PID Control

F. Greg Shinskey
Process Control Consultant

97.1 Introduction

Process control plays an essential role in the safe manufacture of quality products at market demand, while protecting the environment. Flow rates, pressures and temperatures within pipes and vessels, inventories of liquids and solids, and product quality are all examples of measured variables that must be controlled to meet the above objectives. While there are several means available for controlling these variables, the PID family of controllers has historically carried the major share of this responsibility and, because of their simplicity and reliability, will continue to do so in the future.

The acronym PID stands for the three principal modes of the controller, each of which bears a mathematical relationship between the controlled variable *c* and the **manipulated variable** *m* driven by the controller output. The *P*roportional mode relates changes in *m* to changes in *c* through a proportional gain. The *I*ntegral mode moves the output at a rate related to the deviation of *c* from its desired value, known as the set point or reference *r*. Finally, the *D*erivative mode moves the output in response to the time derivative or rate of change of *c*. Interestingly, the mathematical relationships were actually determined *after* controllers had already been created to solve process-control problems. The integral mode was initially called *automatic reset*, and the derivative mode *hyper-reset* or *pre-act* [1].

97.2 Open and Closed Loops

Figure 97.1 describes the process and controller in functional blocks connected in a loop. Inputs to the process are manipulated *m* and load *q* variables, usually rates of flow of material or energy into or out of the process. The **load** may be a single variable or an aggregate, either independent or manipulated or controlled by another controller. If independent, it is often unmeasured, with its value inferred by the level of controller output required to maintain the controlled variable at set point. **Noise** *u* is shown affecting the controlled variable directly, typically caused by local turbulence in flow, pressure, and liquid-level measurements, or by nonuniformity of streams whose composition is measured.

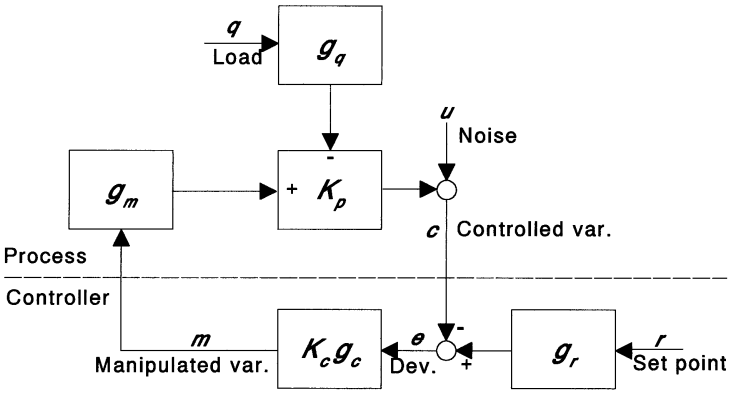

FIGURE 97.1 Process and controller connected in a loop.

Open-Loop Responses

In the absence of automatic control, the controlled variable is subject to variations in the load, and to manual adjustments to m intermittently introduced by operators. These cause variations in c following both the steady-state and dynamic functions appearing in Fig. 97.1:

$$c = K_p\left(mg_m - qg_q\right) + u \tag{97.1}$$

where K_p is the steady-state gain, g_m and g_q are the dynamic-gain vectors in the manipulated and load paths, respectively, and u is the noise level. The vectors have both magnitude and phase angle which are functions of the frequency or period of the signal passing through. In the **open loop,** variations in m are likely to be steps introduced by operators, but variations in q could take any form — steps, ramps, random variations, or cycles — depending on the source of the disturbance. Steps are easily introduced manually, and contain all the frequencies from zero to infinity; consequently, they are useful for evaluating loop response and testing the effectiveness of controllers. Noise is typified by random variations in a frequency range above the bandwidth of the control loop.

The dynamic elements typically consist of deadtime and lags. However, liquid level is the *integral* of the difference between inflow and outflow [2], in which case K_p in Fig. 97.1 is replaced by an integrator. Any difference between inflow and outflow will then be integrated, causing liquid level to continue rising or falling until the vessel limit is reached. This process has no self-regulation, and therefore cannot be left indefinitely in an open loop.

