ME 360 Introduction to Control Systems: Lab #2

PD / PID Control, Steady State Error and Stability Analysis

In this laboratory, you will

1) design and implement PD and PID controllers,
2) conduct steady state error analysis
3) determine the stability limits for your design.

Every group will use separate controller design specifications, which have been posted on the last page of this instruction. It is expected that students will have designed their PD controllers and prepared their command trajectories before attending the first laboratory session. Students will be responsible for constructing their own real-time files and saving their data. The TAs will offer assistance as required. Every group must work independently. Communication between groups is not allowed.

The following timeline is recommended for completing this laboratory:

- **Pre-lab:** Design & simulate digital PD controller, determine PID controller stability limits, perform steady state error analysis.
- **In-Lab:** Experimentally implement PD controller, collect step response and trajectory tracking data. Experimentally implement PID controller, collect trajectory tracking data.
- **Final calculations, simulations and write-up:** Process recorded data, re-simulate PD response using experimentally recorded step and trajectory inputs. Incorporate experimentally identified Coulomb friction model into simulations, repeat tracking simulations for PD and PID controllers, compile results and prepare lab report.
- **Due date: 2 weeks after conducting the lab, beginning of Lab 3. (March 27th-31st)**

1 Pre-Lab Instructions

1.1 Proportional-Derivative (PD) Controller Design:

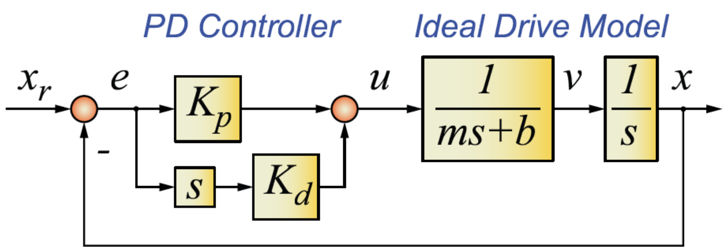

Figure 1: PD controlled servo system.

An ideal model of a PD controlled drive system is shown in Figure 1. x_r [mm] and x [mm] denote the commanded and actual (measured) position respectively. $e = x_r - x$ is the tracking error. The control signal u [V] is generated proportionally to the value of the tracking error, and its time derivative de/dt, through the proportional and derivative gains K_p [V/mm] and K_d [V/(mm/sec)] respectively. In the drive model, m [V/(mm/sec^2)] and b [V/(mm/sec)] represent the equivalent inertia and viscous damping.

1.1a. Determine the values of equivalent inertia (m) and damping (b). (hint: In Laboratory 1, the drive's velocity response was identified in the form $V(s)/U(s) = K_v/(\tau_v s + 1)$ which is equivalent to $V(s)/U(s) = 1/(ms+b)$.)

1.1b. Determine the values of K_p and K_d which result in the given specifications of natural frequency ω_n and damping ratio ζ for the closed loop poles. The specifications for each group have been posted on page 8.
<u>Hint:</u> Consider the resemblance between the closed loop transfer function denominator, and that of an ideal 2nd order system.

1.1c. Compare the closed loop <u>step response</u> (in Matlab) of the PD controlled system with an ideal 2nd order closed-loop model: $X(s)/X_r(s) = \omega_n^2/(s^2 + 2\zeta\omega_n s + \omega_n^2)$. Explain the similarities and differences between the two responses, elaborating with control theory and your engineering judgment. Comment on rise time and overshoot values in each case.

