ME 406 ADVENTURE 10

Diophantine Direct Design

Course Value: 20 points

Deadline: COB 10 / 12 Nov 2004

INTRODUCTION

When we have a precise transfer function model of the system, it is possible to directly design a dynamic prefilter / return path controller by solving the Diophantine equations. If the closed loop poles are chosen wisely, the theoretical performance of such a control system can rival that of the state feedback controller.

OBJECTIVE

The objective of this adventure is to:

+ Design a dynamic prefilter / return path compensator.
+ Implement the controller on the ECP hardware.
+ Plot the system root locus using Matlab
+ Prepare the lab worksheet during the lab period (no out of class analysis required).

A. PROCEDURE

For this lab, we will use the same setup as Lab 9. You should have the first cart configured with two 500g brass masses and the second one with four. A lumped parameter schematic is shown in the figure below.

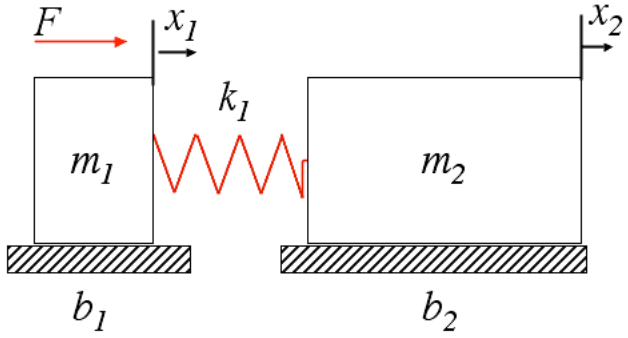

Figure 1: Plant Schematic

We will base our design on the transfer function model from the motor 'force' input to the position of mass 2 output. Here is the transfer function in terms of system parameters:

$$\frac{Y_2}{F} = \frac{Kk_1}{s\left(m_1 m_2 s^3 + \left(b_1 m_2 + b_2 m_1\right)s^2 + \left(k_1\left(m_1 + m_2\right) + b_1 b_2\right)s + k_1\left(b_1 + b_2\right)\right)} \qquad (1)$$

Recall that K is a static gain that makes the transfer function output scaled in units of counts. Be sure you have precise estimates of the system paramters from your previous work. A good system model is everything in this design methodology.

This time, we will use the dynamic prefilter / return path configuration. The overall block diagram is shown below.

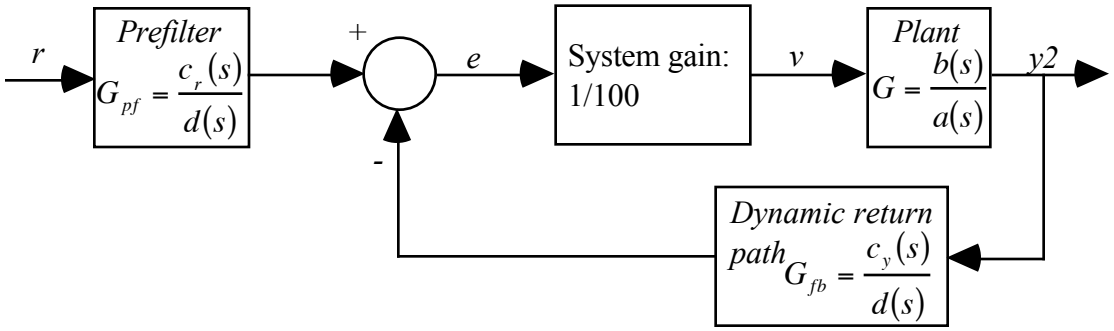

Figure 2: Closed-loop Block Diagram

The direct design method using rational transfer functions is described in the course text, section 7.10. I have also provided a Matlab function **diophanbtb.m** that will automatically design the dynamic return path transfer function. (Hey, after all it is 10th week.) The user only needs to provide the plant transfer function, and the desired closed-loop pole locations. Keep in mind that the total number of closed loop poles must be at least $2n$-1 where n is the number of open loop plant poles. I think it is worthwhile to review some of the content of section 7.10 here, in order to clarify the relationship between what you read in the textbook, what you compute using Matlab, and what you implement on the ECP apparatus. Note that I have included the polynomial descriptions directly from the textbook in Figure 2 above. What is not obvious from the textbook description is that the compensator denominator polynomial $d(s)$ can be moved to the left of the sum block. In fact this is necessary to make the prefilter and return path transfer functions proper.

Use the following steps to design full-order and reduced order compensators for the 2DOF system. You will need a 'good' closed-loop pole set. Since we are trying to match the performance of the pole-placement design of Lab 9, let's use the same closed-loop poles. The added complexity is that we must choose at least seven closed-loop poles. For the full-order design, use the closed-loop poles from sets 1 and 4 below.

Set 1	Set 2	Set 3	Set 4
$s = -7.533 \pm 27.28j$	$s = -8.7164 \pm 26.7j$	$s = -9.3632 \pm 26.0j$	$s = -9.6046 \pm 25.516j$
$-18.53 \pm 10.35j$	$-23.59 \pm 9.53j$	-35.1	-60.124
		-26.96	-22.841

1. Enter your plant transfer function (in terms of counts) into the Matlab workspace. Enter the eight poles from sets 1 and 4 into a single array called **ps**. Then by typing `C = diophanbtb(G_F2x2,ps)` at the Matlab prompt, you will obtain the feedback path transfer function C. Multiply the numerator coefficients by 100 to correct for the internal system gain. Write the result on the worksheet provided. (A1)

2. The purpose of the prefilter is to cancel out the unwanted (extra) plant poles. In the best case, some of the poles will be significantly faster than the others. These fast poles are called 'estimator poles'. Consider the poles of Set 4 to be the estimator or unwanted poles (they are only slightly faster--you are welcome to check this).

