

# Lecture 4 – PID Control

## Continuous Time

90% (or more) of control loops in industry are PID

- PI
- PD
- PID
- Robustness

# PI control

- First-order system:

$$\tau \dot{y} = -y + u + d$$

- P control + integrator for cancelling steady state error

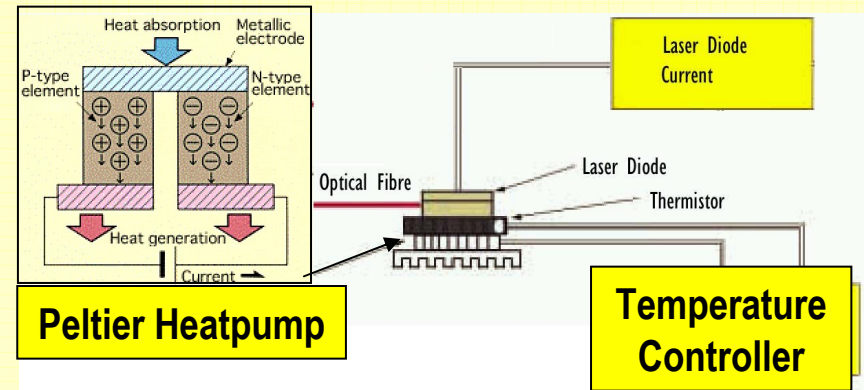
$$e = y - y_d;$$

$$\dot{v} = e$$

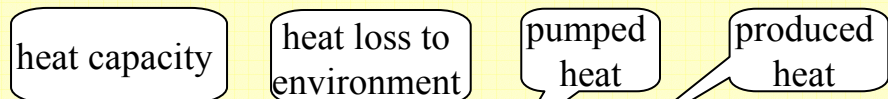
$$u = -k_I v - k_P e$$

## Example:

- WDM laser-diode temperature control



$$y(t) = \text{temperature} - \text{ambient temperature}$$



$$\tau \dot{y} = -y + u + d$$

- Other applications

- ATE
- EDFA optical amplifiers
- Fiber optic laser modules
- Fiber optic network equipment

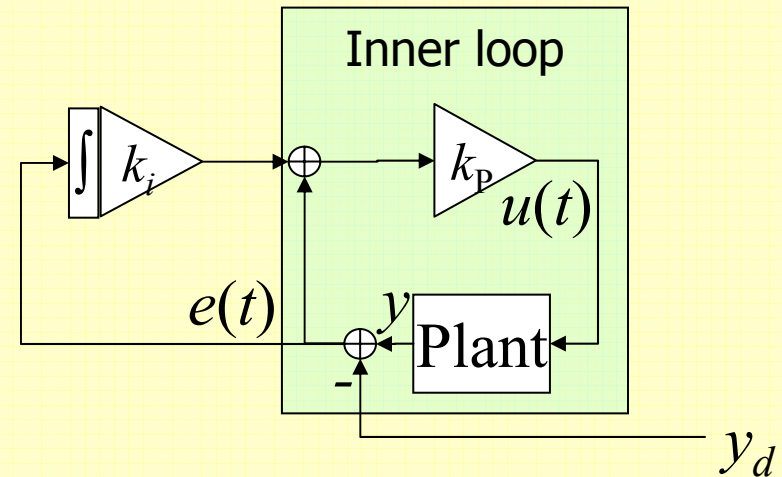
# PI control

- P Control
- + Integrator for canceling steady state error

$$e = y - y_d$$

$$u = -k_p e - k_i \int e dt$$

Cascaded loop interpretation:



$$u = k_p (e - e_d)$$

$$e_d = k_i \int e dt$$

$$k_I = k_i \cdot k_P$$

# PI control

- Closed-loop dynamics

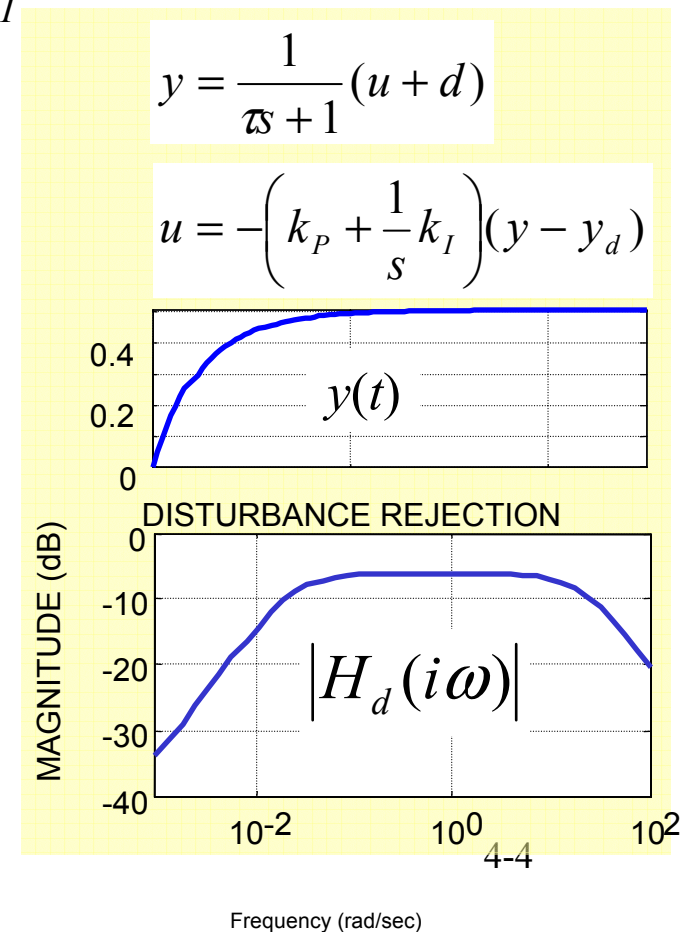
$$y = \frac{sk_P + k_I}{s(\tau s + 1) + sk_P + k_I} y_d + \frac{s}{s(\tau s + 1) + sk_P + k_I} d$$

