Introduction, The PID Controller, State Space Models

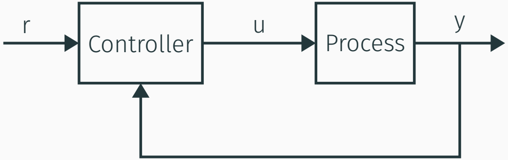
Automatic Control, Basic Course, Lecture 1

November 7, 2017

Lund University, Department of Automatic Control

Content

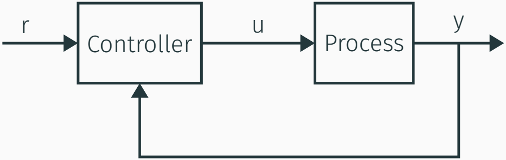
Introduction



- Reference value **r**
- Control signal **u**
- Measured signal/output **y**

The problem/purpose: Design a controller such that the output follows the reference signal as good as possible

The Simple Feedback Loop

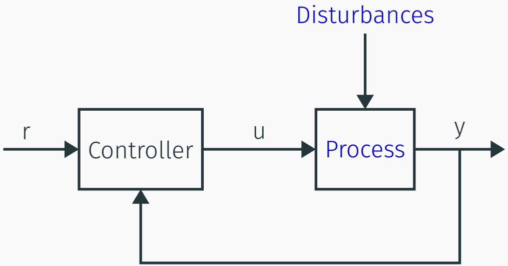


- Reference value **r**
- Control signal **u**
- Measured signal/output **y**

The problem/purpose: Design a controller such that the output follows the reference signal as good as possible

Note on terminology: Process, Controlled system, Plant etc...

The Feedback Loop



- Reference value **r**
- Control signal **u**
- Measured signal/output **y**

The problem/purpose: Design a controller such that the output follows the reference signal as good as possible despite disturbances and uncertainties in process.

Find the Control Problem - 1

Find the Control Problem - 1

- Reference value - Desired temperature
- Control signal - E.g., power to the AC, amount of hot water to the radiators
- Measured value - The temperature in the room


CANCEL
+ RES
− SET
ON-OFF
MPH
0
20
40
60
80
100
120
140

Find the Control Problem - 2

- Reference value - Desired speed
- Control signal - Amount of gasoline to the engine
- Measured value - The speed of the car

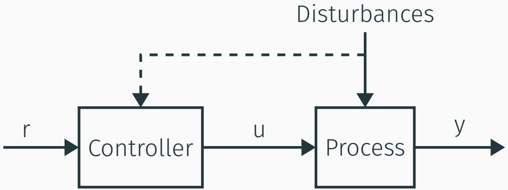
Find the Control Problem - 3

Find the Control Problem - 3

- Reference value - Number of bacterias
- Control signal - “Food” (sugar and O_2)
- Measured value - E.g., pH or oxygen level in the tank

Feedforward

Some systems can operate well without feedback, i.e., in open loop.



Examples of open loop systems?

Feedforward vs. Feedback

Benefits with feedback:

- Stabilize unstable systems
- The speed of the system can be increased
- Less accurate model of the process is needed
- Disturbances can be compensated
- WARNING: Stable systems might become unstable with feedback

Feedforward vs. Feedback

Benefits with feedback:

- Stabilize unstable systems
- The speed of the system can be increased
- Less accurate model of the process is needed
- Disturbances can be compensated
- **WARNING:** Stable systems might become unstable with feedback

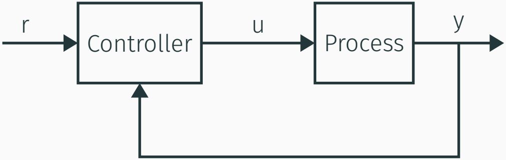
Feedforward and feedback are **complementary** approaches, and a good controller typically **uses both**.