Determine the prefilter numerator polynomial (unscaled) using the poles of set 4, and the poly command. For instance: `cr = poly(ps(5:8))`. Write the polynomial on the worksheet (A2)

3. The prefilter denominator should be identical to that of the return path. In order for the output to match the reference signal, the prefilter must have the same static gain as the return path. In other words, the zeroth order coefficient of the prefilter numerator should be equal to the lowest order coefficient in the return path numerator. Use the following steps to extract the return path denominator polynomial, and scale the prefilter accordingly.

```
[num,den] = tfdata(C);  % extract numerator and denominator coefficients
den = den{:}            % convert from cell to array
num = num{:}
cr = num(length(num))/cr(length(cr))*cr % scale the prefilter numerator...
```

Now write down the scaled prefilter transfer function on the worksheet (A3). Note that the lowest order coefficient of the prefilter numerator should be equal to the lowest order coefficient in the return path numerator. The denominator of the prefilter should be identical to the return path denominator.

4. Import your plant model, return path, and prefilter transfer functions into SISOTOOL as G, H, and F, respectively. On the root locus plot, zoom in so that you just capture the compensator poles and zeros. Uncheck Open-Loop Bode under the View menu. Use File->Print to Figure to export the system root locus. Print the root locus, and attach it to your worksheet (A4) (you are, of course strongly encouraged to first copy and paste the figure to Word, and make sure it looks good in Print Preview).

5. Configure the ECP software for a closed-loop step of 1500 counts with a dwell time of 3000ms.

6. Implement your design on the ECP apparatus. Select Setup->Control Algorithm. Press the 'Dynamic Prefilter / Return Path' radio button and press 'Setup Algorithm'. Carefully enter the gains--refer to the block diagram provided on the setup dialogue to correlate your transfer function polynomials to the gains as specified by ECP. Note that the coefficients are listed from lowest to highest order. Also note that you only need enter the denominator gains once (the set-up assumes that you will use the same denominator in both compensator transfer functions.) Be sure to select the Encoder 2 radio button (we are controlling the position of mass 2). 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.

7. Obtain the closed-loop response, and plot Commanded Position, Encoder 1 Position, and Encoder 2 Position. For best results, export the data to Matlab, make a plot that meets the course graphical standards, copy and paste to word, and print. Attach the plot to your worksheet.

Repeat steps 1-7 above for a reduced order design. Use the following modifications:

1. Delete the slower, real pole from Set 4 (-22.841). If you have the poles entered in order, you can use simply `C = diophanbtb(G_F2x2,ps(1:7))` to design the feedback TF. Rescale the numerator coefficients, and write down your result on the worksheet (B1).

2. The prefilter numerator will need only cancel the remaining three poles of Set 4:

`cr = poly(ps(5:8))`. Write the result on the worksheet (B2).

3. The code given in step 3 should work for this case as well. Write down the scaled prefilter transfer function on the worksheet (B3).

4. Print the root locus, and attach it to your worksheet (B4).

5, 6. No changes.

7. You are encouraged to modify the prefilter static gain in order to reduce the steady state error. For instance, if your initial step response settles out at 1400 counts, multiply all of the prefilter numerator gains by 15/14, and retry the step. Iterate this process as much as you deisre to eliminate steady state error. Write down your final prefilter transfer function on the worksheet (B5).

D. REPORTING THE ADVENTURE **Names**_____

Return to CM_____ _____

The following worksheet with specified attachments is the only deliverable for this lab.

A. Full-order design.

1. Write the Return Path transfer function below (corrected for units)

$G_{fb}(s) =$ ----------

2. Write the prefilter numerator polynomial (unscaled) below:

$c_r(s) =$

3. Write the prefilter transfer function (scaled) below:

$G_{pf}(s) =$ ----------

4. Print out the Root locus for your full order design. Did the design method result in complex compensator poles and/or zeros? What is the likelihood that you would have selected these compensator poles in a traditional (lead / lag or notch) design. Are any of the compensator poles / zeros in the right half plane? What kind of problems can this cause?

5. Print out the system response, and attach to this worksheet. Is the response comparable to your design from Lab 9? Why or why not? Comment specifically on the steady state error value.

B. Reduced-order design. **Names**_____

1. Write the Return Path transfer function below (corrected for units) _____

$G_{fb}(s) =$ ----------

2. Write the prefilter numerator polynomial (unscaled) below:

$c_r(s) =$

3. Write the prefilter transfer function (scaled) below:

$G_{pf}(s) =$ ----------

4. Print out the Root locus for your full order design. Did the design method result in complex compensator poles and/or zeros? What is the likelihood that you would have selected these compensator poles in a traditional (lead / lag or notch) design. Are any of the compensator poles / zeros in the right half plane? What kind of problems can this cause?

5. Print out the system response, and attach to this worksheet. Is the response comparable to your design from Lab 9? Why or why not? Comment specifically on the steady state error value.

Write down the final the prefilter transfer function (scaled) below:

$G_{pf}(s) =$ ----------