Lab #3

DC Motor

MECHENG 552

Team 4B

Saptadeep Debnath
Manavendra Desai
Ellen Kim
David Radtke

Professor Shorya Awtar
25 October 2019

Department of Mechanical Engineering
University of Michigan, Ann Arbor

Contents

List of Figures

List of Tables

1 Overall System Modeling (42pts)

Mechanical model

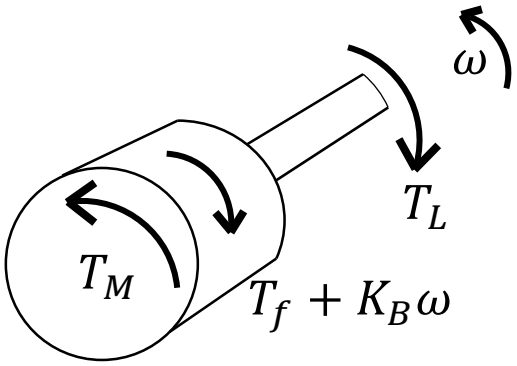

Figure 1: Lumped physical model for the mechanical aspects of the D.C motor

Figure 1 shows the free body diagram for the D.C motor. T_M is the torque available at the motor. $T_f + K_B\omega$ is the friction and viscous damping torque. They internally load the motor and reduce the torque available at the shaft. T_L is the load torque the motor must overcome while ω is the motor shaft speed.

In the above free body diagram, it is assumed that

1. The D.C motor shaft is rigid.
2. Lubricants contained in the motor behave as Newtonian fluids, hence their viscosity remains unchanged over our range of operation. Therefore, viscous damping is always proportional to motor speed.
3. Friction torques (static and dynamic) remain unchanged over our range of operation.

Equation of motion for the motor rotor relating the motor driving torque T_M and rotor position θ is given by

$$J\ddot{\theta} + B\dot{\theta} = T_M - T_L - T_f \tag{1}$$

In the absence of an external load at steady state, $\ddot{\theta} = 0$ and $T_L = 0$. Therefore,

$$B\dot{\theta_{ss}} = T_{M,ss} - T_f$$

Electrical model

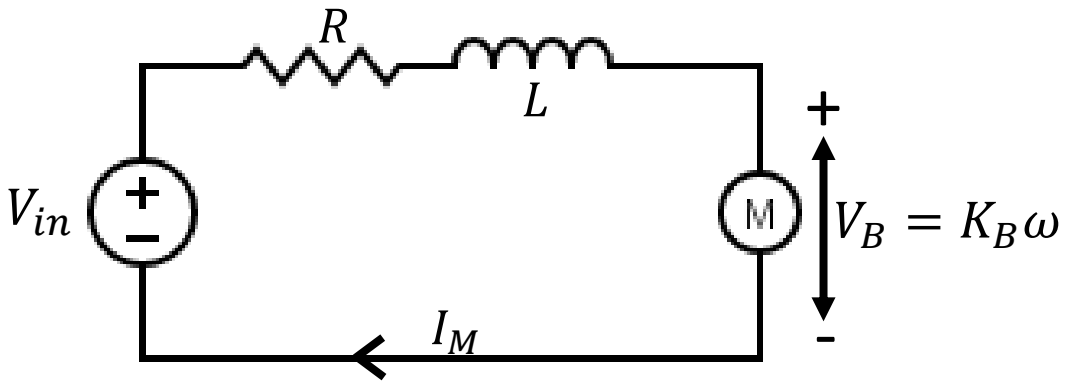

Figure 2: Lumped physical model for the electrical aspects of the D.C motor.

Figure 2 shows the simplified electric cirucit diagram for the D.C motor. V_{in} is the voltage applied across the motor leads, by the servo amplifier. R and L are the motor resistance and inductance respectively. V_B is the back emf generated due to a motor shaft speed ω. I_M is the resulting current in the motor.

In developing the above simplified lumped physical model, it is assumed that

1. The resistance and inductance remain unchanged over our range of operation.
2. The magnetic field of the permanent magnets of the D.C motor remains unchanged over our range of operation.
3. The torque constant remains unchanged over our range of operation.
4. The slope of the Torque - Speed curve, i.e, the damping constant D (as given in the Pittman motor datasheet), remains unchanged over our range of operation.

$$V_{in} - K_B\dot{\theta_{ss}} = I_M R + L\frac{dI_M}{dt} \tag{2}$$

At steady state,

$$V_{in} - K_B\dot{\theta_{ss}} = I_{M,ss}R = \frac{T_{M,ss}R}{K_T}$$

Combining Eq. 1 and Eq. 2, and writing in the s domain, V_{in} can be related to $\dot{\theta}$ (or ω) as follows -

$$\omega(s) = V_{in}(s)\left(\frac{K_T}{K_T K_B + (Js+B)(Ls+R)}\right) - T_f\left(\frac{Ls+R}{K_T K_B + (Js+B)(Ls+R)}\right) \tag{3}$$

$$\theta(s) = \frac{\omega(s)}{s} \tag{4}$$

Eq. 3 and Eq. 4 relate the inputs V_{in} and T_f (a disturbance) to the relevant outputs ω and θ. Any noise that may arise due to the quantization scheme of the encoder and the time discretized loop of myRio has been ignored.

(a) Parameter Identification (20)

All the physical parameters needed in the previously defined DC motor model are shown below.

Table 1: Physical parameters of importance in the DC motor model. Values have been taken from the motor datasheet.

Parameter	Symbol	Unit	Value
Rotor inertia	J	kg-m^2	8.5E-06
Viscous damping	B	N-m-s	3.7E-06
Torque constant	K_T	N-m-A^{-1}	4.24E-02
Back EMF constant	K_B	V-rad^{-1}-s^{-1}	4.24E-02
Dynamic friction torque	$T_{f,d}$	N-m	5.6E-03
Inductance	L	mH	1.97
Resistance	R	Ω	1.85
No-Load current	I_{nl}	A	0.18
Servo-amplifier gain	K_a	A/V	0.99

However, if the motor datasheet was unavailable, the following experiments can be carried out to identify the necessary parameters.

Resistance - Stall the motor shaft by holding it with your hand. Provide a high enough D.C voltage input to the servo-amplifier so as to drive a high enough current $I_M \sim O(10^2)$ mA in the motor. Measure the voltage V across the motor leads using the *power connector* pins 1 (Motor A) and 2 (Motor B) on the mounting card. Read I_M into myRio through the current monitor pin 6 (also a power connector pin) on the mounting card. Resistance can then be calculated using the relation $R = V/I_M$.

Back EMF Constant and **Torque Constant** - Run the motor at a constant speed $\omega \sim$ 1000 to 3000 rpm by supplying a suitable D.C voltage input to the servo-amplifier. Once again measure the voltage V across the motor leads using the *power connector* pins 1 (Motor A) and 2 (Motor B) on the mounting card. Read motor current I_M using myRio and the current monitor pin 6 (also a power connector pin) on the mounting card. Back EMF Constant can then be calculated as $K_B = (V - I_M R)/\omega$. Based on the physics of a D.C motor, the torque constant $K_T = K_B$.

Viscous Damping and **Dynamic Friction Torque** - Run the motor at two speeds, $\omega 1$ and $\omega 2$. Record the corresponding motor currents $I_{M,1}$ and $I_{M,2}$. Since the motor operates with no external load, the only loads present are internal, in the form of a dynamic friction torque $T_{f,d}$ and viscous damping $B\omega$. Therefore, under steady state conditions, the motor torque $T_m = I_M K_T = T_{f,d} + B\omega$. Using two sets of readings for I_M and ω, we obtain

$$B = K_T \left(\frac{I_{M,1} - I_{M,2}}{\omega 1 - \omega 2} \right)$$

$$T_{f,d} = I_M K_T - B\omega$$

No Load Current and **Static Friction Torque** - Starting with a zero D.C voltage input V to the servo-amplifier, increase V up till the point the motor just begins to rotate continuously. Under such an operating condition, the motor current measured is the no load current I_{nl} and the static friction torque is given by $T_{f,s} = I_{nl}K_T$.
Rotor Inertia and Inductance - Provide a sinusoidal D.C voltage input V to the servo-amplifier. Record the resulting sinusoidal motor current I_M and sinusoidal motor speed ω through myRio. Evaluate $\dot{I_M}$ and $\dot{\omega}$. The following pair of equations can then be used to determine J and L.

$$J\dot{\omega} + B\omega = I_M K_T - T_{f,d}$$

$$V_{ref} = I_M R + L\dot{I_M} + K_B\omega$$

V_ref is the voltage applied across the motor leads by the servo-amplifier and can be measured using the *power connector* pins 1 (Motor A) and 2 (Motor B) on the mounting card.
Servo amplifier gain - Run the D.C motor by supplying a known D.C voltage input V to the servo-amplifier. Measure motor current I_M using using myRio and the current monitor pin 6 (also a power connector pin) on the mounting card. Use the relation K_a to obtain the servo amplifier gain.

Table 2: Physical parameters of importance in the DC motor model. Experimentally obtained parameter values have been compared with those given in the Pittman motor datasheet.

