Implementation

Control Engineering (Reguleringsteknik)

Christoffer Sloth

`chsl@mmmi.sdu.dk`

SDU Robotics
The Maersk Mc-Kinney Moller Institute
University of Southern Denmark

SDU

Agenda

Matematiske og grafiske metoder til syntese af lineære tidsinvariante systemer:[1]

- diskret og kontinuert tilstandsbeskrivelse
- analyse i tid og frekvens
- stabilitet, reguleringshastighed, følsomhed og fejl
- digitale PI, PID, LEAD og LAG regulatorer (serieregulatorer)
- tilstandsregulering, pole-placement og tilstands-estimering (observer)
- optimal regulering (least squares) og optimal tilstands-estimation (Kalman-filter)

Færdigheder:
Efter gennemførelse af kurset kan den succesfulde studerende:

- kunne analysere, dimensionere og implementere såvel kontinuert som tidsdiskret regulering af lineære tidsinvariante og stokastiske systemer

Kompetencer:
Efter gennemførelse af kurset kan den succesfulde studerende:

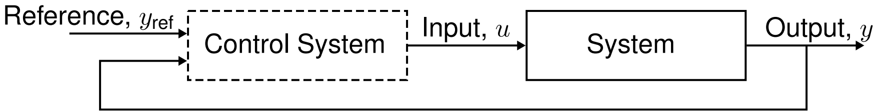
- anvende og implementere klassiske og moderne reguleringsteknikker for at kunne styre og regulere en robot hurtig og præcist

[1] Based on https://fagbesk.sam.sdu.dk/?fag_id=39673

The twelve lectures of the course are

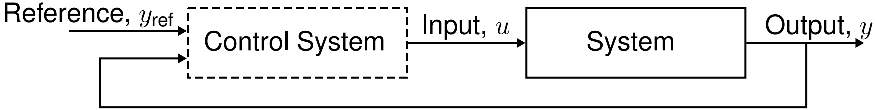
- **Lecture 1**: Introduction to Linear Time-Invariant Systems
- **Lecture 2**: Stability and Performance Analysis
- **Lecture 3**: Introduction to Control
- **Lecture 4**: Design of PID Controllers
- **Lecture 5**: Root Locus
- **Lecture 6**: The Nyquist Plot
- **Lecture 7**: Dynamic Compensators and Stability Margins
- **Lecture 8**: Implementation
- **Lecture 9**: State Feedback
- **Lecture 10**: Observer Design
- **Lecture 11**: Optimal Control (Linear Quadratic Control)
- **Lecture 12**: The Kalman Filter

Task: Design a cruise control for a car.



- **Control Input**: Throttle position u
- **Measured Output**: Velocity of the car y
- **Reference Input**: Desired velocity of the car y_{ref}

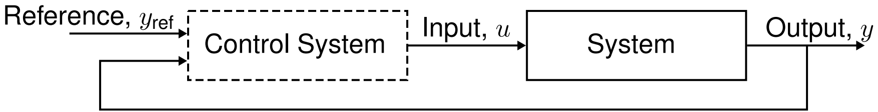
Task: Design a cruise control for a car.



Today, an answer to the following question is provided:

1. How can one ensure that a control is stable despite uncertainties in the system?

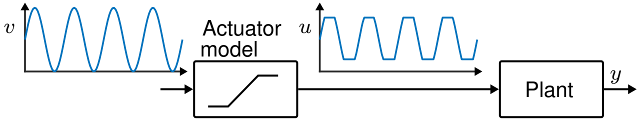
Task: Design a cruise control for a car.



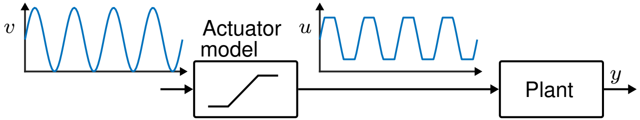
Today, an answer to the following question is provided:

1. How can one ensure that a control is stable despite uncertainties in the system?
   - Uncertainties may affect the *gain* of the system (e.g. the inertia).
   - Uncertainties may affect the *phase* of the system (e.g. communication delays between controller and sensor).

Real electromechanical systems have saturations on their physical variables. In particular, the output of any actuator is limited from above and below.



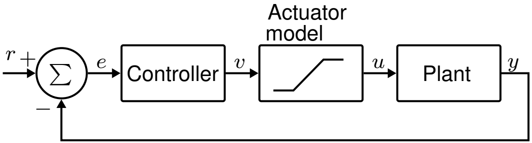
Real electromechanical systems have saturations on their physical variables. In particular, the output of any actuator is limited from above and below.



