MEM 530

Bhautik (Brian) Amin

Homework 1

Code located in Appendix

Problem 1 Part A: Determine Poles and Zeros of Transfer Function

For this problem, the transfer function was simplied and broken up into its numerator and denominator and brought into MATLAB. Using the tf() function, a transfer function model was generated within MATLAB. The poles of the transfer function are obtained by setting the denominator of the transfer function equal to 0 and solving for the roots of the equation. This was performed using the pole() function in MATLAB, yielding the following results:

poles =

-0.4030 ± 1.0717i, -0.0023 ± 0.0728i

To compute the zeros, set the numerator of the transfer function equal to 0 and solve for roots as well. In MATLAB this was performed using the zero() function to yield:

zeros =

-77.7937 + 0.0000i

-0.0032 ± 0.0757i

Problem 1 Part B: Is the system stable?

Looking at the poles and zeros of the transfer function we generate a pole-zero map using the MATLAB function pzmap()

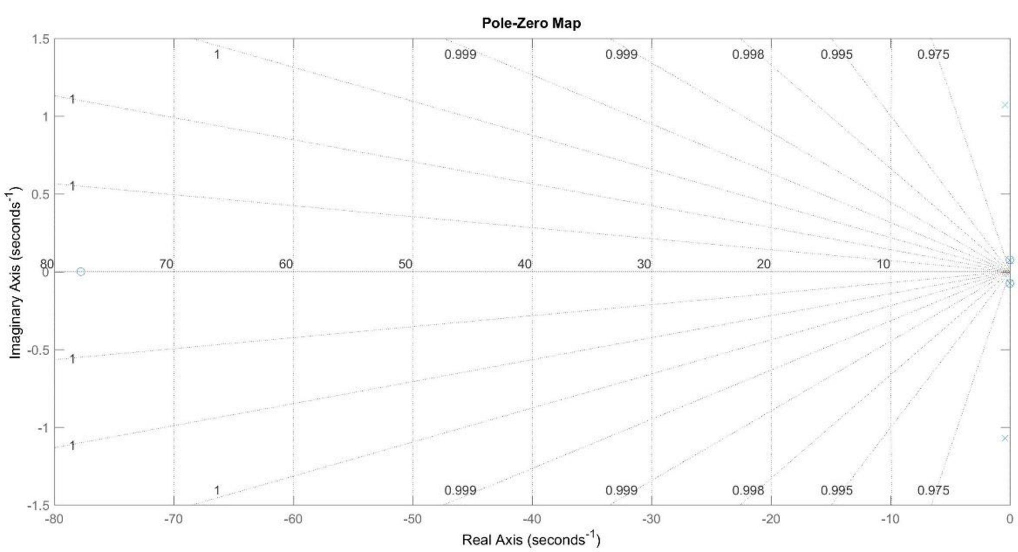


Figure 1. Pole Zero Map of the Given Transfer Function

The poles are denoted as x's and zeros denoted as o's (Figure 1). Looking at the poles on the chart, we can see that they all lie within the left hand plane (Zoom in using MATLAB and pan to see more clearly). If we also pay attention to the values of these poles themselves, we see that they have a negative real part, as well as there exist repeated roots. So with the poles being in the left hand plane, and having negative real parts we can say that the system is BIBO stable. BIBO stability deals with the response of a system to input, with zero initial conditions. BIBO stability is also valid for this system as the impulse response of the system within the time domains moves to zero as time approaches infinity (Figure 2).