Symbol	Unit	Datasheet Value	Experimental value
J	kg-m^2	8.5E-06	-
B	N-m-s	3.7E-06	5.9E-06
K_T	N-m-A^{-1}	4.24E-02	4.15E-02
K_B	V-rad^{-1}-s^{-1}	4.24E-02	4.15E-02
$T_{f,d}$	N-m	5.6E-03	5.1E-03
$T_{f,s}$	N-m	-	6.2E-03
L	mH	1.97	-
R	Ω	1.85	1.61
I_{nl}	A	0.18	0.15
K_a	A/V	1	0.99

A significant difference is observed between the experimental and datasheet values of the motor viscous damping coefficient B ($\sim$ 60% error). Since $B \sim O(10^{-6})$, small errors in measuring motor shaft speed ω using a position sensor can result in a large deviation in B. Such errors are inherent in a digital measurement system such as the encoder. There will always exist an inherent lag in its evaluation of ω due to time being discretized. Also, based on our experimental procedure, errors in evaluating K_T will get compounded while calculating B.

(b) Actuator Driver Modeling (2)

Actuator driver has a bandwidth of around 2.5 kHz. It is therefore assumed that the gain K_a A/V is unaffected by the system dynamics and remains unchanged over the range of frequencies of our interest (*O(1)* Hz). The driver is hence modelled as a constant gain block.

(c) Sensor Modeling (2)

The sensor (encoder) used in this experiment is unaffected by the system dynamics, i.e. having a transfer function,

$$H_s = 1$$

Therefore the sensor is modelled as a unity gain in the Simulink environment. Whereas, in the real life implementation of the controller in LabVIEW, the encoder block outputs the number of counts of the pulse edge rising and falling and needs to be converted into angles. Counts per revolution (CPR) of the encoder is 500, which is then multiplied by 4; as quadrature decoding is implemented by the encoder to decode the signal, which gives the total number of counts per revolution. Dividing this by 360 gives the relation between counts to angles in degrees. The counter value from the encoder is passed through a gain block of 0.18 in the LabVIEW to get the angular position in degrees. Additionally, a gain block of 0.0175 is also added to convert the angles measured in degrees to radians.

(d) Position Control Block Diagram (3)

The system shown in figure 3, is considered to be operating in the current driving mode, i.e, the current does not change with time. Hence, the electrical dynamics of the motor can be omitted from the motor transfer function, which can be further written as:

$$\theta(s) = \frac{1}{s} \cdot I_m(s) \cdot \left(\frac{K_T}{(Js + B)} \right) - T_{NL} \tag{5}$$

The plant transfer function as shown in Fig. 3 can be given as:

$$\frac{\theta(s)}{T_{in}} = \frac{1}{s} \cdot \frac{1}{(Js + B)} \tag{6}$$

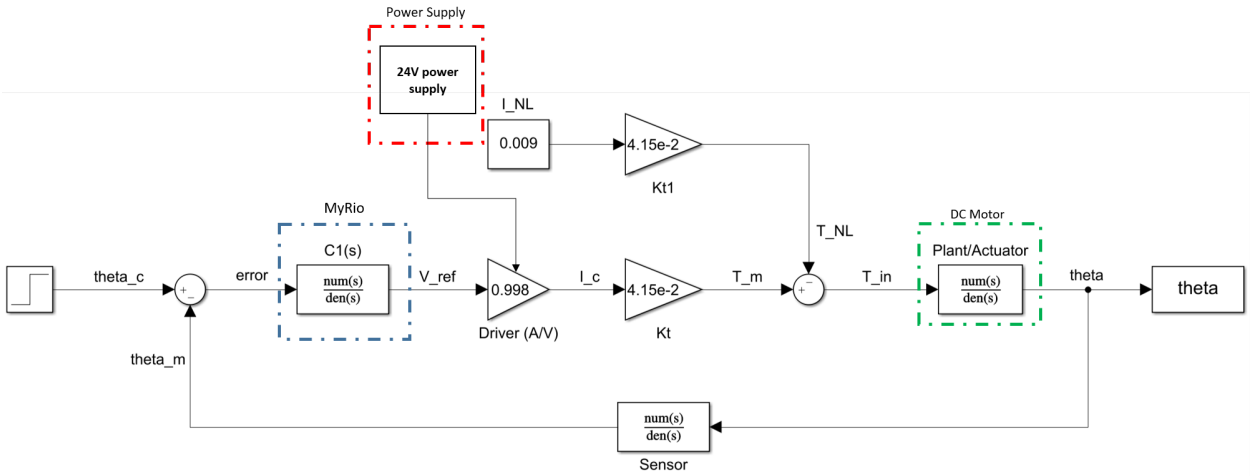

Figure 3: Position Control Block diagram

(e) Velocity Control Block Diagram (3)

The plant transfer function for the velocity control block diagram as shown in figure 4 is given as:

$$\frac{\omega(s)}{T_{in}} = \frac{1}{(Js + B)} \tag{7}$$

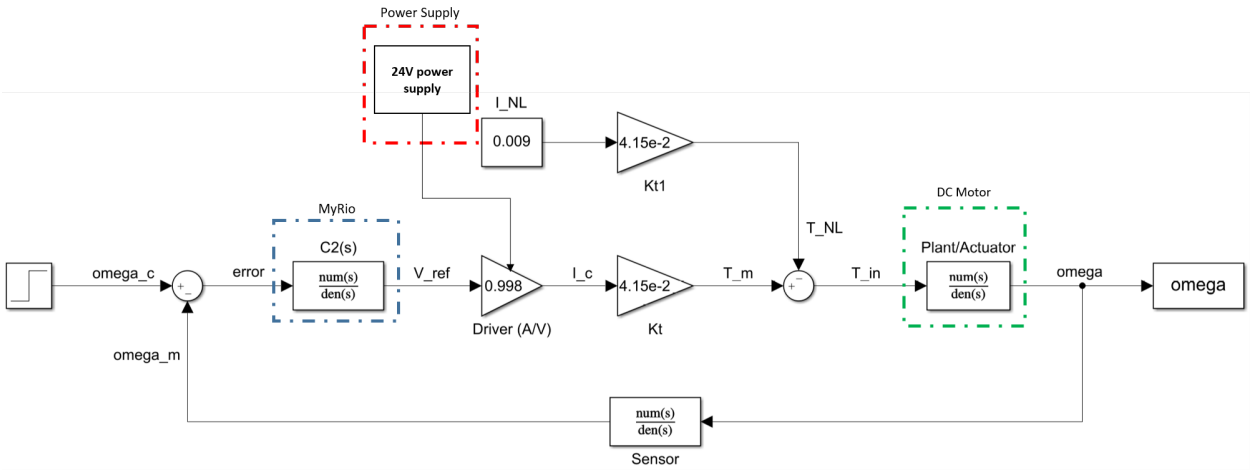

Figure 4: Velocity Control Block diagram

(f) Simulink Model (5)

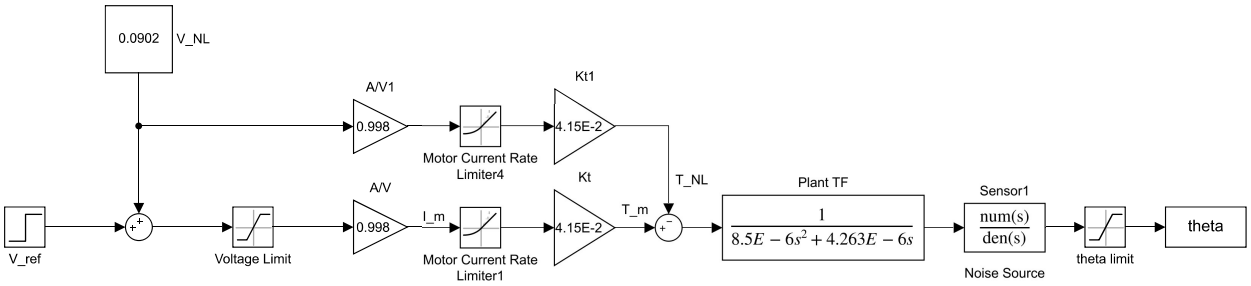

Figure 5: Open Loop Block Diagram for angular position

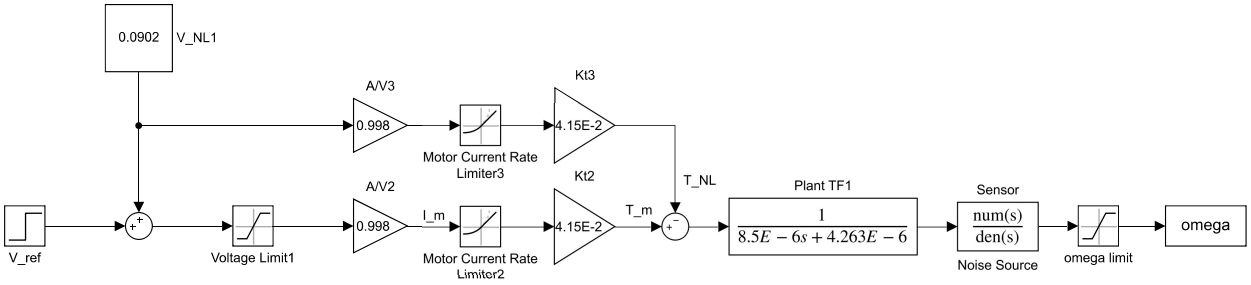

Figure 6: Open Loop Block Diagram for angular velocity

2 Position Control using Driver Current Mode of Driver (36pts)

(a) Controller Design in MATLAB/SIMULINK (21)

