ASEN 2803 Dynamics & Controls Lab

Lab 3: Rotary Arm Control – Lab Worksheet

Spring 2025

TEAM INFORMATION

Group Number
Lab 3 Group 2-01

Group Members
Brandon Luk
Ava Rocker
Matthew Buccio
Ritik Sarraf

INSTRUCTIONS

- Before proceeding, make sure that every member of your group has read the Lab 3 Assignment Document, including the Appendix section.
- Please read each question on the Lab Worksheet carefully and then fill in the boxes with your answers.
- Note that the size of the answer boxes is not intended to limit your response. Feel free to make the boxes larger if necessary.
- When attaching figures, please make sure to include plot titles, axes labels (including units), and a legend if multiple plots are shown in the same figure.
- When complete, save your Lab Worksheet as a **PDF document** and submit the completed PDF via Gradescope.
- ***Please remember to assign pages to your submission in Gradescope***.

THEORY & SIMULATION

Question 1.1

Review the derivations provided in the appendices and derive Eq. 17 and 18 beginning with Eq. 13.
Hint: First, solve Eq. 13 for V_{IN}. Substitute the resulting expression into Eq. 16. Finally, arrange like terms and divide through to get the closed loop transfer function.

Insert derivation or picture of derivation

Deriving 17 with eq 13 & 16

$$\frac{\theta_L}{V_{in}} = \frac{k_g k_m/(JR_m)}{S(S+k_g^2 k_m^2/(JR_m))} \quad (13)$$

$$V_{in} = \frac{\theta_L(s)(S+k_g^2 k_m^2/(JR_m))}{k_g k_m/(JR_m)}$$

(16) $V_{in} = V_{PD} = k_{p\theta}(\theta_D - \theta_L) - s k_{D\theta}\theta_L$ sub in result...

$$\Rightarrow \frac{\theta_L(s)(S+k_g^2 k_m^2/(JR_m))}{k_g k_m/(JR_m)} = k_{p\theta}(\theta_D - \theta_L) - s k_{D\theta}\theta_L$$

$$\Rightarrow \frac{\theta_L(s)(S+k_g^2 k_m^2/(JR_m))}{k_g k_m/(JR_m)} + k_{p\theta}\theta_L + k_{D\theta} s \theta_L = k_{p\theta}\theta_D$$

$$\Rightarrow \theta_L\left(\frac{(s)(S+k_g^2 k_m^2/(JR_m))}{k_g k_m/(JR_m)} + k_{p\theta} + k_{D\theta} S\right) = K_{p\theta}\theta_D$$

$$\Rightarrow \frac{\theta_L}{\theta_D} = \frac{k_{p\theta}}{\frac{(s)(S+k_g^2 k_m^2/(JR_m))}{k_g k_m/(JR_m)} + k_{p\theta} + k_{D\theta} S}$$

$$\Rightarrow \frac{\theta_L}{\theta_D} = \frac{k_{p\theta} k_g k_m/(JR_m)}{S^2 + S\left(k_g^2 k_m^2/(JR_m) + K_{D\theta} k_g k_m/(JR_m)\right) + k_{p\theta} k_g k_m/(JR_m)}$$

✓

Going from 17 to 18

$$(17)\quad \frac{\theta_L}{\theta_D} = \frac{k_{p\theta} k_g k_m/(J R_m)}{s^2 + s\left(k_g^2 k_m^2/(J R_m) + K_{D\theta} k_g k_m/(J R_m)\right) + k_{p\theta} k_g k_m/(J R_m)}$$

d = coefficients

$$d_0 = \left(\frac{k_{p\theta} \cdot k_g \cdot k_m}{J R_m}\right), \quad d_1 = \left(\frac{k_g^2 \cdot k_m^2}{J R_m}\right) + \left(\frac{K_{D\theta} \cdot k_g \cdot k_m}{J R_m}\right), \quad d_2 = 1$$

$$\Rightarrow \frac{\theta_L}{\theta_D} = \frac{\text{gains}}{d_2 s^2 + d_1 s + d_0} \Rightarrow \text{in Standard form} \ldots \quad s^2 + 2\zeta\omega_n s + \omega_n^2$$

$$\Rightarrow \boxed{\omega_n^2 = d_0 = \frac{k_{p\theta} k_g k_m}{J R_m} \quad (18) \checkmark}$$

$$\sqrt{\omega_n^2} = \omega_n = \sqrt{\frac{k_{p\theta} k_g k_m}{J R_m}}$$

$$\text{Now. } 2\zeta\omega_n = d_1 \Rightarrow \zeta = \frac{d_1}{2\omega_n} = \frac{\dfrac{k_g^2 k_m^2 + K_{D\theta} k_g k_m}{J \cancel{R}_m}}{2\sqrt{\dfrac{k_{p\theta} k_g k_m}{J \cancel{R}_m}}}$$

$$\boxed{\zeta = \frac{k_g^2 k_m^2 + K_{D\theta} k_g k_m}{2\sqrt{k_{p\theta} k_g k_m J R_m}} \quad (18) \checkmark}$$

Next, develop a MATLAB simulation for the closed loop behavior of the rigid arm (Eq. 17). Use the physical and electrical parameters given in the spreadsheet to determine the parameters for the equations of motion derived in Question 1.1. The following sample code can help you get started:

```
%% Closed Loop System
num = n1;
den = [d2 d1 d0];
sysTF = tf(num,den);

%% Step Response
[x,t] = step(sysTF);
```

Note 1: If you want to simulate a more complicated input, explore the `lsim` command in MATLAB to find the theoretical response of your system to a specified input $u(t)$. This is useful when performing comparisons with experimental data, where the reference values and time are also recorded in the data file.

