



Intro. Computer Control Systems: F11

Discrete-time control

Dave Zachariah

Dept. Information Technology, Div. Systems and Control



F10: Quiz!



F10: Quiz!

- 1) Sensitivity function $S(s)$
- a does not affect control \uparrow
 - b amplifies output disturbances \uparrow
 - c is stable \downarrow



F10: Quiz!

- 1) Sensitivity function $S(s)$
 - a does not affect control \uparrow
 - b amplifies output disturbances \uparrow
 - c is stable \downarrow

- 2) Sensitivity function $S(s)$ and its complement $T(s)$
 - a sum to 1 \uparrow
 - b sum to $G_o(s)$ \uparrow
 - c sum to a complex-valued number \downarrow

F10: Quiz!

- 1) Sensitivity function $S(s)$
 - a does not affect control \uparrow
 - b amplifies output disturbances \uparrow
 - c is stable \downarrow

- 2) Sensitivity function $S(s)$ and its complement $T(s)$
 - a sum to 1 \uparrow
 - b sum to $G_o(s)$ \uparrow
 - c sum to a complex-valued number \downarrow

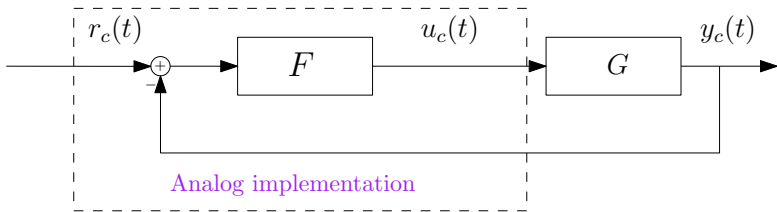
- 3) If $|T(i\omega)|$ is below $1/|\Delta_G(i\omega)|$
 - a the unknown system $G^0(s)$ is stable \uparrow
 - b the closed-loop system can be stabilized despite unknown system $G^0(s)$ \uparrow
 - c the output becomes real-valued \downarrow



Digital controllers

Digital systems and discrete-time clocks

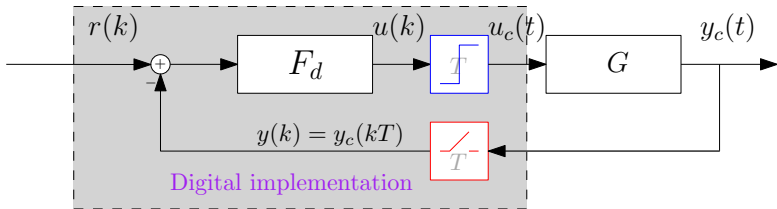
Example: Simple linear feedback control system



Notation: continuous-time signals $y_c(t)$, etc.

Digital systems and discrete-time clocks

Example: Simple linear feedback control system



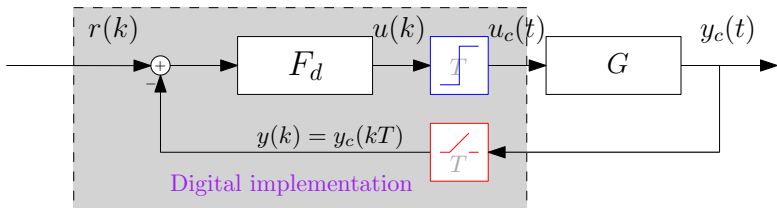
Note: discrete-time signals with sample interval T

- ▶ **A/D:** Sampling feedback signal $y(k) = y_c(kT)$
- ▶ **D/A:** Generate continuous-time input signal $u(t)$



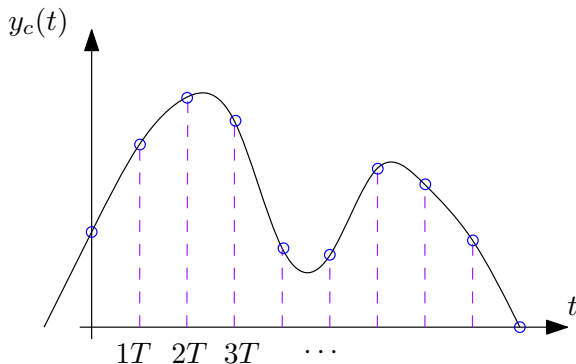
Sampling continuous-time output

Digital systems and discrete-time clocks



A/D: Sampling feedback signal $y(t) = y_c(kT)$

Sampling the feedback signal



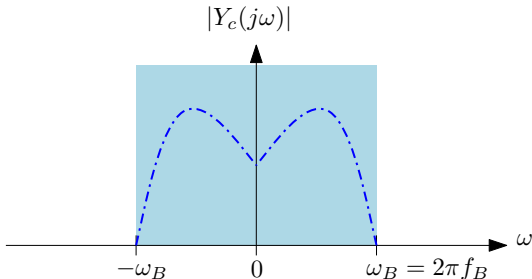
$$y(k) = y_c(kT), \quad k = 0, 1, \dots$$

