EEE342 Lab Report #1

Orkun İbrahim Kök

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

1. Introduction

The purpose the lab is to identify and control a DC motor system. In the first part is about hardware connection and configuration of the kit. In the second part, a first order approximation for a DC motor is will be used to identify the DC motor. In third part, 3 different PI controller will be designed and the effects of P and I coefficients (K_p and K_i) will be observed. Finally, choosing the optimal PI controller amongst the three, a closed loop response will be obtained.

2. Laboratory Content

2.1 Hardware Connection and Configuration

Fig. 1 shows the received velocity data for the DC motor.

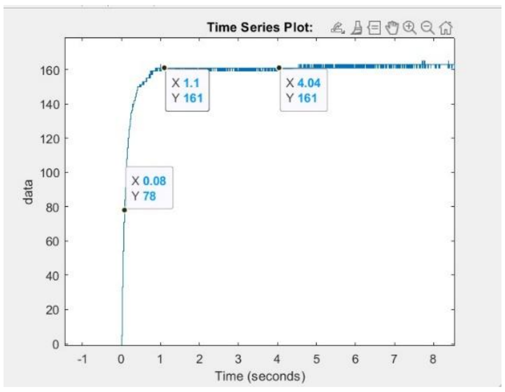

Fig. 1: Velocity vs Time graph for part 1.

The received data y(t) for the DC motor has a noise with it, as can be seen from Fig. 1.

2.2 System Identification in Time Domain

As in the preliminary work, an estimation for the first order approximation of the DC motor is again asked. The proposed first order approximation equation is given below.

$$G(s) = \frac{K}{\tau s + 1}$$

Where K is the gain, τ is the time constant. To find these values, the points on the Fig. 1 are going to be used.

$$Y(s) = U(s) \times G(s)$$

Here, U(s) is the 12V step response.

$$Y(s) = \frac{12}{s} \times \frac{K}{\tau s + 1}$$

Using partial fraction decomposition, y(t) is obtained.

$$Y(s) = \frac{A}{s} + \frac{B}{s + {}^{1}/_{\tau}}$$

Doing the necessary mathematical operations, A and B values are found.

$$A = 12K, B = 12K\tau$$

So the inverse Laplace transform of Y(s) is obtained and can be seen below.

$$Y(s) = \frac{12K}{s} - \frac{12K\tau}{s + {}^{1}/_{\tau}}$$

$$y(t) = 12K(1 - e^{-\frac{t}{\tau}})$$

Now, using the points on the Fig. 1, K value will be found. As seen from Fig. 1, as time (t) goes to infinity, the graph converges to 161. So equation 12K to 161 will give the value for K.

$$12K = 161$$

$$K = \frac{161}{12} = 13.42$$

For τ value, it is known that at 5τ, the plot settles. 5τ value is measured on the Fig. 1 as 1.1 seconds. Below equation gives the value of τ.

$$5\tau = 1.1\ second$$

$$\tau = \frac{1.1}{5} = 0.22\ second$$

As a result, the obtained first order approximation equation for the DC motor is

$$G(s) = \frac{13.42}{0.22s + 1}$$

Fig. 2 shows both hardware and simulation graphs of the DC motor. Hardware data belongs to Part 1, and simulation data belongs to the obtained first order approximation.

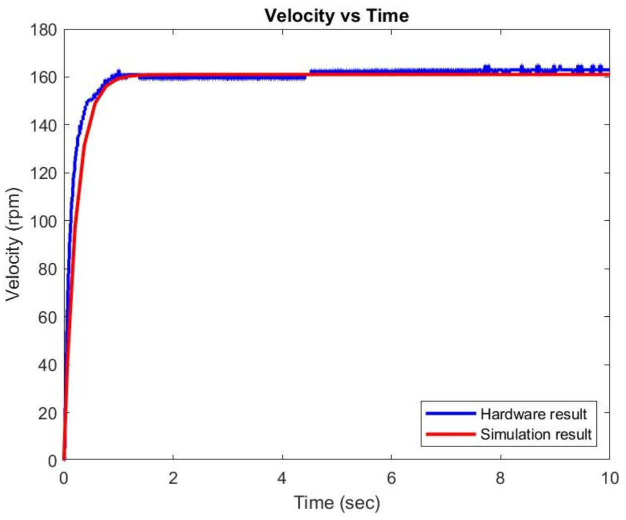

Fig. 2: Hardware and Simulation results for part 2.

