Designing PI Controllers for Stabilizing DC Motor Systems

Oğuz Altan

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

1. Introduction

The aim of this laboratory assignment is to design a PI controller such that the step response of the closed velocity loop has least settling time, so that the system gives a fast step response. For an LTI system, the settling time is minimum when the maximum value of the real parts of poles of the closed loop transfer function is minimized. As we aim to find the least settling time, picking the poles, which are furthest away from the imaginary axis, is an important factor.

The step responses of open loop DC motors are aimed to be obtained with first order approximations. This lets us to design our PI controller, using root locus technique that controls the DC motor in our system. In this laboratory assignment, we are to design 3 different PI controller that stabilize the system.

2. Laboratory Content

2.1 System Identification in Time Domain

In this part, we implement first order approximations for step inputs $r(t) = 4u(t)$ and $5u(t)$ and obtain hardware results using the DC motor. We approximate the step input response using the equation

$$Y(s) = \frac{K_{step}}{s}\frac{A}{s+B}$$

Transforming to time domain:

$$y(t) = \int_{-\infty}^{t} K_{Step} u(\tau) A e^{-B(t-\tau)}\, d\tau$$

For these two step input responses, we sample two points on the data, as there are two unknowns A and B and approximate the step input response.

The response of the system in time domain can be written as:

$$y(t) = K_{step} A e^{-Bt} \int_0^3 e^{B\tau}\, d\tau$$

After the integration is solved the resulting equation is:

$$y(t) = \frac{K_{step}A}{B} - \frac{K_{step}Ae^{-3B}}{B}$$

$$y(t) = \frac{K_{step}A}{B}\left(1 - \mathrm{e}^{-\mathrm{Bt}}\right)$$

For the $r(t) = 4u(t)$ input, we got:

$$y(t) = \frac{4A}{B} - \frac{4Ae^{-3B}}{B} = \frac{4A}{B}(1 - e^{-3B})$$

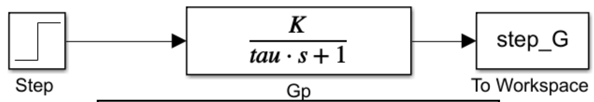

Figure 1: Simulink Block Diagram

We sample two points on the data, which is obtained from hardware test using DC motor, at t=3 and t = 5. At t=3, we get output 71 and at t= 5, we get output 73. Then, we solve the equation above and find the unknowns A and B.

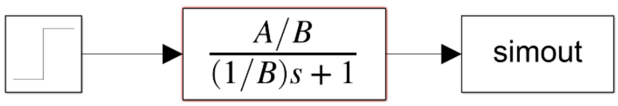

Figure 2: First Order Approximaton Block Diagram

Implementing the first order approximation:

$$G_p(s) = \frac{A_1}{s + B_1} = \frac{K_1}{(\tau_1 s + 1)}$$

$$A_1 = 21.94$$
$$B_1 = 1.166$$

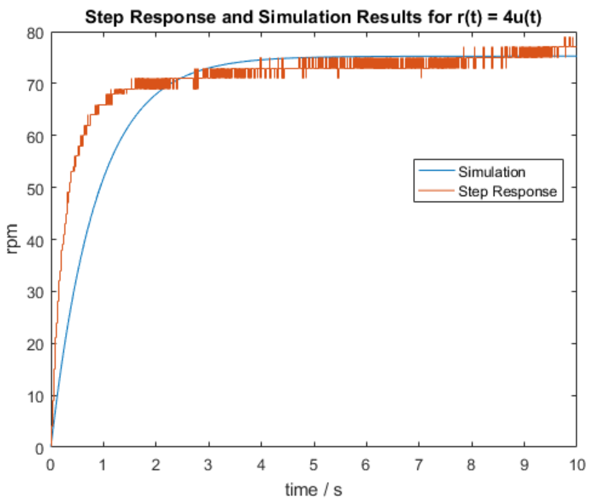

Figure 3: First Order Approximaton Result for 4u(t)

