The PID Control Algorithm

How it works, how to tune it, and how to use it

2nd Edition

John A. Shaw
Process Control Solutions
December 1, 2003

John A. Shaw is a process control engineer and president of Process Control Solutions. An engineering graduate of N. C. State University, he previously worked for Duke Power Company in Charlotte, N. C. and for Taylor Instrument Company (now part of ABB, Inc.) in, N. Y. Rochester He is the author of over 20 articles and papers and continues to live in Rochester.

Table of Contents

Table of Figures

CHAPTER 1 INTRODUCTION

Process control is the measurement of a process variable, the comparison of that variables with its respective set point, and the manipulation of the process in a way that will hold the variable at its set point when the set point changes or when a disturbance changes the process.

An example is shown in Figure 1. In this example, the temperature of the heated water leaving the heat exchanger is to be held at its set point by manipulating the flow of steam to the exchanger using the steam flow valve. In this example, the temperature is known as the *measured* or *controlled variable* and the steam flow (or the position of the steam valve) is the *manipulated variable*.

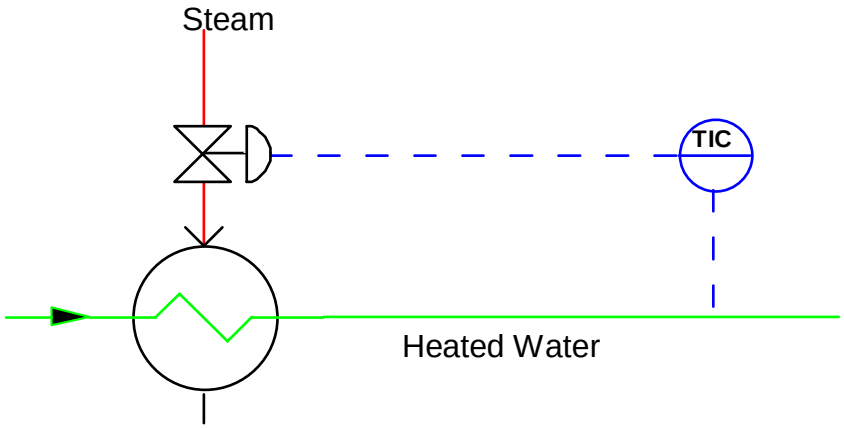


Figure 1 Typical process control loop – temperature of heated water.

Most processes contain many variables that need to be held at a set point and many variables that can be manipulated. Usually, each controlled variable may be affected by more than one manipulated variable and each manipulated variable may affect more than one controlled variable. However, in most process control systems manipulated variables and control variables are paired together so that one manipulated variable is used to control one controlled variable. Each pair of controlled variable and manipulated variable, together with the control algorithm, is referred to as a *control loop*. The decision of which variables to pair is beyond the scope of this publication. It is based on knowledge of the process and the operation of the process.

In some cases control loops may involve multiple inputs from the process and multiple outputs to the processes. The first part of this book will consider only single input, single output loops. Later we will discuss some multiple loop control methods.

There are a number of algorithms that can be used to control the process. The most common is the simplest: an on/off switch. For example, most appliances use a thermostat to turn the heat on when the temperature falls below the set point and then turn it off when the temperature reaches the set point. This results in a cycling of the temperature above and below the set point but is sufficient for most common home appliances and some industrial equipment.

To obtain better control there are a number of mathematical algorithms that compute a change in the output based on the controlled variable. Of these, by far the most common is known as the PID (Proportional, Integral, and Derivative) algorithm, on which this publication will focus.

First we will look at the PID algorithm and its components. We will then look at the dynamics of the process being controlled. Then we will review several methods of tuning (or adjusting the parameters of) the PID control algorithm. Finally, we will look as several ways multiple loops are connected together to perform a control function.

1.1 THE CONTROL LOOP

The process control loop contains the following elements:

- The measurement of a process variable. A sensor, more commonly known as a transmitter, measures some variable in the process such as temperature, liquid level, pressure, or flow rate, and converts that measurement to a signal (typically 4 to 20 ma.) for transmission to the controller or control system.

- The control algorithm. A mathematical algorithm inside the control system is executed at some time period (typically every second or faster) to calculate the output signal to be transmitted to the final control element.

- A final control element. A valve, air flow damper, motor speed controller, or other device receives a signal from the controller and manipulates the process, typically by changing the flow rate of some material.

- The process. The process responds to the change in the manipulated variable with a resulting change in the measured variable. The dynamics of the process response are a major factor in choosing the parameters used in the control algorithm and are covered in detail in this publication.

The interconnection of these elements is illustrated in Figure 2.

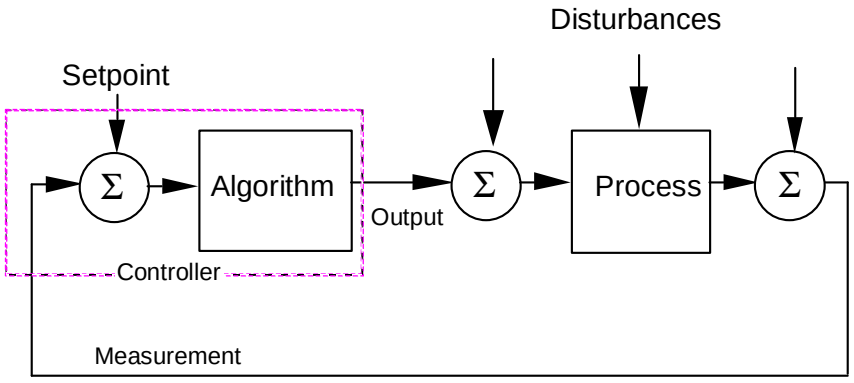


Figure 2 Interconnection of elements of a control loop.

The following signals are involved in the loop:

- The *process measurement*, or *controlled* variable. In the water heater example, the controlled variable for that loop is the temperature of the water leaving the heater.

- The *set point*, the value to which the process variable will be controlled.

- *One or more load variables*, not manipulated by this control loop, but perhaps manipulated by other control loops. In the steam water heater example, there are several load variables. The flow of water through the heater is one that is likely controlled by some other loop. The temperature of the cold water being heated is a load variable. If the process is outside, the ambient temperature and weather (rain, wind, sun, etc.) are load variables outside of our control. A change in a load variable is a *disturbance*.

Other measured variables may be displayed to the operator and may be of importance, but are not a part of the loop.

1.2 ROLE OF THE CONTROL ALGORITHM

The basic purpose of process control systems such as is two-fold: To manipulate the final control element in order to bring the process measurement to the set point whenever the set point is changed, and to hold the process measurement at the set point by manipulating the final control element. The control algorithm must be designed to quickly respond to changes in the set point (usually caused by operator action) and to changes in the loads (disturbances). The design of the control algorithm must also prevent the loop from becoming unstable, that is, from oscillating.

1.3 AUTO/MANUAL

Most control systems allow the operator to place individual loops into either manual or automatic mode.

In manual mode the operator adjusts the output to bring the measured variable to the desired value. In automatic mode the control loop manipulates the output to hold the process measurements at their set points.

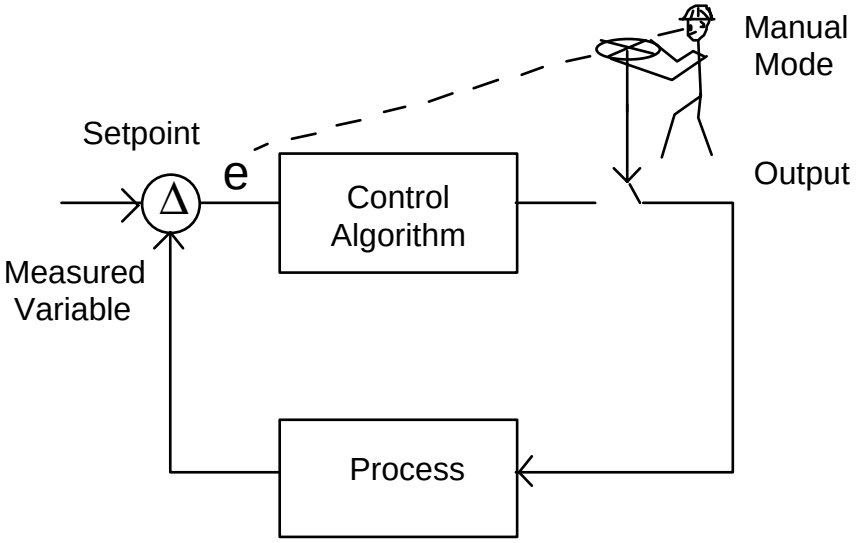


Figure 3 A control loop in manual.

In most plants the process is started up with all loops in manual. During the process startup loops are individually transferred to automatic. Sometimes during the operation of the process certain individual loops may be transferred to manual for periods of time.

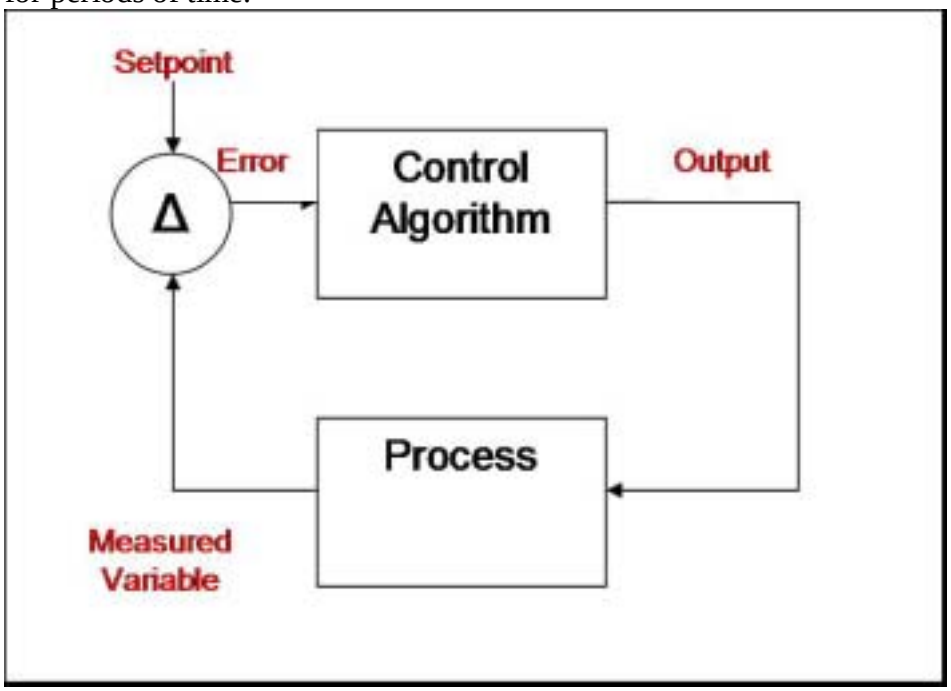


Figure 4 A control loop in automatic

CHAPTER 2 THE PID ALGORITHM

In industrial process control, the most common algorithm used (almost the only algorithm used) is the time-proven PID—**P**roportional, **I**ntegral, **D**erivative—algorithm. In this chapter we will look at how the PID algorithm works from both a mathematical and an implementation point of view.

2.1 KEY CONCEPTS

- **The PID control algorithm does not "know" the correct output that will bring the process to the set point.**

The PID algorithm merely continues to move the output in the direction that should move the process toward the set point until the process reaches the set point. The algorithm must have feedback (process measurement) to perform. If the loop is not closed, that is, the loop is in manual or the path between the output to the input is broken or limited, the algorithm has no way to "know" what the output should be. Under these (open loop) conditions, the output is meaningless.

- **The PID algorithm must be "tuned" for the particular process loop. Without such tuning, it will not be able to function.**

To be able to tune a PID loop, each of the terms of the PID equation must be understood. The tuning is based on the dynamics of the process response and is will be discussed in later chapters.

2.2 ACTION

The most important configuration parameter of the PID algorithm is the *action*. Action determines the relationship between the direction of a change in the input and the resulting change in the output. If a controller is *direct acting,* an increase in its input will result in an increase in its output. With *reverse action* an increase in its input will result in a decrease in its output.

The controller action is always the opposite of the process action.

2.3 THE PID RESPONSES

The PID control algorithm is made of three basic responses, Proportional (or gain), integral (or reset), and derivative. In the next several sections we will discuss the individual responses that make up the PID controller.

In this book we will use the term called "error" for the difference between the process and the set point. If the controller is direct acting, the set point is subtracted from the measurement; if reverse acting the measurement is subtracted from the set point. Error is always in percent.

Error = Measurement-Set point (Direct action)

Error = Set point-Measurement (Reverse action)

2.4 PROPORTIONAL

The most basic response is proportional, or gain, response. In its pure form, the output of the controller is the error times the gain added to a constant known as "manual reset".

Output = E x G + k

where:

Output = the signal to the process
E = error (difference between the measurement and the set point.
G = Gain
k = manual reset, the value of the output when the measurement equals the set point.

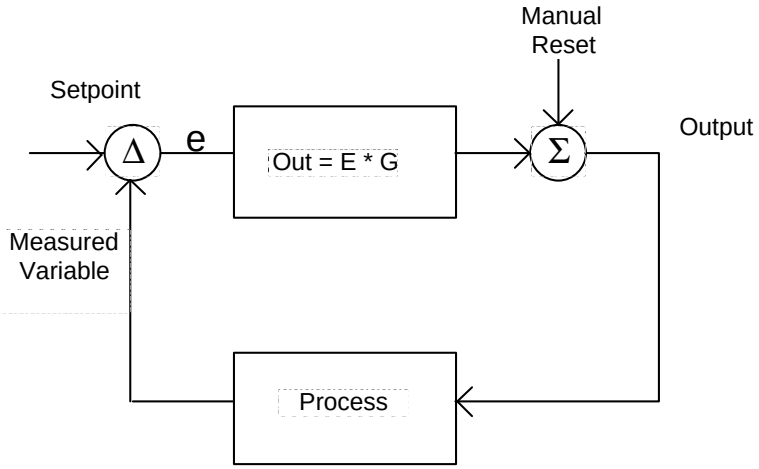


Figure 5 A control loop using a proportional only algorithm.

The output is equal to the error time the gain plus manual reset. A change in the process measurement, the set point, or the manual reset will cause a change in the output. If the process measurement, set point, and manual reset are held constant the output will be constant

Proportional control can be thought of as a lever with an adjustable fulcrum. The process measurement pushes on one end of the lever with the valve connected to the other end. The position of the fulcrum determines the gain. Moving the fulcrum to the left increases the gain because it increases the movement of the valve for a given change in the process measurement.

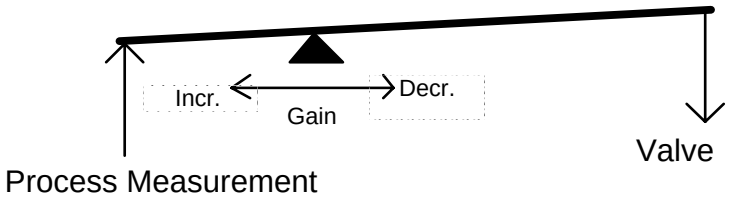


Figure 6 A lever used as a proportional only reverse acting controller.

2.5 PROPORTIONAL—OUTPUT VS. MEASUREMENT

One way to examine the response of a control algorithm is the open loop test. To perform this test we use an adjustable signal source as the process input and record the error (or process measurement) and the output.

As shown below, if the manual reset remains constant, there is a fixed relationship between the set point, the measurement, and the output.

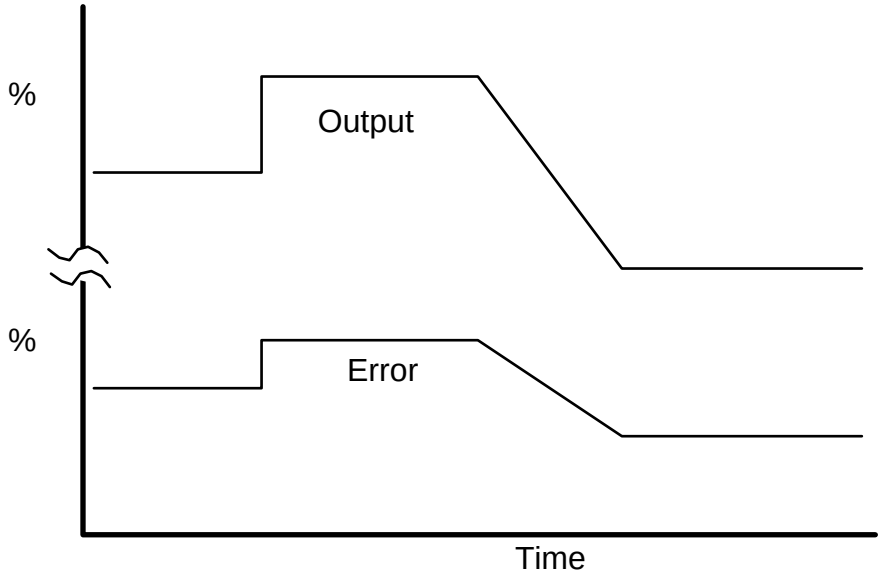


Figure 7 Proportional only controller: error vs. output over time.

2.6 PROPORTIONAL—OFFSET

Proportional only control produces an offset. Only the adjustment of the manual reset removes the offset.

Take, for example, the tank in Figure 8 with liquid flowing in and flowing out under control of the level controller. The flow in is independent and can be considered a load to the level control.

The flow out is driven by a pump and is proportional to the output of the controller.

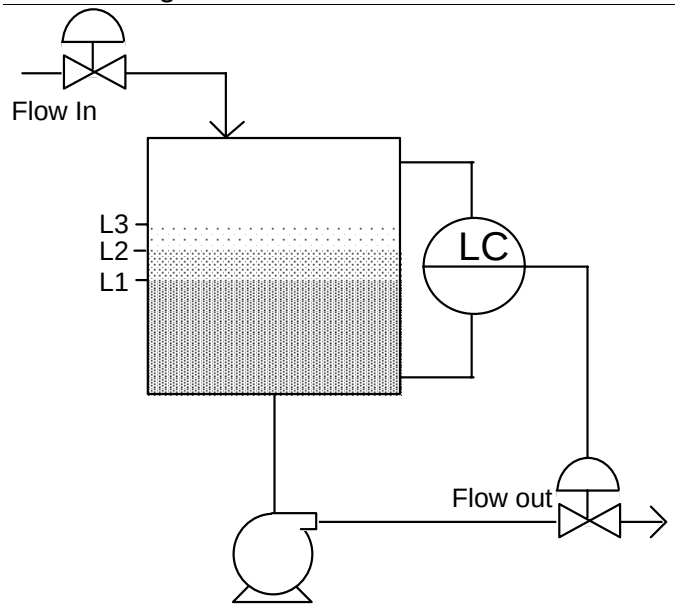


Figure 8 Proportional only level control

The flow from the tank is proportional to the level. Because the flow out eventually will be equal to the flow in, the level will be proportional to the flow in. An increase in flow in causes a higher steady state level. This is called "offset".

