

Lab Assignment 3: DC Motor Servo System

TEAM MEMBERS:

Shiva Ghose, @gshiva

John Peterson, @jrpeters

Peter Turpel, @pturpel

Chan-Rong Lin, @pmelin

—

Teamwork Participation Pledge :: Team 1

I attest that I have made a fair and equitable contribution to this lab and submitted assignment.

My signature also indicates that I have followed the University of Michigan Honor Code, while working on this lab and assignment.

I accept my responsibility to look after all of the equipment assigned to me and my team, and that I have read and understood the X50 Lab Rules.

Name	Email	Signature
Shiva Ghose	gshiva@umich.edu	
John Peterson	jrpeters@umich.edu	
Peter Turpel	pturpel@umich.edu	
Chan-Rong Lin	pmelin@umich.edu	

1.

a.

U1

U1 serves as a difference circuit between $-Ref$ and $+Ref$ with a variable overall gain dictated by the potentiometer.

$$V_{out} = \left(\frac{R_6}{R_5 + R_6} \right) \left(\left(1 + \frac{R_2}{R_1} \right) \left(\frac{R_4}{R_3 + R_4} \right) (-Ref) - \left(\frac{R_2}{R_1} \right) (+Ref) \right)$$

$$V_{out} = \left(\frac{R_6}{R_5 + R_6} \right) \left(\left(1 + \frac{20}{50} \right) \left(\frac{20}{50 + 20} \right) (-Ref) - \left(\frac{20}{50} \right) (+Ref) \right)$$

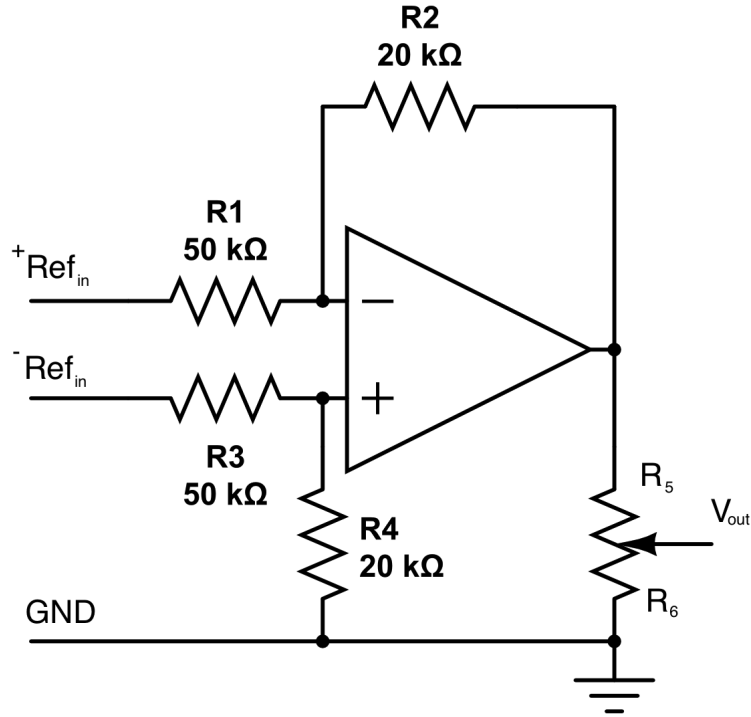


Figure 1: Unit 1 within AMC servo-amplifier

U3

U3 serves as an inverting amplifier which can be tuned by the potentiometer.

$$U3_{out} = - \left(\frac{R_3 + R_4}{R_4} \right) \left(\frac{R_2}{R_1} \right) Ref_{gain}$$

$$U3_{out} = - \left(\frac{R_3 + R_4}{R_4} \right) \left(\frac{10}{5} \right) Ref_{gain}$$

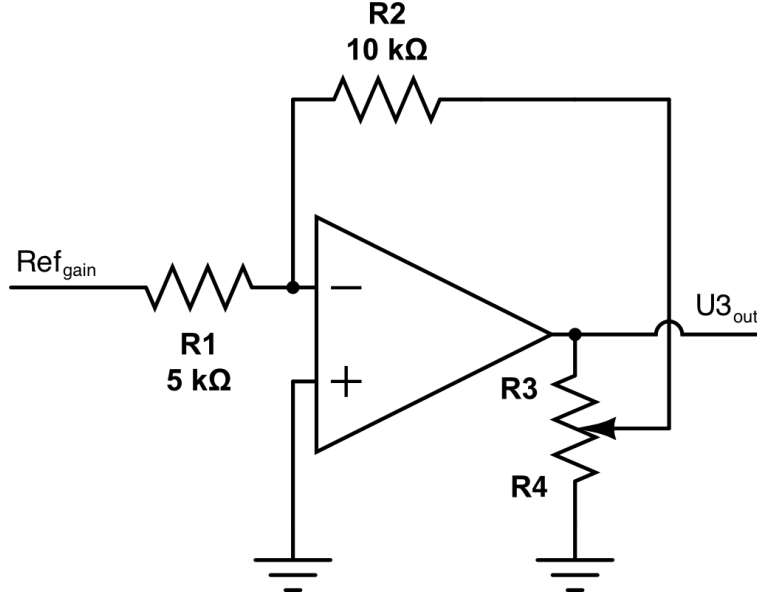


Figure 2: Unit 3 within AMC servo-amplifier

U4

U4 serves as an inverting integrator with a proportional gain when switch 3 is open, and a simple inverting amplifier with the switch closed.

$$U4_{out} = - \left(\frac{1}{R_1} \right) \left(R_2 + \frac{C_1}{s} \right) U3_{out}$$

$$U4_{out} = - \left(\frac{1}{500} \right) \left(500 + \frac{0.01}{s} \right) U3_{out}$$

b.

Current Limiting Resistor. For DC current, an inductor is effectively a short circuit. We need a resistor to limit the maximum current that goes to ground. confirm with him that its a coil in series with resistor

c.

The motor data sheet specifies a max winding temperature of 155 °C; this is the temperature at which the motor can become irreparably damaged. The data sheet also gives a thermal impedance of 11.2 °C/W. Assuming a room temperature of 15 °C, the motor's temperature can only increase by 130 °C before overheating. With these values (and the motor's internal resistance of 1.85 Ω) we can solve for the maximum continuous current:

$$130 = 11.2 * i_{continuous}^2 * 1.85$$

$$i_{continuous} = 2.5048 A$$

With this current and the given motor constant we can then calculate the maximum continuous torque:

$$T_m = k_t * i_{continuous} = 4.24 \times 10^{-2} * 2.5048 = 10.6 \times 10^{-2} Nm$$

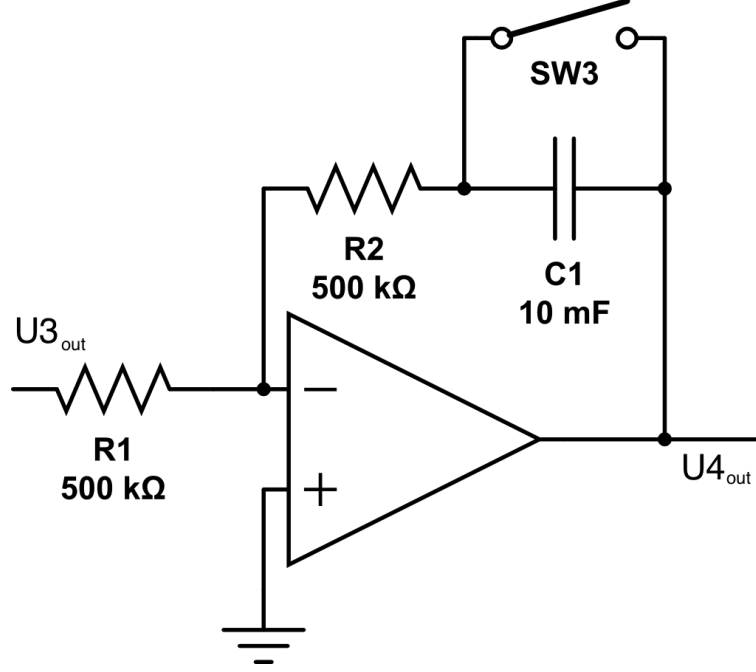


Figure 3: Unit 4 within AMC servo-amplifier

This value is more than the torque value given in the data sheet, implying that our calculated current value is less conservative than the ratings in the data sheet. Based on the given rating for max continuous torque (8.1×10^{-2} Nm), the max continuous current would be 1.91 A, but at that value the temperature rise would be 75.6°C above ambient. This means that at the max torque given by the data sheet, the motor would be well under the temperature limit. It's possible that the given value is either based on a safety factor to prevent overheating, or there are other modes of breakdown that can result from higher currents. If the lower limit is based on temperature, then we should be able to use slightly higher currents given that we are running for short periods.

The motor's data sheet lists a peak current rating of 13.0 A and a stall torque of 0.54 Nm. With this torque value and the given torque constant (4.24×10^{-2} Nm/A), the peak current should be:

$$i_{peak} = \frac{T_{stall}}{k_t} = \frac{0.54}{4.24 \times 10^{-2}} = 12.74 \text{ A}$$

This is less than the given rating for peak current. Choosing this as our peak current limit, the power in the motor is:

$$P = i_{peak}^2 * R_m = 12.74^2 * 1.85 = 300.269 \text{ W}$$

Because this is occurring at stall, there is no mechanical power output from the motor, and all of the power goes to heat. If the system were allowed to go to steady-state with no failures, this would lead to a temperature of:

$$Temp = P * R_t = 300.269 * 11.2 = 3363.0128^\circ\text{C} \text{ (above ambient)}$$

Clearly the system could not physical reach this temperature without failing, but it illustrates the reason that peak current can only be sustained for short times. With the given thermal time constant (13.8 min), we can plot the temperature rise of the motor, as shown in figure 4, and estimate that it would overheat after approximately 32.4 seconds at peak current.

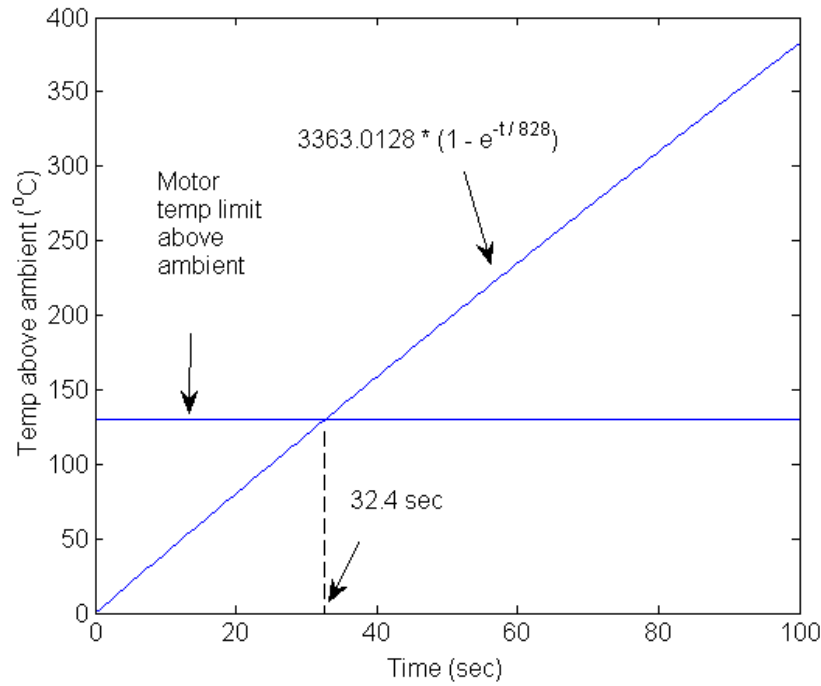


Figure 4: Motor temperature over time at peak current

d.

The 12A8E servo amplifier is capable of outputting maximum continuous and peak currents of $\pm 6\text{A}$ and $\pm 12\text{A}$, respectively (from the data sheet). However, the servo amplifier includes a potentiometer that can be used to reduce both the continuous and peak currents (while maintaining a 1:2 relationship between them). The lab instructions specify that the potentiometer should be adjusted so that the current limits will be $\pm 3\text{A}$ and $\pm 6\text{A}$. Additionally, the continuous current limit can be further reduced (without affecting the peak current) by connecting a current limiting resistor between pins P1-10 and P1-2 on the servo-amp. That was not done in this lab. Assuming the adjustment on the current limiting potentiometer was done correctly, the servo-amp should be capable of sending $\pm 3\text{A}$ continuously and $\pm 6\text{A}$ intermittently to the motor. The continuous current limit from the servo-amp is greater than the continuous current rating of the motor, so we can't rely on the servo-amp to prevent the motor from drawing too much current.

The data sheet for the servo-amp specifies a peak current of $\pm 12\text{A}$ for a maximum of 2 seconds. From information on the Advanced Motion Controls website, this can only be achieved if the current switches across its entire range (-12A to $+12\text{A}$, and vice-versa). Any change of smaller magnitude will not be maintained as long (i.e. switching from 0A to $+12\text{A}$ will only be held for 1 second before the current begins falling). A primary concern with maintaining high currents is overheating, and the servo-amp has a much lower temperature limit than the motor (65°C vs 155°C). This is why the peak current can not be maintained for long, the maximum temperature would be exceeded very quickly and the servo-amp's internal safeties would shut it off. As an example, in the thermal model for the motor at peak current (which is lower than the 6A rating of the servo-amp), 65°C was reached in only 9.8 seconds. Of course, the servo-amp has different thermal properties than the motor, so that specific value is not accurate. However, the basic idea is valid: the servo-amp is capable of passing a lot of current, and has a lower temperature limit than the motor, so there is a danger of it overheating if high currents are maintained. Luckily, the servo-amp has

built in safeties, such as a thermal shut-off.

e.

