Creating a Controller for a Designated DC Motor

J. R. Keusch

Abstract – The goal of the project is to build a controller for a DC motor. The DC motor is then used to functionalize an escalator. The controller used was determined based on parameters received when starting the project. A PI (Proportional-Integral) controller was used and implemented onto a block diagram of an electromechanical system.

Index – Bode Plot, Block Diagram, Controller, Linear, MATLAB, Motor, Poles, Root Locus, S-Domain, Simulink, System, Transfer Functions, Zeroes

I: Introduction

Controllers and control systems are an integral part of Linear Systems and Electrical Engineering in general. Controllers a used to output a desired response from a system. One can be used for motor control, temperature control, and other similar functions. A popular form of controller is a proportional-integral-derivative controller, or PID controller. PID's are unique because they are simple to put together yet have more flexibility than other types of controllers. PIDs can also be simplified to make them work more fluidly with a system. PI's are a popular subsection because they are more simplistic, yet can still be used when requiring zero steady state error. For the problem specified, a PI controller was the most suitable. The procedure of how the parameters of the controller will be given throughout the report as it follows the steps of how it was functionalized.

II: Project Specifications

The goal of the project was to create a controller for a DC motor used to power an escalator while using values initially received. The parameters given were:

Parameter	Value
Ra (ohms)	0.008
La (uH)	26
Kt (uNm/A)	590
N1/N2	7.5
Friction Torque (Nm)	0.01
J (kgm^2)	0.0026
B (uNms/rad)	100
Speed Sensor Gain	1
Desired Overshoot (OS%)	8%
Desired Settling Time (ST)	4s

Also, Ke can be assumed to equal 590e-6 because Kt = Ke in SI units.

With these parameters, a block diagram was made to show to the characteristics of the current input-output relationship. The block diagram was injected with a step input of 0.25 for a time of 130s. The block diagram is shown in the appendix and the results are shown in figure 1.

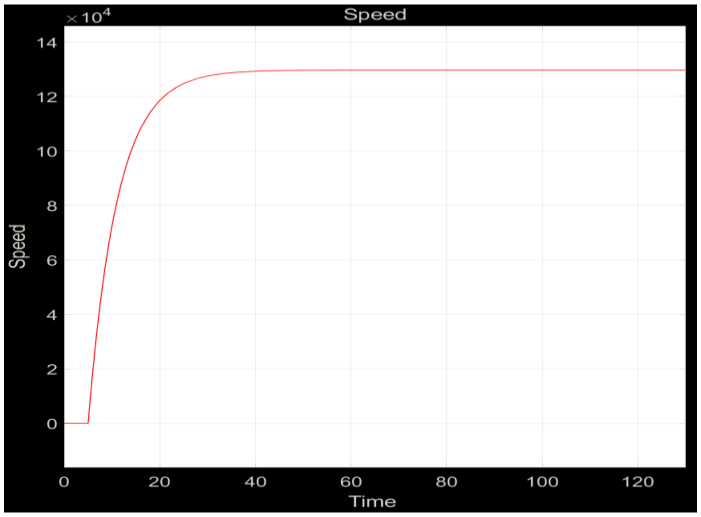


Figure 1: Step Input Response

This block diagram was reduced to a single transfer function that looked like:

$$\frac{Kt * (\frac{N1}{N2})}{(La * J)s^2 + \big((B * La) + (J * Ra)\big)s + B * Ra + \left(\frac{N1}{N2}\right) * Kt * Ke}$$

With this transfer function, the root locus can be modeled once the desired poles are found.

The desired poles can be found using the equations:

$$\sigma_d = \zeta * \omega_n$$

$$j\omega_d = j * \omega_n * \sqrt{1 - \zeta^2}$$

With these values, σ_d is the real part of the pole and $j\omega_d$ is the imaginary part of the pole. To find these values, however, first ζ and ω_n must be found. To find these values, the following equations are used.

$$\zeta = \frac{-\ln\left(\frac{OS\%}{100}\right)}{\sqrt{\pi^2 + \ln\left(\frac{OS\%}{100}\right)^2}}$$

$$\omega_n = \frac{4}{ST * \zeta}$$

Because overshoot percentage and steadying time are both predetermined values, both ζ and ω_n can be found, as well as the desired poles. Using the equations presented, and the values given, the desired poles come out to be:

$$P1, P2 = -1 \pm 1.2438j$$

With these values specified, design of the root locus can begin.

