ECSE 307 - Lab 4

Ali Shobeiri - 260665549
Ismail Faruk

Introduction

The main objective of this laboratory was to get a basic understanding for how feedback control can be used to modify the behaviour of a dynamic system. We will consider P, PI, PD, and PID control of a third order system. We will manually tune the PID values, use algorithmic methods and also the Ziegler Nichols method to generate controller constants and evaluate each method's performance.

Effect of P, I, D:

The purpose of using the Proportional, Integral, and Derivative controllers and their combinations is mainly due to the effects they have on the system to achieve desirable design specifications.

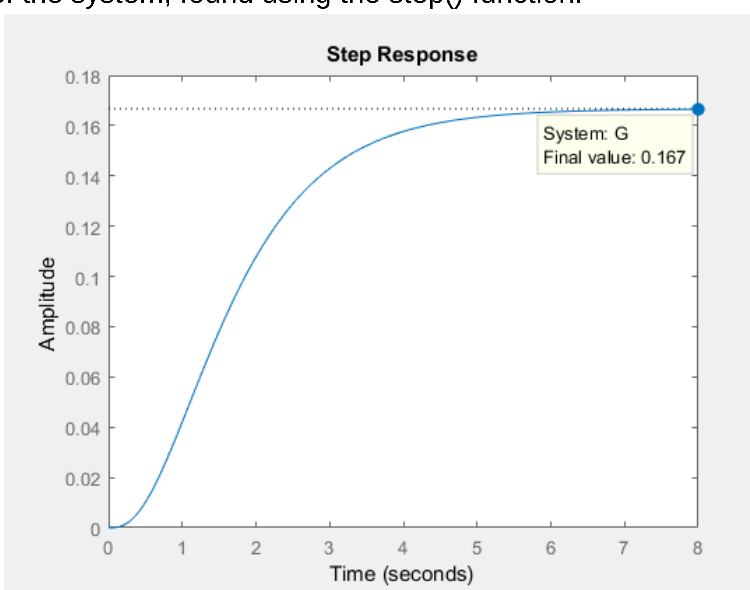
Proportional controllers increase the gain of the output while decreasing the tracking error.

Integral controllers, regardless of their value, would always result in zero steady state tracking error to step input, however, a very high integral gain may make the system unstable.

Derivative controller allows the system to take control action based on the trend in the error signal, though it tends to amplify the noise.

Problem 1

a) Step response of the system, found using the step() function:



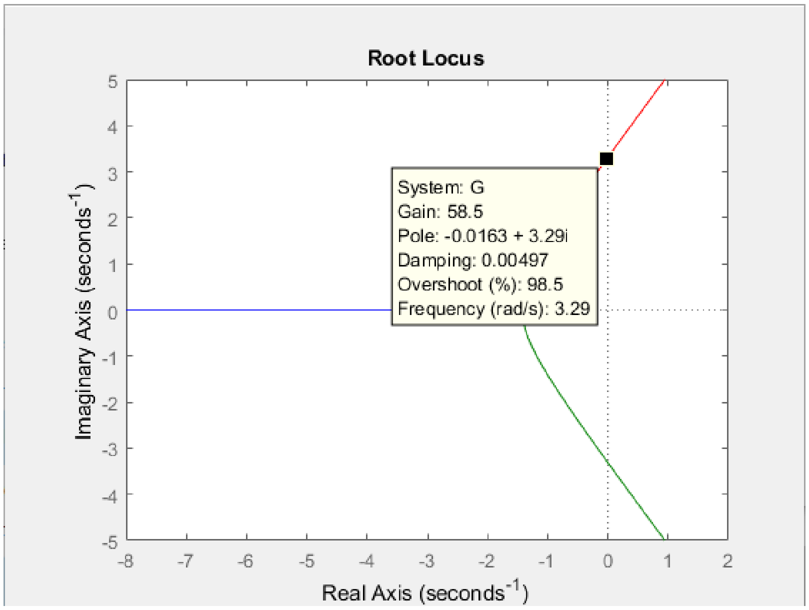
Step Response of System G - Found in Appendix, Matlab Code

c) Step response characteristics of the open loop system, found using stepinfo(G):
Steady state value = 0.167
Steady state error = 1 - 0.167 = 0.833
Rise time = 2.7428 s

