5 Lab 5: Position Control Systems - Week 2

5.7 Introduction

In this lab, you will convert the DC motor to an electromechanical positioning actuator by properly designing and implementing a proportional (P) and a proportional-plus-derivative (PD) controller, both are variations of a proportional, derivative, and integral (PID) controller. An example of this positioning actuator is an automatic door of an accessible entrance.

This lab has three major components: analytical design, simulation, and implementation. The analytical design asks you to design each controller in the Laplace domain by simplifying the provided block diagrams and solving for unknowns. Next, you will simulate the design in Simulink to see if the gain values determined analytically meet the design specifications. Finally, you will implement these controllers in LabVIEW, test the gain values with the DC motor, and verify that your controller meets the specified design requirements.

5.8 Pre-lab

1. Repeat questions 1 through 4 of Section 5.4.1 of week one's LAB 5 manual with the values of K_m and τ_m that you obtained from the step response method in Lab 4.
2. Simulate the step responses of your new controllers for your motor transfer function. Use section 5.4.2 as a guide. **Note: The new saturation block limits need to be -10 to 10.** You will **NOT** have to include a noise source in your simulation.

5.9 Objectives

By the end of this lab, students will be able to:

- Design proportional, proportional + derivative, and proportional + speed controllers to certain specifications
- Implement and test the above controllers

5.10 References

See Section 7.10 and 9.8 of *Feedback Control Systems* (2011), Fifth ed. by Charles L. Phillips and John M. Parr. Note: the same information is in *Feedback Control Systems* (2000), Fourth ed. by Charles L. Phillips and Royce D. Harbor.