The saturation has the following input-output relation

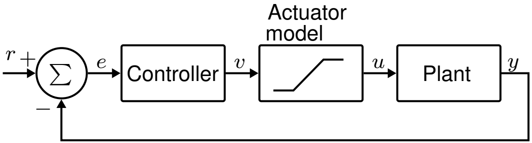
$$u = \text{sat}(v) = \begin{cases} u_{\text{max}} & \text{if } v > u_{\text{max}} \\ u_{\text{min}} & \text{if } v < u_{\text{min}} \\ v & \text{otherwise} \end{cases}$$

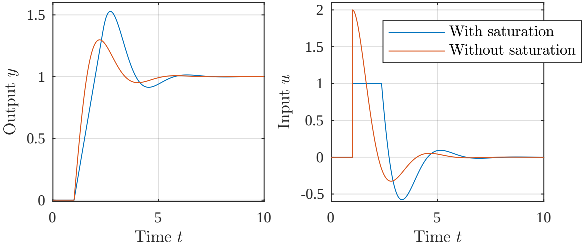
The saturation of actuators must be taken into account, when integral action is present in the controller.



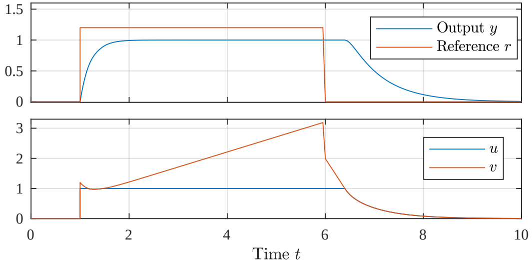
Anti-Windup
Motivation

The saturation of actuators must be taken into account, when integral action is present in the controller. Otherwise, the performance of the system may degrade significantly.





If a setpoint is given to the system that cannot be reached, then the integrator winds up, and a delay is experienced when changing the setpoint value.



It is important to only give a system reachable references.

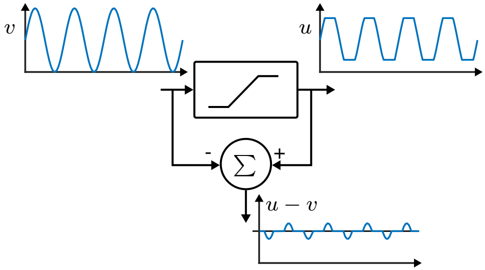
Two anti-windup mechanisms are considered

- Back-Calculation
- Conditional Integration

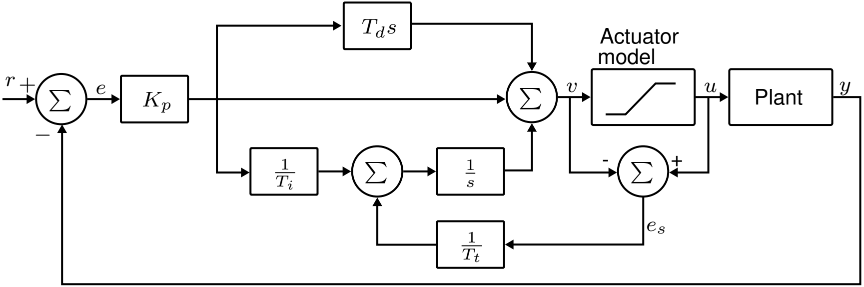
Two anti-windup mechanisms are considered

- Back-Calculation
- Conditional Integration

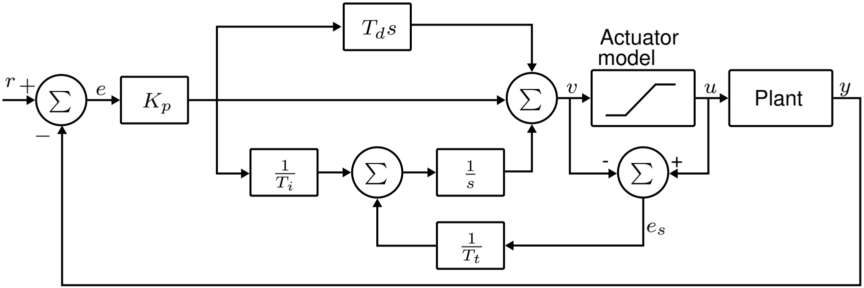
Both methods utilize the following signal



The anti-windup scheme called ***back-calculation*** recomputes the integral term when the system input is saturated, as shown below.



The anti-windup scheme called ***back-calculation*** recomputes the integral term when the system input is saturated, as shown below.



The value of the integrator output is not changed instantaneously, but it is changed based on the ***tracking time constant*** T_t.

The input to the integrator is given by

$$\frac{1}{T_t}e_s + \frac{K_p}{T_i}e$$

where e_s is zero when the system is not saturated.

In steady state, the output of the integrator is constant; hence, its input must be zero, i.e.

$$e_s = -\frac{K_p T_t}{T_i}e$$

in steady state.

The input to the integrator is given by

$$\frac{1}{T_t}e_s + \frac{K_p}{T_i}e$$

where e_s is zero when the system is not saturated.