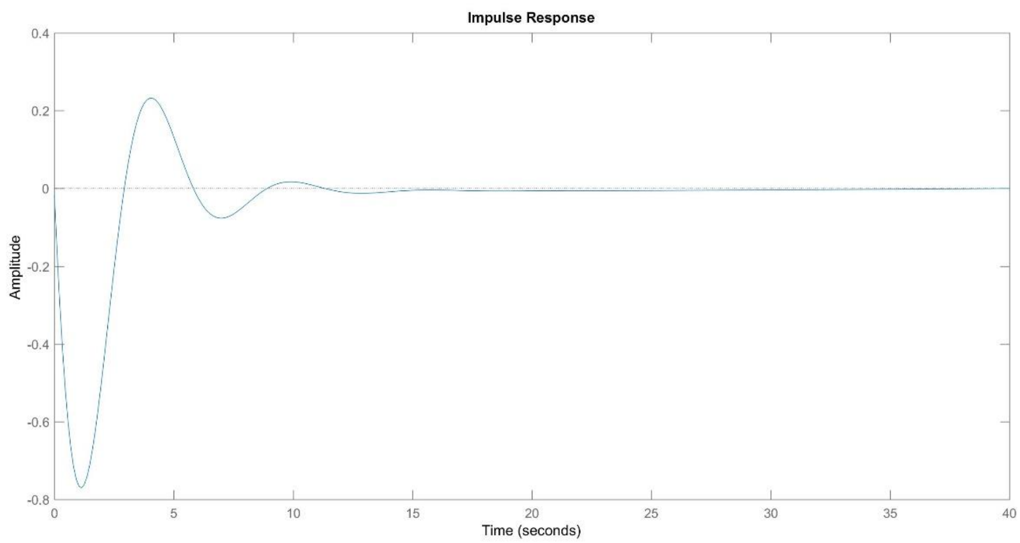


Figure 2. Impulse Response of Given Transfer Function System

If we looked at internal stability, that is – the stability as response to zero input and only to natural initial conditions we can see that the system is not marginally stable. As our roots to the characteristic equation (denominator of transfer function) has repeated roots. However, since the roots have negative real parts we can say that the system is asymptotically stable.

Problem 1 Part B (cont.): What are our natural frequencies and damping ratio?

The natural frequency and damping ratio of our system can be calculated using the damp() function within MATLAB:

Natural Frequency =[0.0728, 0.0728, 1.1450, 1.1450]

Damping ratio = [0.0320, 0.0320, 0.3520, 0.3520]

Problem 1 Part C: Are there any pole-zero cancellations?

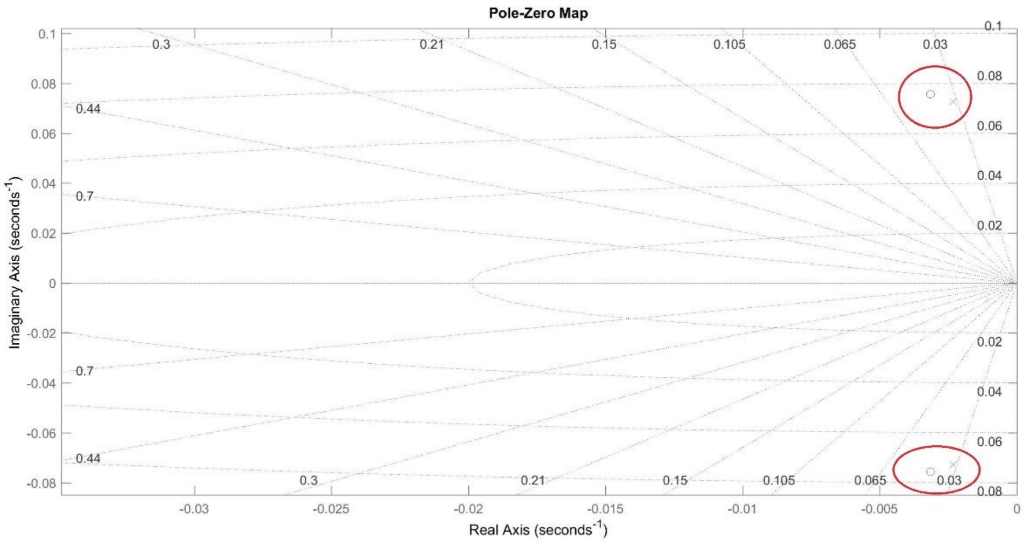


Figure 3. Pole Zero Map Zoomed In

Pole zero cancellation occurs when the poles and zeros share the same location. If we observe the pole and zero values themselves, we see that they are in fact different. If we visualize these close values however, we do notice that they are in a case of near pole zero cancellation, in which they are very close to each other.

Problem 1 Part D: Convert to State Space

To convert a transfer function to state space, a realization for the A, B, C, D matrices must be formed. State space representation uses these realizations to take the following form:

d/dt x = Ax+B

y = Cx+D

In MATLAB we can convert transfer functions into state space using the [A, B, C, D] = tf2ss() function:

And then take these realizations into state space using ss(A, B, C, D)

A =

-0.8107	-1.32	-0.01038	-0.006948
1	0	0	0
0	1	0	0
0	0	1	0

B =

1
0
0
0

C=

C = [-0.01785, -1.389, -0.008854, -0.007961]

D = 0

Problem 1 Part E: Plot step response of system

We can use the step() function to generate a plot of the response of the system to a unit step input function (Figure 4).