Assume first that the level is at its set point of 50%, the output is 50%, and both the flow in and the flow out are 500 gpm. Then let's assume the flow in increases to 600 gpm. The level will rise because more liquid is coming in than going out. As the level increases, the valve will open and more flow will leave. If the gain is 2, each one percent increase in level will open the valve 2% and will increase the flow out by 20 gpm. Therefore by the time the level reaches 55% (5% error) the output will be at 60% and the flow out will be 600 gpm, the same as the flow in. The level will then be constant. This 5% error is known as the offset.

Offset can be reduced by increasing gain. Let's repeat the above "experiment" but with a gain of 5. For each 1% increase in level will increase the output by 5% and the flow out by 50 gpm. The level will only have to increase to 52% to result in a flow out of 600 gpm, causing the level to be constant. Increasing the gain from 2 to 5 decreases the offset from 5% to 2%. However, only an infinite gain will totally eliminate offset.

Gain, however, cannot be made infinite. In most loops there is a limit to the amount of gain that can be used. If this limit is exceeded the loop will oscillate.

2.7 PROPORTIONAL—ELIMINATING OFFSET WITH MANUAL RESET

Offset can also be eliminated by adjusting manual reset. In the above example (with a gain of two) if the operator increased the manual reset the valve would open further, increasing the flow out. This would cause the level to drop. As the level dropped, the controller would bring the valve closed. This would stabilize the level but at a level lower than before. By gradually increasing the manual reset the operator would be able to bring the process to the set point.

2.8 ADDING AUTOMATIC RESET

With proportional only control, the operator "**resets**" the controller (to remove offset) by adjusting the **manual reset**:

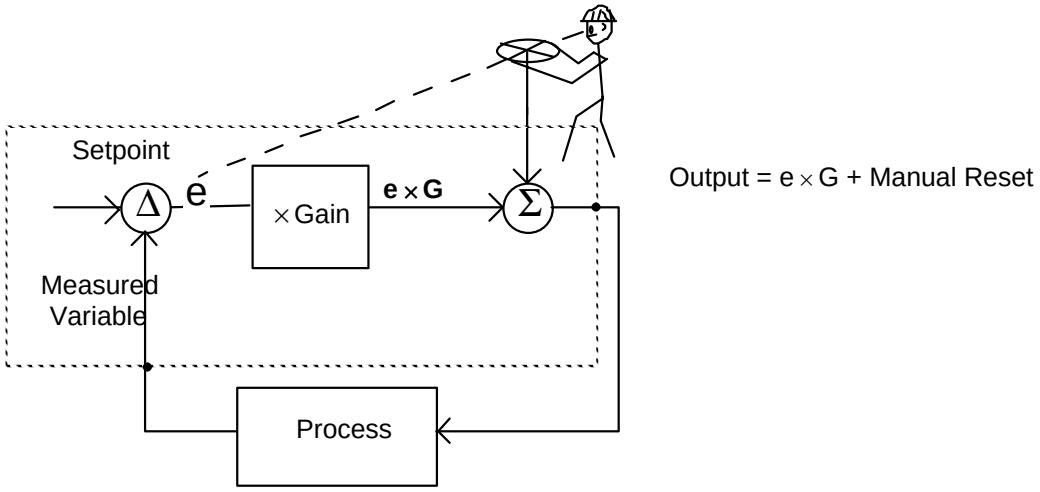


Figure 9 Operator adjusted manual reset

The operator may adjust the manual reset to bring the measurement to the set point, eliminating the offset.

If the process is to be held at the set point the manual reset must be changed every time there is a load change or a set point change. With a large number of loops the operator would be kept busy resetting each of the loops in response to changes in operating conditions.

The manual reset may be replaced by **automatic reset,** a function that will continue to move the output as long as there is any error:

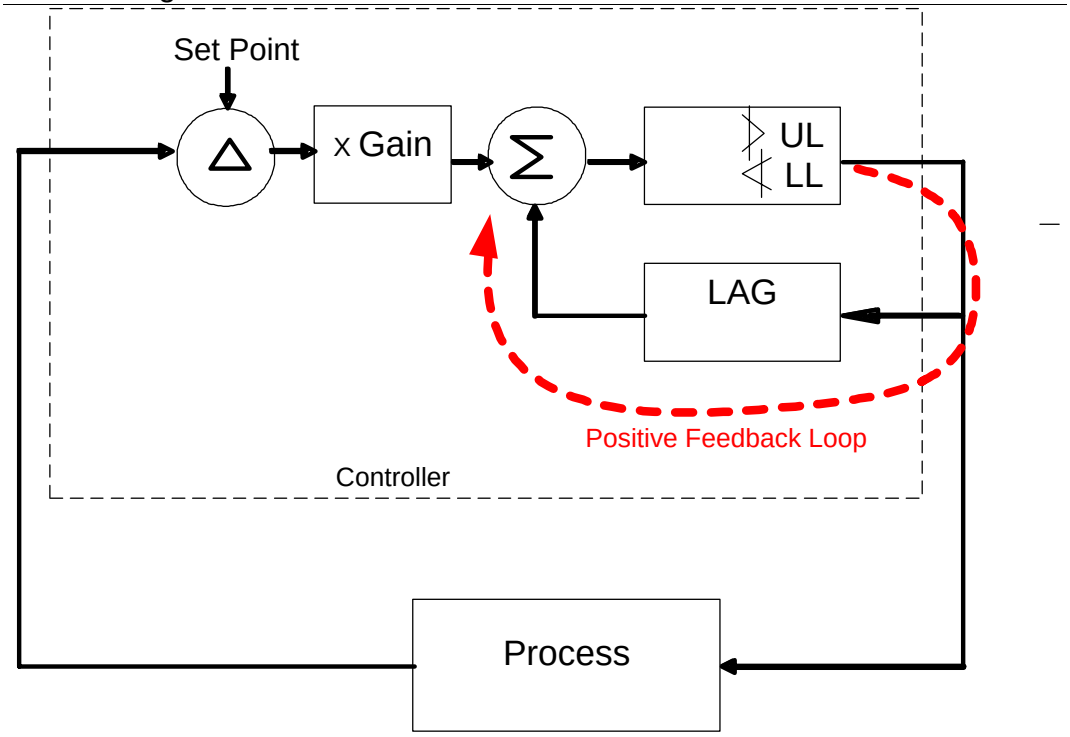


Figure 10 Addition of automatic reset to a proportional controller

The positive feedback loop will cause the output to ramp whenever the error is not zero. There is an output limit block to keep the output within specified range, typically 0 to 100%.

This is called "**Reset**" or **Integral Action**. Note the use of the positive feedback loop to perform integration. As long as the error is zero, the output will be held constant. However, if the error is non-zero the output will continue to change until it has reached a limit. The rate that the output ramps up or down is determined by the time constant of the lag and the amount of the error and gain.

2.9 INTEGRAL MODE (RESET)

If we look only at the reset (or integral) contribution from a more mathematical point of view, the reset contribution is:

$$\mathbf{Out} = \mathbf{g} \times \mathbf{K_r} \times \int \mathbf{e}\, \mathbf{dt}$$

where $\mathbf{g}$ = gain

$\mathbf{K_r}$ = reset setting in repeats per minute.

At any time the rate of change of the output is the gain time the reset rate times the error. If the error is zero the output does not change; if the error is positive the output increases.

Shown below is an open loop trend of the error and output. We would obtain this trend if we recorded the output of a controller that was not connected to a process while we manipulated the error.

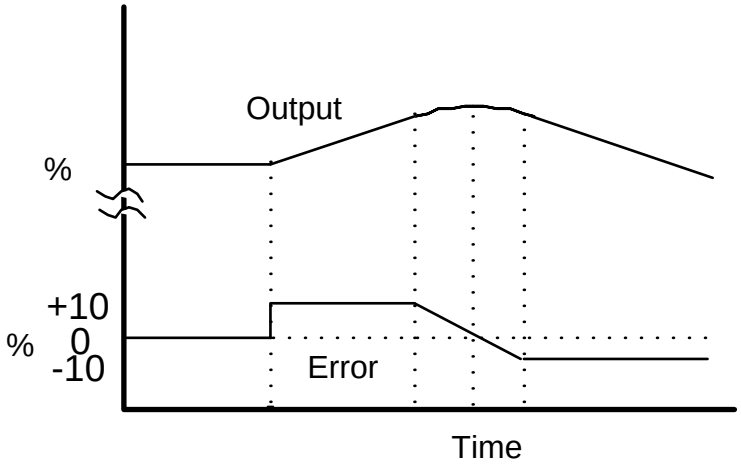


Figure 11 Output vs. error over time.

While the error is positive, the output ramps upward. While the error is negative the output ramps downward.

2.10 CALCULATION OF REPEAT TIME

Most controllers use both proportional action (gain) and reset action (integral) together. The equation for the controller is:

$$\mathbf{Out = g\,(\,e + K_r \int e\,dt\,)}$$

where **g** = gain

$\mathbf{K_r}$ = reset setting in repeats per minute.

If we look an open loop trend of a PI controller after forcing the error from zero to some other value and then holding it constant, we will have the trends shown in Figure 12.

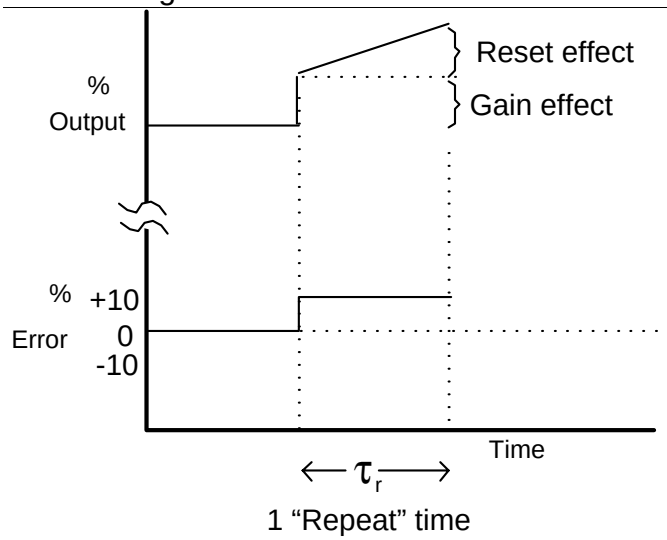


Figure 12 Calculation of repeat time

We can see two distinct effects of the change in the error. At the time the error changed the output also changed. This is the "gain effect" and is equal to the product of the gain and the change in the error. The second effect (the "reset effect") is the ramp of the output due to the error. If we measure the time from when the error is changed to when the reset effect is equal to the gain effect we will have the "repeat time." Some control vendors measure reset by repeat time (or "reset time" or "integral time") in minutes. Others measure reset by "**repeats per minute**." Repeats per minute is the inverse of minutes of repeat.

2.11 DERIVATIVE

Derivative is the third and final element of PID control. Derivative responds to the rate of change of the process (or error). Derivative is normally applied to the process only). It has also been used as a part of a temperature transmitter ("Speed-Act™" - Taylor Instrument Companies) to overcome lag in transmitter measurement. Derivative is also known as Preact™ (Taylor) and Rate.

The derivative contribution can be expressed mathematically:

$$\mathbf{Out} = \mathbf{g} \times \mathbf{K_d} \times \frac{\mathbf{de}}{\mathbf{dt}}$$

where **g** is gain,

K_d is the derivative setting in minutes, and

e is the error

The open loop response of controller with proportional and derivative is shown graphically:

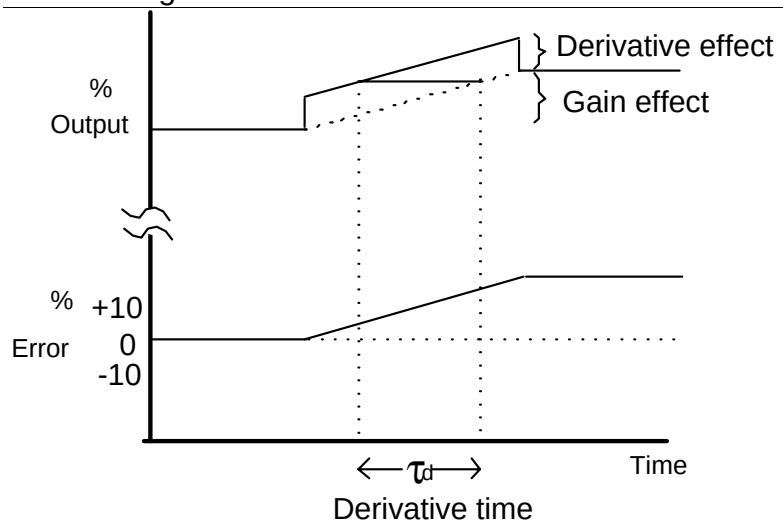


Figure 13 Output vs. error of derivative over time

The derivative advances the output by the amount of derivative time.

This diagram compares the output of a controller with gain only (dashed line) with the output of a controller with gain and derivative (solid line). The solid line is higher than the dashed line for the time that the process is increasing due the addition of the rate of change to the gain effect. We can also look at the solid line as being "leading" the dashed line by some amount of time (τ_d).

The amount of time that the derivative action advances the output is known as the "derivative time" (or Preact time or rate time) and is measured in minutes. All major vendors measure derivative the same: in minutes.

2.12 COMPLETE PID RESPONSE

If we combine the three terms (Proportional gain, Integral, and Derivative) we obtain the complete PID equation.

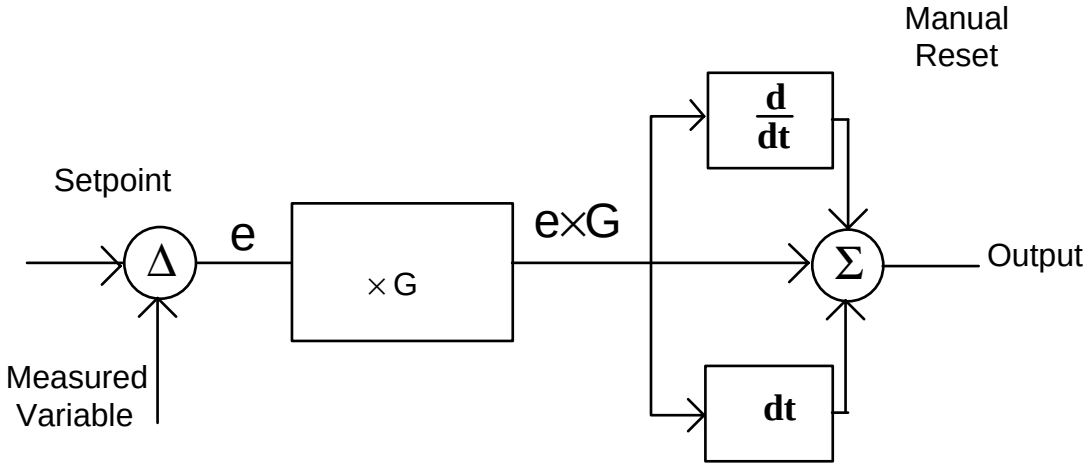


Figure 14 Combined gain, integral, and derivative elements.

This is a simplified version of the PID controller block diagram with all three elements, gain, reset, and derivative.

$$\text{Out} = G\left(e + R\int e\,dt + D\frac{de}{dt}\right)$$

Where

G = Gain
R = Reset (repeats per minute)
D = Derivative (minutes)

This is a general form of the PID algorithm and is close to, but not identical to, the forms actually implemented in industrial controllers. Modifications of this algorithm are described in the next chapter.

2.13 RESPONSE COMBINATIONS

Most commercial controllers allow the user to specify Proportional only controllers, proportional-reset (PI) controllers, and PID controllers that have all three modes. The majority of loops employ PI controllers. Most control systems also allow all other combinations of the responses: integral, integral-derivative, derivative, and proportional-derivative. When proportional response is not present the integral and derivative is calculated as if the gain were one.

CHAPTER 3 IMPLEMENTATION DETAILS OF THE PID EQUATION

The description of the PID algorithm shown on the previous page is a “text book” form of the algorithm. The actual form of the algorithm used in most industrial controllers differs somewhat from the equation and diagram of shown on the previous page.

3.1 SERIES AND PARALLEL INTEGRAL AND DERIVATIVE

The form of the PID equation shown Figure 14, which is the way the PID is often represented in text books, differs from most industrial implementations in the basic structure. Most implementations place the derivative section in series with the integral or reset section.

We can modify the diagram shown above to reflect the series algorithm:

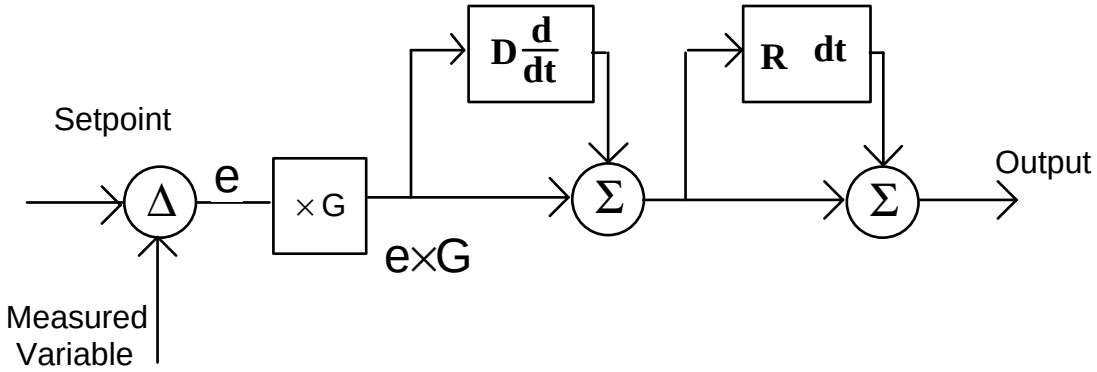


Figure 15 The series form of the complete PID response.

The difference between this implementation and the parallel one is that the derivative has an effect on the integration. The equation becomes:

$$\text{Out} = (RD+1)G\left(e + \frac{R}{(RD+1)}\int e\,dt + \frac{D}{(RD+1)}\frac{de}{dt}\right)$$

where R = the reset rate in repeats per minute,

D = the derivative in minutes,

and G = the gain.

The effect is to increase the gain by a factor of RD + 1, while reducing the reset rate and derivative time by the same factor. Based on common tuning methods, the derivative time is usually no more than about ¼ the reset time (1/R), therefore the factor RD+1 is usually 1.25 or less.