The open loop transfer function is derived as

$$P_1(s) = \frac{\theta}{V_{ref}} = \frac{K_t K_{amp}}{Js^2 + Bs} = \frac{0.04142}{8.5E-6s^2 + 4.263E-6s} \tag{8}$$

The designed controller must satisfy the following conditions:

- Keep steady state error below 2% for step command of 1 radian
- Keep overshoot less than 30% of the step command value of 1 radian
- Provide command tracking: Amplitude should remain within 5% of commanded amplitude of 1 radian, up until 5kHz
- Attenuate 1kHz noise by at least 10 times

From these conditions, the design requirements of the damping ratio and the natural frequency are calculated. The damping ratio is related to the overshoot by

$$\xi = \frac{|ln(\frac{OS}{100})|}{\sqrt{\pi^2 + ln^2(\frac{OS}{100})}}$$

where OS represents the overshoot in percentage. Thus, for the overshoot to be less than 30%, the damping ratio must be greater than 0.3579. As well, for the system to be stable and robust, it must have a phase margin greater than 0 degrees. The phase margin is defined by the open loop cross over frequency. The open loop cross over frequency for equation 8 was measured to be 69.81 rad/sec and the natural frequency defined by

$$\omega_n = \frac{\omega_d}{\sqrt{\xi}} = 116.70\frac{rad}{sec}$$

With these design requirements, they were imposed on the SISO tool's root locus to create bounds. The open loop root locus shows an asymptotes' intersection on the left hand of the s-plane. To implement a system which has good stability and good response speed, we first began with designing a lead controller by placing a zero near the origin and a pole far from the origin. This cause the asymptotes' intersection on the root locus to shift to the left of the complex plane. The final lead position controller from SISO tool was

$$C_1(s) = \frac{9.8836s + 56.25}{s + 185} \tag{9}$$

This controller satisfied all the required conditions as shown in figure 7, 8, and 9 and the results are summarized in table 3.

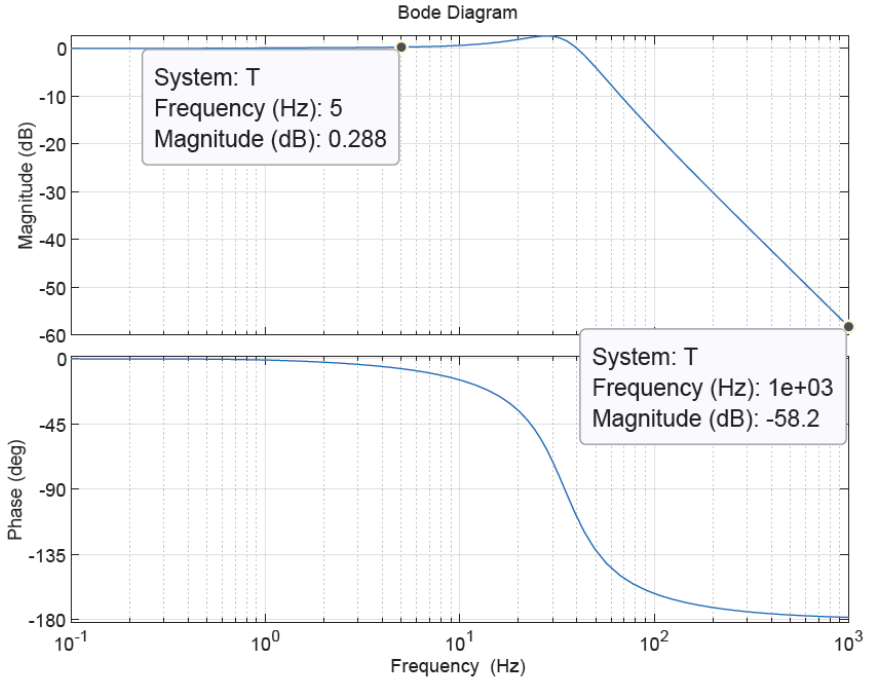

Figure 7: Bode plot of system with position controller from SISO tool implemented. The magnitude at 5Hz and 1kHz are shown to satisfy command tracking and attenuation

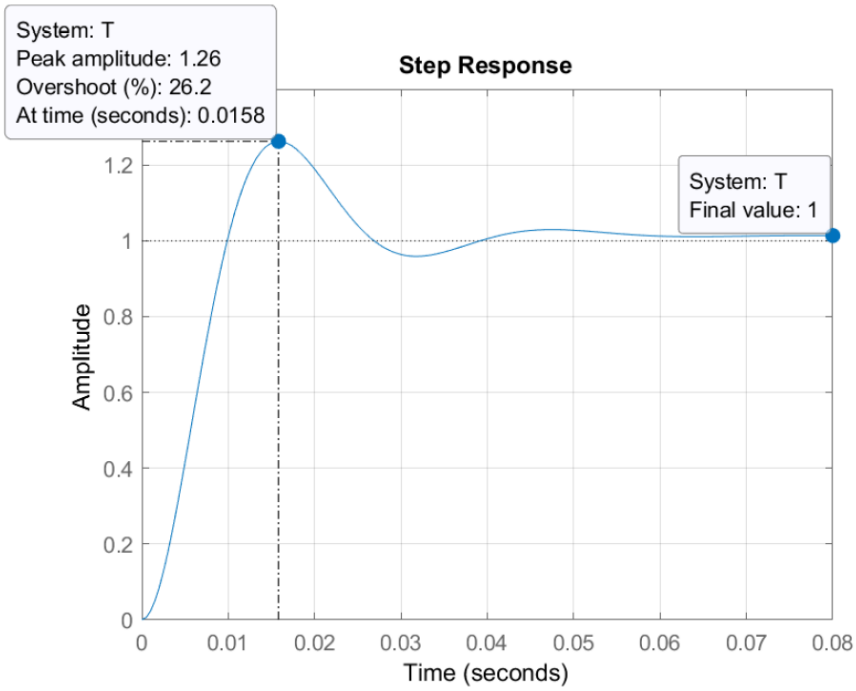

Figure 8: Step response of system with position controller from SISO tool implemented. The overshoot and steady state error is shown to satisfy the requirements. The amplitude of the step response is the angular displacement and is given in radians.

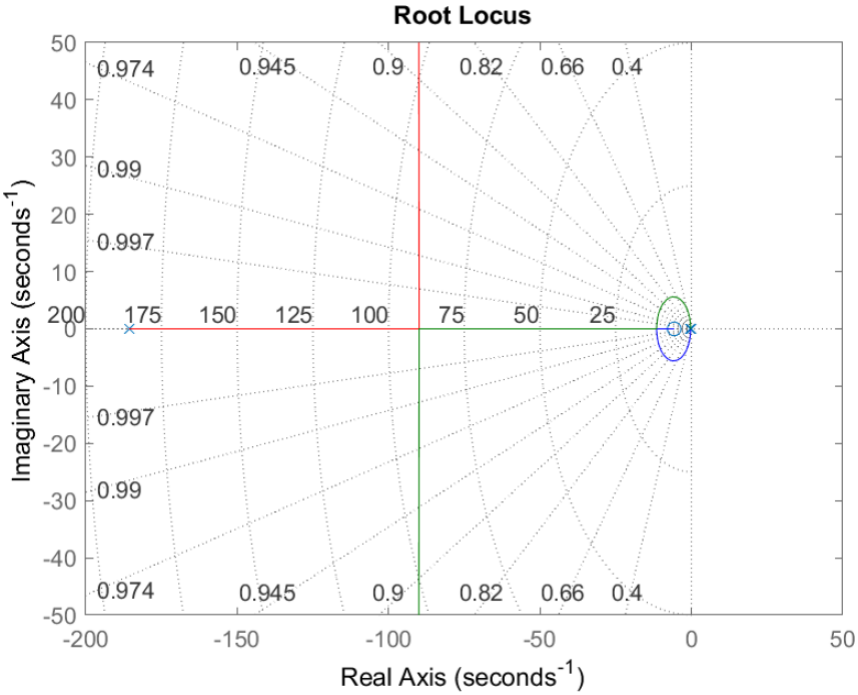

Figure 9: Root Locus of system with position controller from SISO tool implemented. The asymptotes' intersection has been diverted to the left side of the s-plane.

Table 3: Position Controller Specifications from SISO tool

Parameters	Value
Steady State Error	0%
Overshoot	26.2%
Command Tracking	3.5%
Attenuation	812 times
Closed Loop Bandwidth	47.4Hz
Phase Margin	70.5 deg

With the controller from the SISO tool, it was tested in MATLAB's Simulink tool. Along with the transfer function of the controller, plant and the amplifier, the system sees disturbance of no load voltage as well. A saturation block at the output of the controller is placed to act as a practical safety mechanism to avoid damage to physical system later on. The saturation voltage block is from 3.03 to -3.03V. The saturation voltage was calculated by taking the continuous torque, 8.1×10^{-2} Nm, provided by the datasheet, and dividing experimental torque constant and multiplying experimental resistance. With this closed loop system, it produced a step response different from figure 8 as it does not have any overshoot and it has a spike in displacement at 0.017 seconds before reaching the steady state of 1 rad, shown in figure 10. However, since 0% overshoot and this spike does not affect satisfying the requirements mentioned previously, the overshoot will be accepted and the spike will be ignored. The spike seems to have been the overshoot that was seen in the SISO tool step response that had been displaced below the steady state value. The step response in Simulink shows similar results to the results from SISO tool. When a sinusoidal wave of 5Hz with amplitude of 1 is entered into the Simulink system, it results in figure 11, which reaches peak of 1.0335 and -1.0350 radians. This varies from the SISO tool; however, it still satisfies the command tracking of amplitude within 5%. The variation between the SISO tool and the Simulink comes from the added disturbance into the system through the no load current. Despite the variation, since the Simulink results still satisfy the design requirements, the same controller from the SISO tool will be carried into LabVIEW for true physical response of the system through the given controller.

