ECEN 4638: Lab X.1PI

Rane Brown
Kate Schneider

March 3, 2016

Contents

List of Figures

1 Description

This lab will further explore the Torsional Disc System using Proportional-Integral control, with the goal of designing a PI-based cruise control system that works well for inertia changes of ±10%. The controller is then tested on two different Torsional Disc apparatuses to evaluate the design for robustness. The system setup will be similar to what was used in labX.1P; only the bottom disc of the TDS will be used and the four weights will be set at a radius of 6.5cm.

2 System Model

As in labX.1P, the Torsional Disc system can be modeled as an LTI system, with the below equation

$$J\dot{\omega} + c\omega = k_h u \tag{1}$$

Where $J =$ total system inertia, $\omega =$ velocity rad/sec, $c =$ system drag, $k_h =$ hardware gain, $u =$ reference.

2.1 Calculated Parameters

During labX.1P, each team calculated parameters c and k_h for one torsional disc system based on its experimental data. These parameters are listed in the table below.

System	c	k_h
1	0.0082	0.3577
1	**0.0079**	**0.3598**
2	0.0078	0.15
2	0.0081	0.376
3	0.0110	0.387
3	0.010	0.0084
4	0.0111	0.0084
4	0.0184	0.387

Table 1: Torsional Disc System Parameters

For a radius of 6.5 cm, the calculated total system inertia J is 0.0128 kgm^2.

For this experiment, our cruise control system was designed using the calculated parameters from Torsional Disc System 1 (in bold in the above table), and the PI controller was tested on TDS systems 1 and 4.

2.2 Transfer Functions

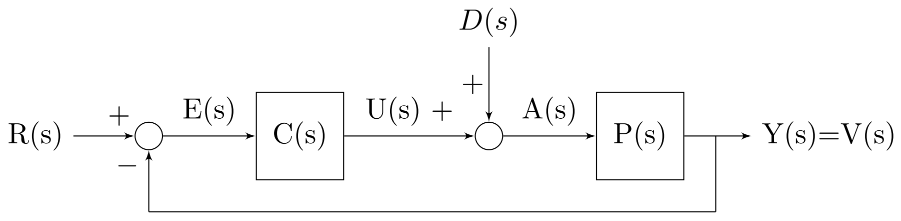

Figure 1: Cruise Control Feedback System

From our LTI model, we see that the system plant is modeled with the below equation.

$$P(s) = \frac{k_h}{Js + c} e^{-T_d s} \tag{2}$$

Where $J =$ total system inertia, $c =$ system drag, $k_h =$ hardware gain, and $T_d =$ time delay. In this lab, we use a PI controller, $C(s) = \frac{k_I}{s} + k_p$.

There are multiple transfer functions which are of interest in our system. The first transfer function of interest is the closed-loop transfer function from reference input R to output Y:

$$H_{yr} = \frac{Y}{R} = \frac{CP}{1 + CP} = \frac{\frac{K}{J}(k_p s + k_I)}{s^2 + (\frac{c}{J}k_p + \frac{c}{J})s + \frac{k}{J}k_I} \tag{3}$$

From the equation for H_{yr}, we can see that the DC gain of this transfer function is 1, as is to be expected for this system.

A second transfer function of interest is the closed-loop transfer function from reference input $R(s)$ to controller output $U(s)$.

$$H_{ur} = \frac{U}{R} = \frac{C}{1 + CP} = \frac{(k_p s + k_I)(s + \frac{c}{J})}{s^2 + (\frac{c}{J}k_p + \frac{c}{J})s + \frac{k}{J}k_I} \tag{4}$$

This transfer function has a DC gain of $H_{ur}(0) = \frac{c}{k}$. This value represents the input that is needed to compensate parameter c in steady state.