In the above sections, we showed that the motor has a continuous and peak current limits of 2.5 A and 12.74 A, and the servo-amp has current limits of 3 A and 6 A, respectively. Within our LabView controller we chose to implement saturation limits of ± 6 A so that we could take advantage of the peak current capability of the servo-amp and motor. We considered setting a saturation limit based on the motor's continuous current limit of 2.5 A, but felt that might have a negative effect on the transient response of the motor. Also, we knew that the servo-amp has built in mechanisms to prevent the current from remaining at the peak value - it automatically lowers it after at most two seconds. Additionally, as we were tuning our LabView controller, we monitored a waveform graph of the attempted output of the controller. This allowed us to adjust our gain values not just for system stability, but also to keep the attempted output within safe limits. Given more resources, we could have implemented more safety measures. For example, we could implement a slow-blow fuse rated a little under the continuous current limit of the motor. This would protect the system from high currents for prolonged times, but still allow for short bursts of higher current during transient conditions.

2.

a.

Using expressions for the back-emf of the motor with a simple model of the motor as a resistance in series with an inductance we can model the electrical characteristics of the motor. We can combine this with a simple lumped parameter model of physical dynamics of the motor to derive an overall transfer function.

$$V_s - e = iR + L \left(\frac{di}{dt} \right)$$

$$e = K_b \dot{\theta}$$

$$\Sigma \tau = \tau_{motor} - \tau_{damping} + \tau_{coulomb} = J \ddot{\theta}$$

$$\tau_{motor} = K_T i \quad \tau_{damping} = b \dot{\theta}$$

$$\tau_{coulomb} = \begin{cases} -\tau_{motor} & \text{if } \dot{\theta} = 0 \text{ and } |\tau_{motor}| < \tau_{friction} \\ -sgn(\tau_{motor}) \cdot \tau_{friction} & \text{if } \dot{\theta} = 0 \text{ and } |\tau_{motor}| > \tau_{friction} \\ -sgn(\dot{\theta}) \cdot \tau_{friction} & \text{if } \dot{\theta} \neq 0 \end{cases}$$

Where: K_t is the motor torque constant; J is the rotor inertia; b is the viscous damping on the rotor; K_b is the back-emf constant of the motor; R is the equivalent resistance of the motor; L is the effective motor inductance; i is the current flowing through the motor; and $\tau_{coulomb}$ is the force of coulomb friction on the rotor.

Neglecting coulomb friction:

$$V_s(s) - s K_b \theta(s) = I(s)(R + sL) \quad \text{and} \quad K_t I(s) = \theta(s)(Js^2 + bs)$$

$$\frac{\theta(s)}{V(s)} = \frac{K_T}{s[(Js + b)(Ls + R) + K_b K_T]}$$

The transfer function relating voltage and position is not particularly convenient but we can obtain a transfer function easily in terms of $I(s)$.

$$K_t I(s) = \theta(s)(Js^2 + bs)$$

$$\frac{\theta(s)}{I(s)} = \frac{K_T}{(Js + b)s}$$

This model relating angular position, $\theta(t)$ with motor current $i(t)$ requires the moment of inertia of the rotor, J , the viscous damping coefficient, b , and the torque constant K_T . Note that coulomb friction has been neglected.

The driver gives us current control as long as we are operating away from the maximum speed of the motor. At maximum speed, the current required by the motor should drop to some steady state value required to overcome losses within the motor.

may need to redo picture a bit to agree with terminology in equation

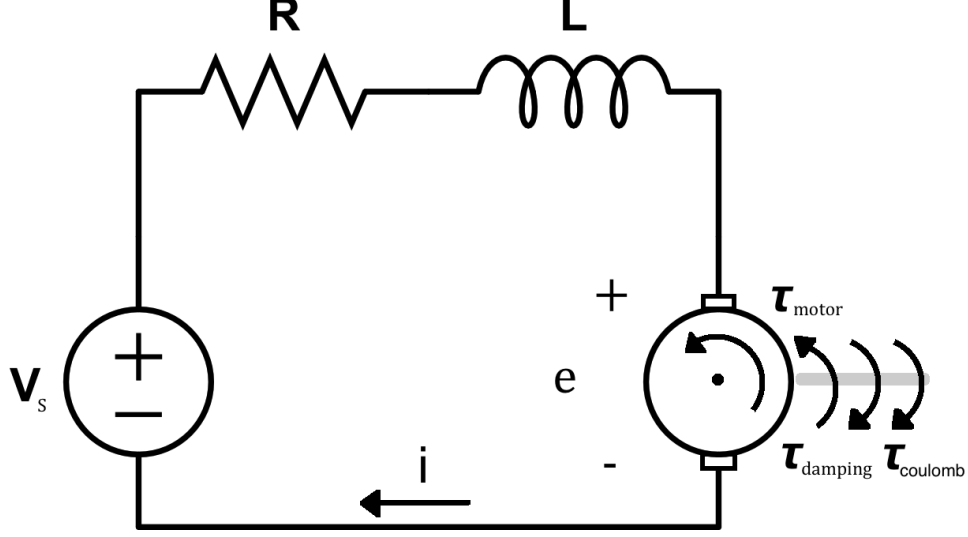


Figure 5: DC motor model

b.

The torque of friction and the viscous damping can be jointly derived in an experiment where for a given constant input voltage to the driver, V_{driver} , we read off the angular velocity. At a constant velocity the following equation holds giving a relationship between τ_{motor} , b , and $\tau_{coulomb}$.

$$\begin{aligned}\Sigma\tau &= \tau_{motor} - b\dot{\theta} + \tau_{coulomb} = J\ddot{\theta} & \ddot{\theta} &= 0 \\ \tau_{motor} &= K_T I_m = b\dot{\theta}_{ss} - \tau_{coulomb}\end{aligned}$$

Assuming that we have the motor torque constant K_t , we can write an equation that relates steady state angular velocity $\theta_{steadystate}$ to motor current which is known to be directly proportional to V_{driver} for suitably small values of V_{driver} .

$$V_{driver} = I_m \quad V_{driver} = \left(\frac{b}{K_T} \right) \dot{\theta}_{ss} - \frac{\tau_{coulomb}}{K_T}$$

From our plot in figure 6 we can obtain (b/K_t) as the slope of each of the linear segments and $\tau_{coulomb}/K_T$ as the y intercepts shown in table 1. We can separately determine the force of static friction $\tau_{staticcoulomb}$ by measuring the smallest input voltage to the driver, and from this how much torque, is required to start the motor from rest.

$$\begin{aligned}\Sigma\tau &= \tau_{motor} - b\dot{\theta} + \tau_{coulomb} = J\ddot{\theta} & \dot{\theta} = \ddot{\theta} &= 0 \\ V_{driver} &= -\frac{\tau_{coulomb}}{K_T}\end{aligned}$$

	Slope ($\frac{V*s}{rad}$)	y-intercept (V)
Forwards	5.4289e-04	0.46374
Reverse	6.72805e-04	-0.60717

Table 1: Extracted slopes and y-intercepts from current speed curve

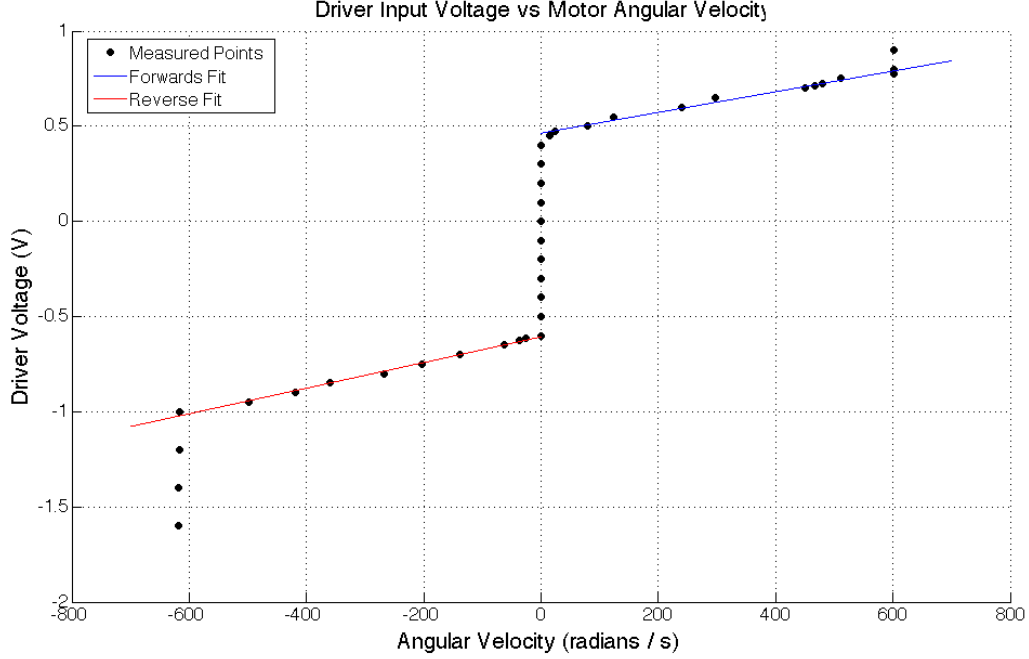


Figure 6: Relationship between angular velocity and steady state voltage applied to motor driver

	Data Sheet Values	Average Estimated Values
$b \text{ (N} \cdot \text{s)}$	$3.7 \cdot 10^{-6}$	$2.5773 \cdot 10^{-5}$
$\tau_{coulomb} \text{ (N} \cdot \text{m)}$	$5.6 \cdot 10^{-3}$	0.044978

Table 2: Comparison of viscous damping and friction torque estimates with values obtained from the data sheet

If we assume the motor constant on the data-sheet of $0.0424 \text{ N} \cdot \text{m/A}$ then we can arrive at the following estimates for b and $\tau_{coulomb}$ which are shown in 2 compared against the values obtained from the data sheet. Our values are within an order of magnitude of the provided values.

To compute the torque constant, K_T we can first compute the back-emf constant, K_b , which we know to be numerically equal to K_T in SI units. To calculate K_b we can measure the internal resistance of the motor, R , using an ohm-meter. Then running the system at a constant current, we can measure the source voltage, V_s that the driver outputs to maintain that current to obtain e . We can use the encoder to measure the angular velocity and use that value with e to solve for K_b :

$$\Sigma V = V_s + V_R + V_L e = V_s + IR + e = 0$$

$$K_b = \frac{e}{\dot{\theta}}$$

We can experimentally determine the viscous damping term, b by applying a known torque to the motor. Once the angular velocity has reached steady state, b can be read off directly. The driver in current mode delivers a known current, which using the torque constant determined above yields the torque.

$$\Sigma \tau_{\text{tau}} = \tau_{\text{motor}} - b\dot{\theta} + \tau_{\text{coulomb}} = J\ddot{\theta} \quad \ddot{\theta} = 0$$

$$b = \frac{\tau_{motor} + \tau_{coulomb}}{\dot{\theta}} \approx \frac{\tau_{motor}}{\dot{\theta}}$$

We can determine the moment of inertia of the rotor, J by starting from rest and sending the motor a known torque while recording the position as a function of time stopping when the system has reached a constant velocity. Then by computing the 1st and 2nd order derivatives at each time step we can assemble a linear system to allow us to jointly solve for J , b , and $\tau_{coulomb}$.

$$A_t = \begin{bmatrix} \ddot{\theta}_t & \dot{\theta}_t & -1 \end{bmatrix} \quad x = \begin{bmatrix} J \\ b \\ \tau_{coulomb} \end{bmatrix} \quad z_t = \tau_{motor}$$

$$Ax = z$$

Rather than attempt the method above, we instead used MATLAB's numerical differential equation solving to simulate the trajectory of the system as a function of time, driver current, I_t and our unknown system parameters J , b , K_t , and $\tau_{coulomb}$. By then comparing the predicted angular positions with the measured angular positions over time, we can construct a non-linear regression to estimate the unknown parameters. We collected a data set where the driver was given a sinusoidal voltage, and we assume that motor current, I_m , is identical to this driver voltage and recorded the motor motion that this input caused.

Table 3 below shows the parameters for our equation of motion obtained from the data sheet. The results, in figure 7 shows the estimated trajectory using these parameter values. It is very clear that these parameter values are quite far off. Attempting to perform nonlinear optimization on our model parameters proved to be very difficult, so we resorted to parameter sweeping to identify appropriate values. Figure 8 shows one of the best results discovered by this parameter sweep. In this case the error has been reduced, but the fit still appear inappropriate.

	Data Sheet Parameters
$J \left(\frac{N \cdot s^2}{m} \right)$	$8.5 \cdot 10^{-6}$
$b \left(N \cdot s \right)$	$3.7 \cdot 10^{-6}$
$\tau_{coulomb} \left(N \cdot m \right)$	$5.6 \cdot 10^{-3}$
$K_T \left(\frac{N \cdot m}{A} \right)$	$4.24 \cdot 10^{-2}$

Table 3: Original model parameter values obtained from the data sheet.