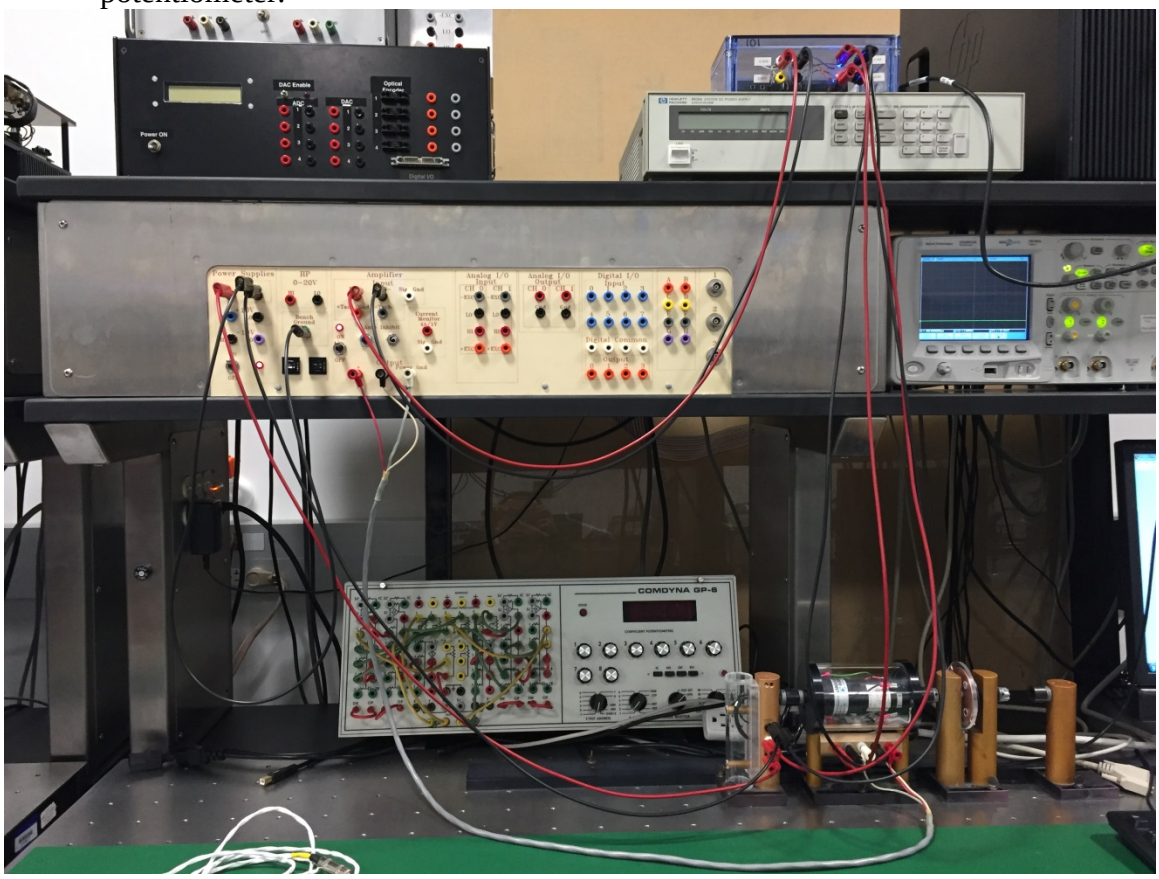
5.11 Lab Exercises

5.11.1 Proportional gain Control

In this part of the lab, you will be implementing the controller designs that were determined in the pre-lab. Figure 1 shows the root locus of a 2nd order system with poles at –1 and –2. If the gain of a proportional type controller is slowly increased from zero, the behavior changes from an over damped (both roots on the real axis of the root locus

plot), to critically damped (double/repeat roots on the real axis), and finally to an under damped system behavior (imaginary component of roots). These three system behaviors will be observed in this section.

- Turn on both power switches on the patch panel and the myRIO.
- Connect the myRIO analog output (AO) 0 to the amplifier input on the patch panel using one of the 3-wire cables with red, black, and white banana jacks. Then connect the output of the amplifier to the DC motor using another wire with red, black, and white banana jacks.
- Connect the 5V power supply from the patch panel to the potentiometer.
- Connect the tachometer to the myRIO analog input (AI) 0 and the potentiometer to the myRIO AI1.
- Use a thumb switch plugged into the “Amp Inhibit” jack on the patch panel to quickly turn the amp on and off.
- Connect the bench ground port on the patch panel to the ground of the potentiometer.



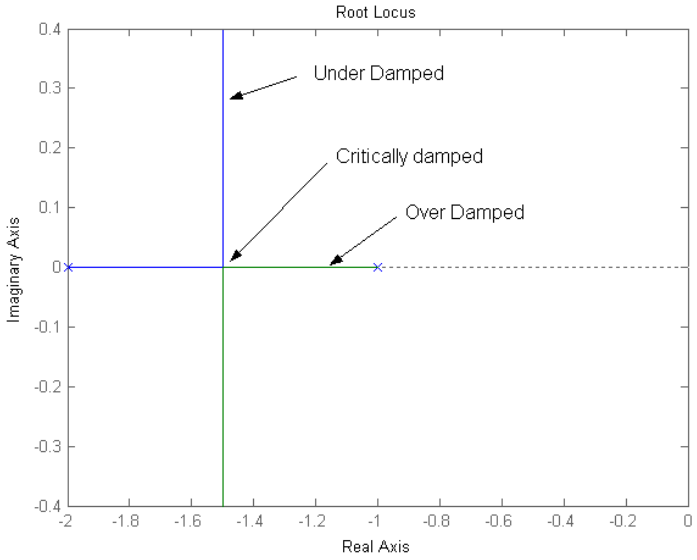


Figure 1: Root locus

- Now you need to build your proportional controller in LabVIEW.
- On the C:\ drive find the directory you created for Lab 4 or create a new directory that has your netID in it.
- In that directory create a new folder like "Lab5."
- Then Browse to N:\labs\se320\exp5\GE320 Starter and copy all the files and folders to your newly created folder.
- Open LabVIEW 2016.
- Open your local copy of GE320 Starter.lvproj
- Expand the files under the myRIO item and double click on "Main.vi" to open the Front Panel.
- Once the Front Panel open CRTL+e will open the block diagram. Note: CRTL+e can be used at any time to toggle between the front panel and block diagram windows.
- The chart and graph have already been set up for you on the Front Panel and several blocks have already been added to the Block Diagram. Note the gain connected to the output of the AI1 block; it is the same potentiometer gain that you calculated in lab 1. Why is this gain needed?
- You will need to add the following blocks to the starter block diagram:
    - **Pulse signal**: Control & Simulation -> Simulation -> Signal Generation
    - **Summation**: Control & Simulation -> Simulation -> Signal Arithmetic
    - **Multiplication**: Control & Simulation -> Simulation -> Signal Arithmetic
- You will need to add one block to the Front Panel:
    - **Horizontal Pointer Slide**: Numeric
- On the Front Panel, right click on the Horizontal Pointer Slide. Go to Visible Items and select digital display. This slider will allow you to adjust the value of K_p while the controller is running on the myRIO.
- Set K_p to the value you found in question 2 of the pre-lab.

