

1 Lab 5: Position Control Systems

1.1 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 (P-D) controller, both are variations of proportional derivative and integral (PID) control. An example of this positioning actuator is the automatic doors of an accessible entrance.

This lab has three major components: analytical design and test. The analytical design (pre-lab questions) ask you to design each controller in the Laplace domain by simplifying the provided block diagrams and solving for the unknowns. Then, you will implement these controllers in the lab, test the gain values with the real DC motor, and ensure your controller meets the specified design requirements.

1.2 Pre-lab

The pre-lab uses previous lab results as well as an understanding of the proposed closed loop system. Refer to the Implementation Diagrams in: Figure 2, Figure 3 and Figure 4 for guidance. Ensure to take into account all the gains!

1. From Lab 3 and Lab 4, we found the following transfer function for a DC motor:

$$\frac{\Omega(s)}{E_a(s)} = \frac{K_m}{\tau_m s + 1}$$

Using integration in the Laplace domain, derive the transfer function for motor position from the velocity transfer function above.

2. Place the position transfer function in a unity feedback loop with proportional gain controller. See Figure 1. Calculate the closed loop transfer function of this system. Hint: it should simplify into the form of an ideal second order system.

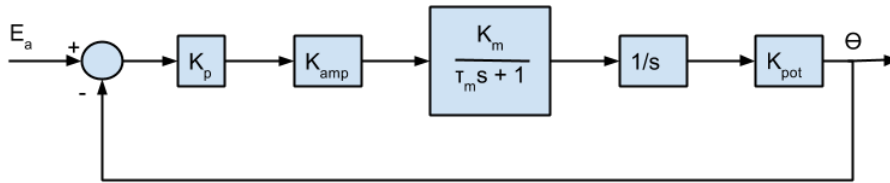


Figure 1: Proportional controller block diagram

Substitute in the values for K_m and τ_m from the step response in Lab 4. Then choose K_p (using Equation 1 and Equation 2) to satisfy the following designs with their constraints. If not possible, explain why:

Design 1:

- i. Peak overshoot < 10 % (Use Eq.1 below or text Fig.4.16)
- ii. Settling time < 800 ms

Design 2:

- i. Peak overshoot < 10 % (Use Eq.1 below or text Fig.4.16)
- ii. Settling time < 180 ms

Equation 1: Peak overshoot

$$M_p = e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}}$$

Equation 2: Settling time

$$t_s = \frac{4.6}{\zeta\omega_n}$$

3. Change the controller in the previous question to a P-D as shown in Figure 2. Calculate the closed loop transfer function and choose values of K_d and K_p so as to satisfy the same design constraints as in question 2, Design 2. Hint: the closed loop transfer function's denominator that you find should simplify into the form of an ideal second order system. There will be one zero in the numerator.

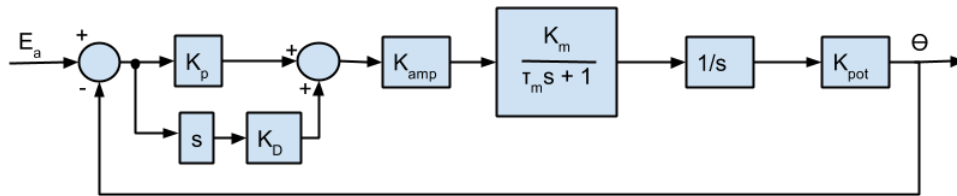


Figure 2: Proportional + Derivative Controller Block Diagram

4. Adjust the system to feedback the voltage from the tachometer as well as from the output potentiometer as in Figure 3. Then choose K_p and K_{p1} to satisfy the same design and the constraints from question 2, Design 2. Hint: the closed loop transfer function should simplify into the form of an ideal second order system.