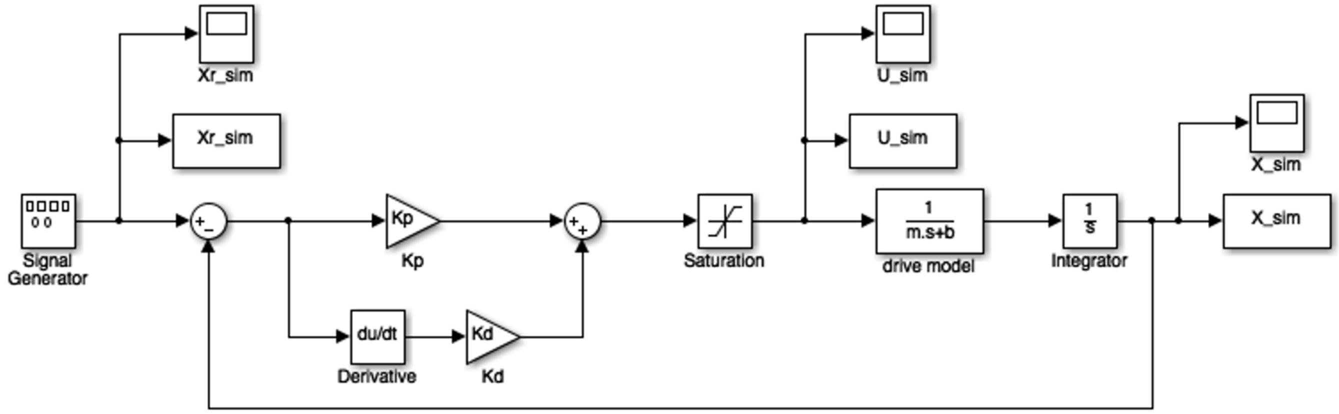

Figure 2: Simulink model for PD controlled servo system

1.1d. Implement your controller in simulation using a Simulink model similar to the one shown in Figure 2. The control signal cannot exceed a magnitude of 10 [V]. This is reflected by incorporating a *Saturation Block* with limits of -10 … +10 [V]. Also, the position measurements are obtained using an incremental encoder. Simulate the step response of the closed loop system to verify the stability of your implementation (you do not need to report this intermediate result).

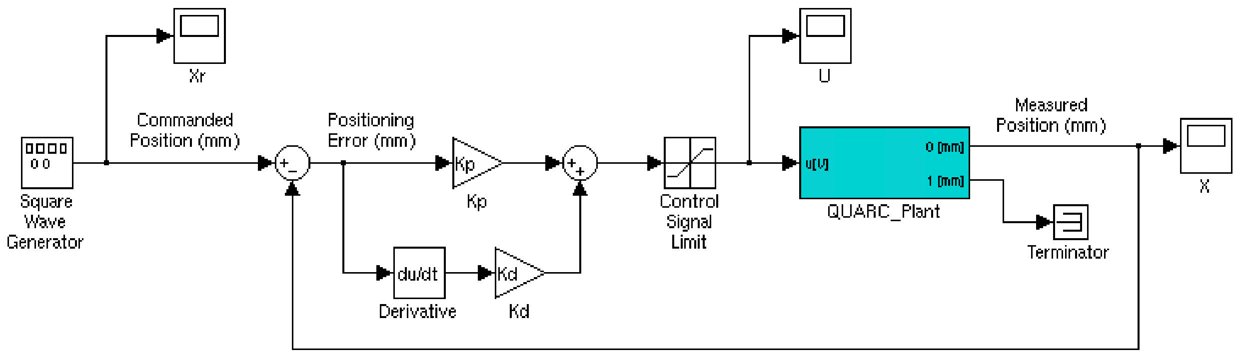

Figure 3: Real-time file for PD controller implementation.

1.2 Proportional – Integral – Derivative (PID) Control and Stability Analysis:

Integral action is frequently used in control systems to eliminate steady state tracking error. By adding an integrator with a gain K_i [V/(mm·sec)] to the PD controller, a Proportional – Integral – Derivative (PID) controller is obtained, as shown in Figure 4 (assuming that the integral gain is nonzero).

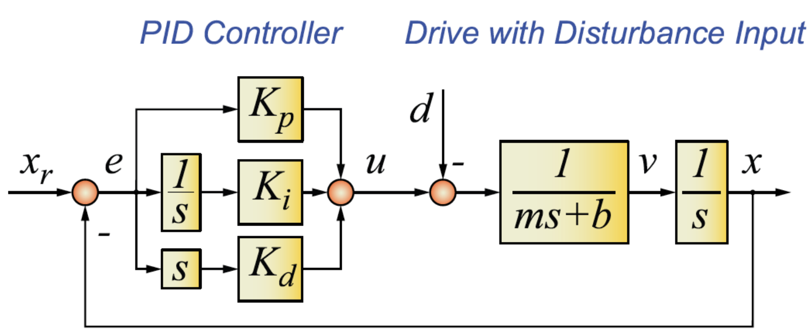

Figure 4: PID controlled drive system.