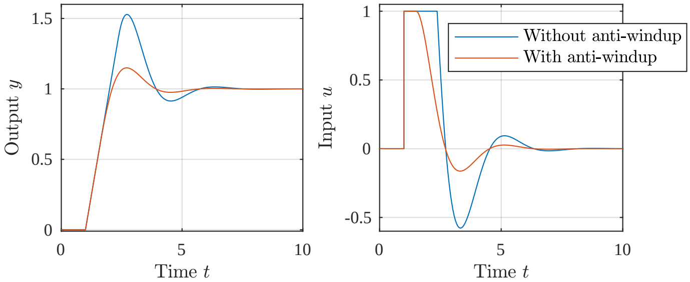
In steady state, the output of the integrator is constant; hence, its input must be zero, i.e.

$$e_s = -\frac{K_p T_t}{T_i}e$$

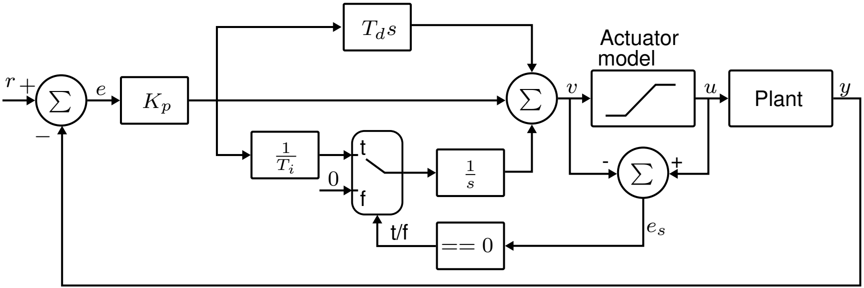
in steady state.Since $e_s = u - v$

$$v = u_{\text{lim}} + \frac{K_p T_t}{T_i}e$$

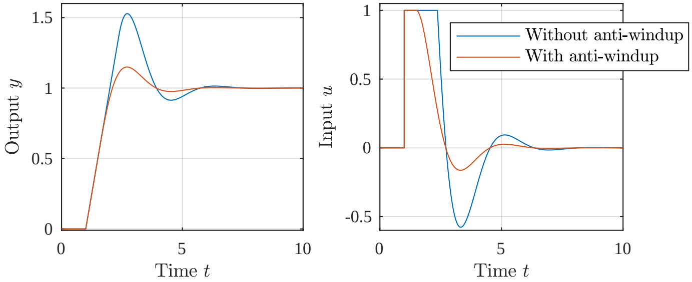
The use of anti-windup (here back-calculation) significantly improves the performance of the system.



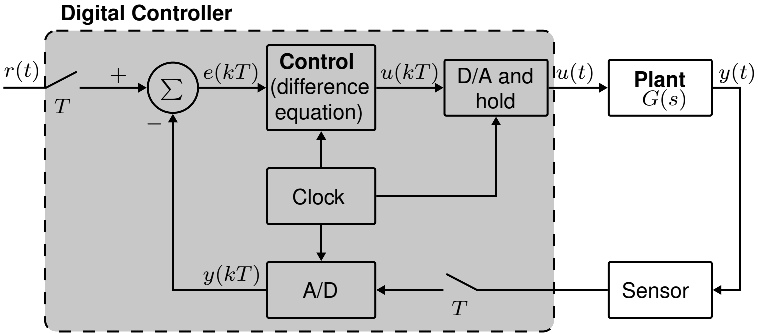
The anti-windup scheme called ***conditional integration*** (also known as clamping) is a bit simpler, and just stops integrating, when the system is in saturation.



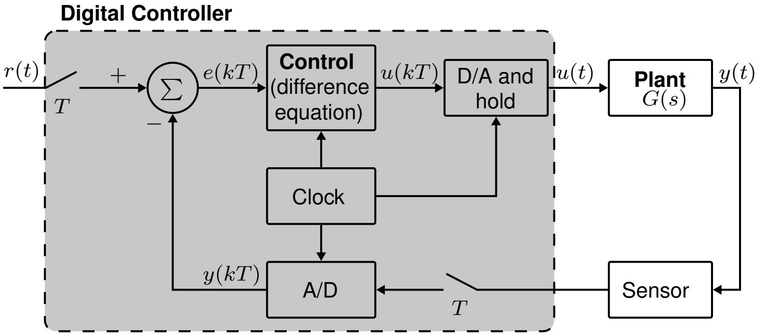
Although conditional integration is simpler than back-calculation, it also improves the performance of the system.



Most control systems are sample data systems, i.e., they consist of both discrete and continuous signals.



Most control systems are sample data systems, i.e., they consist of both discrete and continuous signals.



A ***digital controller*** both sample and quantize signals. In this course the quantization is omitted; thus, ***discrete controllers*** are designed.

There are two approaches to designing discrete controllers

1. **Emulation**: Design continuous controller $K(s)$ and approximate it with $K(z)$ obtained via e.g. Tustin's method.
2. **Discrete design**: Design the discrete controller directly, without computing $K(s)$.