- Connect the blocks as shown in Figure 2.
- Double click on the pulse signal and set the offset to 2.0, the amplitude to 1.0, and the period to 4 seconds.
- Show your TA before running the program.
- Press the arrow button to compile and run the controller.

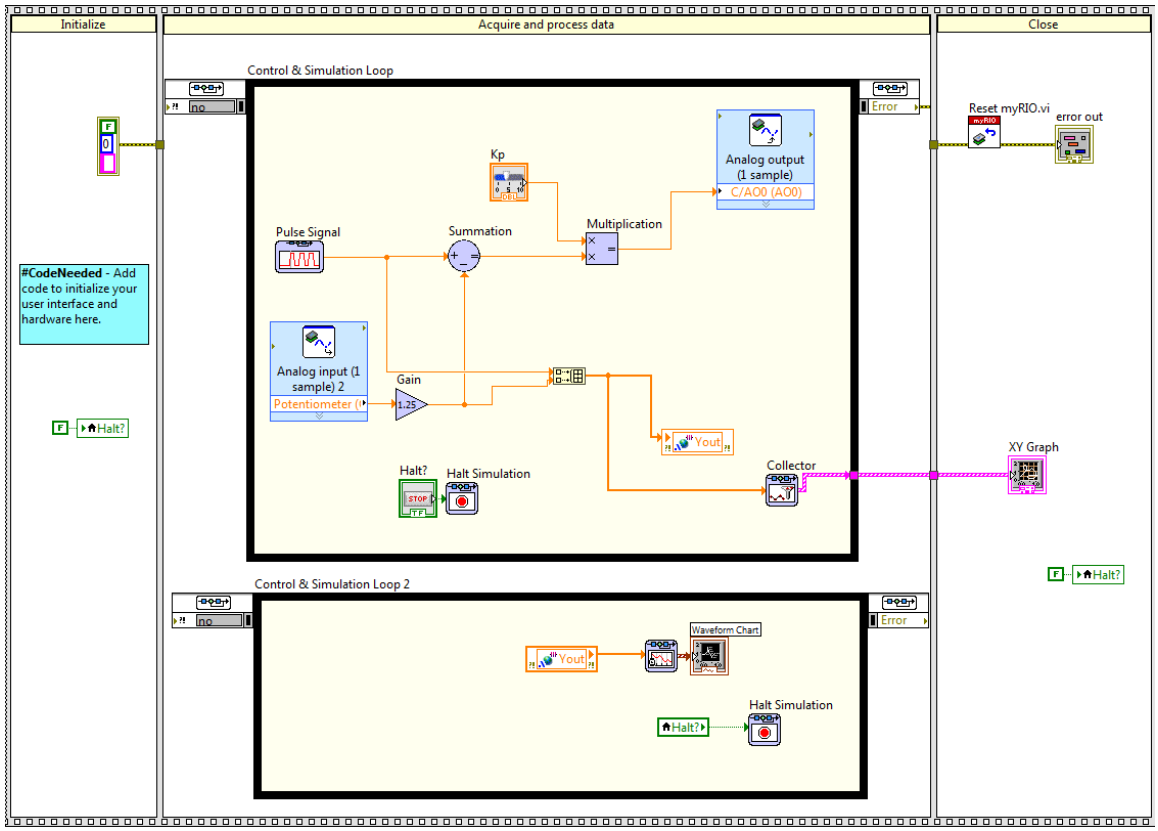


Figure 2: LabVIEW Block Diagram for Proportional Gain Controller

The motor will begin to step back and forth trying to follow the input waveform. You should notice the influence of friction. Is this an over damped, an under damped, or a critically damped system response?

