PI Controller Design by Using Root Locus

Ayhan Okuyan

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

1. Introduction

This laboratory assignment consisted of designing a root locus based on the system that we were given. Hence, we have first identified the system from its step response and modelled it as a first order system. Then, we have approximated the data that we have received from the step response. Hence, we have configured a model for the system. The next process was to define our model as a transfer function and design three different pi controllers for that system as we have done on the Lab2 Preliminary Lab. We have defined the PI controller unknown as z and picked an interval for it from -100 to -1 with a step size of 0.1. By, using root locus, we have optimized the parameter with respect to minimum settling time. Then, we have picked the other two z values as the half and one-third of the original value that we have picked. Then, we have simulated the PI controlled systems on Simulink and plotted the step responses of each. Then, we have implemented the controller that we have configured on the system and ran them accordingly. We have plotted the responses and commented on them. One important note is that the sections bordered are exactly taken from the written MATLAB code.

2. Laboratory Content

2.1. System Identification in Time Domain

First thing that we have done on this laboratory assignment was to identify the system that we were given, which in this case is the DC Motor plant that is assigned to our group. We have configured the system in the same manner as we died in Lab1, which is explained in detail in the Lab Report 1. First, we have pulled the step response of the system to the input *r(t)=4u(t)*. The response of the system is given below.

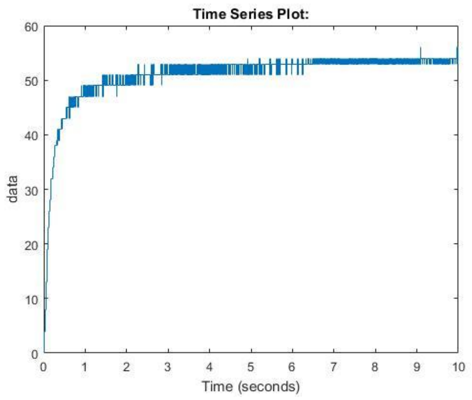

Fig. 1: System's response to 4u(t)

Then, we have used the following first order approximation that we is given below to approximate the data for this response and found the specified constants A and B.

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

$$y(t) = \int_{-\infty}^{t} K_{step}u(\tau)Ae^{-B(t-\tau)}d\tau$$

$$= 4\int_{0}^{t} Ae^{-B(t-\tau)}d\tau = \frac{4A}{B}(1-e^{-Bt})$$

Where K is equal to 4. Then we have picked two points given as

```
% 0.5 43
% 5.93 53
```

and created an equation system with two unknowns and solved for A and B. Then, A and B became,

```
A1 = 44.1942;
B1 = 3.33541;
```

Then we have simulated this system on Simulink and looked at its step response, which is given below.

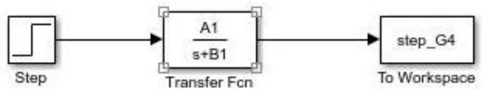

Fig. 2: The constructed Simulink Model for step input r(t)=4u(t)

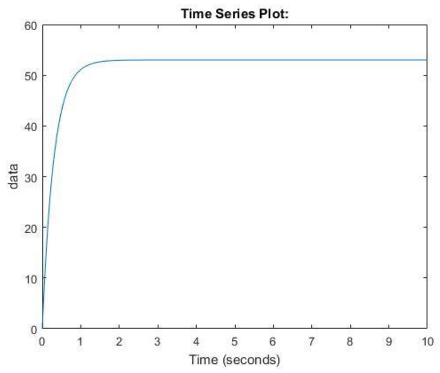

Fig. 3: Approximated step response for the step input r(t)=4u(t)

Hence, we have plotted the real and approximated responses on top of each other to see if this is an acceptable approximation.