Sampling the feedback signal

Nyquist sampling theorem:

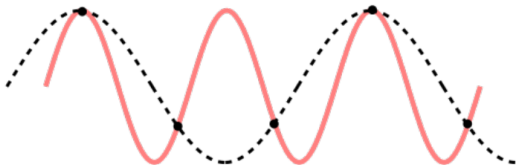
If $y_c(t)$ is **bandlimited** to $\pm\omega_B = \pm 2\pi f_B \Rightarrow$ it can be **recovered exactly** from $y_c(kT)$ when sampling frequency

$$f_s \equiv \frac{1}{T} \geq 2f_B$$



Sampling the feedback signal

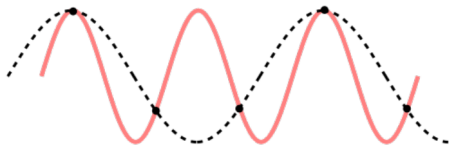
If $T > \frac{1}{2f_B}$ we may not resolve $y_c(t)$ uniquely resulting in distortion/'aliasing'.



High frequency components $f > 1/(2T)$ may appear at low frequencies.

Sampling the feedback signal

If $T > \frac{1}{2f_B}$ we may not resolve $y(t)$ uniquely resulting in distortion/aliasing'.



Practical solution for a given T :

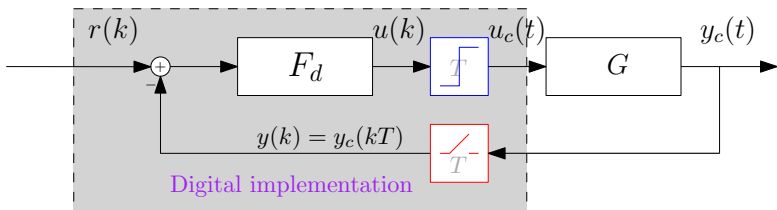
1. Low-pass filter $y_c(t)$ with $L(s)$: output $y_L(t)$ bandlimited to $f_B = 1/(2T)$.
2. Then sample filtered signal $y(k) = y_L(kT)$.



Generating continuous-time input

Digital systems and discrete-time clocks

Example: Simple linear feedback control system

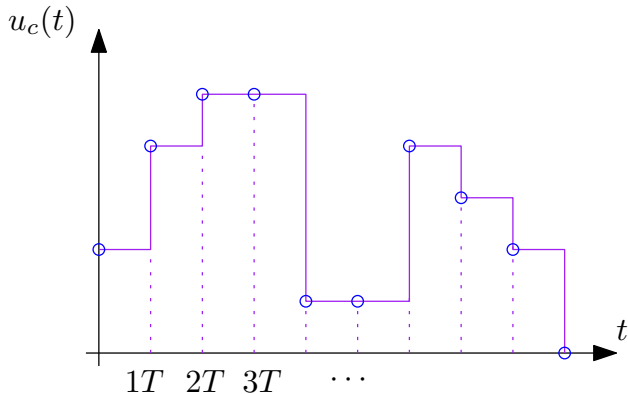


D/A: Generate continuous-time input signal $u(t)$

Generate input signal using zero-order hold

Zero-order hold: continuous-time input is *piecewise constant* for $k = 0, 1, 2, \dots$

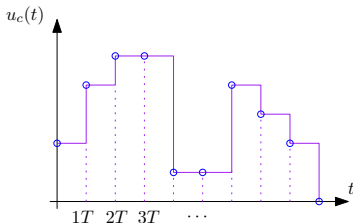
$$u_c(t) = u_c(kT), \quad kT \leq t < (k+1)T$$



Generate input signal using zero-order hold

Zero-order hold: continuous-time input is *piecewise constant* for $k = 0, 1, 2, \dots$

$$u_c(t) = u_c(kT), \quad kT \leq t < (k+1)T$$



Determine state evolution using **zero-order hold** input

$$\dot{x}_c(t) = Ax_c(t) + Bu_c(t),$$

starting at $t = kT$ (instead of 0)

