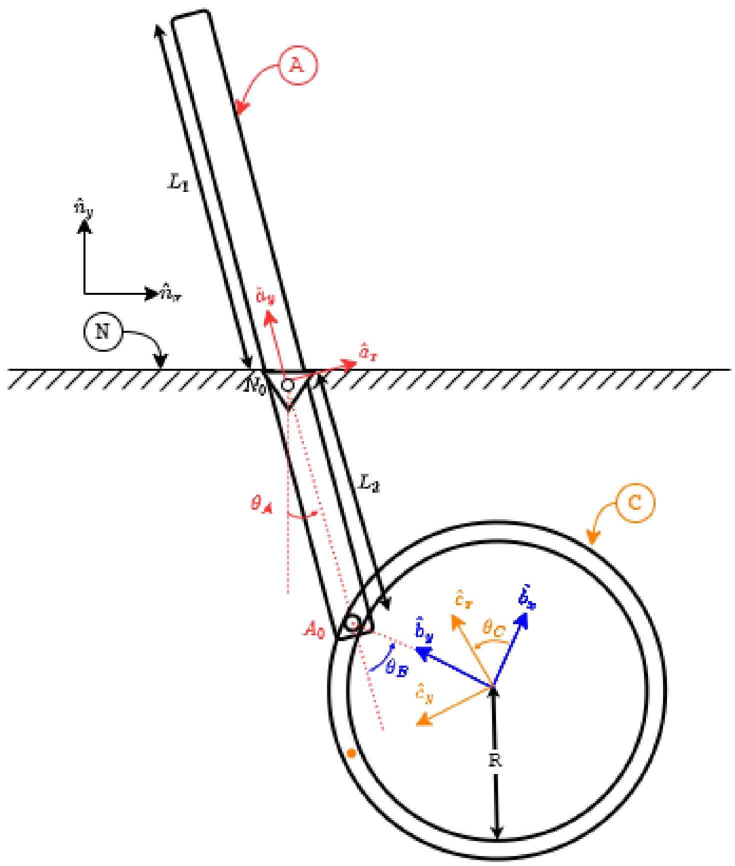


Question:

Given some initial perturbation to a ring connected in a double pendulum set-up as seen below(hyperlinked to YouTube), how can we track the center of mass of the ring?



Model:



Referring to the diagram pictured, the model assumptions are as follows:

- ❖ Rigid body A is modeled as a thin rod, and rigid body C is modeled as a thin ring.
- ❖ A is connected to a Newtonian Frame N with a frictionless revolute joint at N_0. A and C are connected through a frictionless prismatic joint at A_0, i.e. a pin attached to A at A_0 can slide freely in a slot along the circumference of C. Also, C can rotate freely about A_0.
- ❖ The mass in A and C is uniformly distributed.
- ❖ The rod has a simple rotation about N_0, and the ring, has two rotations: one about A_0 and one about C_{cm}.
- ❖ Unit vector $\hat{a}_y$ points from N_0 to A_{cm}. $\hat{b}_y$ points from C_{cm} to A_0, and $\hat{c}_y$ points from C_{cm} to a pre-defined (“painted”) point on the ring C. Also, $\hat{n}_z = \hat{a}_z = \hat{b}_z = \hat{c}_z$, and are pointing perpendicularly out of the plane of the model.
- ❖ There is no aerodynamic drag.

Model Identifiers:

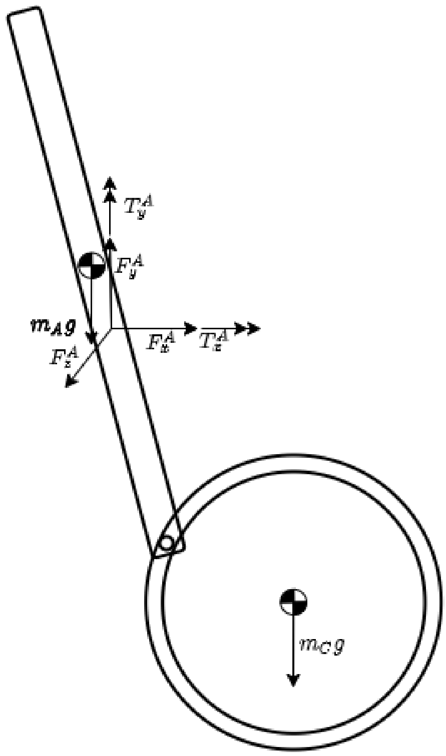
Quantity	**Identifier**	**Type**	**Value**
Earth's gravitational constant	g	Constant	$9.8\ m/s^2$
Mass of A	m_A	Constant	$2\ kg$
Mass of C	m_B	Constant	$1\ kg$
Length from N_0 to far end of rod from the ring	L_1	Constant	$0.6\ m$
Length from N_0 to A_0	L_2	Constant	$0.4\ m$
Radius of C	m_A	Constant	$0.25\ m$
Angle from $\hat{n}_y$ to $\hat{a}_y$ w/ $+\hat{n}_z$ sense	θ_A	Variable	-
Angle from $\hat{a}_y$ to $\hat{b}_y$ w/ $+\hat{n}_z$ sense	θ_B	Variable	-
Angle from $\hat{b}_y$ to $\hat{c}_y$ w/ $+\hat{n}_z$ sense	θ_C	Variable	-

