ME 406 ADVENTURE 7

Control through a Flexible Manipulator

Course Value: 40 points

Deadline: COB 10 / 12 Nov 2004

INTRODUCTION

We will design a dynamic compensator to control the mass-spring damper system. Our compensator will fundamentally change the dynamics of the system.

OBJECTIVE

The objective of this adventure is to:

+ Design a notch-lag compensator to meet transient performance specifications
+ Get practice with the SISOTOOL CAD package
+ Implement the controller as a dynamic forward path on the ECP hardware.
+ Compare the hardware performance with that of a SISOTOOL / LTIVIEWER simulation

A. PRE-LAB

For this lab, we will use only the first cart and mass. Use the heaviest (most lightly damped) configuration from Lab 1. Use the parametric model that you determined in Lab 1 for design. For instance, I found the actual mass to be *m* = 2.7996 kg, stiffness (stiff spring) *k* = 822.77, and damping constant *c* = 16.42, static gain value was *K*= 1912 [couts/volt] (Be sure to use a value based on counts for your design). A lumped parameter model of the system looks like this

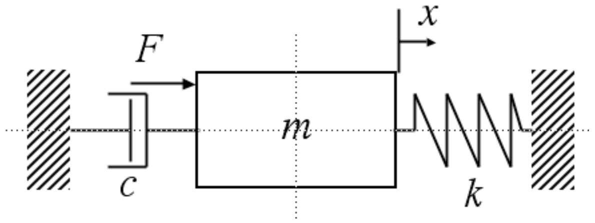


Figure 1: Plant Schematic

Applying first principles, the diferential equation of motion is: $m\ddot{x} + c\dot{x} + kx = KkF$, which makes the transfer function:

$$\frac{X(s)}{F(s)} = \frac{K}{\frac{m}{k}s^2 + \frac{c}{k}s + 1} \tag{1}$$

Substitute your numerical values from Lab 1 for *m*, *c*, *k*, and *K*.

This time, we will use a dynamic forward path (cascade) compensation. The block diagram is shown below. I have included the units with each signal label. It is important to consider these when designing you feedback controller. Most likely, your model from Lab 1 is based on cm. Go ahead and design your compensator based on using cm As shown in the block diagram, there is another static gain in the plant that takes the control signal in counts and divides by 100 to get the voltage that is actually sent to the motor. Thus, to correct your compensator for units, you should multiply all of the numerator gains by 100(counts/volt) / 2196(counts/cm)

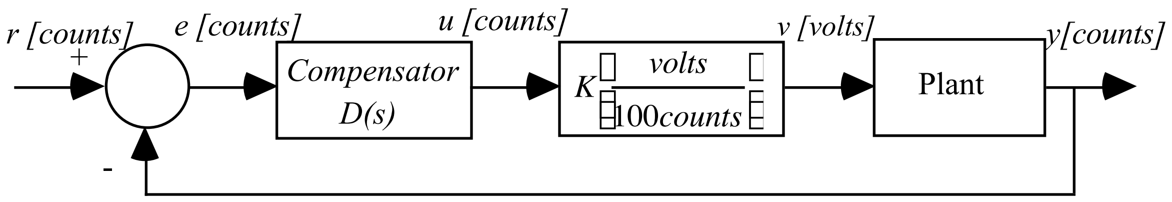


Figure 2: Closed-loop Block Diagram

The system uncompensated root locus should look something like the figure on the next page, left column. Add a complex pair of zeros as close to the lightly damped plant poles as possible.

