ISSN 2229-5518

Effects of a PID Controller in Closed Loop Feedback System

Udeh Tochukwu Livinus, Prof. Natalya Zagorodna, Dibi Tamunomiebaka, Dr. Friz, Udeh Emeka Longinus

Abstract - The PID controller is a simple system. Well-developed architectures exist for building complex systems from the bottom up by combining PID controllers with linear and nonlinear elements such as cascade, mid-range, selector control, and gain scheduling.

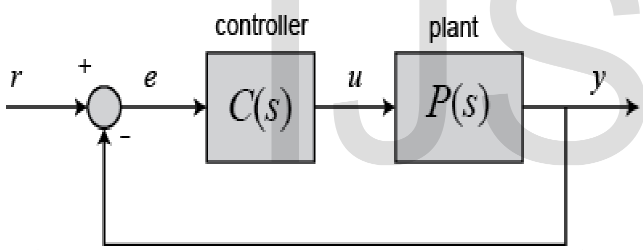
Key words – Proportional, Derivative, Integral, PID, Controller, Closed loop, System, Actuator, Tuning, Automatic, Automated systems, Plant

◆

Introduction

THe Proportional-Integral-Derivative (PID) Algorithm is the most common control algorithm used in industry. In PID control, the algorithm computes the desired Actuator Output by calculating proportional, integral, and derivative responses and summing those three components to compute the Output. Therefore, understanding the effect of each PID component is very important in tuning PID controllers. We will consider the following unity feedback system:

The



output of a PID controller, equal to the control input to the plant, in the time-domain is as follows:

(1)

$$u(t) = k_p e(t) + k_i \int e(t)dt + k_p \frac{de}{dt}$$

First, let's take a look at how the PID controller works in a closed-loop system using the schematic shown above. The variable (e) represents the tracking error, the difference between the desired input value (u) and the actual output (y). This error signal (e) will be sent to the PID controller, and the controller computes both the derivative and the integral of this error signal. The control signal (u) to the plant is equal to the proportional gain (k_p) times the magnitude of the error plus the integral gain (k_i) times the integral of the error plus the derivative gain (k_d) times the derivative of the error.

This control signal (u) is sent to the plant, and the new output (y) is obtained. The new output (y) is then fed back and compared to the reference to find the new error signal (e). The controller takes this new error signal and computes its derivative and it's integral again, ad infinitum.

The transfer function of a PID controller is found by taking the Laplace transform of Eq. (1).

(2).

$$k_p + \frac{k_i}{8} + k_d 8 = \frac{k_d 8^2 + k_p 8 + k_i}{8}$$

k_p = Proportional gain k_i= Integral gain k_d = Derivative gain

The Characteristics of P, I, and D Controllers

A proportional controller (k_p) will have the effect of reducing the rise time and will reduce but never eliminate the **steady-state error**. An integral control (k_i) will have the effect of eliminating the steady-state error for a constant or step input, but it may make the transient response slower. A derivative control (k_d) will have the effect of increasing the stability of the system, reducing the overshoot, and improving the transient response.

The effects of each of controller parameters, (k_p), (k_d), and (k_i) on a closed-loop system are summarized in the table below.

CL RESPONSE	RISE TIME	OVERSHOOT	SETTLING TIME	S-S ERROR
K_P	Decrease	Increase	Small Change	Decrease
K_i	Decrease	Increase	Increase	Eliminate
K_d	Small Change	Decrease	Decrease	No Change

Note that these correlations may not be exactly accurate, because, k_p, k_i, and k_d are dependent on each other. In fact, changing one of these variables can change the effect of the other two. For this reason, the table should

ISSN 2229-5518

only be used as a reference when you are determining the values for k_i, , k_p and k_d.

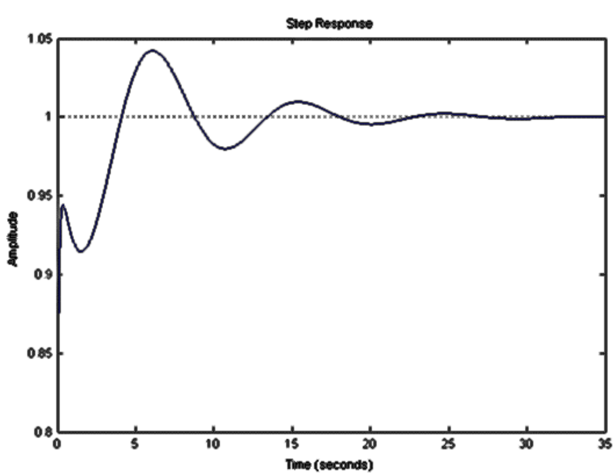
General Tips for Designing a PID Controller