We can also look at the transfer function from the disturbance $D(s)$ to the output $Y(s)$, since again, rejecting disturbance input such as hills and bumps is important in a cruise control system.

$$H_{yd} = \frac{Y}{D} = \frac{P}{1 + CP} = \frac{\frac{k}{J}s}{s^2 + (\frac{c}{J} + \frac{k_p k}{J}s) + \frac{k}{J}k_I} \tag{5}$$

The final closed-loop transfer function of interest is H_{ud}, the transfer function from the disturbance $D(s)$ to the controller output $U(s)$.

$$H_{ud} = \frac{U}{D} = \frac{-CP}{1 - CP} = \frac{-\frac{k}{J}(k_p s + k_I)}{s^2 + (\frac{c}{J} - \frac{kk_p}{J})s + \frac{kk_I}{J}} \tag{6}$$

We will also examine the open-loop transfer function $L(s) = P(s)C(s)$, which will be useful in determining gain and phase margins of our system to ensure stability.

$$L(s) = P(s)C(s) = \frac{k}{Js + c}(\frac{k_I}{s} + k_p) \tag{7}$$

3 Matlab Analysis

3.1 Time Domain

Before implementing our PI controller on a physical TDS, we can formulate and test the design in Matlab to ensure the system meets certain time and frequency domain specifications. In the time domain, relevant specifications include rise time t_r, settling time t_s, and overshoot M_p. Choosing desired ranges for these parameters allows us to estimate ζ and ω_n for a simple second-order system of the form

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{8}$$

$$t_r = \frac{1.8}{\omega_n}, \qquad t_s = \frac{4.6}{\zeta\omega_n}, \qquad M_p = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}} \tag{9}$$

However, when using these estimates, we had to keep in mind that our PI controller also introduces an additional zero to the closed-loop transfer function $H_{yr}(s)$ which affects the time domain response of our system. Most notably this zero increases M_p and decreases t_r. Thus, our design aimed for a larger ζ, while ω_n could be decreased slightly while still meeting rise time specifications.

We first chose desired values of $t_r = 0.5$ sec, and $M_p = 5\%$, and estimated suitable ω_n and ζ for $H_{yr}(s)$ by hand calculation, then modeled the system in Matlab. Iterating on the design in Matlab allowed us to fine-tune the controller, which was then evaluated on the physical system. The specifications above result in the values $\omega_n = 3.6$, $\zeta = 0.69$ which give $k_I = 0.4611, k_p = 0.1548$. The step response plot for the resulting closed loop system H_{yr} is given below, as well as plots of the step responses of H_{er}, H_{yd}, and H_{ur}.