Almost all analog controllers and most commercial digital control systems use the series form. Such tuning methods as the Ziegler-Nichols methods (discussed in Chapter 6) were developed using series form controllers.

Unless derivative is used there is no difference between the parallel (non-interactive) and series (interactive) forms.

3.2 GAIN ON PROCESS RATHER THAN ERROR

The gain causes the output to change by an amount proportional to the change in the error. Because the error is affected by the set point, the gain will cause any change in the set point to change the output.

This can become a problem in situations where a high gain is used where the set point may be suddenly changed by the operator, particularly where the operator enters a new set point into a CRT. This will cause the set point, and therefore the output, to make a step change.

In order to avoid the sudden output change when the operator changes the set point of a loop, the gain is often applied only to the process. Set point changes affect the output due to the loop gain and due to the reset, but not due to the derivative.

3.3 DERIVATIVE ON PROCESS RATHER THAN ERROR

The derivative acts on the output by an amount proportional to the rate of change of the error. Because the error is affected by the set point, the derivative action will be applied to the change in the set point.

This can become a problem in situations where the set point may be suddenly changed by the operator, particularly in situations where the operator enters a new set point into a CRT. This causes the set point to have a step change. Applying derivative to a step change, even a small step change, will result in a “spike” on the output.

In order to avoid the output spike when the operator changes the set point of a loop, the derivative is often applied only to the process. Set point changes affect the output due to the loop gain and due to the reset, but not due to the derivative.

Most industrial controllers offer the option of derivative on process or derivative on error.

3.4 DERIVATIVE FILTER

The form of derivative implemented in controllers also includes filtering. The filter differs among the various manufactures. A typical filter comprises two first order filters that follow the derivative. The time constant of the filters depends upon the derivative time and the scan rate of the loop.

3.5 COMPUTER CODE TO IMPLEMENT THE PID ALGORITHM

There are many ways to implement the PID algorithm digitally. Two will be discussed here. In each case, there will be a section of code (in structured Basic,

easily convertible to any other language) that will be executed by the processor every second. (some other scan rate may be used, change the constant 60 to the number of times per minute it is executed.) In each code sample there is an IF statement to execute most of the code if the loop is in the auto mode. If the loop is in manual mode only a few lines are executed in order to allow for bumpless transfer to auto. Also, while the control loop is in manual, the output (variable OutP) will be operator adjustable using the operator interface software.

3.5.1 Simple PID code

One method of handling the integration and bumpless transfer to automatic mode is an algorithm that calculates the change in output from one pass to the next using the derivative of the PID algorithm, or:

$$\frac{\mathrm{dOut}}{\mathrm{dt}} = \mathrm{Gain} \times \left(\mathrm{ResetRate} \times \mathrm{Error} + \mathrm{Derivative} \times \frac{\mathrm{d}^2\mathrm{Error}}{\mathrm{dt}^2} \right)$$

This program is run every second. If the control loop is in manual, the output is adjusted by the operator through the operator interface software. If the control loop is in Automatic, the output is computed by the PID algorithm.

Each pass the output is changed by adding the change in output to the previous pass output. That change is found by adding:

- the change in error (Err-ErrLast)
- the error multiplied by the reset rate, and
- the second derivative of the error (Err-2*ErrLast+ErrLastLast) times the derivative.

The total is then multiplied by the gain.

This simple version of the PID controller work well in most cases, and can be tuned by the standard PID tuning methods (some of which are discussed later). It has “Parallel” rather than “Series” reset and derivative, and derivative is applied to the error rather than the input only.

Variables:

`Input`	*Process input*
`InputD`	*Process input plus derivative*
`InputLast`	*Process input from last pass, used in deriv. calc.*
`Err`	*Error, Difference between input and set point*
`ErrLast`	*Error from last pass*
`ErrLastLast`	*Error from next to last pass*
`OutP`	*Output of PID algorithm*
`Mode`	*value is ‘AUTO’ if loop is in automatic*
`Action`	*value is ‘DIRECT’ if loop is direct acting*

The PID emulation code:

```
1.   IF Mode = 'AUTO' THEN
2.     InputD=Input+(Input-InputLast)*Derivative*60    derivative
3.    InputLast = Input
4.    Err=InputD-SetP                Error based on reverse action
```

```
5.    IF Action = 'DIRECT' THEN
6.          Err=0 – Err          Change sign of error for direct action
7.    ENDIF
8.    OutP=OutP+Gain*(Err-ErrLast+Reset*Err+Deriv*(Err-ErrLast*2+ErrLastLast))
                                 Calculate the change in output using the derivative of the PID
                                 algorithm, then add to the previous output.
9.    ErrLastLast=ErrLast
10.   ErrLast=Err
11.  ELSE
12.   InputLast=Input            While loop in manual, stay ready for bumpless switch to Auto.
13.   ErrLastLast=Err
14.   ErrLast=Err
15.  ENDIF
16.  IF OutP > 100 THEN OutP=100  Limit output to between
17.  IF OutP < 0 THEN OutP=0      0 and 100 percent
```

The only serious problem with this form of the algorithm occurs when the output has reached an upper or lower limit. When it does, a change in the measurement can unexpectedly pull the output away from the limit. For example, Figure 16 illustrates the set point, measurement, and output of an open loop direct acting controller with a high gain and slow reset. When the input (blue line) rises above the set point (red line) the output (green line) first increases due to the proportional response and then continues to ramp up due to the reset response. The ramp ends when the output is limited at 100%.

Note the spike (noise) in the input at about 5 minutes. That spike results in a spike in the output, in the same direction. Compare this with the similar spike at about 61 minutes. Rather than cause an upward output spike as expected, the spike causes the output to pull away from the upper limit. It slowly ramps back to the limit. This is because the limit blocks the leading (increasing) side of the spike but does nothing to the trailing (decreasing) side of the spike.

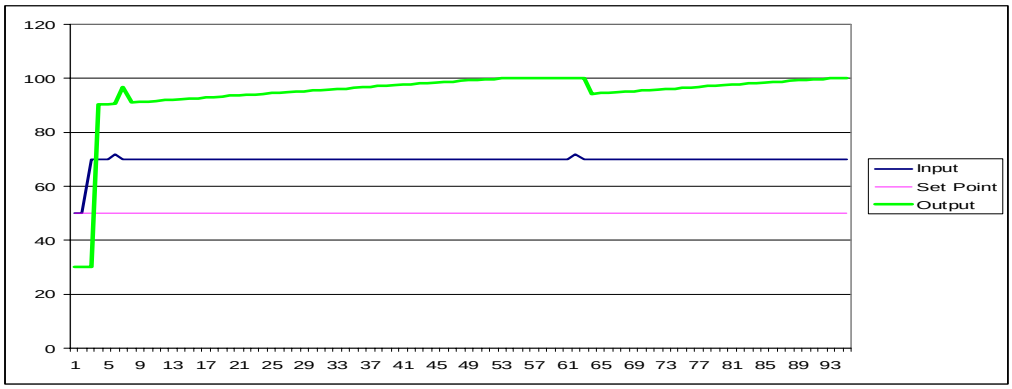


Figure 16 - Effect of input spike

When an input noise spike occurs while the output is below the limit, it causes the output to spike upwards. When the same spike occurs while the output is limited, the spike causes the output to pull away from the limit.

3.5.2 Improved PID code

The method of implementing automatic reset described in section 2.8, using a positive feedback loop (see Figure 10) was first used with pneumatic analog controllers. It can easily be implemented digitally.

There are several advantages of this algorithm implementation. Most important, it eliminates the problems that cause the output to pull away from a limit inappropriately. It also allows the use of external feedback when required.

Variables:

`Input`	*Process input*
`InputD`	*Process input plus derivative*
`InputLast`	*Process input from last pass, used in deriv. calc.*
`Err`	*Error, Difference between input and set point*
`SetP`	*Set point*
`OutPutTemp`	*Temporary value of output*
`OutP`	*Output of PID algorithm*
`Feedback`	*Result of lag in positive feedback loop.*
`Mode`	*value is 'AUTO' if loop is in automatic*
`Action`	*value is 'DIRECT' if loop is direct acting*

The PID emulation code:

```
1.   IF Mode = 'AUTO' THEN
2.      InputD=Input+(Input-InputLast)*Derivative*60   derivative.
3.      InputLast = Input
4.      Err=InputD-SetP              Error based on reverse action.
5.      IF Action = 'DIRECT' THEN Err=0 – Err  Change sign if direct.
6.      ENDIF
7.      OutPutTemp = Err*Gain+Feedback  Calculate the gain time the error and add the feedback.
8.      IF OutPutTemp > 100 THEN OutPutTemp =100   Limit output to between
9.      IF OutPutTemp < 0 THEN OutPutTemp =0       0 and 100 percent.
10.     OutP = OutPutTemp                          The final output of the controller.
11.     Feedback=Feedback+(OutP-Feedback)*ResetRate/60
12.  ELSE
13.     InputLast=Input     While loop in manual, stay ready for bumpless switch to Auto.
14.     Feedback=OutP
15.  ENDIF
```

If external feedback is used, the variable "OutP" in line 11 is replaced with the variable containing the external feedback.

CHAPTER 4 ADVANCED FEATURES OF THE PID ALGORITHM

4.1 RESET WINDUP

One problem with the reset function is that it may “wind up”. Because of the integration of the positive feedback loop, the output will continue to increase or decrease as long as there is an error (difference between set point and measurement) until the output reaches its upper or lower limit.

This normally is not a problem and is a normal feature of the loop. For example, a temperature control loop may require that the steam valve be held fully open until the measurement reaches the set point. At that point, the error will be cross zero and change signs, and the output will start decreasing, “throttling back” the steam valve.

Sometime, however, reset windup may cause problems. Actually, the problem is not usually the windup but the “wind down” that is then be required.

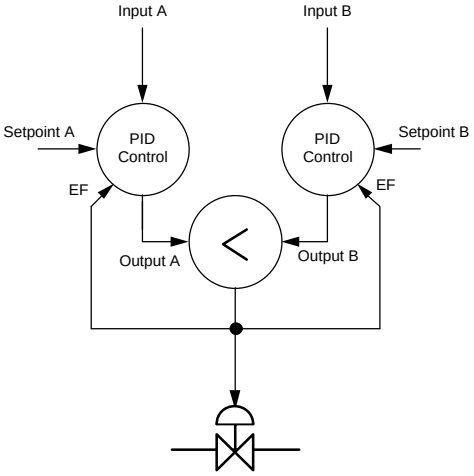


Figure 17 Two PID controllers that share one valve.

Suppose the output of a controller is broken by a selector, with the output of another controller taking control of the valve. In the diagram the lower of the two controller outputs is sent to the valve. Which ever controller has the lower output will control the valve. The other controller is, in effect, open loop. If its error would make its output increase, the reset term of the controller will cause the output to increase until it reaches its limit.

The problem is that when conditions change and the override controller no longer needs to hold the valve closed the primary controller’s output will be very far above the override signal. Before the primary controller can have any effect on the valve, it will have to “wind down” until its output equals the override signal.

4.2 EXTERNAL FEEDBACK

The positive feedback loop that is used to provide integration can be brought out of the controller. Then it is known as external feedback:

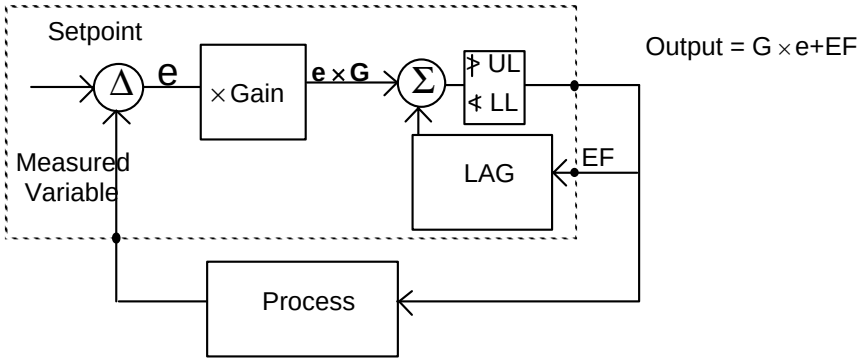


Figure 18 A proportional-reset loop with the positive feedback loop used for integration.

If there is a selector between the output of the controller and the valve (used for override control) the output of the selector is connected to the external feedback of the controller. This puts the selector in the positive feedback loop.

If the output of the controller is overridden by another signal, the overriding signal is brought into the external feedback. After the lag, the output of the controller is equal to the override signal plus the error times gain. Therefore, when the error is zero, the controller output is equal to the override signal. If the error becomes negative, the controller output is less than the override signal, so the controller regains control of the valve.

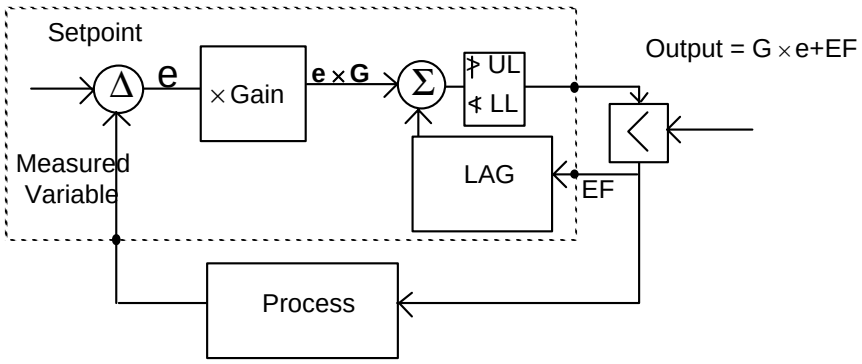


Figure 19 The external feedback is taken from the output of the low selector.

4.3 SET POINT TRACKING

If a loop is in manual and the set point is different from the process value, when the loop is switched to auto the output will start moving, attempting to move the process to the set point, at a rate dependent upon the gain and reset rate. Take for example a typical flow loop, with a gain of 0.6 and a reset rate of 20. If difference

between the set point and process is 50% at the time the loop is switched to automatic the output will ramp at a rate of 10%/second.

Often, when a loop has been in manual for a period of time the value of the set point is meaningless. It may have been the correct value before a process upset or emergency shutdown caused the operator to place the loop into manual and change the process operation. On a return to automatic the previous set point value may have no meaning. However, to prevent a process upset the operator must change the set point to the current process measurement before switching the loop from manual to automatic.

Some industrial controllers offer a feature called “set point tracking” that causes the set point to track the process measurement when the loop is not in automatic control. With this feature when the operator switches from manual to automatic the set point is already equal to the process, eliminating any bump in the process.

The set point tracking feature is typically used with loops that are tuned for fast reset, where a change from manual to automatic could cause the output to rapidly move, and for loops where the set point is not always the same. For example, the temperature of a room or an industrial process usually should be held to some certain value. The set point for the rate of flow of fuel to the heater is “whatever it takes” to maintain the temperature at its set point. Therefore flow loops are more likely to use set point tracking. However, this is a judgment that must be made by persons knowledgeable in the operation of the process.

CHAPTER 5 PROCESS RESPONSES

Loops are tuned to match the response of the process. In this chapter we will discuss the responses of the process to the control system.

The dynamic and steady state response of the process signal to changes in the controller output. These responses are used to determine the gain, reset, and derivative of the loop.

While discussing single loop control, we will consider the process response to be the effect on the controlled variable cause by a change in the manipulated variable (controller output).

5.1 STEADY STATE RESPONSE

The steady state process response to controller output changes is the condition of the process after sufficient time has passed so that the process has settled to new values.

The steady state response of the process to the controller output is characterized primarily by process action, gain, and linearity.

5.1.1 Process Action

Action describes the direction the process variable changes following a particular change in the controller output. A direct acting process increases when the final control element increases (typically, when the valve opens); a reverse acting process decreases when the final control element increases.

For example, if we manipulate the inlet valve on a tank to control level, an increase in the valve position will cause the level to rise. This is a direct acting process. On the other hand, if we manipulate the discharge valve to control the level, opening the valve will cause the level to fall. This is a reverse acting process.

5.1.2 Process Gain

Next to action, process gain is the most important process characteristic. The *process gain* (not to be confused with *controller gain*) is the sensitivity of the controlled variable to changes in a controller output. Gain is expressed as the ratio of change in the process to the change in the controller output that caused the process change.

From the standpoint of the controller, gain is affected by the valve itself, by the process, and by the measurement transmitter. Therefore the size of the valve and the span of the transmitter will affect the process gain.

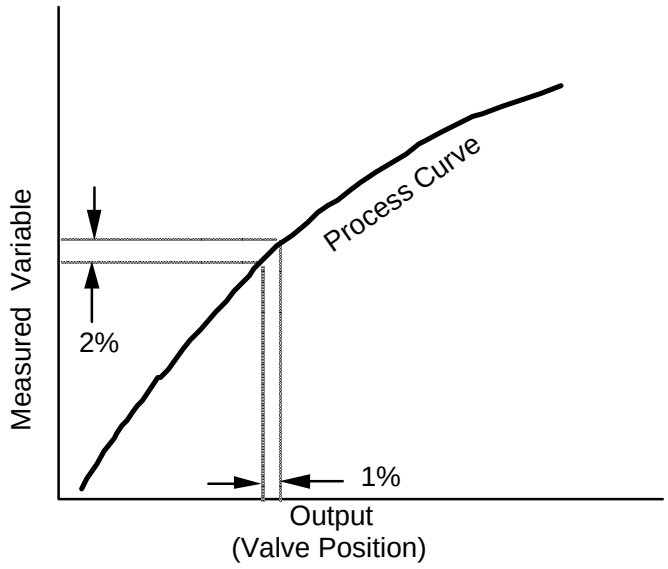


Figure 20 The direct acting process with a gain of 2.

In Figure 20 a 1% increase in the controller output causes the measured variable to increase by 2% . Therefore the process is direct acting and has a process gain of two.

5.1.3 Process Linearity

The gain of the process often changes based on the value of the controller output. That is, with the output at one value, a small change in the output will result in a larger change in the process measurement than the same output change at some other output value.

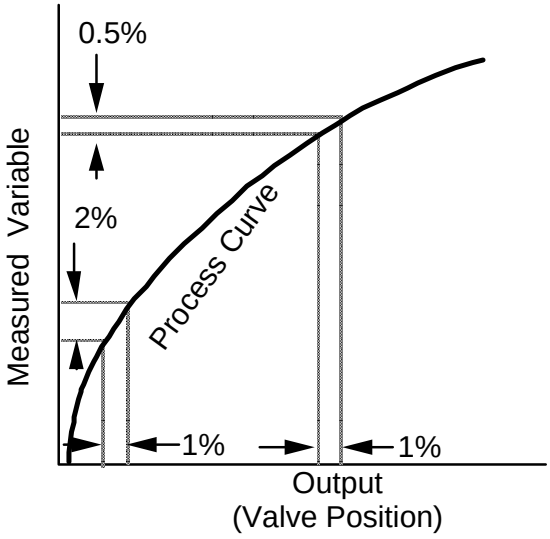


Figure 21 A non-linear process.

