EEE342 Lab-1 Preliminary Report

Fatih Mehmet Çetin

Department of Electrical and Electronics Engineering, Bilkent University, 06800 Ankara, Turkey

1. Introduction

This preliminary work aims to derive a first-order transfer function for a DC motor by analyzing the relationship between input voltage and angular velocity using MATLAB's Simulink. Empirical data is filtered to reduce noise, allowing for accurate estimation of the time constant (τ) and steady-state gain (K). The obtained transfer function is then compared with the filtered data. Then, on the second part of the lab we explore the fundamental principles of Proportional (P) and Proportional-Integral (PI) controllers, analyzing their effects on system response parameters such as overshoot, settling time, peak time and we compare different type of controllers based on these parameters as well as the steady state error.

2. Laboratory Content
2.1. Transfer Function Approximation

We were provided the time and angular velocitiy data of a DC motor on Moodle. The data was noisy due to the sampling process. We were assigned to filter out the raw angular velocity data on Simulink. The low pass filter I used had the following transfer function **(1)**:

$$H_{LPF}(s) = \frac{1}{0.001s+1} \quad \textbf{(1)}$$

The filtering was done on Simulink. I designed the following diagram:

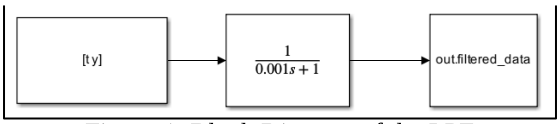

Figure-1: Block Diagram of the LPF

Here you can see the plots for both raw angular velocity data on Moodle and the filtered angular velocity data:

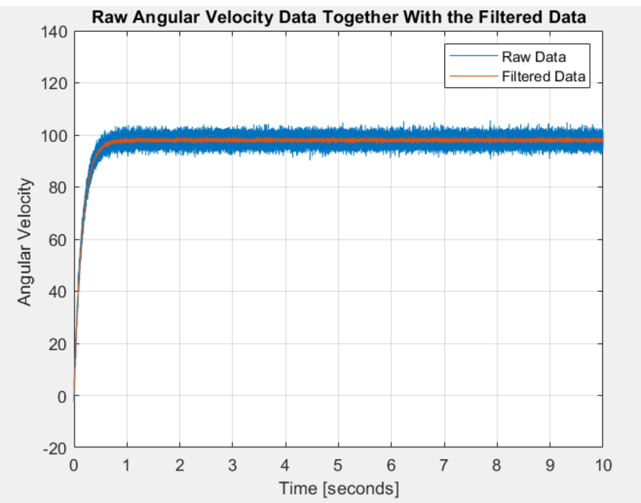

Figure-1: Raw and Filtered Angular Velocity Data

If we consider both plots, it is clear that the mathematical approximation of the angular velocity data is an exponential function with an upper boundary. Here is a mathematical representation of the behaviour of the DC motor **(2)**:

$$\omega(t) = K \cdot \left(1 - e^{-\frac{t}{\tau}}\right) \cdot u(t) \quad \textbf{(2)}$$

In this equation, K is the value of the angular velocity when t → ∞. It is the asymptotical upper-boundary. τ is the time constant which shows us how fast the system reaches to the asymptote.

To find K, we measured the mean value of the filtered angular velocity data from 2 seconds to 10 seconds. K turned out to be 98. To find the time constant, we observed the value of the angular velocity of the filtered data when the time was 0.05 seconds. The following equation was used **(3)**:

$$\omega(0.05) = 98 \cdot \left(1 - e^{-\frac{0.05}{\tau}}\right) \quad \textbf{(3)}$$

The time constant was calculated as 0.1560 sec^{-1}. This finalizes our equation for the angular velocity **(4)**.

$$\omega(t) = 98 \cdot \left(1 - e^{-\frac{t}{0.156}}\right) \cdot u(t) \quad \textbf{(4)}$$

Here you can see the approximation function of the angular velocity:

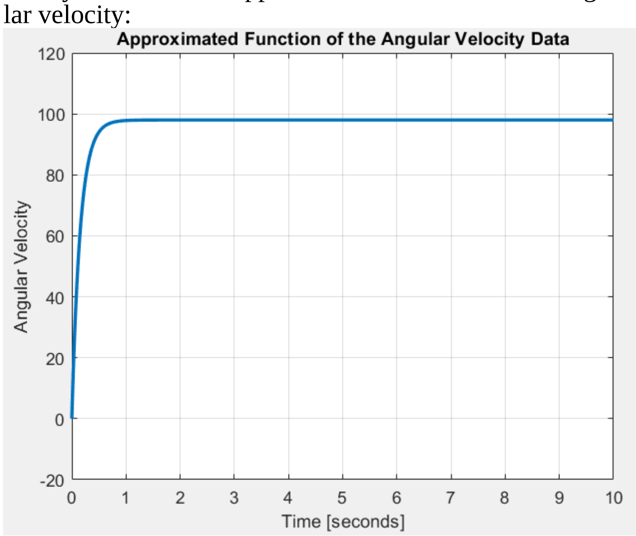

Figure-2: The Approximated Function

At this point, we should calculate the error in our approximation to check the validity of our approximation. Here you can see the plot of the error in our approximation:

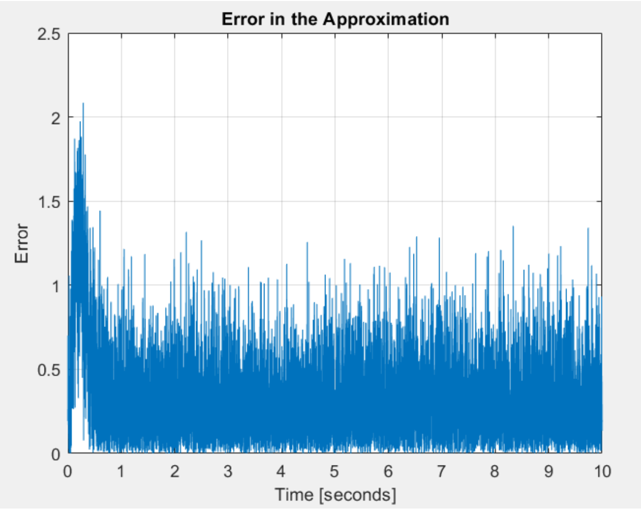

Figure-3: The Error of the Approximation

The expected value of the error in the system is 0.3319, which is relatively negligible considering the approximated function varies up to 98. This means we have a valid approximation.

Here is the laplace transform of the angular velocity **(5)**:

$$W(s) = \frac{98}{s} - \frac{98}{s+\frac{1}{0.156}} \quad \textbf{(5)}$$

We were given that the input was r(t)=6u(t). Then, the entire systems transfer function becomes **(6)**:

$$H(s) = \frac{W(s)}{R(s)} = \frac{\frac{98}{s} - \frac{98}{s+\frac{1}{0.156}}}{\frac{6}{s}} = \frac{104.701}{s+6.41026} \quad \textbf{(6)}$$

The low pass filter has a pole at Re{s}=1000, our transfer function H(s) has a pole at Re{s}=6.41026. As the difference of the magnitude of the poles is higher than 10; the poles don't affect each other. The pole of the transfer function is dominant to the pole of the low pass filter. Therefore, our approximation and results hold true.

2.2. P & PI Controllers

A proportional controller (P-controller) is a type of feedback control system which modifies the system's control signal u(t) in proportion to the error e(t), which is determined by comparing the setpoint and the actual output y(t). P controllers are commonly used in closed-loop control systems to ensure that the output of the system remains as close as possible to the setpoint. [1]

Here is an example block diagram of a P controller system:

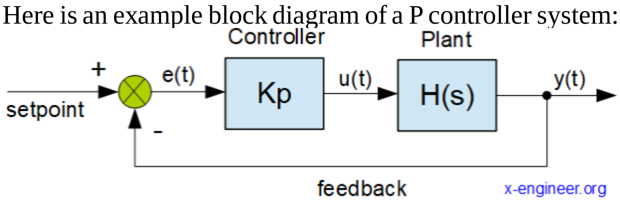

Figure-4: P-Type Controller Block Diagram [1]

In this diagram, Kp is the proportional control gain. e(t) is error signal which is the input of the controller, u(t) is the control signal which is the output of the controller. [1]. An equation between their relation can be given as **(7)**:

$$u(t) = K_p \cdot e(t) \quad \textbf{(7)}$$

A proportional integral controller operates in step by step manner to regulate a control system. It starts by calculating the error, which is the difference between the desired setpoint and current process variable. The controller's proportional (P) component multiplies this error by the proportional gain (Kp) and generates an instantaneous action. which is directly proportional to the error.

The Integral(I) component integrates gain (Ki) to calculate the cumulative sum of previous errors. The control output, which is applied to the system, is then calculated by adding the two components, P and I. This output adjusts the ultimate output of the system by minimizing the error over time and maintain the setpoint. [2]

Here is an example PI controller block diagram:

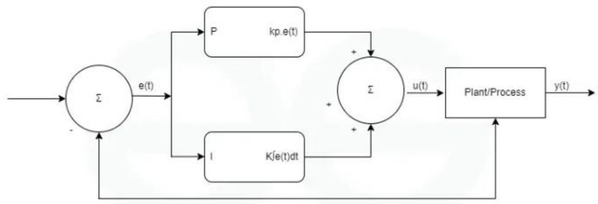

Figure-5: PI-Type Controller Block Diagram [2]

A relation for can be given as **(8)**:

$$m(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(t)dt \quad \textbf{(8)}$$

m(t) is the control signal, K_p is the proportional gain just as the P-controllers. K_i is the integral gain and e(t) is the error.