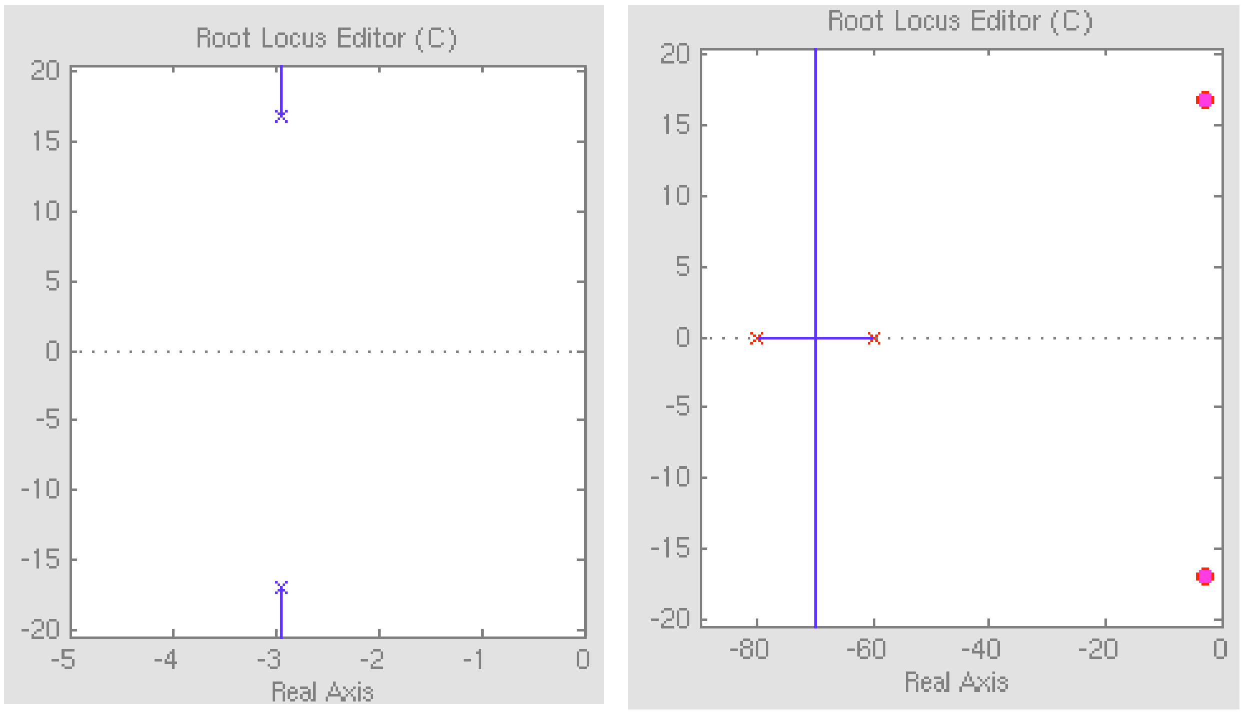
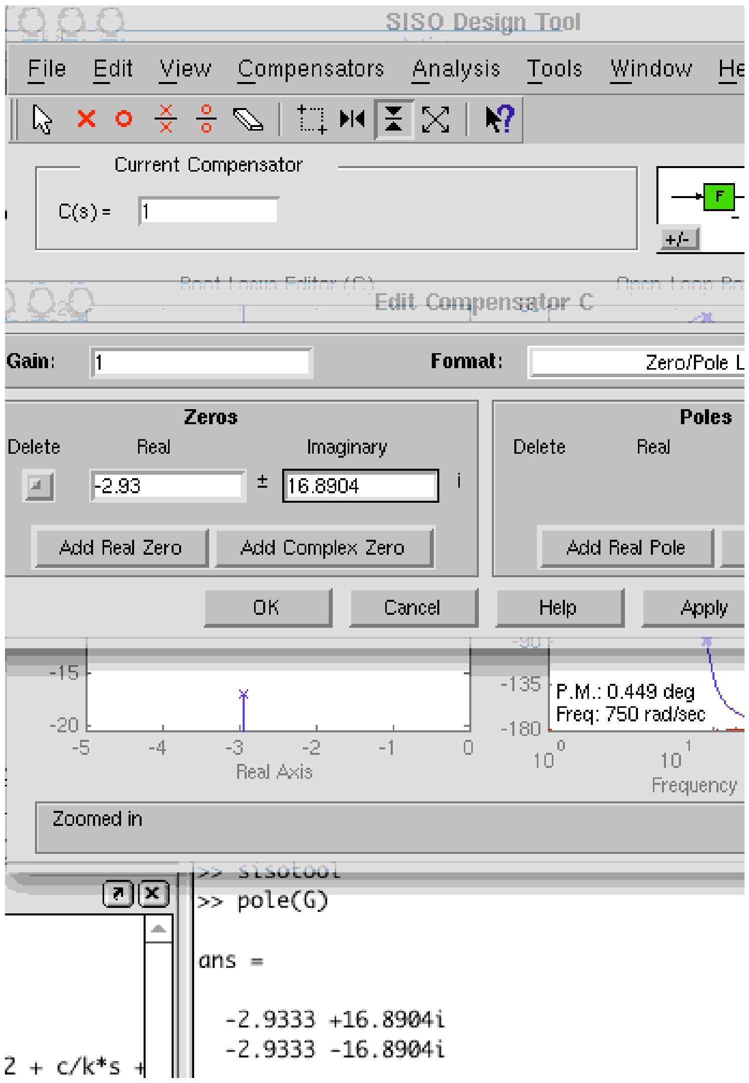


Figure 3: Uncompensated Root Locus and Notch Compensated Root Locus

If you type pole(G) at the command prompt, you can get precise values and then use the Edit compensator dialog to enter them precisely.



```
>> sisotool
>> pole(G)

ans =

  -2.9333 +16.8904i
  -2.9333 -16.8904i
```

```
2 + c/k*s +
```

Figure 4: Direclty Canceling the plant poles in SISOTOOL

Now, add real poles to replace the plant poles at approximately s = -60 and s = -80. You should then have something resembling the right column of Figure 3.

