Christopher Lum
lum@uw.edu

Lecture 07b
Introduction to PID Control

Lecture is on YouTube

The YouTube video entitled ‘Introduction to PID Control’ that covers this lecture is located at https://youtu.be/_VzHpLjKeZ8

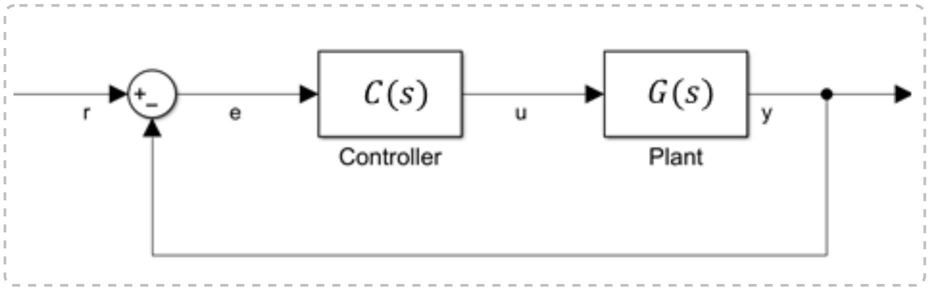
Outline

-Introduction
-PID Control Law
- -Proportional Component
- -Integral Component
- -Derivative Component
- -General Trends

-Example
-Other Videos

Introduction

Consider the standard control architecture we discussed in the ‘Introduction to Feedback Control’ video



One of the simplest control schemes is to simply use

$$C(s) = K$$

where K = static gain

So the control law is therefore

$$u(t) = K\,e(t)$$

This is known as a proportional feedback control system because the control signal applied is directly proportional to the error.

The PID controller is an extension of this idea and is one of the most popular controllers in practice.

PID Control Law

We can generate a controller which maps the error to the control signal using the relationship

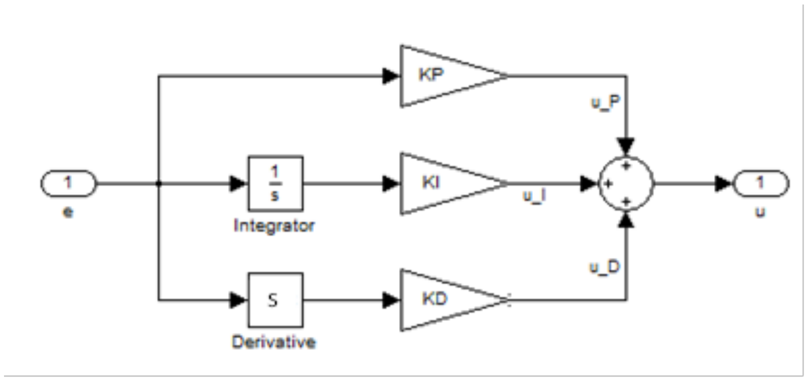
$$u(t) = u_P(t) + u_I(t) + u_D(t) \qquad \textbf{(Eq.1)}$$

where
$$u_P(t) = K_P\,e(t)$$
$$u_I(t) = K_I \int_0^t e(\tau)\,d\tau$$
$$u_D(t) = K_D \frac{d\,e(t)}{d\,t}$$

This is known as a proportional, integral, derivative (PID) controller. The name should be clear because the control signal is made up of three components

$u_P(t)$ is proportional to the error signal
$u_I(t)$ is proportional to the integral of the error signal
$u_D(t)$ is proportional to the derivative of the error signal