There are two approaches to designing discrete controllers

1. **Emulation**: Design continuous controller $K(s)$ and approximate it with $K(z)$ obtained via e.g. Tustin's method.
2. **Discrete design**: Design the discrete controller directly, without computing $K(s)$.

The sampling frequency used for the discrete controller should be about 20 times the closed-loop bandwidth. Otherwise, special care should be takes in the design of the discrete controller.

Consider the PID controller is s-domain

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D s\right) e(s)$$

and its equivalent in time-domain

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{d}{dt} e(t)$$

Consider the PID controller is s-domain

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D s\right) e(s)$$

and its equivalent in time-domain

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{d}{dt} e(t)$$
$$u(t) = u_P + u_I + u_D$$

Consider the PID controller is s-domain

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D s\right) e(s)$$

and its equivalent in time-domain

$$\begin{aligned} u(t) &= k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{d}{dt} e(t) \\ u(t) &= u_P + u_I + u_D \end{aligned}$$

Each of the terms in the PID Controller is computed for discrete times in the following

The *proportional term* at time $kT + T$ is

$$u_P(kT + T) = k_P e(kT + T)$$

The *proportional term* at time $kT + T$ is

$$u_P(kT + T) = k_P e(kT + T)$$

The *integral term* is

$$u_I(kT + T) = k_I \int_0^{kT+T} e(\tau) d\tau$$

The *proportional term* at time $kT + T$ is

$$u_P(kT+T) = k_P e(kT+T)$$

The *integral term* is

$$u_I(kT+T) = k_I \int_0^{kT+T} e(\tau)d\tau$$

$$u_I(kT+T) = \underbrace{k_I \int_0^{kT} e(\tau)d\tau}_{u_I(kT)} + k_I \int_{kT}^{kT+T} e(\tau)d\tau$$

The *proportional term* at time $kT + T$ is

$$u_P(kT+T) = k_P e(kT+T)$$

The *integral term* is

$$u_I(kT+T) = k_I \int_0^{kT+T} e(\tau)d\tau$$

$$u_I(kT+T) = \underbrace{k_I \int_0^{kT} e(\tau)d\tau}_{u_I(kT)} + k_I \int_{kT}^{kT+T} e(\tau)d\tau$$

$$u_I(kT+T) \approx u_I(kT) + k_I \frac{T}{2}\left(e(kT+T) + e(kT)\right)$$

The *proportional term* at time $kT + T$ is

$$u_P(kT+T) = k_P e(kT+T)$$

The *integral term* is

$$u_I(kT+T) = k_I \int_0^{kT+T} e(\tau)d\tau$$

$$u_I(kT+T) = \underbrace{k_I \int_0^{kT} e(\tau)d\tau}_{u_I(kT)} + k_I \int_{kT}^{kT+T} e(\tau)d\tau$$

$$u_I(kT+T) \approx u_I(kT) + k_I \frac{T}{2}\left(e(kT+T) + e(kT)\right)$$

The ***trapezoidal rule*** was used for the numerical integration. Other choices for the numerical integration are possible.

The *derivative term* is obtained by integrating $u_D(t) = k_D\dot{e}(t)$

$$k_D e(t) = \int_0^t u_D(\tau)d\tau$$

The *derivative term* is obtained by integrating $u_D(t) = k_D\dot{e}(t)$

$$k_D e(t) = \int_0^t u_D(\tau)d\tau$$

$$k_D e(kT+T) = \int_0^{kT+T} u_D(\tau)d\tau$$

The *derivative term* is obtained by integrating $u_D(t) = k_D \dot{e}(t)$

$$
\begin{aligned}
k_D e(t) &= \int_0^t u_D(\tau) d\tau \\
k_D e(kT+T) &= \int_0^{kT+T} u_D(\tau) d\tau \\
k_D e(kT+T) &= \underbrace{\int_0^{kT} u_D(\tau) d\tau}_{k_D e(kT)} + \int_{kT}^{kT+T} u_D(\tau) d\tau
\end{aligned}
$$

The *derivative term* is obtained by integrating $u_D(t) = k_D\dot{e}(t)$

$$
\begin{aligned}
k_D e(t) &= \int_0^t u_D(\tau)d\tau \\
k_D e(kT+T) &= \int_0^{kT+T} u_D(\tau)d\tau \\
k_D e(kT+T) &= \underbrace{\int_0^{kT} u_D(\tau)d\tau}_{k_D e(kT)} + \int_{kT}^{kT+T} u_D(\tau)d\tau \\
k_D e(kT+T) &\approx k_D e(kT) + \frac{T}{2}\left(u_D(kT+T) + u_D(kT)\right)
\end{aligned}
$$