Similar for $r(t) = 5u(t)$ input, we collect two samples from the hardware result at t = 3 and t = 7, which give 111 and 114, respectively. We solve the two equations and get:

$$A_2 = 27.5926$$
$$B_2 = 1.20995$$

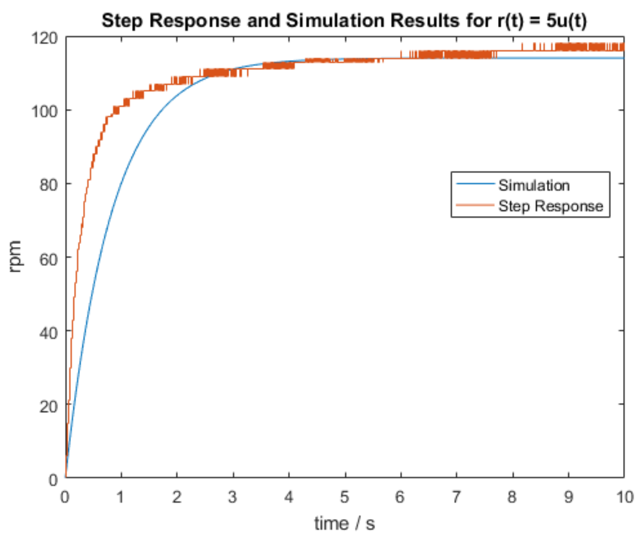

Figure 4: First Order Approximaton Result for 5u(t)

The gains and time constants are found as:

$$K_1 = \frac{A_1}{B_1} = 18.8164$$

$$K_1 = \frac{A_1}{B_1} = 18.8164$$

$$\tau_1 = \frac{1}{B_1} = 0.857633$$

$$\tau_2 = \frac{1}{B_2} = 0.0.82648$$

$$K = \frac{K_1 + K_2}{2} = 20.8106$$

$$\tau = \sqrt{\tau_1 \tau_2} = 0.8419$$

As we use first order approximation, we cannot get the exact output as the hardware gives, because in real life DC motor plant is not first order. We just make approximations.

The resulting transfer function $G_p(s)$ is:

```
g_p =

      -0.1041 s + 20.81
  --------------------------
  0.004209 s^2 + 0.8469 s + 1

Continuous-time transfer function.
```

In this laboratory assigenmnt, we used first order approximations and in the first lab, we had used second order approximations. The result of this first order approximations are better and more consistent that we had obtained in first lab. We get 2 different results, each for two different reference input. The system itself is nonlinear but be linearize it around the reference input.

2.2 PI Controller Design by Using Root Locus

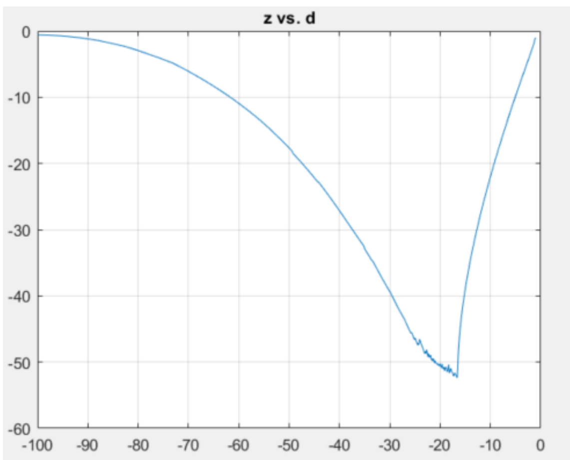

Figure 5: Plot of the Minimum of the Maximum of the Real Parts of the Closed Poles

The minimum is at z = -16.6
Thus, the z1, z2 and z3 are:

z1 = -16.6
z2 = -8.3
z3 = -5.533

The gains corresponding these z values are:

