# MM4 Digital PID Control

Readings:

- http://lorien.ncl.ac.uk/ming/digicont/digimath/dpid1.htm



## What have we talked in Classical Control **MM6 & MM7**?

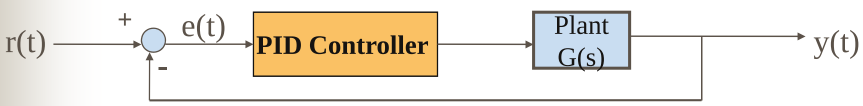
- MM6: PID Controllers
- MM7: Some Practical Issues Using PID Controllers

# Definition of PID Controllers

- **PID Means:**
  - **P**: Proportional (control) $u(t) = Ke(t)$
  - **I** : Integral (control) $u(t) = \frac{K}{T_I}\int_{t_0}^{t} e(\tau)d\tau$
  - **D**: Derivative (control) $u(t) = KT_D\dot{e}(t)$

$$u(t) = K_p(e(t) + \frac{1}{T_I}\int_{t_0}^{t} e(\tau)d\tau + T_D\dot{e}(t))$$

- **PID Control System Structure: cascade control**






# PID Control: **Characteristic Summary**

❑ A proportional controller (Kp) will have the effect of reducing the rise time and will reduce ,but never eliminate, the steady-state error
❑ An integral control (Ki) will have the effect of eliminating the steady-state error, but it may make the transient response worse.
❑ A derivative control (Kd) will have the effect of increasing the stability of the system, reducing the overshoot, and improving the transient response.

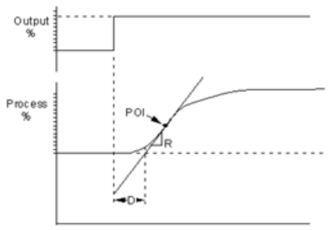
| CL RESPONSE | RISE TIME | OVERSHOOT | SETTLING TIME | S-S ERROR |
|---|---|---|---|---|
| **Kp** | Decrease | Increase | Small Change | Decrease |
| **Ki** | Decrease | Increase | Increase | Eliminate |
| **Kd** | Small Change | Decrease | Decrease | Small Change |

http://www.jashaw.com/pid/tutorial/pid6.html

## Tuning PID: **Quarter decay ratio method(V)**

- Another means of determining parameters based on the ZN open loop is: After "bumping" the output, watch for the point of **inflection** and note:

| | | |
|---|---|---|
| Ti | min | Time from output change to POI |
| P | % | Process value change at POI |
| R | %/min | Rate of change at POI |
| X | % | Change in output |



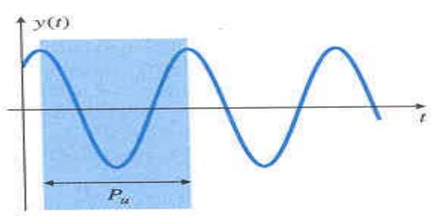

- D is calculated:
  D=Ti - P/R

- D & X are then used ------------→

| | Gain | Reset | Derivative |
|---|---|---|---|
| P | X/DR | — | — |
| PI | 0.9X/DR | 0.3/D | — |
| PID | 1.2X/DR | 0.5/D | 0.5D |



## Tuning PID: **Ultimate Sensitivity Method(II)**




Closed loop tuning method

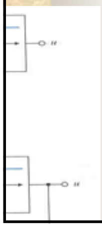
Ziegler–Nichols Tuning for the Regulator $D(s) = K(1 + 1/T_I s + T_D s)$, Based on a Stability Boundary

| *Type of Controller* | *Optimum Gain* |
|---|---|
| Proportional | $K = 0.5K_u$ |
| PI | $K = 0.45K_u$<br>$T_I = 1/1.2P_u$ |
| PID | $K = 0.6K_u$<br>$T_I = \frac{1}{2}P_u$<br>$T_D = \frac{1}{8}P_u$ |

# Content for MM7

Some **practical issues** when developing a PID controler:

- Integral windup & Anti-windup methods
- Derivertive kick
- When to use which controller?
- Operational Amplifier Implementation
- Other tuning methods



$$u(t) = K\left(e(t) + \frac{1}{T_I}\int_{t_0}^{t} e(\tau)d\tau + T_D\dot{y}(t)\right)$$

$$U(s) = K\left(1 + \frac{1}{T_I s}\right)E(s) + T_D sY(s)$$



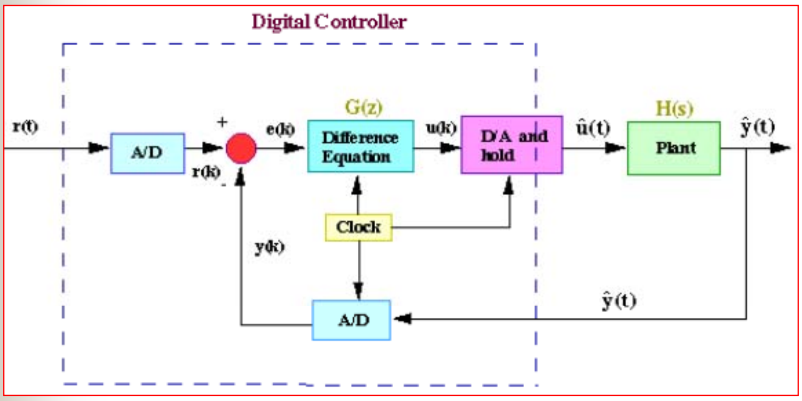
$$G(s) = \frac{Ke^{-\theta s}}{\tau s + 1} \quad \text{(1st order)}$$

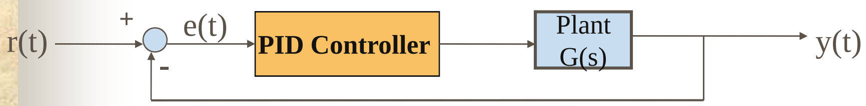
**Table 12.3 Controller Design Relations Based on the ITAE Performance Index and a First-Order plus Time-Delay Model**

| *Type of Input* | *Type of Controller* | *Mode* | *A* | *B* |
|---|---|---|---|---|
| Load | PI | P | 0.859 | −0.977 |
| | | I | 0.674 | −0.680 |
| Load | PID | P | 1.357 | −0.947 |
| | | I | 0.842 | −0.738 |
| | | D | 0.381 | 0.995 |
| Set point | PI | P | 0.586 | −0.916 |
| | | I | 1.03[b] | −0.165[b] |
| Set point | PID | P | 0.965 | −0.85 |
| | | I | 0.796[b] | −0.1465[b] |
| | | D | 0.308 | 0.929 |

[a]Design relation: $Y = A(\theta/\tau)^B$ where $Y = KK_c$ for the proportional mode, $\tau/\tau_I$ for the integral mode, and $\tau_D/\tau$ for the derivative mode.
[b]For set-point changes, the design relation for the integral mode is $\tau/\tau_I = A + B(\theta/\tau)$. [8]