As seen from Fig. 2, blue curve (hardware data) is noisy and has some spikes. However, red curve (simulation data) is less noisy because the data is filtered by a LPF, whose transfer function is given below.

$$H_{LPF}(s) = \frac{1}{0.01s + 1}$$

2.3 PI Controller on Simulation System

For part 3, a PI controller with the following specifications are asked to design:

- Steady-state error is 0,
- Maximum percentage overshoot is < 8%,
- Settling time is < 0.8 s (2% error bound).

The equation for the PI controller was obtained in the preliminary lab. Below equation shows the equation for PI controller.

$$C_{PI}(s) = K_p + \frac{K_i}{s}$$

Initially, it is asked to use K_p=0.5 and K_i=0.5. The input to the system r(t) = 120rpm. Using these values with the obtained first order approximation of the DC motor, gives a plot, which is available at Fig. 3.

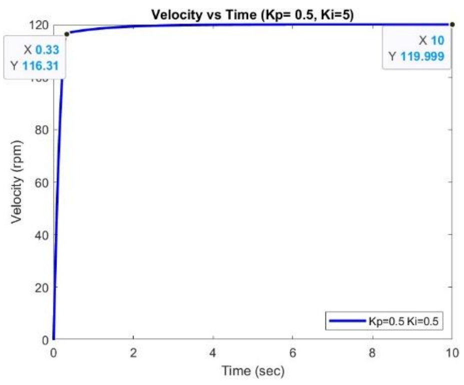

Fig. 3: Velocity vs Time graph of the DC motor with K_p=0.5 and K_i=0.5.

With these values, settling time is measured as 0.33 seconds. Also, an undershoot is observed. As seen from Fig. 3, After the settling time, curve continues increasing, which is defined as undershoot. As time (t) goes to infinity, the curve converges to 119.999, where initially the input to the system is 120. So it can be said that the steady-state error of the system is zero.

For the next PI controller, $K_p = 10K_i$ equality is used. For that case, K_p is set to 1, and K_i value is calculated as 0.1. Fig. 4 shows the corresponding graph for the DC motor response.

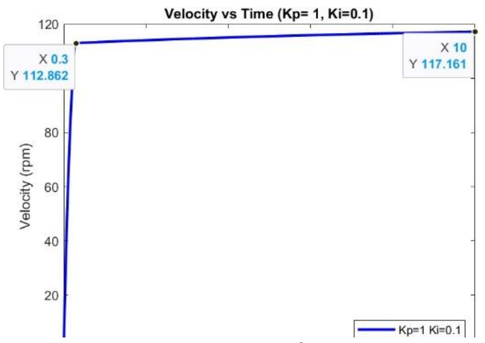

Fig. 4: Velocity vs Time graph of the DC motor with K_p=1 and K_i=0.1.

With K_p=1 and K_i=0.1 values, settling time is measured as 0.3 seconds. This response is again undershoot, because after the settling time, response keeps getting increasing. As mentioned earlier, The input to the system is r(t) = 120rpm, whereas, the last measured value of the system is 117.161. The steady state error is given in the below equation.

$$e_{ss} = \frac{|117.161 - 120|}{117.161} = 2.42\%$$

For the next PI controller, following equality is used to determine K_p and K_i values.

$$K_i = 20K_p$$

K_i value is chosen as 2, therefore K_p value is calculated as 0.1. Fig. 5 shows the response of the DC motor with K_p=0.1 and K_i=2.