- Steady state ( $s = 0$ ):  $y_{SS} = y_d$ .  
No steady-state error!
- Transient dynamics: look at the characteristic equation

$$\tau\lambda^2 + (1 + k_P)\lambda + k_I = 0$$

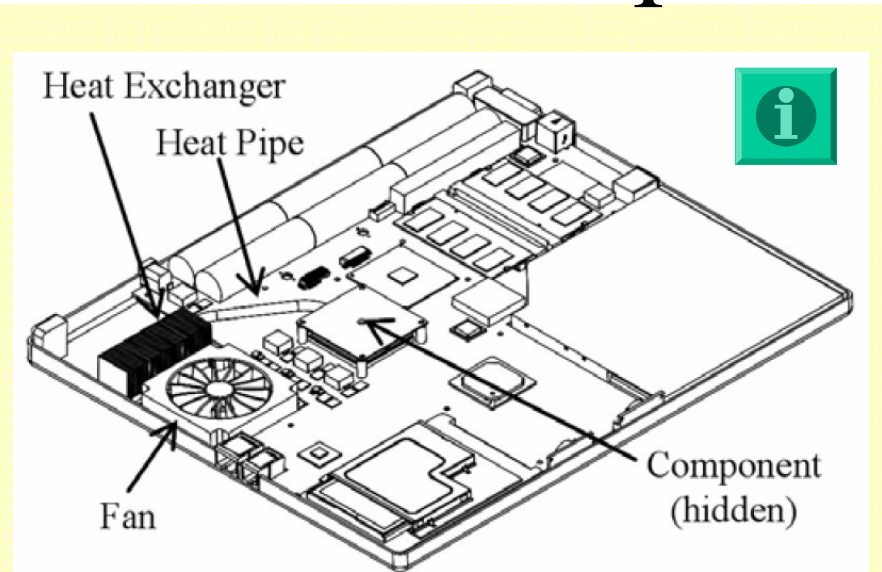
- Disturbance rejection

$$|\hat{y}(i\omega)| = |H_d(i\omega)| \cdot |\hat{d}(i\omega)|$$



# Processor thermal control example

- Thermal control
  - present in most new generation microprocessors: Intel, AMD,...
- Linear control of fan speed
- Minimize acoustic emission and fan power expenditure
- Keep processor cool enough
- Model
  - $\tau_1$  heat pipe time constant
  - $\tau_2$  heat exchanger time constant



