Chapter V

RE: Feedback Loops - Definition - Selection - Interface

When designing motion control systems, the feedback device has got to be the simplest, yet most confusing part of the system. The ability of the feedback device to convert physical action into an electrical signal is the basic reason for using it. However, the selection of the proper kind of feedback device based on its measurement range, stability, linearity, % span error, etc., your resolution requirement, environment, and countless other factors, make this simple device a designers nightmare.

There are five (5) types of feedback loops which will be discussed. The feedback loop(s) can be used individually, or in combination as the designer sees fit. The selection of the feedback loop is based on system performance requirements, overall machine specifications, cost, and qualified tradeoffs (gains in one area offsetting reductions in another).

The definitions of the five feedback methods described, will imply that the system computer is controlling the motion based on the information received back from the feedback device, unless otherwise stated.

As the feedback methods are defined, you'll see that each has their respective place in "helping" to maintain the system motion. Depending upon what the system has to do, the selection of the proper type of feedback method can save hours of mishaps (more commonly referred to as Murphy's Law).

I) Feedback Loop Definitions

When using any of the feedback methods, it's important to know what kind of control your dealing with. A feedback loop implies that the feedback signal is being responded to by a "smart" device. This does not mean to imply that the device is a computer. Only that the device is smart enough to take in the feedback signal, and correct for the errors generated between the given command, and the actual mechanical result.

1) Voltage feedback loop (VFL)

The voltage feedback loop utilizes a DC or AC voltage level generating device to give the controller information about the motion (position, velocity, and/or force). The VFL accuracy, and repeatability will never be better than the ability of the control to analyze, and correct for errors, the mechanical capability of the system, and the general errors of the VFL device itself (analog tach, resolver, etc.).

2) Current feedback loop (CFL)

The current feedback loop utilizes a DC or AC current level to give the control information about systems loading, position, or velocity. Current feedback implies only that the feedback signal is a current signal.

When dealing with motor/amplifier packages, if no other feedback loop (such as an encoder) is used for "outside loop" control, then the motor would be considered operating in a "torque mode".

If, however, the system does utilize an outside feedback loop (such as an encoder) for velocity, or position feedback, then the motor would be operating in a "current mode".

The CFL accuracy, and repeatability will never be better than the ability of the control to analyze, and correct for errors, the mechanical capability of the system, and the general errors of the CFL device itself.

3) Pulse feedback loop (PFL)

The Pulse feedback loop utilizes a pulse producing mechanism to give the controlling device information about the motion (usually just velocity, unless a direction signal is implemented).

The PFL accuracy, and repeatability will never be better than the ability of the control to analyze, and correct for errors, the mechanical capability of the system, and the general errors of the PFL device itself.

4) Digital feedback loop (DFL)

Although the PFL is digital in form, I define the term "digital" as a signal device which passes multiple bit information in parallel to the controller (such as an absolute position encoder, or resolver). The term digital, therefore, applies to multiple bit signaling and handling devices.

The DFL device consists of special coding structures which yield multiple bit position information on power-up (without Home referencing). Within one rotation (cycle) of the DFL device, the position of the motion is known, without the use of CPU counting registers. The DFL device can be thought of as "smarter" than the PFL or encoder device, and is generally used in systems that cannot be moved to a home position on power-up (synchronized, interlocked, or interlaced process lines).

Inaccuracies in the DFL system are the same as in the PFL system, but this approach may also require longer update periods due to the requirement of decoding the multiple bit signal information packets.

5) Open, or No feedback loop (OFL)

Generally reserved in thought for Stepper motor systems (refer to Chapter 11), the requirement of feedback to the computer control is not necessary, as in pure velocity, or torque systems.

When properly calibrated (and with repeatable linearity), a motion system can produce known velocity and torques without the assistance of a feedback device (such as a speed pot. on a motor drive amplifier).

In a stepper control system, the accuracy will usually not be better than 3 to 5% of the stepper motors full step angular value, the round off produced by computer math, and the mechanical capability of the system itself (refer to Chapter XI).

II) Feedback Terminology

The purpose of this section is to discuss how feedback terms relate to the devices receiving the feedback information, as I will use them throughout this handbook.

Perhaps the most often repeated mistake in the field of motion control, is to use the same term when talking about different devices which, in fact, do different things.

Lets begin with an example to show how easy it is to confuse the discussion. Suppose you needed a quadrature encoder for an application, and needed it to supply 2000 "lines per revolution".

How would you refer to the encoder?