The *derivative term* is obtained by integrating $u_D(t) = k_D\dot{e}(t)$

$$k_D e(t) = \int_0^t u_D(\tau)d\tau$$

$$k_D e(kT+T) = \int_0^{kT+T} u_D(\tau)d\tau$$

$$k_D e(kT+T) = \underbrace{\int_0^{kT} u_D(\tau)d\tau}_{k_D e(kT)} + \int_{kT}^{kT+T} u_D(\tau)d\tau$$

$$k_D e(kT+T) \approx k_D e(kT) + \frac{T}{2}\left(u_D(kT+T) + u_D(kT)\right)$$

$$u_D(kT+T) \approx k_D \frac{2}{T}(e(kT+T) - e(kT)) - u_D(kT)$$

By combining the three terms, the control output of the discrete controller is

$$u(kT+T) = \underbrace{k_P e(kT+T)}_{u_P(kT+T)} + \underbrace{u_I(kT) + k_I \frac{T}{2}\left(e(kT+T) + e(kT)\right)}_{u_I(kT+T)} + \underbrace{k_D \frac{2}{T}(e(kT+T) - e(kT)) - u_D(kT)}_{u_D(kT+T)}$$

By combining the three terms, the control output of the discrete controller is

$$u(kT{+}T) = \underbrace{k_P e(kT+T)}_{u_P(kT+T)} + \underbrace{u_I(kT) + k_I \frac{T}{2}\left(e(kT+T)+e(kT)\right)}_{u_I(kT+T)} + \underbrace{k_D \frac{2}{T}\left(e(kT+T)-e(kT)\right) - u_D(kT)}_{u_D(kT+T)}$$

Notice that the terms in the controller depends on the sampling time T. Thus, it needs to be known and constant for implementing the PID controller with constant gains.

By combining the three terms, the control output of the discrete controller is

$$u(kT+T) = \underbrace{k_P e(kT+T)}_{u_P(kT+T)} + \underbrace{u_I(kT) + k_I \frac{T}{2}\left(e(kT+T) + e(kT)\right)}_{u_I(kT+T)} + \underbrace{k_D \frac{2}{T}\left(e(kT+T) - e(kT)\right) - u_D(kT)}_{u_D(kT+T)}$$

Notice that the terms in the controller depends on the sampling time T. Thus, it needs to be known and constant for implementing the PID controller with constant gains.

To analyze the system, the I-term and D-term are z-transformed

$$zu_I(z) = u_I(z) + k_I \frac{T}{2}\left(ze(z) + e(z)\right)$$

$$zu_D(z) = k_D \frac{2}{T}\left(ze(z) - e(z)\right) - u_D(z)$$

By combining the three terms, the control output of the discrete controller is

$$u(kT{+}T) = \underbrace{k_P e(kT+T)}_{u_P(kT+T)} + \underbrace{u_I(kT) + k_I \frac{T}{2}\left(e(kT+T)+e(kT)\right)}_{u_I(kT+T)} + \underbrace{k_D \frac{2}{T}\left(e(kT+T)-e(kT)\right) - u_D(kT)}_{u_D(kT+T)}$$

Notice that the terms in the controller depends on the sampling time T. Thus, it needs to be known and constant for implementing the PID controller with constant gains.

To analyze the system, the I-term and D-term are z-transformed

$$zu_I(z) = u_I(z) + k_I \frac{T}{2}\left(ze(z)+e(z)\right)$$
$$zu_D(z) = k_D \frac{2}{T}\left(ze(z)-e(z)\right) - u_D(z)$$

This gives the following expression for the controller

$$u(z) = \left(k_P + k_I \frac{T}{2}\frac{z+1}{z-1} + k_D \frac{2}{T}\frac{z-1}{z+1}\right) e(z)$$

Note that the PID controller in s-domain

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D s\right) e(s)$$

is given in the z-domain as

$$u(z) = \left(k_P + k_I \frac{T}{2} \frac{z+1}{z-1} + k_D \frac{2}{T} \frac{z-1}{z+1}\right) e(z)$$

Note that the PID controller in s-domain

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D s \right) e(s)$$

is given in the z-domain as

$$u(z) = \left(k_P + k_I \frac{T}{2} \frac{z+1}{z-1} + k_D \frac{2}{T} \frac{z-1}{z+1} \right) e(z)$$

This means that the discrete controller is obtained by replacing s with (this is called the trapezoidal rule or Tustin's Method)

$$\frac{2}{T} \frac{z-1}{z+1}$$

Note that the PID controller in s-domain

$$u(s) = \left(k_P + \frac{k_I}{s} + k_D s \right) e(s)$$

is given in the z-domain as

$$u(z) = \left(k_P + k_I \frac{T}{2} \frac{z+1}{z-1} + k_D \frac{2}{T} \frac{z-1}{z+1} \right) e(z)$$