Note 2: You will need to have the “Control System Toolbox” installed in MATLAB for certain functions to work.

Question 1.2

Use the simulation developed in the previous question to investigate the behavior of the step response of the Rigid Arm system for the gain values provided below. Explore the effects of increasing and decreasing the proportional and derivative gains. Set the amplitude of the step response to be **0.5 rad**.

Gains	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6
K1 – Proportional	10	20	5	10	10	10
K3 – Derivative	0	0	0	1	-1	-0.5

Explain the differences and similarities you see between the different sets of gains:

In set 1, with a proportional gain(Kpt) and Kdt of 10 and 0, the rigid arm experiences one complete oscillation and aligns perfectly with the imaginary line with little to no damping. In set 2, increasing the gain to 20 while keeping the Kdt the same led to an increase in oscillation speed raising amplitude of disturbance, along with more overshooting observed before settling with the imaginary line. The change in hub angle with respect to time increases as Kpt increases. In set 3, Kpt reduced to 5 and Kdt set to 0, there was a much smaller disturbance amplitude, little undershooting observed, and little ringing. Introducing a derivative gain (Kdt) of 1 while keeping Kpt 10, improved the system's response. It reduced the amplitude, reduced oscillation speed and eliminated any overshoot and dampened immediately. This led to a decrease in settling time compared to only using a proportional gain in (Set 1). In set 5, when introducing a negative derivative gain with a Kp value of 10, the system saw a significant increase in disturbance amplitude, significant overshooting (to and fro) motion, pronounced ringing, and a much longer settling time. Using a slightly less negative derivative gain in Set 6 with Kdt of -0.5 produced similar effects, though to a lesser extent. We can deduce that positive derivative gains help dampen, reduce the change in hub angle with respect to time the response by reducing overshoot and ringing, while negative derivative gains amplify instability and increase settling time. One thing that I felt and wanted to portray is that increasing the positive derivative gain at higher magnitude led us to analyze that it causes under-damping, as we could see the rotary arm stopped before the imaginary line and could not align with it. For proportional control, higher gains increase system oscillations and disturbance amplitude, whereas lower gains reduce both.

EXPERIMENT

Question 2.1 – Exploring the Hardware

Using the rigid arm **hardware**, input the gains given in Question 1.2. Set the "Reference Amplitude" to **0 rad**. This will make the system want to stay at **0 rad**.

Perturb the system (ie. give the rigid arm a little push or move the arm about 45 deg to one side). What do you notice? Does it feel different for different gains? Explain what differences you notice/feel? Does this make sense?

When pushing the arm 45 degrees with the given constraints, there is resistance moving away from the control center. When pushed past 60 degrees, you can feel even more resistance. The rigid arm is statically stable so this behavior makes sense. The voltage motor should also increase output in proportion to the changing angle. Once further than 90 degrees away from center, the motor will stop sending voltage as a safety precaution.

Both Negative Gains: Very little resistance, the system is unstable and does not try to settle anywhere. Rotates almost freely if it wasn't stopped.

K1 <1, K3 >1: Arm rotates and stays in one place, there is no set center but it is unstable still.

Ideal Case of K1 >> 1, K3 = 1: More resistance, settles on the center very quickly with no overshoot.

Question 2.2 – Conceptual Understanding

Using the rigid arm **hardware**, explore the effects of increasing and decreasing the proportional and derivative gains with the gain values provided in Question 1.2. Make sure to set the "Reference Amplitude" to **0.5 rad**. Compare the behavior of the hardware with **your MATLAB simulation** and with the **provided MATLAB/Simulink simulation**.

2.2 a) What does a higher/lower ***proportional*** gain do to the system? What are the similarities and differences between the hardware, your MATLAB simulation, and the Simulink simulation?

Proportional gain affects the movement of the rotary arm. Higher the gain; quicker the rotary arm movement meaning it covers more hub angle in less time. Secondly, it has an increased over-dampening effect with higher proportional gain. This implies that it takes more time to settle in and with increased gain it is aligned with the imaginary line at greater precision. On the other hand, with lower proportional gain, it is quite the opposite effect where the rotary arm moves slowly, has less of over-dampening effect and aligns with imaginary line at less precision. These effects are observed while keeping derivative gain constant. We sort of categorized the precision using different methodologies while we modeled the rotary arm. The Physical Rotary arm at the lab tried to help us feel the actual movement, change in hub angle, when arm snaps and how faster the speeds of rotation can be felt with variable proportional and derivative gain. These observations were made in the class with the use of an actual rotary arm while accounting for all the external factors (frictional loss, temperature, pressure, voltage fluctuations) that affect its movement and desired results. While the MATLAB Simulation is good for theoretical observations as it portrays change in hub angle vs time and motor V vs time graph and to analyze how the change in variables tend to affect the movement of the rotary arm while it does not account for any of external factors. Simulink simulation and its use was really fascinating as it makes us use it like real ones at the lab. It can help us analyze the use of variable proportional and derivative gain in much deeper ways and while being able to visualize the movement of the bar and track its movement along the reference line and dampening at greater precision while it is difficult to observe in a physical rotary arm.