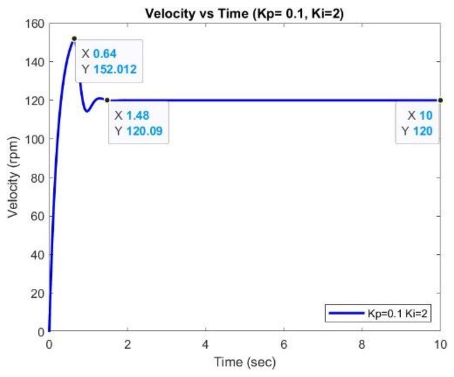

Fig. 5: Velocity vs Time graph of the DC motor with K_p=0.1 and K_i=2.

As seen from Fig. 5, the settling time is measured as 1.48 seconds, different from the previous responses. The steady state error is zero because the input to the system is r(t)=120rpm. However, there is overshoot. The below equation show the overshoot calculation.

$$Percentage\ Overshoot = \frac{152.012 - 120}{120} = 26.67\%$$

Finally, it is time to design a PI controller that meets all three specifications mentioned at the beginning of part 3. The chosen K_p and K_i values are given below.

$$K_p = 2, K_i = 0.8$$

With these values, the response of the system is plotted and can be seen in Fig. 6.

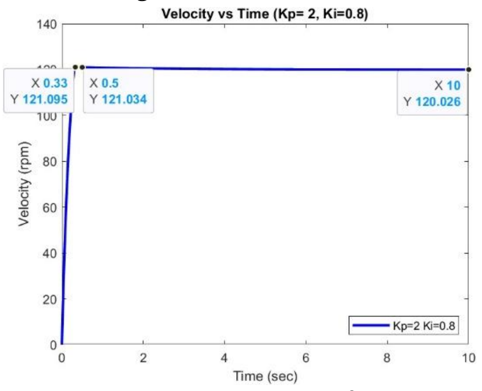

Fig. 6: Velocity vs Time graph of the DC motor with K_p=2 and K_i=0.8.

The settling time of the final PI controller is measured as 0.33 seconds, which meets the specification (<0.8 second). Looking at the overshoot value, the relevant equation is given below.

$$Percentage\ Overshoot = \frac{121.095 - 120}{120} = 0.9125\%$$

The calculated overshoot value also satisfies the specification value, which is specified as <8%. Finally, looking at the steady-state error, it is measured as zero because the final value of the system response is exactly the same with the input value r(t)=120rpm.

2.4 PI Controller on Physical System

In this part, the previously designed PI controller is implemented into the system. The hardware and simulation data are plotted on the same graph. Fig. 7 shows the plots.

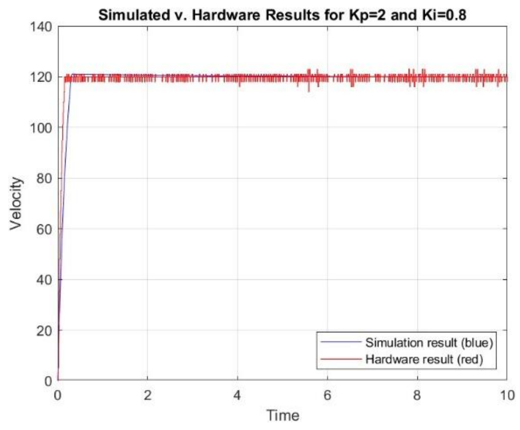

Fig. 7: Simulation and Hardware results of the system with the designed PI controller inserted.

The red curve shows the hardware result and blue curve shows the simulation result with a PI controller inserted with values K_p=2 and K_i=0.8. As expected, hardware result has a noise with it, but simulation data is filtered with a low-pass filter and an additional PI controller is used. Fig. 8 shows the zoomed-in version of the plots.

