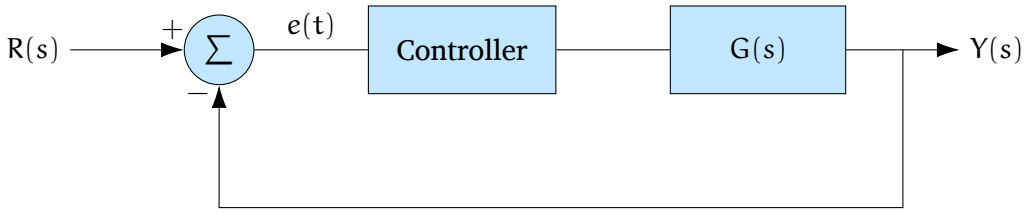
Lab 4: PID Controller Design

1 Objectives

The main objective of this laboratory is to get a basic understanding for how feedback control can be used to modify the behaviour of a dynamic system. In particular, we will consider P, PI, PD, and PID-control of a third order system. We will provide some experience of manually tuning the controllers as well as using MATLAB pre-defined tuning functions and Ziegler Nichols method to find the proper controller.

2 Introduction

The letters PID stand for Proportional, Integral and Derivative control, respectively. PID controllers are by far the most common type of controllers used in industrial systems, mainly because they are relatively simple and still often able to provide good performance. The experiments in this laboratory will confirm this. Consider the following unity feedback system.



The error signal $e(t)$ which is the difference between the input and the feedback, will be sent to the controller. The controller uses this error and provides the control signal to the plant. The relation between the control signal and the error signal depends on the transfer function and characteristics of the controller. In the following, we briefly have a look at different combinations of PID controllers.

2.1 Proportional Control

When the feedback control signal is linearly proportional to the system error, the result is proportional feedback.

$$u(t) = k_P e(t) \quad \text{or} \quad U(s) = k_P E(s).$$

When we are using this controller, increasing the proportional gain decreases the tracking error.

2.2 Integral Control

When a feedback control signal is linearly proportional to the integral of the system error, we call the result integral feedback. The goal of integral control is to minimize the steady state tracking error and the output response to disturbance. This control law is of the form

$$u(t) = k_I \int_0^t e(\tau)d\tau \quad \text{or} \quad U(s) = \frac{k_I}{s} E(s).$$

The integral controller, results in zero steady state tracking error to step input and zero steady state output to constant disturbance no matter what the value of k_I is. On the other hand, a sufficiently high integral gain k_I may make the system unstable.

2.3 Derivative Control

The final term in the classical controller is the derivative controller, which also called *rate feedback*. The goal of derivative controller is to improve closed loop system stability as well as speeding up the transient response and reducing overshoot. In derivative feedback the control law is

$$u(t) = k_D \frac{d}{dt} e(t) \quad \text{or} \quad U(s) = k_D s E(s).$$

Derivative controller is almost never used by itself; it is usually augmented by proportional control. The reason is when $e(t)$ remains constant, the output of a derivative controller would be zero and a proportional or integral controller would be needed to provide a control signal at this time. A key feature of derivative control is that it has an anticipatory behaviour. This means that the derivative controller knows the slope of the error signal, so it takes control action based on the trend in the error signal. One disadvantage of derivative control is that it tends to amplify the noise.

2.4 Proportional Plus Integral Control

Adding an integral term to the proportional controller to achieve the lower steady state errors results in the proportional plus integral (PI) controller:

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau \quad \text{or} \quad U(s) = k_P + \frac{k_I}{s} E(s).$$

Most controllers implemented in practice, if they have an integral term, will also have a proportional term. This combination generally allows for a faster response than a pure integral control alone. On the other hand, the integral term rejects constant disturbances in the system.

2.5 PID Control

Putting all the three terms together results in:

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{d}{dt} e(t) \quad \text{or} \quad U(s) = k_P + \frac{k_I}{s} E(s) + k_D s.$$

If the system is second order or higher, the use of PID controller is required if we wish to have arbitrary dynamics. Note that, in a second order system, by selecting these three parameters the roots can be uniquely determined.

2.6 Effects of PID Parameters

- A proportional controller (k_p) will have the effect of reducing the rise time and will reduce but never eliminates the steady-state error.
- An integral controller (k_I) will have the effect of eliminating the steady-state error for a constant or step input, but it may make the transient response slower.
- A derivative control (k_D) will have the effect of increasing the stability of the system, reducing the overshoot, and improving the transient response.

