Topic #20

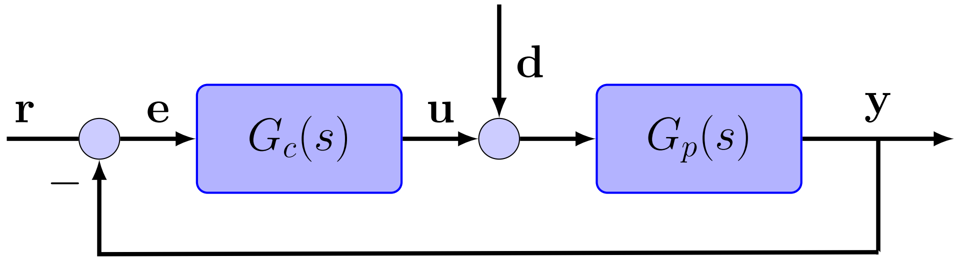
16.30/31 Feedback Control Systems

Digital Control Basics

- Effective Delay
- Emulation

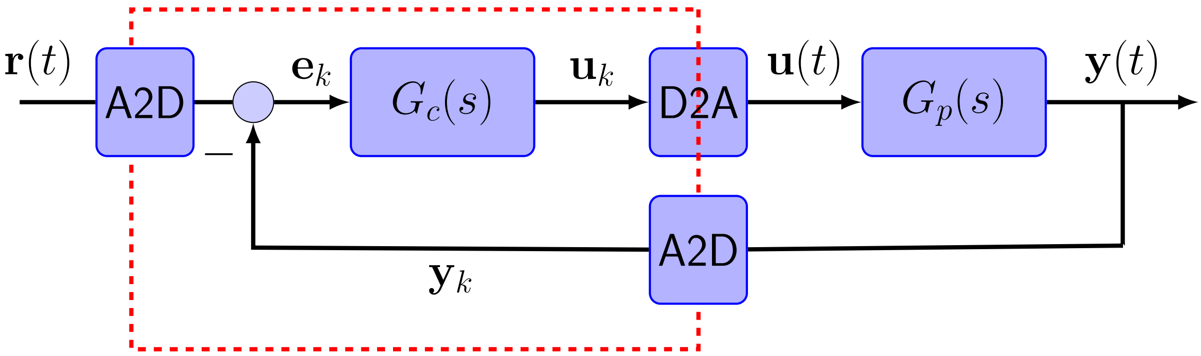
Digital Control

- Control picture so far



- Can implement this using analog circuits, but as you have seen, there are many advantages to implementing these using a computer – much more flexible

- In this case the new picture is:



Digital Control Mechanics

- Digital/discrete control runs on a clock
  - Only uses the input signals at discrete instants in time
  - So continuous $e(t)$ is sampled at fixed periods in time $e(kT_s)$
  - Where T_s is the sampling period and k is an integer

- Must also get information into and out of the computer
  - Requires A/D and D/A operations

- The A/D consists of 2 steps:
  1. Convert a physical signal (voltage) to a binary number, which is an approximation since we will only have a 12-16 bits to cover a $\pm 10V$ range.
  2. Sample a continuous signal $e(t)$ every T_s seconds so that

$$y(t) \Rightarrow y(k)$$

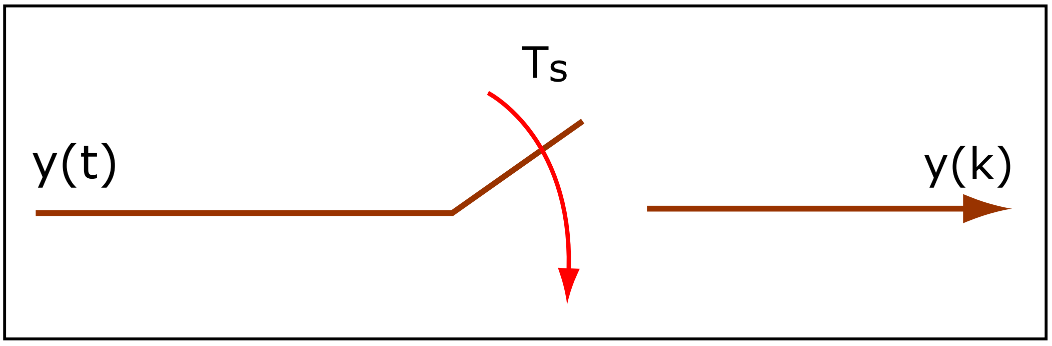


Image by MIT OpenCourseWare.