Now, increase the system gain until the closed-loop poles have a natural frequency of approximately 85 rad/s. This is close to the saturation limit of the motor, but choosing fast poles will give us a more impressive response. At this point, choose Analysis-Response to Step Command. Notice that you have a very fast rise time and settling time, however, the steady state error is very large. We can then introduce lag compensation to improve steady state performance of the controller.

Since we have placed the compensator poles at fast locations, we will be able to add relatively fast lag dynamics. The rule of thumb is that the lag zero should be about one decade below the slowest plant and compensator pole. That would place the lag zero at s = -6 according to the choices I've made above. I would like to increase the system gain by at least a factor of ten, so initially place the lag pole at s = -0.6. Now, experiment with changing the lag pole and zero locations and the system gain. Pay attention to what changes on the Root Locus and Bode plots, as well as the step response. See if you can reduce the steady state error to 2%. Here is my final SISOTOOL window:

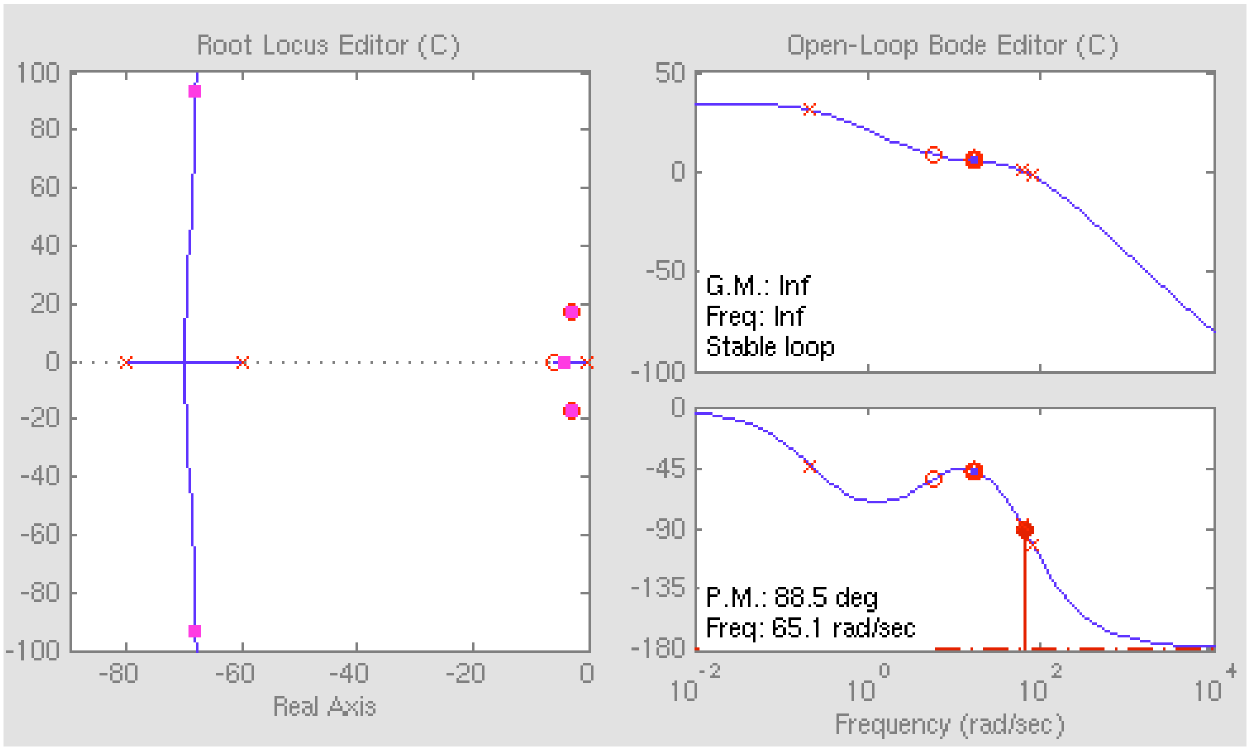


Figure 5: Final Notch Compensator Design Root Locus and Bode Plots

When finished, the sisotool plots should look something like this. Notice that the Bode magnitude spike due to the system lightly damped poles is completely flattened out.

Hopefully our compensator have reduced the height of the resonant peak by a great deal. Now you need to export the compensator, and put it in a form ready to implement on the ECP system. Use the export dialogue to export 'C' from sisotool to the workspace. Then by typing C at the command prompt, Matlab will display the current compensator in zero/pole/gain format. You can use the following commands to convert to numerator and denominator coefficients of the transfer function, the format we need to implement on the hardware:

```
[z,p,k] = zpkdata(C);
z = z{:};
p = p{:};
[num,den] = zp2tf(z,p,k)
```

Now remember to multiply your numerator coefficients by 100.