Closed-Loop Responses

In the **closed-loop,** c responds to load and set point as follows:

$$c = \frac{rg_q}{1 + 1/K_p g_m K_c g_c} - \frac{qg_q}{g_m K_c g_c + 1/K_p} + u \tag{97.2}$$

where K_c is the proportional gain of the controller, g_c is the **dynamic gain** of its integral and derivative modes, and g_r that of its set-point filter. For the typical **self-regulating** process in the open loop, c will take an exponential path following a step change in load as shown by the dashed curve in Fig. 97.2. If the loop is closed, the controller can move m to return c to r along the solid curve in Fig. 97.2, an optimum trajectory having a minimum integrated absolute error (IAE) between c and r. The leading

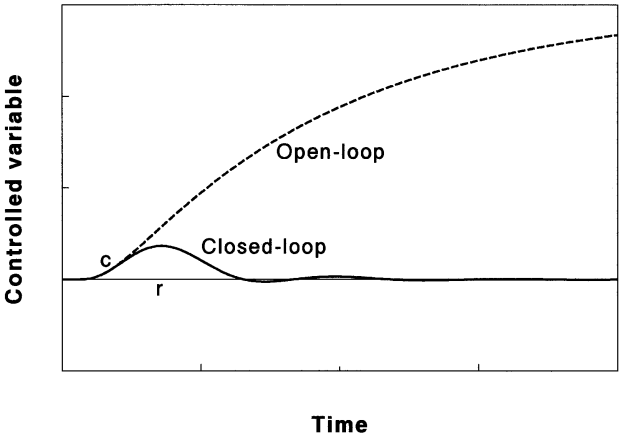

FIGURE 97.2 Closed-loop control minimizes error.

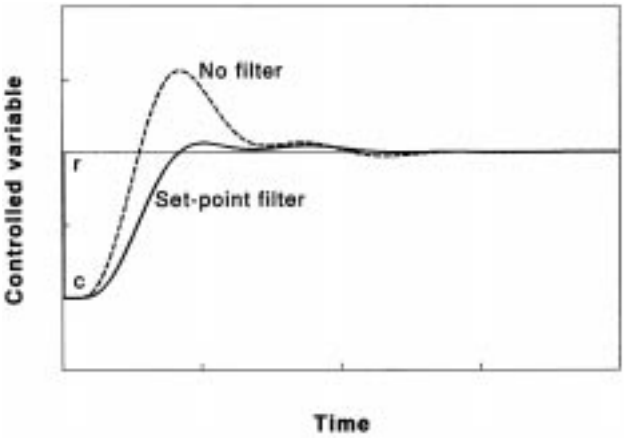

FIGURE 97.3 A lead–lag filter reduces overshoot.

edge of the curve depends on g_q and the trailing edge depends on g_m and the PID settings. If the PID settings are optimized for load response, they will usually cause c to overshoot a change in set point r, as shown by the dashed curve in Fig. 97.3. If the PID settings are then readjusted to reduce set-point overshoot, load response is extended. A preferred solution is to insert a lead-lag filter in the set-point path, which produced the solid curve in Fig. 97.3.

97.3 Mode Selection

The family of PID controllers includes P, I, PI, PD, and two PID controllers. Each has its own advantages and limitations, and therefore its preferred range of applications. Each is outlined along with an application.

Proportional Control

In a **proportional** controller, the deviation between c and r must change for the output to change:

$$m = b \pm K_c(r - c) \tag{97.3}$$

where b is an adjustable bias, also known as "manual reset," and the sign of the deviation is selected to provide negative feedback. Some controllers have proportional gain adjustable as percent **proportional band** P, where $K_c = 100/P$. If m must move to a different steady-state value because of a change in load or set point, the deviation will also change. Therefore the proportional controller allows a steady-state **offset** between c and r whenever m does not equal b which can only be eliminated by manually resetting b. Proportional control is recommended for liquid level, where K_c can be set quite high without loss of stability and there is no economic penalty for offset.

Proportional control of liquid level is actually preferred when manipulating the flow leaving a tank as the feed to a critical downstream process. Setting K_c slightly above 1.0 will keep the level in the tank for all rates of inflow, while minimizing the rate of change of outflow. This application is called **averaging level control** [2].

Integral Control

The **integral** mode eliminates offset by driving the output at a rate proportional to the deviation:

$$\frac{dm}{dt} = C \pm \frac{r - c}{I} \tag{97.4}$$

where t is time, d is the differential operator, and I is the integral time setting; C is the constant of integration, i.e., the initial value of the controller output when placed on automatic. Some controllers have integral time adjusted as integral gain or "reset rate" $1/I$ in inverse time as "repeats per minute." Integration produces a phase lag of 90° between input and output, which increases the response time and period of oscillation of the loop. The integral controller *cannot* be used for liquid level, because two integrators in series form an unstable closed loop [2]. Its use is limited to optimizing the set points of other closed loops which are already stable [2].