The PID Controller

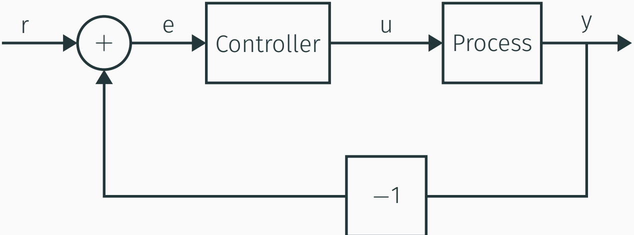
The Error

The input to the controller will be the error, i.e., the difference between the reference value and the measured value.

$$e = r - y$$

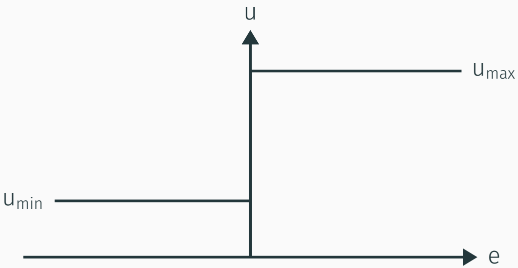


New block scheme:



On/Off Controller

$$u = \begin{cases} u_{max} & \text{if } e > 0 \\ u_{min} & \text{if } e < 0 \end{cases}$$



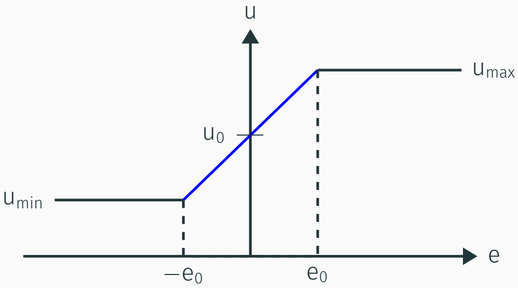
Usually not a good controller. Why?

The P Part

Idea: Decrease the controller gain for small control errors.

P-controller:

$$u = \begin{cases} u_{max} & \text{if } e > e_0 \\ u_0 + Ke & \text{if } -e_0 \leq e \leq e_0 \\ u_{min} & \text{if } e < -e_0 \end{cases}$$



P-part comes from proportional (here affine) to the error e.

The P Part

Idea: Decrease the controller gain for small control errors.

P-controller:

$$u = \begin{cases} u_{max} & \text{if } e > e_0 \\ u_0 + Ke & \text{if } -e_0 \leq e \leq e_0 \\ u_{min} & \text{if } e < -e_0 \end{cases}$$

The control error

$$e = \frac{u - u_0}{K}$$

To have $e = 0$ at stationarity, either:

- $u_0 = u$
- $K = \infty$

The P Part

Idea: Decrease the controller gain for small control errors.

P-controller:

$$u = \begin{cases} u_{max} & \text{if } e > e_0 \\ u_0 + Ke & \text{if } -e_0 \leq e \leq e_0 \\ u_{min} & \text{if } e < -e_0 \end{cases}$$

The control error

$$e = \frac{u - u_0}{K}$$

To have $e = 0$ at stationarity, either:

- $u_0 = u$ (What if u varies?)
- $K = \infty$ (On/off control)

The I Part

Idea: Adjust u_0 automatically to become u.

PI-controller:

$$u(t) = K\left(\frac{1}{T_i}\int^t e(\tau)\mathrm{d}\tau + e\right)$$

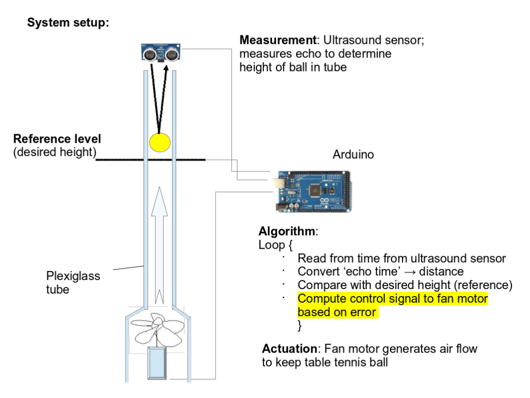
Compared to the P-controller, now

$$u_0(t) = \frac{K}{T_i}\int^t e(\tau)\mathrm{d}\tau$$

At stationary $e = 0$ if and only if $r = y$.