III: Open Loop Characteristics

Before describing the root locus, it is important to understand the function as an open loop and how it behaves. The bode plot is represented in figure 2 and the pole-zero plot is represented in figure 3.

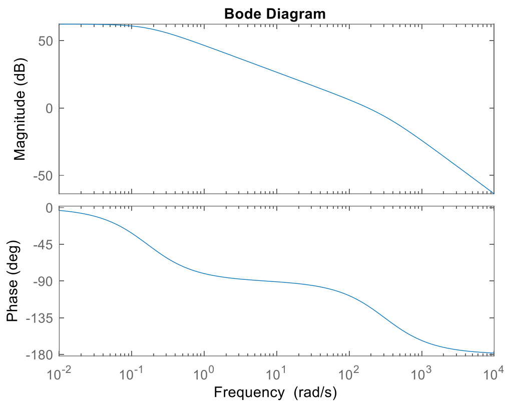


Figure 2: Bode Plot

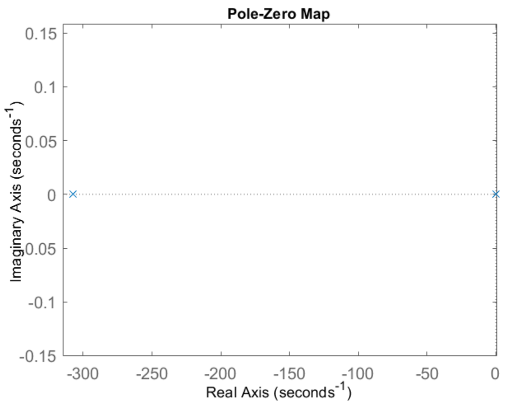


Figure 3: Pole-Zero Plot

The graphs show the change in frequency response with the bode and where the poles from the system's single transfer function are located in the pole-zero plot. The frequency response goes down because all of the poles are on the left side of the Y-axis on the pole-zero plot, and each negative pole adds -90 from the phase bode plot and -20 from the magnitude plot. Since the Bode plot is continuously going down, and we have no zeroes, this indicates the system is stable.

The Pole-Zero plot shows the locations of all the poles and zeros for that specific transfer function. The transfer function from the DC motor contains two poles and no zeros. The poles -307.5667 and -0.164 represent the electrical system and mechanical system respectfully. The electrical system is represented by a larger pole because it is much faster than the mechanical system.

IV: Root Locus and Controller Design

After the desired poles are found, the next step is to create the root locus and to tweak it until it runs through the desired poles. The first step is to multiply the entire transfer function by a step input, which results in the following equation:

$$\frac{Kt*(\frac{N1}{N2})}{(La*J)s^3+((B*La)+(J*Ra))s^2+(B*Ra+\left(\frac{N1}{N2}\right)*Kt*Ke)s}$$

This adds another pole, as s=0 to the root locus. After that is completed, the total angle contribution (TAC) and angle deficiency must be calculated. The formulas for those are:

$$TAC = -(\theta_1+\theta_2+\theta_3)$$

$$AngleDef = TAC + 180^{\circ}$$

These equations are currently unsolvable without $\theta_1, \theta_2,$ and θ_3, so those must be solved for too. To find the values, each angle contribution from the poles must be found. They can be found with the following equations, where P1 = -307.5521 and P2 = -0.164:

$$\theta_1 = \tan^{-1}(\frac{\omega_d}{-\sigma_d - P1})$$

$$\theta_2 = 180° - \tan^{-1}(\frac{\omega_d}{\sigma_d + P2})$$

$$\theta_3 = 180° - \tan^{-1}(\frac{\omega_d}{\sigma_d})$$

Using these formulas, $\theta_1 = 0.2325,\ \theta_2 = 123.9042,$ and $\theta_3 = 128.7980$, TAC = -252.9347, and the angle deficiency is -72.9347. With all these values, the zero needed to help the root locus run through the poles can now be found. The zero is found with the equation:

$$Z = \frac{\omega_d + \sigma_d * \tan(-AngleDef)}{\tan(-AngleDef)}$$

With the zero value, which was z = 1.3818, the root locus will now ideally run through the desired poles. To receive the correct transfer function for the root locus, the current function from earlier must be multiplied by the zero. This results in:

$$\frac{Kt*\left(\frac{N1}{N2}\right)*(S+Z)}{(La*J)s^3+((B*La)+(J*Ra))s^2+(B*Ra+\left(\frac{N1}{N2}\right)*Kt*Ke)s}$$

With this transfer function, the root locus can be found. After correcting the gain, the root locus shown in figure 4 is obtained.