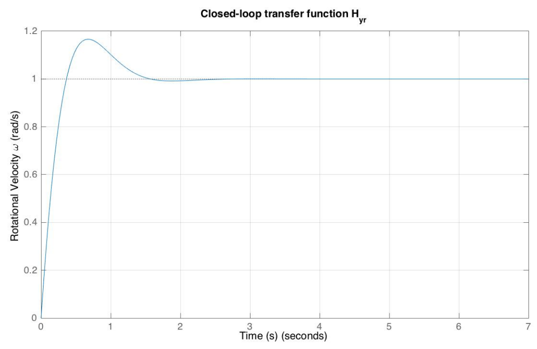

Figure 2: Step Response for H_{yr}

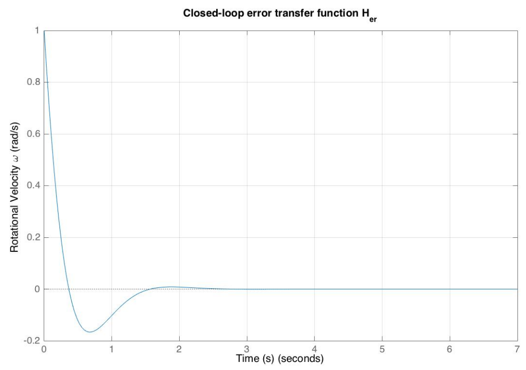

Figure 3: Step Response for H_{er}

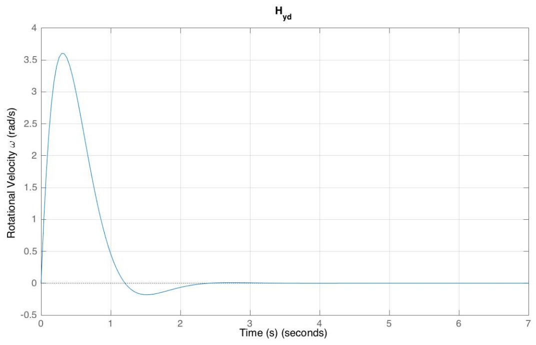

Figure 4: Step Response for H_{yd}

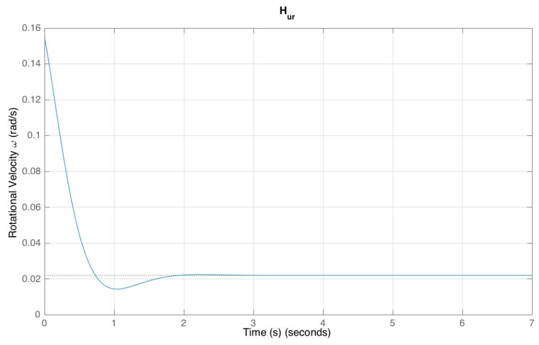

Figure 5: Step Response for H_{ur}

The additional zero results in a greater overshoot for $H_{yr}(s)$ and overall faster response than was predicted by hand calculation for all system transfer functions. In fact for H_{yr}, the specifications as calculated in Matlab are

$$t_r = 0.2749,\ \ t_s = 1.3725s,\ \ M_p = 16.649\%$$

While rise time is within our specifications, overshoot (M_p) is unacceptable. We can further refine our design (by choosing a greater damping ratio, for instance) now that we have observed the effect of the additional zero in Matlab simulations. For the next iteration, we try choosing $k_p = 0.233$ (and keep $k_I = 0.4611$) to increase the damping ratio (ζ). The resulting time domain plots are included below.