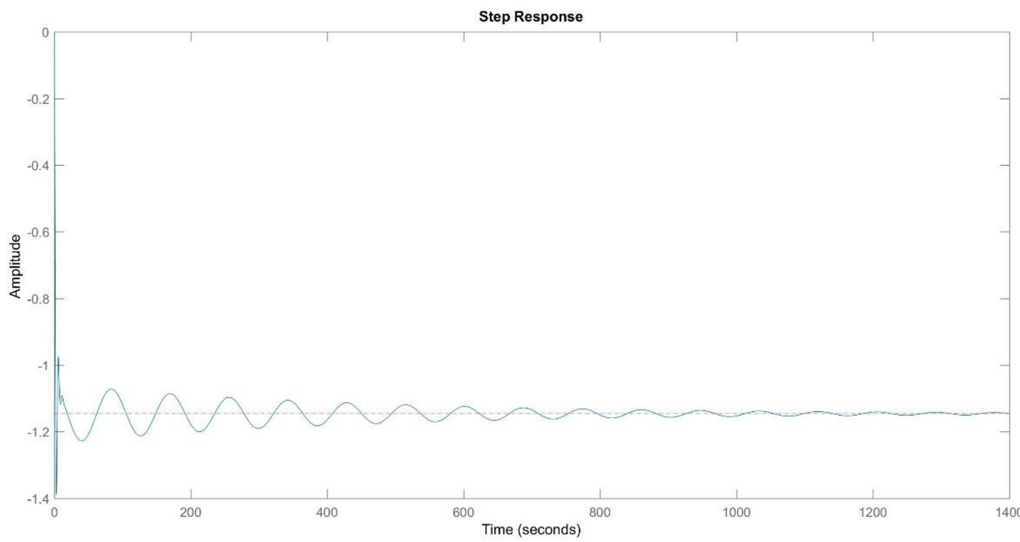


Figure 4. Step response of the given system

Problem 2 Part A: Determine poles and zeros of the transfer function

Here we follow the same lines of code for the variant transfer function. The poles and zeros are calculated as followed:

poles2 = -0.4025 ± 1.0784i

zeros2 = -77.8000

Problem 2 Part B: Is the system stable?

We can plot the poles and zeros to observe possible BIBO stability (Figure 5):

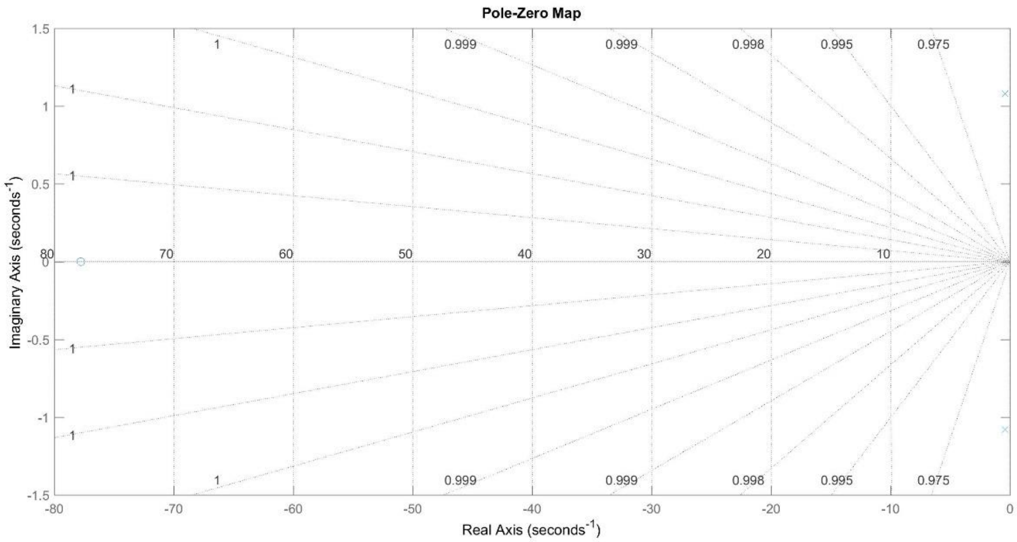


Figure 5. Pole Zero Map for second given transfer function

Taking a closer look at the two poles themselves, we can see that they are indeed in the left hand plane (Figure 6):