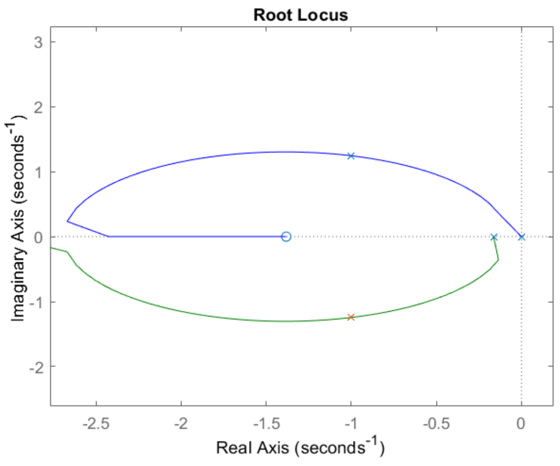


Figure 4: Root Locus

In figure 4, the root locus is shown passing through the desired poles, this means the controller is ready to be designed. First, the PI controller has a specific equation attached to it. The equation is:

$$\frac{s+Z}{s} = K_P + \frac{K_I}{s}$$

Which can be rewritten as:

$$\frac{s+Z}{s} = \frac{s+\frac{K_I}{K_p}}{s}$$

Also, K_P can be found as the gain at the desired poles, which for this root locus is 0.0086. The following equation can be used to solve for K_I:

$$K_I = Z * K_p$$

This equation results in K_I being equal to 0.0119, and a final controller of:

$$PI(s) = 0.0086 + \frac{0.0119}{s}$$

With this equation, the controller has been designed and is moved on to validation.

V: Controller Validation

Since the controller has been fully designed, it now must be tested. First, it must be checked that it can run with the single transfer function and successfully control the system. In figure 5, the results of the controller being connected to the transfer function of the DC motor are shown.

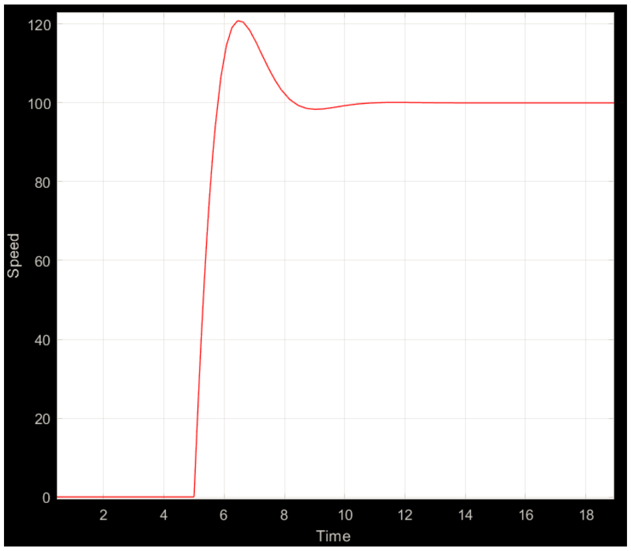


Figure 5: Controller Validation with Single TF

As shown, the controller successfully is able to steady a step input of 100 after around five seconds, with being within 2% of that value within 3-4 seconds. The overshoot percentage is at 20% which is much higher than the desired percentage of 8%. This could be because the controller is untuned or poorly tuned, or because of unexpected frequencies or disturbances.

The next validation is to see how a load torque will impact the signal. In Figure 6, the results of a load torque of 0.05 being added at time 50s is shown.