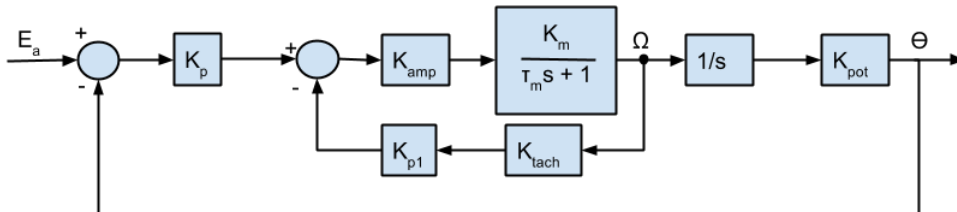


Figure 3: Proportional + Speed Controller Block Diagram

1.3 Objectives

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

- Design proportional, proportional + derivative, and proportional + speed controllers to a specification
- Implement and test the above controllers

1.4 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.

1.5 Lab Exercises

1.5.1 Proportional gain Control

Figure 7 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 of the root locus plot on the real axis), to critically damped (double/repeat roots on the real axis), and finally to an under damped system behavior. 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.

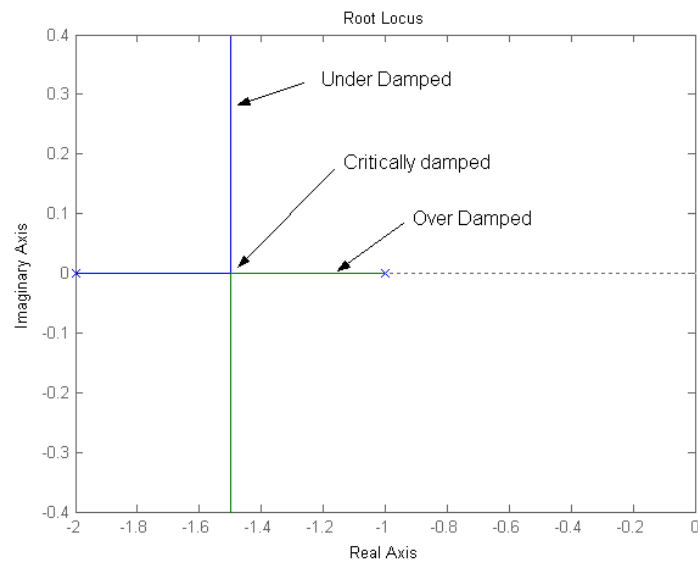
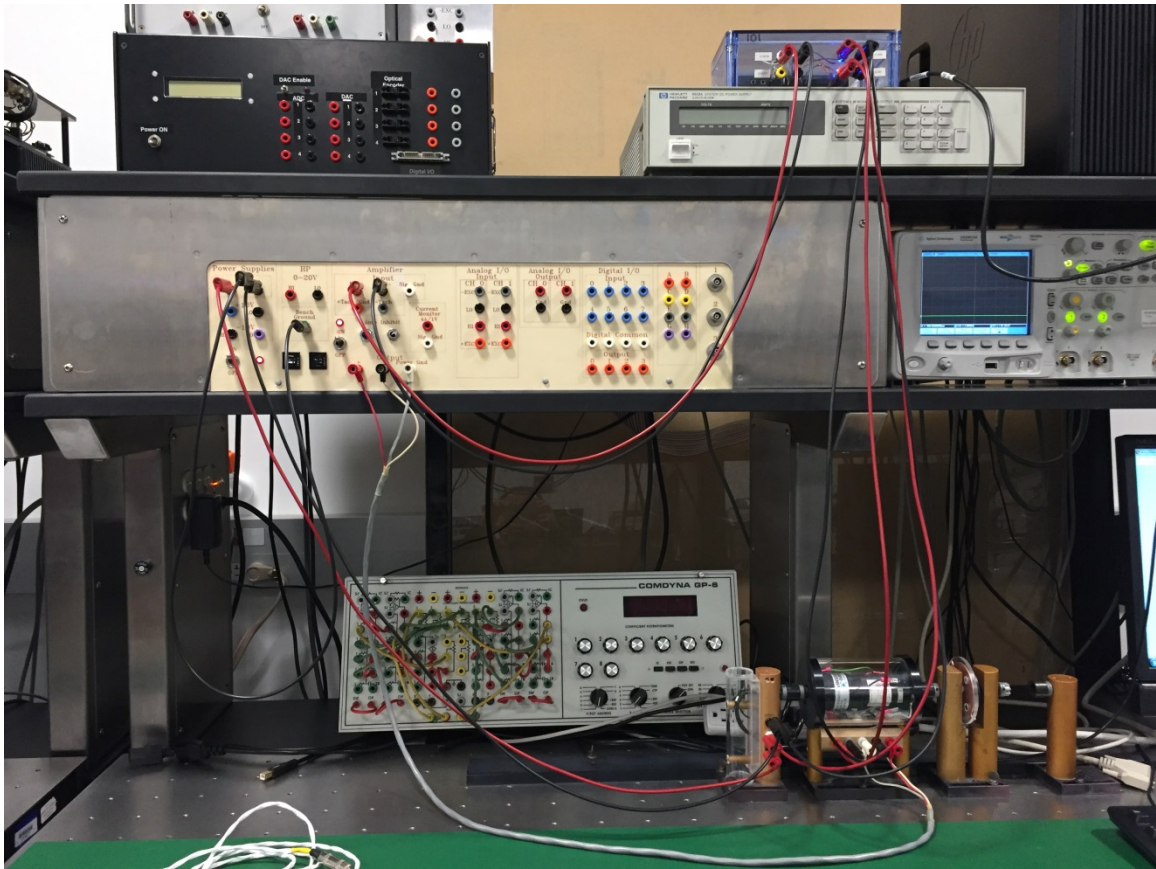


