PID Control

Dr. Ir. Yeffry Handoko Putra, M.T

Session Outlines & Objectives

Outlines

- The PID control algorithms
- The practical aspects

Objectives

- Understand what the PID control is
- Know the functions of each PID control terms
- Be able to select the right combination of PID control element for various process control application objectives
- Know the additional features installed to the controller to be implemented in practice

What is PID Control?

- The PID stands for **P**roportional - **I**ntegrator - **D**erivative
- Also known as **three-term** control
- It's implemented as a computer program today
  - The controller comes in many **different** forms
  - PID control is often combined with logic, sequential functions, selectors, and simple function blocks to build the complicated automation systems

Why PID Control?

- The PID algorithm is simple, easy to understand, and relatively easier to tune than the other controller
  - It became the standard tool when process control emerged in the 1940s
  - In process control today, more than 90% of the control loops are of PID type

Key Concepts (1)

- The PID algorithm doesn't know the correct output to bring the process variable to the setpoint
  - The algorithm must have process measurement to perform
  - It merely continues to move the output in the direction which should move the process toward the setpoint
- The PID algorithm must be 'tuned' for the particular process loop
  - Each of the terms of the PID equation must be **understood**
  - The tuning is **based** on the dynamics of the process response

Key Concepts (2)

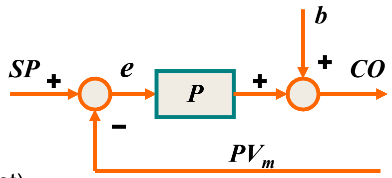
- Manual & Auto modes
  - Manual mode
    - The human operator adjusts the output to operate the process
    - Manual Mode is very useful when unusual conditions exist:
      - Plant start-up
      - Plant shut-down
      - Emergencies
  - Auto mode
    - The control algorithm manipulates the output to hold the process measurements at their setpoints
    - Should be the most common mode for normal operation

PID Control Algorithm

- Comprises three elements:
  - **P**roportional – also known as Proportional Gain or simply Gain
  - **I**ntegral – also known as Automatic Reset or simply Reset
  - **D**erivative – also known as Rate or Pre-Act
- Available in several combinations of these elements:
  - Proportional only (P)
  - Proportional and Integral (PI) (most common)
  - Proportional, Integral, and Derivative (PID)
  - Proportional and Derivative (PD)

Proportional Mode (1)

$$CO = P \cdot e + b$$



P = proportional gain (dimensionless)

CO = controller output (%)

b = bias (%) (also known as manual reset)

e = ($SP - PV_m$) (%) → "reverse action", or

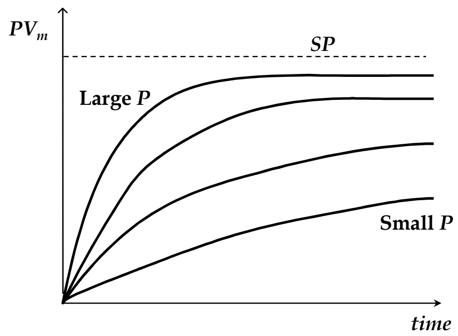
e = – ($SP - PV_m$) (%) → "direct action"

- Some manufacturers use **P**roportional **B**and (***PB***) instead of proportional gain
  - *PB* is the % change in the input which a 100% change in the output

$$PB = \frac{100}{P} \Rightarrow CO = P \cdot e + b = \frac{100}{PB} e + b$$

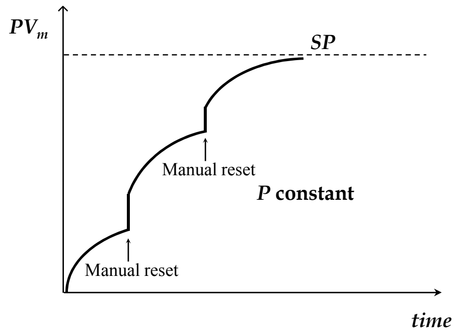
Proportional Mode (2)

- Proportional-only control can produces an **offset**
  - The offset can be **reduced** by **increasing** the controller **gain** (or **decreasing** the **proportional band**). But one cannot make the controller gain arbitrarily large since **too high a gain induces oscillation and/or instability**