Picture from Samson et al,  
Intel Technology Journal, February 2005

Analysis model:  $y = H(s)u + d$

$$H(s) = \frac{g}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

# Processor thermal control cont'd

- Simplified design model

- fit step response

- integrator  $y = \frac{0.4}{50s} u + d$

- P-control

$$u = -k_P(y - v)$$

$$k_P = 8 \quad T_P = 50 / (0.4k_P) = 15.6$$

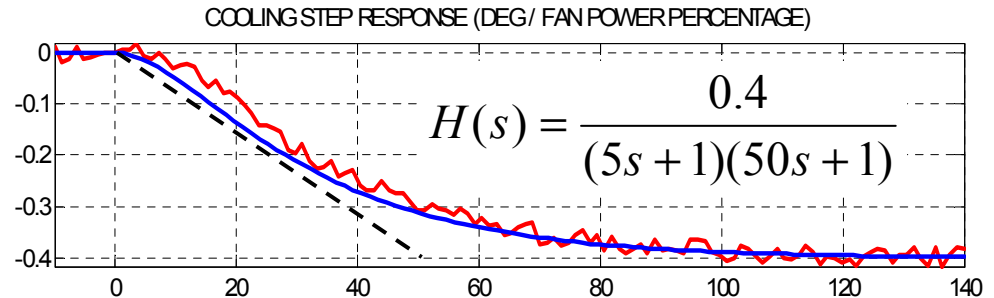
- I-control

- Designed as cascaded to P

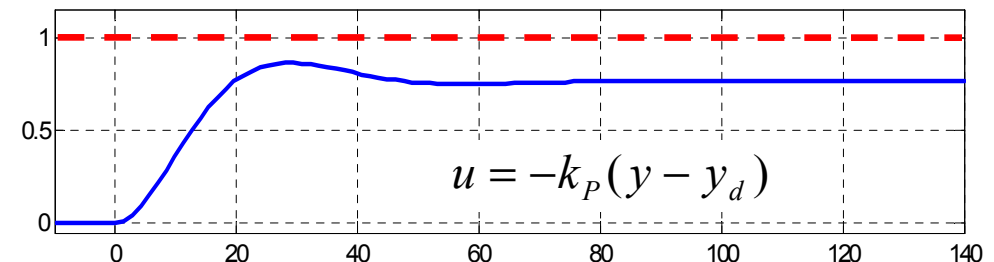
$$\dot{v} = -k_i(y - y_d)$$

$$k_i = 0.025; \quad k_I = k_i k_P = 0.2$$

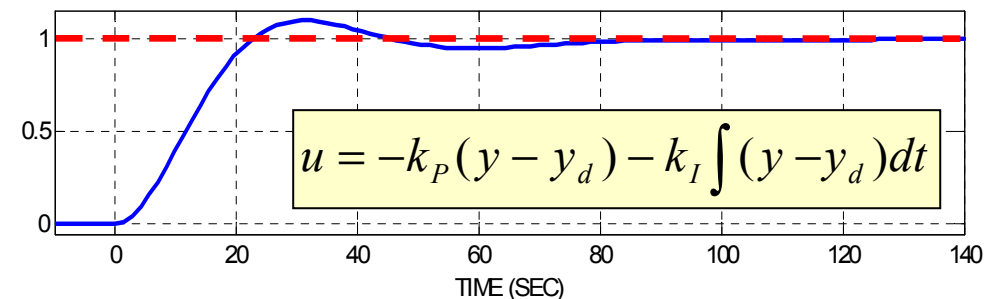
$$T_I = 1 / k_i = 50$$



P-CONTROL



PI-CONTROL



# Phase Locked Loop (PLL) Example

- Phase-locked loop is arguably a most prolific feedback system

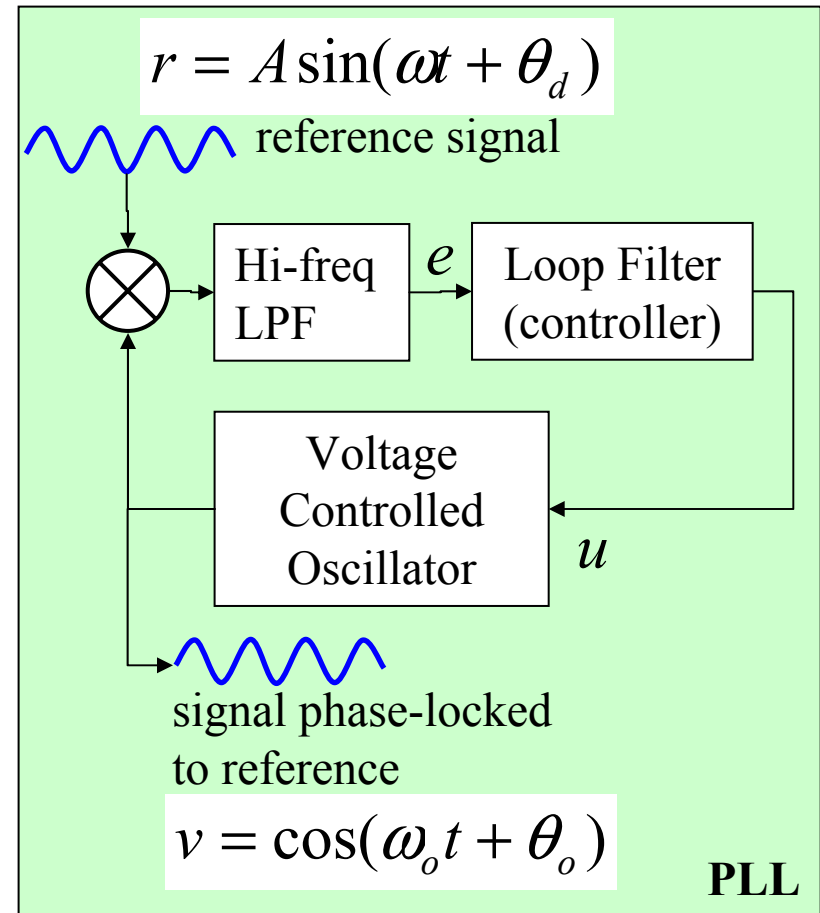
$$e = 2K_m \text{LPF}\langle r \cdot v \rangle$$

$$\begin{aligned} & \text{LPF}\langle 2 \sin(\omega t + \theta_d) \cdot \cos(\omega_o t + \theta_o) \rangle \\ &= \text{LPF}\langle \sin[(\omega - \omega_o)t + \theta_d - \theta_o] \rangle \\ & \quad - \underbrace{\text{LPF}\langle \sin[(\omega + \omega_o)t + \theta_d + \theta_o] \rangle}_{\approx 0} \end{aligned}$$

$$e \approx AK_m \sin(\Delta\omega_o t + \theta_d - \theta_o)$$

$$\Delta\omega = \omega - \omega_o$$

$$\dot{\theta}_o = \Delta\omega_o = K_o u$$



# PLL Model

- Small-signal model:

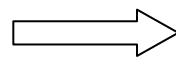
$$\theta = \omega t - \omega_o t + \theta_d - \theta_o \ll 1$$

$$e = K_d \sin(\theta) \approx K_d \theta$$

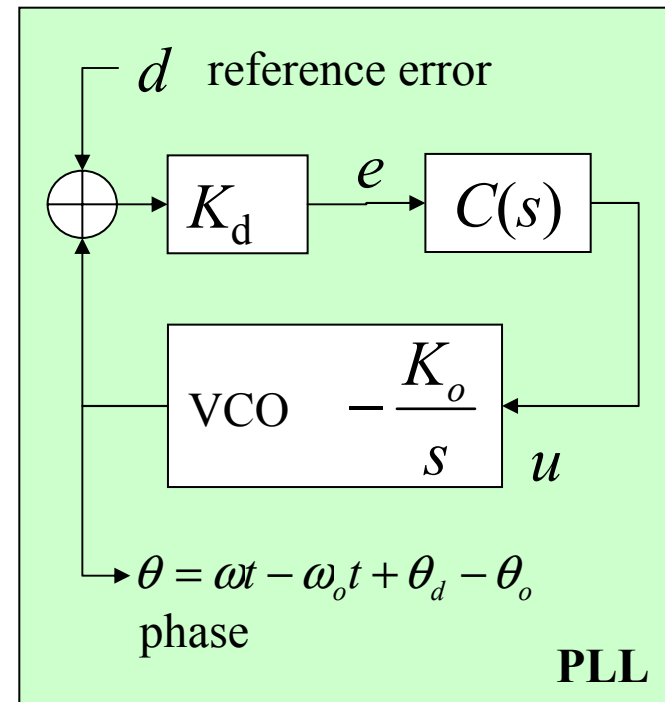
$$\dot{\theta} = \underbrace{\omega - \omega_o}_d + \dot{\theta}_d - \underbrace{K_o u}_{\theta_o}$$

- Loop dynamics:

$$\begin{aligned} \dot{\theta} &= d - K_o u \\ e &= K_d \theta \\ u &= k_P e + k_I \int e \cdot dt \end{aligned}$$



$$\theta = \frac{s}{s^2 + K_o K_d k_P s + k_I} d$$



# PD control

- 2-nd order dynamics

$$\ddot{y} = u + d$$

- PD control

$$e = y - y_d$$

$$u = -k_D \dot{e} - k_P e$$

- Closed-loop dynamics

$$\ddot{e} + k_D \dot{e} + k_P e = d$$

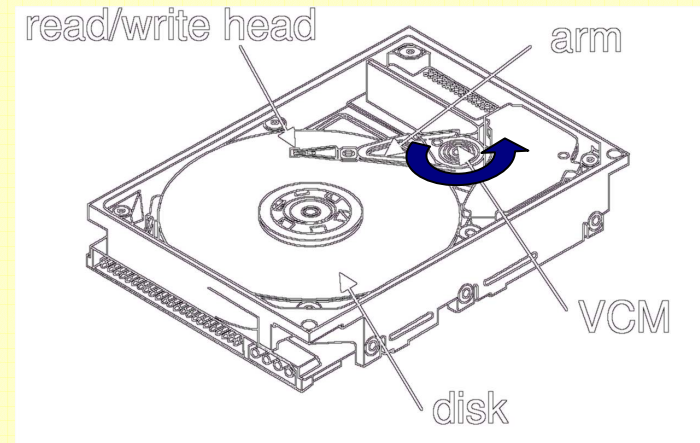
$$e = \frac{1}{s^2 + k_D s + k_P} d$$

- Optimal gains (critical damping)

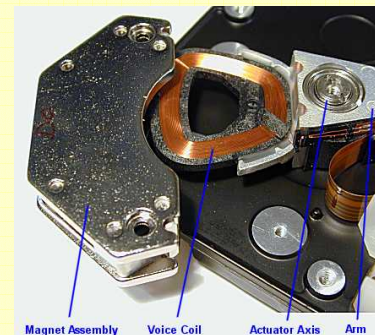
$$k_D = 2\tau; \quad k_P = \tau^2$$

## Example:

- Disk read-write head control



$$J\ddot{\phi} = T_{VCM} + T_{DISTURB}$$



Voice  
Coil  
Motor

# PD control

- Derivative (rate of  $e$ ) can be obtained
  - speed sensor (tachometer)
  - low-level estimation logic
- Signal differentiation, see Lecture 3
  - is noncausal
  - amplifies high-frequency noise