Manual parameter guessing revealed that it was not possible to obtain the measured curve using the model, repeated below.

$$\Sigma_{tau} = K_T I_m - b\dot{\theta} + \tau_{coulomb} = J\ddot{\theta}$$

Clearly there were some effects that this model neglected. Our previous experiment to attempt to determine b and $\tau_{coulomb}$ showed that these parameters are likely not the same in each direction, so we augmented the model to allow b and $\tau_{coulomb}$ to take on different values in the forwards direction and reverse directions yielding the following model:

$$\Sigma_{tau} = \tau_{motor} - b\dot{\theta} + \tau_{coulomb} = J\ddot{\theta}$$

$$\tau_{coulomb} = \begin{cases} -\tau_{forwards} & \text{if } \dot{\theta} > 0 \\ +\tau_{reverse} & \text{if } \dot{\theta} < 0 \\ -\tau_{forwards} & \text{if } \dot{\theta} = 0 \text{ and } \tau_{motor} > \tau_{forwards} \\ -\tau_{motor} & \text{if } \dot{\theta} = 0 \text{ and } 0 \leq \tau_{motor} \leq \tau_{forwards} \\ +\tau_{reverse} & \text{if } \dot{\theta} = 0 \text{ and } \tau_{motor} < -\tau_{reverse} \\ +\tau_{motor} & \text{if } \dot{\theta} = 0 \text{ and } 0 \geq \tau_{motor} \geq -\tau_{reverse} \end{cases}$$

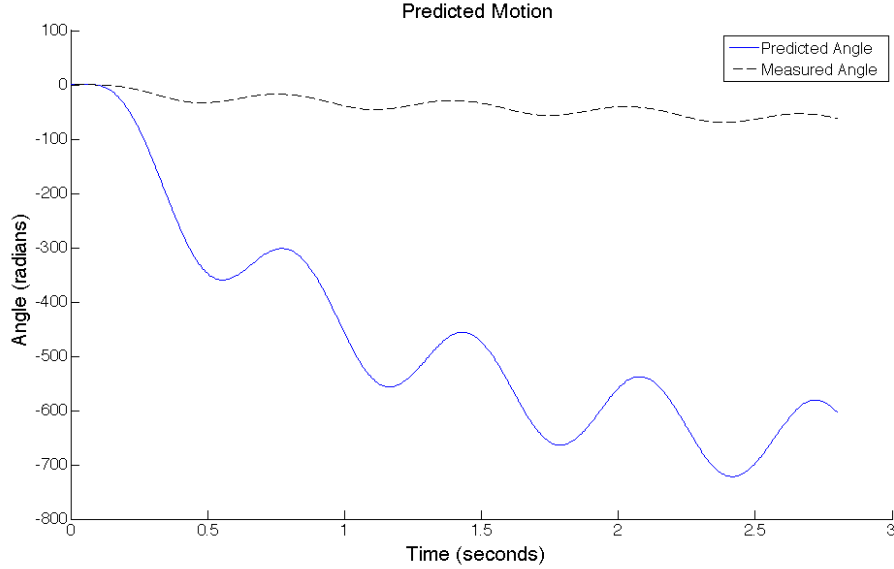


Figure 7: Predicted behavior vs Actual behavior using initial parameter values.

$$b = \begin{cases} b_{forwards} & \text{if } \dot{\theta} > 0 \\ b_{reverse} & \text{if } \dot{\theta} < 0 \end{cases}$$

With this improved model and some intelligent guessing for new parameter values, we arrived at an excellent fit shown in figure 9. The most notable feature of these values is the large difference between the torque of friction in the forwards and reverse directions.

	Data-Sheet Values	Fit Values
$J \left(\frac{N \cdot s^2}{m} \right)$	$8.5 \cdot 10^{-6}$	$3.50514 \cdot 10^{-5}$
$K_t \left(\frac{N \cdot m}{A} \right)$	0.0424	0.0314499
$b_{forwards} (N \cdot s)$	$3.7 \cdot 10^{-6}$	$1.08586 \cdot 10^{-5}$
$b_{reverse} (N \cdot s)$	$3.7 \cdot 10^{-6}$	$2.49301 \cdot 10^{-5}$
$\tau_{forwards} (N \cdot m)$	$5.6 \cdot 10^{-3}$	0.0224389
$\tau_{reverse} (N \cdot m)$	$5.6 \cdot 10^{-3}$	0.0143096

Table 4: Final Parameter Values vs Original Data Sheet Values

Table 3 below shows our final fit values alongside the original values obtained from the data sheet, while the fit curve itself is shown in figure 9. This large difference in values is clearly illustrated by the large difference in behavior between the actual system and our model using the values obtained from the data sheet shown in figure 7. We can also compare these fit values with values obtained from our earlier experiment, after adjusting for the difference in motor torque constant, and notice that our terms are quite similar, however it seems as if this experiment did not accurately estimate the force of friction as the relative values seem to be flipped between the two experiments.

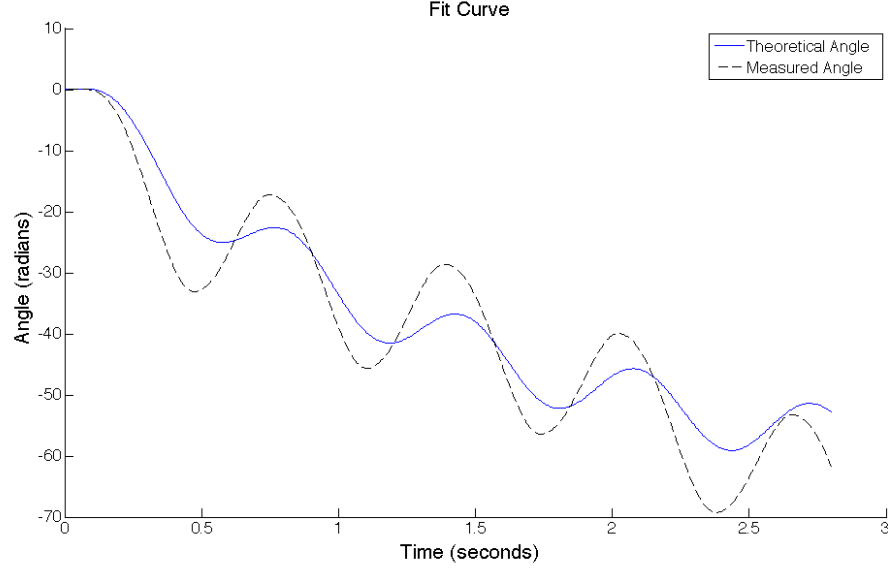


Figure 8: Fit by non-linear optimization and parameter sweeping: $J = 6.06724 \cdot 10^{-5}$ $B = 4.05562 \cdot 10^{-5}$ $F_f = 1.45312 \cdot 10^{-4}$ $K_t = 0.019020$

	Predicted Value	Fit Value
$B_{forwards}$	$1.707338 \cdot 10^{-5}$	$1.08586 \cdot 10^{-5}$
$B_{reverse}$	$2.11611 \cdot 10^{-5}$	$2.49301 \cdot 10^{-5}$
$\tau_{forwards}$	0.0145846	0.0224389
$\tau_{reverse}$	0.0190954	0.0143096

Table 5: Damping and friction torques calculated from earlier experiment vs fit values

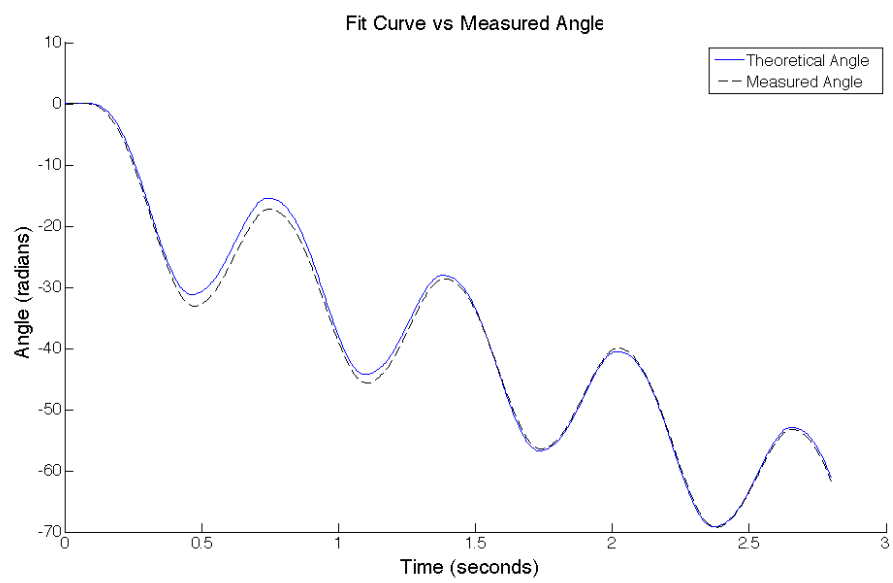


Figure 9: Predicted behavior vs Actual behavior using fit parameter values and enhanced modeling

3.

Simulink Model

Figure 10 shows the overall model of DC motor servo system for both velocity and position control. It consists of four main units:

- DC Motor
- Optical Encoder
- Servo Amplifier, and
- Controller

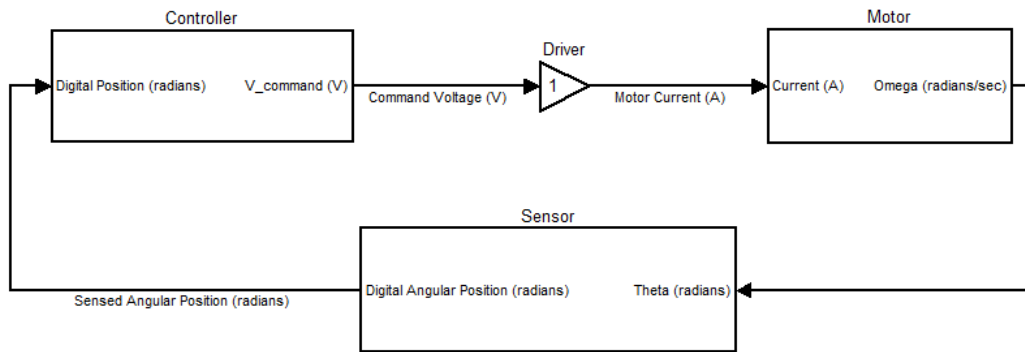


Figure 10: Predicted behavior vs Actual behavior using fit parameter values and enhanced modeling

DC Motor

The DC motor subsystem, seen in figure 11 worked on the principle that the input torque generated was based on the conversion equation:

$$T = K_t * i$$

Where, $K_t = 4.24 \times 10^{-2}$ is the torque constant of the motor. According to the motor datasheet, the rotor inertia, J , is 8.5×10^{-6} , and the viscous damping coefficient, B , is 3.7×10^{-6} .

Friction torque was modeled as a simple coulomb viscous model (see figure 12). As the viscous friction was already taken into account in the motor transfer function, the model of coulomb friction was created using the following simple function:

$$T_f^{+/-}(\omega) = T_{coul}^{+/-}$$

Where, Coulomb Friction Torque: $T_f = 5.6 \times 10^{-3}$ [Nm/A]

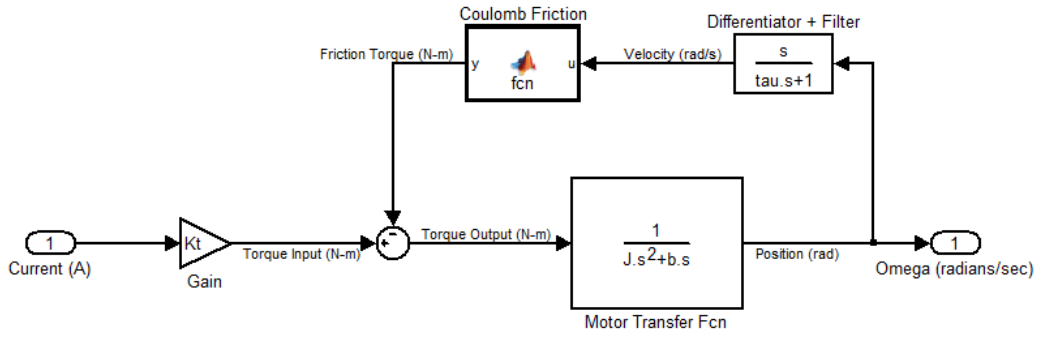


Figure 11: DC Motor Subsystem

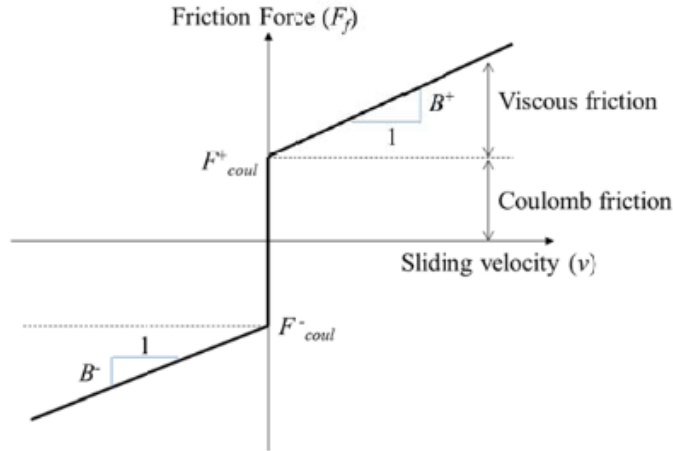


Figure 12: Plot of Friction Force v.s. Velocity

Optical Encoder

The optical encoder (see figure 12) was modeled as a quantizer with the interval size equal to $2\pi/2000$ was used to simulate the effects of the ADC.

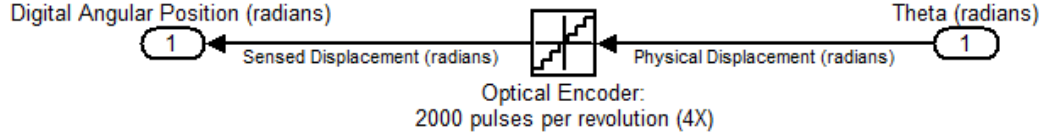


Figure 13: The Optical Encoder Subsystem

Servo Amplifier

The Servo Amplifier was treated as a gain equal to 1 due to its high bandwidth (2.5 kHz). It can be seen in figure 10.

Controller

Two controllers were used to achieve both position and velocity control, they are visible in figures 14 and 15. The saturation limit was set to $\pm 6A$ based on the maximum peak current of the servo-amplifier. An additional concern was that the *RMS* command voltage needed to be lower than the maximum continuous current of the motor (1.92A). We ensured that using limiting our output to ± 6 Volts generally allowed for better tracking stability while not often applying more than 1.92A for continuous periods of time.