Ali Shobeiri - 260665549
Ismail Faruk

Settling time = 5.0039 s
Overshoot = 0

d) Root locus of the open loop system, found using rlocus(G):



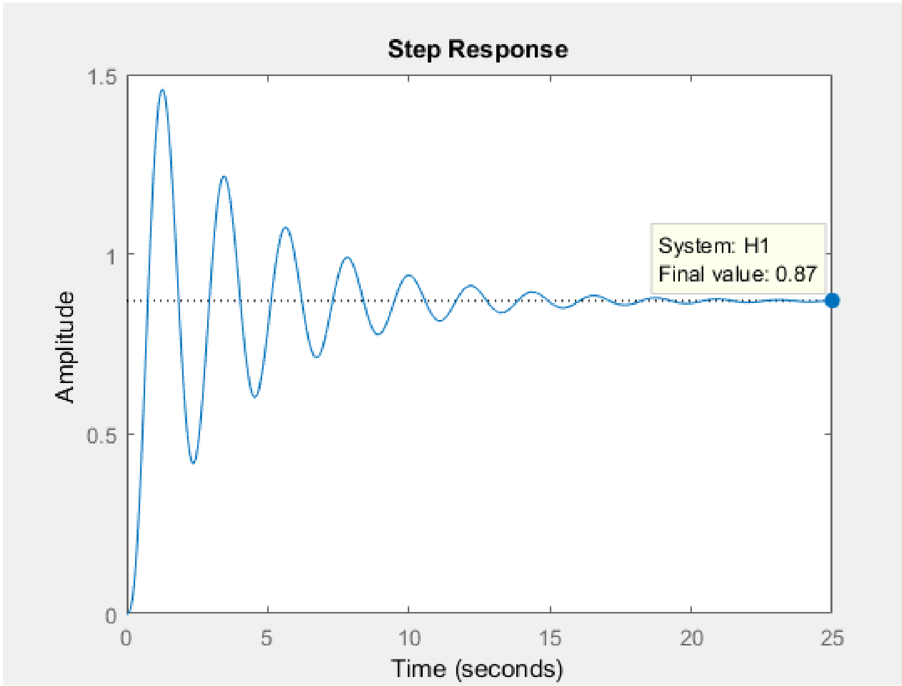
Root Locus of G, shows Effect of Variable K

The point on the root loci is -0.0163 + 3.29j, this can be found by clicking on the root locus plot until the point is on the jw axis.

e) Reading from the plot attached above, the gain at the marginal stability can be seen to be:
Gain: 58.5
Frequency: 3.29 rad/s

Ali Shobeiri - 260665549
Ismail Faruk

f) Step response, with a proportional controller, see section %% Problem 1 P Controller for steps taken to obtain output:



Step Response of System G with a Proportional Controller Added

Using the stepinfo function, the settling time of the system is found to be 15.6105 s.

g) Step response characteristics, with a proportional controller:
Steady state error = 0.13
Rise time = 0.4368 s
Settling time = 15.6105 s
Overshoot = 67.6273