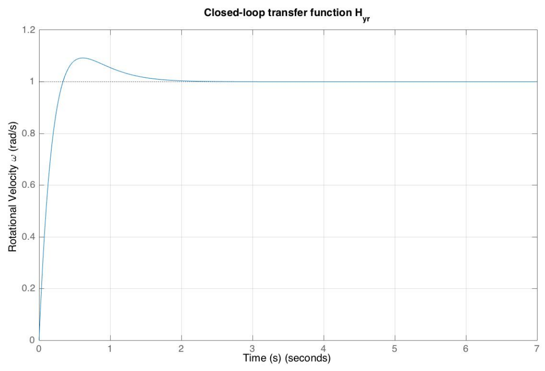

Figure 6: Step Response 2 for H_{yr}

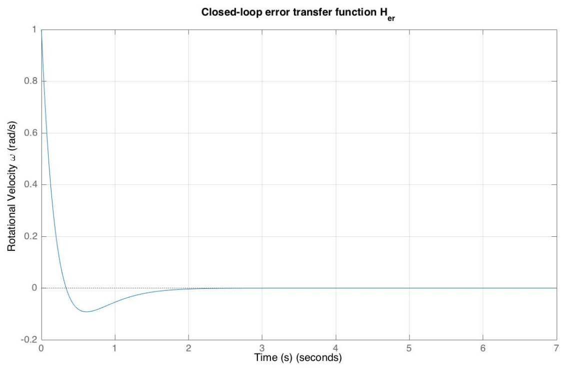

Figure 7: Step Response 2 for H_{er}

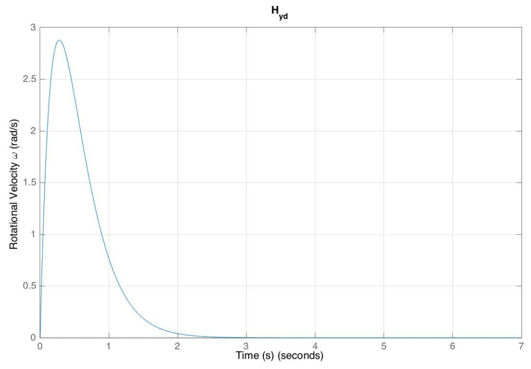

Figure 8: Step Response 2 for H_{yd}

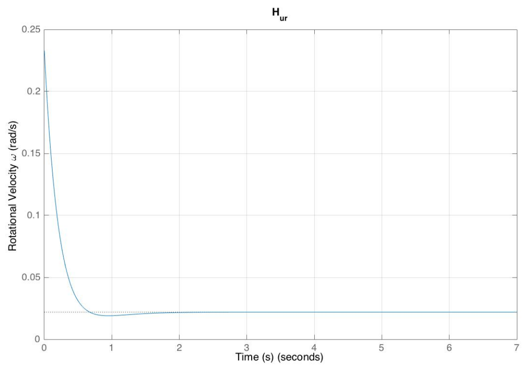

Figure 9: Step Response 2 for H_{ur}

Unfortunately, even while iterating within Matlab, we cannot meet the specification for overshoot requirement. The only two knobs we can adjust to place the poles of our system are k_p and k_I, and we cannot achieve $M_p < 5\%$ without causing the system to be overdamped. The second iteration of our system produced the specifications

$$t_r = 0.2369, \;\; t_s = 1.4108s, \;\; M_p = 9.1959\%$$

While this is an improvement over our first attempt at designing the controller, we still need more fine-grain control to achieve the specifications that define a precise and robust cruise control system. We can also simulate the system's response to different inertias. This was simulated for the second iteration of the controller in Matlab with system inertia $J \pm 10\%$, and the plots of the step response of H_{yr} are included below.

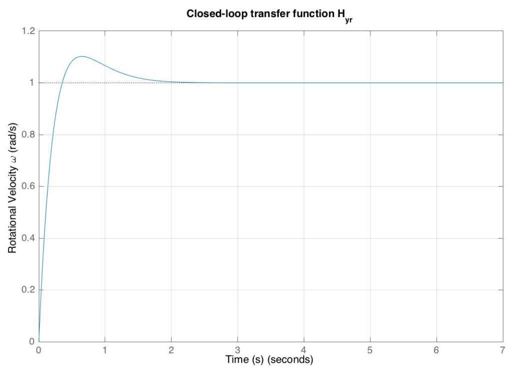

Figure 10: Step Response for $J + 10\%$

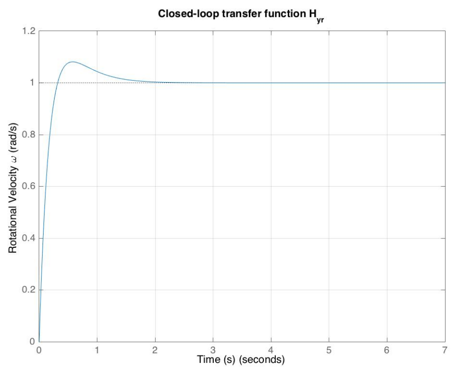

Figure 11: Step Response for $J + 10\%$

Rise time remained within the specification ($t_r < 0.5$ sec) while overshoot varied by $\pm 1.2\%$.

3.2 Frequency Domain

In addition to time domain specifications, certain aspects of a system's frequency domain response are important to take into consideration to ensure the stability and good performance of that system. These values are the phase margin PM, the gain margin GM, and the bandwidth ω_{WB}, which gives the system's half-power frequency. ω_{BW} is important for system performance and reducing output noise. PM, and GM ensure stability of the system, with $45 < PM < 60$ an ideal region for the phase margin, $GM > 6dB$ a good goal for gain margin.

Plotting the frequency response of our first system ($k_I = 0.4611, k_p = 0.1548$) allows us to further comment on the stability and performance of the system.