- Causal (low-pass filtered) estimate of the derivative

$$\dot{e} \approx \frac{s}{\tau_D s + 1} e = \frac{1}{\tau_D} e - \frac{1}{\tau_D} \cdot \frac{1}{\tau_D s + 1} e$$

- Modified PD controller, with estimated derivative:

$$u = -k_D \frac{s}{\tau_D s + 1} e - k_P e$$

# PD control performance

- The performance seems to be infinitely improving for
$$k_D = 2\tau; k_P = \tau^2; \quad \tau \rightarrow \infty$$
- This was a simple design model, remember?
- Performance is limited by
  - system being different from the model
    - flexible modes, friction, VCM inductance
  - sampling in a digital controller
  - rate estimation would amplify noise if too aggressive
  - actuator saturation
  - you might really find *after* you have tried to push the performance
- If high performance is really that important, careful application of more advanced control approaches might help

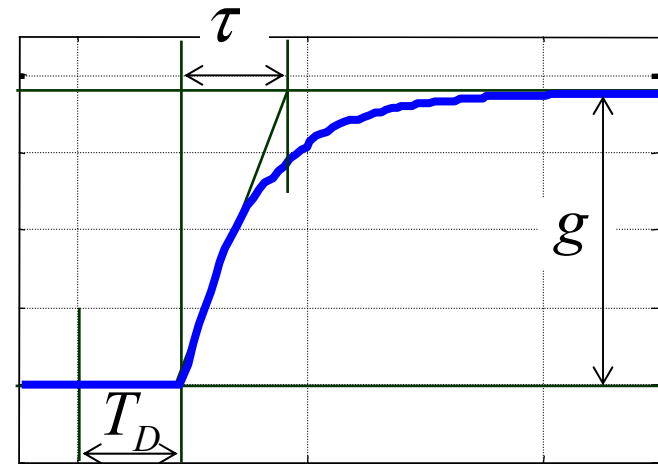
# Plant Type

- Constant gain - I control
- Integrator - P control
- Integrator+disturbance – PI control
- Double integrator - PD control
- Generic second order dynamics - PID control

# PID Control

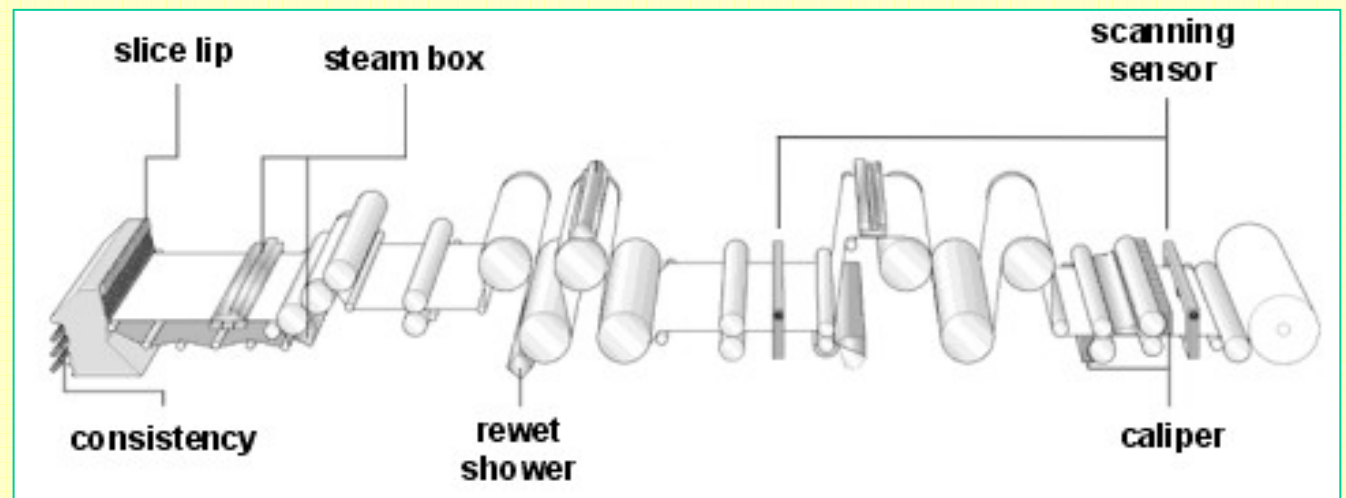
- Generalization of P, PI, PD
- Early motivation: control of first order processes with deadtime