PI Control

The PI controller combines the proportional and integral modes:

$$m = C \pm K_c \left[r - c + \frac{1}{I} \int (r - c) dt \right] \tag{97.5}$$

The deviation and its integral are added vectorially, producing a phase lag falling between 0 and 90°. This combination provides stability with elimination of offset, making it the most common controller used in the fluid-processing industries. It is used almost universally, even in those applications where other controllers are better suited.

PD Control

The addition of **derivative action** to a proportional controller adds response for the rate of change of the controlled variable:

$$m = b \pm K_c \left(r - c - D \frac{dc}{dt} \right) \tag{97.6}$$

where D is the derivative time setting. Derivative action is preferably applied only to c as indicated and not to the set point, which would only increase set-point overshoot. A pure derivative function has a dynamic gain increasing indefinitely with frequency — to avoid instability within the controller, it is usually limited to about 10 by filtering. This provides an optimum combination of responsiveness and

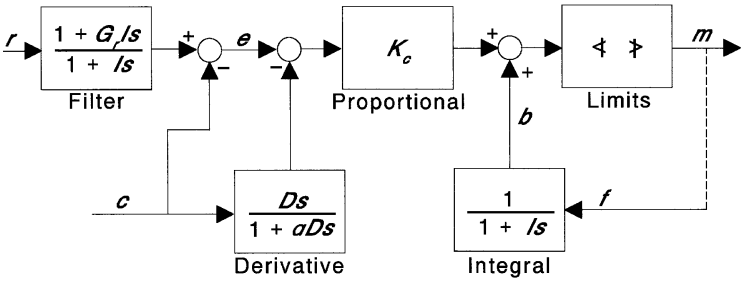

FIGURE 97.4 Commercial interacting PID controller.

noise rejection. Still, the high-frequency gain amplifies noise in flow and liquid-level measurements enough to prevent the use of the derivative mode on those loops; temperature measurements are usually noise-free, allowing it to be used to advantage there. The phase lead provided by the derivative mode reduces the period of oscillation and settling time of a loop, and improves stability.

PD controllers are recommended for batch processes, where operation begins with c away from r, and ends ideally with $c = r$ and flows at zero. The PD controller must be biased for this final output state, with D adjusted to eliminate overshoot, which for a zero-load process is permanent [2].

PID Controllers

The phase lead of derivative action more than offsets the phase lag of integral in a properly adjusted PID controller, resulting in a net phase lead and typically half the IAE of a PI controller applied to the same process. There are two types of PID controllers in common use, which differ in the way the modes are combined [3]. Early controllers combined PD and PI action in series, multiplying those functions rather than adding them. These interacting PID controllers remain in common use, even implemented digitally. Noninteracting controllers combine the integral and derivative modes in parallel, producing a more mathematically pure PID expression:

$$m = C \pm K_c\left[r - c - \frac{1}{I}\int (r-c)dt - D\frac{dc}{dt}\right] \tag{97.7}$$

The interacting controller can be described in the same form, but the coefficients of the individual terms are different [3]. The effective integral time of the interacting controller is $I + D$, its effective derivative time is $1/(1/I + 1/D)$, and its proportional gain is augmented by $1 + D/I$. Thus, the principal result of mode interaction is to require different PID settings for the two controllers applied to the same process.

A block diagram of a commercial interacting PID controller appears in Fig. 97.4, with transfer functions described in Laplace transforms whose operator s is equivalent to the differential operator d/dt. Integration is achieved by positive feedback of the output through a first-order lag set at the integral time. The feedback signal f is taken downstream of the high and low limits, and may be replaced with a constant or other variable to stop integration, in which case, the controller behaves as PD with remote bias. This feature is extremely valuable in preventing reset **windup,** which occurs whenever a controller with integral action remains in automatic while the loop is open, and results in a large overshoot when the loop is then closed [3]. Most noninteracting controllers lack this feature.