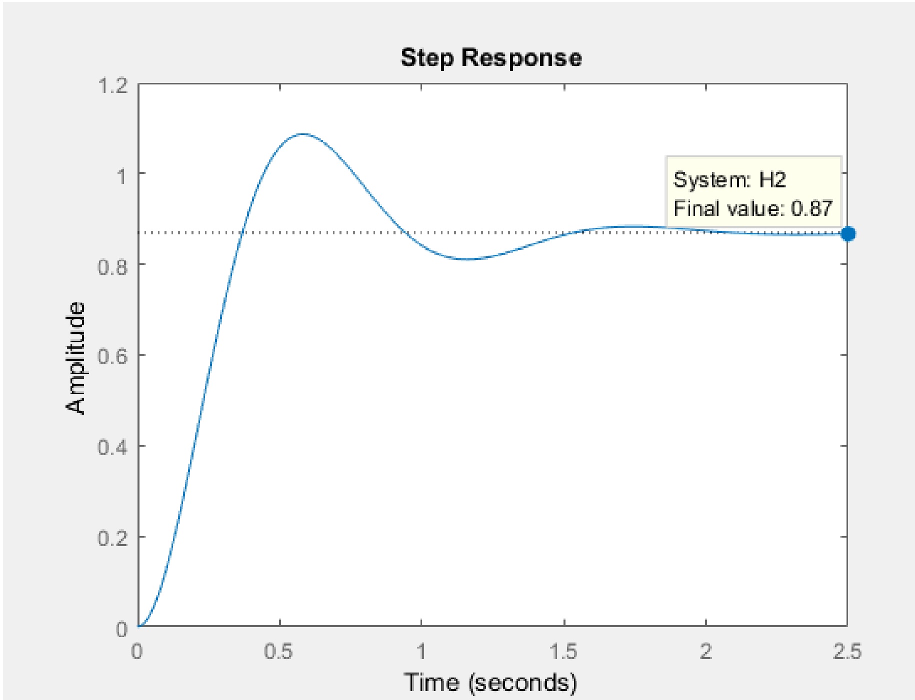
Drawbacks of using the proportional controller:
1. Settling time increased
2. There is a large overshoot vs no overshoot for regular system
3. Added oscillation to the system

Benefits of using the proportional controller:
1. Rise time decreased
2. Steady state error also decreased, although not completely eliminated

When compared to the open loop system, the controlled system has

h) Step response, with a proportional derivative controller

Ali Shobeiri - 260665549
Ismail Faruk



Step Response of System G with a Proportional Derivative Controller Added

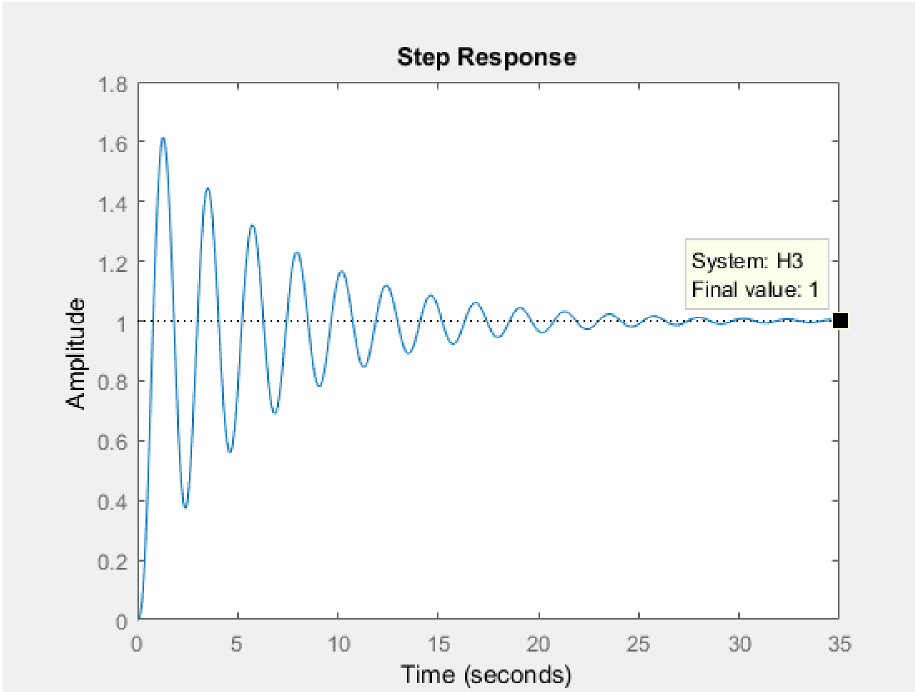
i) Step response characteristics, with a proportional derivative controller
Steady state error = 0.13
Rise time = 0.2494 s
Settling time = 1.4318 s
Overshoot = 24.9544

Possible improvements with a proportional derivative controller:
1. Steady state error is the same as a regular proportional controller yet improved on the no controller case
2. Overshoot is still high, yet lower relative to proportional control
3. Rise time is improved when compared to a proportional control