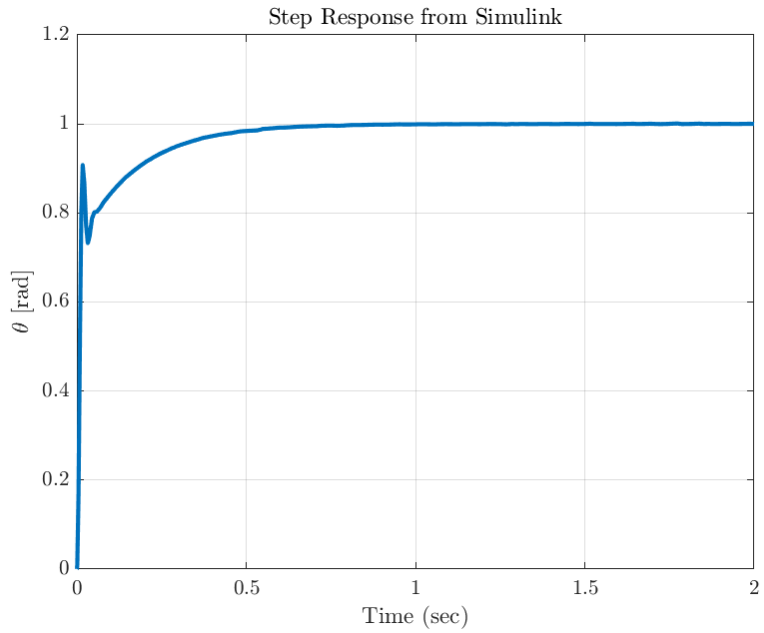

Figure 10: Step response of system with position controller in Simulink. There is a peak in position at 0.017 seconds; however, since the disturbance does not affect the overall steady state of the system, the peak will be ignored. The steady state error is 0% as it reaches final value of 1 radian and there is no overshoot.

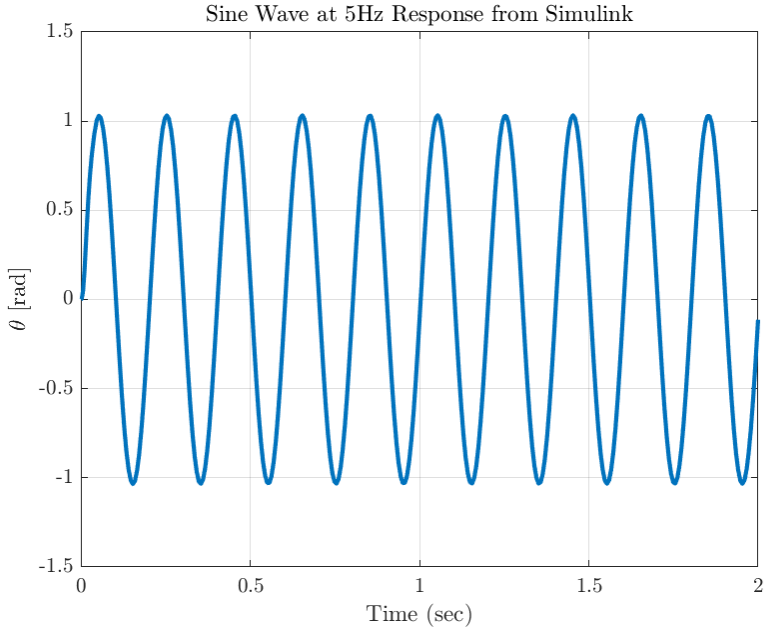

Figure 11: Sine Wave at 5Hz response of system with position controller in Simulink. The peak of the waves occur at 1.0335 and -1.0350 radian, fulfilling the 5% tracking command.

(b) Controller implementation and Evaluation in LabVIEW (15)

When the position controller was implemented into LabVIEW, the position of the zero was moved further away. The final position controller implemented in LabVIEW was

$$C_1(s) = \frac{9.8836s + 500}{s + 185} \tag{10}$$

The difference in the position controller between simulation and real world maybe due to the unmodeled dynamics of the system. Instead of a lead controller, the final controller was a lag controller. Figure 12 shows the step response of the system to the controllers implemented in SISO tool, Simulink and Labview. The step response in LabVIEW met all the design constraints as shown in table 4, but has multiple peaks as it approaches the steady state. In comparison to the SISO tool step response, the LabVIEW response has a smaller overshoot and larger steady state error. In comparison to the Simulink results, the system's response to a step input of 1 rad provide larger overshoot of 0.2%, larger steady state error of 1.2%. However, the general trend between the step response in simulink and labview are similar despite the change in zero location on the position controller. The command tracking was checked by experimentally verifying the amplitudes of the response from 0.1 to 5 Hz as shown in table 5. The command tracking value increased with the new controller implemented; however, satisfies the design constraints still. The experimental closed loop bandwidth was at 62Hz for the new controller and the theoretical closed loop bandwidth from SISO tool was at 47.4Hz. The difference between the two values is 30.8% due to the difference in the controller implemented as the zero of the controller implemented in LabVIEW was much further from the origin, causing a shift in the closed loop bandwidth. In general, the experimental closed loop bandwidth is better than the SISO tool as expected as the controller was fined tuned on LabVIEW for better control and stability.

Table 4: Position Controller Specifications from LabVIEW

Parameters	Value
Steady State Error	0.2%
Overshoot	1.2%
Command Tracking	4.9%
Closed Loop Bandwidth	62Hz

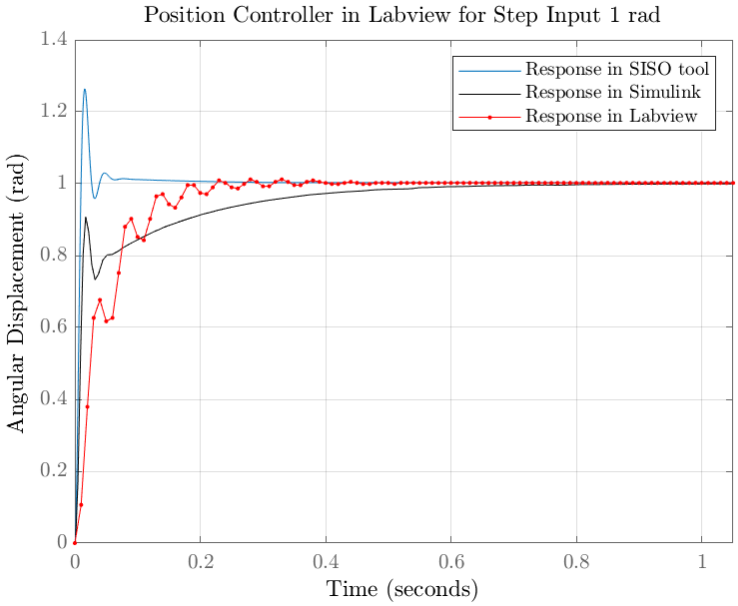

Figure 12: Step response of system with position controller implemented in LabVIEW. The step response shows overshoot of 1.2%, steady state error of 0.2%, and have multiple peaks in angular displacement till reaching steady state.

Table 5: Position controller command tracking from LabVIEW. Command tracking is within 4.6% up to 5Hz.

Frequency [Hz]	Amplitude [rad]
0.1	1.009
0.2659	1.002
0.7071	1.002
1.8803	1.015
5	1.046

3 Velocity Control using Driver Current Mode of Driver (35pts)

(a) Controller Design in MATLAB/Simulink (20)

The open loop transfer function is derived as

$$P_2(s) = \frac{\dot{\theta}}{V_{ref}} = \frac{K_t K_{amp} s}{Js^2 + Bs} = \frac{0.04142}{8.5E-6s + 4.263E-6} \tag{11}$$

The designed controller must satisfy the following conditions:

- Keep steady state error below 2% for step command of 2π radians/sec
- Keep overshoot less than 30% of the step command value of 2π radians/sec
- Provide command tracking: Amplitude should remain within 5% of commanded amplitude of $\frac{\pi}{2}$ radians/sec, up until 5kHz
- Attenuate 1kHz noise by at least 10 times

In theory, there are many controllers of trivial complexity that would allow the system to meet these constraints. The reason for this is that this plant has only a pole and it's on the real axis and it doesn't have any zeros. Therefore, the system should asymptotically approach the command with no overshoot in accordance with its first-order behavior. The only modification that should be necessary is to increase the controller gain in order to bring the closed-loop pole to the proper frequency to meet the bandwidth requirement and noise attenuation requirement. A unity gain proportional controller was considered, but it did not provide the required noise attenuation. From varying the proportional gain of the controller in SISO tool, we decided to start with a value of 0.12.