Additionally PID block represents the controller while τ (tau) represents the filter time-constant for the transfer function from angular position to velocity. It was treated as a variable coefficient.

J	B	K_t	T_f	K_{amp}
Rotor inertia [kgm^2]	Viscous damping [Nms]	Torque constant [Nm/A]	Friction torque [Nm]	Gain of driver [A/V]
8.5×10^{-6}	3.7×10^{-6}	4.24×10^{-2}	5.6×10^{-3}	1

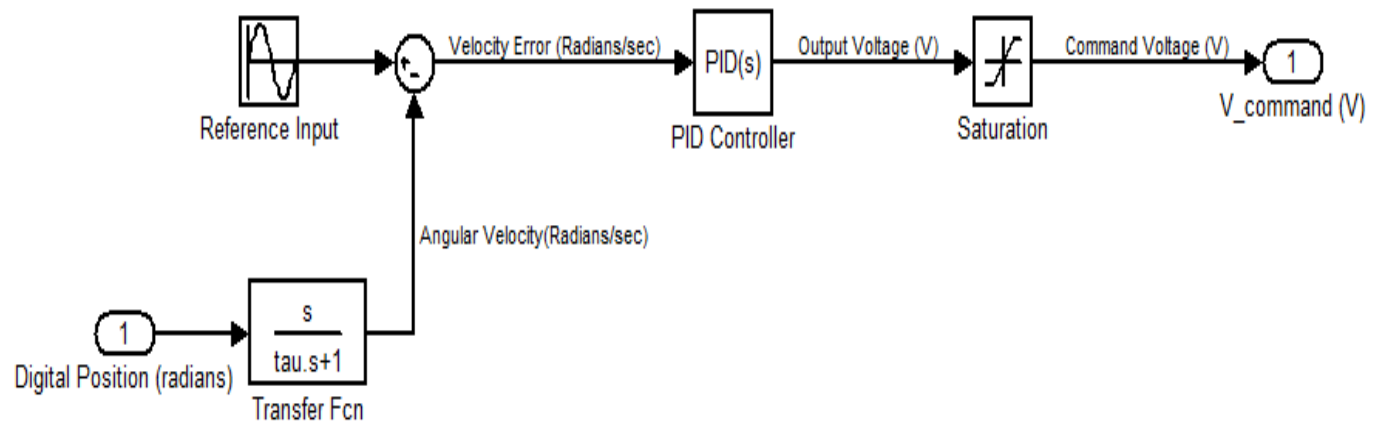


Figure 14: The Velocity Control Subsystem

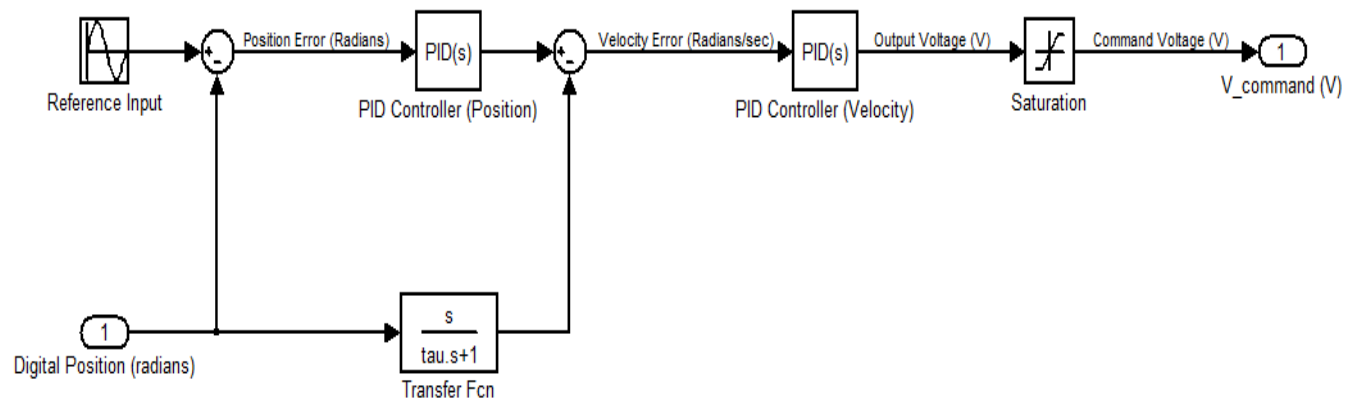


Figure 15: The Position Control Subsystem

Controller Design

The cascaded P/PI controller (see figure 16) is one of the most commonly used controller structure in motion control systems. It is simple and has a very systematic tuning procedure. Thus, we used it as the fundamental structure for both position and velocity control.

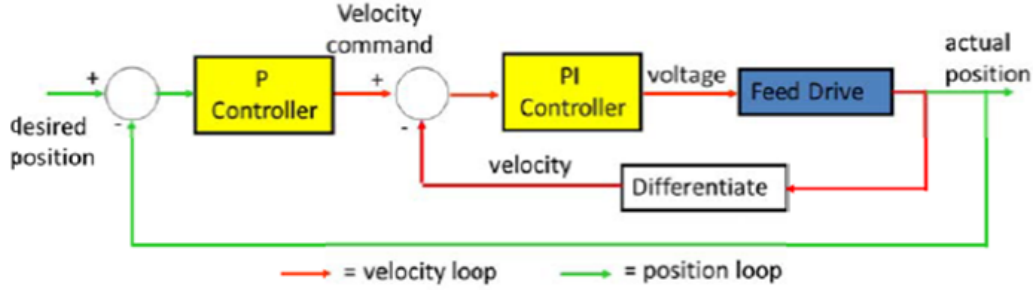


Figure 16: Basic Structure of P/PI Controller

It consists of two cascaded loops. The inner one is the velocity loop. It is a PI controller which aims to ensure that the velocity command is tracked accurately. It is surrounded by the outer position loop which is a P controller that ensures the desired reference point is achieved. Having both loops makes it easy to deal with each loop at one time. The first step is to optimize the PI controller, achieving the best performance in the velocity loop, and then tune the P controller to ensure the ability of position tracking.

Velocity Control System - Dynamics Analysis

Before we started to work on tuning procedures, the overall system dynamics was required to be known since it provided essential information for the guidelines of controller tuning. Frequency response was used to analyze the velocity loop based on different values of time constant: τ (denoted by τ). The following figures (17, 18 and 19) show Bode diagrams for the transfer functions of open-loop (G_{OL}), close-loop (G_{CL}), input disturbance rejection (G_{di}), and output disturbance rejection (G_{do}) with $\tau = 0.05, 0.1$, and 1 , respectively.

- G_{OL} (see figure 17): when τ increases, the gain of G_{OL} becomes lower in medium and high frequencies, so does the phase behavior. It can be observed that a larger τ may make the system unstable in lower frequencies since the phase is close to -180 degrees (smaller Phase Margin) and there are still other delays neglected by assumptions, such as the dynamics of servo amplifier.
- G_{CL} (see figure 18): the bandwidth of G_{CL} decreases as τ increases.
- G_{di} (see figure 18): the bandwidth of G_{di} is the same as that of G_{CL} because the system is uncontrolled.
- G_{do} (see figure 19): its quality in higher frequencies becomes better when τ increases.

Thus we find that as τ goes higher, the quality of output disturbance rejection becomes better but this comes at the expense of reduced bandwidth, which is undesired. In terms of our goals, high bandwidth (up to 5Hz), and good quality of disturbance rejection (60Hz noise signal) need to be achieved at the same time; therefore, the value of τ has to be carefully determined with controllers.

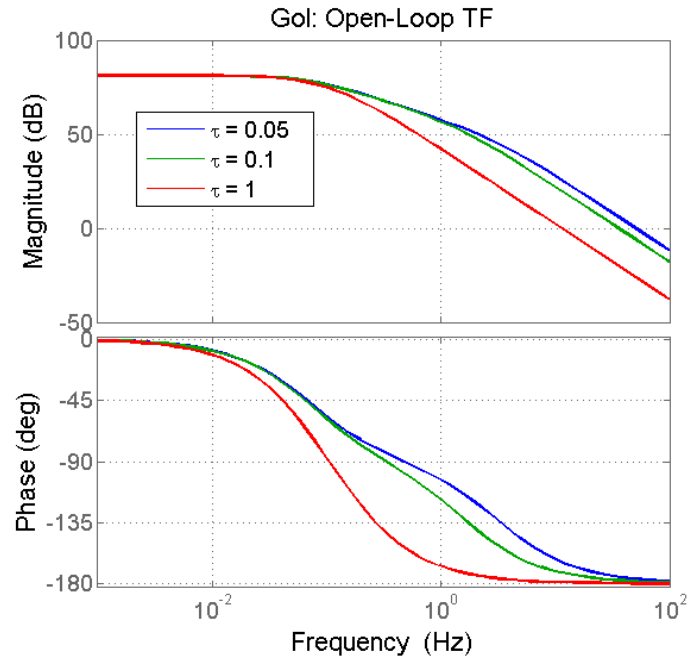


Figure 17: Changes in G_{OL} as τ is increased from 0.05 to 1

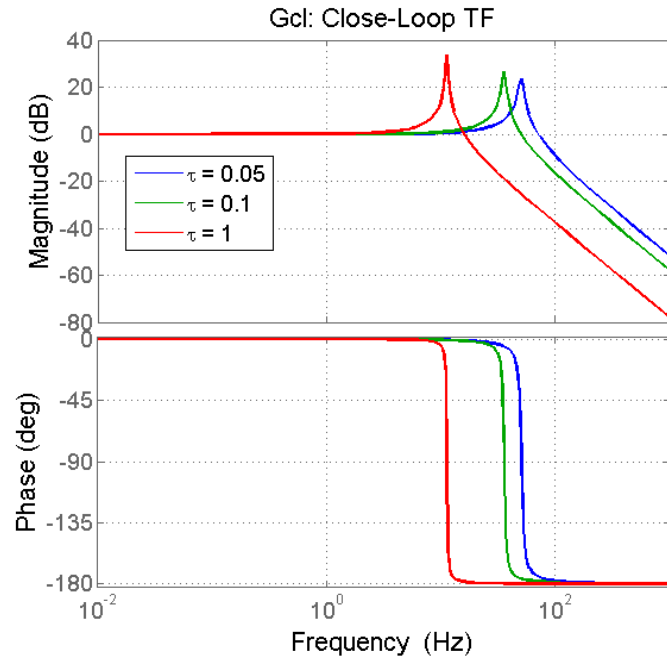


Figure 18: Changes in G_{CL} and G_{di} as τ is increased from 0.05 to 1

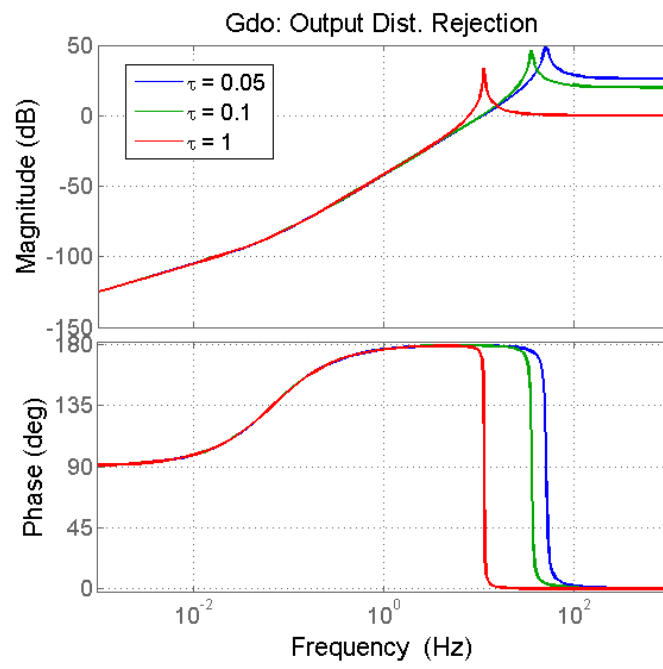


Figure 19: Changes in G_{do} as τ is increased from 0.05 to 1

Velocity Control System Controller Tuning

K_{VP} : Proportional Gain

K_{VI} : Integral Gain

K_{VD} : Derivative Gain

d) Keep steady state error below 2% for a step command of 2π radians/sec

- Initialize τ as 0.05 to begin with.
- Tune K_{VP} as high as possible because when K_{VP} is increased, the crossover frequency as well as the bandwidth of velocity loop will also increase.
- If K_{VP} exceeds a certain value, it may affect the system stability or saturate either the actuator or the current driver. Considering this issue, K_{VP} was limited by the maximum peak current (6A) and continuous current (1.92A), respectively. Thus, K_{VP} was tuned to 15, ensuring the commanded voltage less than 1.92 A in steady state.
- K_{VI} was not required here since the steady-state error was already less than 2%. The reason may be because the filter seemed to provide the similar function to the integral gain.

e) Provide command tracking: Amplitude should remain within 5% of command up until 5Hz for command amplitude of $\pi/2$ radians/sec

- The tracking error of 5Hz command was up to 10%, so that we tried to change τ to achieve better performance.
- When τ was 0.7, the error was minimum and fluctuated below 2%. This modification also generated better track-ability for the step input.

f) Attenuate a 60Hz noise signal coming in either at the servo-amplifier or the sensor by at least 5 times.