Benefits of using a proportional derivative controller:
1. Settling time decreased
2. Rise time decreased
3. Overshoot also decreased

j) Step response, with a proportional integral controller

Ali Shobeiri - 260665549
Ismail Faruk



Step Response of System G with Proportional Integral Controller Added

k) Step response characteristics with a proportional integral controller:
Steady state error = 0
Rise time = 0.4368 s
Settling time = 23.6832 s
Overshoot = 61.3913

Possible improvements of using the proportional integral controller:
1. Overshoot is almost the same as proportional controller
2. Settling time is much greater than any other controller setup
3. Rise time is the same as proportional controller

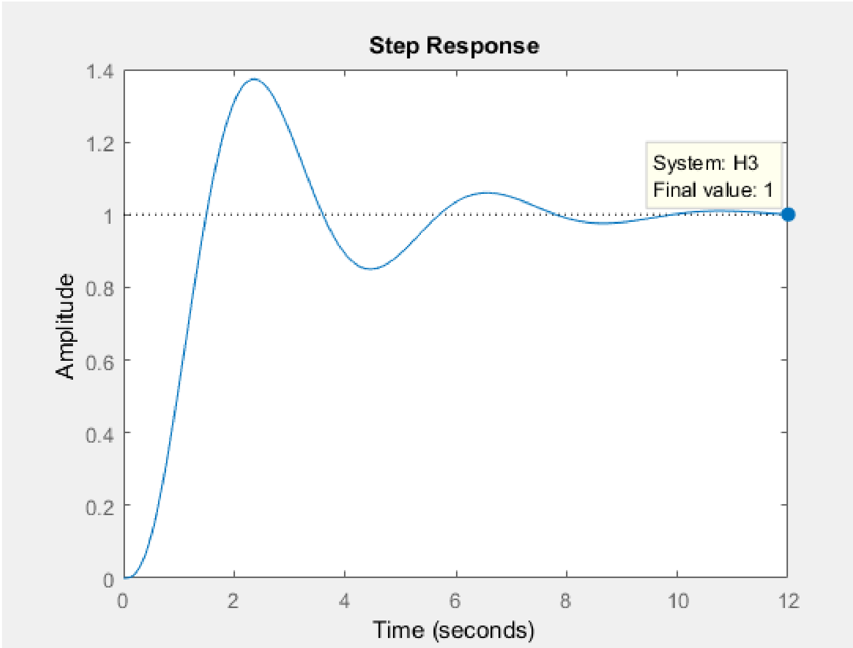
Type	Rise Time (s)	Settling Time (s)	Overshoot	Steady State
No Controller	2.7428	5.0039	0	0.167
P	0.4368	15.6105	67.6273	0.87
PI	0.4368	23.6832	61.3913	1
PD	0.2494	1.4318	24.9544	0.87
PID	0.7352	1.73912	2.9293	1

Benefits of using the proportional integral controller:
1. Steady state error reduced to 0

Ali Shobeiri - 260665549
Ismail Faruk

I) No, we can adjust the value of Kp to see its behavior on the system output.

Step response, with a proportional integral controller, Kp = 10, see appendix %% Problem 1 PI Controller:



Step Response of System G with Reduced Proportional Gain and Integral Controller

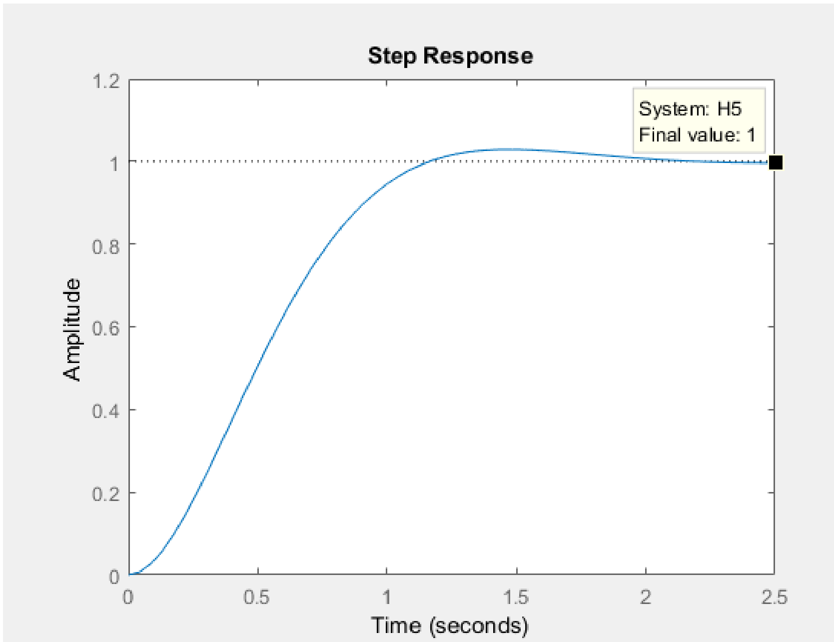
Step response characteristics with a proportional integral controller, Kp = 10:
Steady state error = 0
Rise time = 0.9124 s
Settling time = 9.0786 s
Overshoot = 37.2709

Pros of decreasing kp:
1. Overshoot decreases
2. Settling time decreases

Cons of decreasing kp:
1. Rise time increases

Ali Shobeiri - 260665549
Ismail Faruk

m) Step response with a proportional integral derivative controller, see appendix %% Problem 1 PID Controller:



Step Response of System G with PID Controller Added

Step response characteristics with a proportional integral derivative controller, using stepinfo():
Steady state error = 0
Rise time = 0.7352 s
Settling time = 1.73912 s
Overshoot = 2.9293

Data table of No Controller, P, PI, PD and PID controllers:

Type	Rise Time (s)	Settling Time (s)	Overshoot	Steady State
No Controller	2.7428	5.0039	0	0.167
P	0.4368	15.6105	67.6273	0.87
PI	0.4368	23.6832	61.3913	1
PD	0.2494	1.4318	24.9544	0.87
PID	0.7352	1.73912	2.9293	1

Comparison of PID with other controllers:
1. Although rise time is highest with respect to other controllers, it is still significantly low
2. Settling time is second to PD controllers, with a difference of 0.30732
3. Overshoot is the least compared to all the controllers