$$K_{C1} = \quad 29.4372$$
$$K_{C2} = \quad 8.7421$$
$$K_{C3} = \quad 3.9589$$

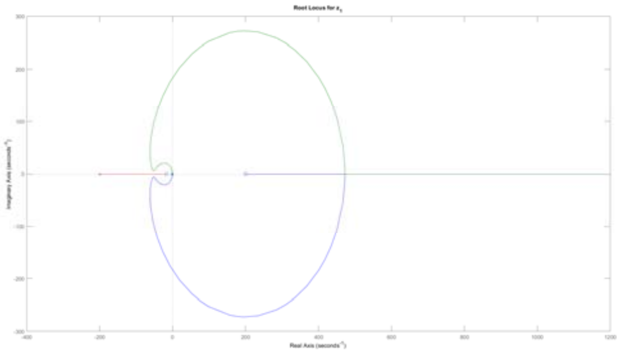

Figure 6: Root Locus for Z1

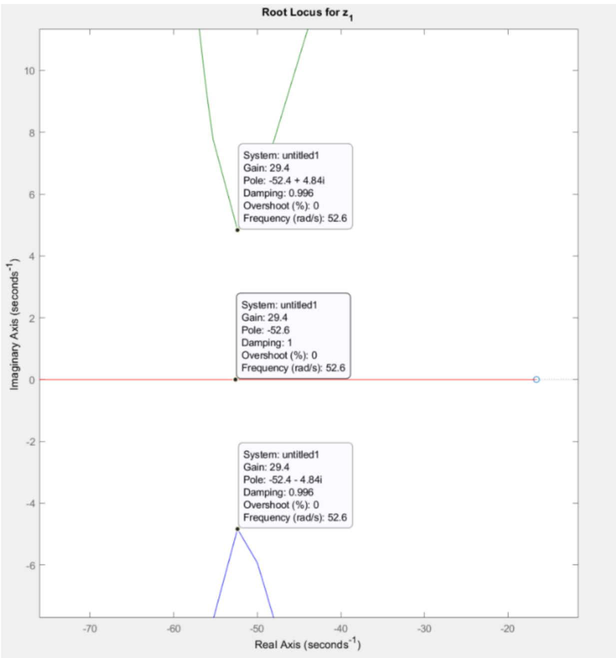

Figure 7: Poles on the Z1 Root Locus

On the plot, it can be seen that gain $K_{C1} = \quad 29.4$

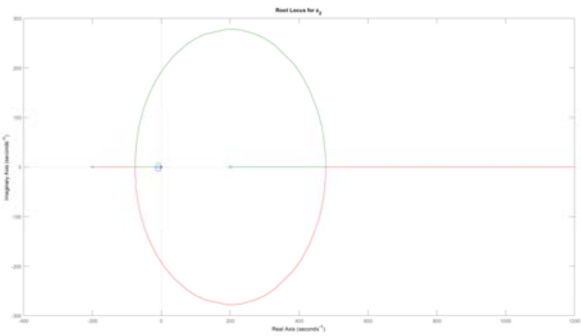

Figure 8: Root Locus for Z2

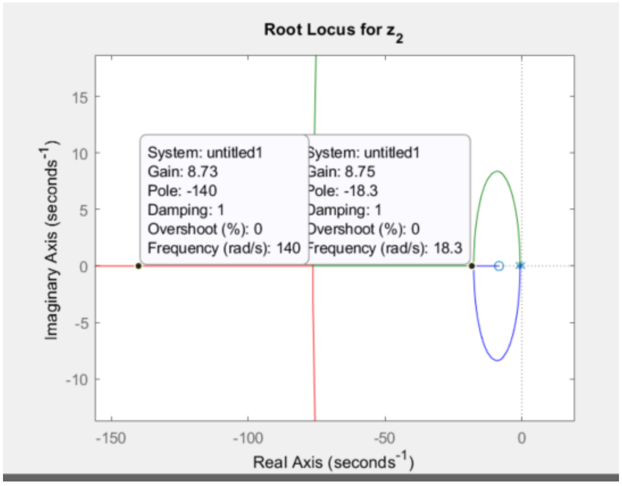

Figure 9: Poles on the Z2 Root Locus

On the plot, it can be seen that the gain $K_{C2} = \quad 8.7421$

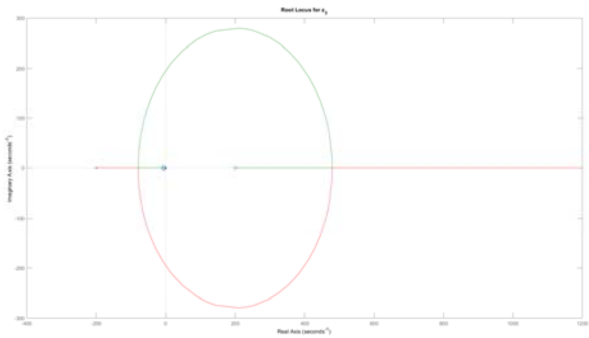

Figure 10: Root Locus for Z3

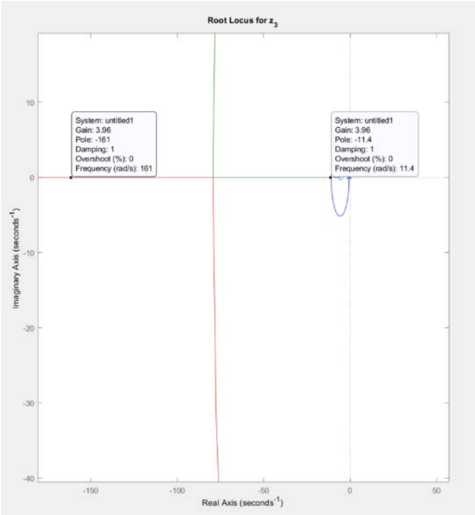

Figure 11: Poles on the Z3 Root Locus

On the plot, it can be seen that the gain $K_{C3} = 3.9589$

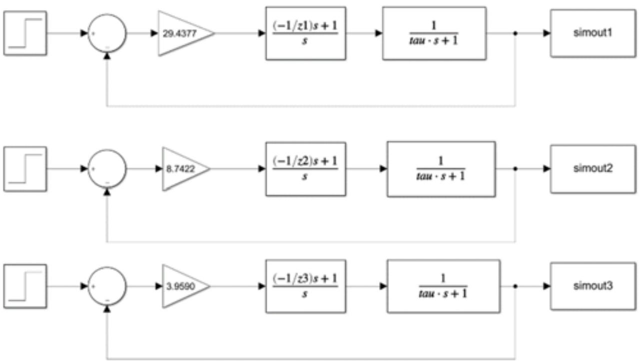

Figure 12: Simulink Block Diagrams for Closed Loop PI Systems

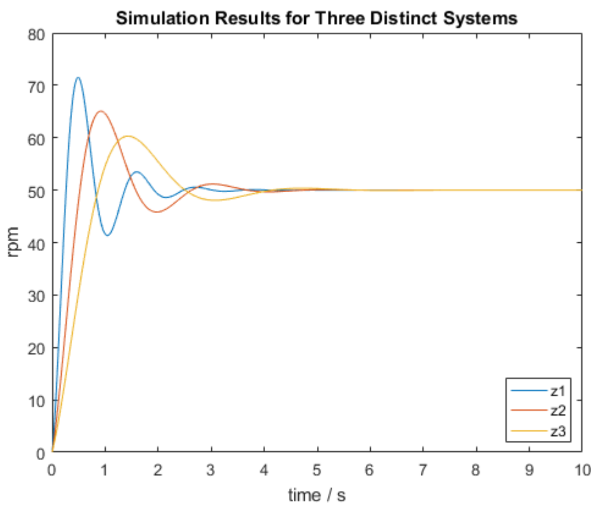