# Digital PID Controllers?

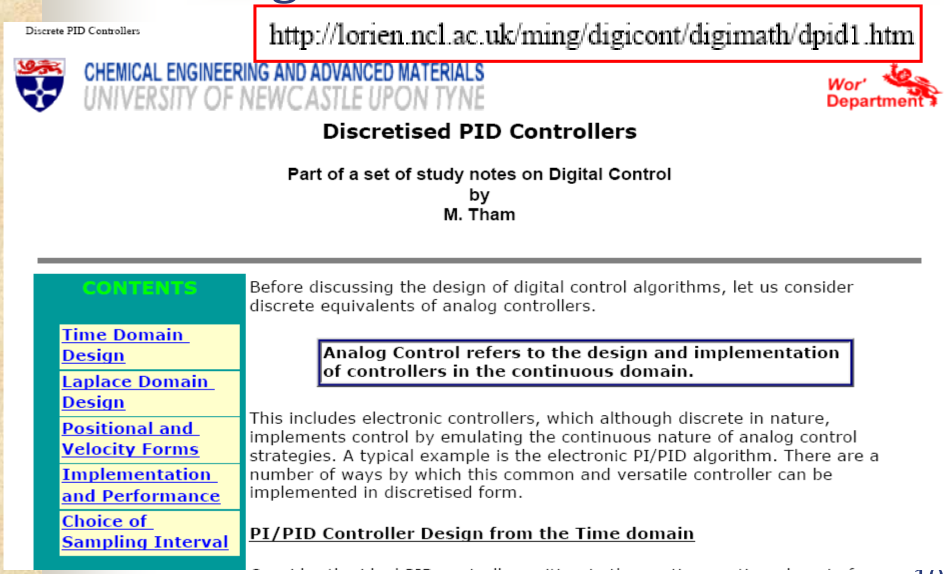





# Reading Material

http://lorien.ncl.ac.uk/ming/digicont/digimath/dpid1.htm



Discrete PID Controllers

CHEMICAL ENGINEERING AND ADVANCED MATERIALS
UNIVERSITY OF NEWCASTLE UPON TYNE

Wor' Department

**Discretised PID Controllers**

**Part of a set of study notes on Digital Control**
**by**
**M. Tham**

**CONTENTS**


- Time Domain Design
- Laplace Domain Design
- Positional and Velocity Forms
- Implementation and Performance
- Choice of Sampling Interval


Before discussing the design of digital control algorithms, let us consider discrete equivalents of analog controllers.

**Analog Control refers to the design and implementation of controllers in the continuous domain.**

This includes electronic controllers, which although discrete in nature, implements control by emulating the continuous nature of analog control strategies. A typical example is the electronic PI/PID algorithm. There are a number of ways by which this common and versatile controller can be implemented in discretised form.

**PI/PID Controller Design from the Time domain**

# Design from Time Domain

- Continuous-time PID controller:

$$u(t) = K_p(e(t) + \frac{1}{T_I}\int_{t_0}^{t} e(\tau)d\tau + T_D\dot{e}(t)) + u_0$$

- Discrete-time approximations:

$$\frac{1}{T_I}\int_{t_0}^{t} e(\tau)d\tau \approx \frac{T_s}{T_I}\sum_{i=0}^{[t]} e(t_i), \quad T_D\dot{e}(t) \approx T_D\frac{e(t_i)-e(t_{i-1})}{T_s}$$

- **Positional** Discrete PID controller:

$$u(t_k) = K_p(e(t_k) + \frac{T_s}{T_I}\sum_{j=0}^{t_k} e(t_j) + \frac{T_D}{T_s}(e(t_k)-e(t_{k-1}))) + u_0$$

$$u(k) = K_p(e(k) + \frac{T_s}{T_I}\sum_{j=0}^{k} e(j) + \frac{T_D}{T_s}(e(k)-e(k-1))) + u_0$$



# Design from Laplace Domain (I)

- Laplace Transformed PID controller:

$$U(s) = K_p(E(s) + \frac{E(s)}{T_I s} + T_D s E(s))$$

- Backward difference transformation

**Converting to $z$-domain**

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

**Forward rectangular rule**

$$s \approx \frac{z-1}{T}$$

**Backward rectangular rule**

$$s \approx \frac{z-1}{Tz}$$

**Trapezoidal rule**

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

## Design from Laplace Domain (II)

- **Velocity** Discrete PID controller:

$$s = \frac{z-1}{T_s z} = \frac{1-z^{-1}}{T_s}$$

$$U(s) = K_p(E(s) + \frac{E(s)}{T_I s} + T_D s E(s))$$

$$U(z) = K_p E(z)(1 + \frac{T_s}{T_I(1-z^{-1})} + \frac{T_D}{T_s}(1-z^{-1}))$$

$$U(z)(1-z^{-1}) = K_p E(z)((1-z^{-1}) + \frac{T_s}{T_I} + \frac{T_D}{T_s}(1-z^{-1})^2)$$

$$u(k) = u(k-1) + K_p(e(k)-e(k-1)) + \frac{K_p T_s}{T_I}e(k) + \frac{K_p T_D}{T_s}(e(k) - 2e(k-1) + e(k-2))$$

$$u(k) = u(k-1) + (K_p + \frac{K_p T_s}{T_I} + \frac{K_p T_D}{T_s})e(k) + (-K_p - \frac{2K_p T_D}{T_s})e(k-1) + \frac{K_p T_D}{T_s}e(k-2)$$