1.2a. Express the drive position x as a function of trajectory command x_r and disturbance input d (i.e. re-derive the tracking and disturbance transfer functions $G_{xr\to x}(s)$ and $G_{d\to x}(s)$).

1.2b. Derive the expressions for theoretical steady state error values e_{ss_traj} and e_{ss_fric} during constant velocity motion.

1.2c. Using Routh's Stability Criterion, derive the mathematical expressions for the PID gains K_p, K_i, and K_d which will guarantee the stability of the closed loop system in Figure 4.

1.2d. Using the PD gains you computed in Part 1.1b, determine the upper limit for the integral gain $K_{i\max}$, required to hold stability. Verify in simulation that an integral gain above this limit (for example $K_i = 1.2K_{i\max}$) will result in an unstable response. Report your simulation result for the smooth trajectory input in accordance with the following format:

1) Position trajectory: x_r and x versus t (on the top);
2) Tracking error: e versus t (in the middle);
3) Control signal: u versus t (on the bottom).

All time axes should be common, hence the signals need to be synchronized.

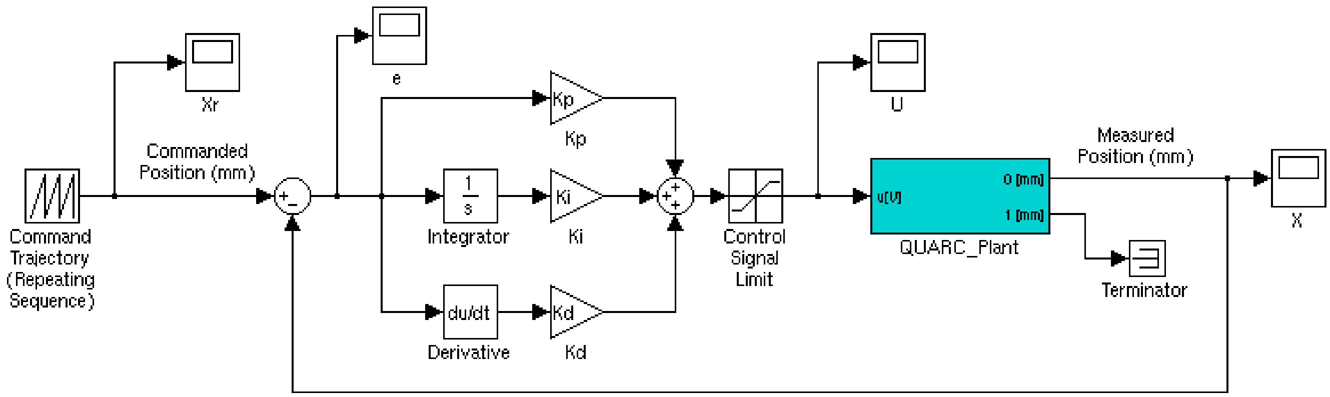

Figure 5: Real-time file for PID controller implementation.

1.2e. Simulate the tracking response of your PID controller for different values of the integral gain. State your observations on how the following error is affected as you modify K_i .($K_i = \alpha K_{i,max}$ $where\ \alpha < 1, \alpha = 1, \alpha > 1$). Report two or three characteristically different cases. Use the format given in 1.2d.

2 In-Lab Instructions

2.1 Proportional-Derivative (PD) Controller Design:

2.1a. Implement the PD controller on the experimental setup. You may modify the file, which you had used in Lab 1. Your implementation should look like the file shown in Figure 3. You may get help from the TA in compiling your real-time Simulink file. Test the closed loop step response by applying a square wave input with ±10 [mm] amplitude at 1 [Hz] frequency*. Capture 2.0 [sec] of data containing the command (x_r), output (x), and control (u) signals. Extract, plot, and save the necessary signals after you transfer them from the real-time scopes to the Matlab workspace.
(*Note: The amplitude was corrected from the previous +/- 1 mm. If you already recorded your data with +/- 1mm this is fine and will not affect your lab report results.)

2.1b. Apply the position command shown in Figure 6, consisting of a forward and backward movement with controlled accelerations, as a *Repeating Sequence* to your closed loop system. (Matlab script to generate this command will be posted on Learn) The displacement is 50 [mm], with 100 [mm/sec] velocity, and 500 [mm/sec^2] acceleration transients. There should be enough pause after each movement so that a forward and backward cycle is completed exactly in 2.000 [sec]. During the experiment, monitor the commanded and actual position, tracking error, and control signal profiles. Capture data for 4.0 [sec] and save the profiles for x_r, x, and u.