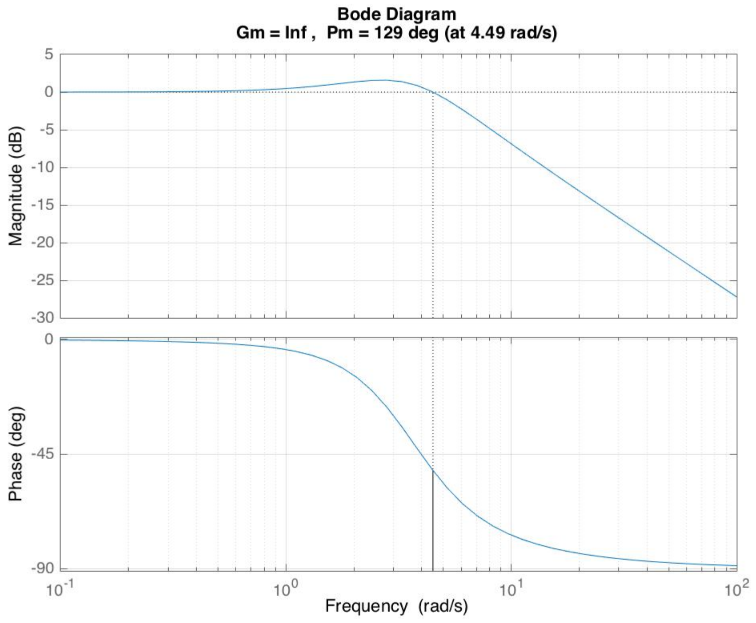

Figure 12: Frequency response for Matlab simulation 1

This system has $GM = \text{inf}$, $PM = 129°$, $\omega_{BW} = 6.55$ rad.

For our second system ($k_I = 0.4611, k_p = 0.233$)

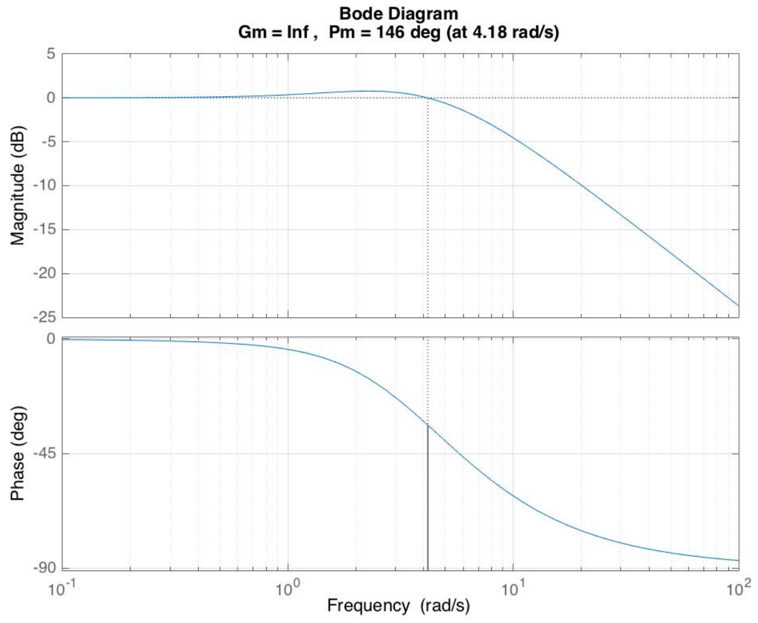

Figure 13: Frequency response for Matlab simulation 2

This system has $GM = \text{inf}$, and a slightly higher phase margin and bandwidth: $PM = 146°$, and$\omega_{BW} = 7.93$ rad.

4 Experimental Analysis

After conducting the matlab analysis in section 3 experimental data was collected from two different torsion disc systems. Data was collected from two systems in order to examine the robustness of the PI controller and make any necessary adjustments.

4.1 Setup

The first step in collecting experimental data was to select a rise time (t_r) and overshoot (M_p) for the system. As an initial starting point values of $t_r = 0.5$ sec and $M_p = 5\%$ were selected. Using these values the damping ζ and natural frequency ω_n were calculated using equations 10 and 11

$$\zeta = \frac{|\ln(0.05)|}{\sqrt{\pi^2 + [ln(0.05)]^2}} = 0.69 \tag{10}$$

$$\omega_n = \frac{1.8}{0.5} = 3.6 \tag{11}$$

These values were used as an initial starting point and adjustments were made based on the response calculated in matlab and the corresponding response on the TDS. In cases where the overshoot became too high the damping was increased. It also became necessary to increase the bandwidth of the system in order to reduce the noise on the live system. Table 2 shows the various calculated values based on necessary changes. For each iteration the value in bold was adjusted to improve the response. The adjustments were determined based on the response of the system to a step input as outlined in section 4.2 and to a ramp disturbance in section 4.3.