This means that the discrete controller is obtained by replacing s with (this is called the trapezoidal rule or Tustin's Method)

$$\frac{2}{T} \frac{z-1}{z+1}$$

In conclusion, a discrete equivalent of a controller $K(s)$ is (MATLAB: `c2d`)

$$K_d(z) = K\left(\frac{2}{T} \frac{z-1}{z+1} \right)$$

Consider the system

$$G(s) = \frac{45}{(s+9)(s+5)}$$

and a PI controller with transfer function

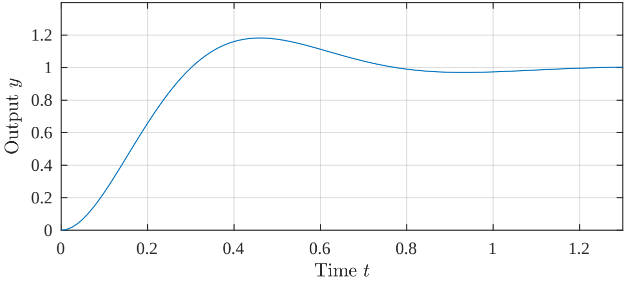
$$K(s) = 1.4\frac{s+6}{s}$$

Consider the system

$$G(s) = \frac{45}{(s+9)(s+5)}$$

and a PI controller with transfer function

$$K(s) = 1.4\frac{s+6}{s}$$



The system has 20% overshoot and a rise time of 0.2 s.

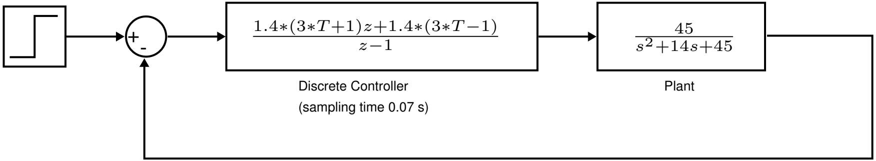
A discrete equivalent controller is given by

$$K_d(z) = K\left(\frac{2}{T}\frac{z-1}{z+1}\right) = 1.4\frac{\frac{2}{T}\frac{z-1}{z+1}+6}{\frac{2}{T}\frac{z-1}{z+1}} = 1.4\frac{(1+3T)z+(3T-1)}{z-1}$$

A discrete equivalent controller is given by

$$K_d(z) = K\left(\frac{2}{T}\frac{z-1}{z+1}\right) = 1.4\frac{\frac{2}{T}\frac{z-1}{z+1}+6}{\frac{2}{T}\frac{z-1}{z+1}} = 1.4\frac{(1+3T)z+(3T-1)}{z-1}$$

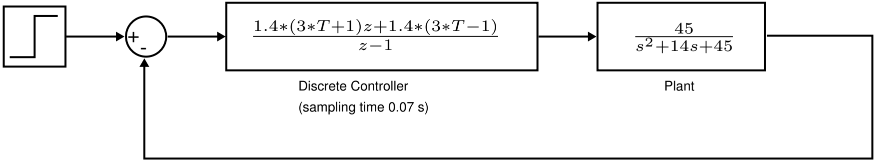
A block diagram of the implementation is given below. Note that the sample rate needs to be specified in the boxes.



A discrete equivalent controller is given by

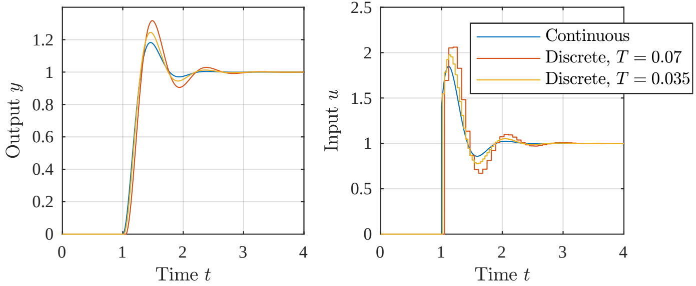
$$K_d(z) = K\left(\frac{2}{T}\frac{z-1}{z+1}\right) = 1.4\frac{\frac{2}{T}\frac{z-1}{z+1}+6}{\frac{2}{T}\frac{z-1}{z+1}} = 1.4\frac{(1+3T)z+(3T-1)}{z-1}$$

A block diagram of the implementation is given below. Note that the sample rate needs to be specified in the boxes.

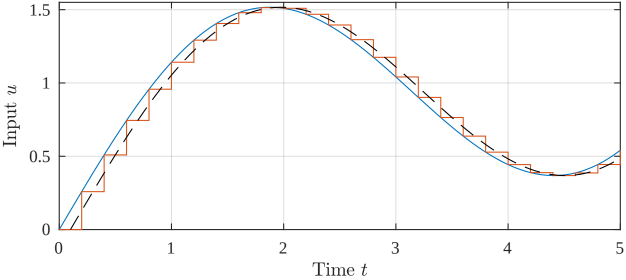


Note that you in practice implement the difference equation - not a discrete transfer function.

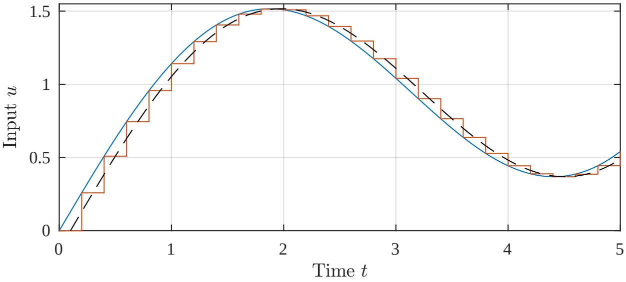
A comparison of controllers with different sampling times show that an decreased sampling time improved the system response.