If you're an encoder designer, you probably define the term "line" as the etching placed on the coding disk which develops the "pulses" to be supplied to the external world. If a customer ordered a 2000 "line" encoder and you sent him your interpretation, you might wind up with a phone call for a replacement unit.

If you're a control designer, you probably define "line" as the vertical "leading", or "trailing" edge of a pulse. Knowing that a quadrature counter is setup to count all of the "pulse" edges (X4 mode), and that the A and B encoder channels each have two edges, you would possibly refer to the encoder as a 500 "pulse" device (2000 / 4). Chances are, you'll get the correct device.

If you're a software writer, you might define "line" as a "count" the encoder has supplied you for position information. You might refer to the encoder as a 2000 "count" device. It's more than likely that you'll be asked to clarify what it is your trying to order.

Whose right? ...
All of the above! ...
Then what's the problem?

The problem is, these people are not "communicating" on a common language basis. Each person can interpret what he hears wrong, because the common-ality that allows talking "apples-with-apples" is missing.

Its important to define the terms up-front, otherwise you might wind up with the wrong "count" device, and a schedule delay.

The following terms are defined as they are used in this report, and in reference to the motion controller.

1) Pulse

A momentary electrical signal which varies between two voltage, or current levels (see Figure 5.1). The repetition (frequency) of the pulse signal generally indicates the speed at which some system parameter is changing (position, velocity, force, etc.).

2) Line

The rising (leading) and falling (trailing) edges of a pulse are considered "lines". Each pulse, therefore, contains two (2) "lines".

3) Count

A summation of sequential electrical signals by either electrical or software means. The signals may be either pulses or lines.

4) Resolution

What each pulse, line, or count is worth as a dynamic dimension such as distance, velocity, or force. It may also be considered the "fineness" of measurement without approximation.

5) Quadrature

The utilizing of two pulses, generated at a 90 Degree phase shift with respect to each other, which signal position (number of counts), velocity (number of counts/second), and direction (Fwd. = Phase A leads Phase B). The phase shift allows the controlling device to determine direction (CW or CCW, Fwd. or Rev., etc.), as well as position, and velocity (see Figure 5.1).

6) Complementary signal

A complementary signal, is an inverted, mirrored, or in the

case of repetitive waveforms (square-wave, sine-wave, etc.) 180 degree phase shifted, replication of the main signal.

7) Differential

By using "complementary" signal pairs to pass information from one device to another, most common mode "noise" injected into signal wires can be eliminated, since injected noise is random (not a pure inverted signal match). At the differential summation input, the noise signals will cancel out leaving only the true signal data (as long as proper wiring techniques have been observed).

8) Single-Ended

A single ended signal is one which has no complement. It is referenced to power supply (signal) common only, and can be quite susceptibility to noisy environments.

9) Pulse and Direction

A Pulse and Direction feedback device utilizes a voltage or current pulse signal to inform the control of the position, or velocity of the system. A second signal line is used to inform the control of the direction the system is traveling. The direction signal will maintain one of two voltage or current levels to indicate the forward or reverse direction. This device is well suited for "real time" counting applications (see Figure 5.1).

10) Count-up Count-Dn

Count up/Count down signal devices use voltage or current levels to indicate position, or velocity. This device uses two signal wires to transmit the signals to the controller. When moving in one direction, the signal is transmitted on one wire, and is transmitted on the other wire device moves in the opposite direction. The individual signals are most often connected to some form of up/down counter circuit for position totalizing (see Figure 5.1).

11) Square Wave

A repetitive pulse which has an equal "on" and "off" duty cycle (time period).

12) Open Collector

The term "open collector" refers to a transistor type output, which can be used to provide power (source), or common (sink) to a separate device. This type of output can be used in a variety of configurations, the most common being the opto-isolator. Using resistors as "pull-up" or "pull-down" mechanisms, the signal cable impedance can be "matched", to enhancing signal transfer frequency.

13) TTL

TTL is a term used for an signal output capability of 5 vdc. at about 1 ma.

14) Line Driver

Line Driver refers to an output which can supply a signal (differential, or single ended) at a given voltage level, and 100 milliamp current capability. The device is generally of "totem pole" construction, and will, therefore, act as both a sink and source to the receiving device.

15) Incremental

A dimension, or measurement relative to the current position.

16) Absolute

A dimension, or measurement relative to a master coordinate system.

17) Tachometer

A signal device which produces a voltage or current level as the speed of the device changes. Not to be confused with frequency devices, the true tachometer, will produce **immediate** velocity information with any motion implied to the tachometer device.