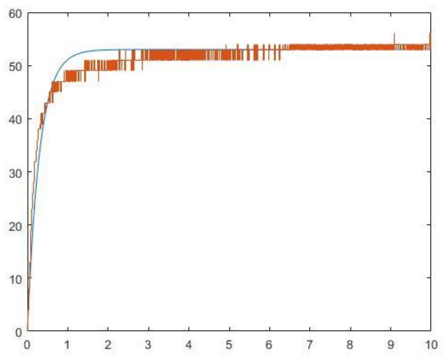

Fig. 4: The obtained and approximated step responses for the step input r(t)=4u(t)

By looking at fig. 4, we can say that this is an acceptable approximation mostly. Then, we have done this for a step input of *r(t)=5u(t)*. The obtained data is given below.

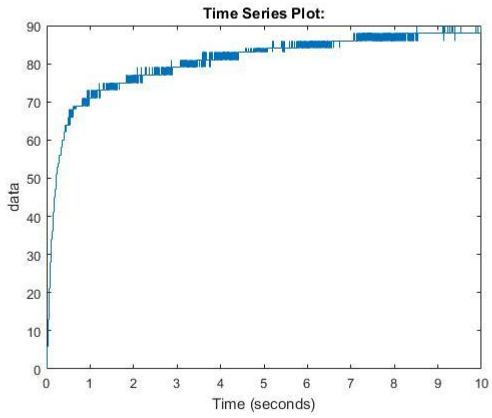

Fig. 5: System's response to 4u(t)

Then we have picked two points on the graph and then approximated the A and B constants. The taken points and their corresponding solutions are below.

```
% 0.34 58
% 5.36 84
A2 = 55.706;
B2 = 3.16511597;
```

Then, we have created the following Simulink model in order to observe the approximated response. The constructed model and the corresponding step response are given below.

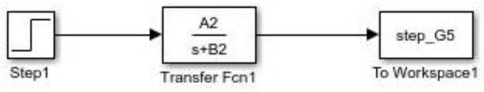

Fig. 6: The constructed Simulink Model for step input r(t)=5u(t)

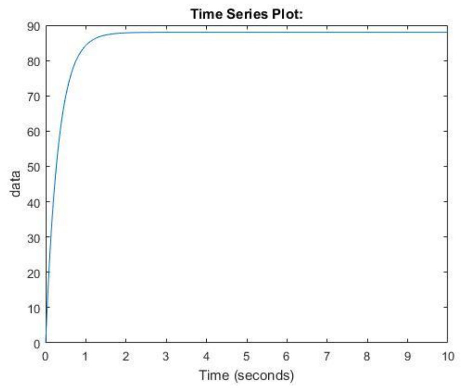

Fig. 7: The constructed Simulink Model for step input r(t)=5u(t)

Hence, we plot these two responses on top of each other.

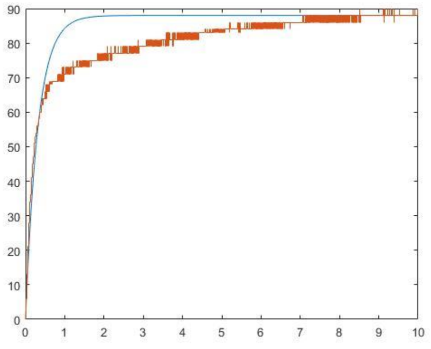

Fig. 8: The obtained and approximated step responses for the step input r(t)=5u(t)

This approximation seems to be cruder than the other one since the transient part is more inconsistent when compared with the approximation of 4u(t). However, since the steady state value and the beginning of the transient response are similar, this approximation was also admissible. Then, from the obtained values of A and B, we decide on the K and τ values of our approximation of the first order system.

$$\frac{A}{s+B} = \frac{A/B}{(1/B)s+1} = \frac{K}{\tau s+1}$$

$$K = \frac{A}{B}, \tau = \frac{1}{B}$$

Then, we approximate the K and τ values using the approximations that were given in the lab assignment.

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

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

Hence, we have written the following script to obtain these values.

```
K1 = A1/B1;
tau1 = 1/B1;
K2 = A2/B2;
tau2 = 1/B2;
K = (K1 + K2) / 2;
tau = sqrt(tau1*tau2);
```

Hence, we have identified the system that we will be controlling. The difference between the two different step responses was that the second data obtained took much longer to achieve to steady state than the first one. They are different since the values they need to achieve were different from each other. By using two different controllers, we were able to achieve a more precise model. When compared with the estimation that we have done on the first lab, this approximation was more precise than the other one. In the lab for the first circuit, we assumed that the second order was dominant and made our assumptions accordingly. However, it was not the case. In this one we used the Padé approximation, which was more precise as it is the best rational approximation for a rational function.