Generate input signal using zero-order hold

Zero-order hold: continuous-time input is *piecewise constant* for $k = 0, 1, 2, \dots$

$$u_c(t) = u_c(kT), \quad kT \leq t < (k+1)T$$

Starting at $x_c(kT)$, general solution:

$$x_c(t) = e^{A(t-kT)}x_c(kT) + \int_{kT}^t e^{A(t-\tau)}Bu_c(t)d\tau$$

Now evaluate at $t = (k+1)T$

Generate input signal using zero-order hold

Zero-order hold: continuous-time input is *piecewise constant* for $k = 0, 1, 2, \dots$

$$u_c(t) = u_c(kT), \quad kT \leq t < (k+1)T$$

Starting at $x_c(kT)$, general solution:

$$x_c(t) = e^{A(t-kT)} x_c(kT) + \int_{kT}^t e^{A(t-\tau)} B u_c(t) d\tau$$

Evaluated at $t = (k+1)T$ with piecewise constant $u_c(t)$:

$$x_c((k+1)T) = e^{AT} x_c(kT) + \left(\int_{kT}^{(k+1)T} e^{A((k+1)T-\tau)} d\tau B \right) u_c(kT)$$



Discrete-time system model

State-space descriptions

State evolution when using zero-order hold input

State $x_c(t)$ at time $(k+1)T$ is determined at time kT as:

$$x(k+1) = Fx(k) + Gu(k)$$

Discrete-time system model

State-space descriptions

State evolution when using zero-order hold input

State $x_c(t)$ at time $(k+1)T$ is determined at time kT as:

$$x(k+1) = Fx(k) + Gu(k)$$

which are discrete-time states using matrices

$$F = e^{AT}$$

$$G = \int_{\tau=0}^T e^{A\tau} d\tau B = [\text{if } A^{-1} \text{ exists}] = A^{-1}(e^{AT} - I)B$$

Discrete-time system model

State-space descriptions

State evolution when using zero-order hold input

State $x_c(t)$ at time $(k+1)T$ is determined at time kT as:

$$x(k+1) = Fx(k) + Gu(k)$$

which are discrete-time states using matrices

$$F = e^{AT}$$

$$G = \int_{\tau=0}^T e^{A\tau} d\tau B = [\text{if } A^{-1} \text{ exists}] = A^{-1}(e^{AT} - I)B$$

Matrix exponential as before

$$e^{At} = \mathcal{L}^{-1}\{(sI - A)^{-1}\}$$



Build intuition

Build intuition from simple systems

Discrete-time state-space description

Example: Continuous-time system with **zero-order hold input**

$$\begin{aligned}\dot{x}_c(t) &= Ax_c(t) + Bu_c(t) \\ y_c(t) &= Cx_c(t),\end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = [1 \quad 1]$$

Build intuition from simple systems

Discrete-time state-space description

Example: Continuous-time system with **zero-order hold input**

$$\begin{aligned}\dot{x}_c(t) &= Ax_c(t) + Bu_c(t) \\ y_c(t) &= Cx_c(t),\end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = [1 \quad 1]$$

[Board: derive discrete-time state-space description]

Build intuition from simple systems

Discrete-time state-space description

Example: Continuous-time system with **zero-order hold input**

$$\begin{aligned}\dot{x}_c(t) &= Ax_c(t) + Bu_c(t) \\ y_c(t) &= Cx_c(t),\end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = [1 \quad 1]$$

[Board: derive discrete-time state-space description]

$$\begin{aligned}x(k+1) &= Fx(k) + Gu(k) \\ y(k) &= Hx(k)\end{aligned}$$

Note: Easy to simulate in computer!



Poles in discrete-time system



Discrete-time system model

State-space descriptions

General solution to $\dot{x}_c(t) = Ax_c(t) + Bu_c(t)$:

$$x_c(t) = e^{At}x_c(0) + \int_{\tau=0}^t e^{A(t-\tau)}Bu_c(\tau)d\tau$$

Discrete-time system model

State-space descriptions

General solution to $\dot{x}_c(t) = Ax_c(t) + Bu_c(t)$:

$$x_c(t) = e^{At}x_c(0) + \int_{\tau=0}^t e^{A(t-\tau)}Bu_c(\tau)d\tau$$

If eigenvalues of A : $\text{Re}\{p\} < 0 \Rightarrow$ stable

Discrete-time system model

State-space descriptions

General solution to $\dot{x}_c(t) = Ax_c(t) + Bu_c(t)$:

$$x_c(t) = e^{At}x_c(0) + \int_{\tau=0}^t e^{A(t-\tau)}Bu_c(\tau)d\tau$$

If eigenvalues of A : $\text{Re}\{p\} < 0 \Rightarrow$ stable

General solution to $x(k+1) = Fx(k) + Gu(k)$:

$$x(k) = F^k x(0) + \sum_{n=0}^{k-1} F^{k-n} Gu(n)$$

Discrete-time system model

State-space descriptions

General solution to $\dot{x}_c(t) = Ax_c(t) + Bu_c(t)$:

$$x_c(t) = e^{At}x_c(0) + \int_{\tau=0}^t e^{A(t-\tau)}Bu_c(\tau)d\tau$$

If eigenvalues of A : $\text{Re}\{p\} < 0 \Rightarrow$ stable

General solution to $x(k+1) = Fx(k) + Gu(k)$:

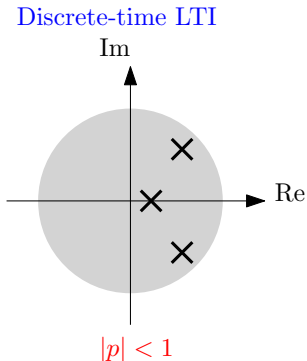
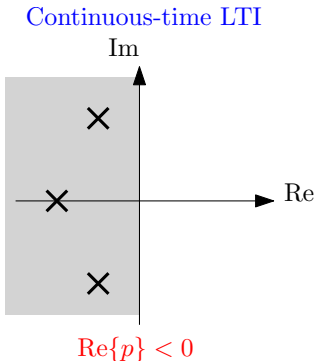
$$x(k) = F^k x(0) + \sum_{n=0}^{k-1} F^{k-n} Gu(n)$$

If eigenvalues of F : $|p| < 1 \Rightarrow$ stable

Discrete-time LTI systems

Eigenvalues/poles

Stability: left half-plane vs. unit circle



Discrete-time LTI systems

Eigenvalues/poles and sampling

Eigenvalues/poles of sampled system

System $\dot{x}_c(t) = Ax_c(t) + Bu_c(t)$ with eigenvalues p_1, \dots, p_n .
Using **zero-order hold input**, sampled system

$$x(k+1) = Fx(k) + Gu(k)$$

\Rightarrow eigenvalues of $F = e^{AT}$:

$$\tilde{p}_1 = e^{p_1 T}, \dots, \tilde{p}_n = e^{p_n T}$$

Discrete-time LTI systems

Eigenvalues/poles and sampling

Eigenvalues/poles of sampled system

System $\dot{x}_c(t) = Ax_c(t) + Bu_c(t)$ with eigenvalues p_1, \dots, p_n .
Using **zero-order hold input**, sampled system

$$x(k+1) = Fx(k) + Gu(k)$$

\Rightarrow eigenvalues of $F = e^{AT}$:

$$\tilde{p}_1 = e^{p_1 T}, \dots, \tilde{p}_n = e^{p_n T}$$

Note: Large T /slower sampling, increases distance $|\tilde{p}| = |e^{pT}|!$



Discrete-time controllers

Discrete-time controllers

- ▶ PID controller:

$$u(k) = K_p e(k) + K_i \underbrace{T \sum_{n=0}^k e(n)}_{\simeq \int_{\tau=0}^t e_c(\tau) d\tau} + K_d \underbrace{\frac{(e(k) - e(k-1)))}{T}}_{\simeq \frac{de_c(t)}{dt}}$$

Discrete-time controllers

- ▶ PID controller:

$$u(k) = K_p e(k) + K_i \underbrace{T \sum_{n=0}^k e(n)}_{\simeq \int_{\tau=0}^t e_c(\tau) d\tau} + K_d \underbrace{\frac{(e(k) - e(k-1)))}{T}}_{\simeq \frac{de_c(t)}{dt}}$$

- ▶ State-feedback controller:

$$u(k) = -Lx(k) + \ell_0 r(k)$$

Plug back into state-space equations to obtain closed-loop system

Discrete-time controllers

- ▶ **PID** controller:

$$u(k) = K_p e(k) + K_i \underbrace{T \sum_{n=0}^k e(n)}_{\simeq \int_{\tau=0}^t e_c(\tau) d\tau} + K_d \underbrace{\frac{(e(k) - e(k-1))}{T}}_{\simeq \frac{de_c(t)}{dt}}$$

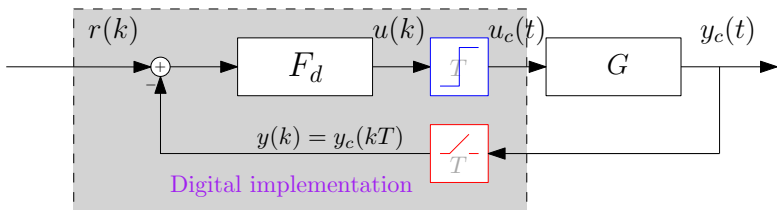
- ▶ **State-feedback** controller:

$$u(k) = -Lx(k) + \ell_0 r(k)$$

Plug back into state-space equations to obtain closed-loop system

More advanced controllers in Automatic Control II!

Discrete-time controllers



More advanced controllers in Automatic Control II!



Summary and recap

- ▶ Sampling output and Nyquist sampling theorem
- ▶ Zero-order hold input and state-space form
- ▶ Discrete-time LTI eigenvalues/poles