Physics:

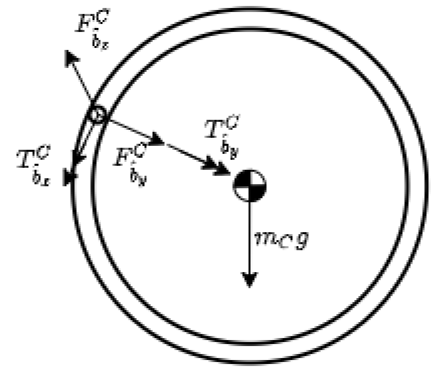
MG Road-Maps:

Variable	Translate/ Rotate	Direction (unit vector)	System S	FBD Of S	About point	**MG road-map equation**
θ_A	Rotate	$\hat{n}_z = \hat{a}_z$	A, B, C	Below	N_0	$\hat{\boldsymbol{n}}_z \cdot (\vec{\boldsymbol{M}}^{S/N_0} = \frac{{}^N d\, {}^N \vec{\boldsymbol{H}}^{S/N_0}}{dt})$
θ_B	Rotate	$\hat{n}_z = \hat{b}_z$	B, C	Below	A_0	$\hat{\boldsymbol{n}}_z \cdot (\vec{\boldsymbol{M}}^{S/A_0} = \frac{{}^N d\, {}^N \vec{\boldsymbol{H}}^{S/A_0}}{dt} + \ldots)$
θ_C	Rotate	$\hat{n}_z = \hat{c}_z$	C	Below	C_{cm}	$\hat{\boldsymbol{n}}_z \cdot (\vec{\boldsymbol{M}}^{S/C_{cm}} = \frac{{}^N d\, {}^N \vec{\boldsymbol{H}}^{S/C_{cm}}}{dt})$

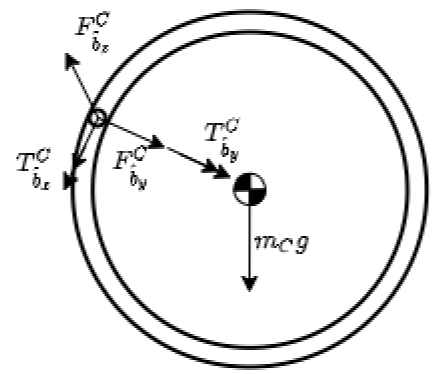
FBDs:



System-A, B, C



System-B, C



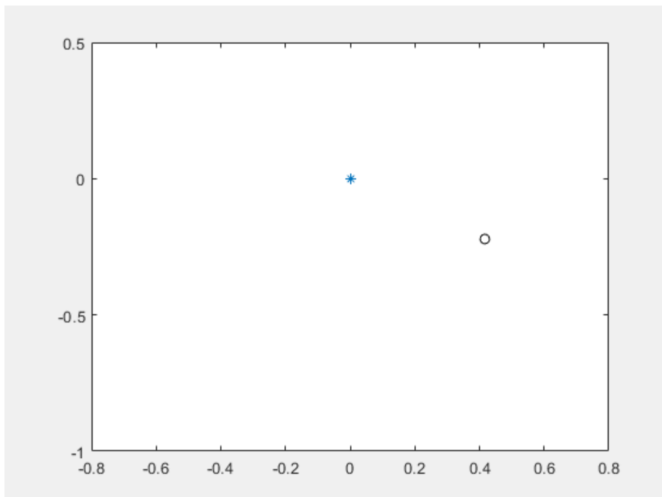
System-C

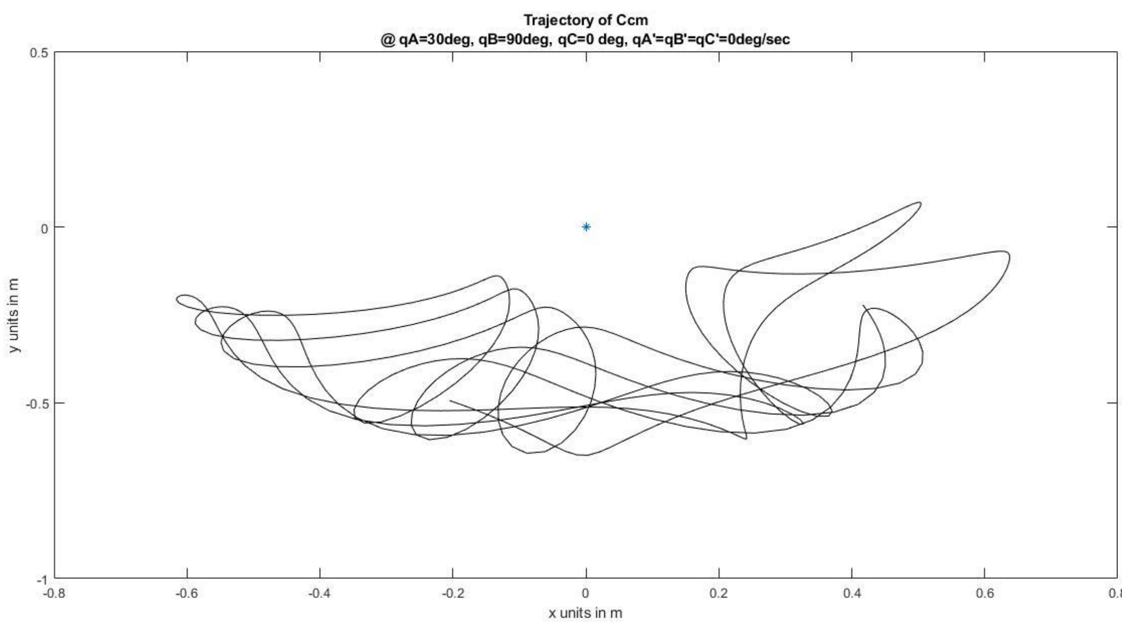
Simplifications/Solve:

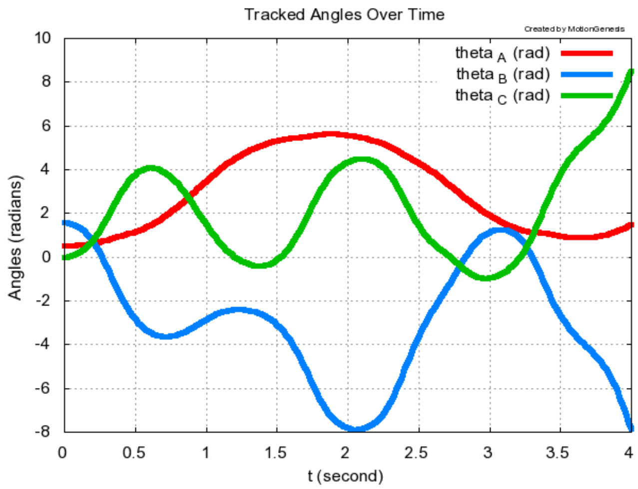
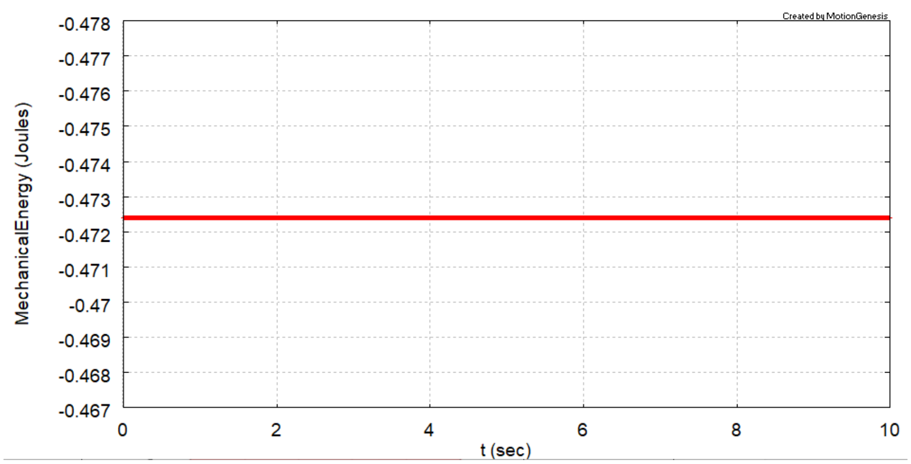
No simplifications were made further from the assumptions listed under the **Model** section. MotionGenesis was utilized to form the ODEs for this system in terms of θ_A'', θ_B'', θ_C'' and also to solve them numerically. Both MotionGenesis and MATLAB were used for plotting and GIF generation. The code used in MotionGenesis & MATLAB can be seen in the **Appendix** section.