2.2 b) What does a higher/lower ***derivative*** gain do to the system? What are the similarities and differences between the hardware, your MATLAB simulation, and the Simulink simulation?

Derivative gain is how much voltage is applied to the system based on how fast the arm is actually moving. Changing the variable was very similar for both the hardware and the software. A lower Kd would cause the system to oscillate more / not settle. While increasing the Kd would increase the settling time and have virtually no overshooting. The range that we used for Kd was from -1 to 3. After 3 the Kd did not seem to change the system to any major effect. Similarly with -1, anymore below that did not seem to affect the system more than it already did.

Seen on the MatLab simulation, as you increase the magnitude of positive derivative gain, the arm takes longer to approach the reference line and stays at the reference line for less time than it would at a positive derivative gain closer to zero. MatLab clearly shows the arm reaching the reference line, while the Simulink simulation does not give a clear definition of if the arm reaches the reference line or not. This is because Simulink is more precise, and shows much more detailed oscillations close to the reference line. Furthermore, the movement of the arm as seen on the Simulink simulation gets smoother and slower as the derivative gain increases in magnitude.

As with increasing the magnitude of negative derivative gain, it approaches the reference line quickly while it also overshoots quite frequently allowing it more time to align with the reference line. But closer to 0 value for the negative derivative gain aligns the bar with the reference line quite well. Higher negative gain takes forever to stop the bar and continues the oscillations with increased dampening effect.

2.2 c) How can the ***overshoot*** be increased/decreased? What are the similarities and differences between the hardware, your MATLAB simulation, and the Simulink simulation?

Overshoot will increase by increasing the magnitude of the derivative gain in either the negative or positive direction. On the other hand, overshoot will decrease as the magnitude of the derivative gain gets closer to zero (decreases) since the arm will have better accuracy. Furthermore, as the magnitude of the proportional gain decreases, the rigid arm movement becomes slower and the system becomes slightly less accurate. But, the decreased proportional gain value also decreases the amount of overshoot giving a sort of trade off between overshoot and performance as seen in Simulink simulation. The hardware shows very similar findings to the Simulink simulation, where both give accurate representations about the arms movement and angle compared to the different gains. The MatLab Simulation is much less accurate than Simulink or the hardware, but still confirms our conclusions about how overshoot can be decreased/increased.

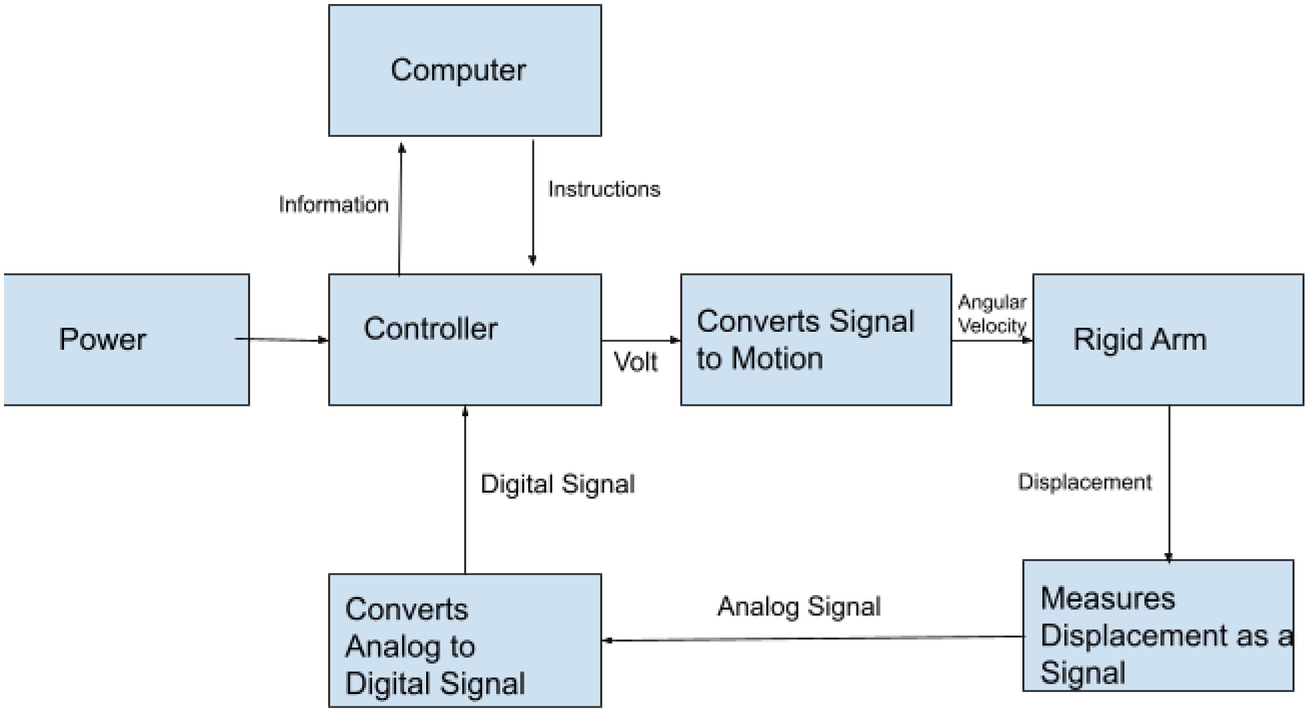
2.2 d) How can the ***settling time*** be increased/decreased? What are the similarities and differences between the hardware, your MATLAB simulation, and the Simulink simulation?