A block diagram of this controller as shown below

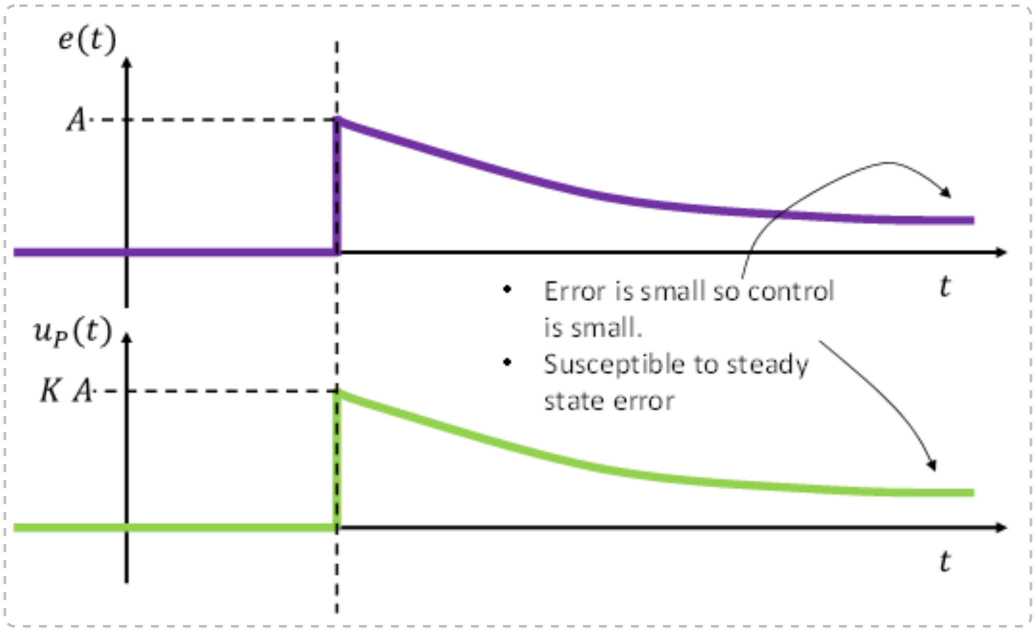


Let us examine each component individually to obtain an understanding of their behavior

Proportional Component

Let us consider the proportional term by itself. In other words, consider $K_I = K_D = 0$

This is the simplest term to understand as the output is simply proportional to the current error.



Some people think of the proportional control as thinking about the present (AKA what is occurring right now)

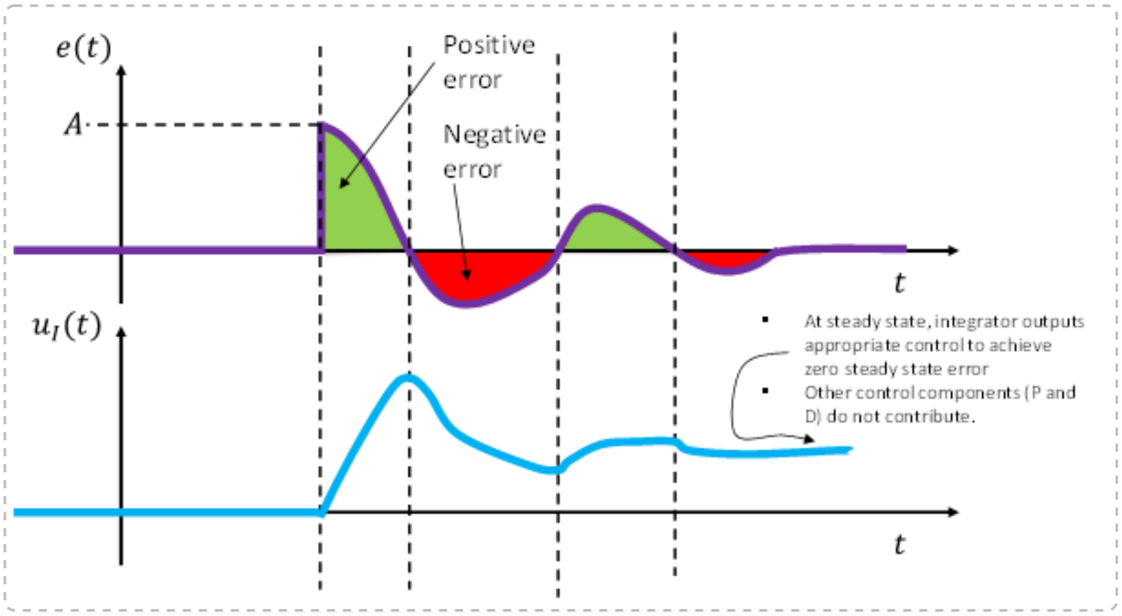
Integral Component

Let us consider the integral term by itself. In other words, consider $K_P = K_D = 0$

An integrator is typically added to eliminate steady state error. This can be seen by considering a paradox. If the system has steady state error, then $u_I(t) = K_I \int_0^t e(\tau)\, d\tau$ goes to ±infinity which cannot occur (draw picture).

-Talk about how integrator state does not stop when error is 0. This is the point when the state is maximum (so it is exerting the maximum amount of control). This leads to overshoot. Some people think of this as "momentum" which may be interesting to think about later when we look at a mass/spring/damper analogy.