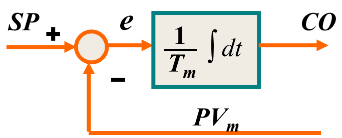
Proportional Mode (2)

- To **remove** offset, the human operator has to "**reset**" the controller manually by adjusting the value of the manual reset (the "b" term)



Integral (Automatic Reset) Mode (1)

$$CO = \frac{1}{T_m}\int e\,dt$$



T_m = integral time (minute per repeat or second per repeat)

CO = controller output (%)

e = ($SP - PV_m$) (%) → "reverse action" or

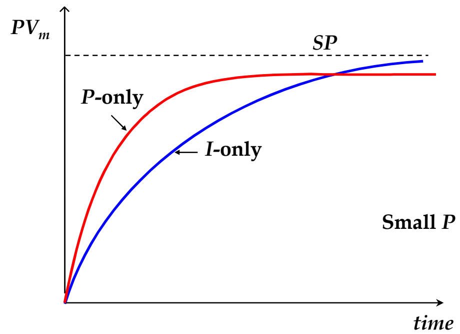
e = **–** ($SP - PV_m$) (%) → "direct action"

- Some manufacturers use **repeat per minute** (or **repeat per second**) instead of minute per repeat (or second per repeat)
  - Repeat per minute (or second) is the time it takes the reset (or integral) element to repeat (reset) the action of the proportional element

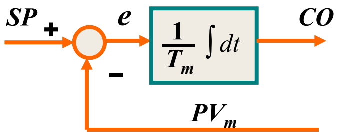
$$\text{repeat per minute } (T_r) = \frac{1}{\text{minute per repeat } (T_m)} \Rightarrow CO = T_r \int e\,dt$$

Integral (Automatic Reset) Mode (2)

- As long as error exists, the controller will change its output; hence it is capable of driving the error to zero
- Speed of response is reduced (compared to P-only mode)



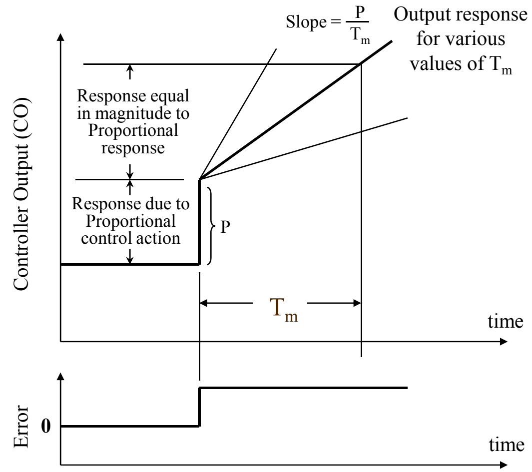
Proportional-Integral Mode (1)



$$CO = P\left(e + \frac{1}{T_m}\int e\,dt\right)$$

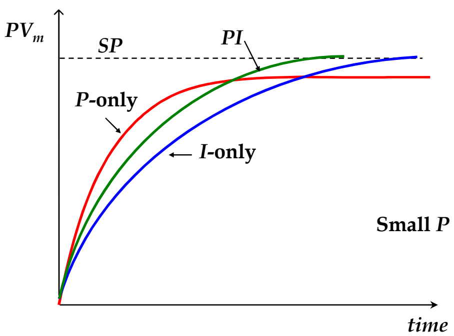
Proportional-Integral Mode (2)

- Response of PI controller to step change in error



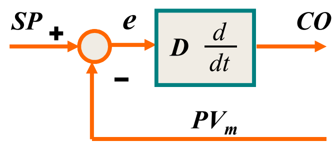
Proportional-Integral Mode (3)

- It combines the best features of the proportional and integral modes
  - The proportional offset is **eliminated** with **little loss** of response speed



Derivative Mode (1)

$$CO = D\,\frac{de}{dt}$$



D = derivative time (minute or second)