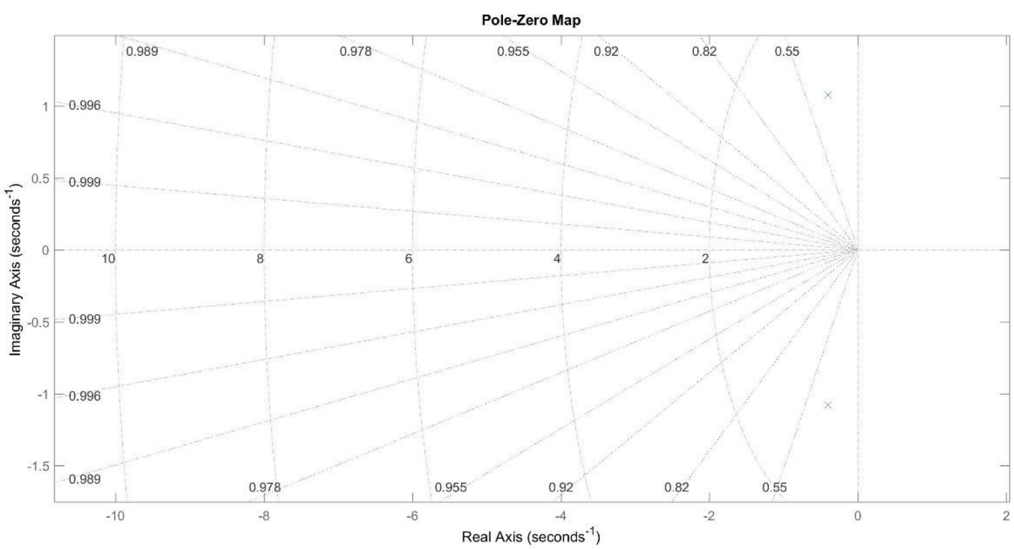


Figure 6. Closer look at the pole zero map for second given transfer function

We can see that the poles themselves numerically have a negative real part, and that the impulse response of this system moves to zero as time goes to infinity (Figure 7) making this system BIBO stable.

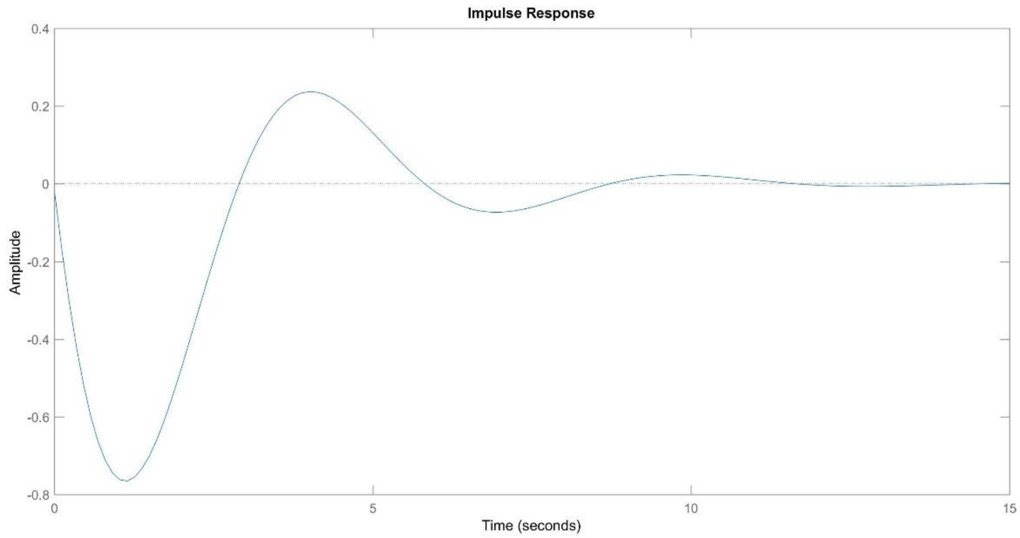


Figure 6. Impulse response of second give transfer function

For internal stability, we see that we have negative real repeated roots, so the system cannot be marginally stable. However, since the roots have a negative real part, this system is asymptotically stable.

Problem 2 Part B (cont.): What are our natural frequencies and damping ratio?