Figure 4: Root locus

- 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\ge320\exp5\SE320 Starter and copy all the files and folders to your newly created folder.
- Open LabVIEW 2019.
- Open your local copy of SE320 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 CTRL+E will open the block diagram. Note: CTRL+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 A/D 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 (Note after each block name the path starting from the block diagram right click menu is described):
 - 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 (Note after each block name the path starting from the front panel right click menu is described):
 - 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. Ask your TA for a value if you did not complete the pre-lab.
- Connect the blocks as shown in Figure 8.
- Double click on the pulse signal and set the offset to 3.0, the amplitude to 1.0, and the period to 4 seconds.
- Show your TA before turning on.
- Press the arrow button to compile and run the controller.

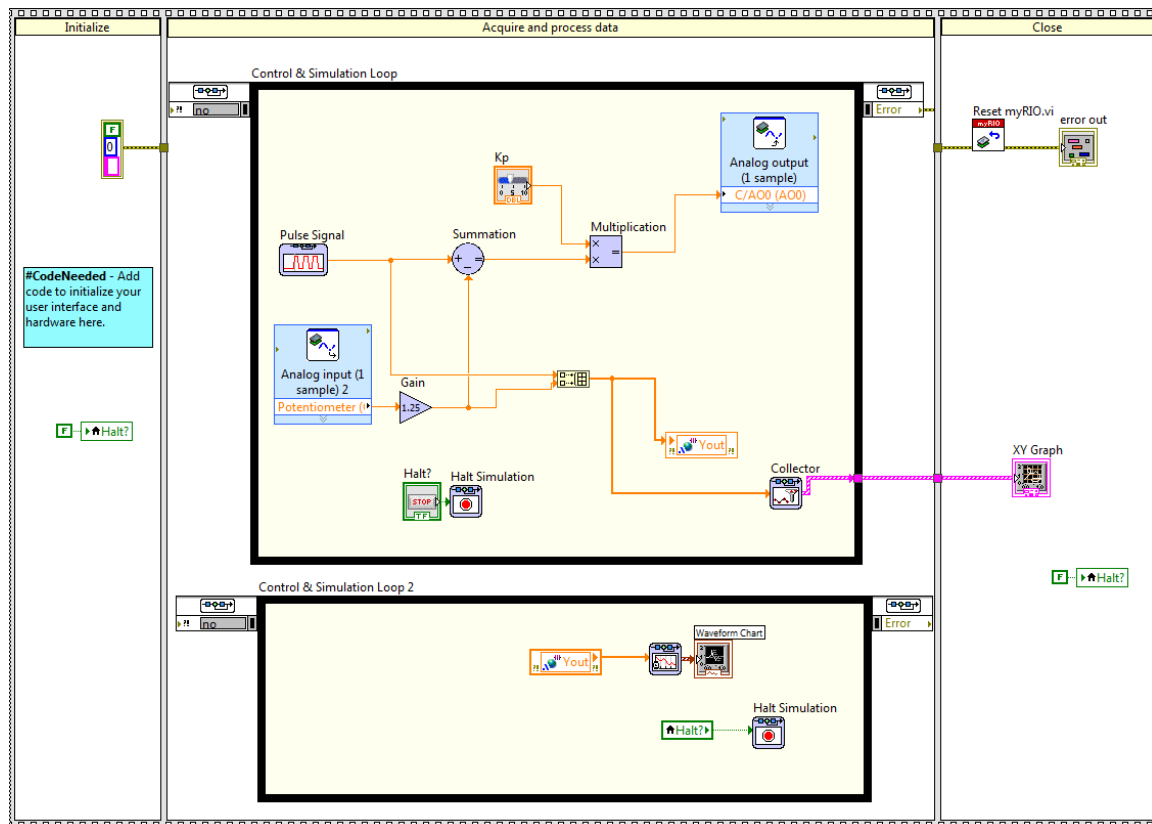


Figure 5: LabVIEW Block Diagram for Proportional Gain Controller

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), and the actual motor.

- Measure and Record % Overshoot and settling time, T_s under the “Prelab Value” section of the data table. Does this meet the design specifications? Why?
- Set the K_p value to 0.75. Record % Overshoot and the T_s . Does this meet the design specifications? Why?
- Tune K_p to achieve a response with approximately 25% overshoot and record this gain and the settling time under the “Experimental” section of your data table. 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.

1.5.2 Proportional-Plus-Derivative Control System

You will now replace the previous controller with a P-D controller similar to the one shown in Figure 9.