The process shown in Figure 21 is non linear. With controller output very low, a 1% increase in the output causes the measured variable to increase by 2%. When

the output is very high, the same 1% output increase causes the process to increase by only 0.5%. The process gain decreases when the output increases.

From the standpoint of controller tuning, the process linearity includes the linearity of the process, the final control element, and the measurement. It also includes any control functions between the PID algorithm and the output to the valve.

5.1.4 Valve Linearity

Valves may be linear or non-linear. A linear valve is one in which the flow through the valve is exactly proportional to the position of the valve (or the signal from the control system). Valves may fall into three classes (illustrated in Figure 22): linear, equal percentage, and quick opening.

Linear valves have the same gain regardless of the valve position. That is, at any point a given increase in the valve position will cause the same increase in the flow as at any other point.

Equal percentage valves have a low gain when the valve is nearly closed, and a higher gain when the valve is nearly open.

Quick opening valves have a high gain when the valve is nearly closed and a lower gain when the valve is nearly open.

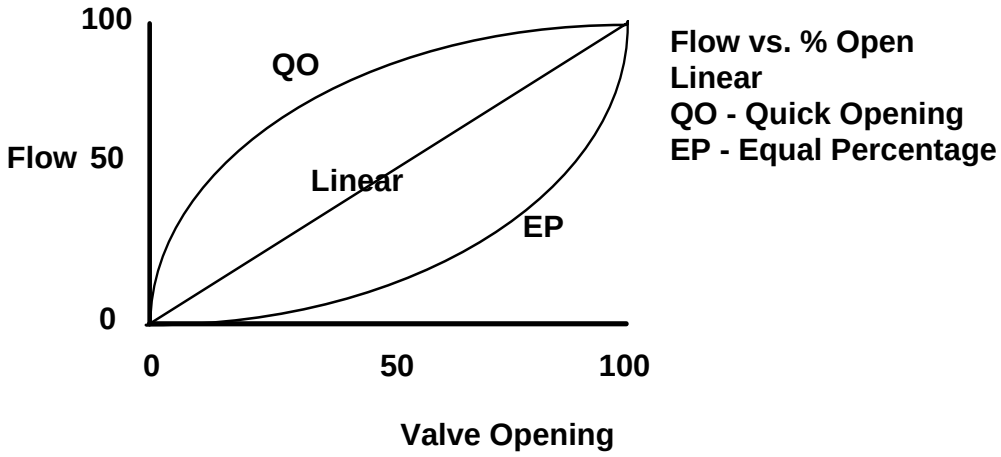


Figure 22 Types of valve linearity.

5.1.5 Valve Linearity: Installed characteristics

Even a linear valve does not necessarily exhibit linear characteristics when actually installed in a process. The characteristics described in the previous section are based on a constant pressure difference across the flanges of the valve. However, the pressure difference is not necessarily constant. When the pressure is a function of valve position, the actual characteristics of the valve are changed.

Take for example the flow through a pipe and valve combination shown in Figure 23. Liquid flows from a pump with constant discharge pressure to the open air. There is a pressure drop through the valve that is proportional to the square of the

flow. Assume that with a valve position of 10% the flow is 100 gpm. Also assume that with the particular size and length of the pipe 100 gpm causes a 10 psi pressure drop across each section of pipe. This leaves a net pressure from of 80 psi across the valve.

Assume now that we wish to double the flow rate to 200 gpm. By doubling the flow, we will increase the pipe pressure drop by a factor of four. With a pressure drop of 40 psi across each section of pipe, we will only have a valve differential pressure of only 20 psi. To make up for the loss of pressure, will have to increase the valve opening by a factor of four to make up for the pressure loss and by a factor of two to double the flow. Therefore, to double the flow rate we will have to open the valve from 10% to 80%. The so called linear valve now has the characteristics of a quick opening valve.

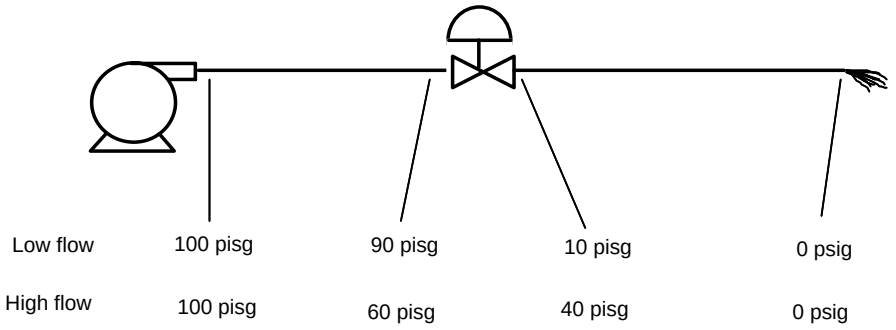


Figure 23 A valve installed a process line.

At *high* flow, the head loss through the pipe is more, leaving a smaller differential pressure across the valve.

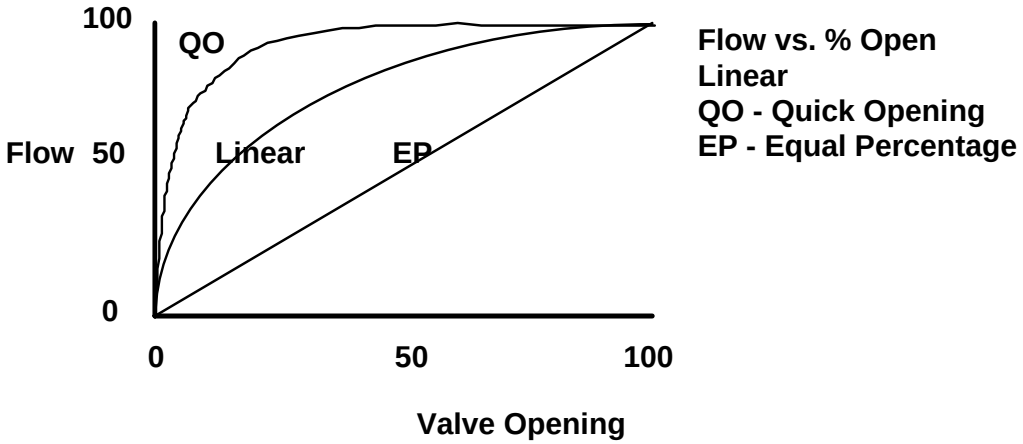


Figure 24 Installed valve characteristics.

5.2 PROCESS DYNAMICS

The measured variable does not change instantly with the controller output changes. Instead, there is usually some delay or lag between the controller output change and the measured variable change. Understanding the dynamics of the loop is required in order to know how to properly control a process.

There are two basic types of dynamics: simple lag and dead time. Most processes are a combination of several individual lags, each of which can be classed as simple lag or dead time.

5.2.1 Dead time

Dead Time is the delay in the loop due to the time it takes material to flow from one point to another. For example, in the temperature control loop shown below, it takes some amount of time for the liquid to travel from the heat exchanger to the point where the temperature is measured. If the temperature at the exchanger outlet has been constant and then changes, there will be some period of time before any change can be observed by the temperature measurement element. Dead time is also called distance velocity lag and transportation lag.

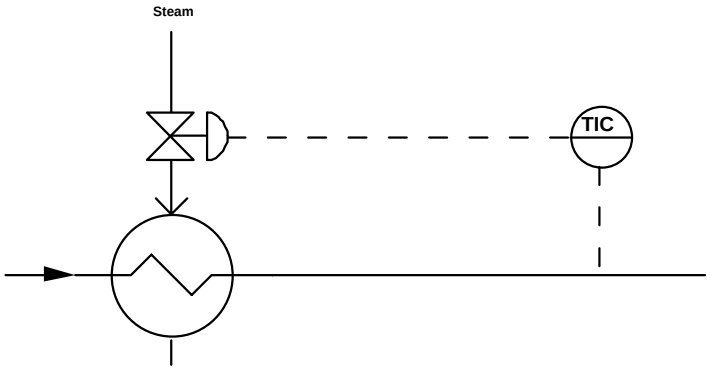
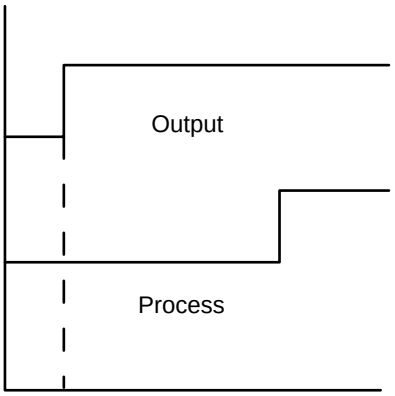


Figure 25 Heat exchanger with dead time

The distance between the heat exchanger and the temperature measurement creates a dead time.

Dead time is often considered to be the most difficult dynamic element to control. This will become apparent in Chapter 6 , controller tuning.



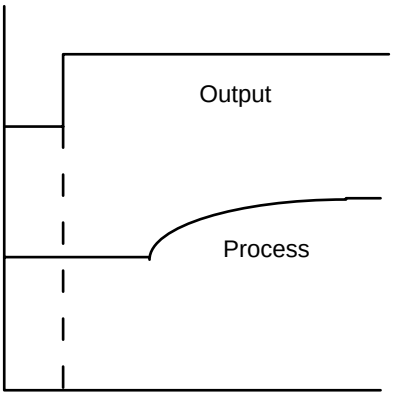


Figure 26 Pure dead time. **Figure 27 Dead time and lag.**

If a process contains both dead time and a lag, the beginning of the lag will be at the end of the dead time.

5.2.2 Self Regulation

For most processes, as a variable increases it will tend to reduce its rate of increase and eventually level off even without any change in the manipulated variable. This is referred to as *self regulation*.

Self regulation does not usually eliminate the need for a controller, because usually the value at which the variable will settle will be unacceptable. The control system will need to act to bring the controlled variable back to its set point.

An example of self regulation is a tank with flow in and out. The manipulated variable is the liquid flow into a tank. The controlled variable is the flow out of the tank. The load is the valve position of the discharge flow. With the valve position constant, the flow out of the tank is determined by the valve position and the level (actually the square of the level). As the level in the tank falls, the pressure (or liquid head) decreases, decreasing the flow rate. Eventually the discharge flow will decrease to the point that it equals the inlet flow, and the level will maintain a constant value. Likewise, if the flow into the tank increases, the level will begin to increase until the discharge flow equaled the inlet flow (unless the tank became full and overflowed first).

This also occurs in temperature loops. Take example, a room with an electric space heater with no thermostat. The room is too cool, so we turn on the space heater. As more heat enters the room, the room temperature increases. However, the flow of heat out through the walls is proportional to the difference between the inside and the outside temperatures. As the room temperature increases, that difference increases, and the heat flow from the room eventually equals the amount of heat produced by the space heater. As the temperature increases the rate of change decreases until the temperature levels off at a higher temperature.

Sometimes the self regulation is sufficient to eliminate any need for feedback control. However, more often the self regulation is not sufficient (the tank overflows or the room becomes too hot), therefore control is still needed. Because of self regulations, for at least some range of controller outputs there will be a corresponding process value.

The self regulation is responsible for the curve shown in the dynamic response of a controlled variable to a change in the measured variable.

5.2.3 Simple lag

The most common dynamic element is the simple lag. If a step change is made in the controller output , the process variable will change as shown in Figure 28.

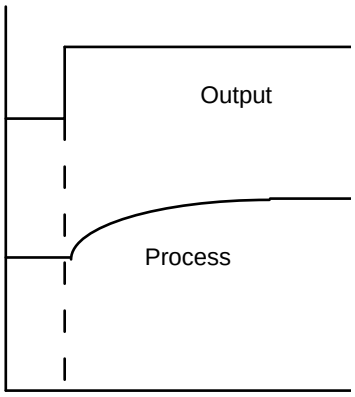


Figure 28 Process with a single lag.

An example of a process dominated by one loop is shown in Figure 29. The flow of the liquid out of the vessel is proportional to the level. If the inlet valve is opened, increasing the flow into the vessel, the level will rise. As the level rises, the flow output will rise, slowing the rate of increase in the level. Eventually, the level will be at the point where the flow out will be equal to the flow in.

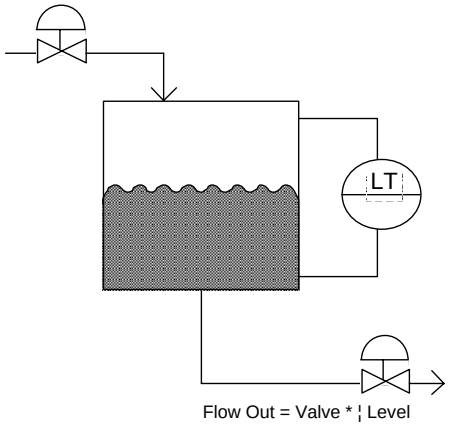


Figure 29 Level is a typical one lag process.

5.2.4 Multiple Lags

Most processes have more than one lag, although some of the lags may be insignificant. Lags are not additive. A response of a multiple lag is illustrated in Figure 30.

The process measured variable begins to change very slowly, and the rate of change increases up to a point, known as the point of inflection, where the rate of change decreases as the measurement approaches its asymptote.

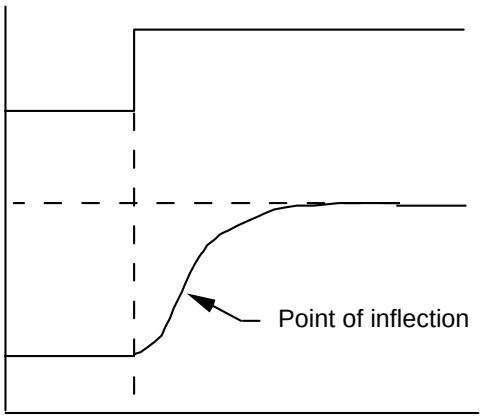


Figure 30 Process with multiple lags.

The first part of the curve, where the rate of change is increasing, is governed primarily by the second largest lag. The second part of the curve, beyond the point of inflection, is governed primarily by the largest lag.

5.2.5 Process Order

Often processes have been described as first order, second order, etc., based on the number of first order linear lags included in the process dynamics. It can be argued that all processes are of a higher order, with a minimum of three lags and a dead time. These lags, which are present in all processes, include the lag inherent in the sensing device, the primary lag of the process, and the time that the valve (or other final control element) takes to move. However, in many processes the smaller lags are so much smaller than the largest lag that their contributions to the process dynamics are negligible.

Dead time is also present in all processes. With pneumatic control, there is some dead time due to the transmission of the pressure signal from the process to the controller, and from the controller to the valve. This is eliminated by electronic controls (unless one considers the transmission of the electric signal, usually a few microseconds or less). With digital controls, there is an effective dead time equal to one half the loop scan rate [2]. In most cases, the loop will be scanned fast enough so that this dead time is insignificant. In some cases, such as liquid flow loops, this dead time is significant and affects the amount of gain that can be used.

Rather than consider a process to be first order, second order, etc., it may be better to consider all loops to be higher order to a degree. As an alternative to process

order, we will characterize processes by the degree to which one first order lag dominates the other lags in the process (not considering any true dead time). *Dominant-lag* processes are those that consist of a dead time plus a single significant lag, with all other lags small compared to the major lag. *Multiple-lag* or *non-dominant-lag* processes are those in which the longest lag is not significantly longer than the next longest lag. One measure of the dominance of a single lag is the value of the process measurement at which the point of inflection (POI) occurs.(see Figure 31) In the most extreme case (only a single lag) the POI occurs at the initial process value. With about three equal, non-interacting lags the POI occurs at about 33% of the difference between the initial and the final process value.[7]

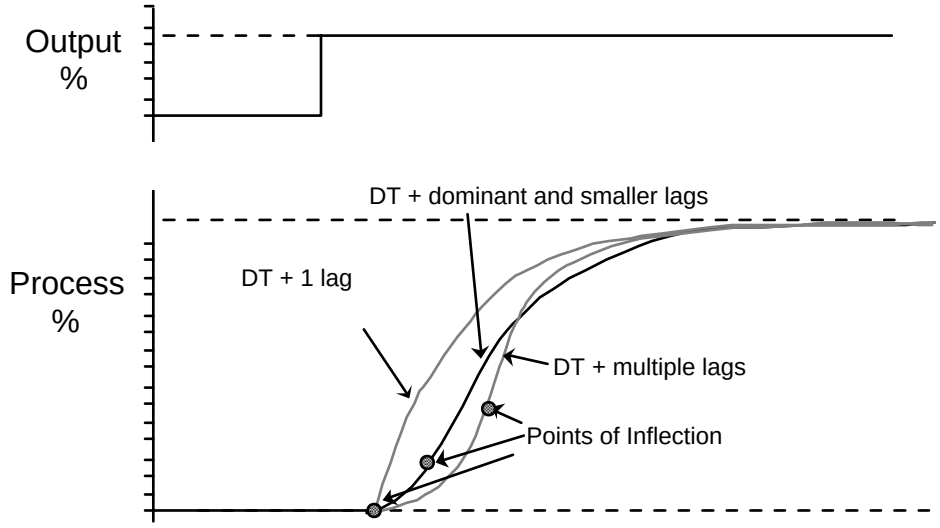


Figure 31 The step response for different numbers of lags.

As the number of lags increase, the value of the process at the point of inflection increases.

5.3 MEASUREMENT OF PROCESS DYNAMICS

Process dynamics usually consist of several lags and dead time. The dynamics differ from one loop to another. The dynamics can be expressed by a detailed list of all of the lags and the dead time of the loop, or they can be approximated using a simpler model.

One such model is a dead time and a first order lag. Graphically, the process response of such a model is:

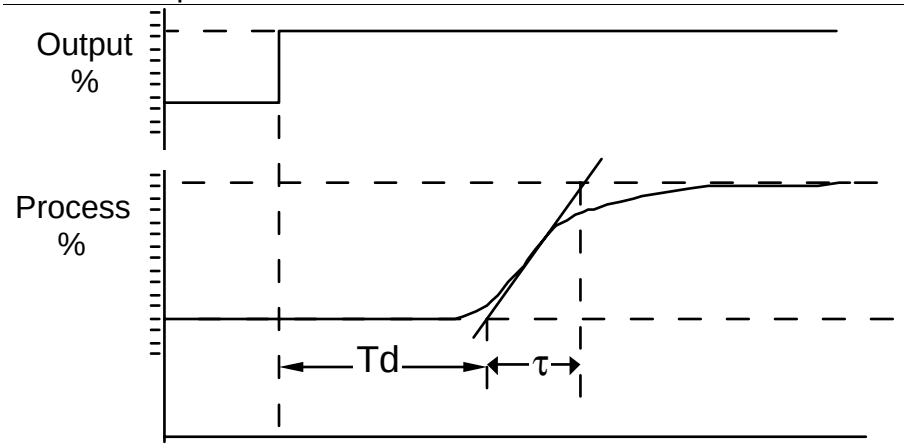


Figure 32 Pseudo dead time and process time constant.