  3. Sampler clearly ignores a large part of the continuous signal.

- The D/A consists of 2 steps as well
  1. Binary to analog
  2. Convert discrete signal (at kT_s) to a continuous one.

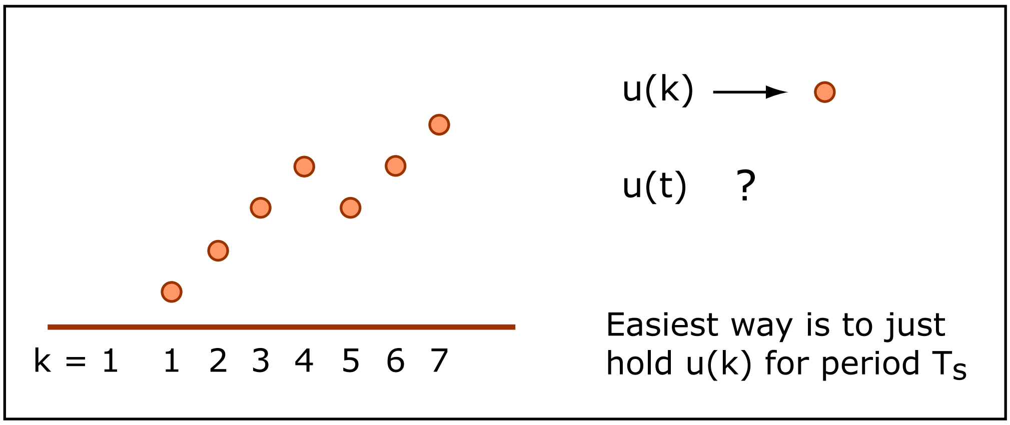


Image by MIT OpenCourseWare.

- Basic approach is to just hold the latest value of $u(k)$ for the entire periods T_s
  - Called a zero-order hold (ZOH)

- Need to determine what impact this "sample and hold" operation might have on the loop transfer function
  - Approximate the A/D as sample
  - Approximate D/A as ZOH
  - Set control law to 1, so $u(k) = e(k)$

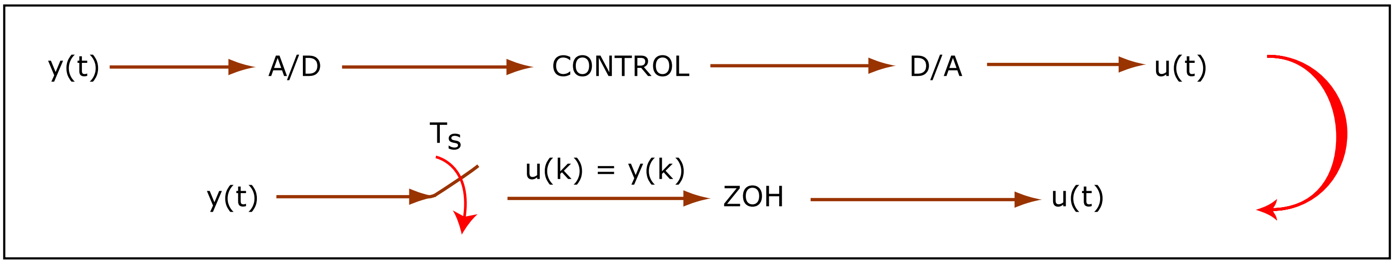
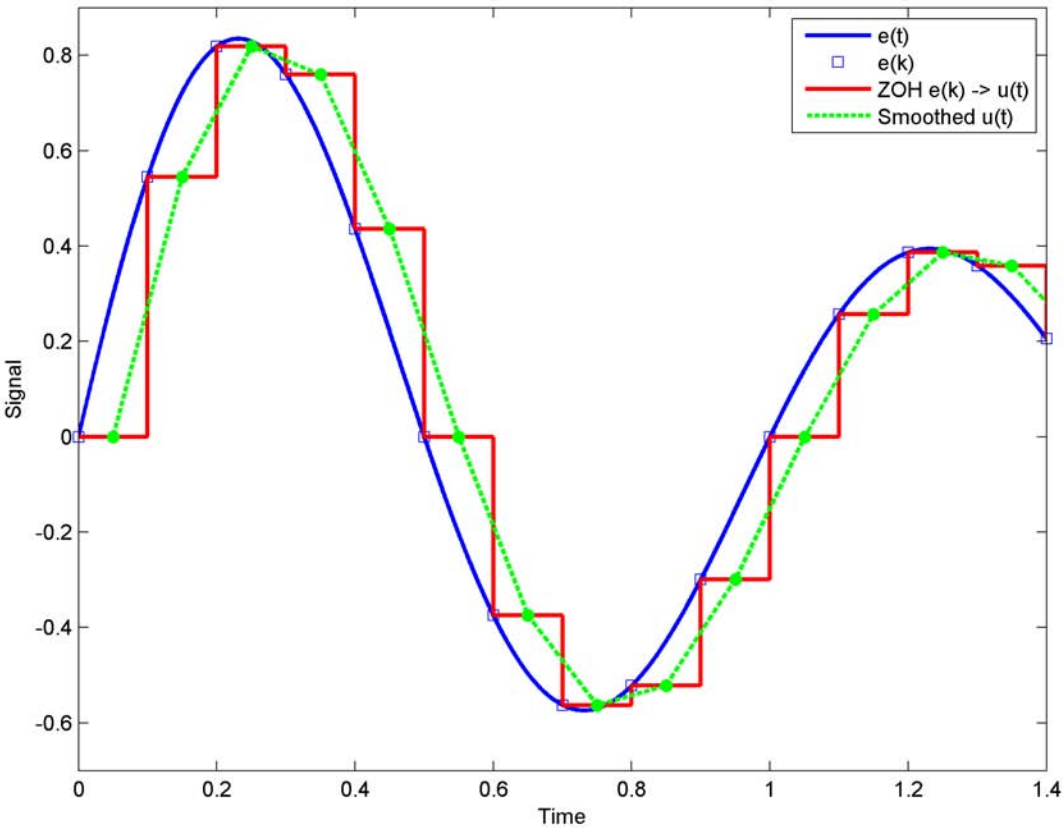


