MCE441/541 Midterm Project
Position Control of Rotary Servomechanism

DUE: ~~11/08/2011~~ 11/15/11

This project counts both as Homework 4 and 50 points of the second midterm exam

1 System Description

A servomechanism is a feedback-controlled motion unit used when maintaining accurate mechanical positioning is important. Servos are installed in CNC lathes, robotic manipulators, aileron/flap/rudder/elevator deflection systems and many other mechanical systems.

In this project, we use a rotary servo unit extracted from a pen plotter. The device is shown in Fig. 1. A DC motor rated at 27.5 V has a small gear mounted on its shaft. A large gear is installed on a second shaft, which also moves a potentiometer (a resistance-based position sensor). The potentiometer has a range of 3 turns and a total resistance of 5kΩ. As discussed in class, a voltage proportional to shaft rotation is generated by powering the potentiometer with a constant DC voltage. The voltage between the wiper and ground (the sensor voltage) is an indication of angular position. In our setup, we power the potentiometer from the data acquisition card's 5V output (nominal).

In our system, the sensor voltage varies between 0 and 4.86 V. Zero corresponds to the clockwise stop of the potentiometer, 4.86V to the counter-clockwise stop. Before operation, we center the shaft at 2.43 V and call that the zero angular position.

The DC motor is driven from a power amplifier. The transfer function from amplifier voltage input to sensor voltage increment/decrement from 2.43 V is approximately:

$$\frac{\Delta V_{pot}(s)}{V_i(s)} = G(s) = \frac{73}{s(s+28)} \tag{1}$$

For simplicity, we regard the sensor voltage increment ΔV_{pot} to be the controlled output. A unit step voltage demand on the output would be, for example, equivalent to an angular position demand of 3/4.86 turns, or 222.22 degrees.

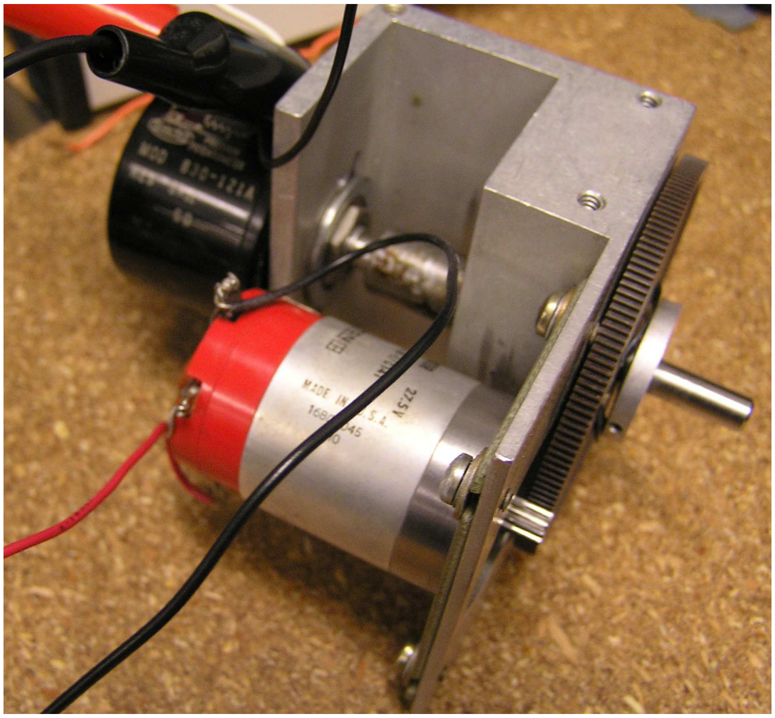

Figure 1: Rotary Servo System

Sensor voltage is read into the computer by a data acquisition card at a rate of 2000 samples per second. The operator enters the desired rotation setpoint from center as a voltage, using (4.86/3) V/turn as a conversion constant. Control computations occur within the computer, and result in a voltage V_i to be applied to the amplifier. This voltage is sent by the same data acquisition card, at the same rate. Fig. 2 shows the DAQ terminal board and Fig. 3 shows the complete system setup.

Design Specifications based on step input.

1. Settling time less than 0.15 seconds.
2. Less than 10 % overshoot.
3. Zero steady-state error.
4. Maximum $V_i = 6\text{V}$ when the input is a step rotation of 90° from center.