2.2. PI Controller Design by Using Root Locus

In this part of the lab, we were supposed to find the most optimal PI controller hyperparameter for our constructed system. For this, we have used the same method that we have implemented on Preliminary Lab 2. The code is given below.

```
format long
close all
h = 0.005;
plnt_poles = [(tau*h) (tau+h) 1];
plnt_zeros = [(-h*K) (K)];
plnt = tf(plnt_zeros,plnt_poles);
int = -100:0.1:-1;

d_vals = [];
for z = int
```

```
    contr_poles = [1 0];
    contr_zeros = [-1/z 1];
    contr = tf(contr_zeros, contr_poles);
    [roots,ks] = rlocus(plnt*contr);
    max_re = max(real(roots));
    min_max_root = min(max_re);
    d_vals = [d_vals min_max_root];
end

% plot the d-values
figure();
plot(int,d_vals)
title('d Values');
xlabel('z');
ylabel('min(max(real(roots)))');
% min=== z=-23.5 d=-57.49

[minimum_d,min_index] = min(d_vals);
z1 = int(min_index);
contr_poles = [1 0];
contr_zeros = [-1/z1 1];
contr = tf(contr_zeros, contr_poles);
figure();
rlocus(contr*plnt);
title('Root Locus for z1');

% plt the root locus for z2 and z3
z2 = z1 / 2;
z3 = z1 / 3;
contr_poles = [1 0];
contr_zeros = [-1/z2 1];
contr = tf(contr_zeros, contr_poles);
figure();
rlocus(contr*plnt);
title('Root Locus for z2')
contr_poles = [1 0];
contr_zeros = [-1/z3 1];
contr = tf(contr_zeros, contr_poles);
figure();
rlocus(contr*plnt);
title('Root Locus for z3');
% gain for z1=2.62, z2=1.07, z3=0.677
kz1 = 15.2;
kz2 = 4.22;
kz3 = 1.73;
```

With the values that we have found, we define the transfer function of our plant as,

$$G_p(s) = \frac{K}{\tau s + 1} \times \frac{1 - 0.005s}{1 + 0.005s}$$

Furthermore, we defined the PI controller that we have implemented as

$$G_c(s) = K_c\left(\frac{\frac{-s}{z} + 1}{s}\right)$$

We have then used the rlocus() method of MATLAB to obtain the poles of the root locus branches for varying K. Even further, we have done this for all the loci with z varying from -100 to -1 with a step size of 0.1. The plot of the minimum real parts of maximum real poles that we have recovered from these root loci is below.

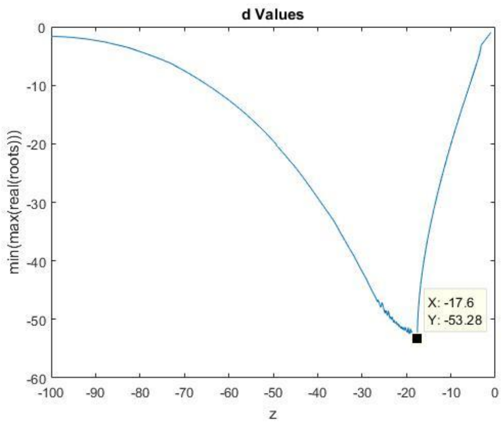

Fig. 9: The plot for the minimum real parts of the maximum poles (z_1)

Here, we pick the minimum value in order to optimize for the settling time which is -59.28. Then, we have plotted the root locus for the minimum z value and decided on the controller gain by looking at that exact point on the root locus, the root locus plotted is given below.