In the next few steps, you will explore how changing K_p impacts the motion of the motor. The goal is to get the motor to step back and forth 90° to match the Pulse Signal reference. You will be comparing the results of your analytical solution (prelab question 2), the simulation, and the actual motor. Finding gains via simulation is common practice in control engineering because it is can be expensive or dangerous to test your design on the real system every time. The original design in simulation can get you close, but is not always perfect, so we still need to test on the system.

- Measure and record percent overshoot and settling time, T_s from the graph in LabVIEW using the cursors. Does this meet the design specifications? Why or why not? Compare the data in the LabVIEW graph with the data from your

Simulink simulation in the prelab. Are percent overshoot and T_s the same in LabVIEW and Simulink?

- Set the K_p value to 0.75. Record percent overshoot and the T_s. Does this meet the design specifications? Why or why not? How does it compare to simulation?
- Tune K_p to achieve a response with approximately 25% overshoot and record this gain the settling time. Compare to simulation.
  Note: You can move the slider on the Horizontal Pointer Slide while the controller is running. However, keep in mind you will only be able to use the cursors to measure data for the last 10 seconds of the simulation.
- Tune K_p to achieve T_s of approximately 300 ms and record this gain along with the percent overshoot. Compare to simulation.

5.11.2 Proportional-Plus-Derivative Control System

You will now replace the Proportional controller with a P-D controller similar to the one shown in Figure 3.