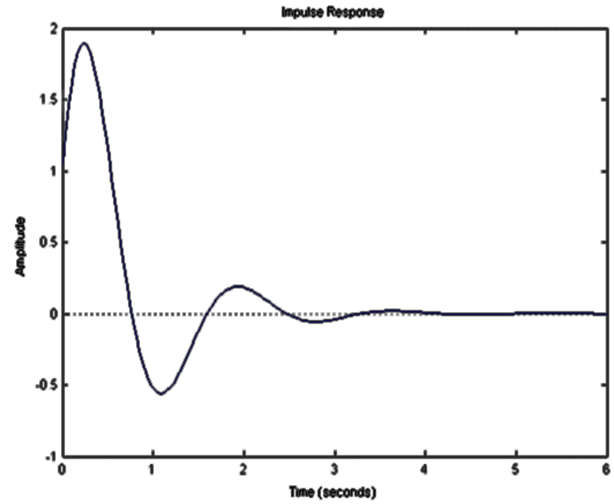
When you are designing a PID controller for a given system, follow the steps shown below to obtain a desired response.

1. Obtain an open-loop response and determine what needs to be improved
2. Add a proportional control to improve the rise time
3. Add a derivative control to improve the overshoot
4. Add an integral control to eliminate the steady-state error
5. Adjust each of Kp, Ki, and Kd until you obtain a desired overall response. You can always refer to the table shown in this "PID Tutorial" page to find out which controller controls what characteristics.

Lastly, please keep in mind that you do not need to implement all three controllers (proportional, derivative, and integral) into a single system, if not necessary. For example, if a PI controller gives a good enough response (like the above example), then you don't need to implement a derivative controller on the system. Keep the controller as simple as possible.

1. Effects of PID Controller in a closed loop feedback system





Setting time = 20.9s Rise Time = 3.62s Overshoot = 4.21%

Proportional Gain Effect on Closed Loop Feedback System

The proportional component depends only on the Error, which is the difference between the Set Point and the Process Variable. The Proportional gain (*Kp*) determines the ratio of Output response to the Error. For instance, if the Error signal has a magnitude of 10, a Proportional Gain of 5 would produce a proportional response of 50. In general, increasing the Proportional Gain will increase the speed of the control system response and also decrease the steady-state error which is the final difference between Process variable and Set Point. However, if the Proportional Gain is too large the Process Variable will begin to oscillate. If *Kp* is increased further, the oscillations will become larger and the system will become unstable.

2. Effect of Integral Gain in a closed loop feedback system

Using kp = 1,

Ki = 0.2, 0.5, 1
K_d = 0

Then the effect of the increase in the value of *Kp* can be seen in the figure below.

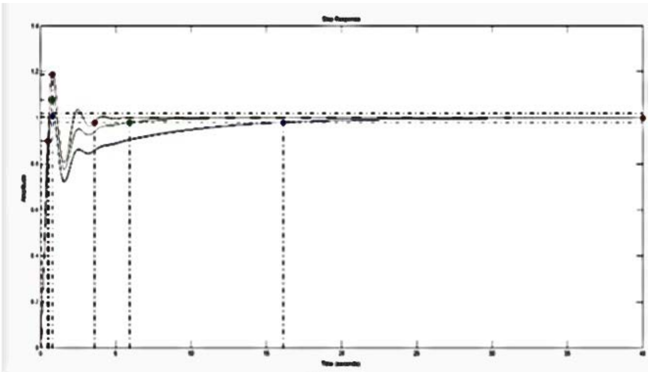


Fig 1.1 – Step Function

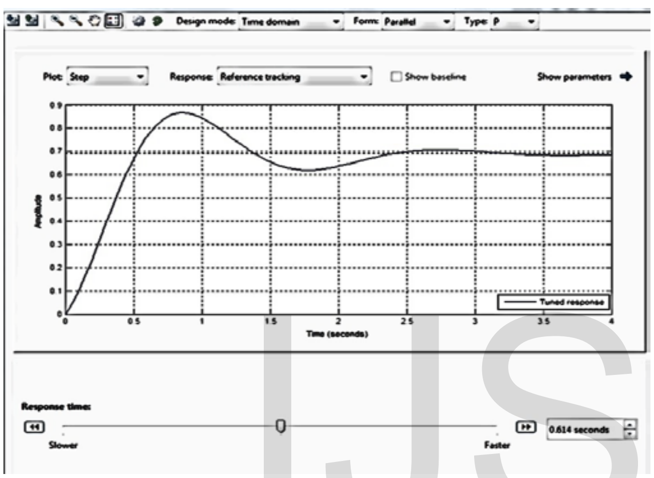


Fig 1.2 – Impulse Response

Table 1.0

System	Settling Time	Rise Time	Over Shoot
S_1= PIDF(1,0.2,0)	16.1	0.48	0.57
S_2= PIDF(1,0.5,0)	5.9	0.436	7.76
S_1= PIDF(1,1,0)	3.6	0.393	18.9

The Integral components integrates the Error over time to overcome the steady-state error. Therefore, the integral response will continually increase over time unless the Error is zero. However, the integral action may cause overshoot, oscillation, and/or instability problems if the selected integral gain (Ki) is too small. Note that smaller values of Ki will have a stronger integral effect on the system response.