$$y = \frac{ge^{-T_D s}}{\tau s + 1} u$$



## Example:

- Paper machine control



# PID Control

- PID: three-term control

$$e = y - y_d$$

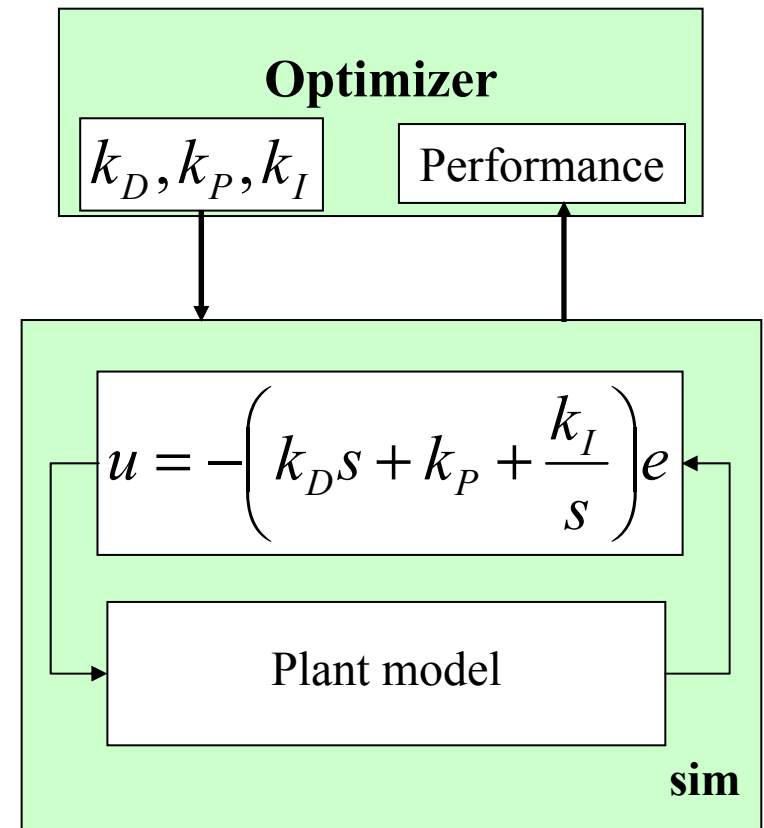
$$u = -k_D \dot{e} - k_P e - k_I \int e \cdot dt$$

- Rate of change in D-term can come from an independent sensor or an estimate (differentiating filter, see Lecture 3)

$$\dot{e} \approx \frac{s}{\tau_D s + 1} e$$

# Tuning PID Control

- Trial and error tuning
  - Design a PI controller
  - Add D term and find a good D gain
  - Tinker with P and I gains
- Model-based tuning
- Formulate performance index
- Numerical optimization
  - For given parameters run a sim, compute performance parameters and a performance index
  - Optimize the performance index over the three PID gains using grid search or Nelder method.



# Zeigler-Nichols tuning rule

- Explore the plant:
  - set the plant under P control and start increasing the gain till the loop oscillates
  - note the critical gain  $k_C$  and oscillation period  $T_C$

- Tune the controller:

	$k_P$	$k_I$	$k_D$
<b>P</b>	$0.5k_C$	—	—
<b>PI</b>	$0.45k_C$	$1.2k_P/T_C$	—
<b>PID</b>	$0.5k_C$	$2k_P/T_C$	$k_P T_C/8$

- Z and N used a Monte Carlo method to develop the rule
- Z-N rule enables tuning if a model and a computer are both unavailable, only the controller and the plant are.

# Integrator anti wind-up

- In practice, control authority is always limited:
  - $u_{\text{MIN}} \leq u(t) \leq u_{\text{MAX}}$
- Wind up of the integrator:
  - if  $|u_c| > u_{\text{MAX}}$  the integral  $v$  will keep growing while the control is constant. This results in a heavy overshoot later
- Anti wind-up:
  - switch the integrator off if the control has saturated

No anti wind-up

$$\begin{aligned} \dot{v} &= e \\ u_c &= -k_I v - k_P e \\ u &= \begin{cases} u_{\text{MAX}}, & u_c > u_{\text{MAX}} \\ u_c, & u_{\text{MIN}} \leq u_c \leq u_{\text{MAX}} \\ u_{\text{MIN}}, & u_c < u_{\text{MIN}} \end{cases} \end{aligned}$$

Anti wind-up

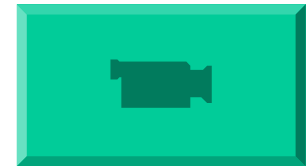
$$\dot{v} = \begin{cases} e, & \text{for } u_{\text{MIN}} \leq u_c \leq u_{\text{MAX}} \\ 0, & \text{if } u_c > u_{\text{MAX}} \text{ or } u_c < u_{\text{MIN}} \end{cases}$$