Differing from decreasing the amount of overshoot, in order to decrease the settling time, you would have to increase the proportional and decrease the derivative gain. Increasing the proportional gain of the system increases the system's response speed and allows the arm to reach its target position in less time. Furthermore, decreasing the system's derivative gain adds damping to the system which decreases the amount of settling time. Although, there are trade offs for both of these gains. Increasing the proportional gain too much can lead to overshoot while too low of a derivative gain will create oscillations. These effects (both positive and negative) can be seen very clearly on the Simulink model and the actual hardware. For the hardware and Simulink, you can physically see the arm reach its destination quicker, have oscillations, or overshoot. The MatLab model gives a graphical model of the movement of the rigid arm and through testing different possibilities for the derivative and proportional gain, the same conclusion can be made.

Question 2.3 – Overall System and Software

2.3 a) Sketch a functional block diagram of the rigid arm system below:

Insert picture of Functional Block Diagram

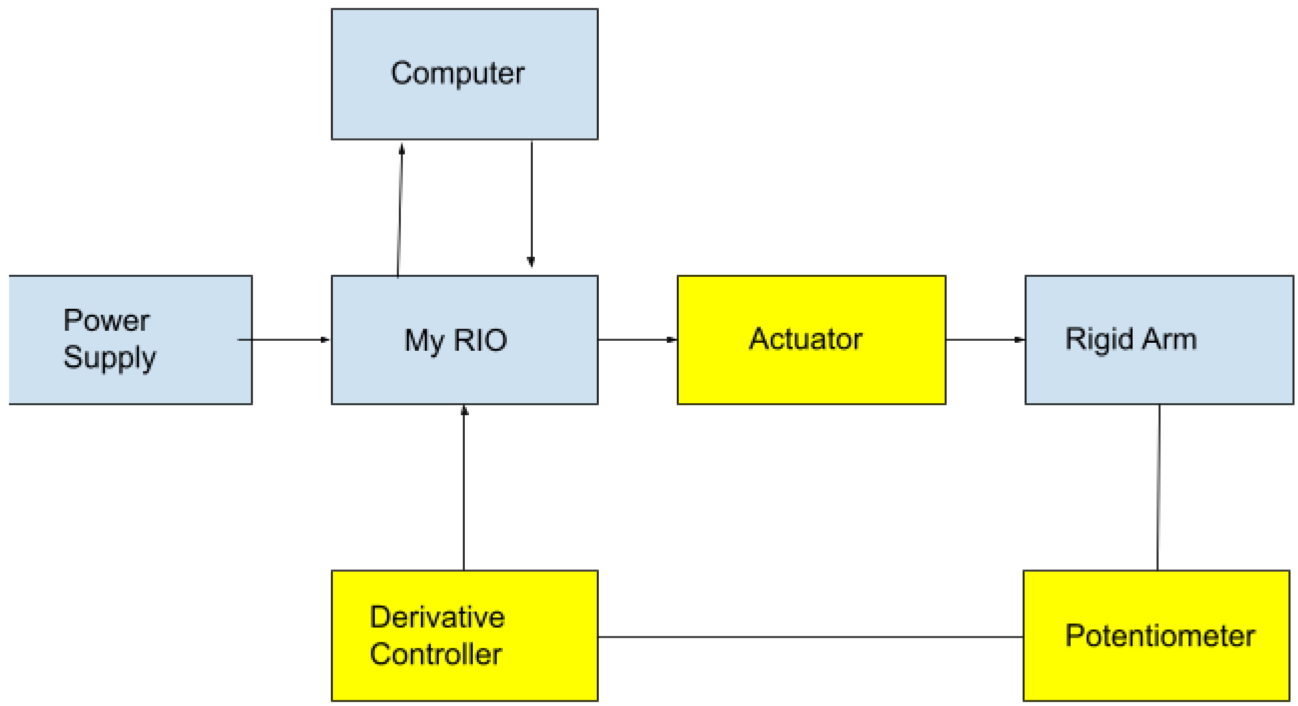


2.3 b) What does each part of the system do (ie. the arm, sensors, myRIO, actuator)? What is the function of each component, and how do they interact with each other?

The arm is used as the indicator of the system allowing for interaction (perturbations) and a visual system response to be recorded. The myRIO microcontroller is the "brains" of the system observing the location of the arm via the potentiometer and sending the calculated voltage to the motor (actuator) to move the arm correctly. The motor is a simple DC unit that, after a gear box, moves the location of the arm based on the incoming voltage and duration. There is also a potentiometer that is used to "see" the current angle of the arm.

2.3 c) Sketch the control block diagram below, and highlight the transfer function(s):

Insert picture of Control Block Diagram



2.3 d) Describe how the control system works to achieve the desired arm position. What is the function of each block, what are their inputs and outputs, and how do they interact with each other?