Ali Shobeiri - 260665549
Ismail Faruk

4. Steady State error is 0

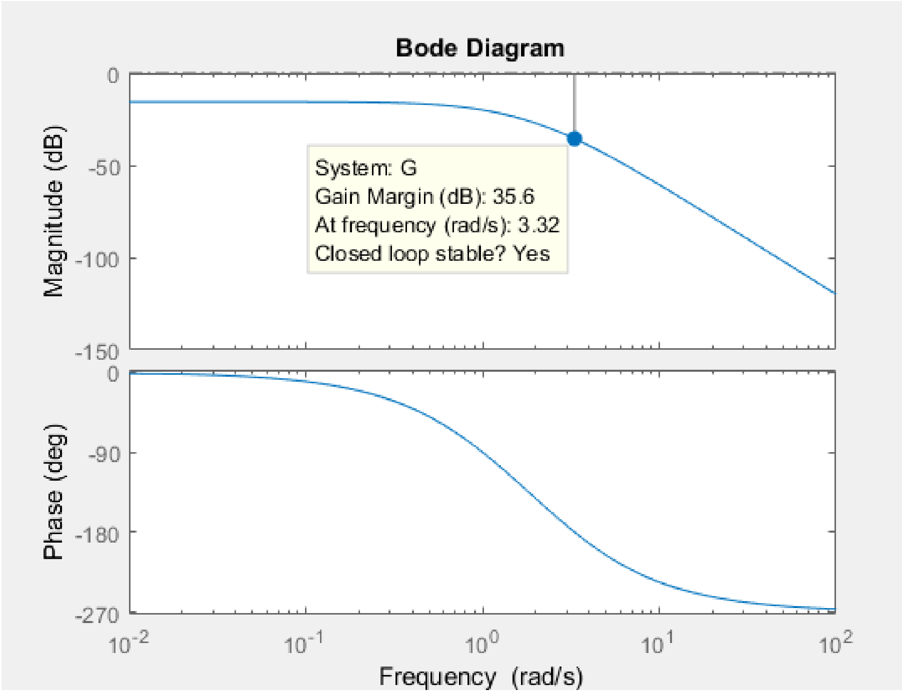
Problem 2

Type	Disturbance Rejection	Reference Tracking	Balanced	Rise Time (s)	Settling Time (s)	Overshoot	Steady State
P	15.2	15.2	15.2	0.7154	5.3911	30.2416	0.717
I	1.84	1.84	1.84	4.0235	12.7324	7.4126	1
PI	Kp = 7.59 Ki = 5.03	Kp = 7.59 Ki = 5.03	Kp = 7.59 Ki = 5.03	1.3383	6.0126	7.7325	1
PD	Kp = 29.9 Kd = 14.8	Kp = 29.9 Kd = 14.8	Kp = 29.9 Kd = 14.8	0.3756	1.9573	18.5846	0.833
PID	Kp = 12.5 Ki = 9.91 Kd = 3.97	Kp = 12.5 Ki = 9.91 Kd = 3.97	Kp = 12.5 Ki = 9.91 Kd = 3.97	1.0665	2.8508	4.5359	1

Problem 3

a) Bode plot of the open loop system

Ali Shobeiri - 260665549
Ismail Faruk



Step Response of System G - Found in Appendix, Matlab Code

b) Ultimate gain is found by using the
Ku = 10 ^ (35.6/20) = 60.26
Pu = 2*pi / (3.32) = 1.89

c) The point on the root loci in the first question is -0.0163 + 3.29j. There we got the following values:
Ku = 58.5
Pu = 2*pi / 3.29 = 1.91

The ultimate Gain is lower and the ultimate period is higher in the first question than that of this section, respectively.

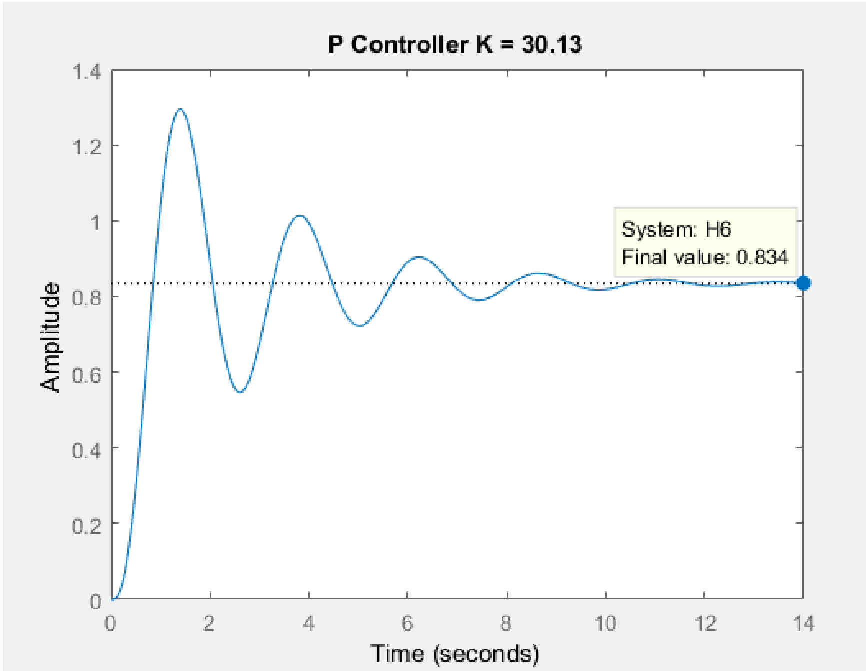
d) Controller Values:
Using the values of Table 3 from lab instructions

Type	Optimum Gain
P	Kp = 30.13
PI	Kp = 27.12, Ki = 0.62827
PID	Kp = 36.16, Ki =1.047, Kd = 0.2388