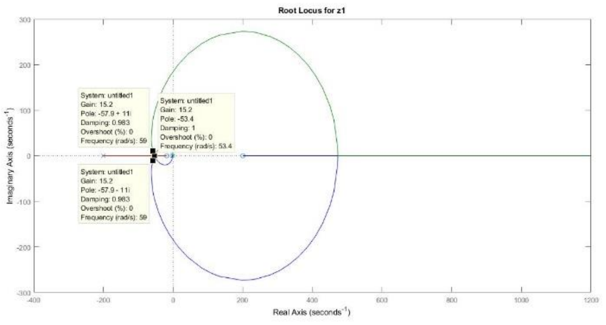

Fig. 10: The root locus for z_1

Here, we observe the gain as 15.2. Then, as requested, we defined the z_2 and z_3 as $z_1/2$ and $z_1/3$ respectively. Then, we use Fig. 9 again to find the pole locations for these values. The plots for these are below.

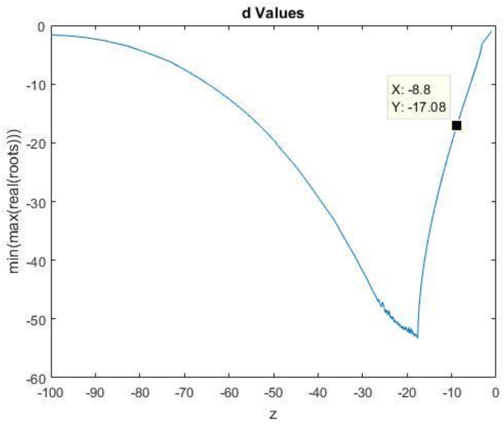

Fig. 11: The plot for the minimum real parts of the maximum poles (z_2) Here, the root is found to be -17.08.

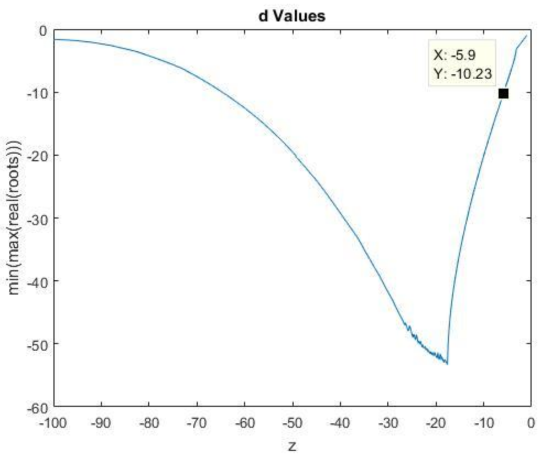

Fig. 12: The plot for the minimum real parts of the maximum poles (z_3)

Here, the root is found to be -10.23. Hence, we draw the root loci for these values also to determine their respective controller gain values.

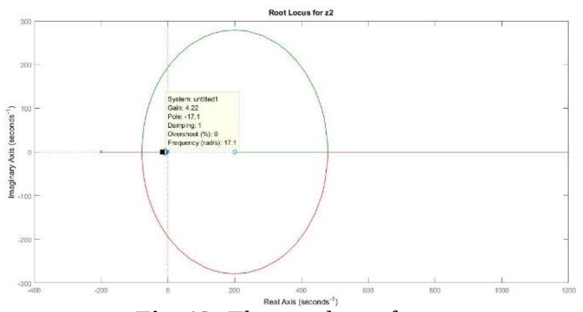

Fig. 13: The root locus for z_2

For, z_2 we observe the gain as 4.22. Hence, we follow the same routine for z_3. The root locus is hence given below.

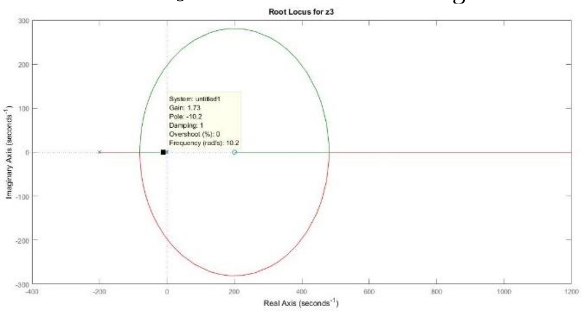

Fig. 14: The root locus for z_3

The gain for z_3 is found to be 1.73.