Figure 13: Simulation Results for z1, z2 and z3 with 50u(t) Input

We claimed that as the z1 case has the best values about pole-zero locations, therefore, this PI 1 controller reaches the steady state in first place and there after come PI 2 and PI 3. This is justified, it can be seen on the plot above, Figure 13. In addition, analyzing the overshoot, we can say that as the first controller PI 1 stabilizes first, it has the most overshoot comparing to the other two controllers.

2.3 Controller Implementation on Hardware

In this part, we implement our just designed PI controllers and take them to test drive. We collected 3 simulation results for 3 different PI controllers and the plot of the data is given below:

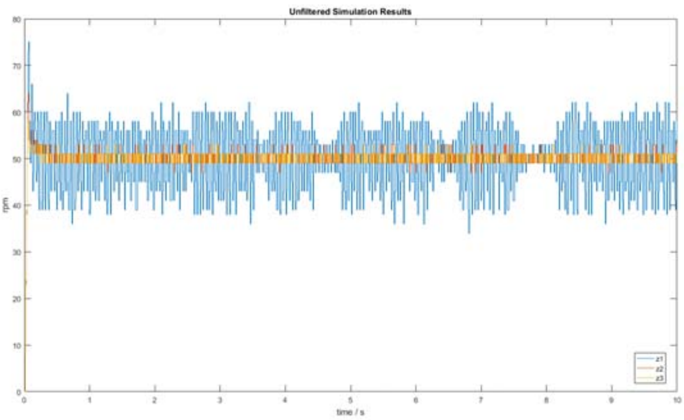

Figure 14: Unfiltered Simulation Results for 3 PI Controllers

We use Low Pass Filter to filter the simulation results to get a clearer plot.