Image by MIT OpenCourseWare.

Sample and Hold Analysis

- Can actually analyze the transfer function $U(s)/E(e)$ analytically
- Can also gain some insight by looking at basic signals



  - $u(k)$ has a standard box car shape
  - Smoothed $u(k)$ by connecting mid-points $\Rightarrow \hat{u}(t)$

- So sampled and held $e(t)$ looks like input $e(t)$, but delay is obvious.

- Analytic study of $U(s)/E(e)$ shows that effective delay of sample and hold is $T_s/2$ on average
  - Can be a problem if T_s is too large

- So why not just make T_s small?
  - Key point is that T_s is how long we have to compute the control command given the measurements received

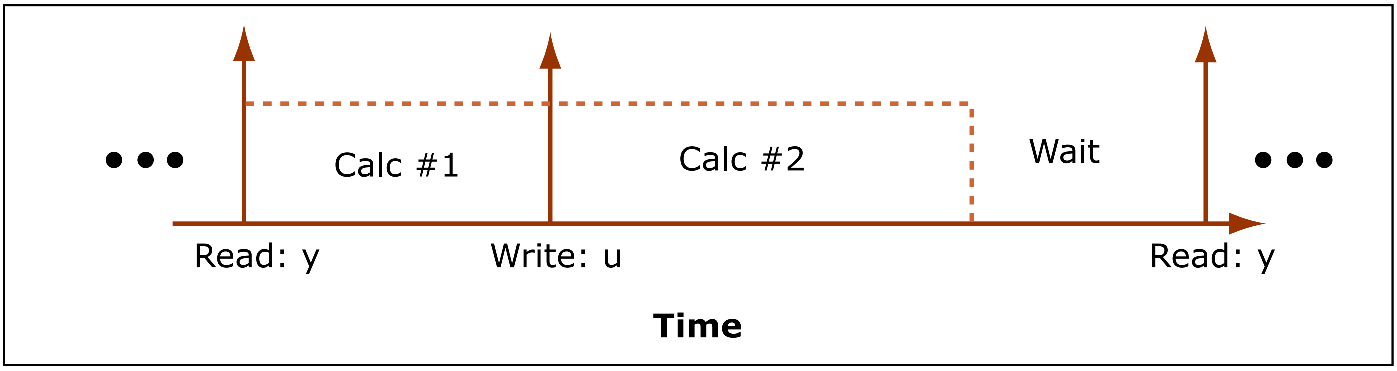


Image by MIT OpenCourseWare.