We can generate these values using the damp() function

Natural Frequencies = [0.0728, 0.0728, 1.1450, 1.1450]

Damping ratio = [0.0320, 0.0320, 0.3520, 0.3520]

Problem 2 Part C: Are there any pole zero cancellations?

Referring to Figure 6, we can see that the single zero is not close to the two poles. So this system has no pole zero cancellations.

Problem 2 Part D: Convert to State Space

As before, we can convert this system to state space using the ss2tf() function within MATLAB. Yielding the following results:

A=

-0.805	-1.325
1	0

B =

1
0

C = [-0.01782, -1.386]

D = 0

Problem 2 Part E: Plot step response

Using the step() function in matlab we generate:

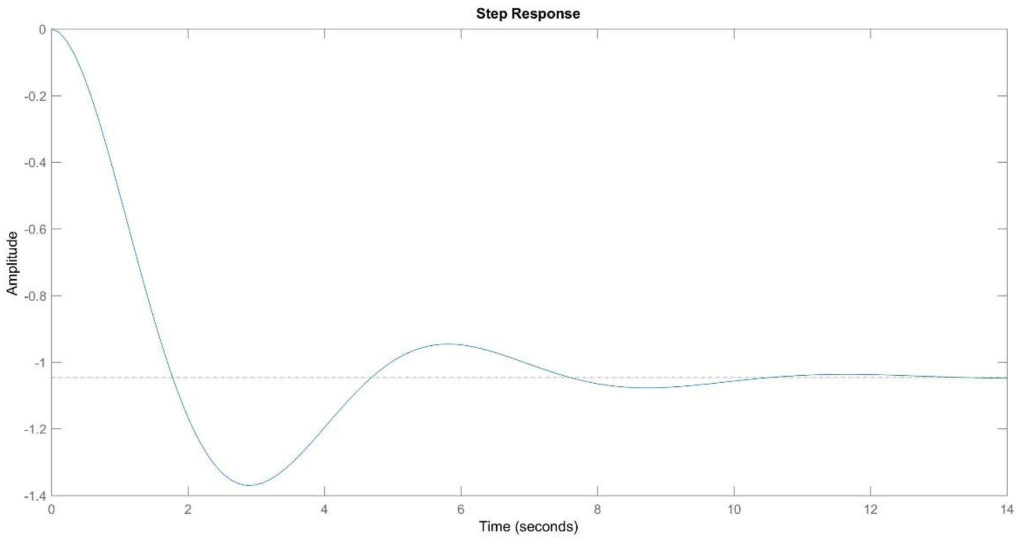


Figure 7. Step response of second given transfer function

Problem 3: Compare Results, can second transfer function be acceptable approximation of first one?

We can compare the step responses between both systems to see how they might differ (Figure 8)

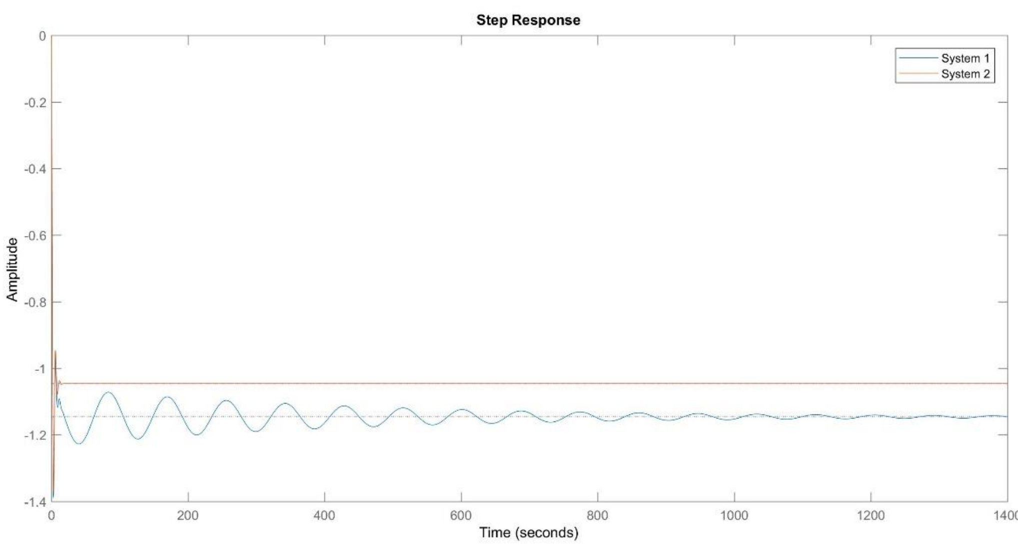


Figure 8. Side by Side Step Response

We can see that the first system (From Problem 1) takes a longer time to settle to its initial state. Whereas, system 2 settles much faster. We can zoom into this plot (Figure 9):