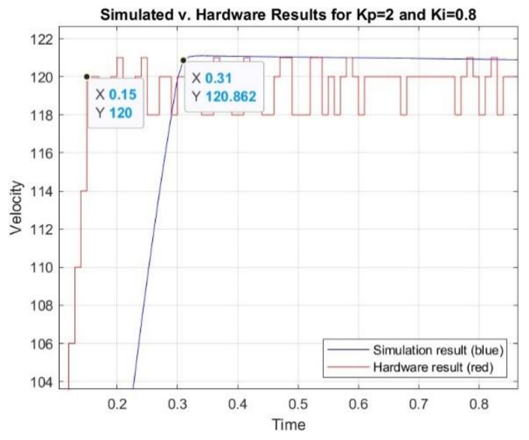

Fig. 8: Simulation and Hardware results of the system with the designed PI controller inserted (Zoomed-in).

As seen from the zoomed-in plot (Fig. 8), the hardware result has lots of spikes, whereas the simulation result is a line without spikes. The settling time of the hardware result is 0.15 second whereas the simulation result has a settling time of 0.33 second. The steady-state error of the hardware result is hard to measure because of the spikes. For the simulation result, as measured earlier, steady-state error is zero. For the overshoot measurement, the spikes in the hardware results are at most has a value of around 121 rpm, and at least a value of around 118 rpm. These values are inside the error range for the overshoot, which is specified as <8%. So, hardware result does not satisfy all three specifications that are earlier mentioned.

3. Conclusion

Initially, the hardware response of the DC motor is obtained. After that, a first order approximation is proposed for the DC motor. The proposed approximation is used for obtaining the simulation results, that are filtered by a low-pass filter. Then, three different PI controllers were proposed with the wanted K_p and K_i values. After these three, an optimal PI controller is designed to use with in simulation that satisfies the asked specifications. Finally, the hardware result is again compared with the simulation results with this time a PI controller inserted.

4. MATLAB CODE

PART 1)

```
vel_1 = out.velocity;
plot(vel_1)



figure; plot(vel_1,"b","LineWidth",2);
hold on; plot(out.vel_sim,"r","Li-
neWidth",2);
xlabel("Time (sec)"); ylabel("Velocity
(rpm)");
title("Velocity vs Time");
legend({"Hardware result","Simulation re-
sult"},"Location","SouthEast");
```

PART 2)

```
figure; plot(vel_2,"b","LineWidth",2);
vel_2 = out.y;
s = stepinfo(vel_2);
disp(s);
xlabel("Time (sec)"); ylabel("Velocity
(rpm)");
title("Velocity vs Time (Kp= 2, Ki=0.8)");
```

PART 3)

```
vel_3 = out_2_08.y;
figure; plot(vel_3,"b","LineWidth",2);
xlabel("Time (sec)"); ylabel("Velocity
(rpm)");
title("Velocity vs Time (Kp= 2, Ki=0.8)");
legend('Kp=2 Ki=0.8', 'Location', 'southe-
ast');
```

```
vel_3_1 = out_1_01.y;
figure; plot(vel_3_1,"b","LineWidth",2);
xlabel("Time (sec)"); ylabel("Velocity
(rpm)");
title("Velocity vs Time (Kp= 1, Ki=0.1)");
legend('Kp=1 Ki=0.1', 'Location', 'southe-
ast');

vel_3_2 = out_01_2.y;
figure; plot(vel_3_2,"b","LineWidth",2);
xlabel("Time (sec)"); ylabel("Velocity
(rpm)");
title("Velocity vs Time (Kp= 0.1, Ki=2)");
legend('Kp=0.1 Ki=2', 'Location', 'southe-
ast');

vel_3_3 = out_05_05.y;
figure; plot(vel_3_3,"b","LineWidth",2);
xlabel("Time (sec)"); ylabel("Velocity
(rpm)");
title("Velocity vs Time (Kp= 0.5, Ki=5)");
legend('Kp=0.5 Ki=0.5', 'Location', 'sout-
heast');
```

PART 4)

```
vel_4 = out.velocity;
figure;
plot(vel_3, 'b'); % blue line → hardware
result
hold on;
plot(vel_4, 'r'); % red line → simulation
result
legend('Simulation result (blue)',
'Hardware result (red)', 'Location', 'so-
utheast');
xlabel('Time');
ylabel('Velocity');
title('Simulated v. Hardware Results for
Kp=2 and Ki=0.8');
grid on;
```