Power is output from the power supply which powers the system. This power goes straight into the My RIO which feeds the computer with info. The Computer then makes plots, and transfers information to the My RIO, which takes those instructions from the computer, and makes them as an input for the actuator. The actuator then tells this input, which is in voltage and generates an angular velocity for the arm. The arm then moves with the generated angular velocity, then generates a displacement as it moves around. This displacement is then sent to the potentiometer, where an analog signal is output to the derivative controller. Where it is then converted to a digital signal to be received by the My RIO. The MyRIO takes this signal and sends it to the computer, which processes it and returns a control output. This output is then sent to the actuator. The yellow highlighted blocks represent the transfer function, which describe how signals are transformed as they pass through different parts of the system. The actuator receives a voltage and converts it into motion, making the rigid arm move. To an observer, it looks like the MyRIO directly causes the arm to move. The potentiometer and derivative control are also transfer functions: they convert the arm's displacement into analog and then digital signals, which are sent back to the MyRIO. These components essentially translate the arm's physical movement into data the system can interpret and use for feedback.

Question 2.4 – Design Controls and Behavior Response

Now you are going to design your own control gains such that your system meets a set of requirements. Your controlled rigid arm should:

- Have **less than 20% overshoot**
- Achieve a **5% settling time in less than 1 second**

Your reference step signal should have an **amplitude of 0.5 rad** and a **period of 10 s**.

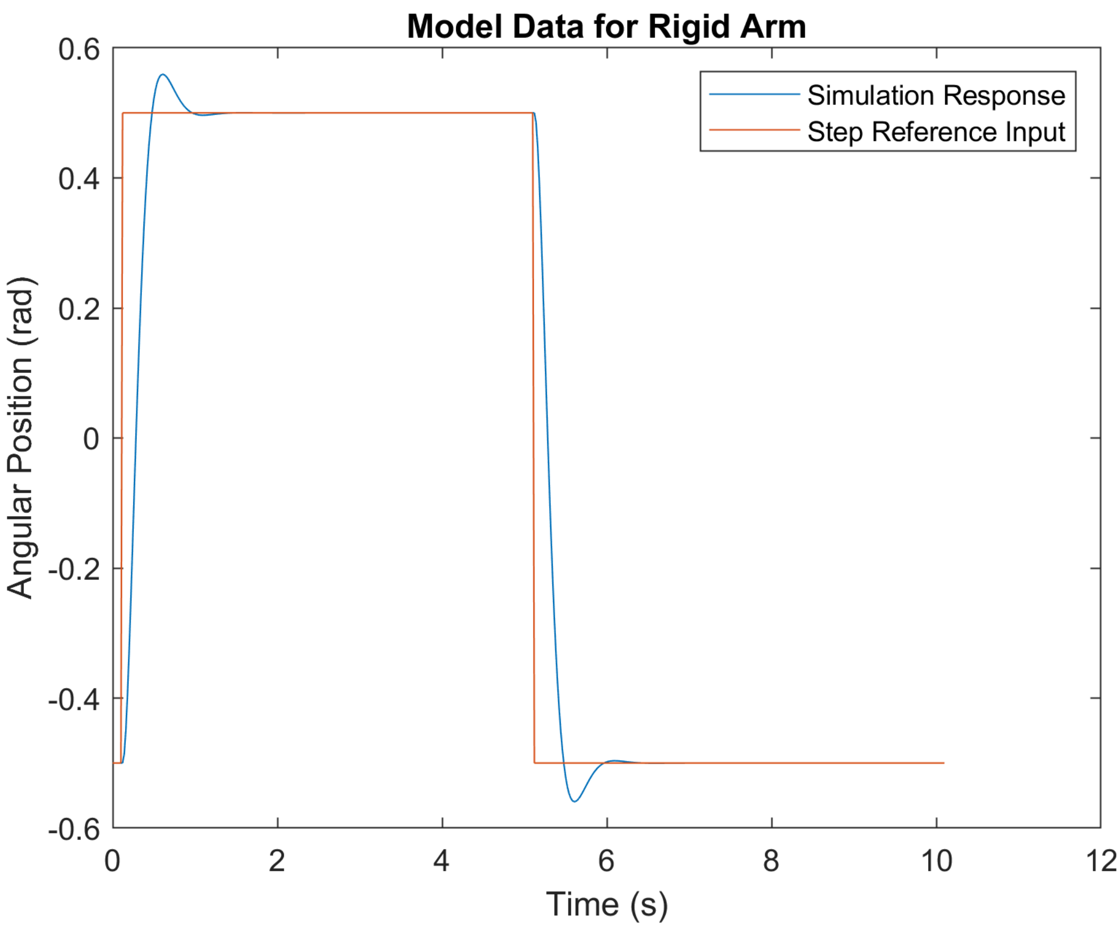
Make sure to show at least 1 cycle in your plots. Make sure to plot the reference step input in all of your plots.

2.4 a) What proportional and derivative gains did you arrive at?

Gains	Value
K1 – Proportional	19
K3 – Derivative	0.5

2.4 b) Input these gains into **<u>your MATLAB simulation</u>**. Do you meet the requirements? Check your time response plot (by inspection) and include it below. If you don't meet the requirements, explain why.