Test	M_p	t_r	ζ	ω_n	K_p	K_I	BW
test1	**5.00**	**0.50**	0.69	3.60	0.1548	0.4611	6.55
test2	19.58	0.090	0.69	**10**	0.469	3.558	19.55
test3	16.23	0.085	**0.80**	10	0.5472	3.558	20.948
test4	14.67	0.039	**0.90**	**20**	1.258	14.23	45.59
test5	13.63	0.038	**0.95**	20	1.329	14.23	47.00
test6	13.92	0.025	0.95	**30**	2.006	32.018	71.08

Table 2: System Response

4.2 Time Domain - Step Response

The values shown in table 2 were calculated from the closed loop transfer function of the system model. The ideal step response from matlab was then compared to experimental data from the TDS. This experimental data was collected by adjusting the K_p and K_I values in Labview and saving the response data to a text file. Data was collected for TDS machine 1 and TDS machine 4.

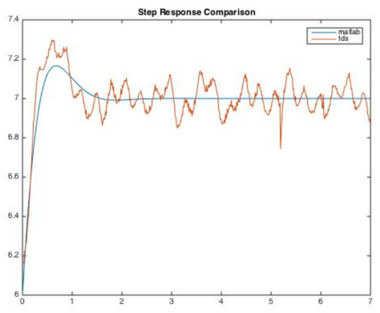

Figure 14: Step Response m1t1

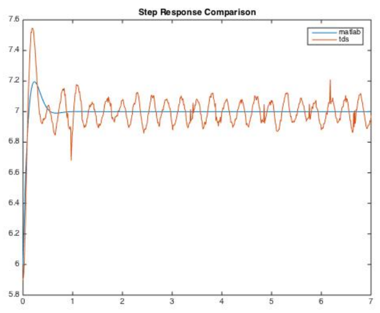

Figure 15: Step Response m1t2

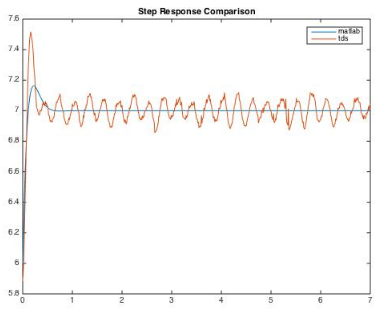

Figure 16: Step Response m1t3

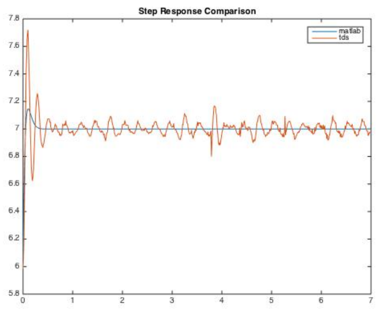

Figure 17: Step Response m1t4

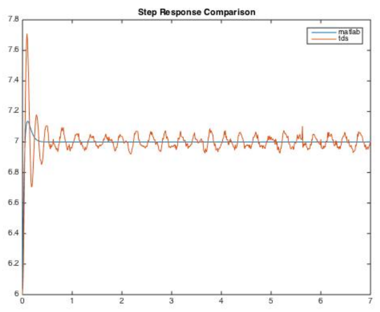

Figure 18: Step Response m1t5

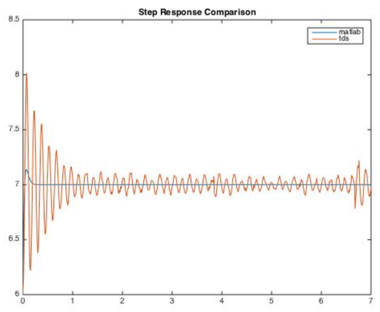

Figure 19: Step Response m1t6

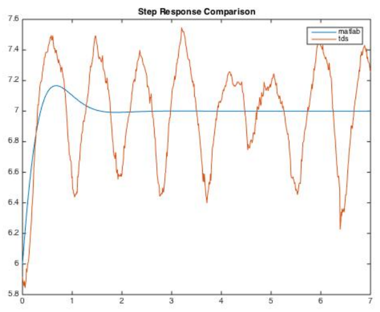

Figure 20: Step Response m4t1

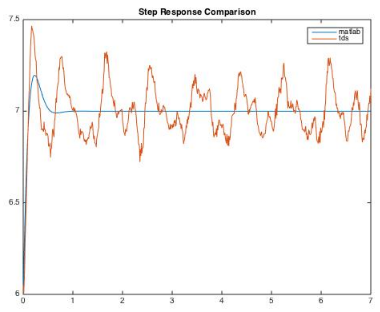

Figure 21: Step Response m4t2

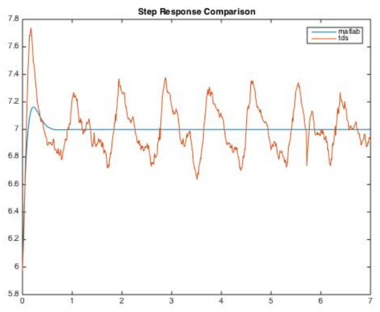

Figure 22: Step Response m4t3

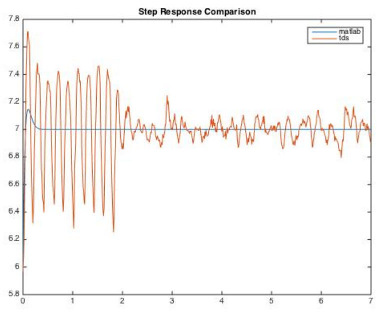

Figure 23: Step Response m4t4

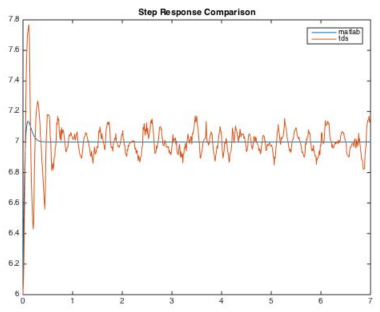

Figure 24: Step Response m4t5