PI controller achieves what we want, if performance requirements are not extensive.

Example of integral action needed — mini-problem (5 min)



(a) Argue why there will be a stationary value if we just use P-control; i.e., $u(t) = K \cdot (h_{ref} - h)$?

(b) How will the stationary value change with the value of the gain K?

(c) What happens if we add integral action with very small gain $\frac{K}{T_i}$? Sketch the behaviour.

Answer mini-problem

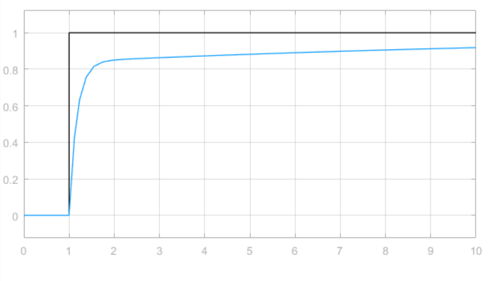
Note: This is not a strict answer and you need to make reasonable assumptions about the process yourself for this to hold.

(a) Argue why there will be a stationary value if we just use P-control; i.e., $u(t) = K \cdot (h_{ref} - h)$?
If $h = h_{ref}$ the control signal $u(t) = K \cdot (h_{ref} - h) = 0$ and the motor shuts off/fan stops spinning and the ball will fall. The process will finally settle to an equilibrium with a positive stationary error $e = h_{ref} - h$ such that the corresponding control signal will keep the ball at a fixed error (e) from the reference.

(b) How will the stationary value change with the value of the gain K?
The control signal to the fan motor $u = K \cdot e$ is the product of the gain and the error; for a higher gain K you can reach stationarity with a smaller stationary error e.

Answer mini-problem, cont'd

(c) What happens if we add integral action with very small gain $\frac{K}{T_i}$? Sketch the behaviour.



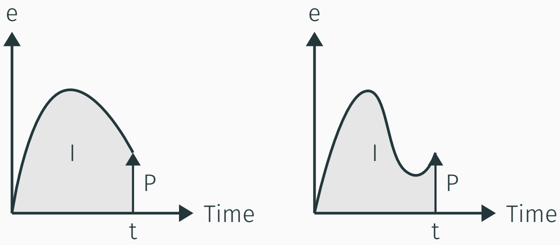
See also separate simulink example/demo.

The D Part

Idea: Speed up the PI-controller by "looking ahead"/"predicting future".

PID-controller:

$$u = K\left(e + \frac{1}{T_i}\int^t e(\tau)d\tau + T_d\frac{de}{dt}\right)$$



Same P- and I-part in both cases, but **very different behavior** of error. The derivative of e contains a lot of information to utilize.

- P acts on the current error,
- I acts on the past error,
- **D acts on the "future"/predicted error.**

State Space Models

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1} u}{dt^{n-1}} + \ldots + b_n u$$

For linear systems **the superposition principle** holds:

$$u = u_1 \implies y = y_1 \text{ and}$$

$$u = u_2 \implies y = y_2 \text{ implies}$$

$$u = c_1 \cdot u_1 + c_2 \cdot u_2 \implies y = c_1 \cdot y_1 + c_2 \cdot y_2$$

and vice versa; We can consider the output from a sum of signals by considering the influence from each component.

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1}y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1}u}{dt^{n-1}} + \ldots + b_n u$$

For linear systems **the superposition principle** holds:

$$u = u_1 \implies y = y_1 \text{ and}$$

$$u = u_2 \implies y = y_2 \text{ implies}$$

$$u = c_1 \cdot u_1 + c_2 \cdot u_2 \implies y = c_1 \cdot y_1 + c_2 \cdot y_2$$

and vice versa; We can consider the output from a sum of signals by considering the influence from each component.