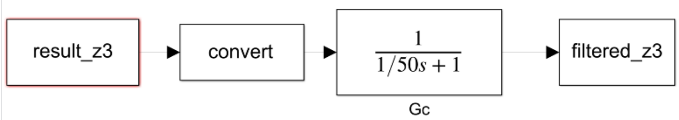

Figure 15: Low Pass Filter Block Diagram

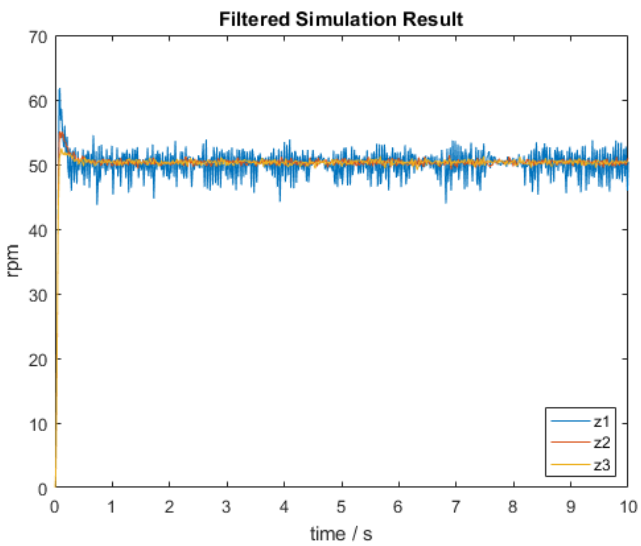

Figure 16: Low Pass Filtered Simulation Results for 3 PI Controllers

Although the results are a little bit noisy, it can be seen that the simulation results are as expected: The PI 1 comes first it reaches to the steady state in first place and there after come two other controllers, PI 2 and PI 3. The claimed that we made in the preliminary report that PI 1 reaches to the steady state as first, is correct and justified with the simulations.

3. Conclusion

In this laboratory assignment, our goal was to make first order approximations on a system with DC motor and design a PI controller with minimal settling time i.e. it reaches steady state, in the shortest amount of time possible. For this, we forced poles, which are closest to each other, to the furthest point according to the imaginary axis.

By using different zero locations, we designed the optimal PI controller. Finally, in the last part, we simulated our 3 PI controllers and observed that PI 1 is the most stable controller, it reaches steady state first.

In our experiments, we used theoretical side of controllers, plants and closed loop feedback control systems. In real life, the results may not be exactly same as we find in simulations. In fact, the real life outputs may have noise as the gain increases, as well as the real life systems oscillate about the steady state value. Thus, to achieve success in real life, we should take care of real life parameters, constraints and models while designing controllers.

REFERENCES

1. Oğuz Altan, "PI Controller Design with Least Settling Time", *Electrical and Electronics Engineering Department, Bilkent University,06800 Ankara, Turkey*

APPENDIX

```
clc; close all;
h = 0.01;
K = 20.8106;
tau  = 0.8419;
plant = tf([-K*h 2*K],[tau*h (2*tau)+h
2]);
maks = zeros(991, 65);
mins = zeros(991, 1);
Ks = zeros(991,1);
count = 1;
z = -100:0.1:-1;
for i = -100:0.1:-1
    g_c = tf([-1 i], [i 0]);
    [R,L] = rlocus(g_c * plant);
    for j = 1:65
        maks(count, j) =
max(real(R(:,j)));
    end
    count = count + 1;
end
[mins, Ks] = min(maks, [], 2);
plot(-100:0.1:-1, mins);
grid; title('z vs. d');
[~, i] = min(mins);
figure;
z1 = z(i);
g_c = tf([-1 z1], [z1 0]);
rlocus(g_c * planttitle('Root Locus for
z_{1}');
figure;
z2 = z1 / 2;
g_c = tf([-1 z2], [z2 0]);
rlocus(g_c * plant);        title('Root
Locus for z_{2}');
figure;
z3 = z1 / 3;
g_c = tf([-1 z3], [z3 0]);
rlocus(g_c * plant);        title('Root
Locus for z_{3}');
hold off;

zs = [z1 z2 z3];
for i = 1:3
  disp('------------- ');
  z = zs(i);
  g_c = tf([-1 z], [z 0]);
  [R, L] = rlocus(g_c * plant);
  a = zeros(1, 68);
  for m = 1:68
      a(m) = max(real(R(:,m)));
  end
  [T,I] = min(a);
  disp(T);
  disp(L(I));
  disp(R(:,I));
end
```