B. THE ADVENTURE CONTINUES - IN THE LAB

During the adventure you will set out to accomplish the following:

1. Set up the environment.

a. Use the same station you used for Lab 1. It should be set up in 1 DOF mode with four 500g brass masses on the first carriage. Connect the light spring to the first carriage. Connect the damper and insert the plug on half turn. If you need help configuring your station, ask your instructor.

b. Position the mass at zero on the position ruler. Under Utility, select Zero Position.

c. Check the encoder polarity. This is an important step that should be done each time before implementing a feedback controller. Some systems have 'reverse polarity' meaning that the sensor says the mass is displaced to the left when it is really displaced to the right. I have even seen a system change its polarity overnight! Log on to the computer and start the ECP executive program under Programs/ECP. Select Setup/Control Algorithm... In the dialog box, select the PID radio button, and press 'Setup Algorithm'. Enter a small value for proportional control such as 0.06. Set the derivative and integral gains to zero. Press Implement Algorithm. Press OK. Push the black button of the ECP control box. Set up the trajectory for a closed-loop step with an amplitude of 1000 counts, and a dwell time of 3000ms. Select Command/Execute and press Run. Select Plotting/Setup Plot. In the dialog, add Commanded Position and Encoder 1 Position to the Left Axis, the click Plot Data. If the Encoder 1 position moves in the same direction as the Commanded position, your system has normal polarity. If Encoder 1 position moves in the opposite direction, you have reverse polarity. This means you must multiply your contoller numerator gains by -1 throughout the rest of this experiment.

d. Select Setup/Control Algorithm... In the dialog box, select the Dynamic Forward Path radio button, and press 'Setup Algorithm'. The compensator format is

$$D(s) = \frac{s_7 s^7 + s_6 s^6 + \cdots + s_0}{r_7 s^7 + r_6 s^6 + \cdots + r_0}$$

where you can choose the gains s_i and r_i, i=0, 1, ..., 7 such that the compensator has up to 7 poles and 7 zeros. Carefully enter the coefficient values for your lead controller--you should only need s_1, s_2, r_1 and r_2. Be sure to include the compensator gain in the numerator terms. Press Implement Algorithm. Press OK. Push the black button of the ECP control box. You might hear the system rattle a bit due to sensor noise in the feedback loop.

2. Recording Step response data:

a. Select Command/Trajectory... In the dialog select the Step radio button and press Setup. Now select Closed Loop Move. The Amplitude should be set to 1000, set the dwell time to 3000 ms. Click OK. Click OK on the Trajectory Configuration dialog.

b. Select Command/Execute and press Run.

d. Watch the response. After the Upload successful dialog completes, click OK.

e. To look at an individual data set, select Plotting/Setup Plot. In the dialog, add Commanded Position and Encoder 1 Position to the Left Axis, the click Plot Data.

f. Export your data. Select Data/Export Raw Data... Browse to a convenient directory, floppy disks are recommended for storing this data. Save as type All files (*.*). Choose a name with meaning, like 'lab7ini.m'.

g. Check whether your system has met the specifications stated in the pre-lab. If not, continue to iterate your design using SISOTOOL (Matlab 6.5 is available on the lab computers). If you are having problems with stability or excessive noise, check that your units agree as shown in the pre-lab near Figure 2. Try making the lag pole faster (moving it to the left) or increasing the compensator gain. Either of these options should speed up the response. Try to get the system to reach within 1% of steady state in less than 3 sec.

h. You can use Secure FX to move your data to your afs space. A shortcut is provided on the lab computer desktop.

C. ANALYSIS

1. Plots: Include root locus and Bode plots of the final compensated system. Import your final design and plant model into SISOTOOL. Export the closed-loop transfer function, and obtain a quick step response using the step command. (Use the syntax [Y,T] = step(T_r2y); to get the response and time vector into the workspace. Multiply y by the step input magnitude in counts. You can then plot the theoretical and experimental responses on the same axis). I have included a sample plot of satisfactory performance at the end of this handout.

D. REPORTING THE ADVENTURE

Submit one Adventure report per team. Each individual will receive the team grade for the adventure. Be sure your report includes, but is not limited to the following,

+ introduction, results/discussion, conclusion, and appropriate appendices
+ Plots of all closed-loop responses
+ Root locus and Bode plots generated during the design phase
+ Compare and contrast this controller performance with the lead/lag controller used in Lab 6
+ List any suggestions for improving the adventure.