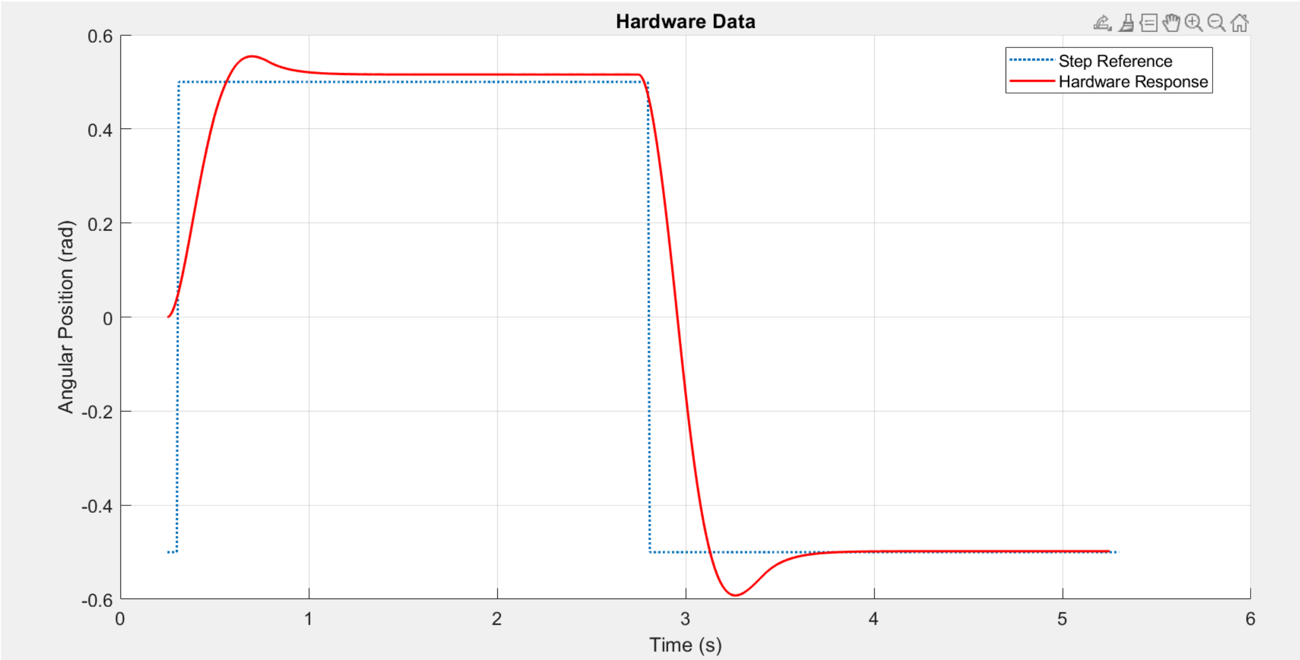
Insert picture of your time response plot

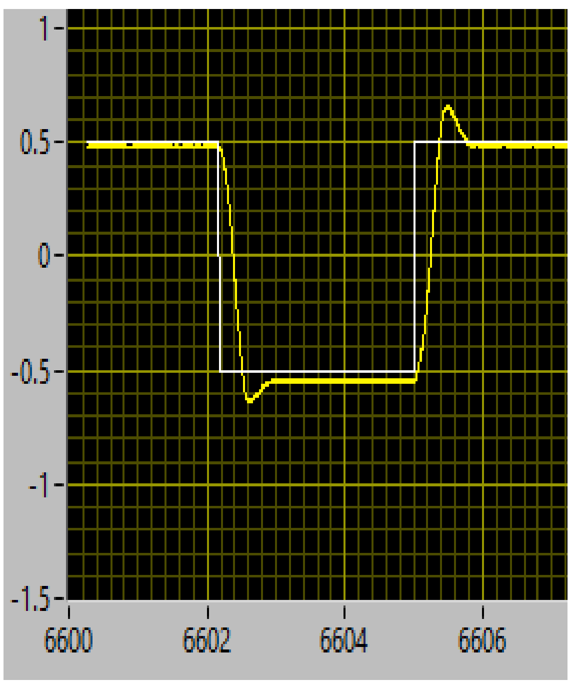


Yes, we did meet the requirements as the MATLAB simulation has the settling time of 0.1s and overshoot of 11.86%. The MATLAB modeling simulation behaves as it is reflected on the graphs.

2.4 c) Now, input these gains into the rigid arm **hardware VI**. Record data and then plot that data in MATLAB. Include the figure of your time response below. Do you meet the requirements in the hardware? If you don't, explain why.