# Robustness

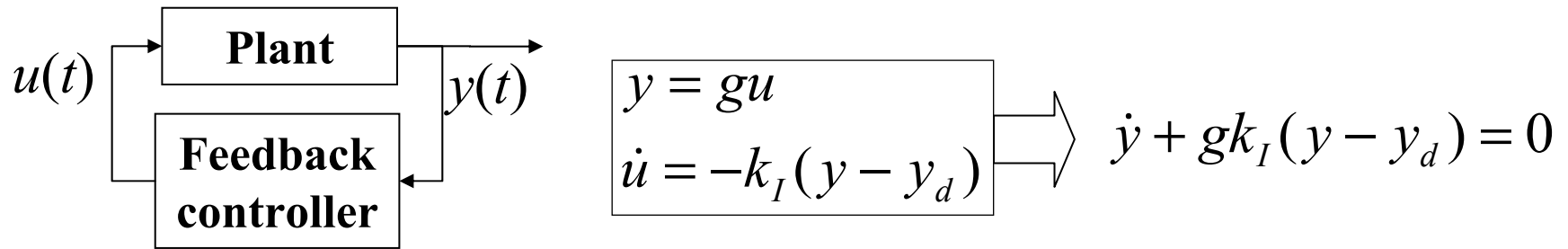
- Ok, we have a controller that works for a *nominal* model.
- Why would it ever would work for *real system*?
  - Will know for sure only when we try - V&V - similar to debugging process in software
- Can check that controller works for a *range* of different models and hope that the real system is covered by this range
  - This is called robustness analysis, robust design
  - Was an implicit part of the classical control design - Nyquist, Bode
  - Multivariable robust control - Honeywell: G.Stein, G.Hartmann, '81
  - Doyle, Zames, Glover - robust control theory

# Modeling uncertainty

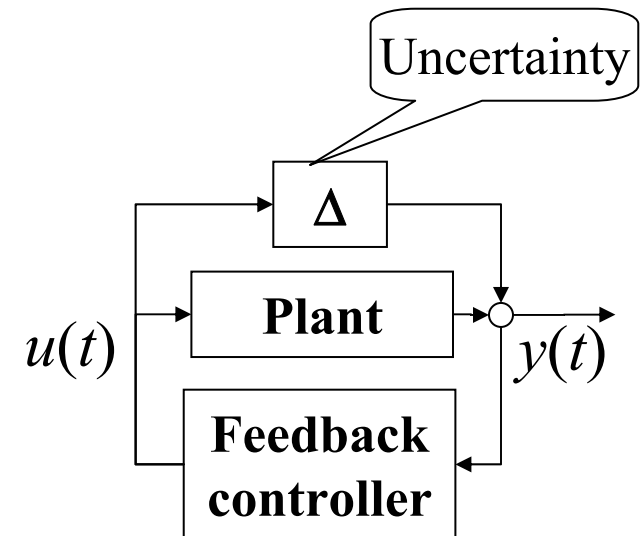
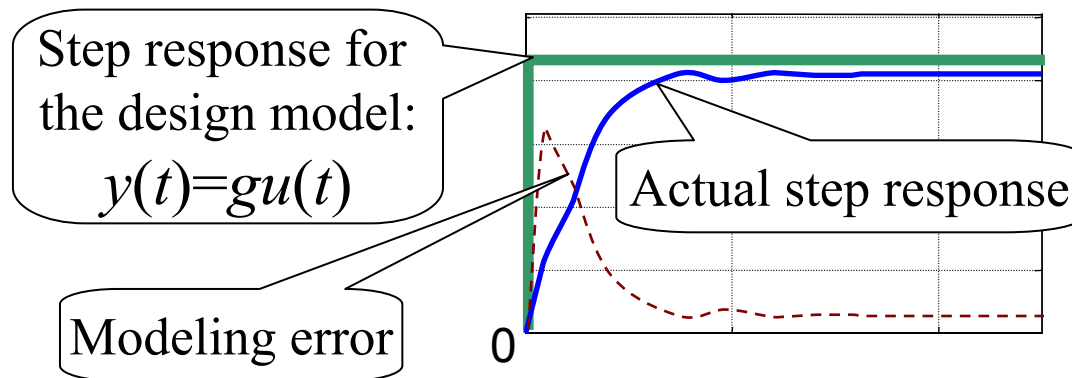
- Modeling uncertainty:
  - unknown signals
  - model errors
- Controllers work with real systems:
  - Signal processing: data  $\rightarrow$  algorithm  $\rightarrow$  data
  - Control: algorithms in a loop with *a real* system
  - It can blow up! (and sometimes it does)
- BIG question: Why controller designed for a model would *ever* work with a *real* system?
  - Robustness, gain and phase margins,
  - Control design model, vs. control analysis model
  - Monte-Carlo analysis



# Control loop analysis



- Why control might work if the process differs from the model?
- Key factors
  - modeling error (uncertainty) characterization
  - time scale (bandwidth) of the control loop



# Robustness - Small gain theorem

- Nonlinear uncertainty!