The following step is to construct all three of the systems that we have configured on Simulink. Then we have plotted the step responses of these with respect to the same step input *r(t)=50u(t)*. We have designed the following schematic and the three step responses put on top of each other are given below.

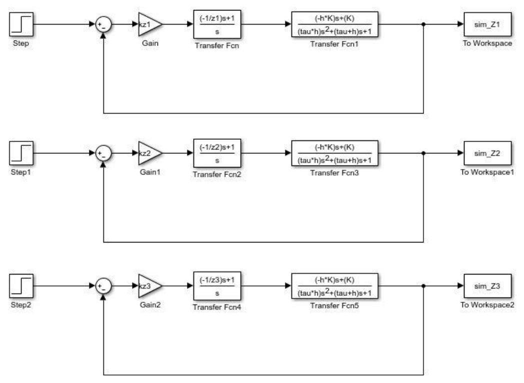

Fig. 15: The schematics for the PI controller systems for z_1, z_2 and z_3

The plot that considers the output for all these systems are given below. In order to observe the transient part better, the simulation is done from 0 to 2 seconds.

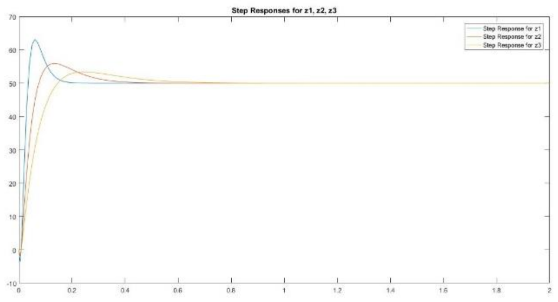

Fig. 16: The step responses for the PI controller systems for z_1, z_2 and z_3

Here, we observe that the least settling time we could get was with z_1 and it was also the one with the highest maximum overshoot. This is expected since the overshoot and settling time are inversely proportional. Some loss is inevitable while trying to achieve the other. In z_2 and z_3, the settling times are higher as expected, however the maximum overshoot values were lower. The code used to construct these plots are given below.

```
figure();
plot(sim_Z1);
hold on;
plot(sim_Z2);
plot(sim_Z3);
title('Step Responses for z1, z2, z3');
legend('Step Response for z1', 'Step Response
for z2', 'Step Response for z3');
```

2.3. Controller Implementation on Hardware

In this part, first we have configured the hardware circuit for each of the z value that we have obtained created the Simulink schematics given below.

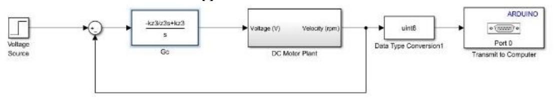

Fig. 17: The hardware implementation schematic for z_3

We have repeated this process for z_1 and z_2 but only this is showed as an example. After these are configured, we have read the data that is given back by the system and plot them.