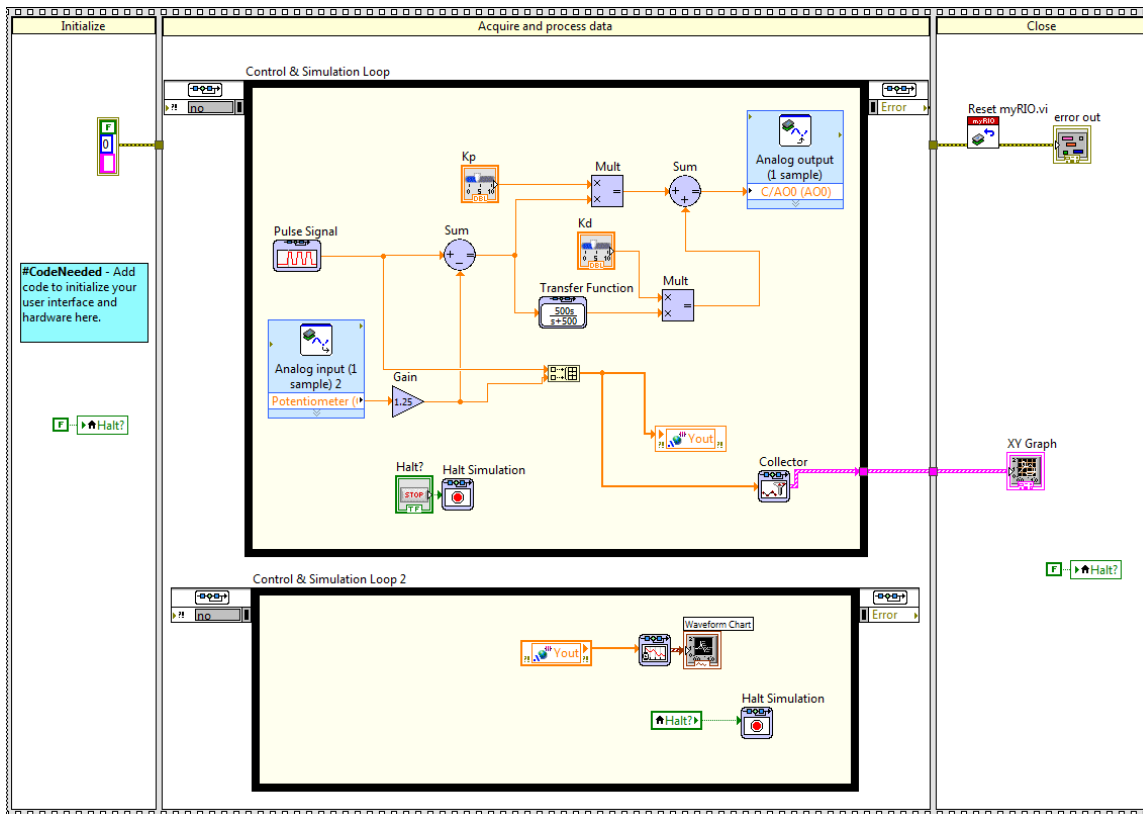


Figure 6: LabVIEW Block Diagram for the Proportional-Derivative Controller

- 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: this will function 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 numeric 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. Ask your TA for a value if you did not complete the pre-lab.
- Go back to the block diagram and make the connections to match Figure 9.

- 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.
- Click the arrow button to compile and run the controller
- With your values from question 3, did the performance satisfy the design constraints? Why? Record the % overshoot and settling time under the “Prelab Value” section of the data table.
- Adjust (if necessary) the K_p and K_d to meet the specifications. Record the % overshoot and settling time under the “Experimental” section of your data table.
- Perturb the values of K_p and K_d with small deviations. What makes this controller better / worse than the proportional controller? What is the fastest settling time you can get that still satisfies the % overshoot specification and does not produce a loud noise due to a large derivative gain?

1.5.3 Position and Speed Feedback Control System

Instead of differentiating the error to add a derivative term in the control law, you will use the tachometer feedback times a gain as the derivative term in your control. See Figure 10.

- Remove the transfer function block of the block diagram.
- 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):
 - 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