You will use a lead/lag controller structure and the root locus to solve this problem. The control transfer function is

$$K(s) = k\frac{s+z}{s+p} \tag{2}$$

Pre-Design Analysis

1. What is the system type? What does this imply for the steady-state error specification?

Figure 2: DAQ Breakout Box

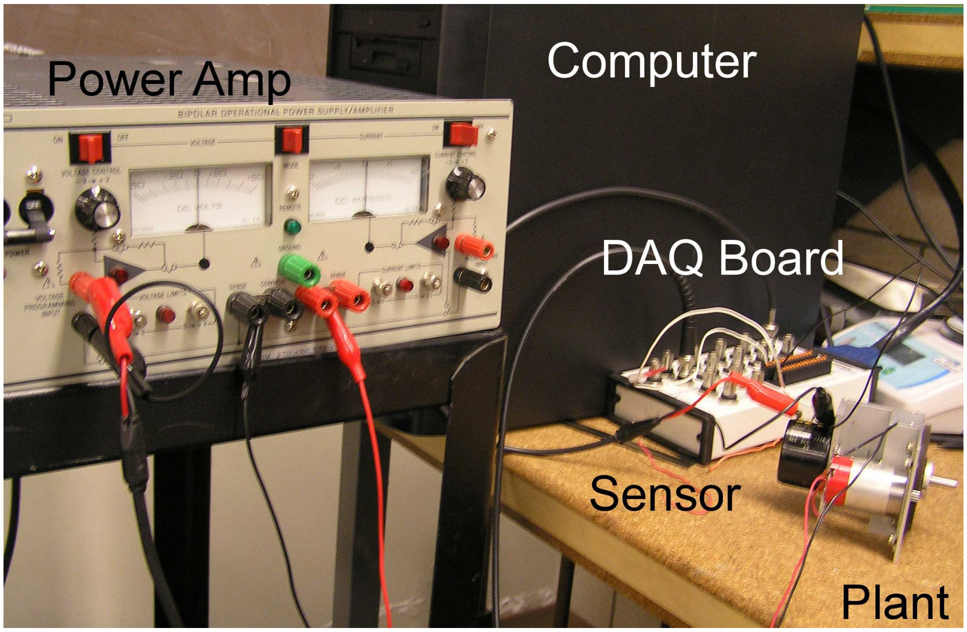

Figure 3: Overall Rotary Servo Control System

2. Obtain a hand sketch of the root locus for 2 cases, $z > p > 28$ and $p > z > 28$.
3. (MCE541 only) Find the center of asymptotes as a function of z and p for both cases and figure out which arrangement can yield a faster settling time.
4. Select either $z > p > 28$ or $p > z > 28$ and superimpose the trapezoidal target region for the closed-loop poles to meet the PO and Tset specifications.

SISOtool tuning

Follow this procedure:

- Create a transfer function object for the plant (Eq. 1).
- Type `sisotool` at the Matlab prompt. Click on System Data in the Architecture tab. Import the plant TF from the workspace and edit the F field (default 1) to the value 0.4050. We do this to apply a 90 degree input (0.4050 V) instead of 1 V, which would correspond to 222.22 degrees. Click OK and go to the SISO Design Task Window. You should see the root locus of the plant only (one pole at zero and one pole at -28).
- Click on the Analysis tab and then on Response to Step Command. A window should pop up showing the response for the sensor voltage in blue, going to 0.4050, and the corresponding control voltage V_i, a green line starting at 0.4050 and going to zero.
- Why does the green line start at 0.4050? Explain clearly.
- It's better to have the sensor voltage response in one plot and the V_i response in another plot. For this, click on Analysis/Other Step Responses. Set Plot 2 to Step and change the check mark for plot 2 from "Closed loop r to y" to "Closed loop r to u". You should get two separate plots now.
- Add a real pole and a real zero, both to the left of the plant pole at -28, using the red icons at the top.
- Proceed by trial-and-error, sliding the positions of the pole and the zero, and adjusting the gain k (slide the pink square). Your goal is to achieve the settling time and percent overshoot without exceeding 6 V for V_i (a slightly higher voltage should still be tolerated).
- Once you find a suitable controller, go to the compensator editor and write down its exact values. In SISOtool, the compensator appears as

  $$C(s) = k_c \frac{\tau_z s + 1}{\tau_p s + 1}$$

  Factor τ_z from the numerator and τ_p from the numerator and combine with k_c so that the control transfer function looks like Eq. 2.

Post-Design Analysis

1. Find the closed-loop transfer function using the final values of k, z and p.
2. Verify that the closed-loop system is stable.
3. What steady-state error is expected if the input command is an angular position ramp at a rate of 90 degrees per second?
4. (MCE541 only): If the realtime experiments show that the steady-state error to step inputs is not zero despite the plant being type I, how would you explain that? What can be done to reduce the sse?
5. Extract the settling time and percent overshoot from a step response of the closed-loop system.

Reporting

Type a short report around the following elements and information:

1. Answers to Pre-Design Analysis questions.
2. Screenshot of SISOtool final design root locus and step response
3. Table showing selected values for k, z and p in Eq. 2 and achieved PO and Tset.
4. Answers to Post-Design Analysis questions.
5. Be ready to provide your controller tuning settings for deployment in realtime during class.