CO = controller output (%)

e = ($SP - PV_m$) (%) → "reverse action" or

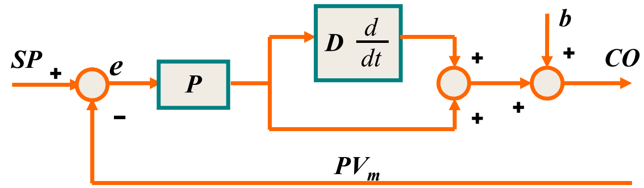
e = – ($SP - PV_m$) (%) → "direct action"

- Speed of response is increased (compared to P-only mode)
- Hypersensitive to noise and other high-frequency disturbances

Derivative Mode (2)

- A steady-state error signal, however, is not recognized by ***D*** controllers, because regardless of how big the error, **its rate of change is zero**. Therefore, derivative-only controllers are not used in practice
- They are usually found in combination with other control elements, mostly in combination with proportional control

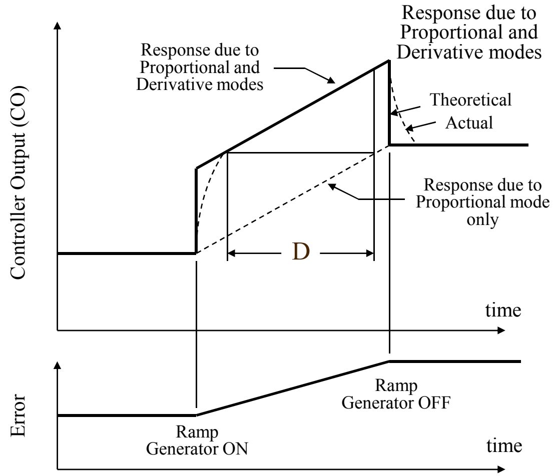
Proportional-Derivative Mode (1)



$$CO = P\left(e + D\frac{de}{dt}\right) + b$$

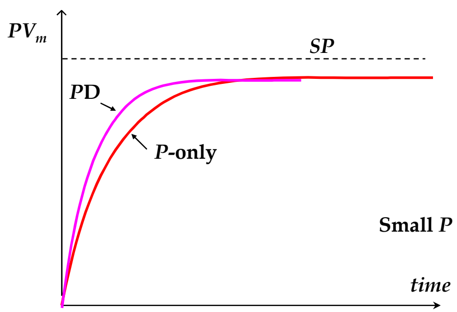
Proportional-Derivative Mode (2)

- Response of PD controller to ramp change in error



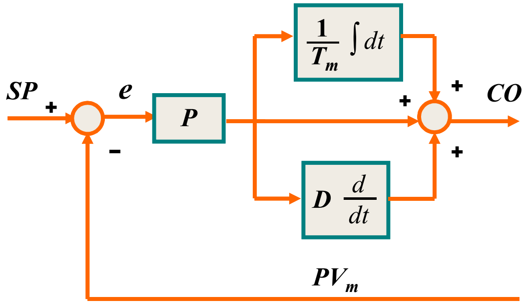
Proportional-Derivative Mode (3)

- PD control can produces an **offset**
- To avoid proportional offset, the bias "***b***" should be set when the PV_m is at setpoint
- Commonly found in slow-response process control, e.g. temperature, pH, composition controls



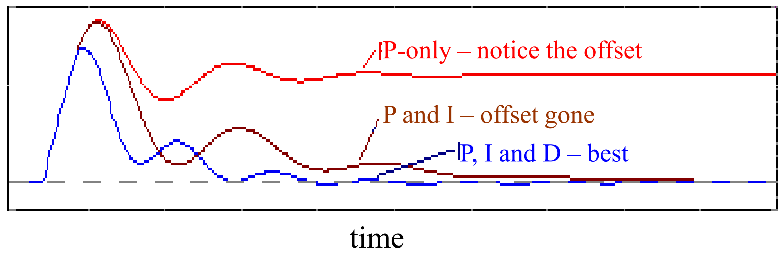
Proportional-Integral-Derivative Mode (1)

- Combine the best feature of P, I, and D terms



$$CO = P\left(e + \frac{1}{T_m}\int e\,dt + D\frac{de}{dt}\right)$$

Proportional-Integral-Derivative Mode (2)