Now let's look at some general concepts of controllers. The following diagram is the system response of a second order system.

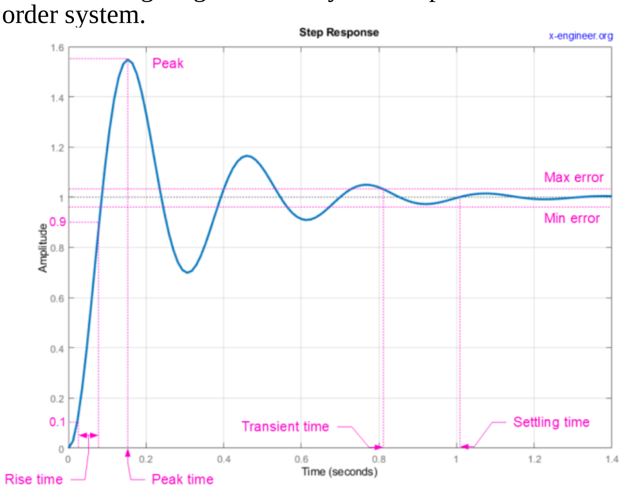

Figure-6: Second Order System Response [2]

Overshoot is the amount by which the system output exceeds the desired final value before settling. It is usually caused by high gain or insufficient damping. Too much overshoot can lead to instability and oscillations.[1]

Settling time is the time required for the system to remain within a small tolerance band. A faster settling time means the system stabilizes quicker. [1]

Peak time is the time at which the system reaches its maximum (peak) value. A system with high overshoot will have a higher peak time. It is influenced by the natural frequency and damping ratio. [1]

Steady state error is the difference between the desired output and actual output as time approaches infinity. It is caused by the system type, disturbances, or insufficient controller action. For a P controller, steady-state error remains nonzero, whereas a PI controller can eliminate it. This is the single-handedly biggest advantage of PI-type controllers over the P-types. An integral controller feeedback is necessary to eliminate steady state error. [1] [2]

As in our DC motor case, where we control the angular velocity of the motor, disturbances like friction and load changes affect our system. A simple Proportional (P) controller alone cannot completely eliminate steady-state error due to the nature of the system. Because a P controller only reacts to instantaneous error, it does not provide continuous correction for steady-state disturbances. An integral controller is necessary because if the steady-state error persists, the integral component keeps increasing until the error is eliminated. In a DC motor velocity control system, the integral term ensures the motor reaches and maintains the exact desired speed.

3. Conclusion

The preliminary work of this lab aimed to derive a first-order transfer function for a DC motor by analyzing the relationship between input voltage and angular velocity using MATLAB & SIMULINK. We obtained a valid approximation for the transfer function of a DC motor system.

On the second part of this preliminary work, two different controller types were introduced and compared based on several parameters such as overshoot, settling time, peak time. The effectiveness of PI-type controllers over P-type controllers was emphasized on reducing the steady state error up to zero.

REFERENCES

1. "Proportional (P) Controller" x-Engineer, 2024. Available: https://x-engineer.org/proportional-controller/ [Accessed: Feb 15, 2025].
2. "Proportional Integral Controller – Control System" GeeksforGeeks, 2024. Available: https://www.geeksforgeeks.org/proportional-integral-controller-control-system/ [Accessed: Feb 15, 2025].

APPENDICES

Matlab Code:

```
load('prelab1_response.mat');
filtdata = out.filtered_data;
plot(t,y);
hold on;
plot(t,filtdata);
grid on;
legend("Raw Data","Filtered Data");
ylim([-20 140]);
xlim([0 10]);
xlabel("Time [seconds]");
ylabel("Angular Velocity ");
title("Raw Angular Velocity Data Together With
the Filtered Data");
hold off;
gain = round(mean(filtdata(20000:100001)));
t0 = filtdata(500);
syms tau
time_cons = eval(solve(gain*(1-exp(-0.05/tau))
== t0, tau));
syms t
approx_out(t) = gain * (1- exp(-1*t/time_cons)
) * heaviside(t);
fplot(approx_out,"LineWidth",2);
grid on;
xlim([0,10]);
ylim([-20,120]);
xlabel("Time [seconds]");
ylabel("Angular Velocity");
title("Approximated Function of the Angular
Velocity Data");
% error = nan(1,10000);
for t = 1:10000
error(t)=abs(filtdata(t*10) - ap-
prox_out(t/1000));
end
plot((1:10000)/1000,error);
grid on;
xlabel("Time [seconds]");
ylabel("Error");
title("Error in the Approximation");
xerror = mean(error);
```