3. Effect of Derivative Gain, in a closed loop feedback system

Kp = 3;
ki = 0.5;
Kd = 0.1, 4.5, 10;

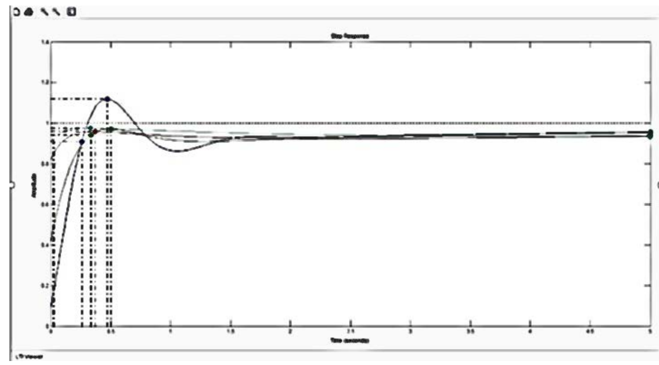


Fig 2.0

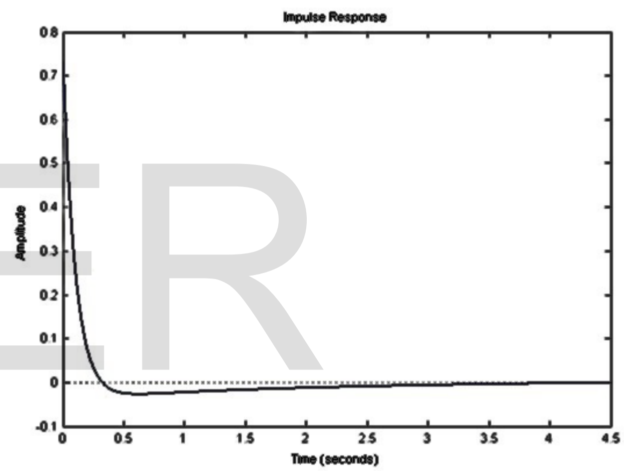


Fig 2.1 – Impulse Response

The table above shows the effect of kd on a system

Table 1.2

System	Settling Time	Rise Time	Over Shoot
S_1= PIDF(3,0.5,1)	14.6	0.414	0
S_2= PIDF(3,0.5,4.5)	19.1	11.4	0
S_3= PIDF(3,0.5,10)	28.2	15.9	0.213

The derivative part of the PID algorithm anticipates future behavior of the Error because the response of the derivative component is proportional to the rate of change of the Error. Therefore, in general the derivative action prevents overshoot and eliminates oscillations. On the other hand, most practical control systems use very small derivative gain (K_d) because the derivative response is highly sensitive to noise in the Process Variable signal. If the sensor feedback signal which represents the Process Variable is noisy, the derivative component can make the control system unstable.

Automatic PID Tuning

Traditionally, PID controllers were tuned manually using simple rules that date back to Ziegler and Nichols in the 1940s. The rules were based on process experiments. The step response method is based on measurement of the open-loop step response. The frequency response method is based on a closed loop experiment where the system is brought to the stability boundary under proportional control. Unfortunately, the traditional rules gave systems with poor performance. Automatic tuning has increased the use of derivative action. It has even been said: "This controller must have automatic tuning because it uses derivative action." Automatic tuning can be done in many ways. In rule-based methods that mimic an experienced instrument engineer, features of the closed-loop response are calculated and controller parameters are adjusted based on empirical rules. Other methods are based on estimation of low-order process models, typically first-order dynamics with time delays. The controller parameters are then determined by a variety of control design methods. Relay auto- tuning is another widely used approach that has proven to be robust and that brings attractive theoretical properties as well.

REFERENCES

1. Bennett, Stuart (1996). "A brief history of automatic control". *IEEE Control Systems Magazine* (IEEE) 16 (3): 17–25.
2. Jump up Bennett, Stuart (June 1986). *A history of control engineering, 1800-1930*. IET. pp. 142–148. ISBN 978-0-86341-047-5.
3. Jump up "Introduction: PID Controller Design". University of Michigan.
4. Wolfe, W.A (1951): "Controller settings for optimum control". Transactions of ASME, 64, pp. 413 – 418
5. Wills. D.M (1962b): "Tuning maps for three-mode controllers" Control Engineering, April, pp. 104 – 108.