Q: Why is this not true for nonlinear systems? Example?

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1} u}{dt^{n-1}} + \ldots + b_n u$$

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1} u}{dt^{n-1}} + \ldots + b_n u$$

An **altenative** to ONE differenial quation of order n^{th} is to write it as a system of n **coupled differential equations, each or order one**.

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1} u}{dt^{n-1}} + \ldots + b_n u$$

An **altenative** to ONE differenial quation of order n^{th} is to write it as a system of n **coupled differential equations, each or order one**.
State space representation:

$$\begin{cases} \dot{x}_1 & = f_1(x_1, x_2, ...x_n, u) \\ \dot{x}_2 & = f_2(x_1, x_2, ...x_n, u) \\ & ... \\ \dot{x}_n & = f_n(x_1, x_2, ...x_n, u) \\ y & = g(x_1, x_2, ...x_n, u) \end{cases}$$

The last row is a (linear) equation relating the **states (x)**, the input u, and the output y.

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1} u}{dt^{n-1}} + \ldots + b_n u$$

An **altenative** to ONE differenial quation of order n^{th} is to write it as a system of n **coupled differential equations, each or order one**.

State space representation:

$$\begin{cases} \dot{x}_1 & = a_{11}x_1 + ... + a_{1n}x_n + b_1 u \\ \dot{x}_2 & = a_{21}x_1 + ... + a_{2n}x_n + b_2 u \\ & ... \\ \dot{x}_n & = a_{n1}x_1 + ... + a_{nn}x_n + b_n u \\ y & = c_1 x_1 + c_2 x_2 + ... + c_n x_2 + du \end{cases}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \\ a_{n1} & a_{n2} & & a_{1nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \\ b_n \end{bmatrix} u$$

$$y = \begin{bmatrix} c_1 & c_2 & \ldots & c_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \\ x_n \end{bmatrix} + du$$

State Space Models

Consider a linear differential equation of order n

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^n u}{dt^n} + b_1 \frac{d^{n-1} u}{dt^{n-1}} + \ldots + b_n u$$

An **altenative** to <u>ONE</u> differenial quation of <u>order</u> n^{th} is to write it as a system of n **coupled differential equations, each or order one**.

State space representation:

$$\begin{cases} \dot{x}_1 &= a_{11}x_1 + ... + a_{1n}x_n + b_1 u \\ \dot{x}_2 &= a_{21}x_1 + ... + a_{2n}x_n + b_2 u \\ & ... \\ \dot{x}_n &= a_{n1}x_1 + ... + a_{nn}x_n + b_n u \\ y &= c_1 x_1 + c_2 x_2 + ... + c_n x_2 + du \end{cases}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \\ a_{n1} & a_{n2} & & a_{1nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \\ b_n \end{bmatrix} u$$

$$y = \begin{bmatrix} c_1 & c_2 & \ldots & c_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \\ x_n \end{bmatrix} + du$$

NOTE: **Only states (x) and inputs (u) are allowed** on the right hand side in Eq.-system above (in f and g) for it to be called a state-space representation!

State Space Models

Linear dynamics can be described in the following form

$$\dot{x} = Ax + Bu$$
$$y = Cx \, (+Du)$$

Here $x \in \mathbb{R}^n$ is a vector with states. States can have a physical ”interpretation”, but not necessary.

In this course $u \in \mathbb{R}$ and $y \in \mathbb{R}$ will be scalars.

(For MIMO systems, see Multivariable Control (FRTN10))

Example

The position of a mass m controlled by a force u is described by

$$m\ddot{x} = u$$

where x is the position of the mass.

Introduce the states $x_1 = \dot{x}$ and $x_2 = x$ and write the system on state space form. Let the position be the output.

Dynamical Systems

	Continous Time	Discrete Time (sampled)
Linear	This course	Real-Time Systems (FRTN01)
Nonlinear	Nonlinear Control and Servo Systems (FRTN05)	

Next lecture: Nonlinear dynamics can be linearized.