Experiment 11

Cruise Control by PID

Part 2 PID Tuning

Tuning a control loop is the adjustment of its control parameters (proportional band/gain, integral gain/reset, derivative gain/rate) to the optimum values for the desired control response. Stability (no unbounded oscillation) is a basic requirement, but beyond that, different systems have different behavior, different applications have different requirements, and requirements may conflict with one another.

PID tuning is a difficult problem, even though there are only three parameters and in principle is simple to describe, because it must satisfy complex criteria within the limitations of PID control. There are accordingly various methods for loop tuning, and more sophisticated techniques are the subject of patents; for this experiment we describe two traditional manual methods for PID tuning.

Designing and tuning a PID controller appears to be conceptually intuitive, but can be hard in practice, if multiple (and often conflicting) objectives such as short transient and high stability are to be achieved. PID controllers often provide acceptable control using default tunings, but performance can generally be improved by careful tuning, and performance may be unacceptable with poor tuning. Usually, initial designs need to be adjusted repeatedly through computer simulations until the closed-loop system performs or compromises as desired.

Some processes have a degree of nonlinearity and so parameters that work well at full-load conditions don't work when the process is starting up from no-load; this can be corrected by gain scheduling (using different parameters in different operating regions). In order to understand the manual tuning we should know how to characterize the system from the step response.

Step-Transient Response

The transient response, shown in figure 1, of a practical control system often exhibits damped oscillations before reaching steady state.

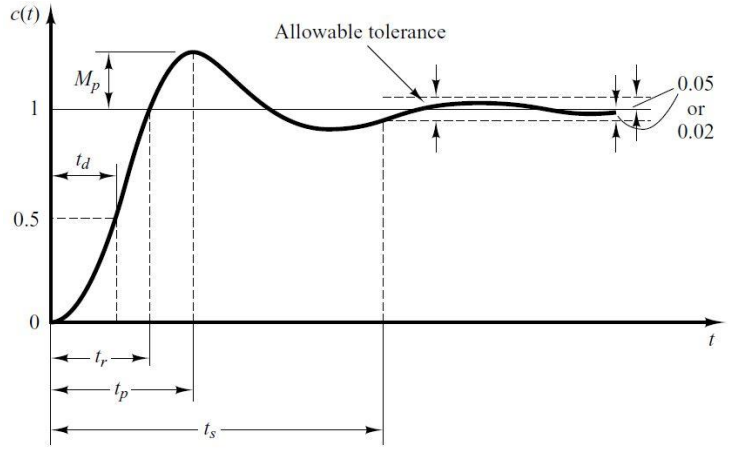

Fig 1 Transient and Steady-State Response Analyses

In specifying the transient-response characteristics of a control system to a unit-step input, it is common to specify the following:

- Delay time, t_d
- Rise time, t_r
- Peak time, t_p
- Maximum overshoot, M_P
- Settling time, t_s

These specifications are defined in what follows and are shown graphically in figure 1.

1. Delay time, t_d: The delay time is the time required for the response to reach half the final value the very first time.

2. Rise time, t_r : The rise time is the time required for the response to rise from 10% to 90%, 5% to 95%, or 0% to 100% of its final value. For underdamped second order systems, the 0%to 100% rise time is normally used. For overdamped systems, the 10% to 90% rise time is commonly used.

3. Peak time, t_p:The peak time is the time required for the response to reach the first peak of the overshoot.

4. Maximum (percent) overshoot, M_P: The maximum overshoot is the maximum peak value of the response curve measured from unity. If the final steady-state value of the response differs from unity, then it is common to use the maximum percent overshoot. It is defined:

$$M_P = \frac{c(t_p) - c(\infty)}{c(\infty)} \times 100\%$$

The amount of the maximum (percent) overshoot directly indicates the relative stability of the system.

5. Settling time, t_s : The settling time is the time required for the response curve to reach and stay within a range about the final value of size specified by absolute percentage of the final value (usually 2% or 5%). The settling time is related to the largest time constant of the control system. Which percentage error criterion to use may be determined from the objectives of the system design in question.

The time-domain specifications just given are quite important, since most control systems are time-domain systems; that is, they must exhibit acceptable time responses. (This means that, the control system must be modified until the transient response is satisfactory.)

First Method (trial and error)

If the system must remain online, one tuning method is to first set K_i and K_d values to zero. Increase the K_P until the output of the loop oscillates, then the K_P should be set to approximately half of that value for a "quarter amplitude decay" type response. Then increase K_i until any offset is corrected in sufficient time for the process. However, too much K_i will cause instability. Finally, increase K_d, if required, until the loop is acceptably quick to reach its reference after a load disturbance. However, too much K_d will cause excessive response and overshoot. A fast PID loop

tuning usually overshoots slightly to reach the setpoint more quickly; however, some systems cannot accept overshoot, in which case an overdamped closed-loop system is required, which will require a K_p setting significantly less than half that of the K_p setting that was causing oscillation. Table 1 presents the effects of *increasing* a parameter independently.

Table 1 Effects of *increasing* a parameter independently

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
K_p	Decrease	Increase	Small change	Decrease	Degrade
K_i	Decrease	Increase	Increase	Eliminate	Degrade
K_d	Minor change	Decrease	Decrease	No effect in theory	Improve if K_d small

Second Method (Ziegler–Nichols)

Another heuristic tuning method is formally known as the Ziegler–Nichols method. As in the method above, the K_i and K_d gains are first set to zero. The proportional gain is increased until it reaches the ultimate gain, K_u, at which the output of the loop starts to oscillate. K_u and the oscillation period T_u are used to set the gains as shown in table 2:

Table 2 Ziegler–Nichols method

Control Type	K_p	K_i	K_d
P	$0.50K_u$	—	—
PI	$0.45K_u$	$0.54K_u/T_u$	—
PID	$0.60K_u$	$1.2K_u/T_u$	$3K_uT_u/40$

<u>Procedure</u>

1) Repeat the steps 1 to 5 in part I of the experiment.
2) Apply both methods which were presented in this part.
3) From the serial plotter of IDE monitor the step response of both methods.
4) Record and compare your data of both methods in term of accuracy, stability and speed of response.