## Comparison of Position and Velocity PID

$$u(t_k) = K_p(e(t_k) + \frac{T_s}{T_I}\sum_{j=0}^{t_k} e(t_j) + \frac{T_D}{T_s}(e(t_k) - e(t_{k-1}))) + u_0$$

$$u(k) = K_p(e(k) + \frac{T_s}{T_I}\sum_{j=0}^{k} e(j) + \frac{T_D}{T_s}(e(k) - e(k-1))) + u_0$$

$$u(k) = u(k-1) + K_p(e(k)-e(k-1)) + \frac{K_p T_s}{T_I}e(k) + \frac{K_p T_D}{T_s}(e(k) - 2e(k-1) + e(k-2))$$

$$u(k) = u(k-1) + (K_p + \frac{K_p T_s}{T_I} + \frac{K_p T_D}{T_s})e(k) + (-K_p - \frac{2K_p T_D}{T_s})e(k-1) + \frac{K_p T_D}{T_s}e(k-2)$$

$$u(k) = K_p(e(k) + \frac{T_s}{T_I}\sum_{j=0}^{k} e(j) + \frac{T_D}{T_s}(e(k) - e(k-1))) + u_0$$

$$u(k-1) = K_p(e(k-1) + \frac{T_s}{T_I}\sum_{j=0}^{k-1} e(j) + \frac{T_D}{T_s}(e(k-1) - e(k-2))) + u_0$$

$$u(k) - u(k-1) = K_p(e(k)-e(k-1)) + \frac{K_p T_s}{T_I}e(k) + \frac{K_p T_D}{T_s}(e(k) - 2e(k-1) + e(k-2))$$

## Practice – Initial control $u_0$

- Initial control $u_0$ corresponds to the steady-state control output level
- It can be assumed to be any reasonable level

$$u(k) = K_p(e(k) + \frac{T_s}{T_I}\sum_{j=0}^{k} e(j) + \frac{T_D}{T_s}(e(k) - e(k-1))) + u_0$$

$$u(k) = u(k-1) + (K_p + \frac{K_p T_s}{T_I} + \frac{K_p T_D}{T_s})e(k) + (-K_p - \frac{2K_p T_D}{T_s})e(k-1) + \frac{K_p T_D}{T_s}e(k-2)$$



## Practice – Integral Windup

- Positional PID algorithm – Integral Windup problem

$$u(k) = K_p(e(k) + \frac{T_s}{T_I}\sum_{j=0}^{k} e(j) + \frac{T_D}{T_s}(e(k) - e(k-1))) + u_0$$

- Velocity PID algorithm does not have this problem!

$$u(k) = u(k-1) + (K_p + \frac{K_p T_s}{T_I} + \frac{K_p T_D}{T_s})e(k) + (-K_p - \frac{2K_p T_D}{T_s})e(k-1) + \frac{K_p T_D}{T_s}e(k-2)$$

# Practice – Tuning Methods

- All tuning methods for continuous-time PID control design apply
- Sometimes, a factor of $0.5T_s$ is usally added to the process dead-time to account for the delay caused by the sampler



# Practice – Sampling Interval

- Aliasing problem
- Emulation method: Sample as fast as possible
- However, too fast a sampling is wasteful of resources.
  - the cost of implementation will increase because more capable components must be installed
  - a DCS typically has many hundreds of input-output channels to administer. The functioning of the DCS will degrade significantly if every control loop is to sample at the highest frequency possible.
  - fast sampling intervals will mean that high frequency components such as noise will also be captured in the signal, and this is not necessarily beneficial to the performance of the control loop.

**The choice of an appropriate sampling interval should be based on the dynamics of the process being controlled.**