Derivative action combines a differentiator Ds with a lag (filter) of time constant αD, where α represents the inverse of the high-frequency gain limit of typically 10. Figure 97.4 also shows a lead-lag set-point filter having a lag time constant of integral time I and lead time of G_rI, which produces a gain of G_r to a step in set point. This gain is adjustable from 0 to 1, with 1 eliminating the filter and zero imposing a first-order lag. The effect of this lag is equivalent to eliminating proportional action from r in Eqs. 97.5 and 97.7, an optional feature in some PID controllers. The gain adjustment offers more flexibility in

optimizing set-point response as in Fig. 97.3; G_r is set around 0.3 for lag-dominant processes such as that described in Figs. 97.2 and 97.3, and closer to 1 for deadtime-dominant processes [3].

97.4 Controller Hardware

Pneumatic proportional controllers were first used early in this century, with integral and then derivative functions added around 1930. Pneumatic controllers are still in use in hazardous areas, in remote gas fields where they are operated by gas, and for simple tasks such as regulating temperatures in buildings. Electronic analog controllers began to replace pneumatic controllers in the late 1950s, but their functions were implemented using digital microprocessors beginning around 1980. Digital control began in mainframes in the 1960s, but now is available is programmable logic controllers (PLC), personal computers (PC), and in distributed control systems (DCS).

Pneumatic Controllers

The simplest pneumatic controllers are proportional units used to control heating, ventilating, and air-conditioning (HVAC) in buildings. However, complete PID controllers with functionality similar to that shown in Fig. 97.4 are also available, used for such demanding tasks as temperature control of batch exothermic reactors and regulation of flows and pressures on offshore oil platforms. There are panel-mounted units available as well as weatherproof models used for field installation. Most have auto–manual transfer stations with bumpless transfer between the two modes of operation. Their principal limitations are a speed of response limited by transmission lines (lengths beyond 30 m are unsuitable for flow and pressure control) and lack of computing capability (required for nonlinear characterization and automatic tuning).

Electronic Controllers

The first electronic controllers mimicked their pneumatic counterparts while eliminating transmission lags. Eventually remote auto–manual transfer and remote tuning were added, and microprocessors brought these controllers calculation and logic functions, signal conditioning and characterization, and auto- and self-tuning features as well. Multiple controllers are even available in a single station for implementing cascade and feedforward systems. The bandwidth of electronic *analog* controllers extends to 10 Hz, and in some units even further. *Digital* controllers execute their calculations intermittently rather than continuously; most are limited to 10 Hz operation, which with digital filtering results in an effective deadtime of 0.1 s, reducing bandwidth to about 1.3 Hz.

Electronic controllers are used extensively in field locations and dedicated to control of individual units such as pumps, dryers, wastewater-treatment facilities — wherever only a few loops are required. Peer-to-peer communication is available in some controllers for incorporation into networks, and most can communicate with personal computers where data can be displayed on a graphic interface. Configuration (selection of scales, control modes, alarms, and other functions) can be done either via keys on the controller faceplate or through a PC.

Digital Controllers

While the stand-alone digital controller evolved from electronic analog models, centralized digital controllers originated with digital computers. Mainframe computers were first used for direct digital control (DDC) where valves were manipulated by the computer, and for supervisory control, where the computer positioned set points of analog controllers. Gradually minicomputers and microprocessors replaced the mainframe, with functions becoming *distributed* among field input–output modules, workstations, control modules, etc., in clusters and other configurations as DCS. PCs are used for some workstations, and even for direct control in some plants.

Where many loops are controlled by a single processor, the interval between samples is likely to be 0.5 s or even longer. Users tend to keep expanding the functions demanded of the processor, thereby extending the interval between samples. This reduces the bandwidth of DCS controllers to values too low for combustion and compressor controls. Some digital PID algorithms produce an *incremental* output Δm, requiring an integrator downstream to produce m. These are not available in proportional or PD action because they have no fixed bias b, only a constant of integration C which is subject to change whenever the controller is operated manually.

The PLC was originally a replacement for relay logic. Eventually PID loops were added, which generally operate much faster (e.g., 100 Hz) than other digital controllers. However, many have nonstandard algorithms and lack the functionality of other PID controllers, such as proportional and PD control, windup protection, derivative filtering, set-point filtering, nonlinear characterization, and autotuning.

97.5 Tuning Controllers

A controller is only as effective as its **tuning:** the adjustment of the PID settings relative to the process parameters to optimize load and set-point response as described in Figs. 97.2 and 97.3. Tuning is required when the controller is first commissioned on a loop, and may have to be repeated if process parameters change appreciably with time, load, set point, etc. Tuning requires the introduction of a disturbance in either the open or closed loop, and interpretation of the resulting response. *Autotuning* essentially automates manual procedures, whereas *self-tuning* can recognize and correct an unsatisfactory response without testing.