In general, the behavior of the integrator is as shown below


$e(t)$
A
Positive error
Negative error
t
$u_I(t)$
At steady state, integrator outputs appropriate control to achieve zero steady state error
Other control components (P and D) do not contribute.
t

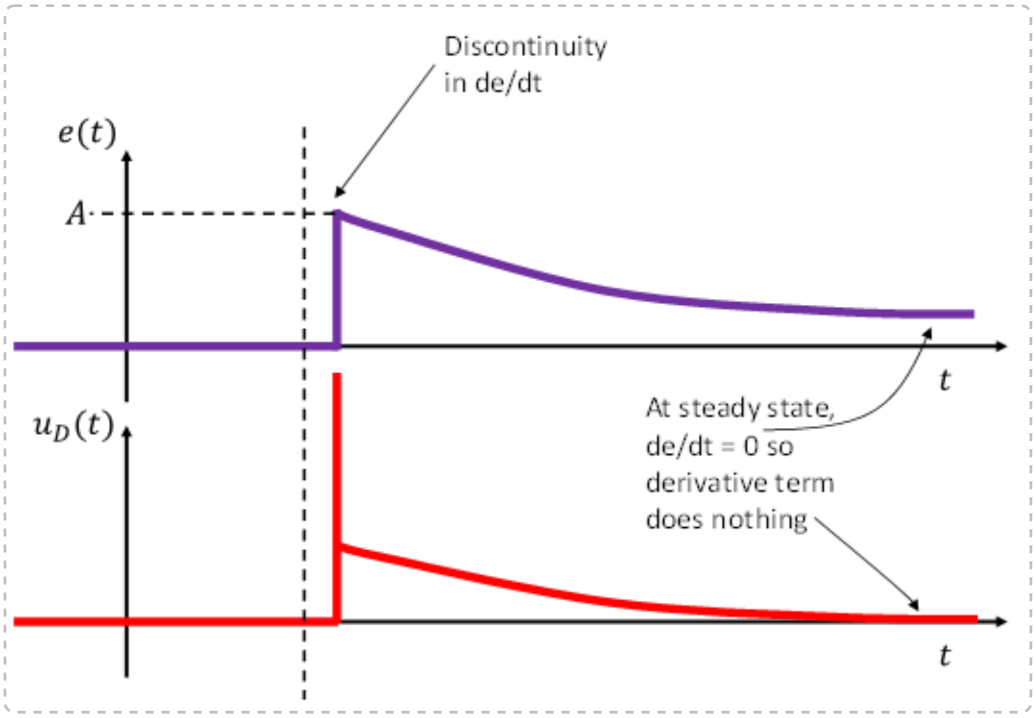
Due to the tendency of the integrator to oscillate, it is very rarely used by itself (it is almost always combined with a P or PD component)

We will look at this in much greater detail in ‘Practical Implementation Issues with a PID Controller’ https://youtu.be/yr6om0e0oAQ

Some people think of the integral control as thinking about the past (AKA what has happened in the past)

Derivative Component

We finally consider the derivative term. As we will see later, the derivative term cannot operate effectively without the other terms so we consider it in conjunction with other terms.

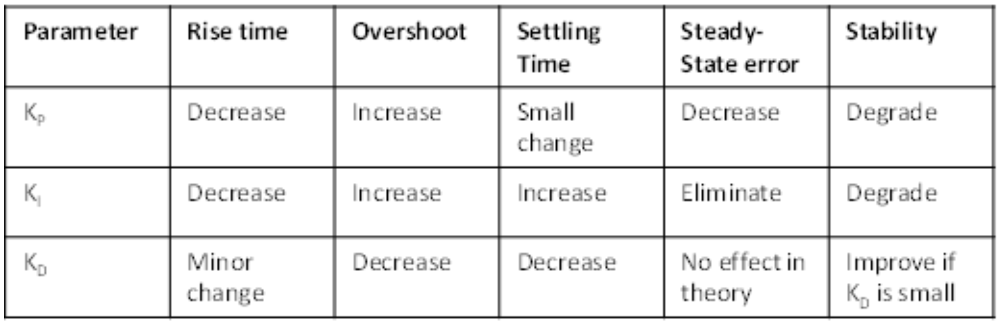


Some people think of the derivative control as thinking about the future (AKA what the system is going to do in the future).

General Trends

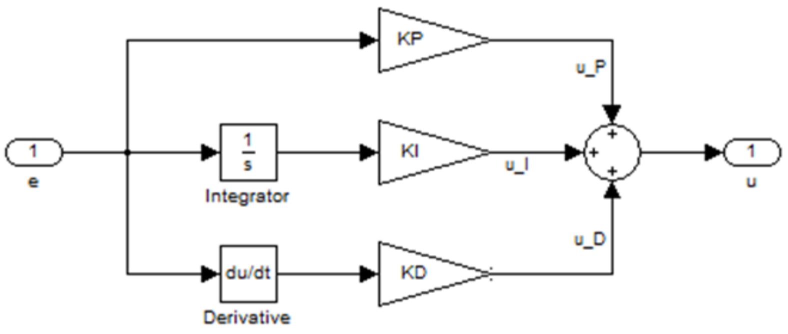
In general, the effects of increasing a component independently is summarized below

General effects of increasing a parameter independently



Parameter	Rise time	Overshoot	Settling Time	Steady-State error	Stability
K_P	Decrease	Increase	Small change	Decrease	Degrade
K_I	Decrease	Increase	Increase	Eliminate	Degrade
K_D	Minor change	Decrease	Decrease	No effect in theory	Improve if K_D is small

This type of controller can be implemented easily in Simulink (note the use of a numerical derivative rather than a pure *s*, we will come back to this later when we discuss practical implementation issues for PID controllers)



There are some issues with this implementation that we will revisit later in the class. Despite this limitation, let us use this form of controller to help design K_P, K_I, and K_D.