Insert picture of your time response plot

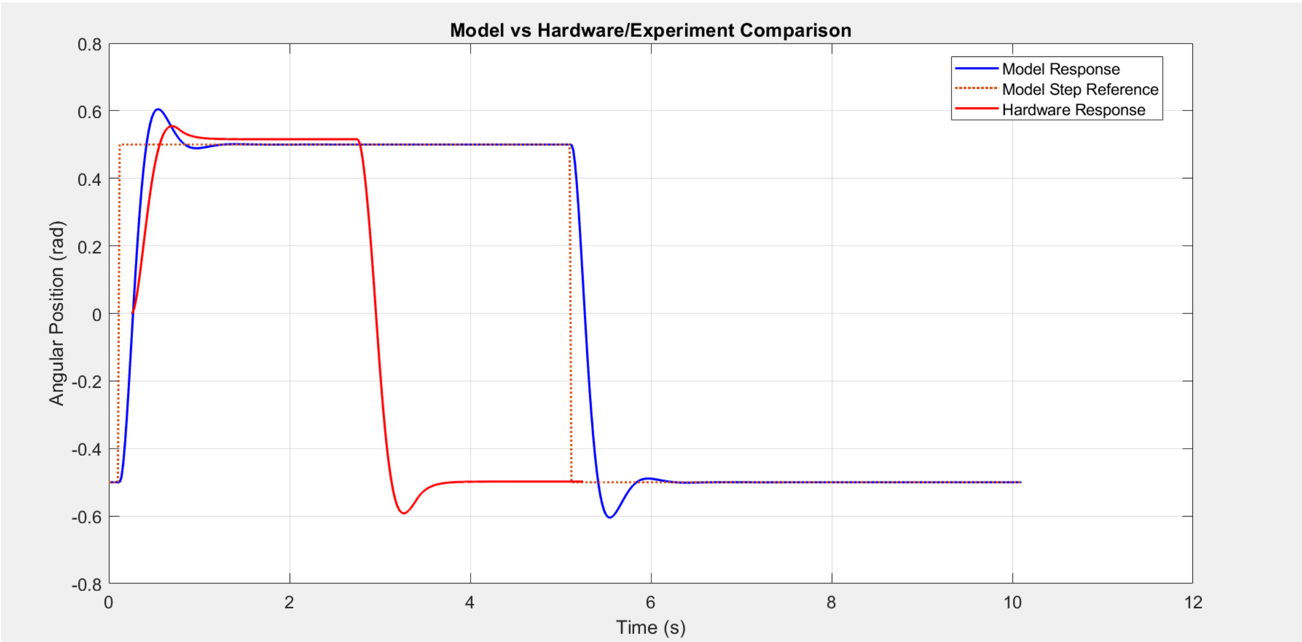




Yes, when comparing the hardware plots with MATLAB they are about the same with similar amplitude of oscillations. Getting technical, the % overshoot was 11.86% and the settling time was about .6 seconds so these values do fall within the given parameters.

2.4 d) Now, compare the response from **<u>your MATLAB model</u>** with the response from the **<u>hardware</u>**. Plot both time responses together and explain the differences between the two. What do you think is causing the difference?

Insert picture of your time response plot with both time responses plotted on the same figure



While comparing MATLAB with hardware responses from graphs, it seems like the modeled plots have higher amplitude curves while the hardware graphs have smoother and flatter curves that can be vividly observed as the oscillation changes its direction. This might be because of the frictional effect between the rotating materials in context to real world applications while the MATLAB graph does not account for any frictional force. Also, both of them overshoot the reference step function. Another difference between the hardware and our model is that the hardware doesn't return to the reference amplitude, while the model always returns exactly to the reference amplitude.

2.4 e) How can you modify your gains to meet the requirements in the hardware? Try these new gains in the hardware and show if they meet the requirements by plotting the output of the VI in MATLAB.

Insert picture of your time response plot

Our simulation values did meet the required restrictions with the hardware as well.

2.4 f) Beware of the 10V limit on V_{IN}. How does your model and experimental data differ if the control output reaches this value? Look at the Control Voltage plot in the hardware VI.

When 10 or more volts is sent through the system by the control output, the controller will automatically cut off the hardware. This is evident on the "Motor V" plots, there is a physical limitation on the motor that caps the voltage at ± 10 volts. This shows that our mode is capable of higher voltages, but the hardware is not. This also indicates that the hardware cannot respond as quickly as the model predicts. Because the hardware moves more slowly, it experiences less ringing compared to the model. This is because the model does not have the physical constraints like voltage, whereas the hardware does.