The effects of each of controller parameters, k_P, k_I, and k_D on a closed-loop system are summarized in Table 1.

Table 1 Effects of PID Parameters on a closed loop system

Parameter	Rise Time	Overshoot	Settling Time	e_{ss}
k_P	Decrease	Increase	Small Change	Decrease
k_I	Decrease	Increase	Increase	Eliminate
k_D	Small Change	Decrease	Decrease	No Change

Note that these correlations may not be exactly accurate, because the parameters are dependent on each other. Therefore, the table should only be used as a reference when you are determining the values for each of the parameters.

3 Finding the PID Gains- A Trial and Error Approach

We investigate this approach with an example.

3.1 Problem 1

Consider the following system transfer function.

$$G(s) = \frac{1}{(s+1)(s+2)(s+3)} \tag{1}$$

a. Draw the step response of the system.

b. Find the step response information of this system using the following function.

```
stepinfo(G);  % Gives the step response characteristics.
```

c. What are the values for the steady state error, rise time, settling time and overshoot?

d. Plot the root locus of the open loop system using

```
rlocus(G); % Plots the root loci of the system G
```

e. By clicking on the root loci try to find the gain and frequency at the marginal stability. The marginal stability is the point on the root loci where at least one of the poles are on the $j\omega$ axis entering the right half plane.

f. This system is not fast enough (look at the step response). We want to design a controller that reduces the rise time, the settling time, and eliminates the steady-state error. Using Table 1, we see that the proportional controller reduces the rise time, increases the overshoot, and reduces the steady-state error. Add the proportional controller. Using the following code, find the step response characteristics for $k_P = 40$. Plot the step response.

```
C_P = pid(40);
open_loop = series(C_P, G);
H1 = feedback(open_loop,1);
hold on;
figure;
step(H1);
stepinfo(H1);
```

In the first line of the above code, `pid()` creates PID controller transfer function. For more information about `pid()` use

```
doc pid
```

g. Compare the step response characteristics of the controlled system with the open loop system. In your opinion, what are the drawbacks of using proportional controller? What are the benefits of using the proportional controller?

h. Using Table 1, one can observe that the derivative controller reduces the overshoot and the settling time. Keep the proportional controller and add the derivative controller. Using the following code, find the step response characteristics for $k_P = 40$ and $k_D = 30$. Plot the step response.

```
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);
```

i. Compare the step response characteristics of the closed loop system in this part with the P controller. In your opinion, what characteristics can be improved? What are the benefits of adding derivative controller to the system?

j. Using Table 1, we see that the integral controller decreases the rise time, increases both the overshoot and the settling time, and eliminates the steady-state error. Keep the proportional controller and add the integral controller. Using the following code, find the step response characteristics for $k_P = 40$ and $k_I = 10$. Plot the step response.

```
C_PI = pid(40,10,0);
open_loop_PI = series(C_PI, G);
H3 = feedback(open_loop_PI,1);
hold on;
figure;
step(H3);
stepinfo(H3);
```

k. Compare the step response characteristics of the closed loop system in this part with the P controller. In your opinion, what characteristics can be improved? What are the benefits of adding integral controller to the system?

l. In the last part, do we need to keep $k_P = 40$? Change k_P to 10 and observe the step response characteristics. Plot the step response. What are the pros and cones of decreasing k_P?

m. Now we want to use P, I, and D together. Use the following code and find the step response characteristics for $k_P = 19$, $k_D = 8$ and $k_I = 12$. Plot the step response. Compare the results with the previous parts.

```
C_PID = pid(19,12,8);
open_loop_PID = series(C_PID, G);
H5 = feedback(open_loop_PID,1);
hold on;
figure;
step(H5);
stepinfo(H5);
```

4 Automatic PID Tuning with MATLAB

MATLAB provides tools for automatically choosing optimal PID gains which makes the trial and error process described above unnecessary. The MATLAB automated tuning algorithm chooses PID gains to balance performance (response time, bandwidth) and robustness (stability margins). By default the algorithm designs for a 60 degree phase margin. It is possible to change the options of MATLAB algorithm with `pidtuneOptions()` command. You can tune the PID controller with specific options using the following functions.

```
opts = pidtuneOptions('DesignFocus',Value) % Define options for pidtune command.
C = pidtune(sys,type,opts) % Designs a 'type' PID controller for the plant 'sys'
                           % with respect to options 'opts'.
```

In `pidtune()`, `type` determines the controller type to design and can be `'P'`, `'I'`, `'PI'`, `'PD'`, `'PID'`.