- The noise signal with amplitude 0.1 was used to test disturbance rejection. For the input disturbance (from servo amplifier), over 90% was rejected; while none of the output disturbance (from the sensor) was rejected but even enlarged. Thus increasing τ was required to get better results.
- When τ was changed from 0.7 to 6, the goal of five-times attenuation was achieved. However, we were unable to maintain sinusoidal command tracking due to the lower bandwidth. This was a trade-off which was expected.
- Considering the limitation of the PI controller, the derivative controller was introduced to add phase lead in our control loop, broadening the bandwidth. By trial and error, finally the three objectives were achieved with the following values of the coefficients:

$$K_{VP} = 25$$

$$K_{VI} = 0$$

$$K_{VD} = 0.2 \quad (N^1 = 100), \text{ and}$$

$$\tau = 6.6$$

¹Causality filter for the differentiator

Velocity Control System Simulation Results

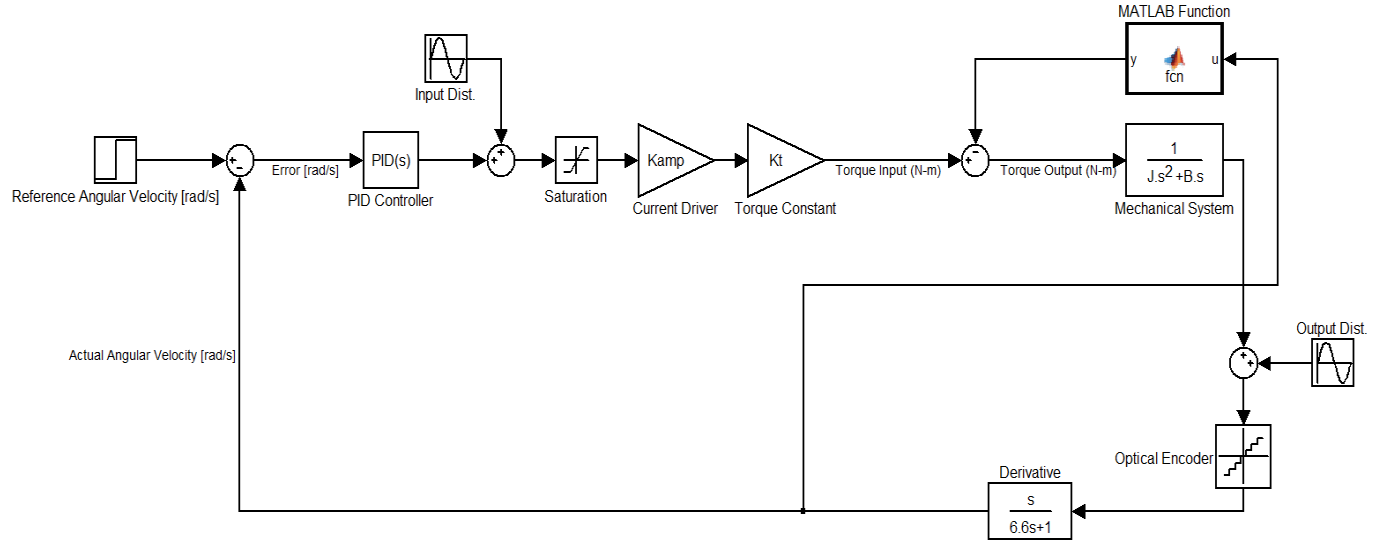


Figure 20: Block Diagram of Velocity Control system

K_{VP}	K_{VI}	K_{VD}	N	τ
25	0	0.2	100	6.6

4.

a.

When we implemented the controller and values from our simulation in part 3, we were unable to achieve stable output in the physical system. Therefore, we kept the same general approach - a velocity controller nested within a position controller with feed forward - but we re-tuned the system with different gain values. Because we have a velocity controller inside of our position controller, we first tuned the velocity control on its own (see section 6). Once it was working, we then implemented it in the inner loop of our position controller and tuned the position control gains. In the end we were not able to meet all of the control goals with one controller. We designed one controller with good step response and noise attenuation, and then modified the gain values to improve the frequency response separately. The results for the step response and

Reference	Input Amplitude [rad/s]	Output Amplitude [rad/s]	%Error	Commanded Voltage [Max. V]
(d)Step	2π	6.278	0.08%	0.15
(e)Sinusoidal	$\pi/2$	1.641	4.46%	1.99
(f)Input Disturbance	0.1	0.002	2.05%	0.18
Output Disturbance	0.1	0.02	80.0%	0.91

noise attenuation are given below.

Gains	P	I	D	Feed Forward	Filter τ
Position Loop	0.1	20	0	1	6.5797
Velocity Loop	22.5	22.5	1.125	-	6.5797

Table 6: Values of position controller with nested velocity controller for step response and noise attenuation

Figure 21 shows the system's response to a position step input of π radians, and figure 22 shows a zoomed in view after it had settled. The peak error here is 0.11%, which greatly exceeds the goal of 2% error. Additional results for position steps of different magnitudes are included in section 5b.

Compare to simulation results

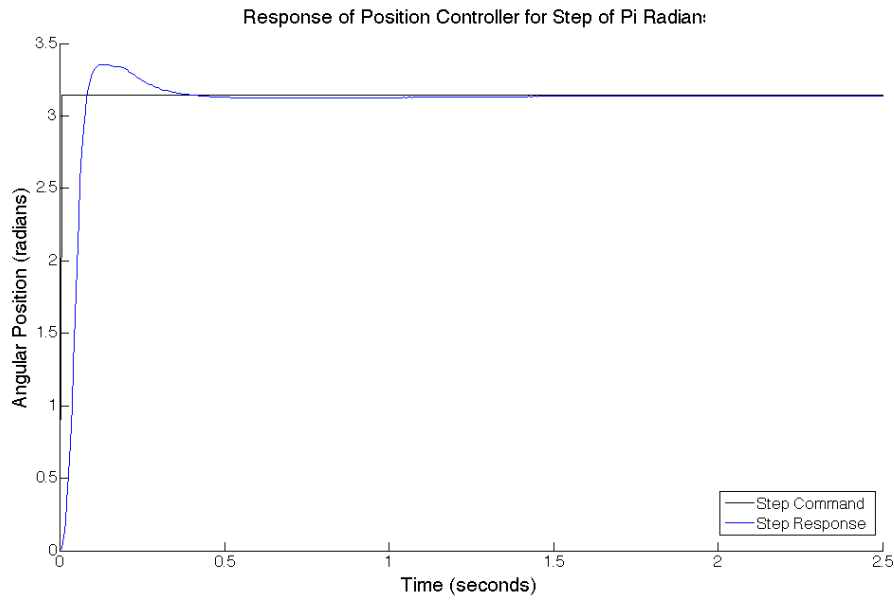


Figure 21: Step response of position controller

Figure 23 shows the noise attenuation during a step input of π radians. To simulate the noise within LabView, we added a 60 Hz sine signal with an amplitude of 1 to the output of the DAQ assistant. Figure 24 shows a zoomed in view after the step has taken place. For the step response with no noise, our steady state error was approximately 0.003, and with the noise the highest error is 0.060. Given that we were inputting a noise amplitude of one and the error only increased by 0.057, we have attenuated the noise by approximately 17.54 times. This exceeds the goal of a ten times reduction.

Compare to simulation results

The position controller we were using for the step response and noise attenuation did not perform well with sinusoidal inputs, therefore we modified the gains while keeping the overall architecture the same. The relevant values are given in table 7, and the following figures show the response at various frequencies with an amplitude of π . Figure 25 shows the error as a function of input frequency. During our tests the position output remained stable up to 5 Hz, but the output amplitude error exceeded 5% just beyond 4 Hz.

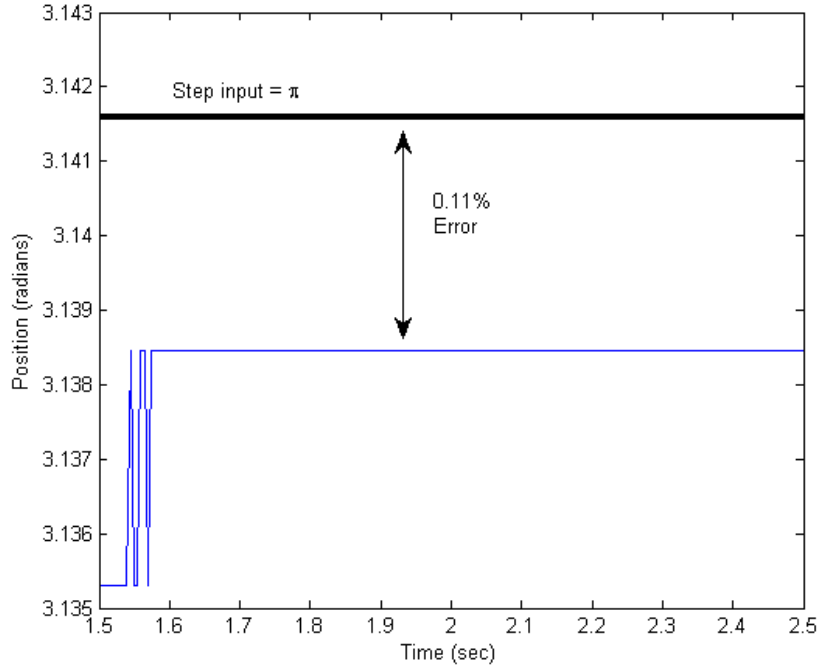


Figure 22: Steady state error for position step input of π

This does not meet the goal of less than 5% error up to 5 Hz.

Compare to simulation

Gains	P	I	D	Feed Forward	Filter τ
Position Loop	0	20	0	1	6.5797
Velocity Loop	3	7	0.4	-	6.5797

Table 7: Values of position controller with nested velocity controller for frequency response

To determine the closed-loop bandwidth of the system, we slowly increased the frequency of the input signal until the output amplitude just crossed -3 dB (for a input of π this corresponded to an output of 2.22). We found that the bandwidth of our position control system was 5.27 Hz, as shown in figure 26.

Even though we included many nonlinear factors in our simulation, we still faced a lot of trouble meeting all of the performance specifications, particularly trying to do so with just one controller. In general, we found that when we adjusted gains to meet the step requirement, the frequency response usually got worse, and vice-versa. It's possible that some of the values used in our simulation are incorrect (even after testing and regression), or that there are other factors we did not account for, such as the dynamics of the servo-amp. In the end we were able to meet the steady state error for step input and noise attenuation requirements, and were very close to the frequency response requirement.

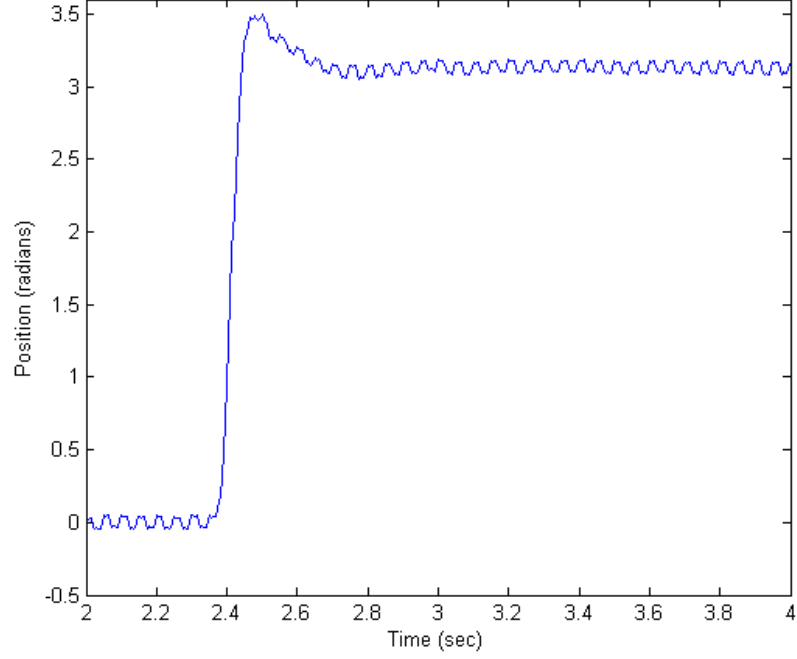


Figure 23: Step response for input of π with a 60 Hz noise signal (amplitude = 1)

Frequency (Hz)	Input Amplitude	Output Amplitude	Magnitude (dB)	%Error
1	π	3.13845	-0.008693172	-0.100033771
2	π	3.02221	-0.336504689	-3.800067888
3	π	3.079209	-0.174214105	-1.985733367
4	π	3.030152	-0.313709167	-3.547266176
5	π	2.473084	-2.078220099	-21.27929134
5.27	π	2.181877	-3.166392169	-30.54869805

Table 8: Velocity control data.