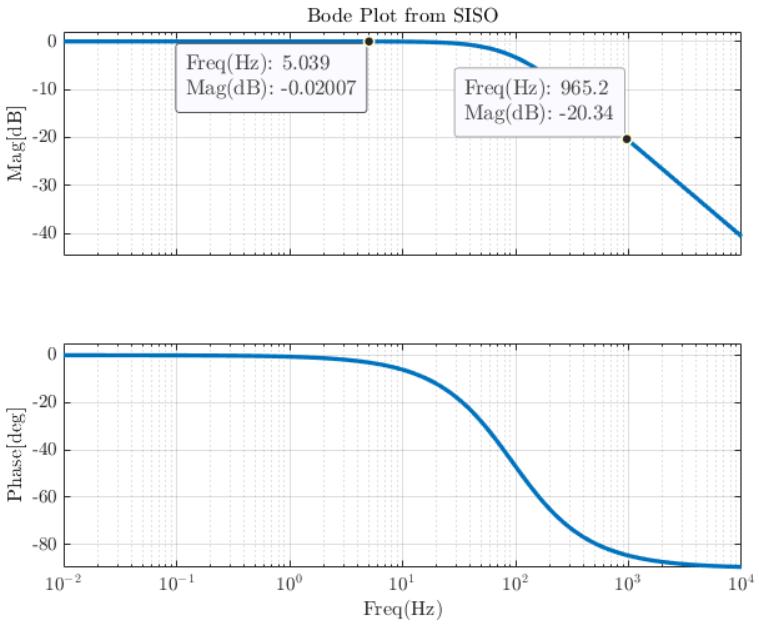

Figure 13: Bode plot of system with velocity controller from SISO tool implemented. The magnitude at 5Hz and 1kHz are shown to satisfy command tracking and attenuation

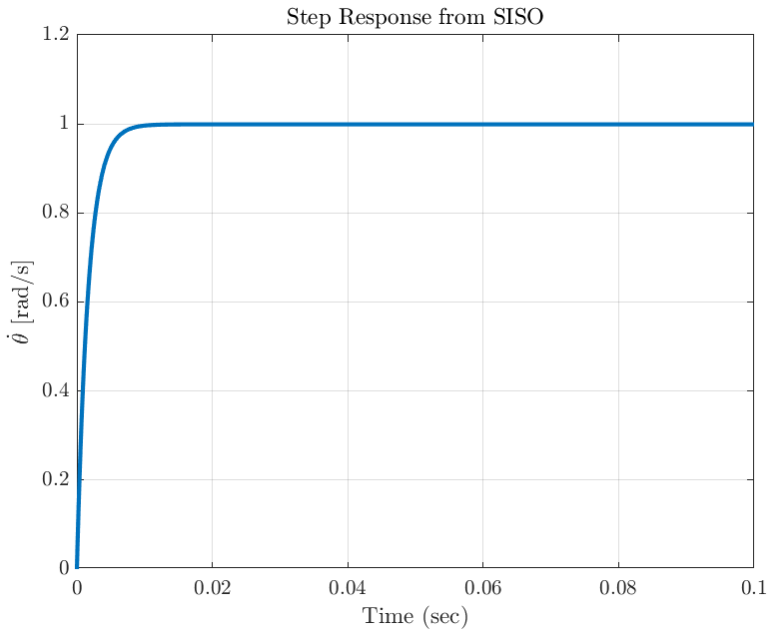

Figure 14: Step response of system with Velocity controller from SISO tool implemented. The overshoot and steady state error are shown to satisfy the requirements. The amplitude of the step response is the angular velocity and is given in radians/sec.

Table 6: Velocity Controller Specifications from SISO tool

Parameters	Value
Steady State Error	0%
Overshoot	0%
Command Tracking	0.2%
Attenuation	10.8 times
Closed Loop Bandwidth	93.3Hz
Phase Margin	90 deg

We proceeded to test the step response and freqeuncy response controller in Simulink which returned figures 15 and 16. The command tracking error at 5Hz was found to be 0.2%, which matches the value from SISO tool. The steady state error of the step response was 0.1% at 0.1 seconds. Higher resolution for the calculations doesn't seem to reduce this value, nor does calculating error at 100 seconds. This indicates that the steady state error is the result of a difference between the SISO tool and our Simulink model like the friction torque or voltage saturation. As with the SISO tool there was no overshoot.

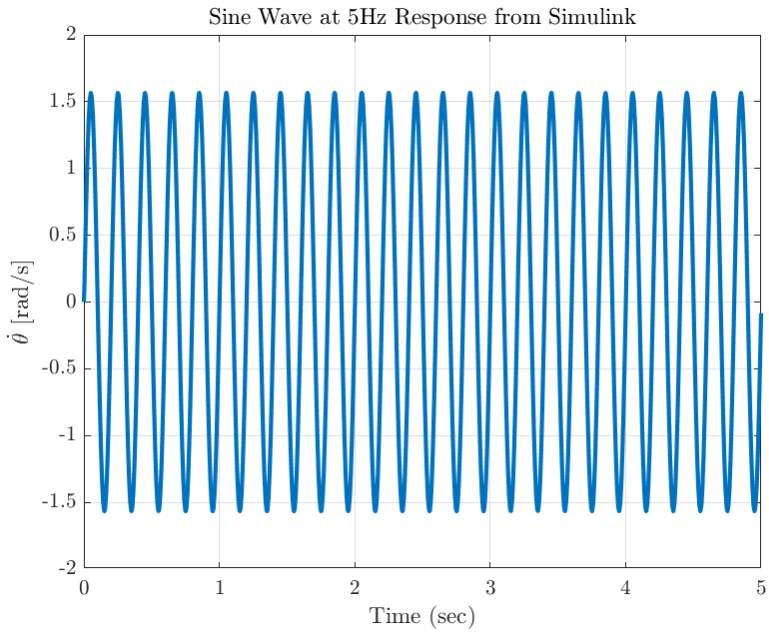

Figure 15: Response of the system in Simulink at 5Hz. There is a 0.2% error at the peaks and troughs of the wave.

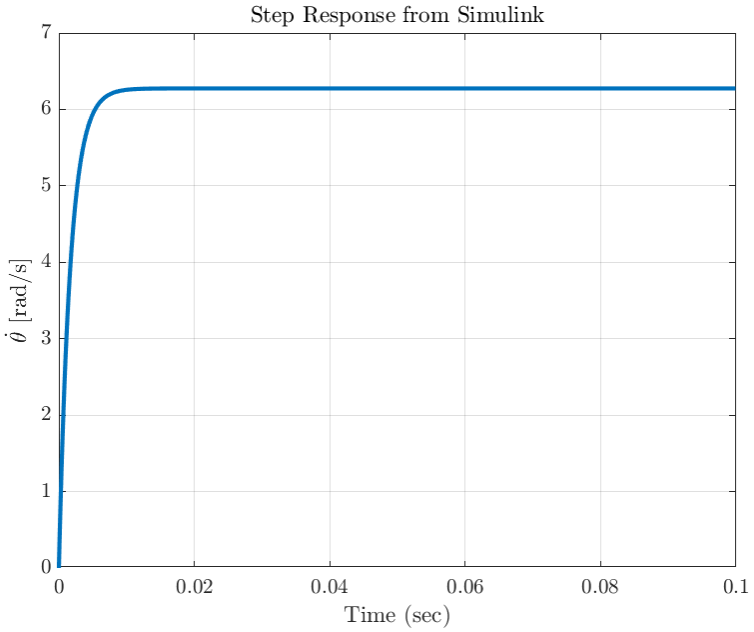

Figure 16: Step response of system with Velocity controller from Simulink model. The overshoot and steady state error are shown to be similar to the SISO tool's results.

(b) Controller Implementation and Evaluation in Lab (15)

When the velocity controller was implemented for the physical system, it garnered results that were very different from the model's predictions. In general, the velocity values were sporadic which ended up causing system instability when we used only a proportional controller. In order to compensate for this we implemented both a first-order filter for our encoder measurements, as well as adding a pole to our controller. To increase stability we decided to add a zero as well. We finally arrived at the controller shown in Eq. 12

$$C_2(s) = \frac{0.5s + 40}{s + 150} \tag{12}$$

With this controller we were not able to measure the overshoot because the current swing when turning on the power switch in LabVIEW drew too much power from the power supply and caused it to shut off. This is despite our best attempts to limit the power draw by setting our voltage command saturation to as low as ± 1.75 volts. We suspect that the nature of the method that the driver uses to drive the current will attempt to draw on a voltage higher than our $V_{command}$. We were still able to measure a steady-state value for the 2π by incrementally increasing the values of our numerator until we arrived at the described values. The resulting steady-state response on average meets the steady-state error requirement, but in general it does not.