Figures 14 through 19 show the plots of the matlab data and the collected experimental data for machine 1. Figures 20 to 24 show the data comparisons for machine 4. From this information it is possible to calculate the actual rise time and overshoot of the system for different K_I and K_p values. Table 3 shows the calculated t_r and M_p values.

matlab t_r	mach1 t_r	mach4 t_r	matlab M_p	mach1 M_p	mach4 M_p
0.275	0.09	0.19	16.646	29.78	54.66
0.09	0.05	0.05	19.58	54.58	46.45
0.085	0.05	0.03	16.23	51.71	73.95
0.039	0.03	0.03	14.67	71.97	71.37
0.038	0.03	0.03	13.63	70.93	77.01
0.025	0.02	n/a	3.916	101.66	n/a

Table 3: t_r and M_p Comparison

The overshoot amounts for the proportional integral controller quickly become large. Even with maximum damping it was not possible to reduce the overshoot by the desired amount. This large overshoot was partly due to increasing the natural frequency to reduce the steady state noise. With more iterations and experiments it would be possible to get a slightly better response.

4.3 Time Domain - Disturbance

Another important aspect of of designing the PI controller was viewing the response of the system to a ramp disturbance. As shown from the transfer functions and calculations in section **??** there should be a constant steady state error to a ramp disturbance. As seen in figures 25 and 26 there is a constant error to a ramp disturbance on the physical system as expected. In the plots the response with no disturbance is shown in blue and the response to the ramp disturbance is shown in orange.