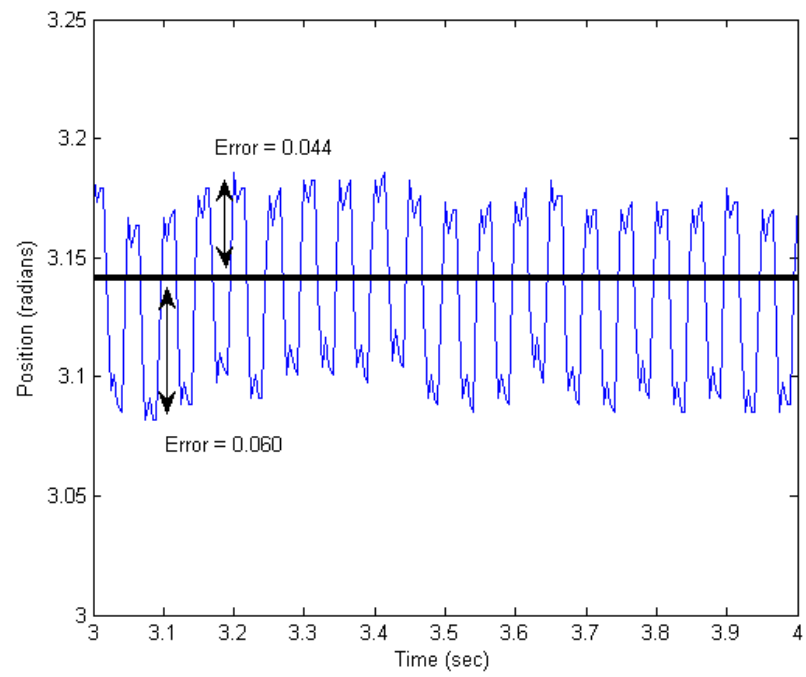


Figure 24: Steady state step response for input of π with a 60 Hz noise signal (amplitude = 1)

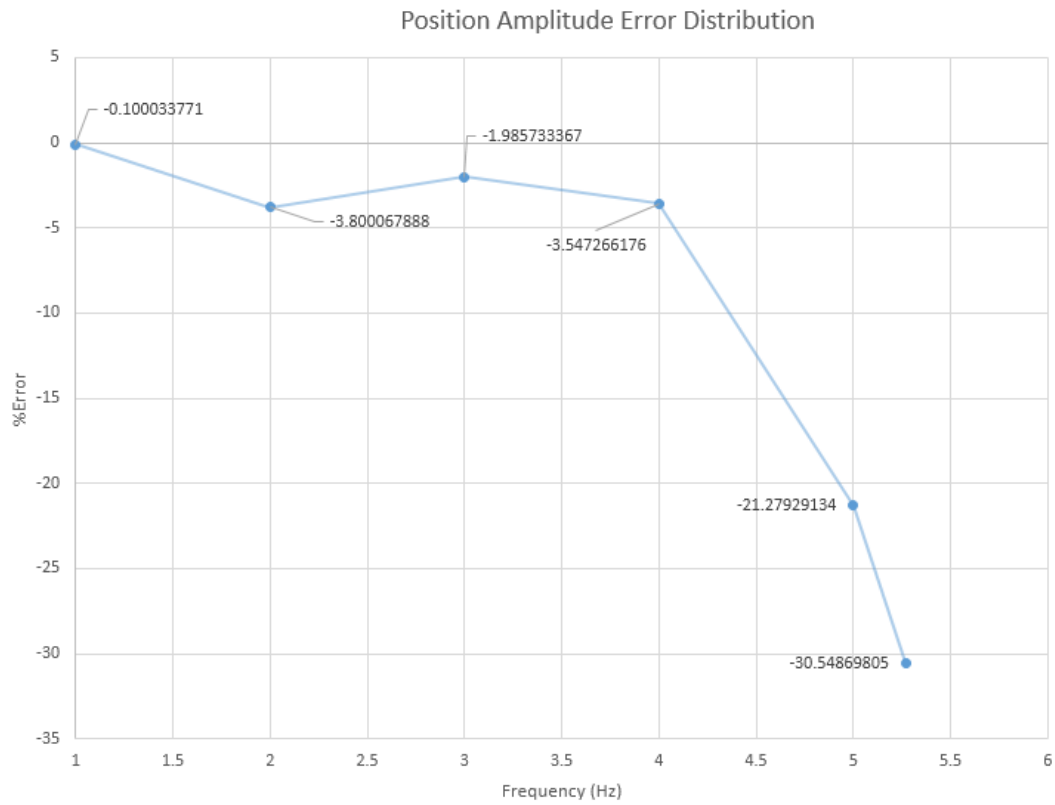


Figure 25: Position error vs. frequency for an input amplitude of π radians

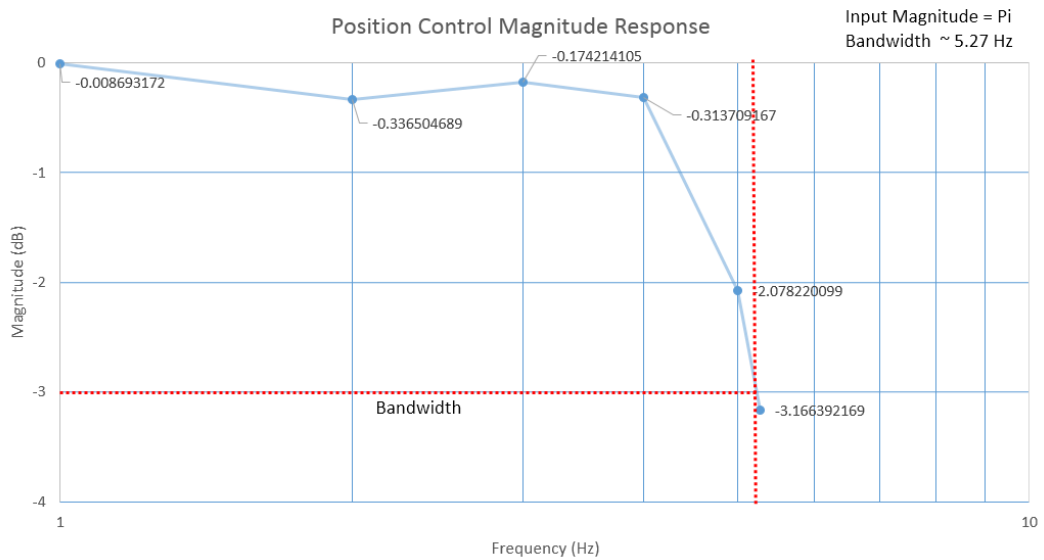


Figure 26: Output magnitude vs. frequency used to determine bandwidth of position controller

5.

a. Coulomb Friction

The bearings in the motor are the primary source of coulomb friction in the motor encoder system. This could be especially true in the current configuration of the motor because Pittman's website only specifies that they use ball or sleeve bearings in their products. If these are only radial bearings with no thrust capability, then the motor's performance could be affected by the vertical mounting we are using. Additionally, there is friction between the brushes and commutator. In section 2b we discussed and carried out experiments and modeling to determine the friction value.

Setting aside the earlier parameter identification, there are some simple experiments that could be run to estimate the friction in the system. If we assume that the torque constant given in the data-sheet is correct and that the gain of the servo-amp is exact and known, then for a given command voltage, the motor should be capable of a specific torque output. By slowly increasing the voltage from zero until the motor just begins to turn. From this, the torque caused by static friction would be equal to the product of the torque constant and the input voltage (because there is a 1:1 relationship between command voltage and applied current).

A similar method could be used to determine sliding friction. By applying a constant voltage to the system, the motor would come to a steady-state velocity. If we the coefficient of damping is known, then the torque caused by sliding friction is equal to the motor torque ($K_t * V_{command}$) minus the torque from damping (the product of the damping coefficient and velocity). This was carried out in part 2b.

Friction was included in our Simulink model from the beginning of our modeling, as presented in part 3. At the beginning of our tests we used approximations based on the data-sheet values, but as we performed more regression analysis, we obtained new values that should be closer to the true values of the physical system.

b. Saturation

Position Saturation

comment on actuation saturation

Step Input Amplitude (Radians)	1π	1.5π	2π	3π	4π
Rise Time (seconds)	0.0700002	0.0800000	0.0850000	0.110000	0.125
Settling Time (seconds)	0.290000	0.320000	0.28000	0.54500	1.45500
Percent Overshoot (%)	6.79997	16.8000	31.8500	65.0331	91.0003
Steady State Error (%)	0.100034	0.594022	0.150008	0.13335	0.399713

Table 9: Step Response for a variety of input positions. Where rise time is defined as reaching 90% of the final value while settling time is the time after which the function remains within 2% of its final value