Example

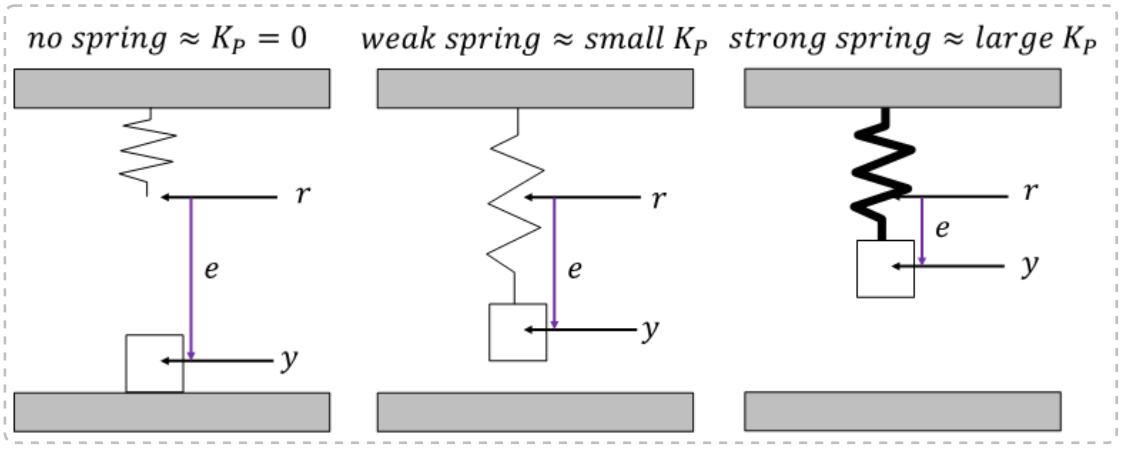
Proportional Control

The proportional control is

$$u_P(t) = K_P\, e(t)$$

In essence, we apply control that is proportional to the error. This is how a spring operates

$$F_s(t) = k\, e(t)$$



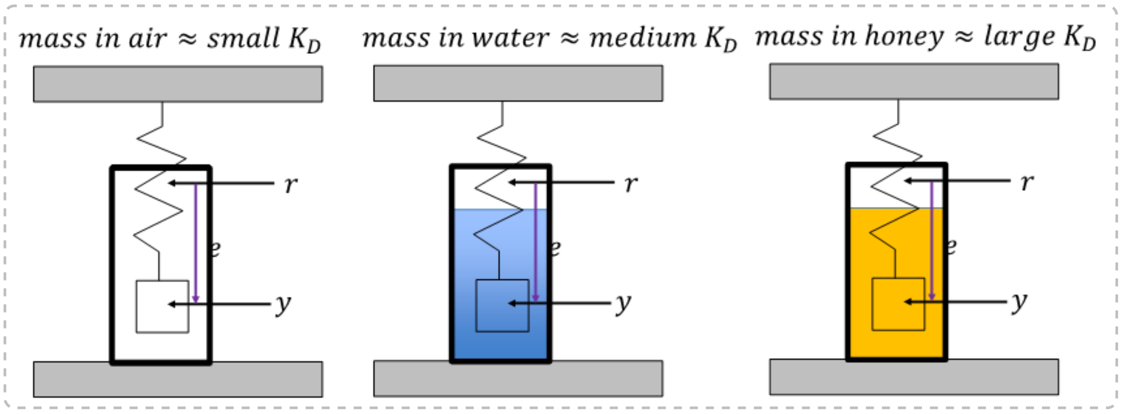
Derivative Control

The derivative control is

$$u_D(t) = K_D \frac{d\,e(t)}{d\,t}$$

In essence, we apply control that is proportional to the time rate of change of error. This is how a damper operates

$$F_D = c\,\dot{e}(t)$$


mass in air ≈ small K_D
mass in water ≈ medium K_D
mass in honey ≈ large K_D
r
e
y

Integral Control

The integral control is

$$u_I(t) = K_I \int_0^t e(\tau)\, d\tau$$

Show how this operates

Other Videos

-Practical Implementation Issues with a PID Controller

-Designing a PID Controller Using the Ziegler-Nichols Method

-Designing a PID Controller Using the Root Locus Method