Additional PID Concepts

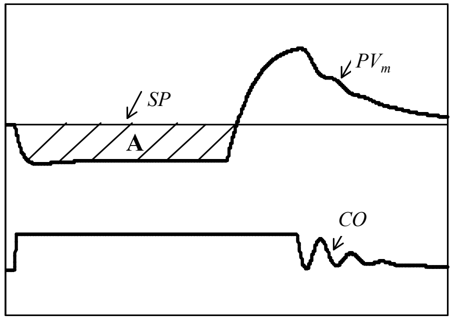
Interactive vs. Non-interactive PID Algorithm

- Refer to interaction between the reset and derivative terms
- Also known as ‘series’ or ‘parallel’
- Almost all analog controllers are interactive
- Many digital controller are non-interactive, some are interactive
- The only difference is in the tuning of controller with derivative

Reset Windup (1)

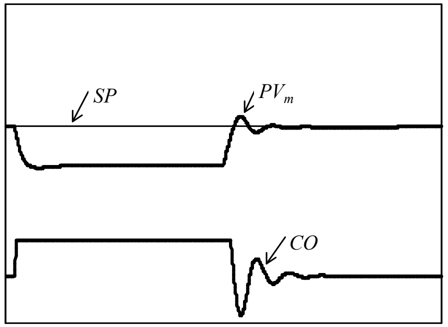
- All actuators have limitations:
  - Ex.: A motor has limited speed, a valve cannot be more than fully opened or fully closed
- Windup phenomena is caused by the interaction of integral action and saturations
- When this happens the feedback loop is broken and the system runs as an open loop because the actuator will remain at its limit independently of the process output
  - If a controller with integrating action is used, the error will continue to be integrated. This means that the integral term may become very large or, colloquially, it “winds up”
  - It is then required that the error has opposite sign for a long period before things return to normal
  - The consequence is that any controller with integral action may give large transients when the actuator saturates

Reset Windup (2)



- Note that controller output saturates causing area “A” to accumulate by the integral action
- After the disturbance returns to its normal level, the controller output remains saturated for a period of time causing an upset in PV_m

Anti-Reset Windup



- When the manipulated variable saturates, the integral is not allowed to accumulate
- When control returns, the controller takes immediate action and the process returns smoothly to the setpoint

Methods for Anti-Reset Windup

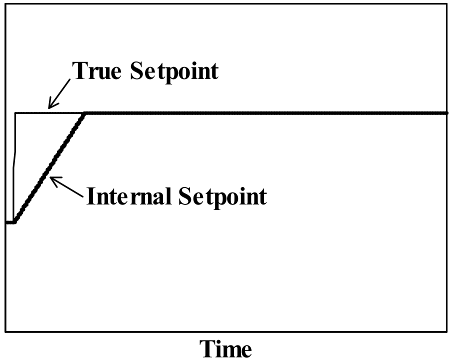
- Turn off the integral when a valve saturates or a control loop is not in use.
- Clamp the controller output to be greater than 0% and less than 100%.
- Apply internal reset feedback
- Apply external reset feedback

Bumpless Transfer (1)

- Practically all controllers can be run in two modes: **manual** or **automatic**
- When the system is in manual mode, the control algorithm produces a control signal that may be different from the manually generated control signal, or vice versa. It is necessary to make sure that the two outputs coincide at the time of switching. This is called ***bumpless transfer***
- With bumpless transfer, an internal setpoint is used for the controller and the internal setpoint is ramped at a slow rate from the initial conditions to the actual desired setpoint to order to provide a smooth startup of a control loop

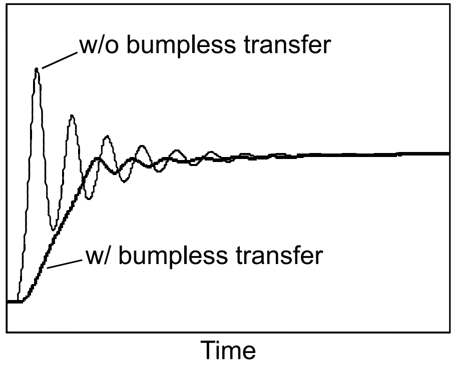
Bumpless Transfer (2)

- ❑ Comparison of true and internal setpoint



Bumpless Transfer (3)