c. Quantization

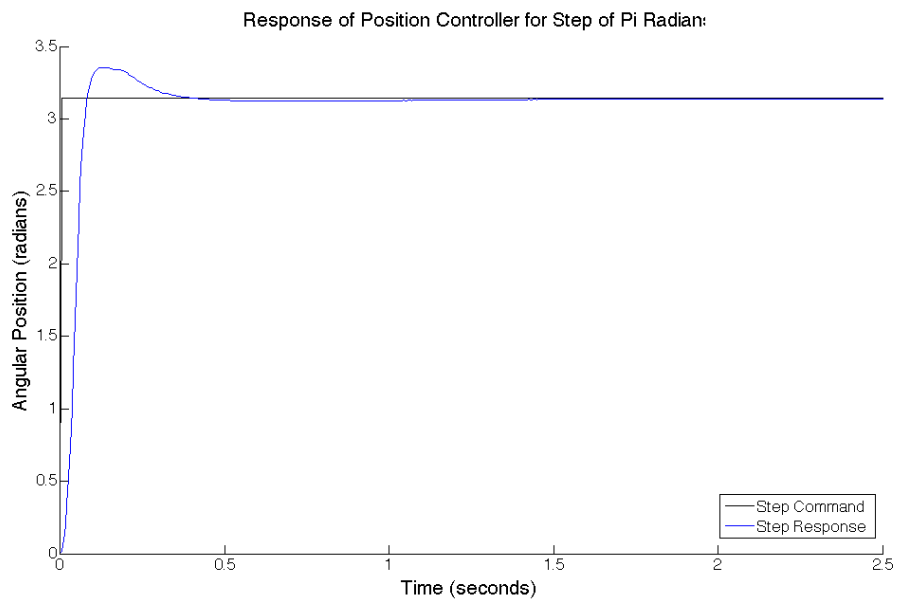


Figure 27: Step response of position controller

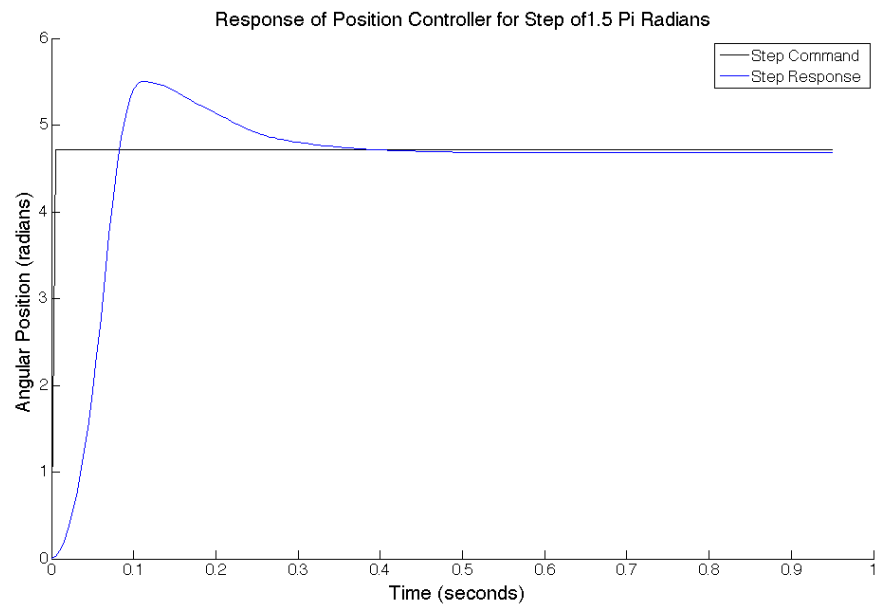


Figure 28: Step response of position controller

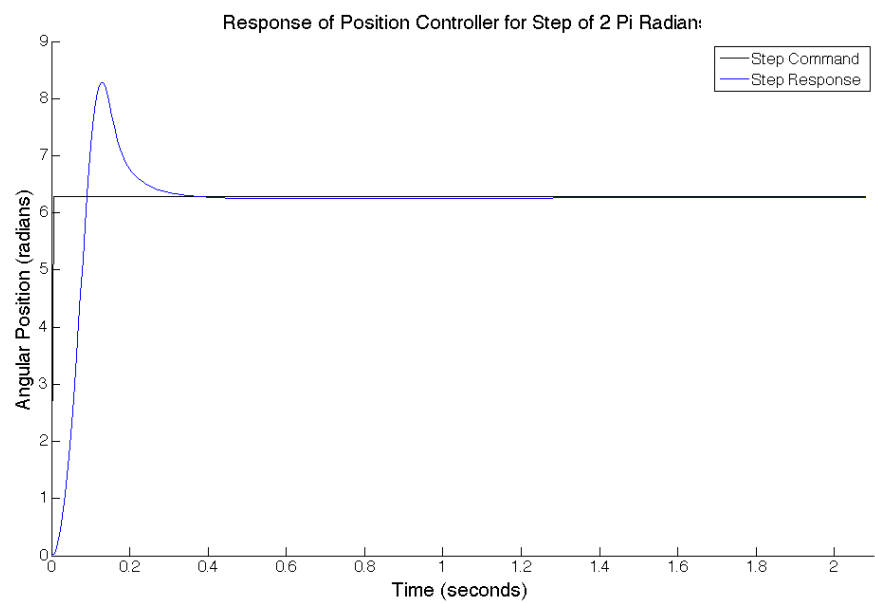


Figure 29: Step response of position controller

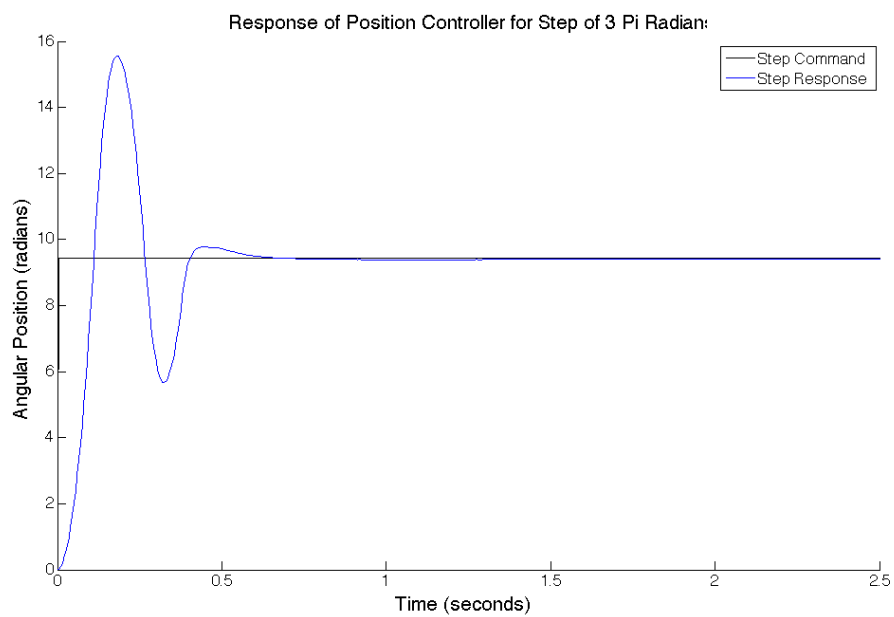


Figure 30: Step response of position controller

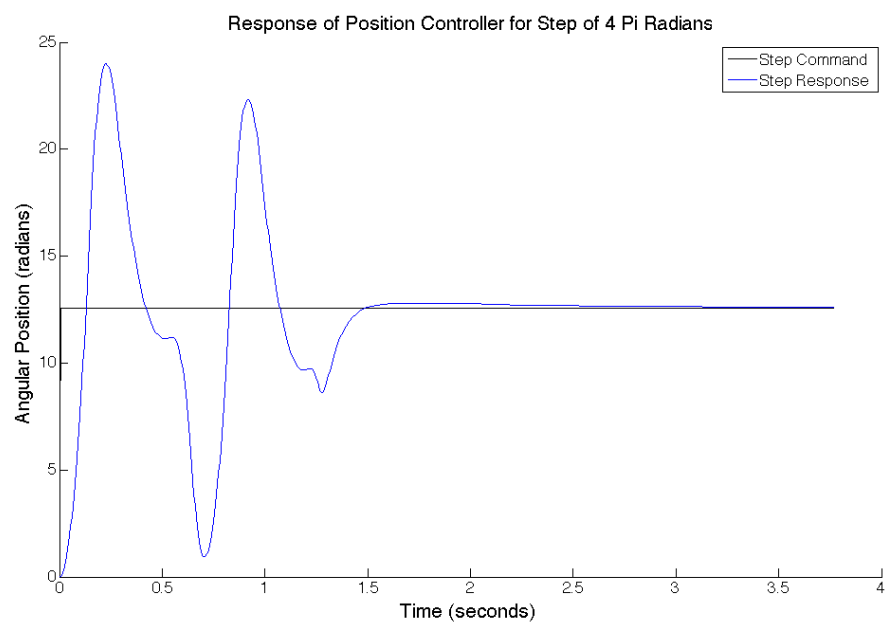


Figure 31: Step response of position controller

6.

As was discussed in section 4, work on the velocity controller for the real system was performed before the position controller, because we needed to nest the velocity loop within the position loop. The gains we used in the simulation were not able to stabilize the system with the performance we wanted, so we re-tuned the gain values. As with position control, we were not able to use one controller to meet all of the requirements: we designed one controller that could meet the step and noise requirements, and then modified the gain values to improve the frequency response separately. The results for the step response and noise attenuation are given below. The gains used here are the same as in the step position controller's nested velocity loop.

Gains	P	I	D	Filter τ
Velocity Controller	22.5	22.5	1.125	6.5797

Table 10: Values of velocity controller for step response and noise attenuation

Figure 32 shows the system's response to a velocity step input of 2π radians, and figure 33 shows a zoomed in view after it had settled. The peak error here is approximately 0.16% (and on average is much less), which greatly exceeds the goal of 2% error. Additional results for velocity steps of different magnitudes are also shown in the following tables and figures.

Compare to simulation results

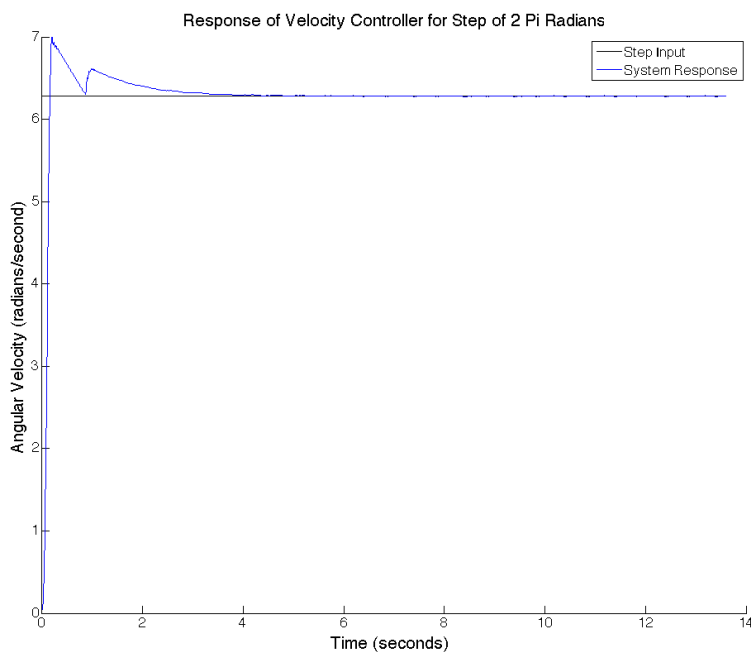


Figure 32: Velocity Controller Step Response

Figure 39 shows the noise attenuation during a step input of 2π radians/sec. To simulate the noise within LabView, we added a 60 Hz sine signal with an amplitude of 1 to the output of the DAQ assistant, just as with the position controller. Figure 40 shows a zoomed in view after the step has taken place.

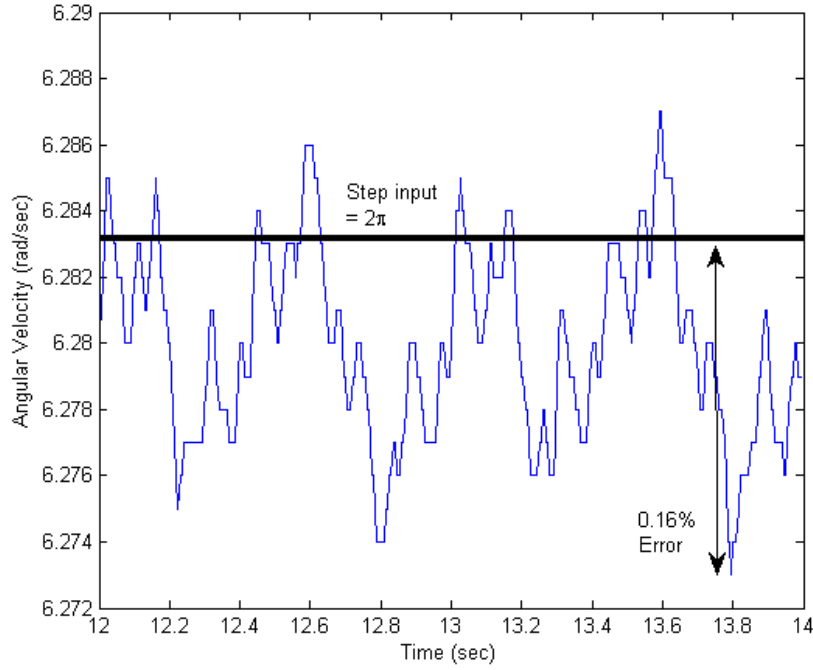


Figure 33: Steady state error for step input of $2 \cdot \pi$ rad/sec

Step Input Amplitude (Radians)	$0.5 \cdot \pi$	$1 \cdot \pi$	$1.5 \cdot \pi$	$2 \cdot \pi$	$5 \cdot \pi$	$10 \cdot \pi$
Rise Time (seconds)	0.11	0.115	0.125	0.14	0.23	0.385
Settling Time (seconds)	1.5700	1.6700	1.95500	1.9700	2.3050	2.8550
Percent Overshoot (%)	4.3415	6.1564	7.9919	11.3925	21.091	27.1553
Steady State Error (%)	0.11779	0.041862	0.053597	0.058333	0.050267	0.050634

For the step response with no noise, our peak steady state error was 0.010, and with the noise the highest error is 0.20. Given that we were inputting a noise amplitude of one and the error only increased by 0.19, we have attenuated the noise by approximately 5.26 times. This meets the goal of a five times reduction.

Compare to simulation results

Step Input Amplitude (Radians)	$2 \cdot \pi$	$2 \cdot \pi$ with Noise
Rise Time (seconds)	0.14	0.135
Settling Time (seconds)	1.9700	-
Percent Overshoot (%)	11.3925	29.6000
Steady State Error (%)	0.058328	0.054649

The velocity controller we were using for the step response and noise attenuation did not perform well with sinusoidal inputs, therefore we modified to gains to try to improve the frequency response. The relevant values are given in the table 11, and the following figures show the response at various frequencies with an amplitude of $\pi/2$. Figure 41 shows the error as a function of input frequency. During our tests the velocity output remained stable up to 5 Hz, but the output amplitude error increased greatly. The error

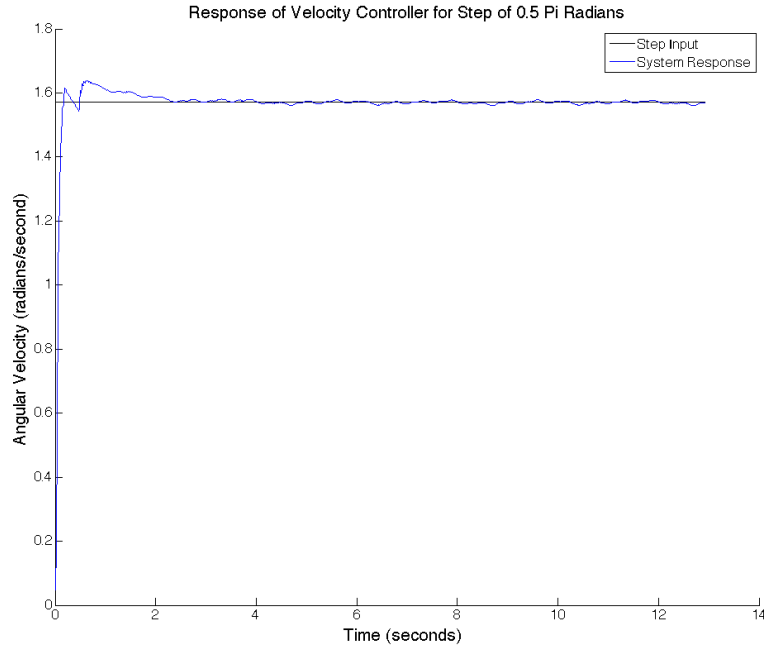


Figure 34: Velocity Controller Step Response

exceeded 5% just beyond 2.2 Hz. This does not meet the goal of less than 5% error up to 5 Hz.

Compare to simulation

Gains	P	I	D	Filter τ
Velocity Loop	2	9	0.3	6.5797

Table 11: Values of velocity controller for frequency response

Predicted Bandwidth = *SOMETHING*

Experimental Bandwidth = 2.9Hz

To determine the closed-loop bandwidth of the system, we slowly increased the frequency of the input signal until the output amplitude just crossed -3 dB (for a input of $\pi/2$ this corresponded to an output of 1.11), just as we did in section 4 to determine the bandwidth of the position controller. We found that the bandwidth of our position control system was 2.9 Hz, as shown in figure 42.

Just like with the position controller, we faced a lot of trouble meeting all of the performance specifications with just one controller. Again, we found that when we adjusted gains to meet the step requirement, the frequency response usually got worse, and vice-versa. As we mentioned in section 4 ,it's possible that some of the values used in our simulation are incorrect, or that there are other factors we did not account for. In the end we were able to meet the steady state error for step input and noise attenuation requirements, but were not very close to the frequency response requirement.

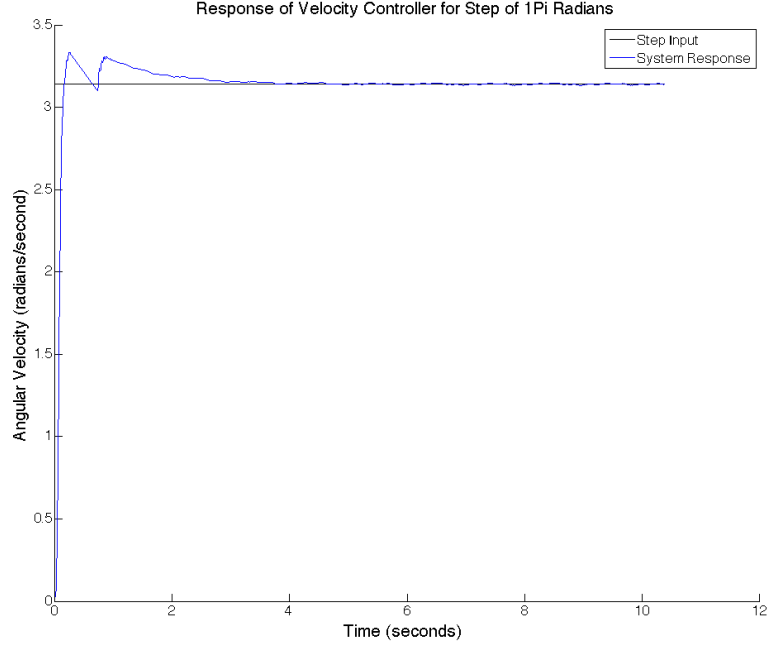


Figure 35: Velocity Controller Step Response

Frequency (Hz)	Input Amplitude	Output Amplitude	Magnitude (dB)	%Error
1	$\pi/2$	1.649	0.422015572	4.978600463
2	$\pi/2$	1.632143	0.332766594	3.905450513
2.9	$\pi/2$	1.11778	-2.955270847	-28.83991508
3	$\pi/2$	1.06516	-3.374100561	-32.18980833
4	$\pi/2$	0.581875	-8.625803574	-62.956687
5	$\pi/2$	0.47833	-10.32784514	-69.54856643

Table 12: Numerical results of velocity control.

