PID Control Lecture Notes

Selina Varouqa | BS18-02 | Control Theory Spring 2020 | Innopolis University

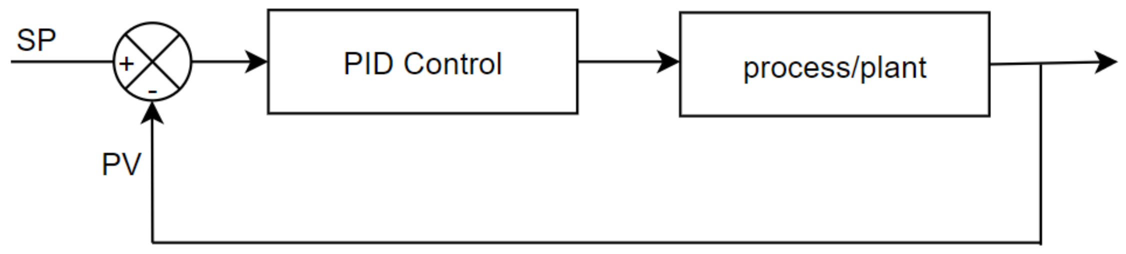
Overview

PID Controller (proportional-integral-derivative controller) is a control loop feedback mechanism that is widely used in industrial control systems.

The PID controller algorithm involves three *separate* parameters:

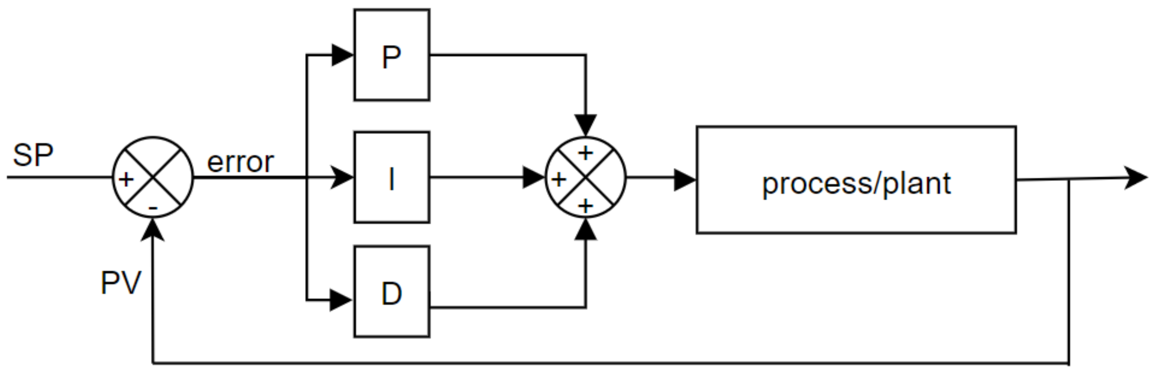
- **P (proportional)** depends on the present errors
- **I (integral)** depends on the accumulation of past errors
- **D (derivative)** depends on future errors

On a general view, it looks like the diagram below, where the **Setpoint (SP)** is the value that we **want** the process to be. So the PID controller compares the setpoint with the **Process Variable (PV)** which is the actual value that we have. For example, if our setpoint is the same as our process variable, then the controller does not have to do anything.



However, if there is a difference between the SP and the PV we have an error and control is needed. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions.

How does the PID work?



We can see from the diagram that it goes as follows:

1. The PV is subtracted from the SP to create the Error
2. The error is multiplied by one, two or all of the calculated P, I and D actions (depending on which controller we are using)
3. Then the results are added together and sent to the controller output

P Control

Output power is directly proportional to control error. Which means the higher the proportion coefficient, the less the output power at the same control error. If the proportional gain is too high, the system can become unstable.

Proportional Term

It is calculated by the following formula, Defining $u(t)$ as the controller output:

$u(t) = P_{out} = K_p \cdot e(t)$

where K_p is proportional gain, and $e(t)$ is controller error $e(t) = SP – PV(t)$

So as we can see that in P control, the controller simply multiplies the error by the proportional gain K_p to get the controller output.

PI Control

If we put proportional and integral actions together, we get the PI controller.

Integral Term

The integral term accelerates the movement of the process towards **setpoint** and eliminates the **steady-state error** (it is the difference between the desired final output and the actual one) that occurs with a pure proportional controller.