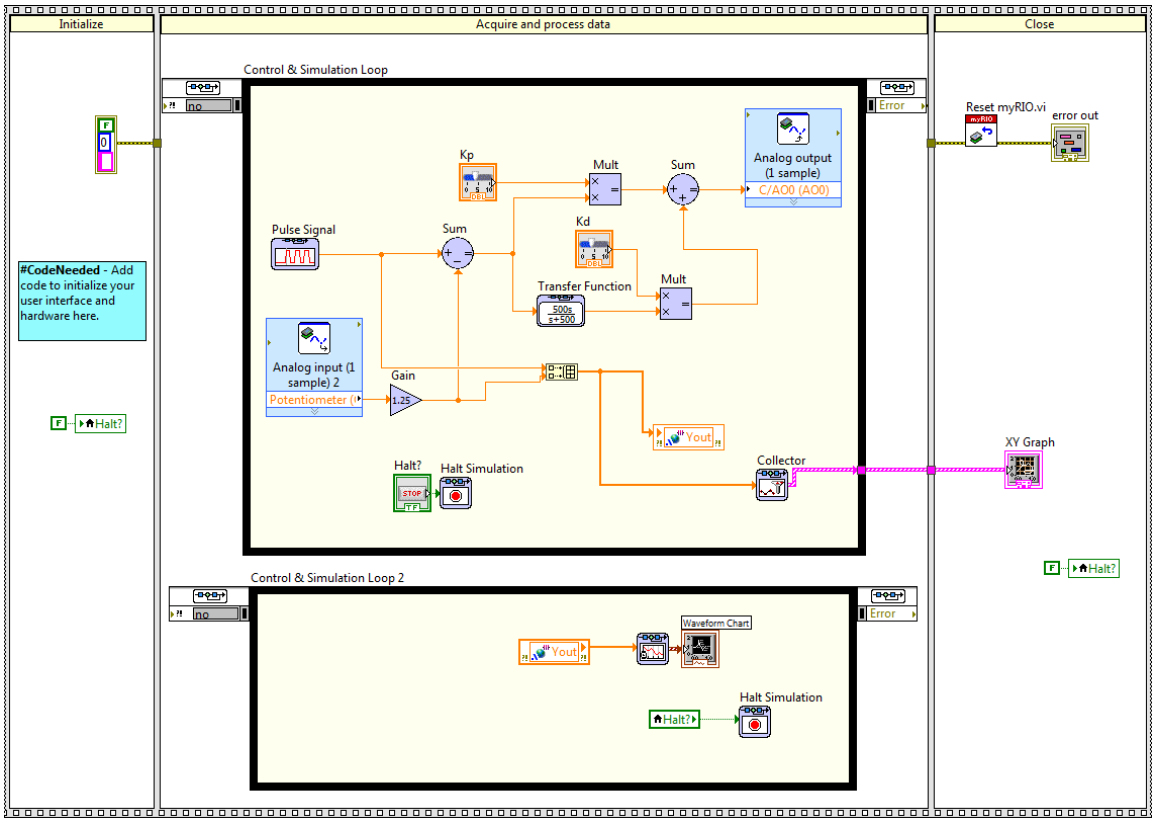


Figure 3: LabVIEW Block Diagram for the Proportional-Derivative Controller

- You will need to add the following blocks to the proportional controller block diagram (Note after each block name the path starting from the block diagram right click menu is described):
  - **Summation**: Control & Simulation -> Simulation -> Signal Arithmetic
  - **Multiplication**: Control & Simulation -> Simulation -> Signal Arithmetic

    - **Transfer Function**: Control & Simulation -> Simulation -> Continuous Linear Systems
      Note: the transfer function will serve as the derivative in this controller.
- You will need to add one block to the Front Panel (Note after each block name the path starting from the front panel right click menu is described):
    - **Horizontal Pointer Slide**: Numeric
      Note: there should be two sliders now, one for K_p and one for K_d
- On the Front Panel, right click on the K_d Horizontal Pointer Slide. Go to Visible Items and select digital display. This slider will allow you to adjust the value of K_d and K_p while the controller is running on the myRIO.
- Set K_p and K_d to the value you found in question 3 of the pre-lab.
- Go back to the block diagram and make the connections to match Figure 3.
- Double click on the transfer function and change the coefficients, so that your transfer function is:

$$\frac{500s}{s + 500}$$

- Show the TA before running the program.
- Click the arrow button to compile and run the controller
- With your values from question 3, did the performance of the motor satisfy the design constraints? Why or why not? Record the percent overshoot and settling time under the "Prelab Value" section of the data table. Compare to the simulation results for the same gain values.
- Adjust (if necessary) the K_p and K_d to meet the specifications. Record the percent overshoot and settling time under the "Experimental" section of your data table. Compare to simulation results with these same gains.
- Alter the values of K_p and K_d with small deviations. What makes this controller better or worse than the proportional controller?
- Set the value of K_p to 1.5 and K_d to 0.1. What do you notice about the effect (or lack thereof) of noise with a large derivative gain on the step response, and motor chatter? Record your observations in the data table.

5.11.3 Position and Speed Feedback Control System

Instead of differentiating the error for the derivative term in the control law, you will use the tachometer feedback scaled by a gain as the derivative term in your control. See Figure 4.

- Remove the transfer function block of the block diagram.
- You will need to add the following blocks to the proportional controller block diagram
    - **Gain**: Control & Simulation -> Simulation -> Signal Arithmetic
    - **myRIO Analog Input**: myRIO. In the Configuration window that automatically opens, select C/AI0 as the Channel.
- You will use the same Horizontal Pointer Slide from the derivative for the K_{p1} gain.