The `opts` determines the additional tuning options for the `pidtune()` design algorithm. In this laboratory, we will focus on tuning the `'DesignFocus'`, which is set using

```
opts = pidtuneOptions('DesignFocus', Value);
```

where `Value` can be

- `'balanced'` (default): Tunes the controller to balance reference tracking and disturbance rejection.
- `'reference-tracking'`: Tunes the controller to favour reference tracking, if possible.
- `'disturbance-rejection'`: Tunes the controller to favour disturbance rejection, if possible.

4.1 Problem 2

Now consider $G(s)$ in Equation 1. For the unit feedback controlled system, find the step response characteristic and plot the step response using the following code.

```
% CHOOSE ONE OF THE OPTIONS BELOW
% opts = pidtuneOptions('DesignFocus','disturbance-rejection');
% opts = pidtuneOptions('DesignFocus','reference-tracking');
% opts = pidtuneOptions('DesignFocus','balanced');

% CHOOSE ONE OF THE OPTIONS BELOW
% 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);
```

- Enter the values of k_P, k_I, and k_D in Table 2 for each combination of the above options.
- Plot the step response and write the step response characteristics (rise time, settling time, ...) for each combination of the above ddoptions.

Table 2 PID Controller Tuning using MATLAB

Type	Disturbance Rejection	Reference Tracking	Balanced
P			
I			
PI			
PD			
PID			

For more information about how to use these functions use:

```
doc pidtune
doc pidtuneOptions
```

Matlab also provides a GUI for PID controller design. Try the following function in the command window.

```
pidTuner
```

In the above GUI you can find PID coefficients for a given transfer function as well as a given set of input/output data.

5 PID Tuning with Ziegler Nichols Method—Ultimate Sensitivity Approach

Ziegler and Nichols gave two methods for tuning the PID controller of a model. The step response method and the ultimate sensitivity method.

In the ultimate sensitivity method the criteria for adjusting the parameters are based on evaluating the amplitude and frequency of the oscillation of the system at the limit of stability. To use this method, a proportional controller is used. The proportional gain is then increased until the system becomes marginally stable. The corresponding gain is defined as K_u (called the ultimate gain) and the corresponding period is defined as the period of oscillation P_u (called the ultimate period). Experiences has shown that the controller setting according to Ziegler Nichols rules provide acceptable closed loop response. The tuning parameters are determined as shown in Table 3.

Table 3 Ziegler Nichols Tuning for PID controller based on the Ultimate Sensitivity Method

Type of Controller	Optimum Gain
P	$k_P = 0.5K_u$
PI	$k_p = 0.45K_u$, $k_I = \frac{1.2}{P_u}$
PID	$k_p = 0.6K_u$, $k_I = \frac{1}{0.5P_u}$, $k_D = 0.125P_u$

5.1 Problem 3

Consider the transfer function in Equation 1. We want to find the P, PI, and PID controller gains using the ultimate sensitivity method.

a. First, we have to find the ultimate gain and period. This is possible in Matlab by both bode plot and root locus of the open loop system. Follow these steps:

   1. Plot the Bode of the open loop system.
   2. Right click on the Bode plot, select `characteristic >> Minimum stability margins`. The value of the gain margin (the minimum gain for which the system is marginally stable) and its frequency will show up on the bode plot.

At this step, we have the marginal gain of the system and its angular frequency. Now, we want to find K_u and P_u. The relation between K_u and gain margin (G_m) and P_u and the angular frequency (f_{G_m}) is as follows.

$$G_m = 20\log_{10} K_u, \quad P_u = \frac{2\pi}{f_{G_m}}$$

b. Find the value of K_u and P_u.

c. Compare the values of K_u and P_u, which you found in this section, with the values you found in the first section using the root locus.

d. Use Table 3 and find the P, PI, PID controller coefficients. Plot the step response of the closed loop controlled system for each of P,PI, and PID controllers. Using stepinfo();, find the step response characteristics of the closed loop system again for P, PI, PID.

6 Assignment

In a report format, answer all of the questions. In the report include:

- An introduction about the purpose of the lab.
- A summary of the effect of P, I, and D controllers.
- The answers to the questions.
- A table containing all of the gains you found during this lab for each section.
- A table containing rise time, settling time, overshoot, peak time, and steady state error for all of the cases in each section.
- Explanation of the steps to answer the laboratory questions.
- A conclusion about what you have learned.

All figures should have a legend and a caption. Include your code in the report appendix.

The assignment is due 7 days after your lab.