18) Analog

A signal which gives data as an infinitely variable voltage or current "level". The signal is generally converted to digital information, using Analog-to-Digital converters, prior to being used by the controller. In many cases, the signal is used in its original analog form (such as with tachometer feedback to motor amplifiers).

19) Digital

A signal (or multiple signals) which is not infinite in range. It is usually restricted to two voltage or current levels, such as 5 vdc., and ground (TTL).

20) Sink

A device termed "sink", will supply a signal ground to the receiving device.

21) Source

A device termed "source" will supply a voltage or current to the receiving device with respect to the power supply (signal)

ground.

22) DAC

Digital-to-Analog-Converter. This device is responsible for the conversion of digital computer signals into Analog signals (see #18 above).

****

When specifying a feedback device, be sure to use the terminology of the place you are ordering from, or you'll probably get a phone call asking for part clarification.

****

III) Feedback Resolution and Selection

The first two questions which must be answered when selecting a feedback loop are ...

1) What is the system measuring?
2) How accurate does it have to be?

The first question focuses on selecting the type of feedback device. The choice of feedback device is dependent upon what is being measured.

The second question, requires you to determine the resolution (or fineness of measurement) of the feedback device.

If the axis will have an encoder feedback device, but no tachometer, tighten the resolution by a factor of at least 5. Most designers forget that a tach-less motor drive system (one which is using an encoder to provide the velocity as well as the position feedback) will need to move in order to resolve the velocity part of the algorithm. Therefore, when the unit is at "standstill" (ie. - not commanded to move), any offset in the zero balance of the axis due to gravity, motor amplifier adjustment, etc., will cause motion to occur. This motion can only be counteracted by forcing the DAC to drive the axis in the opposite direction.

The hovering of the axis at standstill can and will absorb several position (resolution) counts. In view of this, its good practice to tighten up the resolution to insure the system does not need to move outside the tolerance boundary when controlling.

A general rule, is to set the actual feedback resolution to be a factor of 5 to 10 times the resolution required to ensure that all system errors remain well within the tolerance of the system specification.

IV) Feedback Interface Considerations

Interfacing two devices together, implies that the designer will take the time to ensure all electrical items (such as voltage, current, frequency, noise, shielding, etc.) have been taken into consideration. This is necessary if the two devices are not only to work together electrically, but to ensure the devices to pass the signal information without loss of signal integrity.

For the purpose of the interface discussion, assume that an optical encoder has been selected to provide the feedback. Although a conceptually simple device, the optical encoder does require a careful examination of the various types of encoder output signals available, and how to interface them to a digital system.

Optical encoders fall into two basic categories ...

1) Absolute and,
2) Incremental.

Output from an absolute encoder represents the absolute position, within one rotation, of the encoder rotor. Each bit of resolution in an absolute encoder requires an additional code ring on the encoder disk. The primary advantage of an absolute encoder is in its ability to provide absolute position data within the encoder resolution at power up, hence its name. The absolute encoder's primary disadvantage, is the increased size of the code disk and the related increase in cost as resolution increases.

Incremental encoders fall into several subcategories (see Figure 5.1) ...

1) Tachometer,
2) Pulse and Direction,
3) Up/Down and,
2) Quadrature.

To maintain continuity of discussion, the remainder of this handbook will center around the quadrature incremental encoder, generally referred to a quadrature encoder.

The quadrature encoder has at least two output signals most commonly referred to as Channel A and Channel B. As shown in Figure 1, the Channel B quadrature signal is offset 90 degrees from the Channel A quadrature signal.

One major benefit of quadrature output signals is the ability to sense the direction of rotation from the sequence of phase generation. For the purpose of this discussion, the convention of clockwise (CW) rotation represented by Channel A leading Channel B and counter clockwise (CCW) represented by Channel B leading Channel A will be used. Using the one count per encoder edge, if Channel A rises before Channel B, a CW count is generated. If Channel A falls before Channel B, a CCW count is generated.