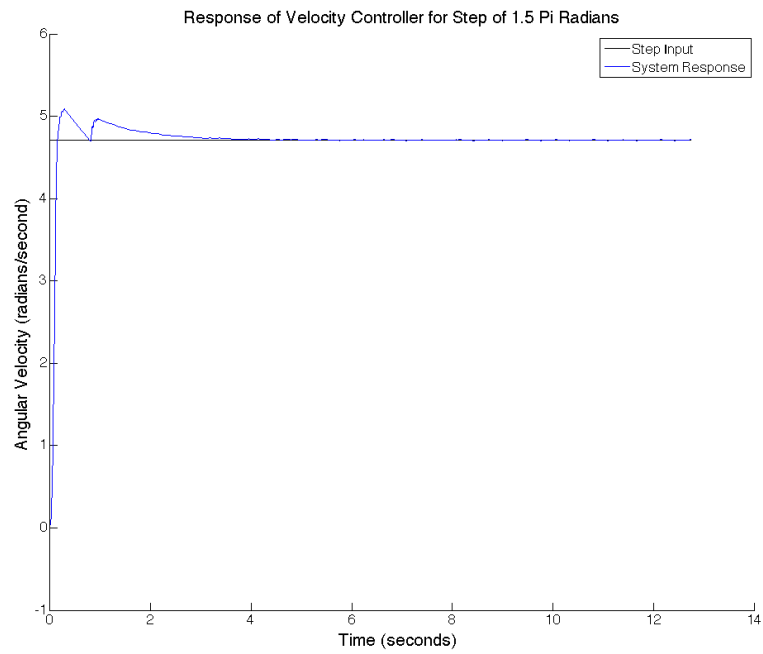


Figure 36: Velocity Controller Step Response

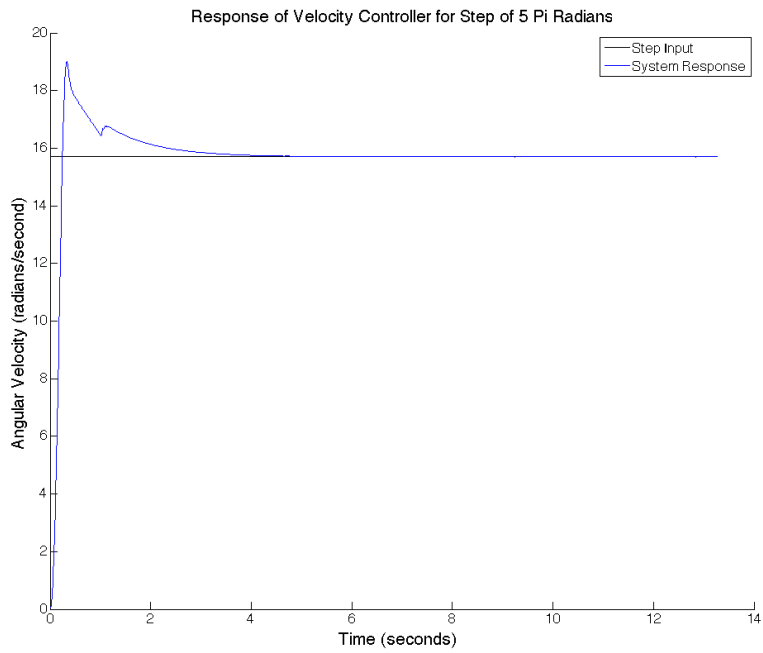


Figure 37: Velocity Controller Step Response

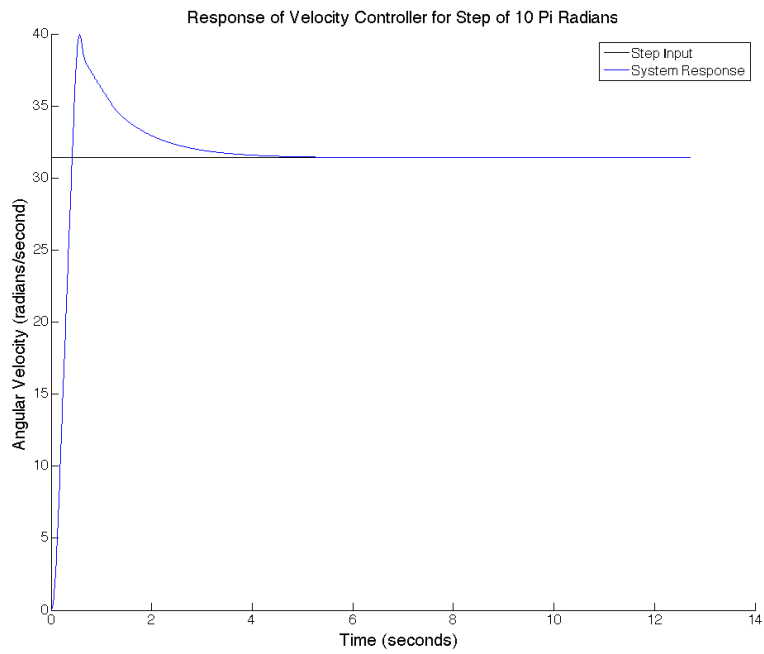


Figure 38: Velocity Controller Step Response

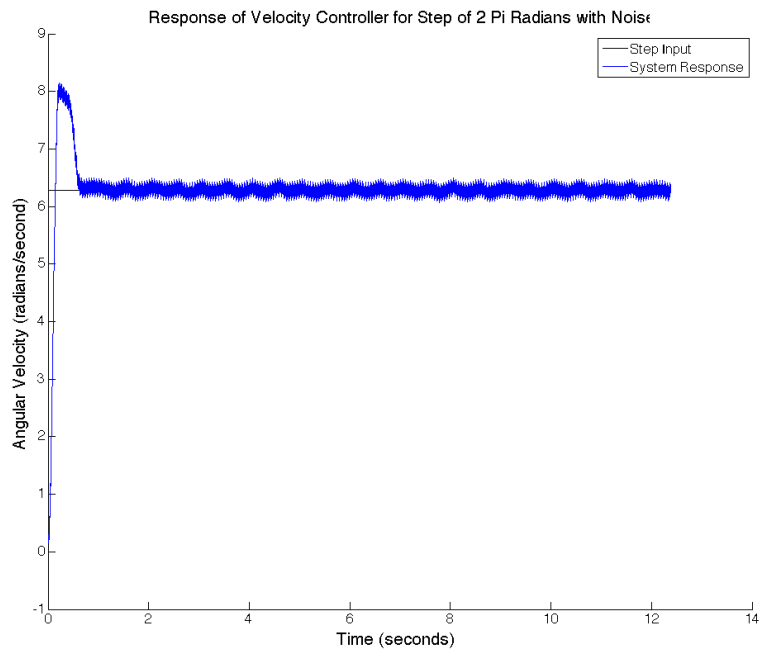


Figure 39: Velocity step response for input of $2 \cdot \pi$ with a 60 Hz noise signal (amplitude = 1)

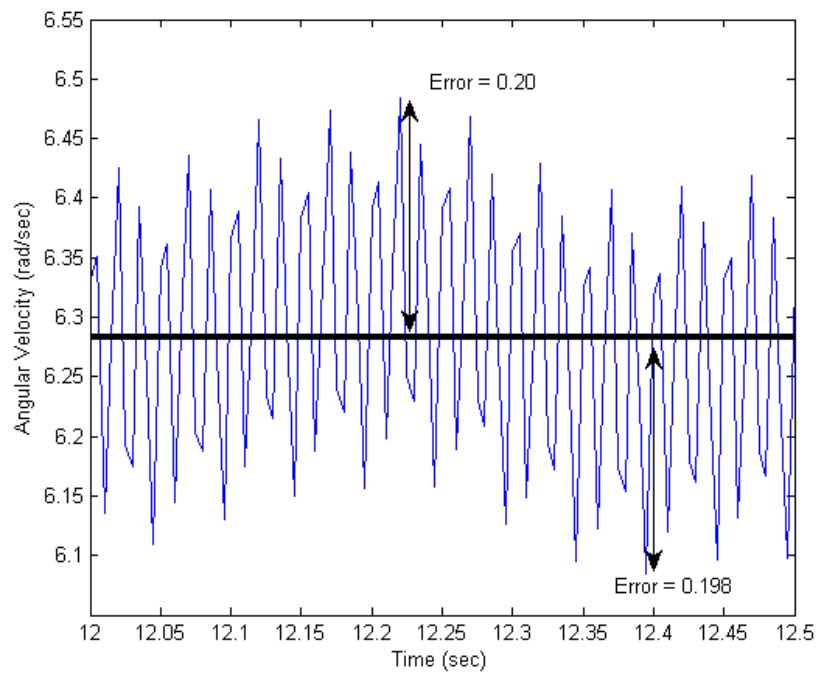


Figure 40: Steady state velocity step response for input of $2\cdot\pi$ with a 60 Hz noise signal (amplitude = 1)

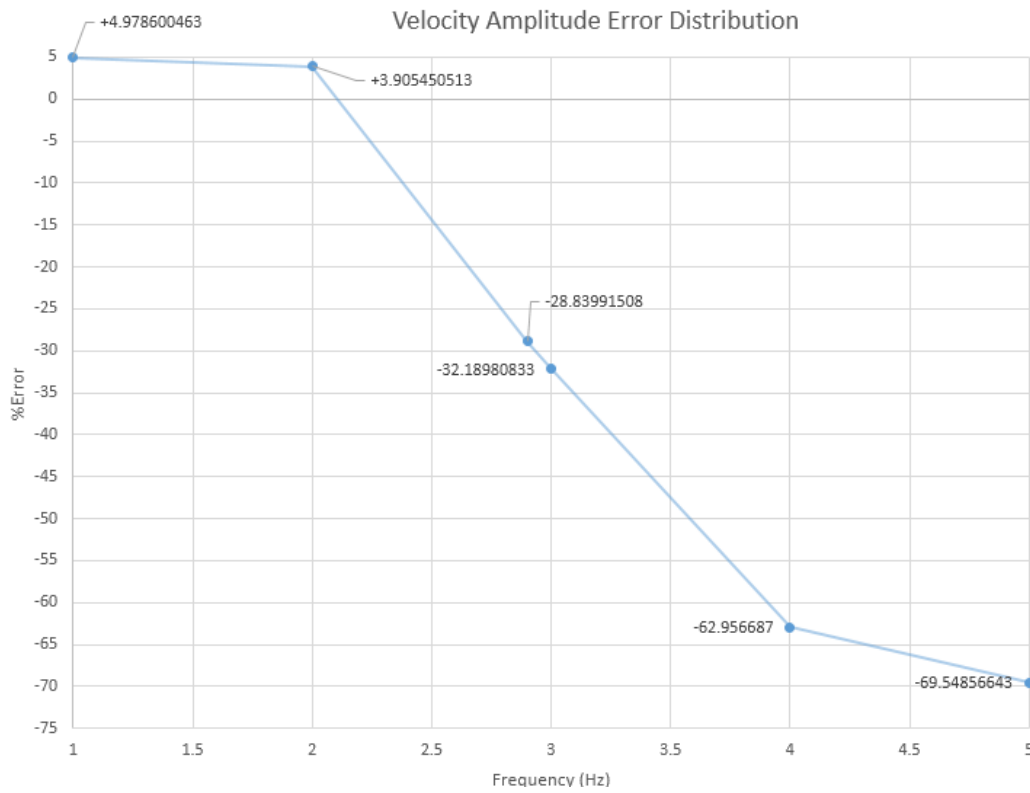


Figure 41: Velocity error vs. frequency for an input amplitude of $\pi/2$ rad/sec

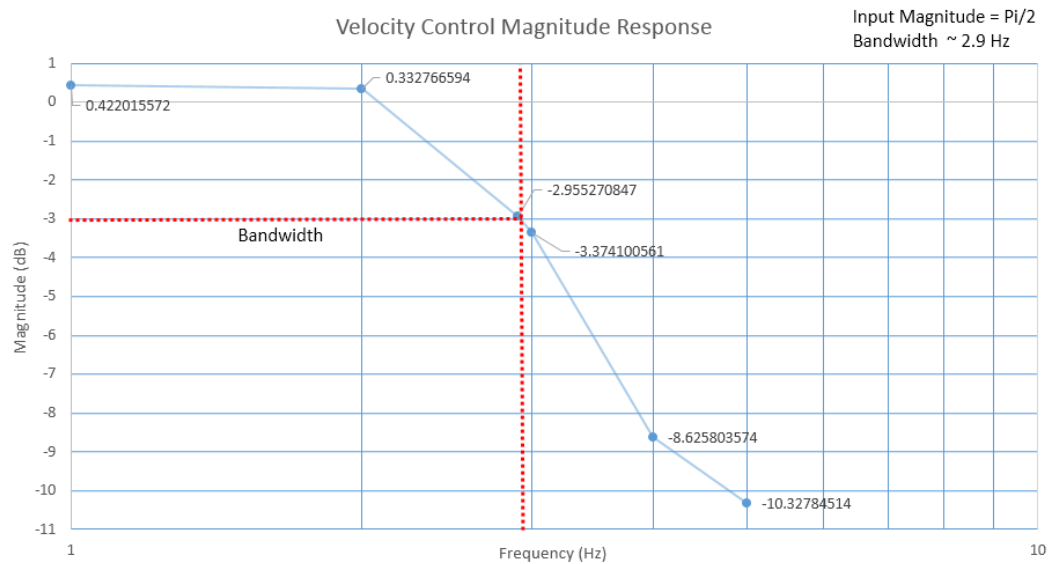


Figure 42: Output magnitude vs. frequency used to determine bandwidth of velocity controller