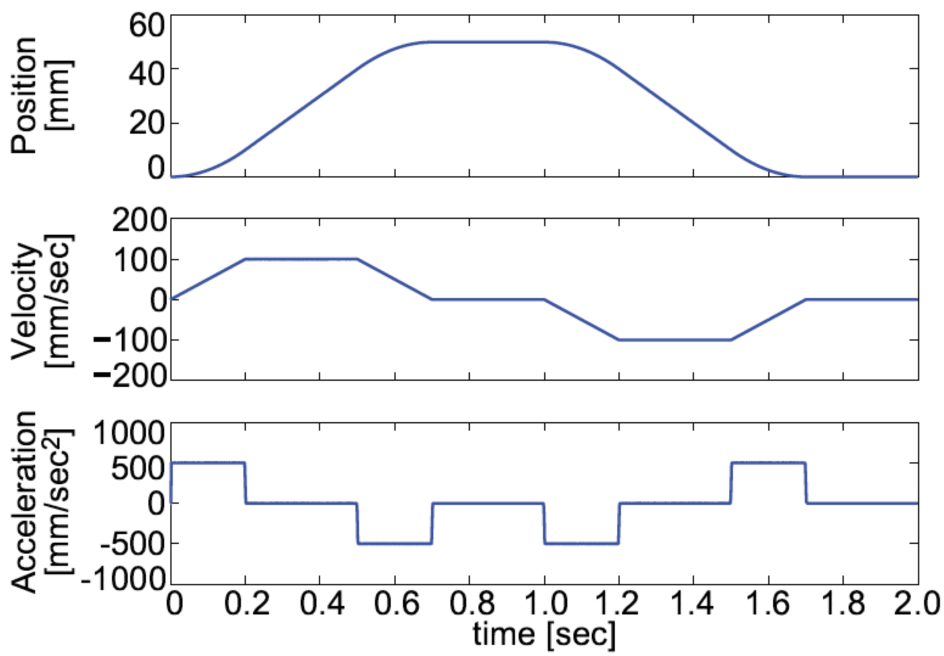

Figure 6: Command trajectory for tracking experiments.

2.2 Proportional – Integral – Derivative (PID) Control and Stability Analysis:

2.2a. Implement and test your PID controller on the experimental setup, as shown in Figure 5. To have a reasonable amount of integral action, while also guaranteeing stability, set $K_i = K_{i\max}/3$ as a rule of thumb. Use the smooth trajectory in Figure 6 as the input. Capture the profiles for x_r, x, and u for a duration of 4.0 [sec], as you did in to Part 2.1b. Using the commanded trajectory recorded during the experiment, and the same value for integral gain ($K_{i\max}/3$), conduct the tracking simulation. Show the simulation and experimental results on top of each other, as described in Part 1.2d. Elaborate on the similarities and/or differences between the two.

3 Post-Lab Instructions

3.1 Proportional-Derivative (PD) Controller Design:

3.1a. Apply the experimentally recorded position commands for the step input and smooth trajectory to your simulation (Figure 2). Present the results separately for each case. Overlay the simulated and experimental results on top of each other. Your graphs should be in same format as 1.2d. All time axes should be common, hence the signals need to be synchronized. Comment on the similarities and/or discrepancies between the simulation and experimental results. Support your comments with control theory and your engineering judgment.

Steady Error Analysis:

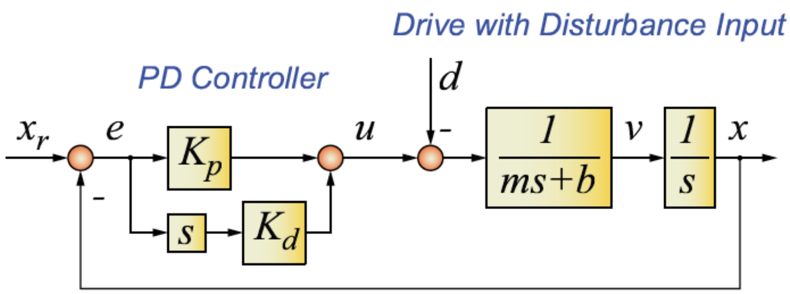

Figure 7: PD controlled system with disturbance input.