- ❑ Control Performance with and without bumpless transfer



Derivative on Process Rather than Error (1)

The Facts:

- A step change in the set point results in a step change in the process
- The derivative term acts on the rate of change of the error
- The rate of change of a step change is very large
- An operator step change of the setpoint would causes a very large change in the output, upsetting the process

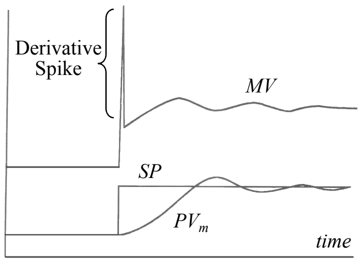
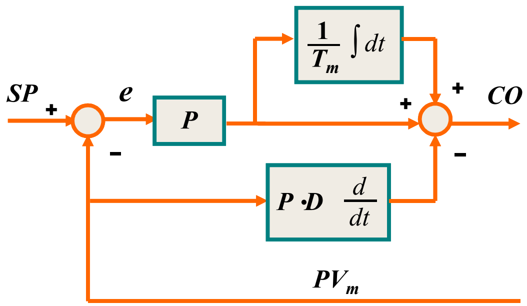


Fig. Process variable and valve response to a setpoint change using standard PID

Derivative on Process Rather than Error (2)

Solution: Let derivative act only on **process** rather than error



$$CO = P\left(e + \frac{1}{T_m}\int e\right) - P \cdot D\,\frac{dPV_m}{dt}$$

Derivative on Process Rather than Error (3)

- Process variable and valve response to a setpoint change

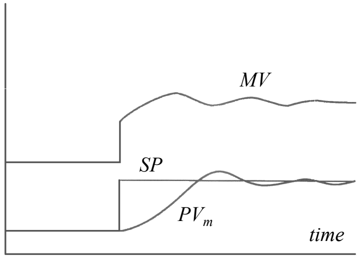


Fig. Process variable and valve response to a setpoint change using "**Derivative on Process Measurement**" PID

Derivative on Filtered Process Rather than Process (1)

The Fact:

- The derivative mode is hypersensitive to noise

Gain on Process Rather than Error (1)

The Facts:

- In applications with high gain, a step change can result in a sudden, large movement in the valve
- Not as severe as derivative effect, but still can upset the process

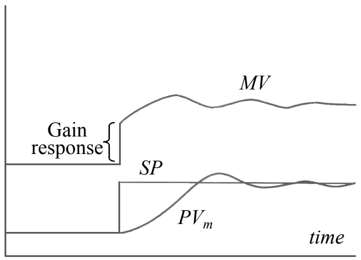
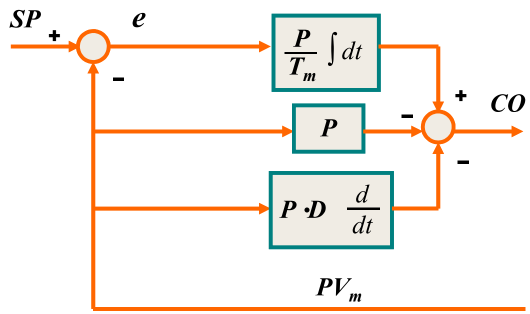


Fig. Process variable and valve response to a setpoint change using "**Gain on error and Derivative on Process Measurement**" PID

Gain on Process Rather than Error (2)

Solution: Let gain act only on process rather than error



$$CO = \frac{P}{T_m}\int e\,dt + P\left(PV_m + \frac{dPV_m}{dt}\right)$$

Gain on Process Rather than Error (3)