It is calculated as following, Defining I_{out} as the output of integral control:

$I_{out} = K_i \int_0^t e(\tau) d\tau$

where K_i is the integral gain, τ is the variable of integration, and $e(t)$ is controller error $e(t) = SP – PV(t)$

Combining P and I

When we combine both of the proportional and the integral term, defining $u(t)$

as the PI controller output it would be:

$u(t) = K_p \cdot e(t) + K_i \int_0^t e(\tau) d\tau$

PID Control

As it was stated above, derivative control depends on future errors. So when we add the derivative term, we can allow to have bigger P and I gains and still keep the loop stable, giving you a faster response and better loop performance.

Derivative Term

It is calculated as following, Defining D_{out} as the output of derivative control:

$$D_{out} = K_d \frac{de(t)}{dt}$$

where K_d is the derivative gain, and $e(t)$ is controller error $e(t) = SP – PV(t)$

PID Combined

When we combine the derivative term with both of the proportional and the integral term, defining $u(t)$ as the PI controller output it would be:

$$u(t) = K_p \cdot e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$$

> *In the lectures of Professor Savin, the notation was as follows:*
>
> - K_p *is the same as* K_p *above*
> - K_d *is the same as* K_d *above*
> - $u(t)$ *is the same as* $u(t)$
> - q *as* PV *for* p *term*
> - q^* *as* SP *for* p *term*
> - *therefore,* $e(t) = q^* - q$ *for* p *term*
> - $\dot{q}$ *as* PV *for* d *term*
> - $\dot{q}^*$ *as* SP *for* d *term*
> - *and therefore,* $\dot{e}(t) = \dot{q}^* - \dot{q}$ *for* p *term*

Transfer Function of PID controller

The transfer function in the **Laplace Domain** (using Laplace integral transformation that converts a function of a real variable t as *time* to a function of a complex variable s as *complex frequency*) of the PID controller is:

$$Tf = K_p + K_i/s + K_d s$$

General Effects of Each Control Parameter

The effects of each control parameter (K_p, K_i, K_d) on a closed-loop system are summarized in the table below:

CP	rise time	settle time	s-s error	overshoot
K_p	decrease	small change	decrease	increase
K_i	decrease	increase	decrease	increase

CP	rise time	settle time	s-s error	overshoot
K_d	minor change	decrease	no change	increase

rise time: the time required for a signal to change from a specified low value to a specified high value.

settle time: the time required for an output to reach and remain within a given error band following some input stimulus.

s-s error: steady state error (described above).

overshoot: the occurrence of a signal or function exceeding its target.

Tuning Methods

In (Ang, 2007), the PID controllers tuning methods are classified and grouped according to their nature and usage:

- **Analytical methods**: PID parameters are calculated through the use of analytical relations based on a process representation
- **Heuristic methods**: from practical experience (manual tuning) and are coded through the use of artificial intelligence techniques
- **Frequency response methods**: the frequency response characteristics of the controlled process is used to tune the PID controller.
- **Optimization methods** : utilizing a numerical optimization method for a single composite objective or use computerized heuristics or an evolutionary algorithm for multiple design objectives
- **Adaptive tuning methods**: these methods are based on automated online tuning, where the parameters are adjusted in real time

Applications of PID Control

- temperature control:
    - metal treatment by heat
    - baking
- pH neutralization control
- position control (robotic arm control)

Sources:

> 1. *https://www.intechopen.com/books/matlab-a-fundamental-tool-for-scientific-computing-and-engineering-applications-volume-1/pid-control-design#B2*
> 2. *https://www.ao-tera.com.ua/list/us/technology/0/246.html*
> 3. *https://controlguru.com/the-p-only-control-algorithm/*
> 4. *https://www.sciencedirect.com/topics/engineering/proportional-controller*
> 5. *http://www.cds.caltech.edu/~murray/books/AM08/pdf/am06-pid_16Sep06.pdf*
> 6. *http://engineering.ju.edu.jo/Laboratories/07-PID%20Controller.pdf*

7. *Farid Golnaraghi, Benjamin C.Kuo; Automatic Control Systems; Ninth Edition*
8. *https://ieeexplore.ieee.org/abstract/document/1453566*