The dynamics can be approximated by two numbers: τ is the process time constant. It is approximately equal to the largest lag in the process. Td is the pseudo dead time and approximates the sum of the dead time plus all lags other than the largest lag.

5.3.1 First Order Plus Dead Time Approximation

Several tuning methods (such as the Ziegler-Nichols open loop method) are based on an approximation of the process as a combination of a single first order lag and a dead time, known as the First Order Plus Dead Time (FOPDT) model. These methods identify the process by making a step change in the controller output. The process trend is recorded and graphical or mathematical methods are used to determine the process gain, dead time, and first order lag.

Process gain is the ratio of the change in the process to the change in the controller output signal. It depends upon the range of the process measurement and includes effects of the final control element.

Pseudo dead time (Td) is the time between the controller output change and the point at which the tangent line crosses the original process value. The pseudo dead time is influenced by the dead time and all of the lags smaller than the longest lag in the process.

Process time constant (τ) is the rate of change of the process measurement at the point at which the rate of change is the highest. The time constant is strongly influenced by the longest lag in a multiple lag process.

The ratio of the pseudo dead time to the process time constant is often referred to as an “uncontrollability” factor (Fc) that is an indication of the quality of control that can be expected. The gain (for a P, PI, and PID controller) at which oscillation will become unstable is inversely proportional to this factor. Smith, Murrill, and Moore, [5] proposed that the factor be modified by adding one half of the sample time to the dead time for digital controllers.

5.4 LOADS AND DISTURBANCES

The process measurement is affected not only by the output of the control loop but by other factors called loads. These can include such factors as the weather, the position of other valves, and many other factors.

An example is shown in Figure 33. The level of the tank is controlled by manipulating the valve on the discharge line. However, the level is also affected by the flow into the tank. In fact, the flow into the tank has just as much effect on the level as the flow out of the tank. The inlet flow is therefore a load.

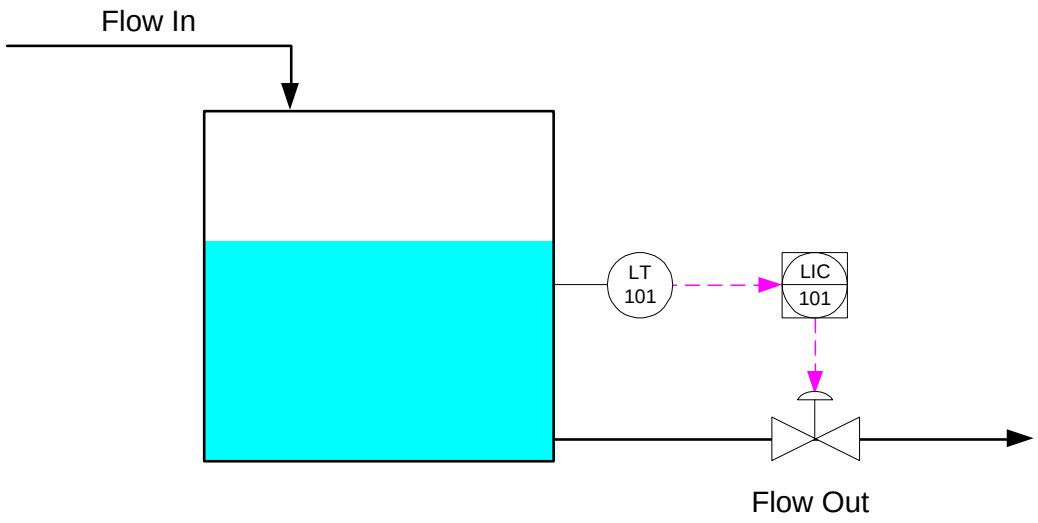


Figure 33 Level control

The level is the controlled variable, the flow out is the manipulated variable, and the flow in is the load. A change in the flow in is a disturbance that requires a response by the controller.

In a steam heater, the temperature is controlled by the valve on the steam line. However, the temperature is also affected by the temperature of the air around the vessel, although not nearly as much as by the steam valve. This temperature outside of the vessel is also a load.

The changes in a load are called *disturbances*. Almost all processes contain disturbances. They can be as major as the effect of the inlet flow on the vessel or as minor as the effect of weather on the temperature loop.

CHAPTER 6 LOOP TUNING

Once a loop is configured and started up, in order for it to work correctly someone has to put the correct gain, reset, and derivative values into the PID control algorithm.

6.1 TUNING CRITERIA OR “HOW DO WE KNOW WHEN ITS TUNED”

One of the most important, and most ignored, facets of loop tuning is the determination of the proper tuning of a loop.

6.1.1 The extremes: instability or no response

The loop performance must fall between two extremes. First, the loop must respond to a change in set point and to disturbances. That is, an error, or difference in the process and the set point, must eventually result in the manipulation of the output so that the error is eliminated. If the gain, reset, and derivative of the loop are turned to zero there will be no response.

The other extreme is instability. An unstable loop will oscillate without bound. A set point change will cause the loop to start oscillating, and the oscillations will continue. At worst, the oscillations will grow (or diverge).

Proper tuning of a loop will allow the loop to respond to set point changes and disturbances without causing instability.

6.1.2 Informal methods

There are several rules of thumb for determining how the quality of the tuning of a control loop.

(1/4 wave decay).

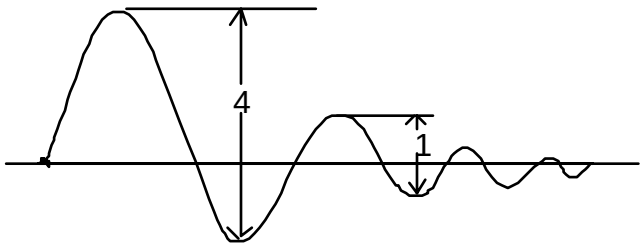


Figure 34 Quarter wave decay.

Traditionally, quarter wave decay has been considered to be the optimum decay ratio. This criterion is used by the Ziegler Nichols tuning method, among others. There is no single combination of tuning parameters that will provide quarter wave decay. If the gain is increased and the reset rate decreased by the correct amount the decay ratio will remain the same.

Quarter wave decay is not necessarily the best tuning for either disturbance rejection or set point response. However, it is a good compromise between instability and lack of response.

For some loops the objective of the tuning is to minimize the overshoot following a set point change.

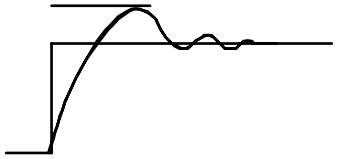


Figure 35 Overshoot following a set point change.

e rejection.

For other loops the primary concern is the reduction of the effect of disturbances.

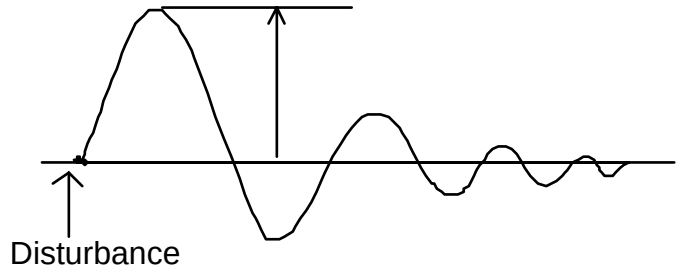


Figure 36 Disturbance Rejection.

The choice of methods depends upon the loop's place in the process and its relationship with other loops.

6.2 MATHEMATICAL CRITERIA—MINIMIZATION OF INDEX

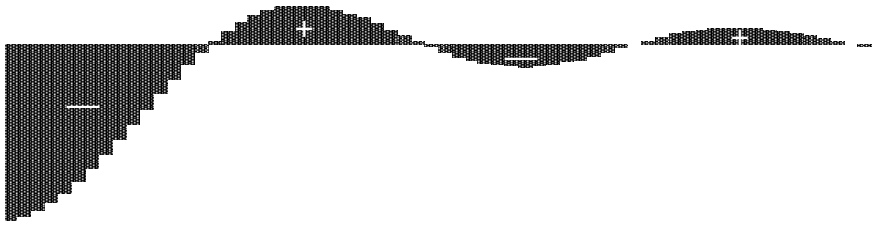


Figure 37 Integration of error.

There are several criteria for evaluating tuning that are based on integrating the error following a disturbance or set point change. These methods are not used to test control loops in actual plant operation because the usual process noise and

random disturbances will affect the outcome. There are used in control theory education and research using simulated processes. The indices provide a good method of comparing different methods of controller tuning and different control algorithm.

IAE - Integral of absolute value of error

$$\int |e| \, dt$$

ISE - Integral of error squared

$$\int e^2 \, dt$$

ITAE - Integral of time times absolute value of error

$$\int t \, |e| \, dt$$

ITSE - Integral of time times error squared:

$$\int t \, e^2 \, dt$$

Of these methods, the IAE and ISE are the most common.

6.3 ZIEGLER NICHOLS TUNING METHODS

In 1942 J. G. Ziegler and N. B. Nichols, both of the Taylor Instrument Companies (Rochester, NY) published a paper [1] that described two methods of controller tuning that allowed the user to test the process to determine the dynamics of the process. Both methods assume that the process can be represented by the model (described above) comprising the process gain, a “pseudo dead time”, and a lag. The methods provide a test to determine process gain and dynamics and equations to calculate the correct tuning.

The Ziegler Nichols methods provide quarter wave decay tuning for most types of process loops. This tuning does not necessarily provide the best ISE or IAE tuning but does provide stable tuning that is a reasonable compromise among the various objectives. If the process does actually consist of a true dead time plus a single first order lag, the Z-N methods will provide quarter wave decay. If the process has no true dead time but has more than two lags (resulting in a “pseudo dead time”) the Z-N methods will usually provide stable tuning but the tuning will require on-line modification to achieve quarter wave decay.

Because of their simplicity and because they provides adequate tuning for most loops, the Ziegler Nichols methods are still widely used.

6.3.1 Determining the First Order Plus Dead Time model

The Ziegler Nichols method, as well as several other methods for controller tuning, rely on a model of the process that comprises one first order lag plus dead time. The FOPDT model parameters can be determined from the actual process using a simple process reaction test. The output from the controller is increased (or decreased) in a step change, and the reaction of the process is recorded. The process gain is the ratio of the change in the process (in percent) to the change that had been made in the controller output.

Several methods have been proposed to calculate the pseudo dead time, and time constant from the reaction curve.

Ziegler and Nichols proposed a graphical method (Figure 38) using a tangent line drawn through the steepest part of the curve (the point of inflection). The line continues below the original process value. The time between the output change and the point at which the tangent line crosses the original process line is called the lag (the term Pseudo Dead Time will be used in this paper). The slope of the line is then calculated. The original Ziegler-Nichols formulas used the slope or the rate of change rather than the time.

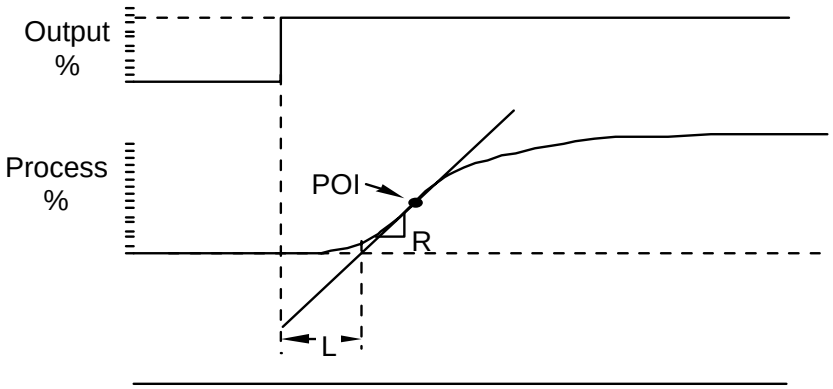


Figure 38 The Ziegler-Nichols Reaction Rate method.

Another graphical method (Figure 39), which is the mathematical equivalent of the original Ziegler-Nichols method, is known as the “tangent method”. In this method the same tangent line is drawn, but the process time constant is the time between the interception of the tangent and the original process value line and the eventual process value line. The formulas that are commonly provided for the Ziegler-Nichols open loop method use the process time constant.

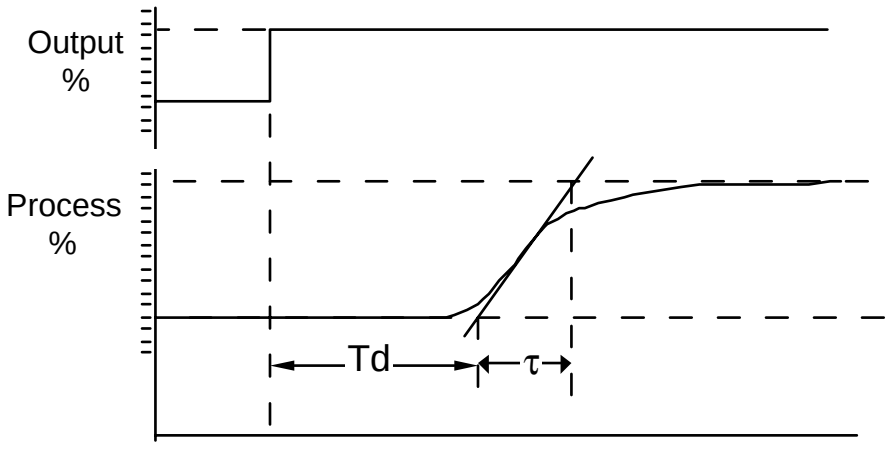


Figure 39 Tangent method.

These two methods will provide identical results when applied perfectly, that is, no error in the drawing of the line and no noise in the process signal.

Two additional methods are the mathematical equivalent of the previous two only if the process dynamics really did comprise only a single order lag and a dead time. When the process differs from this model, the following two methods will provide different results that may actually provide better tuning.

The first of these is sometimes known at the “tangent-and-point method” (T+P) Figure 40 [6]. In this method the same tangent line is drawn as before and used to calculate the pseudo dead time as before. However, a point equal to 63.2% of the value between the original and the ultimate process measurement is made on the tangent line. The time between the end of the pseudo dead time and the time at which the tangent line goes through the 63.2% point is the process time constant. This method will give the same results as the first two when the process is truly a dead time plus first order lag. As the values of the smaller lags increase, the tangent and point method gives a smaller time constant than the two graphical methods.

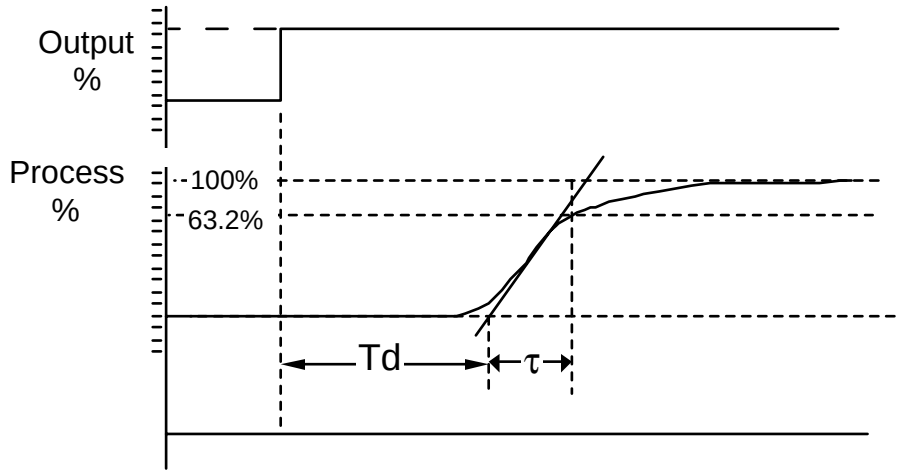


Figure 40 The tangent plus one point method.

Another variation is the “two point method”, proposed by C. Smith [3, p141], illustrated in Figure 41. This method does not require the drawing of a tangent line but measures the times at which the process changes by 28.3% (t_1) and 63.2% (t_2) of the total process change. Then two formulas are used to calculate the pseudo dead time and the process time constant.

process time constant	$\tau = 1.5(t_1 - t_2)$
pseudo dead time	$T_d = t_1 - \tau$

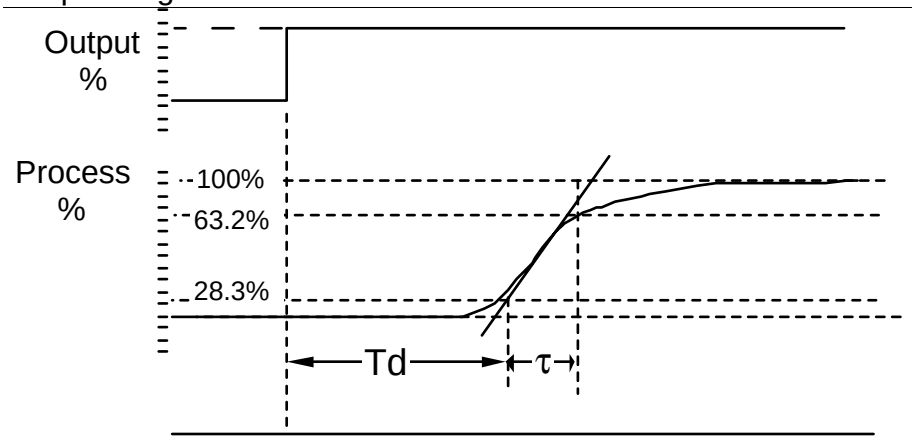


Figure 41 The two point method.

One advantage of the two point method is that it does not require drawing the tangent line. This improves the accuracy, particularly for processes with noise. This method will provide the same pseudo dead time and time constant when the process dynamics are dominated by a true dead time and single lag. Like the tangent and point method, the two point method provides different values for the pseudo dead time and the process time constant when the process dynamics comprise multiple lags. When used with the multiple lag processes, the one point and two point methods also provide better tuning. In table 3, tests within each set all result in the same pseudo dead time and time constant using the tangent line method. However, using the two point method the pseudo dead time and time constant are different. For multiple lag processes the two point method results in a longer dead time and shorter time constant than tangent methods. This provides more conservative tuning using any of the FOPDT tuning methods.

6.3.2 Ziegler Nichols open loop tuning method

The first method is the open loop method, also known as the “reaction curve” method. This method calculates the actual values of the of the assumed process model (the gain, pseudo dead time, and lag).

For this method to work the process must be “lined out”, that is, not changing. With the controller in manual, the output is changed by a small amount. The process is then monitored.