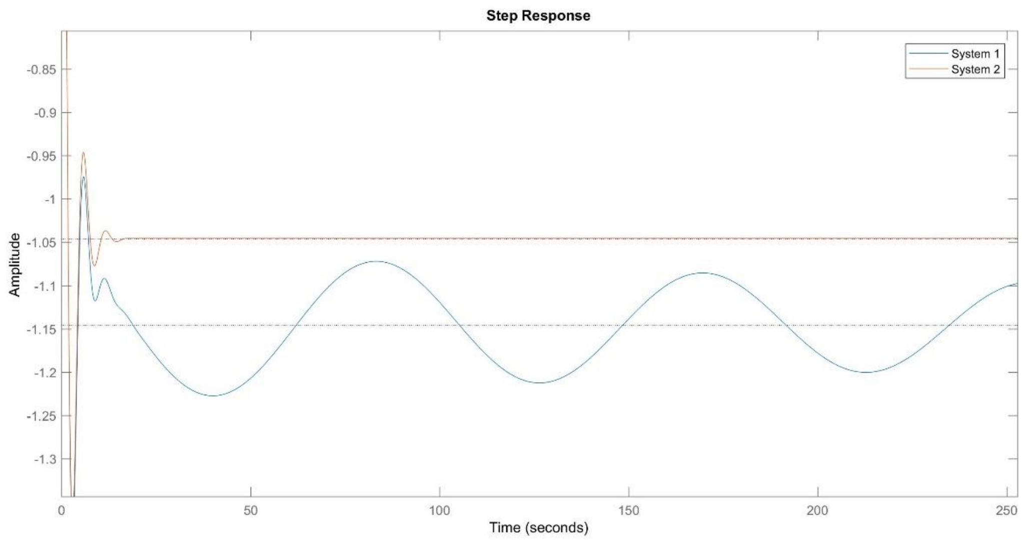
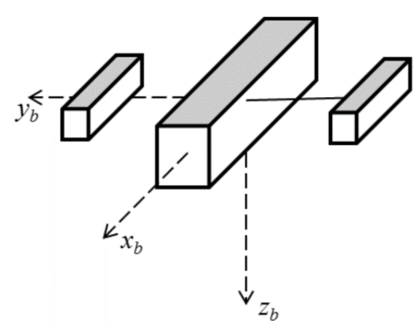


Figure 9. Zoomed in Side by Side step response

We can see that system 2 has a bit more over shoot, but the settling time is much faster. There does not seem to be any phase lag between either response either. System 2 may not be a good approximation of system 1, as it does not oscillate as system 1 does. If oscillation is not desirable however, system 2 is a good approximation of the initial response as shown in system 1 prior to the oscillation.

Problem 4 Part A: Moment of Inertia of Body without Tip masses

For a rectangle whose rotating about an axis going through the center the moment of inertia J can be calculated as:



$$J = \left(\frac{1}{12}\right) * mass * (height^2 + width^2)$$

The parameters of the system were brought into MATLAB and the moment of inertia for the body was calculated as:

$$166.6667\ kg * cm^2 = \left(\frac{1}{12}\right) * (10kg) * (10^2 cm + 10^2 cm)$$

Or 0.01666667 kg*m^2

Problem 4 Part B: Moment of Inertia for Tips

$$8.333\ kg * cm^2 = \left(\frac{1}{12}\right) * (2kg) * (5^2 cm + 5^2 cm)$$

Or 0.0008333 kg*m^2

Problem 4 Part C: Moment of Inertia for whole system

In order to calculate the moment of inertia for the entire system, the moment of inertia for each tip much be related back to the main center of gravity point. Using the parallel axis theorem we can conclude:

$$2.6667 * 10^3 kg * cm^2 = 166.6667 kg * cm^2 + (2kg * 25^2 cm) * 2$$

Or 0.2667 kg * m^2. This final value stems from the moment of inertia for the main body, plus the moment of inertia at the tips, related back to the main body (With the distance given as 25 cm). Multiplied by 2 as there were two tip masses.