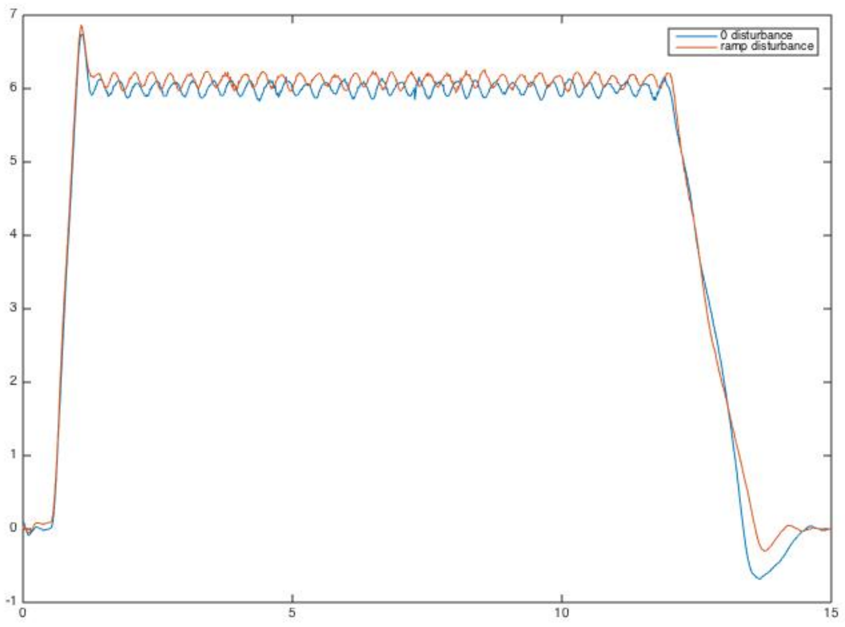

Figure 25: Ramp Disturbance m1

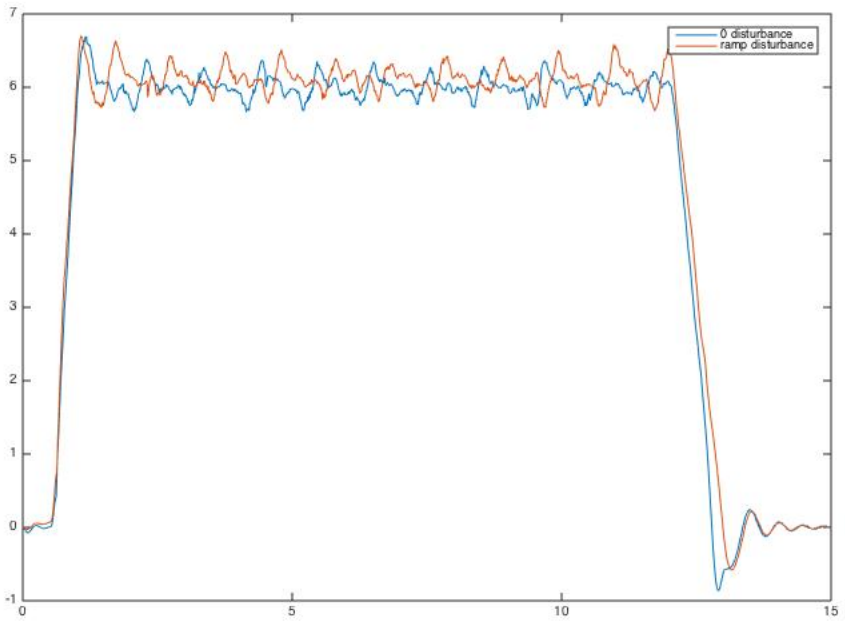

Figure 26: Ramp Disturbance m4

5 PI Controller Design Summary

Designing a PI controller is a complex and iterative process as outlined in the above sections. Analyzing a model of they system using matlab is a starting point that can provide some useful information. For example, viewing the various transfer functions helps to provide a baseline for the expected response to different inputs and disturbances. Calculated system parameters such as overshoot, rise time, natural frequency, and damping provide initial K_p and K_I values. On the actual TDS the response varies from the simulation due to variables that are not present in the ideal model. Iteratively updating system values gradually improves the system response but a PI controller is not sufficient to eliminate all undesirable characteristics. Further labs will explore more complex controllers which will continue to improve the response characteristics.