The following values are calculated using one of the methods described above:

Kp Process Gain

τ Process Time Constant

T_d Pseudo Dead Time

The gain, reset, and Preact are calculated using:

	Gain	Reset	Derivative
P	$\frac{\tau}{T_d\ Kp}$		
PI	$0.9\frac{\tau}{T_d\ Kp}$	$\frac{.3}{T_d}$	
PID	$1.2\frac{\tau}{T_d\ Kp}$	$\frac{.5}{T_d}$	.5Td

6.3.3 Ziegler Nichols closed loop tuning method

The closed loop (or ultimate gain method) determines the gain that will cause the loop to oscillate at a constant amplitude. Most loops will oscillate if the gain is increased sufficiently.

The following steps are used:

1. Place controller into automatic with low gain, no reset or derivative.
2. Gradually increase gain, making small changes in the set point, until oscillations start.
3. Adjust gain to make the oscillations continue with a constant amplitude.
4. Note the gain (Ultimate Gain, $G_{u,}$) and Period (Ultimate Period, $P_{u.}$)

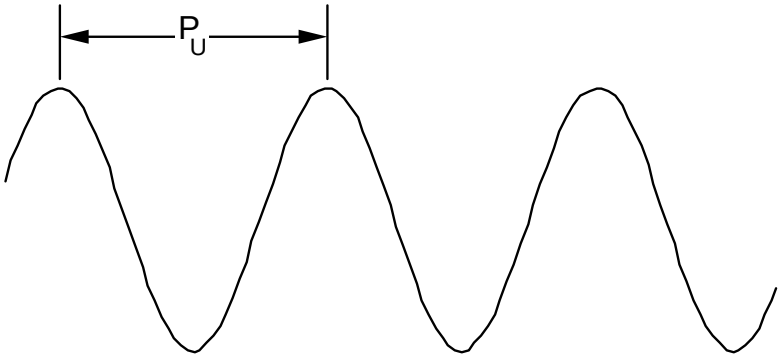


Figure 42 Constant amplitude oscillation.

The gain, reset, and Preact are calculated using:

	Gain	Reset	Preact
P	$0.5\ G_U$	—	—
PI	$0.45\ G_U$	$\frac{1.2}{P_U}$	—
PID	$0.6\ G_U$	$\frac{2}{P_U}$	$\frac{P_U}{8}$

6.4 COHEN-COON

The Cohen-Coon method is similar to the Ziegler-Nichols reaction rate method in that it makes use of the FOPDT model to develop the tuning parameters. The parameters (shown below) are more complex, involving more arithmetic

operations. As can be seen from the tables, the Cohen-Coon method will result in a slightly higher gain than the Ziegler-Nichols method. For most loops it will provide tuning closer to quarter wave decay and with a lower ISE index than the Ziegler-Nichols open loop method [4].

	Gain x Kp	Reset Rate/τ	Derivative/τ
P	$\frac{\tau}{T_d}$ +.333		
PI	$.9\frac{\tau}{T_d}$ + .082	$\frac{3.33(\tau/Td)[1+(\tau/Td)/11]}{1+2.2(\tau/Td)}$	
PID	$1.35\frac{\tau}{T_d}$ + .27	$\frac{2.5(\tau/Td)[1+(\tau/Td)/5]}{1+.6(\tau/Td)}$	$\frac{0.37(\tau/Td)}{1+0.2(\tau/Td)}$

6.5 LOPEZ IAE-ISE

Integral Absolute Error and Integral Squared Error are two methods of judging the tuning of a control loop. (see below). A method of selecting tuning coefficients to minimize the IAE or ISE criteria for disturbances was developed by Lopez, et. al. [5].

This method uses the FOPDT model parameters and a set of equations (Table 9) to calculate the tuning parameters. Tests show that the parameters provide results close to the minimum IAE or ISE, particularly when the actual process dynamics are similar to the FOPDT model (the process contains a true dead time and one lag). When the process has multiple lags the equations do not provide the best possible tuning, but they still provide better tuning (lower IAE and ISE indices) than the other methods.

	Gain x Kp	Reset Rate	Derivative
P	$1.411Td/\tau^{-.917}$		
PI	$1.305Td/\tau^{-.959}$	$(\tau/.492)(Td/\tau)^{.739}$	
PID	$1.495Td/\tau^{-.945}$	$(\tau/1.101)(Td/\tau)^{..771}$	$.56\tau(Td/\tau)^{1.006}$

6.6 CONTROLLABILITY OF PROCESSES

The gain at which a loop will oscillate depends upon the dynamics of the loop. In general, a loop that has no dynamic elements other than one first order lag will not oscillate at any gain. A loop with dead time or with multiple lags will oscillate at some gain.

If we refer to the model used on the Ziegler Nichols open loop test, the gain at which a loop will exhibit undamped, sustained oscillations (the ultimate gain in

the Ziegler Nichols closed loop test) will depend upon the ratio of the process time constant (τ) to the pseudo dead time (Td).

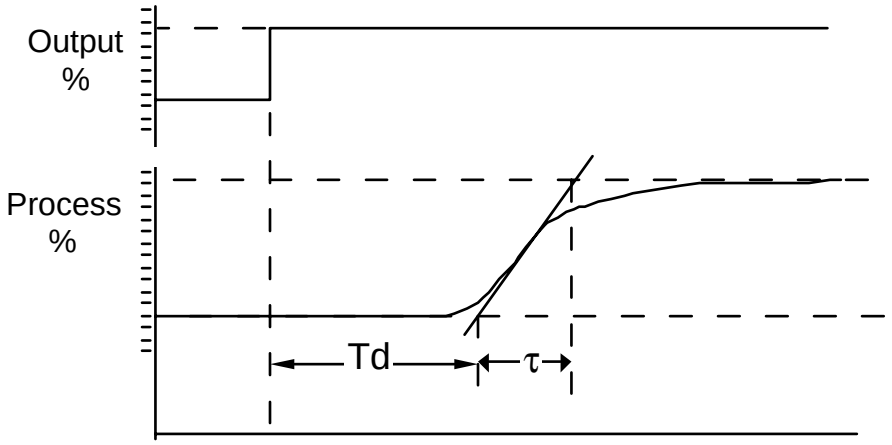


Figure 43 Pseudo dead time and lag.

The importance of this fact goes beyond finding the best tuning parameters. There is an advantage to a loop that can have a higher gain. If a loop can have a higher gain it will have greater rejection of disturbances and will respond more rapidly to set point changes. Therefore, it is advantageous to be able to increase the gain if doing so will not cause the loop to become unstable.

Remember that the time constant is proportional to the largest lag in the system. The pseudo dead time is based on the dead time and all other lags. The allowable gain (and the gain required for quarter wave decay) can be increased by increasing the ratio $\frac{t}{Td}$. An increase in either dead time or in any lag other than the longest lag will decrease the ratio and therefore decrease the allowable gain. The loop scan period has the effect of process dead time. Increasing the scan period will decrease the ratio and the allowable gain. Also, adding any lag smaller than the longest lag, (for example, adding a large well to a thermocouple or a filter to a noisy loop) will decrease the allowable gain.

6.7 FLOW LOOPS

Flow loops are too fast to use the standard methods of analysis and tuning. The speed of the loop compared to the update rate of the display prevents collection of the data for either the open loop or closed loop methods. However, there are some rules of thumb regarding the tuning of flow loops.

Typically, the tuning of a flow loop using digital control is:

Gain = 0.5 to 0.7
Reset = 15 to 20 repeats/min.
No Derivative

6.7.1 Analog vs. Digital control:

The guidelines above are based on digital controller such as that used in Distributed Control Systems. Some flow loops using analog controllers are tuned with higher gain. However, the same loop may go unstable if a digital controller is used.

While normally a digital control system will provide response very similar to that of an analog control system, the performance can be quite different with fast loops such as flow. With an analog controller, the flow loop has a time constant (τ) of a few seconds and pseudo dead time much smaller than one second. However, with a digital controller, the scan rate of the controller can be considered dead time. Although this dead time is small, it is large enough when compared to τ to cause instability when the gain is higher than one.

CHAPTER 7 MULTIPLE VARIABLE STRATEGIES

In previous chapters we have looked at a single PID controller, the process response of a single variable to a change in a single manipulated variable. However, real processes usually have more than one process measurement and more than one variable. In the next chapters we will examine several common control strategies that involve multiple process measurements and sometimes multiple outputs to the process.

Multiple variable strategies have several forms:

In Cascade Control (Chapter 8) there is one controlled variable, to be held at its set point. However, a second measured variable may be used to assist in the control of the primary controlled variable.

Ratio Control (Chapter 9) maintains a single control loop (a flow loop) at its set point, however that set point is manipulated to maintain that flow at a desired ratio to another flow.

In Override control (Chapter 10) a control loop normally maintains a single controlled variable at its set point, however, the manipulated variable will be taken over if necessary to prevent some other variable from going outside a limit.

While feedback control compares a measured variable to its set point to manipulate the output, feed forward control (Chapter 11) uses a measurement of the disturbance to manipulate the output before the controlled variable has departed from the set point.

CHAPTER 8 CASCADE

Cascade is the most common form of multiple variable control. In cascade control the controller manipulates the set point of another controller rather than a final control element. In this chapter we will discuss when it is used, how it is applied, and how it is tuned.

8.1 BASICS

In many processes there some process variables that we want to hold at a specific desired value, and other intermediate variables that change as necessary to hold the first type of variable at its desired set point. For example, in Figure 44 the temperature of liquid leaving the heat exchanger is our controlled variable, the one we want to hold at a set point, and we will do so by manipulating the steam valve. The flow of steam to the exchanger directly affects the temperature, but we do not care how much is flowing as long as the temperature is controlled.

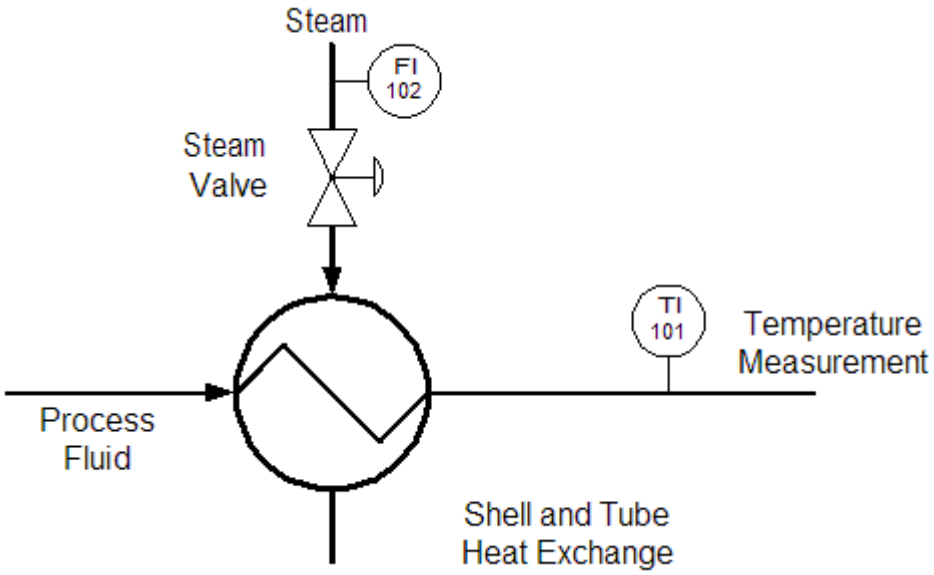


Figure 44 - Heat exchanger

The amount of steam required will depend upon the flow rate of process fluid and the difference between the inlet temperature and the set point of the outlet temperature.

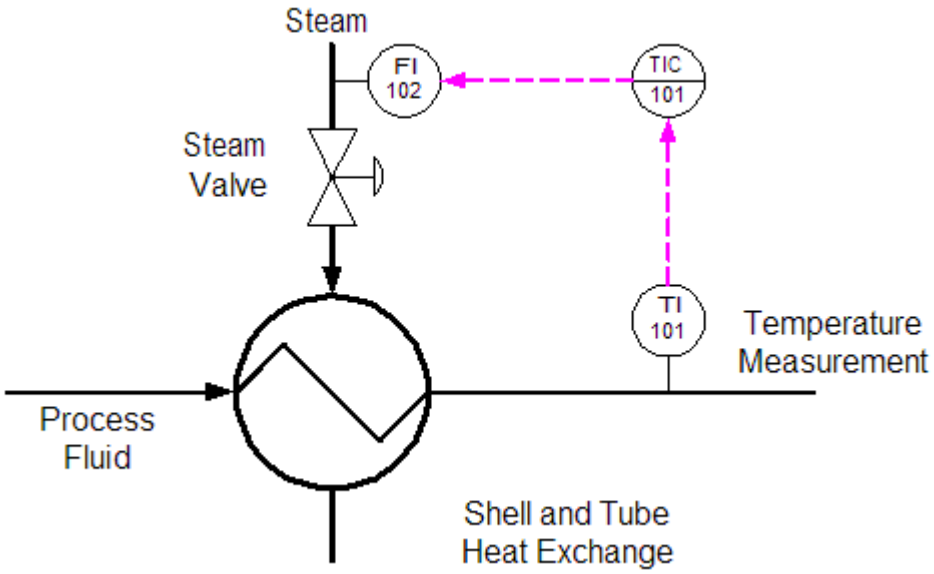


Figure 45 - Heat exchanger with single PID controller.

We can control the temperature using a single PID controller with the temperature as its input and its output connected to the steam valve, as shown in Figure 45. This arrangement will control the temperature. However, there are some problems in the control:

The steam header pressure may change, causing a sudden reduction in the steam flow to the exchanger. The temperature controller will bring the temperature back to its set point, but, because of the slow tuning required of the temperature controller, the correction will take longer than desired.

The temperature loop, likely containing multiple lags and dead time, is a more difficult loop to tune. Non-linearities in the valve will further complicate the tuning.

Use of cascade control, as shown in Figure 46, will correct both of these problems. If the header pressure changes, causing a change in the flow, that change will be detected by the flow measurement and immediately corrected.

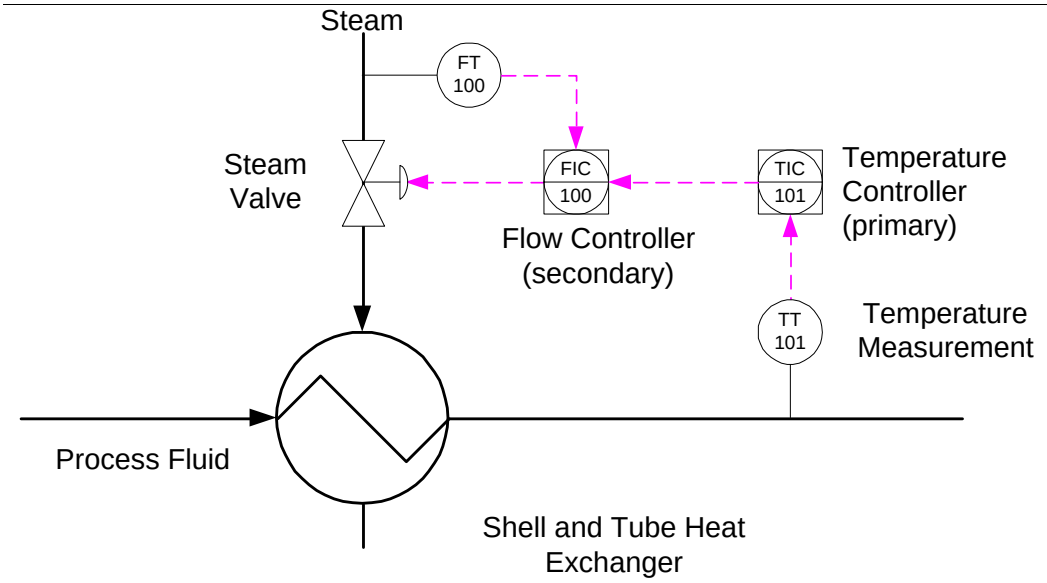


Figure 46 - Heat exchanger with cascade control.

8.2 CASCADE STRUCTURE AND TERMINOLOGY

The “outer” loop of a cascade loop pair is called the *primary* loop. This loop controls the variable that must be held at a specific set point. The “inner” loop of the pair is called the *secondary* loop. This loop controls the variable that will influence the primary variable. The primary and secondary loops are sometimes referred to as the “master” and “slave” loops.

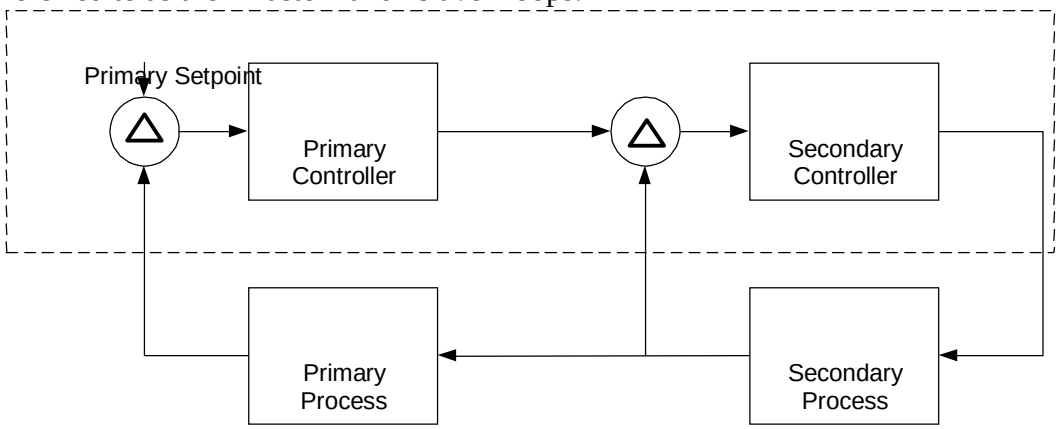


Figure 47 - Cascade block diagram

8.3 GUIDELINE FOR USE OF CASCADE

Cascade should be used when the following conditions are met:

- A variable that has a great influence over the primary variable exists,
- This variable has a dynamic time at least four times faster than the primary loops dynamics,

- The variable can easily be controlled.

Otherwise, cascade probably will not work well.

Other factors that will influence the decision to use cascade are:

- If the secondary variable is measured for other reasons, such as data recording or operator information, there is little if any additional cost to cascade. If the secondary variable would not be measured except to allow cascade control, then the costs of installing the measurement will need to be justified.

- If there are constraints that are to be placed on the secondary variable, such as a limit of the steam flow in the heat exchanger example, it may be simpler to limit the primary controller's output than to use override control (see Chapter 10).

- If there are time in which the operator will need to control the secondary variable directly (such as during startup), the secondary loop will be needed, so, as long as the guidelines above are met, cascade may as well be implemented.