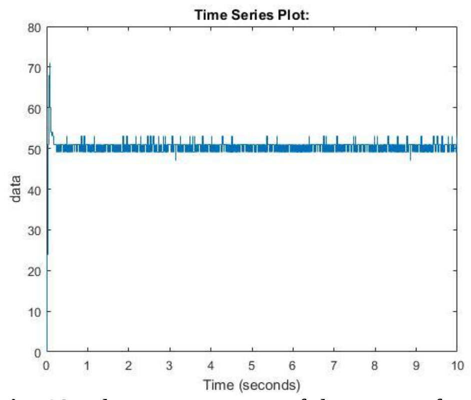

Fig. 18: The step response of the system for z_1

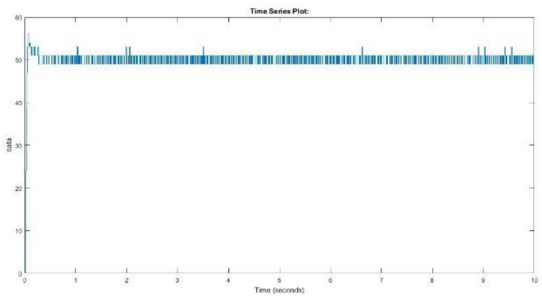

Fig. 19: The step response of the system for z_2

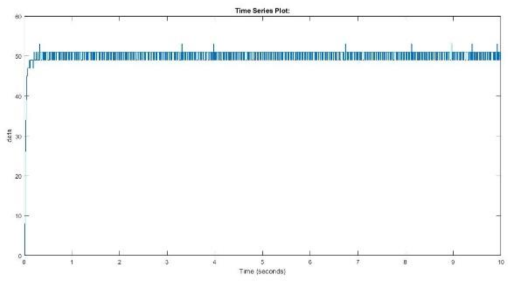

Fig. 20: The step response of the system for z_3

Hence, we plotted these three on the same plot in order to compare the responses that we have obtained from our designed PI controlled feedback systems. The code is given below.

```
figure();
plot(positionz1);
hold on;
plot(positionz2);
plot(positionz3);
title('Output of z1, z2, z3');
legend('z1', 'z2', 'z3');
```

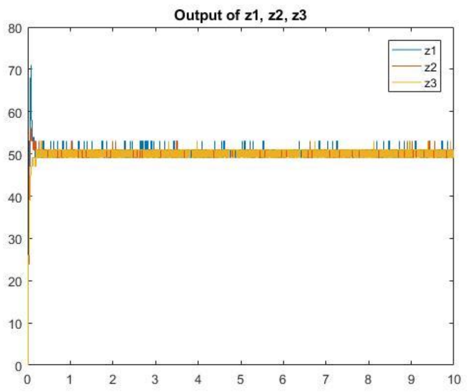

Fig. 21: The step response of the system for z_1, z_2 and z_3

This plot is not informative at all, since the responses are mixed with each other. Then, we have used the given low pass filter lab2_LPF.slx in order to filter those responses and see a clearer output. The filter schematic is given below.

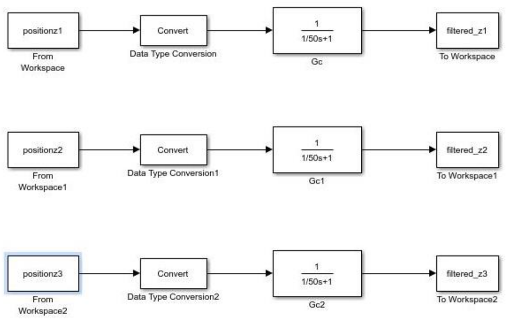

Fig. 22: The Simulink models for the low pass filter.

The code and the plotted filtered responses are given below.

```
figure();
plot(filtered_z1);
hold on;
plot(filtered_z2);
plot(filtered_z3);
title('Low Pass Filtered z1, z2, z3');
legend('Filtered for z1', 'Filtered for z2',
'Filtered for z3');
```

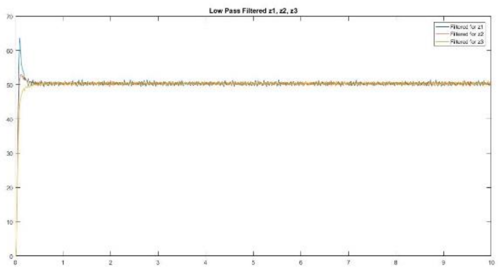

Fig. 23: The filtered responses for the step responses

These results mainly match the models that we have estimated. However, one difference is that the maximum overshoot of the third z value is smaller than the steady state value. The results with the settling time is as expected since the settling times are as z_1, z_2 and z_3 in ascending order.

3. Conclusion

In this lab assignment, we were asked to design a PI controller and optimize it for settling time. We have first simulated the system from the data that we have read from the DC Motor plant system. Then we have approximated the parameters for first order approximation from the models that we are given. Then, using the preliminary lab, we have found the parameters for three different PI controllers. Then we have tested these controllers on the actual systems and commented on them. Overall, this lab assignment was a success and we were able to design a PI controller that is optimized for the settling time.

REFERENCES

1. Dorf ,Richard C and Bishop, Robert H. Modern Control Systems. Essex: Pearson. Thirteent edition. (2017)