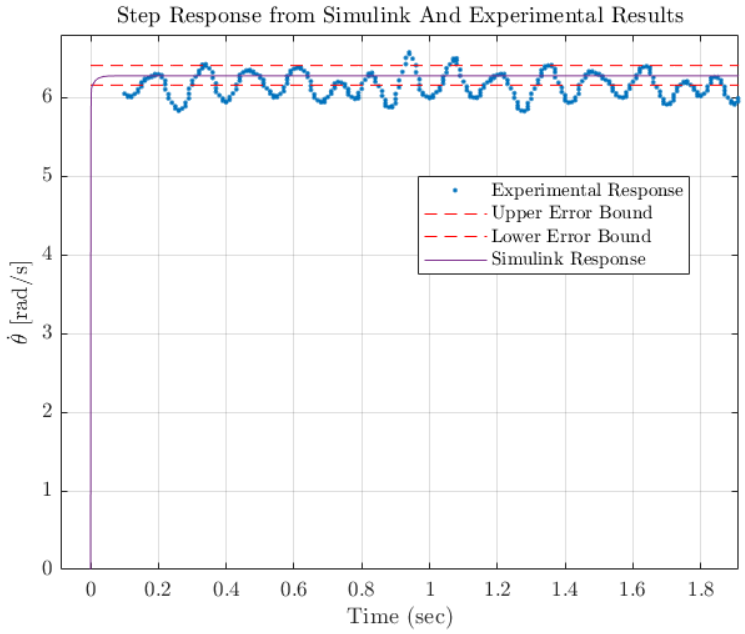

Figure 17: Experimental Response And Simulink Response of the Final System. The average error in steady-state is 1.96%. The error bounds are shown as dashed lines 2% above and below the command velocity.

Measurements for command tracking were taken linearly from 2Hz to 5Hz. We then compared the maximum and minimum values of the response wave to our command value of

2π radians/sec. Our controller was close to meeting the requirements for the crests with a maximum error of 5.9%, but due to a large offset that shifted the response up it failed to meet the requirements. This can be shown in Table **??**.
To measure the bandwidth we increased the frequency for a command velcoity of 1 rad/sec until the response was equal to 0.7, the half power bandwidth. We found the bandwidth for this controller to be 29Hz as shwon in table 7

Table 7: Experiemental Velocity Controller Specifications

Parameters	Value
Steady State Error	1.96%
Overshoot	N/A%
Command Tracking	See table 8
Closed Loop Bandwidth	29Hz

Table 8: Experimental Results For Velocity Command Tracking

Frequency [Hz]	Crest Error [%]	Trough Error [%]
2	2.5	19.2
3	3	16.8
4	5.9	19.2
5	2.1	21.1

4 Controller Implementation Practical Considerations (47pts)

(a) Continuous vs. Discrete Control (14)

Both the controllers implemented (angular position and angular velocity) are designed in continuous time domain. The controller designed in SISO tool is assumed to be in continuous time domain. For the Simulink, the input is continuous but the solver is discrete. Whereas, the LabVIEW Control and Simulation loop is in discrete time domain. This difference in the time domain between the controller modeling and controller implementation introduces an inherent lag in the overall system.

Theoretically, the system is expected to perform better with a lower loop cycle time since the loop cycle time of LabVIEW is directly related to the quantization step size of the data signal, i.e. it is possible to collect more data points with a smaller step size. However because the encoder is quantized, the step size is limited by the encoder quantization effect. Conversely, the system response should worsen as the cycle loop time is increased due to

collecting less data points, which was observed experimentally.

The Simulink environment is first initialized with 1 ms loop cycle time and then the loop cycle time increased gradually. Initially the system behaves as expected and we get a stable response. But at a certain loop cycle time it breaks down at 12 ms ($\approx$ 83 Hz). Further increasing the loop cycle time drastically changes the system response.

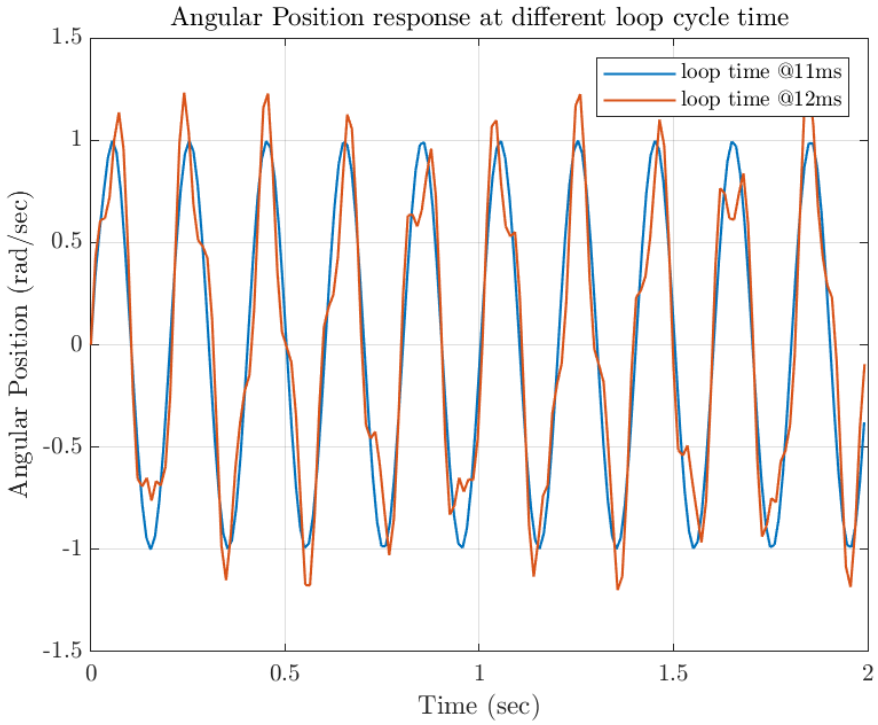

Figure 18: Velocity response at different loop cycle time as observed in Simulink. The system is given a sine input of 1 unit amplitude and 5 Hz frequency.

Similar experiment is carried out on the motor in LabVIEW , where the loop cycle time is increased gradually from the initial cycle time of 5 ms. The system responds as expected where the system starts behaving erratically as the loop cycle is increased. As can be seen in the figure 19 the system response drastically changes at 11 ms ($\approx$ 91 Hz), which is similar to the loop cycle time at which the system response changes drastically in Simulink.

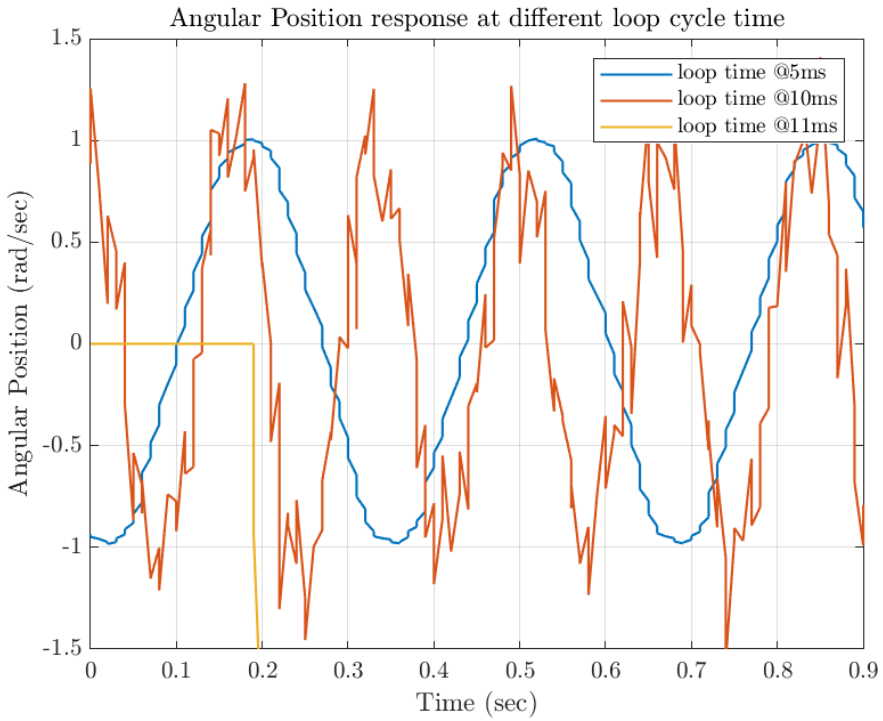

Figure 19: Velocity response at different loop cycle time as observed in LabVIEW. The system is given a sine input of 1 unit amplitude and 3 Hz frequency.