At one time there was a greater cost of cascade, and it caused more complexity for the operator. However, with modern DCS and other digital control systems the only cost of cascade is the cost of measuring the secondary variable, and the operational complexity can be reduced.

8.4 CASCADE IMPLEMENTATION ISSUES

8.4.1 Mode transfer

In most cases a cascade loop does not always operate in full cascade operation, just as a simple PID loop is not always in automatic.

In order to implement a cascade control scheme the secondary controller must have the capability of remote set point mode. This option was required when purchasing analog controllers but is usually a user selectable option on digital controllers. In DCS controls the user may have to select a PID function block or select the remote set point option on a more general PID function block.

Some manufactures provide, in the PID block, a single mode "switch" with the options of Manual, Auto, and Cascade (or remote set point). Other manufacturers provide two switches, Manual/Auto and Local set point/Remote set point. Either way, the operator will have the ability to place the control loop into *manual*, and manipulate the output to the valve directly; *auto*, with the ability to manipulate the set point of the secondary loop; and *cascade* (or remote set point) to allow the primary controller to operate in automatic and control the process.

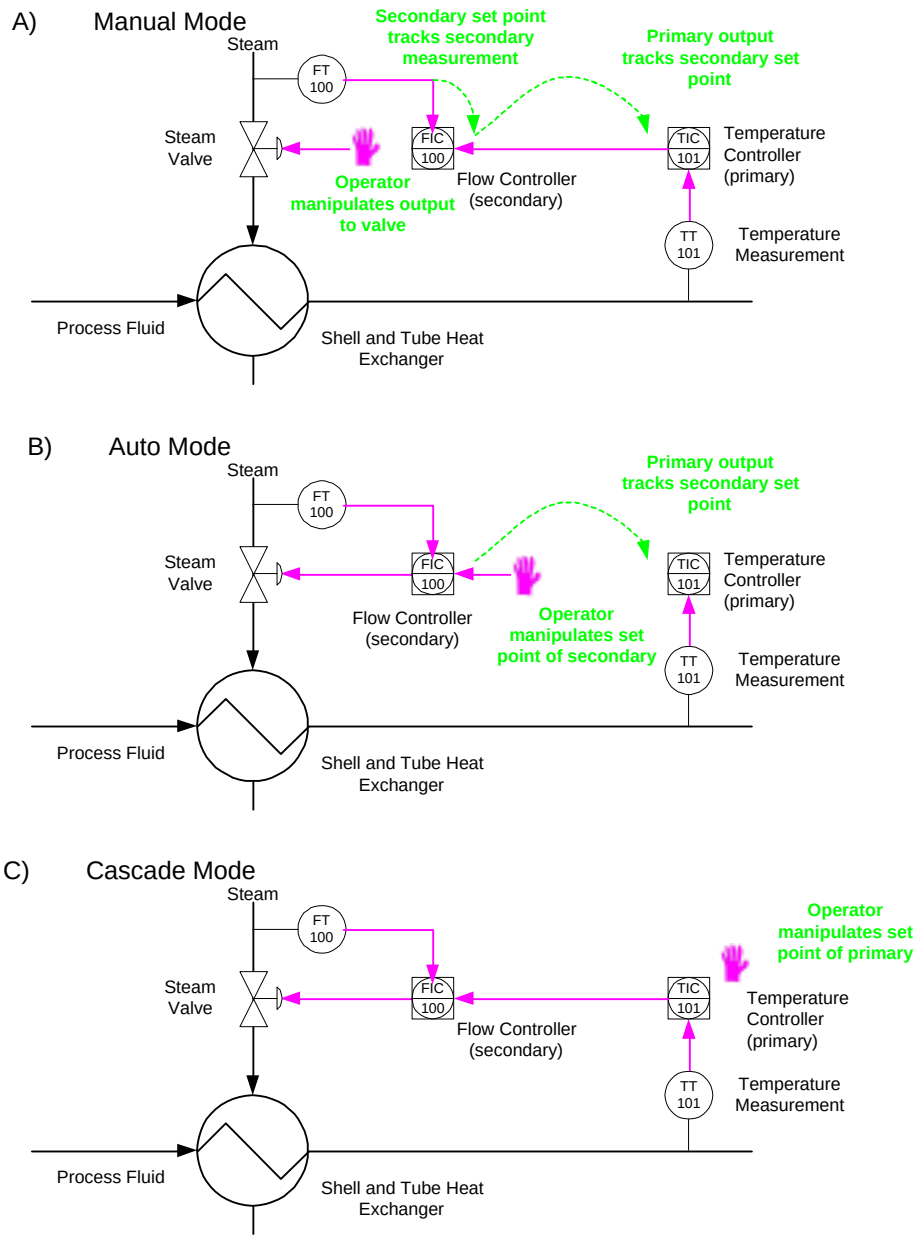


Figure 48 - The modes of a cascade loop.

The secondary controller can be in manual, with the secondary set point tracking the secondary measurement. The secondary controller may be in automatic, with the primary output tracking the secondary set point. Or the secondary controller may be in cascade mode with the primary controller in automatic.

In Figure 48 we see three possible modes of a cascade control scheme. In A, manual mode, the operator is adjusting the valve directly while monitoring the flow. In B, the operator has placed the flow loop (secondary) into automatic and is adjusting its set point. In C, the operator has placed the flow loop into cascade or remote set point mode and may adjust the set point of the temperature controller, if desired.

An important consideration for the implementation is that the transfer between one mode and another must be bumpless. That is, the act of changing the mode must have no immediate effect on the output to the valve. The most common

technique to deal with this need is tracking. In Figure 48 A, while the operator adjusts the output to the valve, the set point of the flow loop tracks the actual flow. At the same time the output of the temperature controller tracks the set point of the flow controller (with the flow value converted to percent). Therefore, when the operator switches the flow controller to automatic, the set point is the same as the actual flow, and no sudden change occurs to the valve.

In Figure 48 B, while the operator adjusts the set point of the flow controller the output of the temperature controller continues to track the flow set point. So when the operator switches the flow control to cascade (see Figure 48 C) the output of the temperature controller is already at the correct value for that time, and there is no immediate change in the flow set point or the output to the valve. After the switch to cascade, if the operator adjusts the temperature set point or there are load changes, the temperature controller will manipulate the flow controller to increase or decrease the steam flow as needed.

8.4.2 Windup

Windup is caused when the reset function of a controller causes the output of the controller to continue to change even though the change in output has no effect on the process. (see Section 4.1).

This affect cascade control when the output of the primary controller drives the set point of the secondary controller above the highest possible process measurement. The valve is opened fully, but the set point of the primary is not satisfied, so the output of the primary, and the set point of the secondary, continues to increase.

To use the example of the heat exchanger, suppose that the process liquid being heated enters the exchanger colder than normal. The temperature controller increases the steam flow set point, causing the steam valve fully open. Assume that the upper limit of the range of the steam flow set point is greater than the maximum steam flow with the valve fully open, and that with the valve fully open, there is not enough steam to heat the process fluid to its set point. The temperature controller will continue to ramp up its output until the output reaches a limit, typically 100%. At this point the temperature controller is “asking” the flow controller to provide more steam than is possible. The temperature controller is “wound up”.

Even if the temperature almost reaches its set point, this windup will eventually occur because the reset action of the controller will continue to ramp up the output until the temperature reaches or exceeds its set point.

At this point, the only problem that has occurred is the deviation in the set point. The windup is not yet an issue. However, suppose that the temperature of the process fluid flowing to the heat exchanger increases. The full flow of steam is now more than sufficient to heat the fluid to its set point, and the temperature will increase. Once the temperature crosses the set point the output will begin to decrease. However, it will have to decrease from 100% to the value (in percent) of the actual steam flow before the valve will begin to close. By the time this occurs the temperature may well have climbed to well above the set point. It is this “wind down” that is the problem with reset windup.

Reset windup in this case could be prevented by setting a limit on the output of the controller to the value that corresponds to the maximum steam flow. However, in some cases the maximum steam flow is not always the same. In the next section we will examine another, more general, method of preventing reset windup in cascade loops.

8.5 USE OF SECONDARY VARIABLE AS EXTERNAL FEEDBACK

One method of eliminating reset windup in the primary loop is to connect the secondary loop's process variable (converted from measurement units to percent) to the external feedback of the primary controller. (See section 4.2).

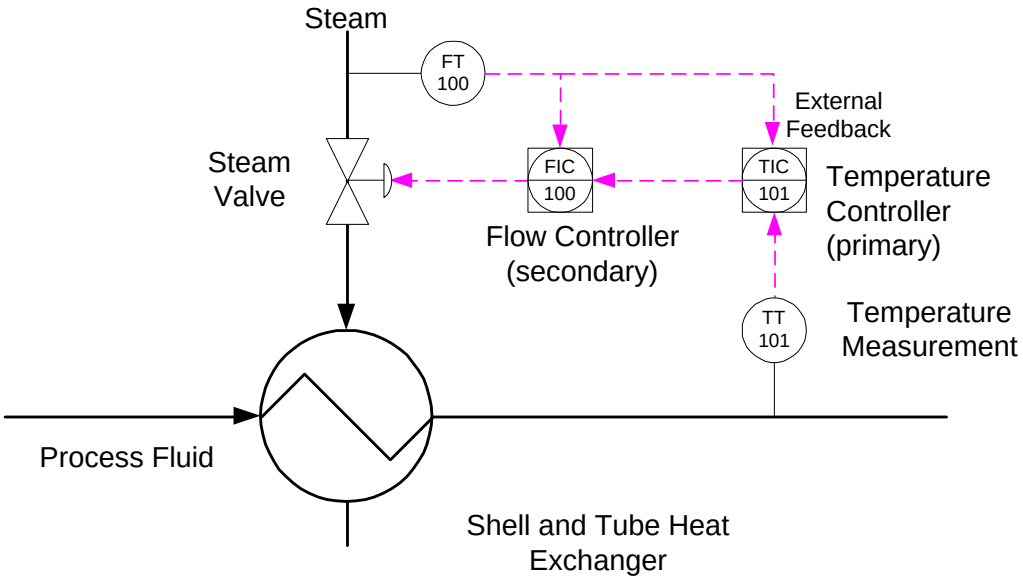


Figure 49 - External Feedback used for cascade control

Because the output of the primary controller is the error multiplied by the gain, the primary controller will be "asking" the secondary controller for more steam than is actually flowing if the temperature is too low, but will "ask" for less steam one the temperature rises above its set point. The secondary controller will respond by beginning to close the valve at that point.

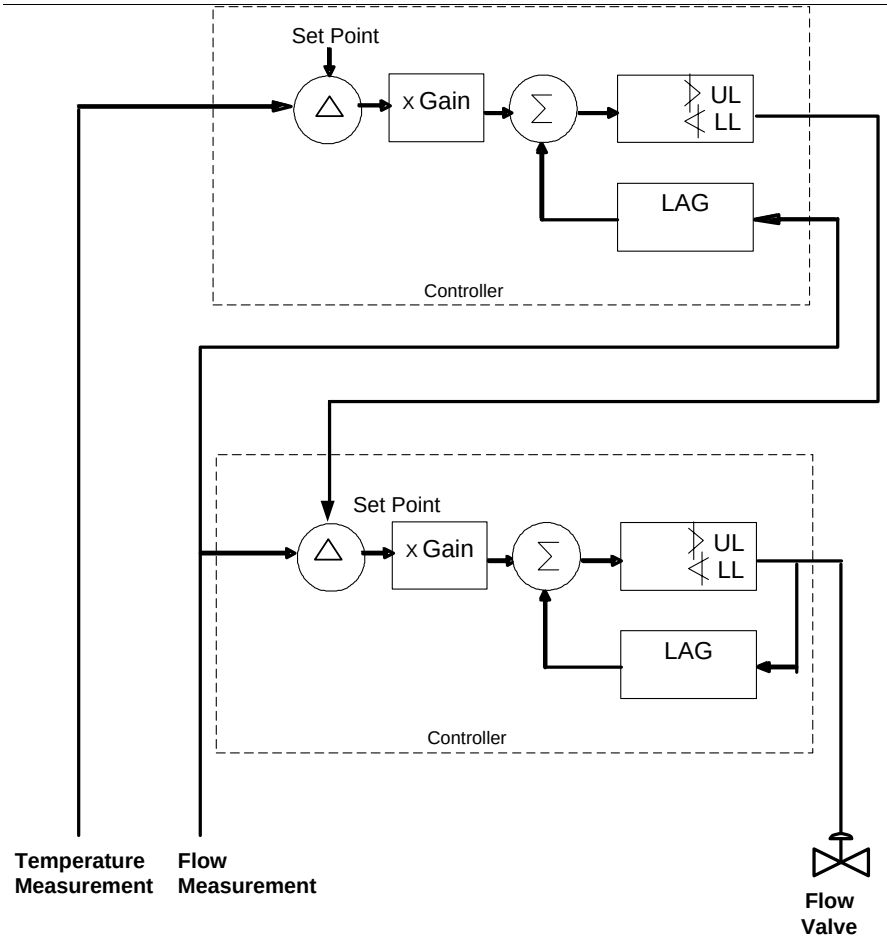


Figure 50 – Block Diagram of External Feedback for Cascade Loop

8.6 TUNING CASCADE LOOPS

The secondary loop is tuned first, using the tuning methods described in Chapter 6 When the loops is tuned, careful attention should be paid to the response to set point changes. The primary loop is left in manual while the secondary loop is tuned.

After the secondary loop is tuned, the primary loop is tuned with the secondary loop in automatic.

CHAPTER 9 RATIO

Ratio is another common form of multiple loop control. The object of ratio control is to maintain two or more flows at a constant ratio even when the flows are changing. Most often, the flows being control are blended, that is, mixed together. However, there are some situations where flows not mixed together are controlled by ratio control. For example, we may wish to maintain a flow of steam to the reboiler of a distillation column in ratio with the feed to the column even though the two never mix. For most examples and discussion here we will use the flow mixing example.

9.1 BASICS

Ratio control usually involves the control of one flow (known as the "controlled flow") in ratio to another flow (known as the "wild flow"). The wild flow may be controlled; if so it is controlled by an unrelated loop.

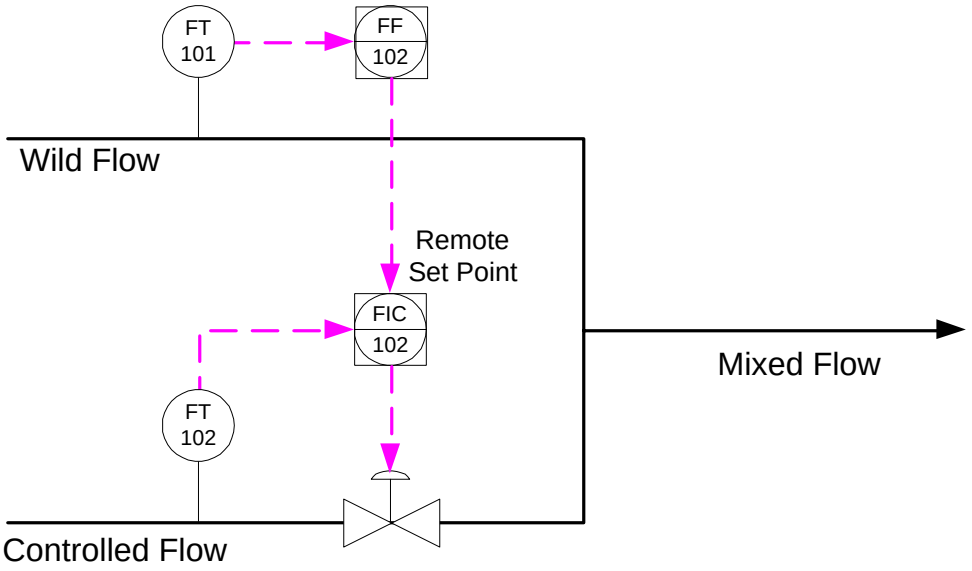


Figure 51 - Simple Ratio Loop

In Figure 51 the wild flow is measured by FT-101. The controlled flow is measured by FT-102 and controlled by controller FIC-102 which as a remote set point (similar to the secondary loop of a cascade control pair). The signal from FT-101 is multiplied (in block FF-102) by the desired ratio, which can be adjusted by the operator.

In actual implementation on most control systems, the remote set point to the controlled flow controller (FIC-102) is expecting a percentage signal (100% will result in a set point at the maximum range of the controller). Therefore the ratio unit (FF-102) will multiply the actual flow by the ratio and then convert the signal to a percentage based on the range of FIC-120. In some cases, the controller may be able to accept a remote set point in engineering units, requiring no

normalization. In most cases the operator will be able to adjust the ratio in FF-102.

9.2 MODE CHANGE

Just as in the case of cascade loops, the operator will have a choice of operating the controlled flow controller in manual, adjusting the valve position directly, placing the flow controller in automatic and adjusting the set point, or placing the controlled flow controller into remote set point (cascade) mode and allowing the ratio control to take place. This mode change will likely take place during the startup of the process.

In order to prevent a “bump”, or sudden change in the set point of the flow controller when the operator switches from local set point to remote set point (ratio control) the ratio is often recomputed, either continuously while the controller is in local set point or when the switch to ratio control is made, so that the ratio at the time of the switch is the actual ratio. After the switch to ratio control the operator can then adjust the ratio.

9.3 RATIO MANIPULATED BY ANOTHER CONTROL LOOP

In blending applications the purpose of the ratio control is to achieve some desired mix of the two flows, usually to result in a particular physical property, such as density. In order to ensure the correct ratio is used, the physical property may be measured using an analyzer. The analyzer signal can then be used as the input of a PID controller that adjusts the ratio. In this case, the ratio is a secondary loop and the analyzer controller is the primary loop in a cascade pair of loops. The output of the analyzer loop is converted from 0 to 100% to the number to be multiplied by the wild flow signal to provide the controlled flow set point. (see Figure 52). All of the features of cascade loops described in Chapter 8 apply to this control scheme.