A signal sampled with zero-order hold is in average delayed by $T/2$, where T is the sampling time.

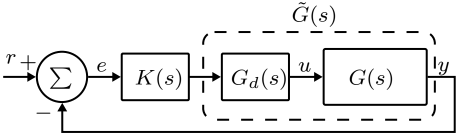


A signal sampled with zero-order hold is in average delayed by $T/2$, where T is the sampling time.

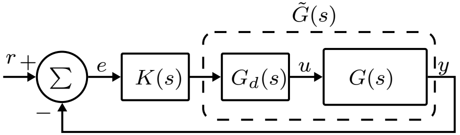


It is appropriate to include this delay in the model when designing a controller by emulation.

To incorporate the delay caused by zero order hold in the design, the continuous controller should be designed for the system $\tilde{G}(s)$, where $G_d(s)$ models the delay of $T/2$.



To incorporate the delay caused by zero order hold in the design, the continuous controller should be designed for the system $\tilde{G}(s)$, where $G_d(s)$ models the delay of $T/2$.

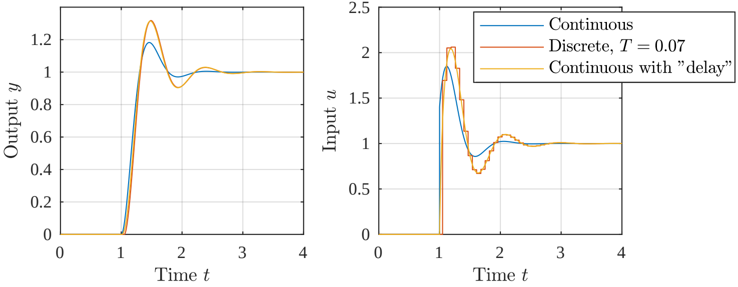


The delay is typically modeled using a first-order system with a time constant of $T/2$, i.e.,

$$G_d(s) = \frac{1}{T/2s + 1}$$

Other models such as Padé approximations may also be used.

It is seen that the behavior of the delayed system ($G_d(s)G(s)$) is very close to the behavior of the sampled data system.



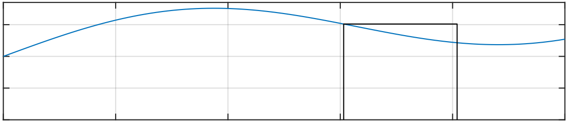
Digitization

Emulation: Design Procedure

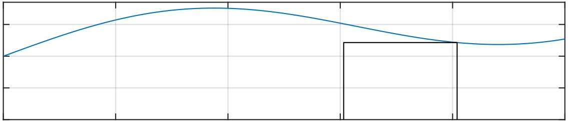
A discrete controller can be designed by emulation for the system $G(s)$ according to the next procedure.

1. Design continuous compensation for the system $G_d(s)G(s)$, where $G_d(s)$ approximates a delay of $T/2$.
2. Derive the discrete controller by applying Tustin's rule or the matched pole-zero method (other discretization methods exist, but the mentioned methods are preferred).
3. Analyze the design by simulation or experimentally.

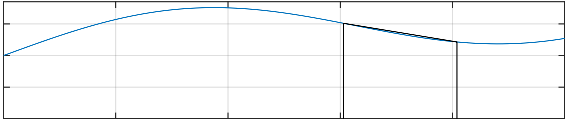
The following integration rules are commonly used.



Forward Rectangular Rule



Backward Rectangular Rule



Trapezoid Rule

The following rules are used for approximating discrete transfer functions by substituting s by the following expressions

Method	Approximation
Forward rule	$s \leftarrow \frac{z-1}{T}$
Backward rule	$s \leftarrow \frac{z-1}{Tz}$
Trapezoid rule	$s \leftarrow \frac{2}{T}\frac{z-1}{z+1}$

The following rules are used for approximating discrete transfer functions by substituting s by the following expressions

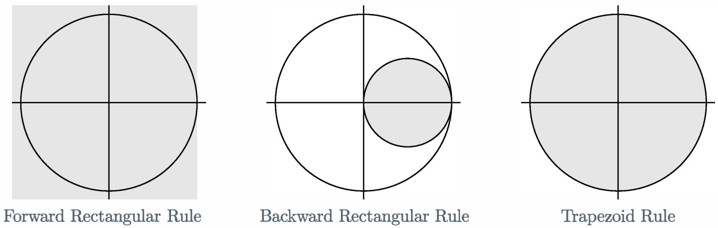
Method	Approximation
Forward rule	$s \leftarrow \frac{z-1}{T}$
Backward rule	$s \leftarrow \frac{z-1}{Tz}$
Trapezoid rule	$s \leftarrow \frac{2}{T}\frac{z-1}{z+1}$

To get the transfer function from a discrete transfer function, the variable z is replaced by the following expressions

Method	Approximation
Forward rectangular rule	$z \leftarrow 1 + Ts$
Backward rectangular rule	$z \leftarrow \frac{1}{1-Ts}$
Bilinear rule	$z \leftarrow \frac{1+Ts/2}{1-Ts/2}$

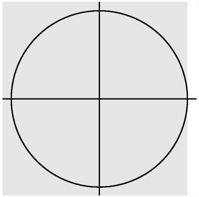
Digitization

The left half-plane is mapped to different regions of the z-plane (shaded area) dependent on the numeric approximation method.