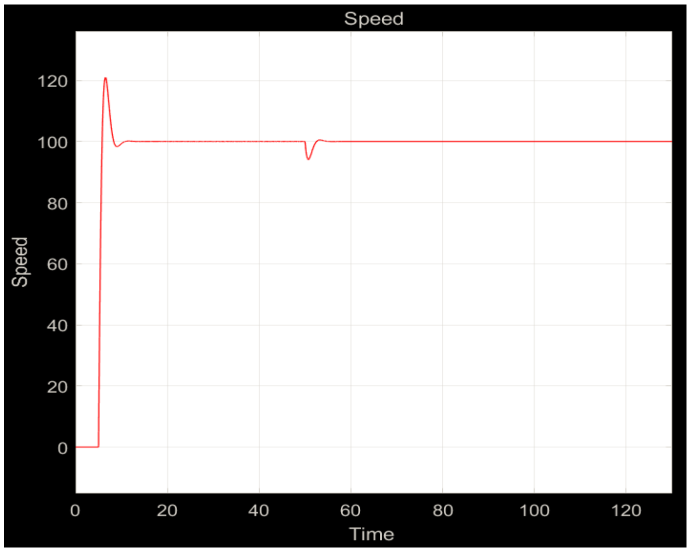


Figure 6: Controller Validation with Load Torque

As shown in the figure, the load torque affects the signal, but the controller brings the value back to the steady state. This helps validate that the controller is doing its job and is successfully controlling the system.

VI: Influence of The Controller

The design of this controller could be influential on many levels given the circumstances. For example, this controller could be more accurate than a specific one in use in terms of load, making it safer for those who use it. It could also be more effective in terms of power, making it cheaper to use and more cost efficient. It could potentially be easier and cost less to repair as well. It could be an easier control to make, resulting in it being more accessible for companies and, for if some reason it was to be used domestically, for consumers. Most of its influence depends on its efficiency in specific fields compared to other controllers in its field, but there is a serious possibility that this controller could have real-world uses if it can beat the status quo of any of these specs currently in the general market.

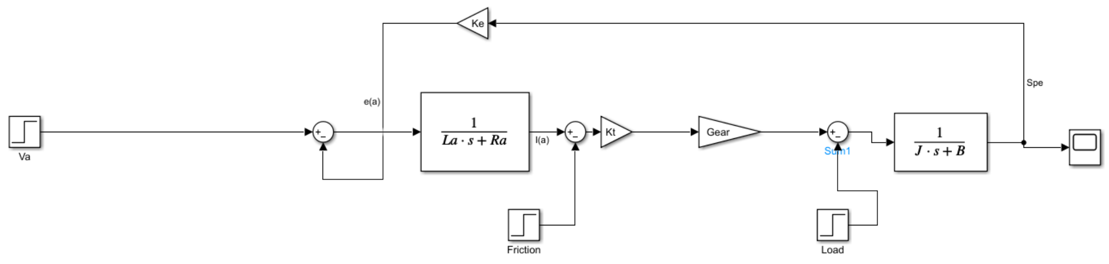
To make improve it further for the general public, it would be important to focus on the points mentioned previously. For example, if this controller was tested to make it the safest it could possibly be, it could rival current controllers and help the general safety of the product. The same could be said for all of the other specs mentioned earlier, as well as many more.

VII: Conclusion

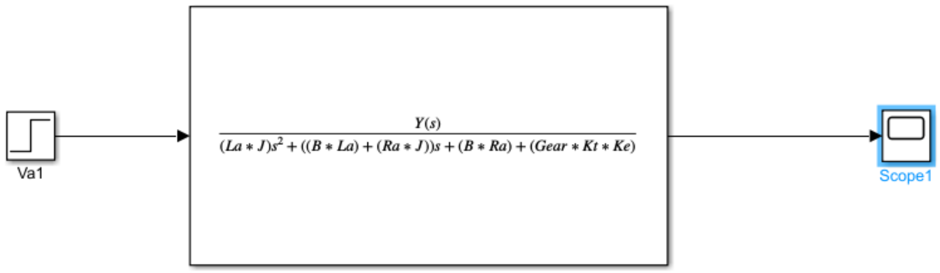
Throughout this project, using MATLAB's, a controller for a DC motor used to power an escalator was designed and tested. To be able to do this, many other steps occurred, such as making and reducing a block diagram for a custom electromechanical system, the a root locus design, and finding the angle contributions and deficiency, as well as the zero that would help the root locus run through the desired poles.

Although the percentage overshoot was not the same as the desired overshoot that was specified, the controller still was able to control the system to stay in the desired value, even when a load was applied.

VII: Appendix



Block Diagram 1


Va1
$\frac{Y(s)}{(La*J)s^2 + ((B*La)+(Ra*J))s + (B*Ra) + (Gear*Kt*Ke)}$
Scope1

2 - Reduced Block Diagram