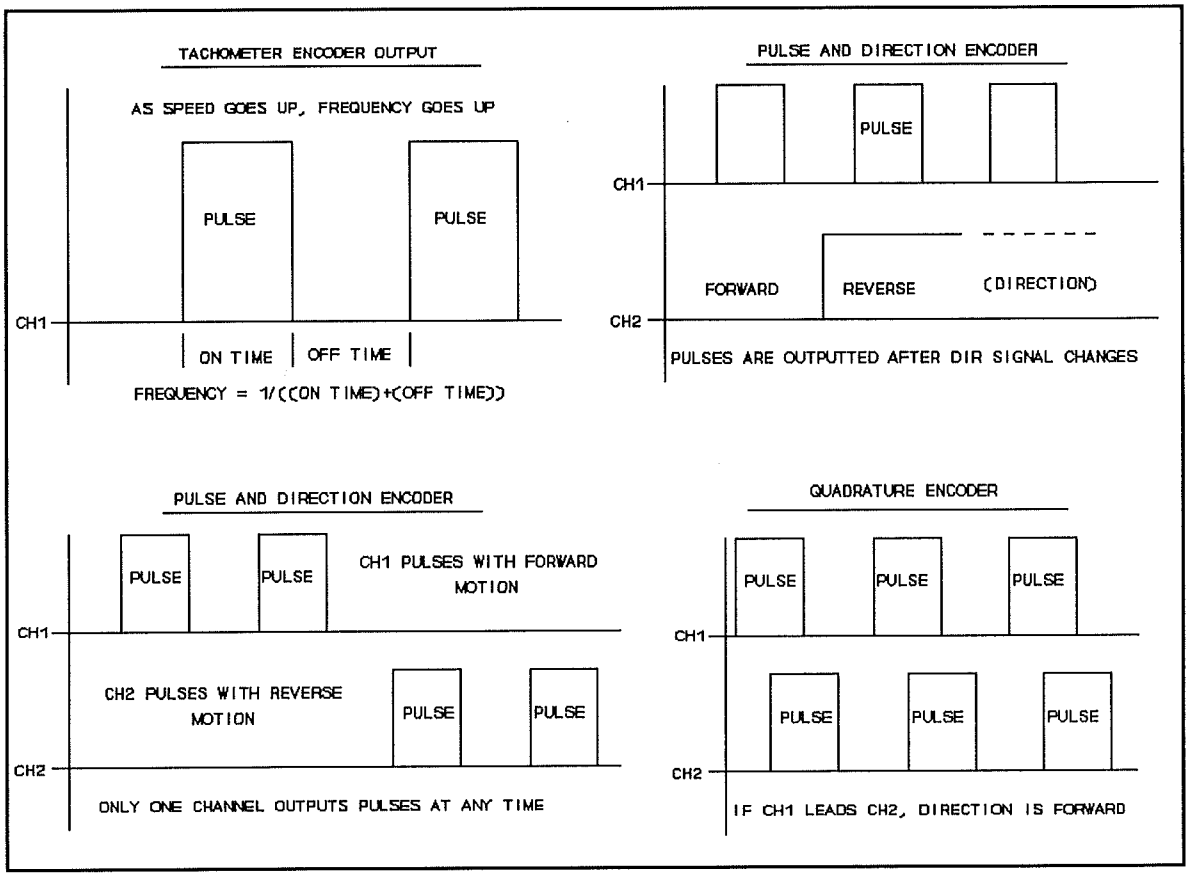

Figure 5.1

This ability to determine direction is critical if encoder rotation stops on a pulse edge. Any vibration at this point will cause the pulse edge output to toggle up and down. Without the ability to decode direction, the counter would count each transition through the rising edge of the signal and loose position. By decoding the quadrature signal to determine direction, the counter would count up on the rising edge of Channel A, and down on the falling edge of Channel A maintaining an accurate position count.

Another benefit of quadrature output is the ability to electronically multiply the counts generated during one encoder cycle (Figure 2). In the times 1 (X1) mode, up counts are generated on the rising edges of Channel A, and all down counts are generated on the falling edges of Channel A. In the times 2 (X2) mode, both the rising and falling edges of Channel A are used to generate counts resulting in a doubling of the resolution, although still leaving both edges of Channel B unused, except for direction determination. In the times 4 (X4) mode, the rising and falling edges of Channel A and Channel B are used to generate counts. This effectively increases the resolution capability by a factor of four.

It should be noted that the actual increased resolution is

only as accurate as the quadrature phasing between Channel A and Channel B. Any variance from quadrature in the Channel A and B phase relationship results in count-to-count positional errors.

Another feature available as an option on many quadrature encoders, is the index pulse. An index pulse is shown in Figures 1 and 2 along with the Channel A and B wave forms. An index pulse occurs only once per encoder revolution, and is used to establish an absolute mechanical reference position within one 360 degree encoder rotation.

A disadvantage of incremental encoders is that they are relative in nature. In other words, on power up, a system employing a quadrature encoder has no way of determining the current position within a 360 degree rotation of the encoder. The index pulse only, can provide a signal that which, when located, will give an absolute mechanical encoder shaft orientation.