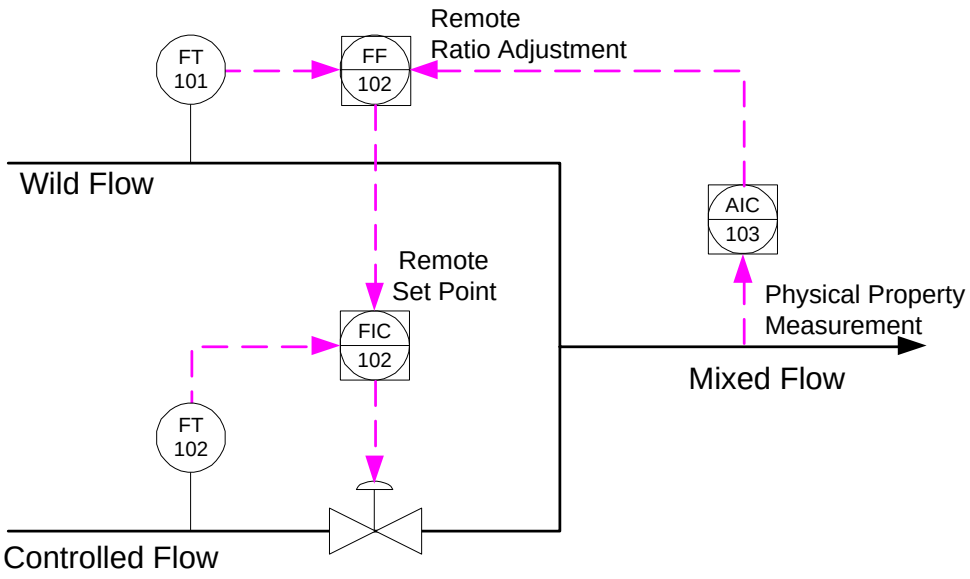


Figure 52 - PID loop manipulates ratio.

9.4 COMBUSTION AIR/FUEL RATIO

A special case of ratio control is the air/fuel ratio in the combustion controls for a boiler or furnace.

One type of combustion control is known as parallel control, where there is a control loop for the air and another for the fuel. The set points of both are set by the output of another controller, typically the steam pressure controller (often known as the “boiler master”). See Figure 53.

The fuel flow is typically measured in gallons per minute for fuel oil or standard cubic feet per minute for gas. The air is not usually measured and indicated in any particular engineering units. The measurement range for the air is the range that corresponds to the range for the fuel (for example, if the fuel flow is at 75% of its range, the corresponding air flow will also be 75% of its range).

Normally a slight excess of air is supplied to the boiler. As the demand for heat (the firing rate) increases and decreases, the boiler master will increase and decrease the set points of the air and fuel flows simultaneously. To adjust the ratio of air to fuel, the typical scheme places a multiplier in the air loop between the input from the air flow signal and the air flow controller. To increase the air/fuel ratio, the value of the multiplier is decreased, causing the flow controller to “see” a reduced air flow and open the damper.

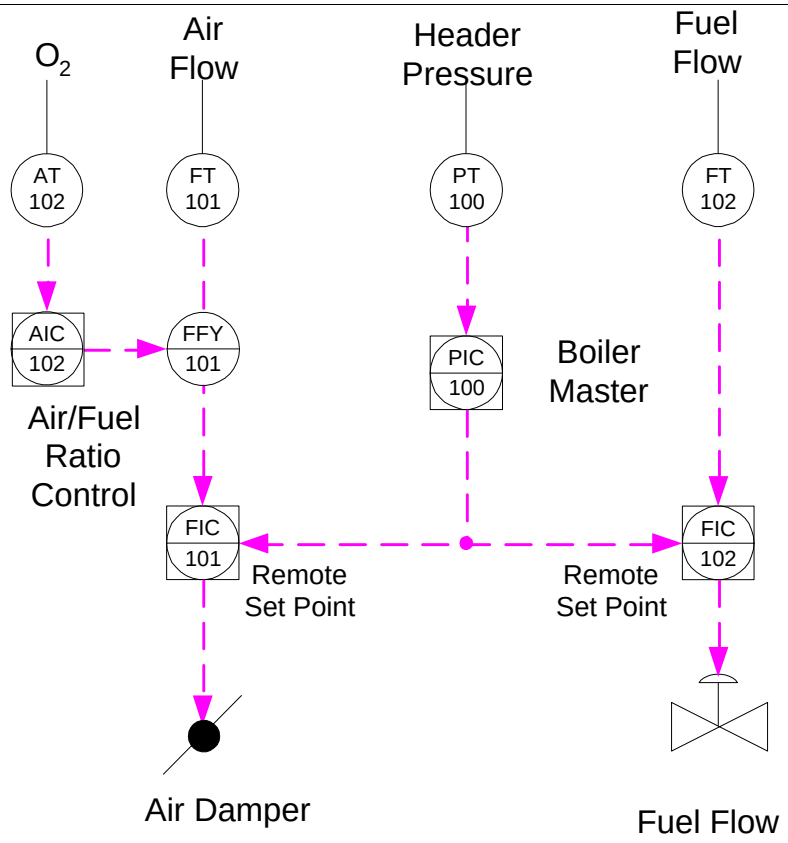


Figure 53 - Air and Fuel Controls

This control scheme manipulates the set points of the air and fuel flow in parallel in order to maintain the correct air/fuel ratio. The ratio is adjusted by a multiplier in the air flow measurement.

As shown in Figure 53, the air/fuel ratio can be a multiplier adjusted by the operator or it can be a cascade with the remote signal from the output of an O_2 controller. This controller will measure the O_2 in the stack gas and manipulate the ratio to keep the stack O_2 at the correct value.

CHAPTER 10 OVERRIDE

In most cases the manipulation of a valve will be done by a control loop to hold one particular process variable at its set point. However, there are situations where there are other factors that may be more important than keeping the particular control valve at its set point.

For example, we may have a control loop that admits the correct amount of steam into a heat exchange to produce hot water at the correct temperature. However, we might then decide that we must prevent the pressure in the steam header from falling below a given value even if doing so prevents the water from being heated to its set point.

We might also have a flow loop that draws a desired amount of water from a tank. But we decide that if the tank level is too low, we will cut back on the water we draw, even below the set point for water flow, to protect the pump from low level.

These are examples of *constraint*, or *override*, control. We can use override control in any situation where we have a control loop to maintain a controlled variable at its set point by adjusting a manipulated variable, however, there is a more important consideration that may limit the manipulated variable or need to take control of it.

10.1 EXAMPLE OF OVERRIDE CONTROL

In the example shown in Figure 54 a control loop (TIC-101) normally manipulates the steam flow to a heat exchanger to keep the liquid flow from the exchanger at its set point. However, if, due to lack of capacity of the boiler or demand on the exchange being too great, the steam header pressure falls below a limit we will close the steam valve to keep the steam header at its limit.

To implement this control we add a second controller (PIC-102) and a low signal selector (a device or software function block with two inputs. The output is equal to the lowest of the two inputs), shown in the diagram as a circle with a “less than” symbol (<).

The set point of PIC-200 is the low limit for the steam header pressure. If the header pressure is above the set point the output of the controller will remain high, not affecting TIC-101 and its manipulation of the valve. However, if the header pressure drops below the set point of PIC-200, the output of the pressure controller will decrease below the output of TIC-101, taking over the control of the valve.

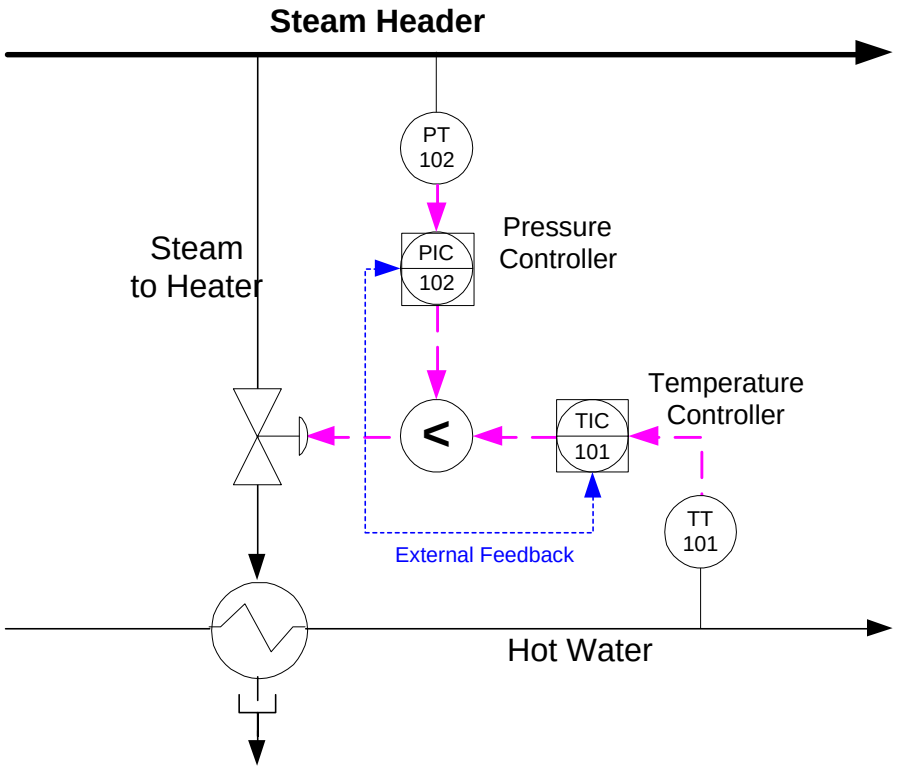


Figure 54 - Override Loop

10.2 RESET WINDUP

In our example when the pressure controller takes control of the steam valve away from the temperature controller the temperature will fall below its set point. The temperature controller will respond by increasing its set point in an attempt to open the valve. However, because the pressure controller and low selector prevents the temperature controller from opening the valve, the temperature controller will be open loop. In its vain attempt to open the steam valve the reset of the temperature controller will drive its output up to 100% (or other limit). This is known as reset windup.

If, in our example in the previous section, the temperature controller does not have control of the steam valve the temperature falls below its set point, the integral action in the controller will cause the output to continue to increase until it reaches 100%. This is known as reset windup.

The problem with reset windup is that after the header pressure reaches its set point and the pressure controller output increases, the valve will continue to open. The temperature controller will not begin to close the valve until the temperature rises above its set point. Then, as the temperature rises above its set point the controller will lower the output. This "reset winddown" will cause an overshoot in the temperature set point.

10.2.1 One solution: External Feedback.

If the PID algorithm uses a positive feedback loop for integration, then that positive feedback loop can use the output of the selector (the signal to the valve) as the feedback.

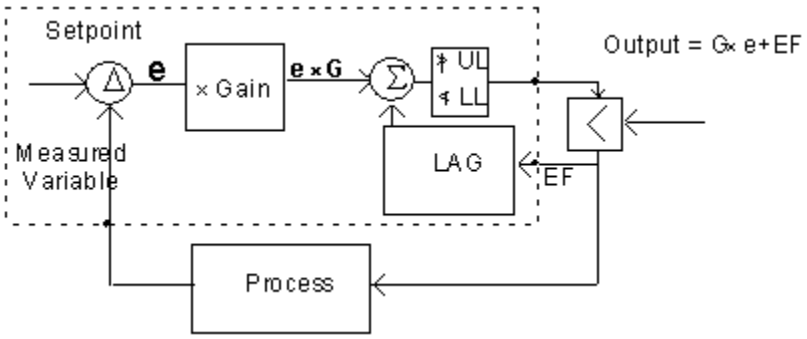


Figure 55 - External Feedback and Override Control

When the external signal to the low selector (<) has control of the valve, the controller output is equal to the error multiplied by the gain. As the error decreases, the controller output approaches the other signal. As the error changes sign, the controller output takes over control and begins to close the valve.

The external feedback (EF on the diagram) is the output of the low selector. The output of the temperature controller is the gain multiplied by the error added to the external feedback. If the error is positive (temperature below set point) the output of the temperature controller is higher than the output of the selector. However, when the temperature rises and equals the set point, the error will be zero and the controller output will equal the selector output.

As soon as the temperature exceeds the set point (negative error) the output of the controller will be less than the output of the selector, decreasing the output of the selector and closing the valve.

10.3 COMBUSTION CROSS LIMITING

A special case of override control is the “cross limiting” feature shown in Figure 56. (This is normally part of the combustion control shown in Figure 53 but was left off for simplicity)

If the measured fuel flow exceeds the firing rate demand (output from the pressure control) the high selector will pass the measured fuel flow (in percent) rather than the firing rate to the set point of the air flow control. If the measured air flow is below the firing rate demand signal the low selector will pass the air flow signal rather than the firing rate demand to the set point of the fuel flow. This will assure that the fuel flow will not exceed the air flow even if either flow is slow to respond to its controller.

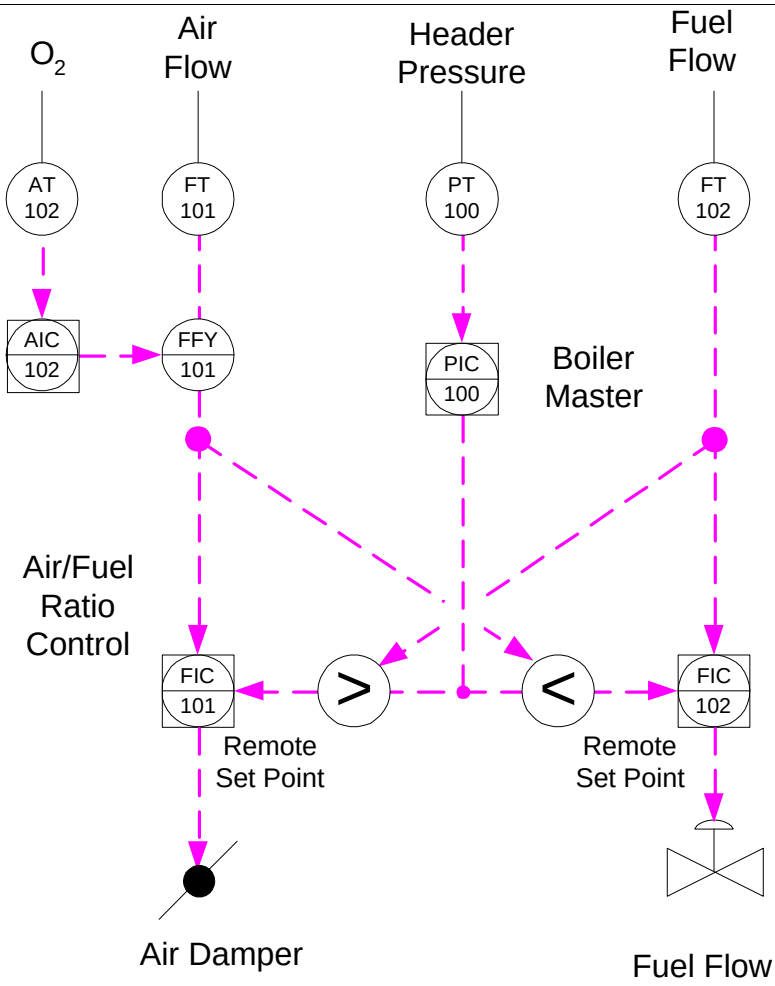


Figure 56 - Combustion Cross Limiting

The high (>) and low (<) selectors prevent the air set point from falling below the measured fuel flow, and prevent the fuel set point from rising above the measured air flow.

CHAPTER 11 FEEDFORWARD

In *feedback* control we measure the controlled variable, compare it to a set point, and then take corrective action after the controlled variable has departed from its set point, usually as a result of a disturbance (change in a load variable).

If instead we can measure the loads and change the manipulated variable to anticipate the effect of the load change we might keep the controlled variable from departing from the set point (or, at least, keep the error short). This strategy is called feedforward control.

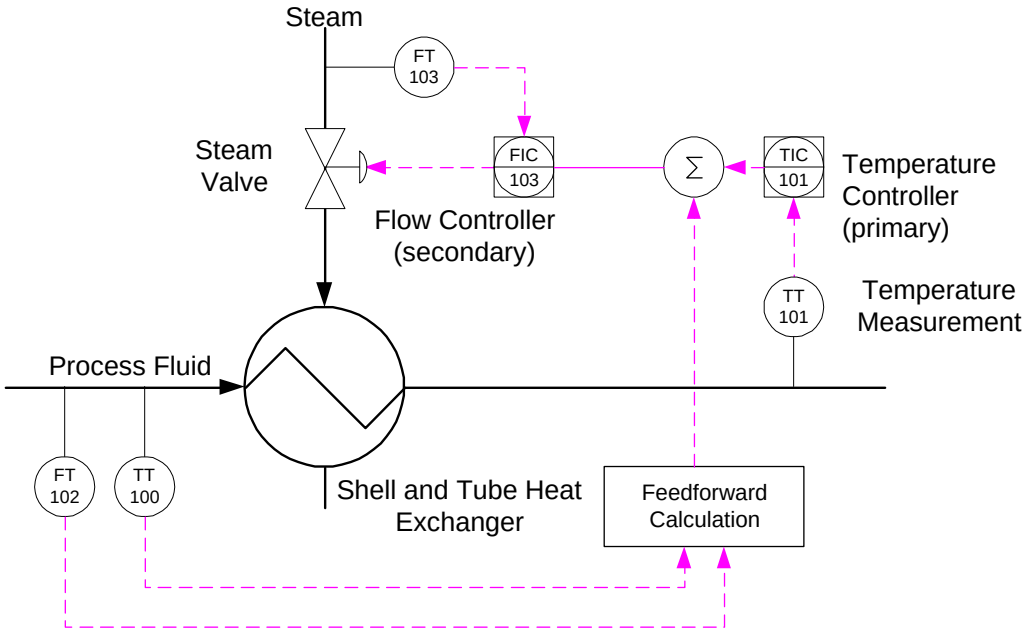


Figure 57 - Feedforward Control of Heat Exchanger

The feed flow and feed temperature are used to compute the steam flow set point, with the temperature controller used as a trim.

In the example shown in Figure 57 the feed forward calculation block calculates the amount of steam needed to heat the process fluid to the set point based on the flow and inlet temperature. This calculation is known as the “feed forward calculation”.

Because there are usually disturbances that cannot be measured and the feed forward calculation is not perfect, the calculation will not result in the exact amount of steam flow. Therefore the output of the feedback controller is added to the feed forward calculation to continuously trip the steam flow set point to keep the temperature at its set point.

CHAPTER 12 BIBLIOGRAPHY

1. J. G. Ziegler, N. B. Nichols, Optimum Settings for Automatic Controllers, ASME Transactions, Vol. 64, 1942, pp. 759-768.

2. A. B. Corripio, *Tuning of Industrial Control Systems*, Instrument Society of America, 1990.

3. C. L. Smith, *Digital Computer Process Control*, International Textbook Company, 1972.

4. J. A. Miller, A. M. Lopez, C.L. Smith, P. W. Murrill, A *Comparison of Controller Tuning Techniques*, Control Engineering, Dec. 1967, pp. 72-75.

5. A. M. Lopez, P. W. Murrill, C. L. Smith, "Controller Tuning Relationships Based on Integral Performance Criteria," *Instrumentation Technology*, V. 14 [Nov. 1967], p. 57.

6. H. Wade, Course notes for *Principles of Applied Automatic Control* (an ISA short course), Instrument Society of America, 1992.

7. J. Shaw, *Analysis of Traditional PID Tuning Methods*, presented at the Instrument Society of America conference, Chicago, IL, September, 1993.