  - Usually `wait` period is short, but length of `calc 1` and `calc 2`, A/D and D/A operations depend on the complexity of the algorithm and the quality of the equipment
  - But quality ↑ ⇒ cost ↑↑
  - Typically would set sampling frequency $\omega_2 = \frac{2\pi}{T_s} \approx 15\omega_{BW}$

Control Law

- Basic compensator

$$G_c(s) = K_c \frac{s+z}{s+p} = \frac{U(s)}{E(s)}$$

- Equivalent differential equation form

$$\dot{u} + pu = K_c(\dot{e} + ze)$$

- Differential equation form not useful for a computer implementation - need to approximate the differentials

$$\dot{u}|_{t=kT_s} \approx \frac{1}{T_s}\left[u((k+1)T_s) - u(kT_s)\right] \equiv \frac{u_{k+1} - u_k}{T_s}$$

  - This uses the forward approximation, but others exist

- Then $\dot{u} + pu = K_c(\dot{e} + ze)$ approximately becomes

$$\frac{u_{k+1} - u_k}{T_s} + pu_k = K_c\left(\frac{e_{k+1} - e_k}{T_s} + ze_k\right)$$

or

$$u_{k+1} = (1 - pT_s)u_k + K_c e_{k+1} - K_c(1 - zT_s)e_k$$

which is a recursive *difference equation*, that can easily be implemented on a computer.

- Similar case for state space controllers

$$\begin{aligned} \dot{\mathbf{x}}_c &= A_c \mathbf{x}_c + B_c \mathbf{e} \\ \mathbf{u} &= C_c \mathbf{x}_c + D_c \mathbf{e} \end{aligned}$$

and $\dot{\mathbf{x}}_c \approx \frac{\mathbf{x}_c(k+1) - \mathbf{x}_c(k)}{T_s}$ so that

$$\begin{aligned} \frac{\mathbf{x}_c(k+1) - \mathbf{x}_c(k)}{T_s} &= A_c \mathbf{x}_c + B_c \mathbf{e} \\ \mathbf{x}_c(k+1) &= (I + T_s A_c)\mathbf{x}_c(k) + T_s B_c \mathbf{e}(k) \\ \mathbf{u}(k) &= C_c \mathbf{x}_c(k) + D_c \mathbf{e}(k) \end{aligned}$$

Computer Code Layout

- Given previous information u_k and e_k and new information y_{k+1} and r_{k+1}, form e_{k+1}
- Need to use the difference equation to find u_{k+1}

$$u_{k+1} = (1 - pT_s)u_k + K_c e_{k+1} - K_c(1 - zT_s)e_k$$

  - Then let $u_{old} = (1 - pT_s)u_k - K_c(1 - zT_s)e_k$, so that

$$u_{k+1} = K_c e_{k+1} + u_{old}$$

  - Also define constants $\gamma_1 = (1 - pT_s)$ and $\gamma_2 = -K_c(1 - zT_s)$

- Then the code layout is as follows:

initialize u_{old}

while $k < 1000$ **do**
  $k = k + 1$
  sample A/D's (read y_{k+1}, r_{k+1})
  compute $e_{k+1} = r_{k+1} - y_{k+1}$
  update $u_{k+1} = K_c e_{k+1} + u_{old}$
  output to D/A (write u_{k+1})

  update $u_{old} = \gamma_1 u_{k+1} + \gamma_2 e_{k+1}$
  wait

end while

- Note that this outputs the control as soon after the data read as possible to reduce the delay
  - result is that the write time might be unknown
  - Could write u_{k+1} at end of wait – delay is longer, but fixed

Summary

- Using a digital computer introduces some extra delay
  - Sample and hold $\approx T_s/2$ delay
  - Holding $u(k)$ to end of loop $\approx T_s$ delay
  - So the delay is $\approx T_s/2$–$3T_s/2$

- Emulation process – design the continuous control accounting for an extra
$$\frac{\omega_c T_s}{2} \cdot \frac{180^\circ}{\pi} = \frac{\omega_c 2\pi}{2\omega_s} \cdot \frac{180^\circ}{\pi} = \frac{\omega_c}{\omega_s} 180^\circ$$
to the PM to account for the delay.

- With $\omega_s \approx 15\omega_{BW}$, delay effects are small, and the cts and discrete controllers are similar

- `c2dm.m` provides simple ways to discretize the continuous controllers
  - Lots of different conversion approaches depending on what properties of the continuous controller you want to preserve.

16.30 / 16.31 Feedback Control Systems
Fall 2010