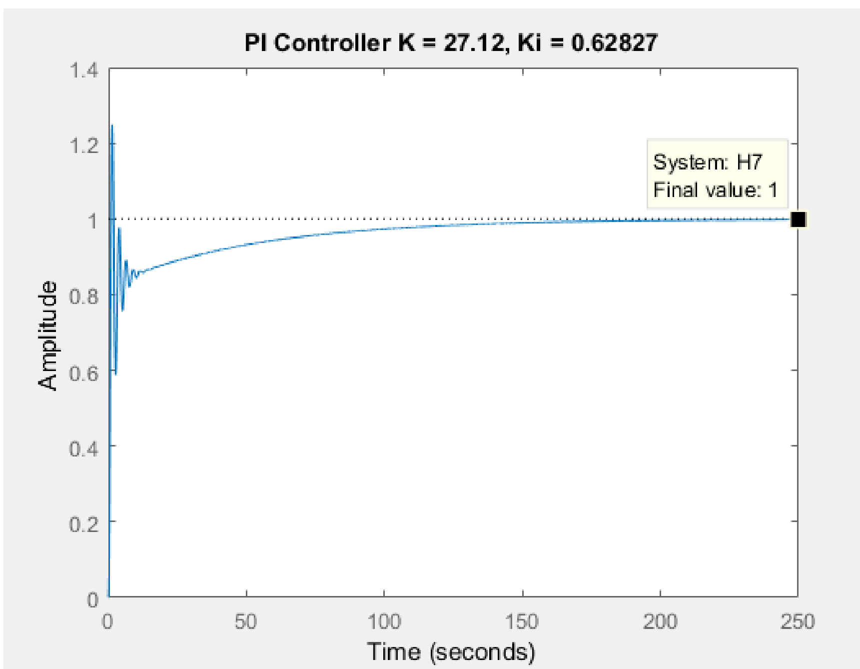
Step response with PI controller:

ECSE 307 - Lab 4

Ali Shobeiri - 260665549
Ismail Faruk

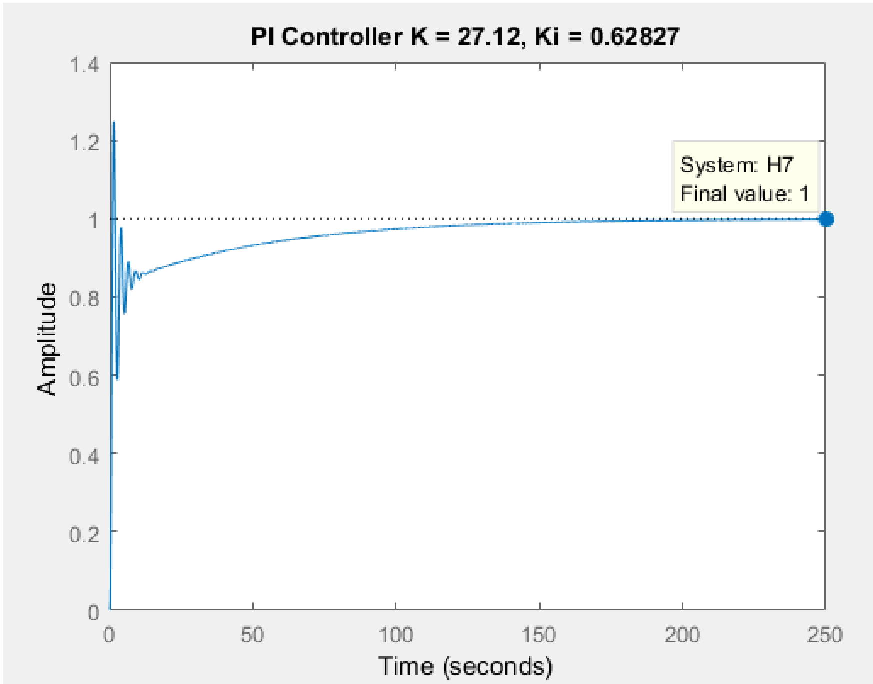


Step response with PI controller



Ali Shobeiri - 260665549
Ismail Faruk

Step response with PID controller



Step response characteristics of P, PI, PID controllers:

Type	Rise Time (s)	Settling Time (s)	Overshoot	Steady State
P	0.5031	9.9155	55.1110	0.834
PI	0.6209	114.2505	24.9551	1.00
PID	0.5127	77.1730	40.3691	1.00

It is clear to see that the Ziegler Nichols method is an unreliable method to determine an efficient controller schema. The Ziegler Nichols method is primarily used to determine a set of controller characteristics that could be used as a good starting point to perform trial and error analysis.