(b) Numerical Derivative (12)

Introducing the 'D' term in LabVIEW to evaluate the motor shaft angular velocity does introduce spikes. Their occurrence was most prominent when the power switch running the motor current is switched on or switched off. The dominant frequency content ν_{spike} of these spikes was found to be $\sim$ 125 Hz. It was measured using one cycle time of the spiked waveform of the angular velocity obtained when a running motor was switched off.
The time constant τ for the first order filter is chosen so that $\frac{1}{\tau} < \nu$. For our case, $\tau = 0.5$ seconds.
If τ is too large, the filter may attenuate the true motor shaft position ν_{signal} and a reduced angular angular velocity ω may be fed back to the controller as $1/\tau << \nu_{signal}$. This can happen even in the absence of spikes, i.e, when the simulation loop time in LabVIEW is not too small. If τ is too low, the filter may generate spikes of its own as $1/\tau >> 1$ and feed it back to the controller.
Spikes resulting from optical encoder quantization, if not filtered, will give an incorrect and spiked measured response to the controller. This will lead to incorrect error calculations and spiked control input voltages to the servo-amplifier. This may cause the power supply to saturate or the motor shaft velocity to oscillate.
Quantization effect of the encoder was studied on the motor shaft velocity controller. The quantization step size Δ for the given encoder be given as $\frac{2\pi}{(2000)(\omega)}$, where ω is the shaft velocity. As seen in Figure 20, giving a step input of $\omega = 1$ rad/s and using $\Delta = 0.0031$, we get the nominal response for our controlled D.C motor shaft velocity model. As we raise Δ,

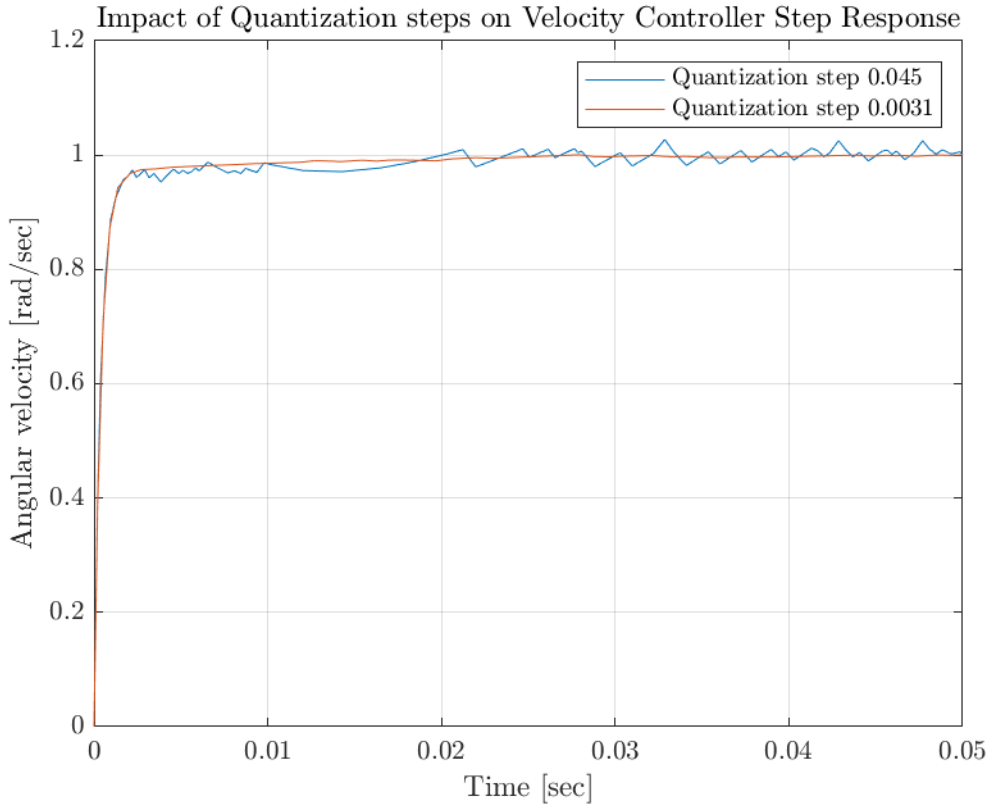

Figure 20: Quantization steps on Velocity Controller Step Response

the response begins oscillating. At $\Delta = 0.045$, the oscillations are large enough to drive ω beyond 1.02, violating the 2% steady state error requirement and significantly affecting the controller performance.

(c) Saturation (9)

In LabVIEW, we had implemented a saturation after the controller to prevent damage to the system. However, by placing the saturation limit, when a higher voltage then 3.03V is asked by the controller, the system cannot meet the value required. Thus, the saturation limit define the maximum voltage that can be sent out of myRio and the maximum torque that the motor can provide. The saturation limits may be modified if the the controller's performance is not satisfactory. Currently, the saturation limit for the controller is set by the continuous torque value. The controller's performance is improved if it is capable of using higher voltage values even for a short period of time. Thus, if the saturation limit is based on the peak torque value to produce high voltage momentarily, then it would improve the controller's performance.

In the Simulink simulation, there are 3 main saturation limits. There is the voltage limit, motor current rate limit, and speed limit. The voltage limit is placed after the controller and protects our system. The motor current rate limit is placed after the voltage output from

the controller is transformed to current. The motor current rate is based on the datasheet electrical time constant of 1ms to replicate how the electrical components in real life cannot have infinite current rate. The speed limit is placed after the plant transfer function and before the feedback to represent the maximum motor speed of 552 rad/sec. With these saturation limits in the Simulink simulation, the step input amplitude was varied to see the impact it has on the time response. The output saturates when the voltage after the controller reaches its limits. Figure 21 shows step input from 0.29 to 0.31 radians and the voltage begins to saturate from step input of 0.3 radians. This is shown through the initial slope for step input of 0.3 and 0.31 radians as they collide. This indicates that the maximum change in angular displacement with respect to time has been reached. The saturation in voltage does not affect the steady state but it does affect the response time. Thus, as the step input amplitude increases greater than 0.3 radians, the intial slope of step response would be the same. Experimentally on LabVIEW, the saturation occurs on 1.08 radians as shown in figure 22. The experimental saturation input amplitude and the simulated saturation input amplitude vary from each other. This may be due to the simulink not considering the power supply response time limit. The power supply data sheet indicates response time of 30ms and this would impact the peak response time to be larger than the Simulink output values.

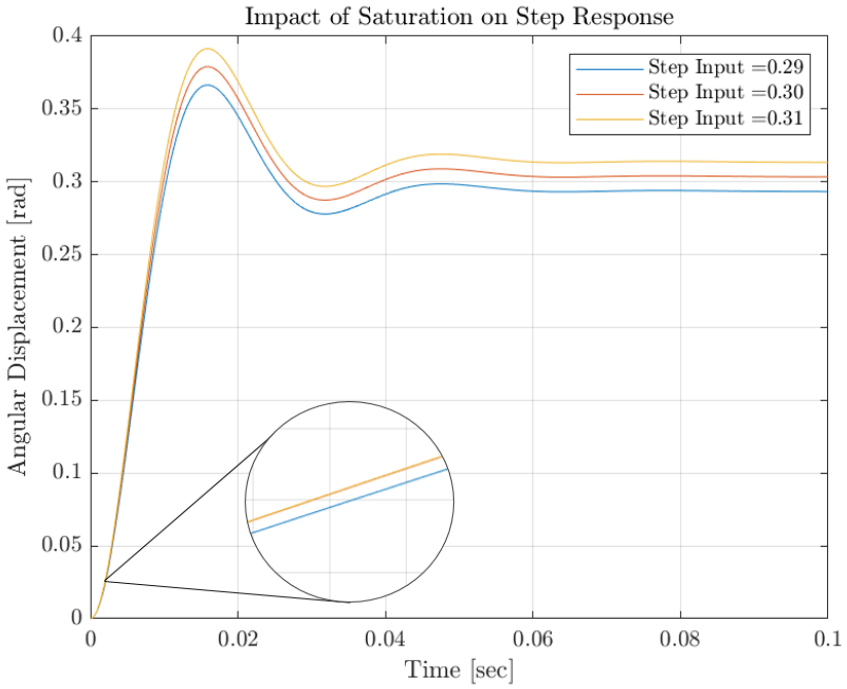

Figure 21: Impact of saturation limit on step response in Simulink. The amplitude of step response was varied from 0.29 to 0.31 radians. The voltage after the controller begins to saturate from 0.30 radians. Thus, the initial slope of the angular displacement for 0.3 and 0.31 radians are same.

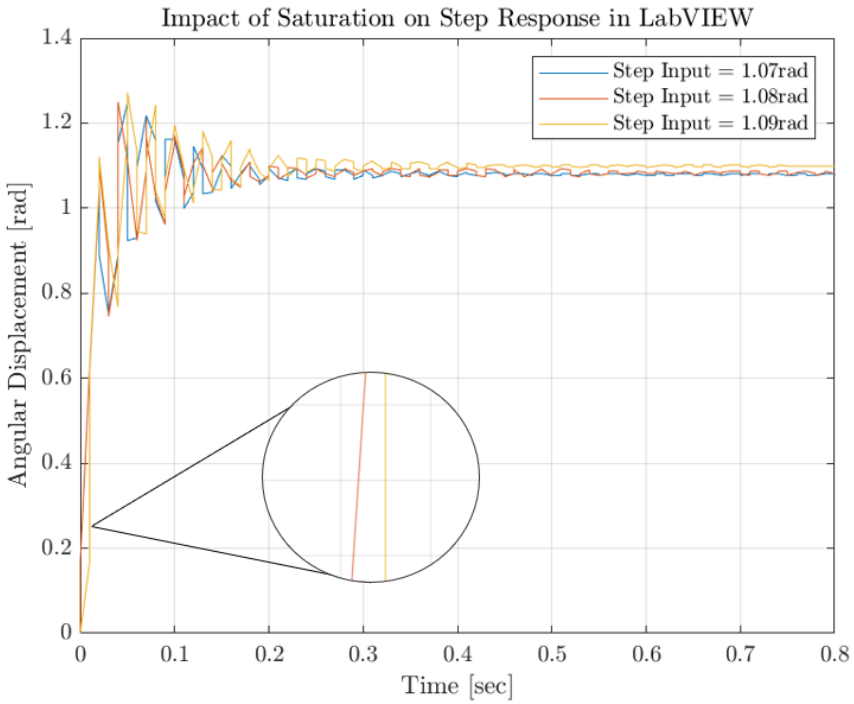

Figure 22: Impact of saturation limit on step response in LabVIEW. At 1.08 radian amplitude of step response, the system saturated and step input of greater than1.08 radians have the same initial .