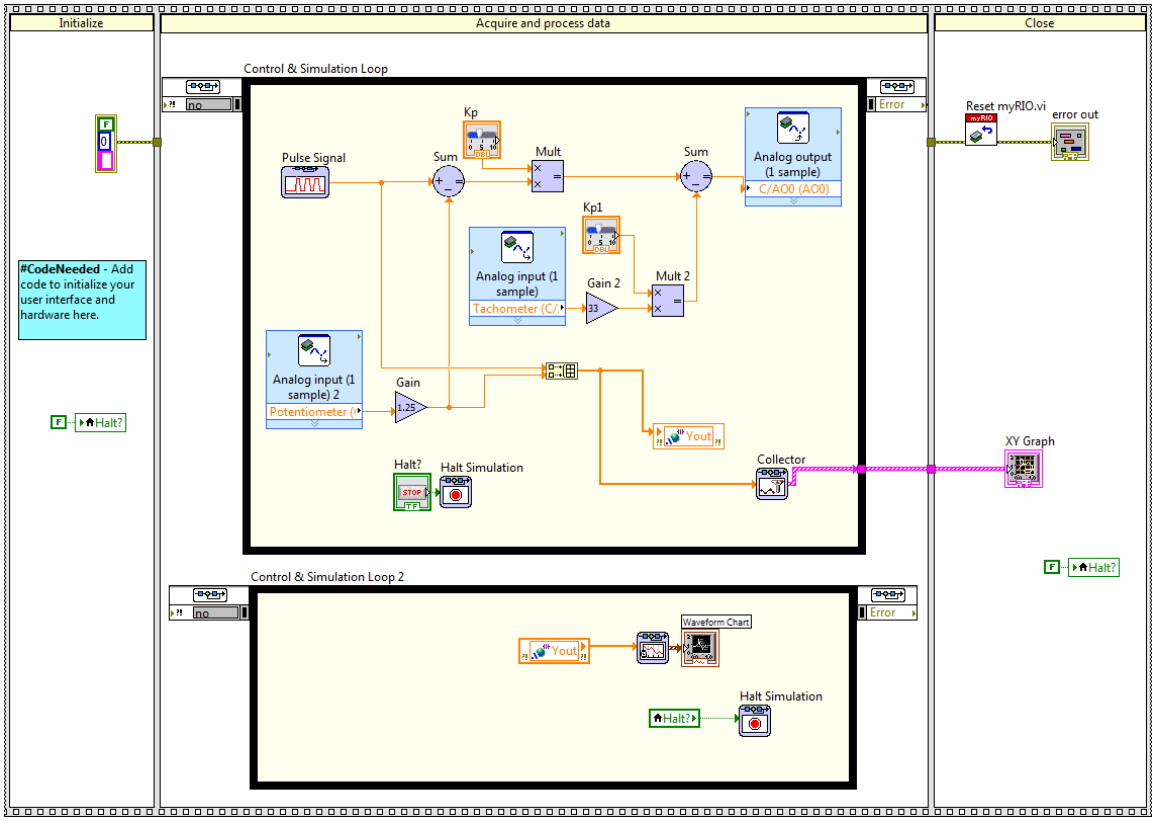


Figure 4: LabVIEW Block Diagram of the Proportional and Speed controller

- Connect the blocks to match Figure 4.
- Set K_p and K_{p1} to the value you found in question 4 of the pre-lab. You will need to multiply K_{p1} by the reciprocal of K_{tach} ($1/0.03 \approx 33$). Ask your TA for a value if you did not complete the pre-lab.
- Show the TA before running the program.
- Do these gains meet the requirements? How does it compare to simulation?
- Adjust the gains (if required) to meet the requirements. Compare to simulation results with same adjusted gains. Make note of the differences in the output as you adjust K_p and K_{p1}. What impacts does each gain have on the performance?
- Set the value of K_p to 1.5 and K_{p1} to 0.1. What do you notice about the effect (or lack thereof) of noise with a large velocity gain on the step response, and motor chatter?
- What makes this controller better or worse than the previous two?
- Record your final gain values in your data table.
- Turn off all power and disconnect all of the wires.

5.12 Post-lab

1. When implementing the P-Controller, what did you observe with steady state error when adjusting Kp from a smaller value to a larger value? What causes the steady state error?
2. Were there significant differences between your pre-lab controller gains and the values you obtained in the lab? Explain.
3. What are the performance differences between the controllers? What about their settling time? Percent overshoot?
4. How did your expectations for which controller would perform better in a noisy environment compare from simulation (week one) to the experiment (week two)?

5.13 Data Sheet

Proportional Gain Controller				
	Prelab	Experiment		
Kp		0.75		
% Overshoot			≈ 25 %	
Settling time				≈ 300 msec
Comparison to simulation's Overshoot and Settling Time				

Proportional + Derivative Controller			
	Prelab	Experimental	
K_p			1.5
K_d			0.1
% Overshoot			N/A
Settling Time			N/A
Comparison to simulation's Overshoot and Settling Time			N/A
Effects of noise, comments on motor chatter	N/A	N/A	

Proportional + Speed Controller			
	Prelab	Experimental	
K_p			1.5
K_{p1}			0.1
% Overshoot			N/A
Settling time			N/A
Comparison to simulation's Overshoot and Settling Time			N/A
Effect of noise, comments on motor chatter	N/A	N/A	