The Ziegler method returns a set of values that guarantee the system is not unstable, yet may not return the most optimal or efficient result.

Ali Shobeiri - 260665549
Ismail Faruk

Conclusion:

It is demonstrated in the lab that a P controller is useful for reducing steady state error and also reduces the rise time, but it ends up increasing the overshoot and settling time. A PI controller reduces the overshoot by a significant amount and reduces the steady state error to zero, while maintaining a low rise time, but it increases the settling time. A PID controller helps reduce the overshoot and settling time, while maintaining a low rise time and a zero steady error.

Tuning a controller algorithmically is much more reliable than when using the Ziegler Nichols method. The Ziegler Nichols method is a good approximation to give controller values that will return a stable system, yet it’s properties are far too general and are not practical in use for designing controllers for specific tasks.

Appendix

Gain Table

Ultimate Gain, Problem 1. d)	Ultimate Gain, Problem 3. b)
58.5	60.26

Code

```
%% Problem 1 a - f
s = tf('s');
G = 1/((s+1)*(s+2)*(s+3));
step(G);
figure;
rlocus(G);
stepinfo(G)
%% Problem 1 P Controller
C_P = pid(40);
open_loop = series(C_P, G);
H1 = feedback(open_loop,1);
hold on;
figure;
```

Ali Shobeiri - 260665549
Ismail Faruk

```
step(H1);
stepinfo(H1)
%% Problem 1 PD Controller
C_PD = pid(40,0,30);
open_loop_PD = series(C_PD, G);
H2 = feedback(open_loop_PD,1);
hold on;
figure;
step(H2);
stepinfo(H2)
%% Problem 1 PI Controller
C_PI = pid(10,10,0);
open_loop_PI = series(C_PI, G);
H3 = feedback(open_loop_PI,1);
hold on;
figure;
step(H3);
stepinfo(H3)
%% Problem 1 PID Controller
C_P1 = pid(19,12,8);
open_loop_PID = series(C_P1, G);
H5 = feedback(open_loop_PID,1);
hold on;
figure;
step(H5);
stepinfo(H5)
%% Problem 2 PID Controller Tuning
% opts = pidtuneOptions('DesignFocus','disturbance-rejection');
opts = pidtuneOptions('DesignFocus','reference-tracking');
% opts = pidtuneOptions('DesignFocus','balanced');

% type = 'P';
% type = 'I';
% type = 'PI';
% type = 'PD';
type = 'PID';
C_auto = pidtune(G,type,opts);
open_loop_auto = series(C_auto, G);
H_auto = feedback(open_loop_auto,1);
hold on;
figure;
step(H_auto);
stepinfo(H_auto)

%% Problem 3
bode(G)

%% Problem 3 Zeigler
  %% P Controller
```

Ali Shobeiri - 260665549
Ismail Faruk

```
C_P1 = pid(30.13, 0, 0);
open_loop_PID = series(C_P1, G);
H6 = feedback(open_loop_PID,1);
step(H6);
title('P Controller K = 30.13');
stepinfo(H6)
   %% PI Controller
C_PI1 = pid(27.12,  0.62827, 0);
open_loop_PID = series(C_PI1, G);
H7 = feedback(open_loop_PID,1);
step(H7);
title('PI Controller K = 27.12, Ki = 0.62827');
stepinfo(H7)


   %% PID Controller
C_PID1 = pid(36.16, 1.047, 0.2388);
open_loop_PID = series(C_PID1, G);
H8 = feedback(open_loop_PID,1);
step(H8);
title('PID Controller K = 36.16, Ki = 1.047, Kd = 0.2388');
stepinfo(H8)
```