A more realistic drive model would also include the effect of friction as a second input, as shown in Figure 7. In the figure, d [V] represents the control signal equivalent disturbance, which mainly originates from the Coulomb friction in the drive mechanism, when other disturbance forces are not applied. Considering that the magnitude of Coulomb friction should ideally be constant (d_c [V]), the disturbance model can be expressed as:

$$d = d_c \cdot \mathrm{sgn}(v) \quad , \quad \text{where} \quad \mathrm{sgn}(v) = \begin{cases} +1 & \text{if} \quad v > 0 \\ 0 & \text{if} \quad v = 0 \\ -1 & \text{if} \quad v < 0 \end{cases} \tag{1}$$

3.1b. Express the drive position x as a function of both the trajectory command x_r, as well as the control signal equivalent disturbance input d in the form: $x(s) = G_{xr\to x}(s) \cdot x_r(s) + G_{d\to x}(s) \cdot d(s)$. In servo control literature, $G_{xr\to x}(s)$ is frequently referred to as the *tracking transfer function*, and $G_{d\to x}(s)$ is referred to as the *disturbance transfer function*.

3.1c. The performance of a controller is usually evaluated based on how small the following error can be made in the presence of various command and disturbance inputs. Express the tracking error e as a function of the command x_r and disturbance d in the form: $e(s) = G_{xr\to e}(s) \cdot x_r(s) + G_{d\to e}(s) \cdot d(s)$.

3.1d. Including the effect of Coulomb friction, determine the theoretical expression for steady state following error during <u>constant velocity motion</u>. *<u>Hint 1:</u> You will need to use the Final Value Theorem. <u>Hint 2:</u> Use the principle of superposition to break the steady state error into two components:* $e_{ss} = e_{ss\_traj} + e_{ss\_fric}$, where e_{ss_traj} represents the steady state error due to a constant velocity command and e_{ss_fric} represents the steady state error due to opposing Coulomb friction. Elaborate on how these error components are affected by the commanded velocity v_{ref}, friction magnitude d_c, drive parameters m and b, controller gains K_p and K_d, and the closed loop natural frequency ω_n [rad/sec].

3.1e. Carefully observe the experimental tracking result recorded in Part 2.1b, particularly the value of following error during constant velocity motion. Use the formulation developed in Part 3.1d to determine the average value of Coulomb friction (in terms of equivalent control signal) d_c [V] in your experimental setup. Explain your methodology in the calculations. Also indicate the theoretical value of steady state tracking error in the absence of friction.

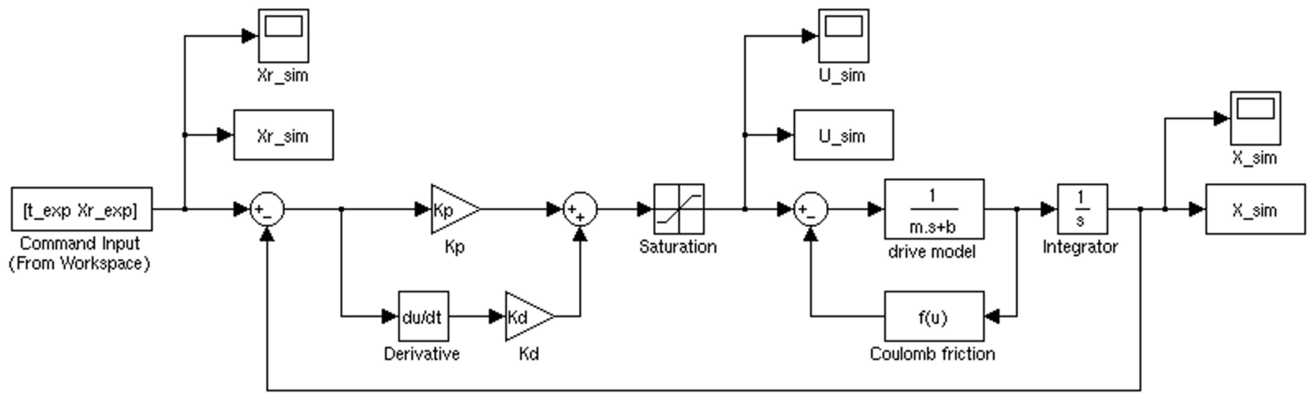

Figure 8: Simulink model of PD controlled system with Coulomb friction.