Results and Discussion:

The pendulum is given a perturbation such that θ_A= 30 deg, θ_B= 90 deg, θ_C= 0 deg, θ'_A= 0 rad/sec, θ'_B= 0 rad/sec, θ'_C= 0 rad/sec initially. No is shown as a * and the Ccm position is shown as a **o** in the below GIF (link).





	Tracking our variable model identifiers over time demonstrate the chaotic nature of the system as there is no natural progression for the variables over time. Furthermore, no natural relationship between the variables is intuitively seen from the graph.
	The Mechanical Energy, $K + U$, of the system stays constant over the entire time. This occurs because only conservative forces (i.e. gravity) act on the system, and no energy is either gained or lost.

Appendix:

MG Code:

```
NewtonianFrame N
RigidBody A % Rod
RigidBody C % Ring
RigidFrame B % to characterize the rotation of Ccm about the joint at Ao

Variable qA'' % Az> measure of angle from Nx> to Ax>
Variable qB'' % Bz> measure of angle from Ax> to Bx>
Variable qC" % Bz> measure of C's angular velocity in B
Constant L1=0.6 m
Constant L2=0.4 m
Constant r=0.25 m
Constant g=9.8 m/s^2
A.SetMass(mA=2 kg)
C.SetMass(mC=1 kg)
A.SetInertia(Acm, IAxx, IAyy, IAzz=mA*L^2*1/12)
C.SetInertia(Ccm, ICxx, ICyy, ICzz= mC*r^2)

%Specifying rotations
A.RotateZ(N,qA)
B.RotateZ(A,qB)
C.RotateZ(B, qC)

%Specifying points
Acm.Translate(No, (L1-L2)/2*Ay>)
Ao.Translate(Acm, -(L1+L2)/2*Ay>)
Ccm.Translate(Ao, -r*By>)

%Specifying forces
System.AddForceGravity(-g*Ny>)

Dynamics[1]=Dot(Az>,System(A,C).GetDynamics(No))
Dynamics[2]=Dot(Az>,C.GetDynamics(Ao))
Dynamics[3]=Dot(Az>,C.GetDynamics(Ccm))

KineticEnergy = System.GetKineticEnergy()
GravityPotentialEnergy = System.GetForceGravityPotentialEnergy(  -g*Ny>,  No  )
MechanicalEnergy = KineticEnergy + GravityPotentialEnergy
xc=Dot(Nx>,Ccm.GetPosition(No))
yc=Dot(Ny>,Ccm.GetPosition(No))
```

```
Input  tFinal = 10 sec,  tStep = 0.02 sec,  absError = 1.0E-08
Input  qC = 0 deg, qC'=0 rad/sec, qA=30 deg, qB=90 deg, qA'=0 deg/sec, qB'=0 deg/sec

Solve(Dynamics=0,qA",qB",qC")

Output t sec, qC deg, qA deg, qB deg, xc m, yc m, Mechanical Energy Joules

ODE() MIPSI.m

%Plot MIPSI.1 [1,5]
Quit
```

After running the MIPSI.m file generated using the above code, the following code was used to generate the GIF on MATLAB.

```
vars = load( 'MIPSI.1' );
t = vars(:,1);
xc = vars(:,2);
yc = vars(:,3);
h = figure;
filename = 'track_Ccm';

for n = 1:length(xc)
    plot(0,0,"Marker","*");
   hold on;

   for i=1:n
      a=1-i/n;
      b=1-i/n;
      c=1-i/n;
      plot(xc(i),yc(i),"Marker","o","Color",[a b c]);
      axis([-0.8,0.8,-1,0.5]);
   end

   hold off;
   drawnow

   % Capture the plot as an image
   frame = getframe(h);
   im = frame2im(frame);
   [imind,cm] = rgb2ind(im,256);

   % Write to the GIF File
   if n == 1
```

```
        imwrite(imind,cm,filename,'gif', 'Loopcount',inf);
    else
        imwrite(imind,cm,filename,'gif','DelayTime',0.05,'WriteMode','append');
    end
end
```