Appendix:

```
%% Problem 1 Part A: Determine Poles and Zeros of Transfer Function
% Generate transfer function in matlab
num = [-0.01785 -1.38873 -0.0088536 -0.0079611]
denom = [1 0.81066 1.32005596 0.01038106 0.0069483]
tfsys = tf(num, denom)
%%
% Compute the poles of the transfer function model
poles = pole(tfsys)
%%
% Compute the zeros of the transfer function model
zeros = zero(tfsys)
%% Problem 1 Part B: Is the system stable?
% Looking at the poles and zeros we can generate
% pole-zero plot. For BIBO stability the poles must be in the left
hand
% plane
pzmap(tfsys)
grid on
%%
% We can see here that our poles are in the left hand plane, so this
% system is BIBO stable. BIBO stability also states that the roots of
our
% transfer function must have negative real parts. We can observe:
real(poles)
%%
% That our values are indeed negative real, as well as repeated roots.
% Finally, we can plot an unit impulse response of our system to see
if it
% goes to zero as time approaches infinity
impulse(tfsys)
%%
% Our system reaches zero as time moves to infinity, so its BIBO
stable in
% all cases
%%
% If we looked into internal stability, we see that with our repeated
roots
% of our transfer function our system is not marginally stable. Since
the real roots have a negative
% part however, we are asymptotically stable.
%%
% What are our natural frequencies and damping ratio?
[W, zeta] = damp(tfsys)
%% Problem 1 Part D: Convert to State Space Form
[A, B, C, D] = tf2ss(num, denom)
sys = ss(A, B, C, D)
%% Problem 1 Part E: Plot step response
step(sys)
%% Problem 2 Part A: Determine Poles and Zeros of Transfer Function
% Generate transfer function in matlab
```

```
num2= [-0.01782 -1.386396]
denom2 = [1 0.805 1.325]
tfsys2 = tf(num2, denom2)
%%
% Compute the poles of the transfer function model
poles2 = pole(tfsys2)
%%
% Compute the zeros of the transfer function model
zeros2 = zero(tfsys2)
%% Problem 2 Part B: Is the system stable?
% Looking at the roots of the denominator (the poles) we can generate
% pole-zero plot. For BIBO stability the poles must be in the left
hand
% plane
pzmap(tfsys2)
grid on
%%
% We can see here that our poles are in the left hand plane, so this
% system is BIBO stable. BIBO stability also states that the roots of
our
% transfer function must have negative real parts. We can observe:
real(poles2)
%%
% That our values are indeed negative real, as well as repeated roots.
% Finally, we can plot an unit impulse response of our system to see
if it
% goes to zero as time approaches infinity
impulse(tfsys2)
%%
% Our system reaches zero as time moves to infinity, so its BIBO
stable in
% all cases
%%
% If we looked into internal stability, we see that with our repeated
roots
% of our transfer function our system is not marginally stable. Since
the real roots have a negative
% part however, we are asymptotically stable.
%%
% What are our natural frequencies and damping ratio?
[W, zeta] = damp(tfsys)
%% Problem 2 Part D: Convert to State Space Form
[A2, B2, C2, D2] = tf2ss(num2, denom2)
sys2 = ss(A2, B2, C2, D2)
%% Problem 2 Part E: Plot step response
step(sys2)
%% Problem 3 Compare both systems
% We can compare the responses between both systems
step(sys)
hold on;
step(sys2)
legend('System 1', 'System 2')
```

```matlab
%% Problem 4 Part A: Moment of Inertia of Body without Tip Masses
% J = (1/12)*M*(h^2 + w^2) for xaxis
%% Setup Variables
m_body = 10 %kg
m_tips = 2 %kg
d_body = {10,10,30} %cm
d_tips = {5, 5, 10} %cm
y = 25 %cm
J_body = (1/12)*m_body*( (d_body{1}^2) + (d_body{2}^2) ) %In kg*cm^2
%% Problem 4 Part B: Moment of Inertia for Tips
J_tips = (1/12)*m_tips*( (d_tips{1}^2) + (d_tips{2}^2) ) %In kg*cm^2
%% Problem 4 Part C: Moment of Inertia for Entire System
J_total = (J_body) + ((m_tips*(y^2))*2)
```