Manual Tuning

Ziegler and Nichols [1] developed the first effective tuning methods, and these are still used today. Their open-loop method steps the controller output to produce a response like the broken curve in Fig. 97.2. The apparent deadtime (delay) in the response and the steepest slope of the curve are then converted into appropriate settings for PI and PID controllers using simple formulas. The open-loop method is most accurate for lag-dominant processes, but the closed-loop method is more broadly applicable. It is based on inducing a uniform cycle by adjusting the gain of a controller with only its proportional mode effective (D is set to 0 and I at maximum). The period of the cycle and the controller gain are then used to calculate appropriate PID settings.

The open-loop method has been extended to other processes [3], but remains limited to the accuracy of step-test results. Fine tuning must be done with the loop closed: with the deviation zero, place the controller in manual, step the output, and immediately transfer to automatic — this simulates a closed-loop load change. The resulting response should appear like the solid curve in Fig. 97.2, having a first peak which is symmetrical, followed by a slight overshoot and well-damped cycle. If the overshoot is excessive, I and/or D time should be increased; in the case of undershoot, they should be decreased. (To simplify tuning of PID controllers, I and D can be moved together, keeping the I/D ratio at about 4 for noninteracting controllers and 3 for interacting controllers.) If damping is light, lower the proportional gain; if recovery is slow (producing an unsymmetrical first peak), raise the proportional gain. The set point should *not* be stepped for test purposes unless set-point response is important, as for flow controllers.

Autotuning

Some autotuning controllers use a step test in the open loop, implementing Ziegler–Nichols rules. A single pulse produces more accurate identification, and a doublet pulse is better still [3]. A closed-loop method replaces proportional cycling with relay cycling, where the output switches between high and low limits as c crosses r [3]. The period of the resulting cycle is used to set I and D, and its amplitude relative to the distance between output limits to set K_c. The principal limitation of the step and relay

autotuning methods is that only two response features are used to identify a complex process. The estimated PID settings may not fit the specific process particularly well, resulting in instability in some loops and sluggishness in others. The autotuning function cannot monitor the effectiveness of its work, as self-tuning does.

Self-Tuning

A self-tuning controller need not test the process, but simply observe its closed-loop response to set-point and load changes with its current PID settings. It then performs fine-tuning as described above to bring the overshoot or damping or symmetry of the response curve closer to optimum. This may require several iterations if the PID settings are far from optimum, but can eventually converge on optimum response, and readjust as necessary whenever process parameters change. Without a test disturbance, mischaracterization is possible, especially for a sinusoidal disturbance, where detuning may result when tightening would give better control.

Defining Terms

Averaging level control: Allowing the liquid level in a tank to vary in an effort to minimize changes in its outflow.

Closed-loop response: The response of a controlled variable to changes in set point or load with the controller in the automatic mode.

Derivative action: The change in controller output responding to a rate of change in the controlled variable.

Dynamic gain: The ratio of the change in output from a function to a change in its input which varies with time or with the frequency of the input.

Integral action: The rate of change of controller output responding to a deviation between the controlled variable and set point.

Load: A variable or aggregate of variables which affects the controlled variable.

Manipulated variable: A variable changed by the controller to move the controlled variable.

Noise: A disturbance having a frequency range too high for the controller to affect.

Offset: Steady-state deviation between the controlled variable and set point.

Open-loop response: The response of a controlled variable to process inputs in the absence of control action.

Proportional action: The change in controller output responding directly to a change in the controlled variable.

Proportional band: The percentage change in the controlled variable required to drive the controller output full scale.

Self-regulation: The property of a process through which a change in the controlled variable affects either the flow into or out of a process in such a way as to reduce further changes in the controlled variable.

Tuning: Adjusting the settings of a controller to affect its performance.

Windup: Saturation of the integral model of a controller which results in the controlled variable subsequently overshooting the set point.

References

1. J. G. Ziegler and N. B. Nichols, Optimum settings for automatic controllers, *Trans. ASME*, November 1942, 759–768.
2. F. G. Shinskey, *Process Control Systems*, 4th ed., New York: McGraw-Hill, 1996, 22–25, 25–28, 192–194, 388–390.
3. F. G. Shinskey, *Feedback Controllers for the Process Industries*, New York: McGraw-Hill, 1994, 68–70, 176–178, 157–164, 148–151, 155–156.