- Operator gain

$$\|Gu\| \leq \|G\| \cdot \|u\|$$

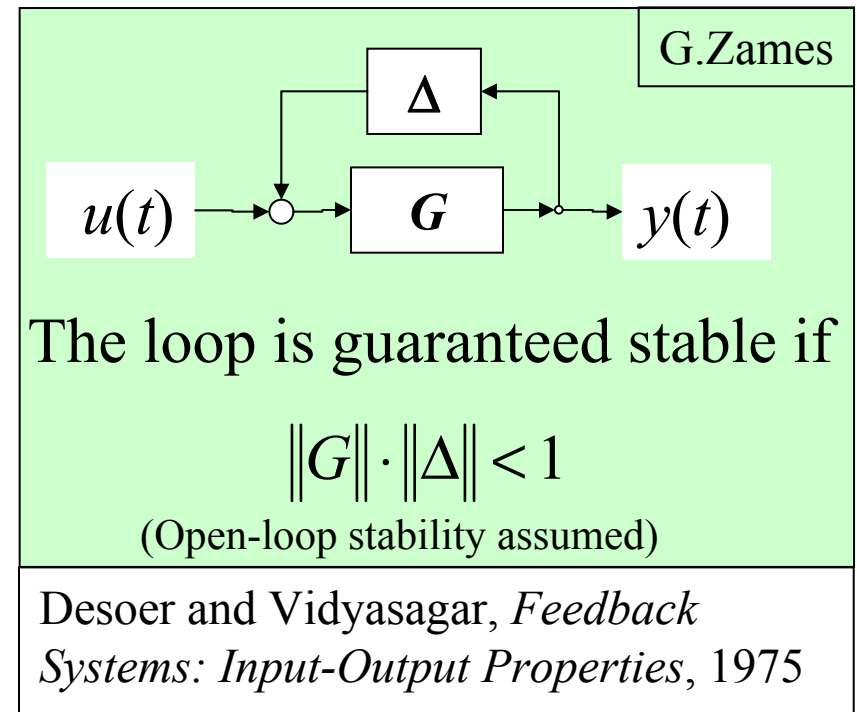
- $G$  can be a nonlinear operator

- $L_2$  norm

$$\|u\|^2 = \int u^2(t) dt = \frac{1}{2\pi} \int |\tilde{u}(i\omega)|^2 d\omega$$

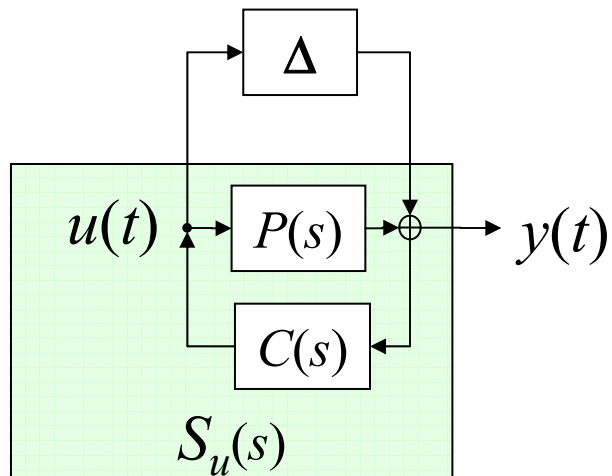
- $L_2$  gain of a linear operator

$$\|Gu\|^2 = \frac{1}{2\pi} \int |G(i\omega)\tilde{u}(i\omega)|^2 d\omega \leq \underbrace{\sup(|G(i\omega)|^2)}_{\|G\|^2} \cdot \underbrace{\frac{1}{2\pi} \int |\tilde{u}(i\omega)|^2 d\omega}_{\|u\|^2}$$



# Robustness

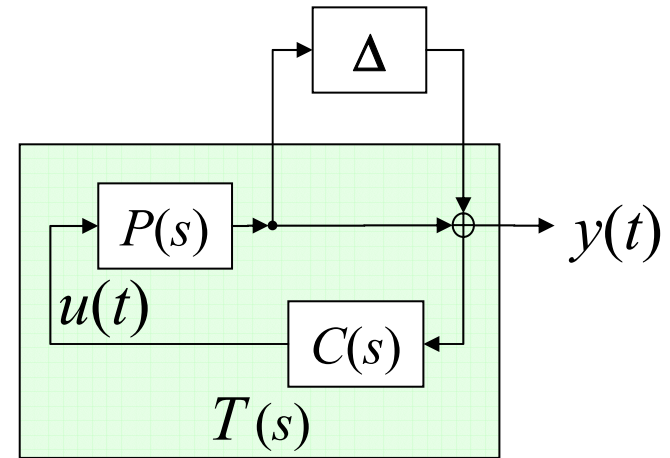
- Additive uncertainty



Condition of robust stability

$$\underbrace{\left| \frac{C(i\omega)}{1 + P(i\omega)C(i\omega)} \right|}_{\|S_u\|} \cdot \underbrace{|\Delta(i\omega)|}_{\|\Delta\|} < 1$$

- Multiplicative uncertainty



Condition of robust stability

$$\underbrace{\left| \frac{P(i\omega)C(i\omega)}{1 + P(i\omega)C(i\omega)} \right|}_{\|T\|} \cdot \underbrace{|\Delta(i\omega)|}_{\|\Delta\|} < 1$$