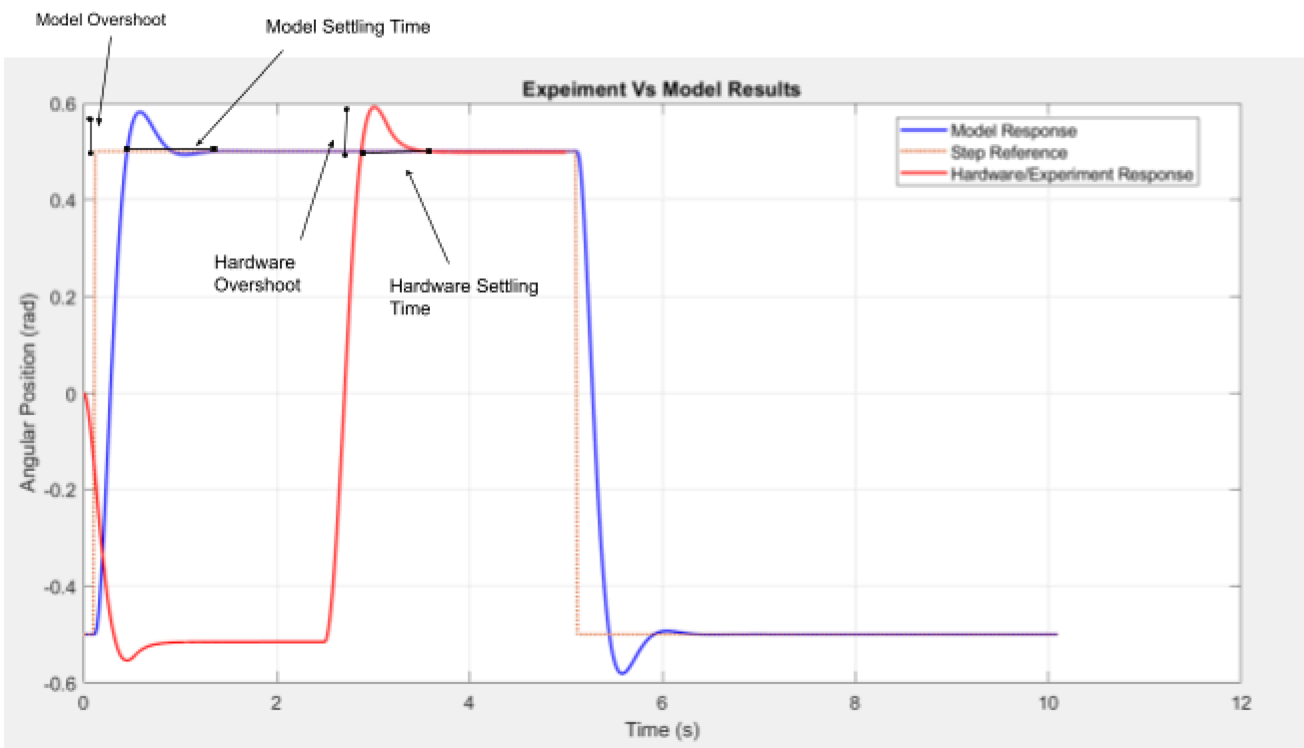
RESULTS AND ANALYSIS

Question 3.1 – Results

NOTE: It has been observed that the physical units may exhibit different behavior when moving clockwise vs. counterclockwise. You only need to analyze a single transition. This means you are free to pick the direction that reasonably matches your model and ignore the opposing transition.

For the rigid arm, plot and compare the experimental results with the model results from the gains you selected to meet the performance objectives. Label the **overshoot** and **5% settling time** on the plot. Attach the result and explanation below:

Insert picture of your time response plot and your explanations



Hardware response had little undershooting from the mean angular position (0.5 rad) after changing the direction whereas model response didn't. It has a similar effect on the damping for the next set of arm rotation and so on. The percentage overshoot is 16.4% for the mode and the settling time is +/- 0.5 seconds. The model had less overall amplitude compared to the hardware. However, the hardware did settle immediately while the model can be seen overshooting once, then settling. This could be due to natural damping affecting the hardware, which our simulation did not originally account for.

Question 3.2 – Analysis

3.2 a) How and why do the experimental results differ from the theory?

The main reason that the experimental results differ from the theory is because the theory assumes certain factors are constant or are not accounted for, whereas the real life simulation must account for certain factors and cannot assume certain factors are constant. For example, one of the biggest assumptions in the theoretical model is that friction is negligible, which is not true in the experimental model. Friction can cause damping or delay effects in the experimental simulation that are not accounted for in the theoretical, causing significant differences in the results. Furthermore, air resistance can also affect the experimental system by increasing damping and possibly decreasing the natural frequency. With significant air resistance one would see less overshoot than expected and a slower return to equilibrium, differing from the theoretical results.
Moreover, due to friction it did not end up having smoother curve while changing the direction because the rotation of arm accounted for frictional and air resistance while the MATLAB simulation did not. There is noise that can be observed on the curve for the experimental results plots and that might be due to vibration while arm rotation.

3.2 b) We assumed negligible natural damping (B) to be zero. Is this a good assumption for this experiment? Why or why not?

This is not a great assumption, when we compared the model to the hardware using the same proportional and derivative gains, the model responded more quickly. The hardware, however, moved more slowly and exhibited less overshoot and ringing than the model predicted, due to the account of natural damping present in the hardware. Simulating natural damping in our model would be rather difficult however as there are many factors that can cause natural damping. So for ease of access and a more 'perfect' simulation we assumed it to be negligible.

3.2 c) How can you determine an approximation for the natural damping in the real system?

Because the system is underdamped, the best way to determine an approximation for the natural damping in the real system would be by using the logarithmic decrement method. This method represents the rate at which the amplitude of a free damped vibration decreases. This method works by simply looking at the height of two peaks in the system and calculating how much smaller the second peak is than the first, this is possible because for an underdamped system the peaks decay exponentially over time. Simply speaking, the faster the system decays, the higher the damping will be, and the difference between the two systems will be the natural damping.

3.2 d) Compare your improved model (the one that includes the effects of natural damping in the real system) to the model without friction and to the real system.

The improved model with natural damping has a lower degree of over- and undershooting effects and aligns with the mean angular position more quickly than without friction and dampening effects. This helps the rotary arm from snapping at very high hub angular velocity. This lower velocity caused less overshoot, shorter settling time, and less ringing. The real-world application of the improved model could be using it in the car as a shock absorber, vibration control, reducing noise, and preventing any damage to the physical structure of the car. The lack of frictional effect could increase the settling time for the rotating arm or shock absorber used in the car.