3.1f. Incorporate the effect of Coulomb friction into your simulation, as shown in Figure 8. You may use a *User Defined Function* in Simulink for implementing the friction model in Equation 1. Re-run your simulation using the experimentally recorded smooth command trajectory in Part 2.1b, and overlay the simulated and experimental tracking results once more (Part 3.1a). Comment on the similarity and/or discrepancy between the simulation and experimental results.

3.2 Proportional – Integral – Derivative (PID) Control and Stability Analysis:

3.2a. Overlay the experimental results obtained in Parts 2.1b and 2.2a on top of each other. Comment on the contribution of integral action to the servo performance. Does integral control provide better tracking and disturbance rejection? What are the limitations? You may refer to your calculations and mathematical derivations from Parts 1.2b, 1.2c, and 3.1d to back up your comments.

3.2b. Plot pole-zero maps of the PD and PID controlled closed loop tracking transfer functions. Comment on the pole and zero locations, also referring to natural frequency and damping ratio values. You may use the Matlab commands *pzmap()* and *damp()* to assist in your calculations. Indicate how the closed loop pole locations are affected with the introduction of integral action.

Report format and submission:
Every group will submit one report, in the online LEARN dropbox for Lab 2.

ME 360 Lab 2 - PD Controller Design Specifications

Do not forget to multiply your natural frequency (ω_n) by ‘2π’ to convert from [Hz] to [rad/s].

For example, if ω_n is given as 5 [Hz], it is equivalent to 10π [rad/s].

MON-8:30-1		MON-8:30-2		MON-8:30-3		MON-8:30-4		MON-10-1	
ω_n [Hz] 7.0	ζ [ ] 0.7	ω_n [Hz] 7.0	ζ [ ] 0.6	ω_n [Hz] 8.0	ζ [ ] 0.8	ω_n [Hz] 9.0	ζ [ ] 0.5	ω_n [Hz] 10.0	ζ [ ] 0.7

MON-10-2		MON-10-3		MON-10-4		TUE-8:30-1		TUE-8:30-2	
ω_n [Hz] 7.5	ζ [ ] 0.5	ω_n [Hz] 6.5	ζ [ ] 0.4	ω_n [Hz] 8	ζ [ ] 0.4	ω_n [Hz] 9	ζ [ ] 0.4	ω_n [Hz] 10	ζ [ ] 0.4

TUE-8:30-3		TUE-8:30-4		TUE-10-1		TUE-10-2		TUE-10-3	
ω_n [Hz] 9.5	ζ [ ] 0.7	ω_n [Hz] 8.0	ζ [ ] 0.6	ω_n [Hz] 9.0	ζ [ ] 0.8	ω_n [Hz] 10.0	ζ [ ] 0.5	ω_n [Hz] 6.5	ζ [ ] 0.7

TUE-10-4		THU-8:30-1		THU-8:30-2		THU-8:30-3		THU-10-1	
ω_n [Hz] 7.5	ζ [ ] 0.4	ω_n [Hz] 8.5	ζ [ ] 0.6	ω_n [Hz] 9.5	ζ [ ] 0.8	ω_n [Hz] 6.0	ζ [ ] 0.6	ω_n [Hz] 7.0	ζ [ ] 0.8

THU-10-2		THU-10-3		THU-10-4		FRI-8:30-1		FRI-8:30-2	
ω_n [Hz] 8.0	ζ [ ] 0.5	ω_n [Hz] 9.0	ζ [ ] 0.6	ω_n [Hz] 10.0	ζ [ ] 0.8	ω_n [Hz] 6.5	ζ [ ] 0.5	ω_n [Hz] 7.5	ζ [ ] 0.7

FRI-8:30-3		FRI-10-1		FRI-10-2		FRI-10-3		FRI-10-4	
ω_n [Hz] 10.0	ζ [ ] 0.6	ω_n [Hz] 8.0	ζ [ ] 0.7	ω_n [Hz] 8.5	ζ [ ] 0.5	ω_n [Hz] 8.5	ζ [ ] 0.8	ω_n [Hz] 8.5	ζ [ ] 0.4