- Process variable and valve response to a setpoint change

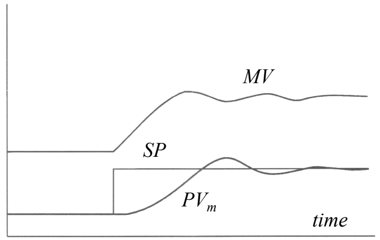
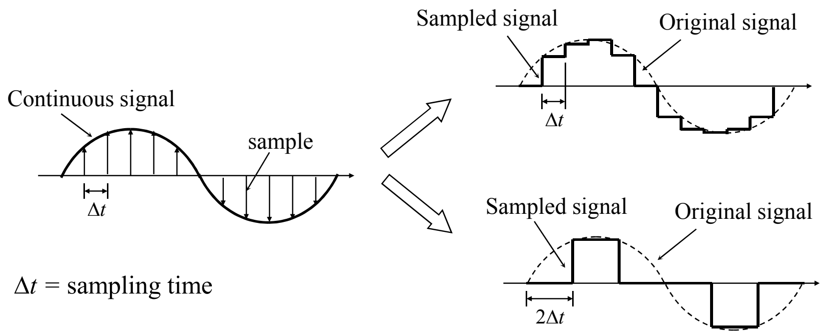


Fig. Process variable and valve response to a setpoint change using "**Gain and Derivative on Process Measurement**" PID

Digital PID Algorithms (1)

- Data acquisition concepts



Shannon's sampling theorem:

The sampling frequency must be greater or equal to two times of the highest frequency occurring in the signal to be sampled

Digital PID Algorithms (2)

- Integral

$$\int e\,dt \cong \Delta t \cdot \Sigma\, e_i$$

- Derivative

$$\frac{de}{dt} \cong \frac{e_i - e_{i-1}}{\Delta t}$$

Continuous form:

$$CO = P\left(e + \frac{1}{T_m}\int e\,dt + D\frac{de}{dt}\right)$$

Digital form:

$$CO = P\left[e_i + \frac{\Delta t}{T_m}\sum e_i + \frac{D}{\Delta t}(e_i - e_{i-1})\right]$$

i = sampling instant

Digital PID Algorithms (3)

- Two forms of digital PID algorithms:
  - Positional form

$$CO_i = P\left[e_i + \frac{\Delta t}{T_m}\sum e_i\,dt + \frac{D}{\Delta t}(e_i - e_{i-1})\right]$$

  - Velocity form
    - Inherently have anti reset windup feature

$$CO_i = CO_{i-1} + P\left[(e_i - e_{i-1}) + \frac{\Delta t}{T_m}e_i + \frac{D}{\Delta t}(e_i - 2e_{i-1} + e_{i-2})\right]$$

Several Offered PID Algorithms (1)

Distributed Control System (DCS)

- Honeywell TDC 3000
  - Offers 4 (four) PID equations; A, B, C, and D

TDC 3000	P mode	I mode	D mode
Algorithm A	Error	Error	Error
Algorithm B	Error	Error	Measurement
Algorithm C	Measurement	Error	Measurement
Algorithm D	Not used	Error	Not used

Several Offered PID Algorithms (2)

Distributed Control System (DCS)

- Foxboro I/A Series
- Yokogawa Centum CS 3000
- Bailey
- ABB

Several Offered PID Algorithms (3)

Programmable Logic Controller (PLC)

- Modicon 984

TDC 3000	P mode	I mode	D mode
P-only			
PI			
PID			

Several Offered PID Algorithms (3)

Programmable Logic Controller (PLC)

- Modicon
- Allan-Bradley
  - PLC-5
  - SLC500
- Siemens
- Fuji Electric

Guidelines for Common Control Loops (1)

Flow and liquid pressure control

- Fast response with no time delay (no pipe/transportation)
- Usually with small high-frequency noise
- PI controller with intermediate controller gain

Liquid level control

- Noisy due to splashing and turbulence
- High gain, low integral action of PI controller for integrating process
- Conservative setting for averaging control when it is used for damping the fluctuation of the inlet stream

Guidelines for Common Control Loops (2)

Gas pressure control

- Usually fast and self regulating
- PI controller with small integral action (large reset time)

Temperature control

- Wide variety of the process nature
- Usually slow response with time delay
- Use PID controller to speed up the response

Guidelines for Common Control Loops (3)

Composition control

- Similar to temperature control usually with larger noise and more time delay
- Effectiveness of derivative action is limited
- Temperature and composition controls are the prime candidates for advance control strategies due to its importance and difficulty of control

Session Summary

- PID control, which is the most widely used control algorithm in process control application, comes in different forms and terms
- Each of the terms of the PID equation must be understood to obtain a right combination of the PID control elements for various process control application objectives