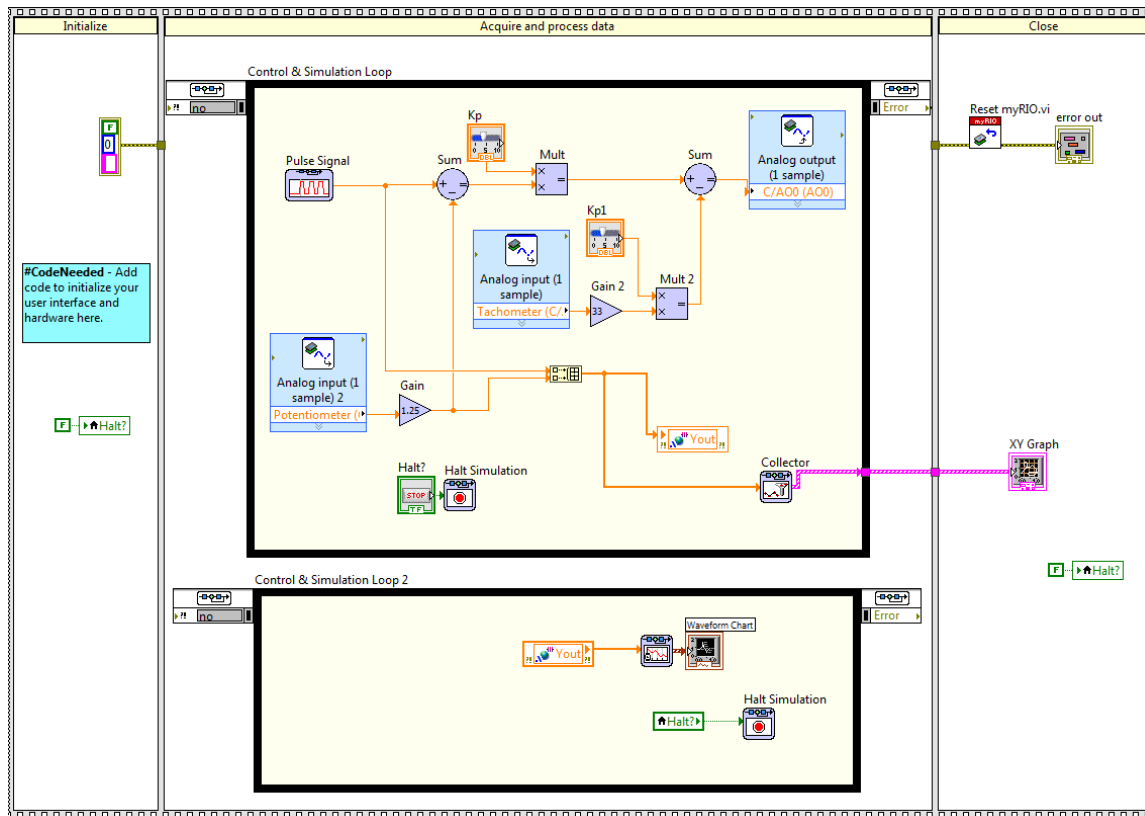


Figure 7: LabVIEW Block Diagram of the Proportional and Speed controller

- Connect the blocks to match Figure 10.
- Set K_p and K_{p1} to the value you found in question 4 of the pre-lab. Ask your TA for a value if you did not complete the pre-lab.
- Show the TA before turning on.
- Set the K_{p1} and K_p values to the pre-lab values (question 4) You will need to multiply the value in the prelab by 0.03 (K_{tach}) before setting the value in LabVIEW. Do these gains meet the requirements?
- Adjust the gains (if required) to meet the requirements. 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?
- What makes this controller better / worse than the previous two?
- Record all data in your data table.
- Turn off everything and unhook all of the wires.

1.5.4 Simulation (TO INCLUDE IN YOUR REPORT):

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. Note that, it is common that the design in simulation can get you close, but is not always perfect, so we still need to test on the system.

Using MATLAB's Simulink, create three block diagrams to simulate the systems in questions 2, 3, and 4 using the gains you calculated in those questions. In order to do so, start with loading the file "SE320_lab5starter.mdl" by typing its name in the matlab command prompt. This file can be found at the website: http://coecl.ece.illinois.edu/ge320/SE320_lab5starter.zip In addition, use Figures 1, 2, and 3 as a guide to create your Simulink simulations. Ensure that you include all the following gains and system components:

Motor Potentiometer Gain:	Use the value you found in Lab 1 (but also given to you in the figures below)
Signal Generator:	Use a square wave with a 0.25Hz frequency and amplitude of 1V.
Constant:	Offset the signal input by 2.5V.
Gains:	K_p , K_d and K_{p1} depending on the controller.
Amplifier Gain:	It has a gain of 2.4.
Derivative Approximation:	Use a transfer function from the Continuous set of blocks in the Simulink library.
Transfer function:	Use the Gain of 18 and time constant of 0.07. Set the Friction value to the value you found in Lab 3.
Saturation:	Limit the control input signal from -100 to 100.
Random Number:	Set Mean to 0, Seed to 0, Sample Time to -1 (inherited). Initially set Variance to 0.