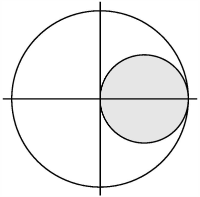


Digitization

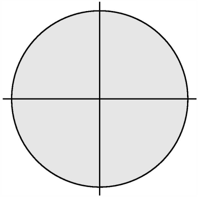
The left half-plane is mapped to different regions of the z-plane (shaded area) dependent on the numeric approximation method.



Forward Rectangular Rule



Backward Rectangular Rule



Trapezoid Rule

Discretization of a stable system using the forward rectangular rule may lead to an unstable discrete system.

The discrete design method relies on a discretized plant model. The discrete transfer function of a system $G(s)$ and preceding zero-order hold is

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\}$$

where $\mathcal{Z}$ denotes the z-transformation.

The discrete design method relies on a discretized plant model. The discrete transfer function of a system $G(s)$ and preceding zero-order hold is

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\}$$

where $\mathcal{Z}$ denotes the z-transformation.

Subsequent to the discretization, the control system can be analyzed based on the closed-loop transfer function

$$T_{\text{cl}}(z) = \frac{K(z)G(z)}{1 + K(z)G(z)}$$

Discrete Design

Overview

The discrete design method relies on a discretized plant model. The discrete transfer function of a system $G(s)$ and preceding zero-order hold is

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\}$$

where $\mathcal{Z}$ denotes the z-transformation.

Subsequent to the discretization, the control system can be analyzed based on the closed-loop transfer function

$$T_{\text{cl}}(z) = \frac{K(z)G(z)}{1 + K(z)G(z)}$$

Since the structure of the continuous and discrete closed-loop transfer functions are the same, the stability of the system can be studied via the characteristic equation

$$1 + K(z)G(z) = 0$$

The discrete design method relies on a discretized plant model. The discrete transfer function of a system $G(s)$ and preceding zero-order hold is

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\}$$

where $\mathcal{Z}$ denotes the z-transformation.

Subsequent to the discretization, the control system can be analyzed based on the closed-loop transfer function

$$T_{\text{cl}}(z) = \frac{K(z)G(z)}{1 + K(z)G(z)}$$

Since the structure of the continuous and discrete closed-loop transfer functions are the same, the stability of the system can be studied via the characteristic equation

$$1 + K(z)G(z) = 0$$

Consequently, the controller $K(z)$ can be designed via root locus or any of the other methods.

The following procedure should be followed to designing a discrete controller

1. Transform the continuous-time plant into discrete time as follows

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\}$$

2. Design the feedback controller $K(z)$ using the same approaches as for a continuous-time.
3. Verify the design on the sample data system.

Consider the following system preceded by a ZOH

$$G(s) = \frac{1}{s^2}$$

Design a controller such that the closed-loop system has a damping ratio $\zeta = 0.7$ and natural frequency $\omega_n = 0.3$ rad/s.

Consider the following system preceded by a ZOH

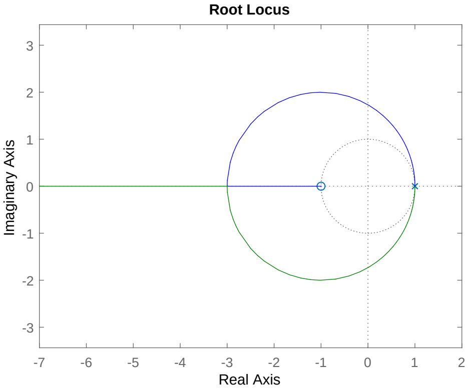
$$G(s) = \frac{1}{s^2}$$

Design a controller such that the closed-loop system has a damping ratio $\zeta = 0.7$ and natural frequency $\omega_n = 0.3$ rad/s.

The discrete transfer function is given by

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\} = (1 - z^{-1})\mathcal{Z}\left\{\frac{1}{s^3}\right\} = (1 - z^{-1})\frac{T^2}{2}\frac{z(z+1)}{(z-1)^3} = \frac{T^2}{2}\frac{z+1}{(z-1)^2}$$

To find a controller, a root locus is generated for a P-controller applied to the system.



A second try for a controller structure is a PD-controller on the form $K(s) = K\frac{z-\alpha}{z}$ with $\alpha = 0.85$.