(d) Coulomb Friction (12)

Two sources of Coulomb Friction in the given D.C motor system are :

1. Sliding between motor shaft and motor shaft bearings.
2. Sliding between contact brushes and split rings.

Coulomb friction has two forms, static and dynamic. Experiments to determine both of them were suggested in Section 1b. The static friction can be determined by gradually increasing the voltage input to the servo-amplifier. When the motor just begins to rotate continuously, the motor torque $I_M K_T$ under this condition gives the static friction torque. To measure the dynamic friction torque, the motor is run at multiple speeds and the correspond motor currents are measured. The relation $T_m = I_M K_T = T_{f,d} + B\omega$ can then be used to solve simultaneously for the dynamic friction torque $T_{f,d}$ and viscous damping constant B.
Figure 23 highlights the addition of the friction torque T_f and the no load voltage V_{nl} required to provide the necessary no load current I_{nl}. T_f is modelled as a disturbance. Since it reduces the effective motor torque available at the motor shaft, it is *subtracted* from the motor torque. V_{nl} is a feed-forward compensation that helps mitigate the negative feedback of friction.
Figure 24 shows the varying time response of the system as T_f is raised. The system is able to achieve the command motor shaft velocity of 1 rad/s up till a (loading) T_f which is

33.5 times the nominal value. Raising T_f further causes it to exceed the maximum available motor torque and the shaft remains at rest. We must keep in mind that the saturation block *Voltage Limit* shown in Figure 23 restricts the voltage input to the amplifier to 3.03 V so as to not saturate the power supply. This means that the maximum gain that can be provided to V_{nl} is $3.03/0.0902 \sim 33.5$. This implies that T_f cannot be increased more than 33.5 times its nominal value. Therefore, increasing the T_f 33.6 times saturates the voltage to supply to the servo-amplifier at 3.03 V. As a result, the motor cannot provide the necessary torque to overcome the high friction load.

In general, if the friction is high, from a control systems design stand-point we can increase V_{nl} so as to raise the current I_{nl} provided for feed-forward compensation. However, this moves the operating point of the D.C motor closer to saturation (of the power supply, motor current or servo-amplifier) and reduces the margin available for fault-free functioning of the motor.

Appendix A: Source Code

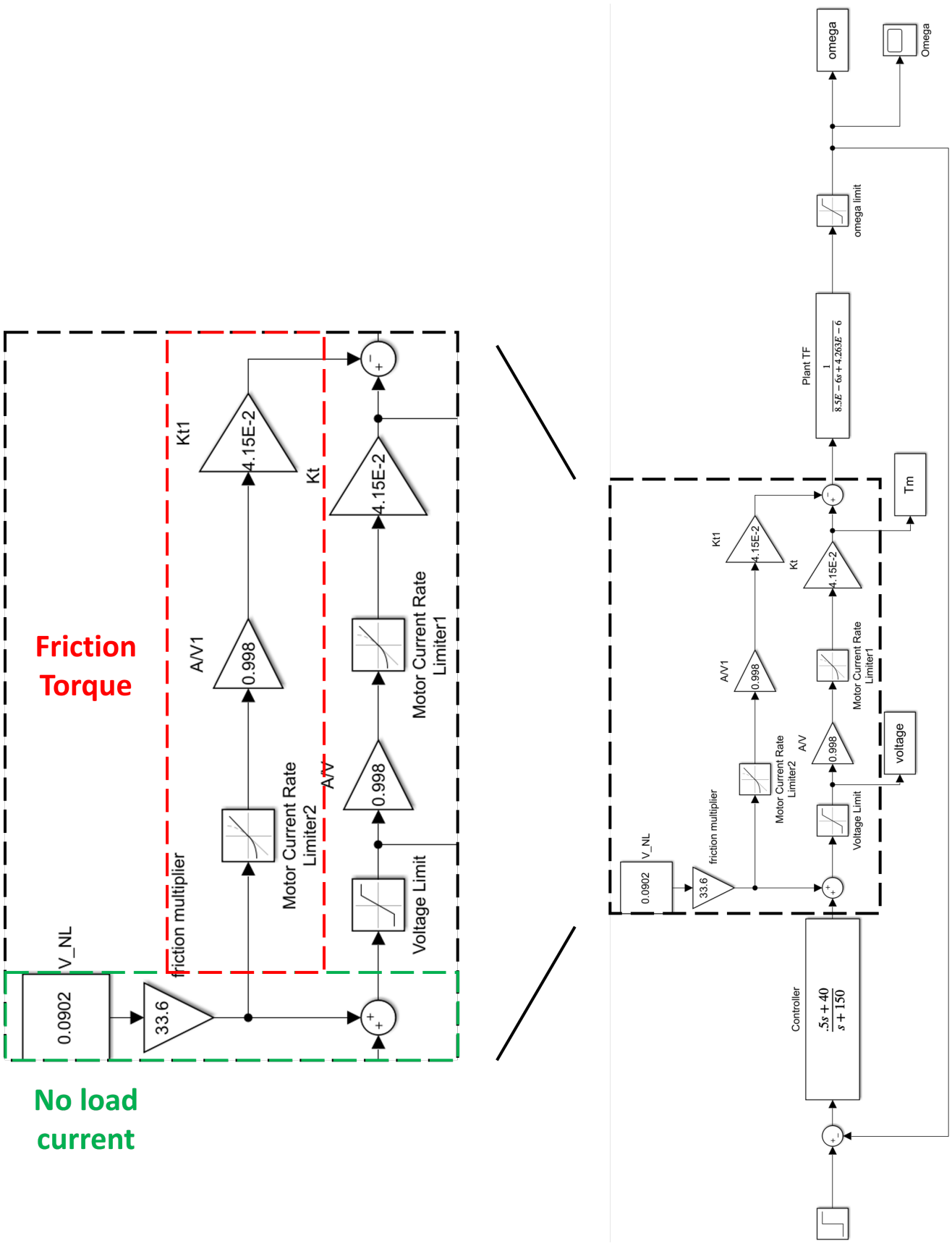

Figure 23: Incorporating friction model in the Simulink D.C motor model

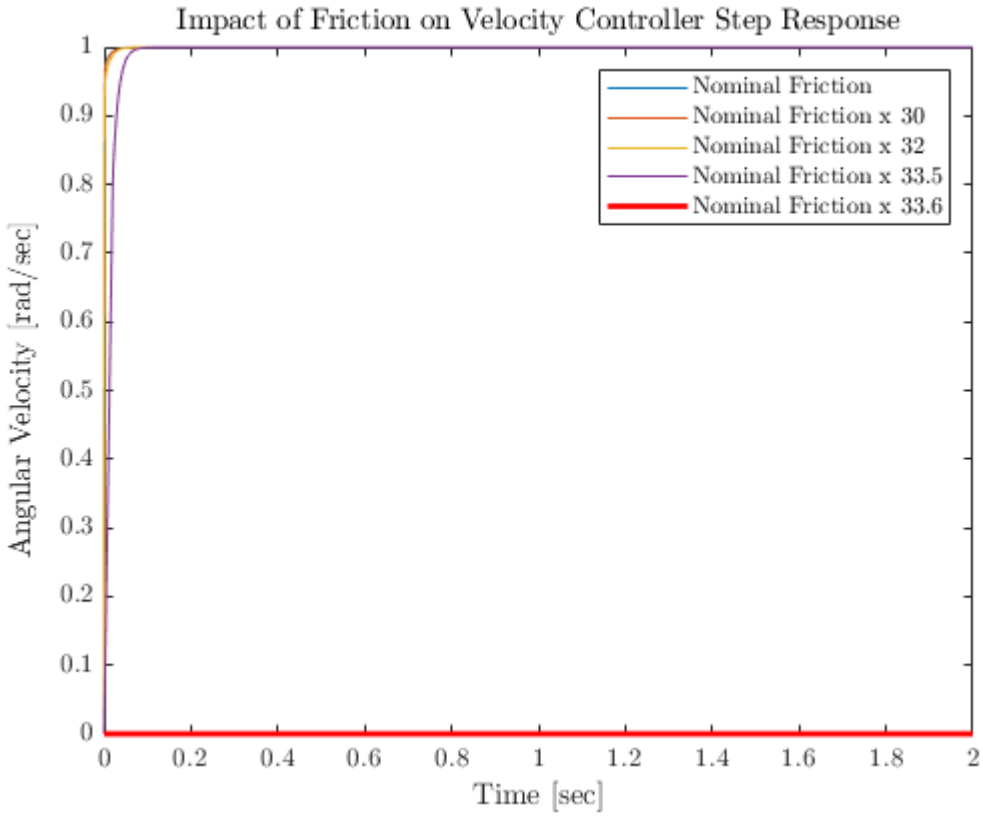

Figure 24: Effect of Coulomb friction on D.C motor controller performance