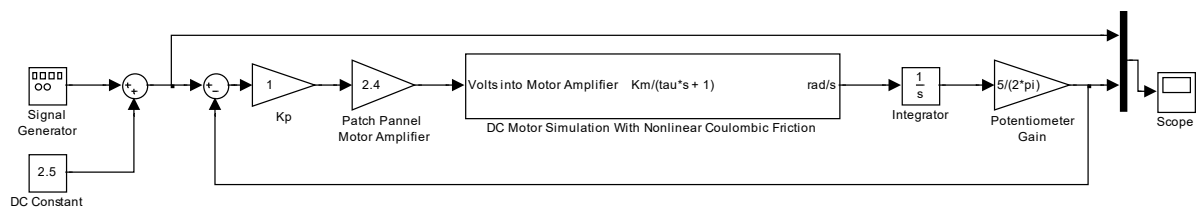


Figure 8: Proportional Gain Control Simulink Block Diagram

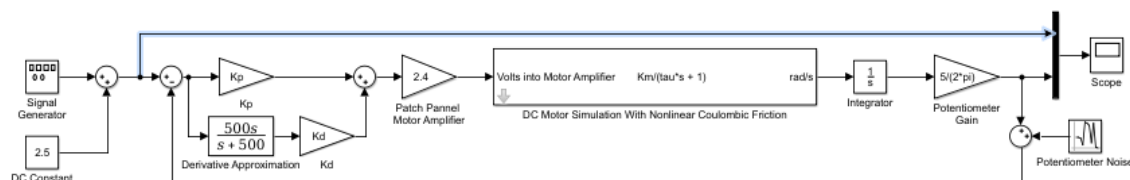


Figure 9: Proportional-Derivative Controller Simulink Block Diagram

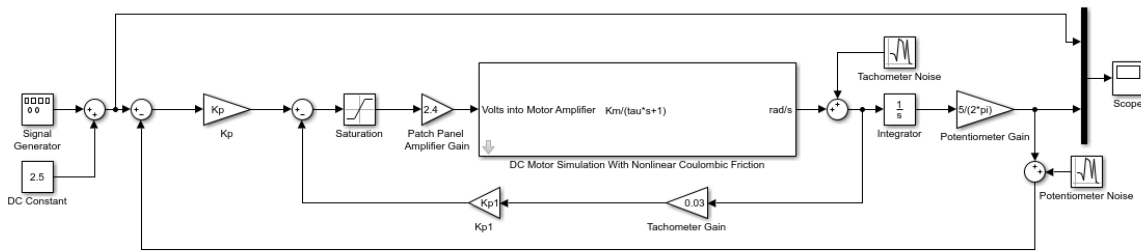


Figure 10: Proportional and Speed Controller Simulink Block Diagram

For each controller, use the gains you found in the experiment and recorded in the datasheet. Compare the simulation response to the experimental response. Do the simulated response meet the design specifications stated in the prelab.

Now we will examine the effect of noise on the performance of the Proportional-Derivative and Proportional + Speed Controller. In each random number block, set Variance to 0.01. Look at the scope of the output and compare the effect of noise on each controller. Experiment with different values for noise. Which controller seems to be more effected by the noise?

1.6 Post-lab

1. When implementing the P-Controller, what did you observe with steady state error when adjusting K_p 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. Which of these controllers do you see having problems in a noisy (electrical noise/interference) environment? Why?

1.7 Data Sheet

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

	Prelab Value	Experimental
K _p		
K _d		
% Overshoot		
Settling Time		
Comparison to simulation's Overshoot and Settling Time		

	Prelab Value	Experimental
K _p		
K _{p1}		
% Overshoot		
Settling time		
Comparison to simulation's Overshoot and Settling Time		