The index signal can be used to perform a number of tasks in the system, such as resetting or presetting the position counter and/or generating an interrupt signal to the system CPU. While some manufactures gate the index signal internally, often the index signal is wider than 1/4 encoder cycle and must be gated with the Channel A and Channel B signals to produce an index signal that does not overlap either a Channel A or B edge. This helps to insure that for whatever purpose the index pulse is used, it occupies no more than one count.

Square wave quadrature encoders generally have two types of output signal drivers ....

1) Single-ended, and
2) Differential.

Single-ended encoders have one output signal for Channels A, and B referenced to a common connection. Single-ended outputs are generally either open-collector or totem-pole (TTL or CMOS). An open-collector output, the simplest of the three, requires only one transistor for each Channel. Some manufactures include a pull up resistor from the Channel output to the plus (+) voltage which eliminates its being categorized as an open collector (which is generally used to indicate some form of power source isolation). True open collector outputs do not provide pull up resistors. The customer interface does, therefore, consult the encoder spec. sheet to determine the resistor to use. Open collector output encoders should only be used over short distances in electrically quiet environments.

A quadrature encoder with totem-pole outputs employs either a TTL or CMOS buffer between the Channel A and B signals and the output of the encoder. This buffer is usually a Schmitt-trigger to provide sharp transitions in the quadrature output signals. Totem-pole output signals are directly compatible with their

relative logic families (TTL or CMOS) and can be interfaced to other logic families as long as good inter-family interface techniques are observed. While TTL encoders do provide better drive capabilities than open-collector types, they are still limited to TTL voltage levels which limits the length of line they are capable of driving. Again, the manufacturer's spec sheets should be reviewed to determine current drive capability and other parameters. An advantage of the CMOS type encoder over the TTL encoder is a wider voltage range of the output signal. CMOS supply voltages can be as high as 15 volts, whereas TTL voltages are restricted to 5 +/-0.25 vdc.

The differential (complementary) output quadrature encoder is the second major encoder type. The Channel A, and B differential outputs consist of two complimentary output signals that are specifically referenced to each other. Differential output encoders using line driver buffers, allow the differential output encoder to reap the benefits of increased common-mode noise immunity and increased drive capability. These type encoders are preferable in electrically noisy environments, and when installed at long distances from the controller.

Encoders are most often thought of as a position feedback mechanism for motion control systems and are manufactured in many forms for that purpose, the most common being rotary or linear. The type chosen is determined by the design and the type of motion. Servo motor feedback is often accomplished by attaching a rotary encoder to the back of the motor. In a linear table application, however, there is an argument for obtaining the servo feedback from a linear scale (encoder) directly attached to the table.

Another type of quadrature encoder becoming increasingly popular is the panel mount encoder. This style encoder is being used in many modern designs to replace functions previously performed by potentiometer. On some manufacturer's oscilloscopes and logic analyzers, panel mount encoders are being used as multi-function controls to set parameters and make selections from on screen menus. In systems that are primarily digital, using a panel mount encoder eliminates the need for converting the potentiometer's analog output into digital form.

One popular application of the encoder is as a Digital-Read-Out device (DRO). An encoder interface, either designed into a system, or added as a board level product, simplifies the implementation of DRO features. Using a card or system level encoder interface board, eliminates the external box usually associated with DRO systems. An internal system usually has a faster response than systems requiring communicating links. An onboard encoder interface can also be used for other purposes such as closing the loop on a motion system as well as giving to the host computer information about position, velocity, force, or any other conceivable use of the encoder.

	Resistive Potentiometer	Digital Potentiometer
Output	Analog	TTL compatible - digital
Contact Wiper	Yes: and the wiper will become electrically "noisy" with time	No: the signals are optically coupled. There is no internal electrical noise
CW/CCW turns	Usually limited to one	Unlimited number of turns
Retention of position info.	Position information is saved if power to the system is lost	Position information is lost if power to the system is lost
Resolution Limitations	Limited by the analog to digital converter and resistive surface nonlin-earities.	Limited by No. of pulse per revolution or visual display as appropriate.
Typ. cyc life	50,000 to 100,000 cycles	> 1 million revolutions

The table shown is from Hewlett-Packard Application Note 1025.

Whatever the final design, an important consideration in any feedback system is system phasing. Boiled down to basics, phasing simply means the encoder feedback relates as expected to the encoder movement. In some cases this is a simple matter of reference. When the panel encoder turns clockwise, the position counter should count up, and vice-versa. The result of incorrect phasing can be disastrous. In servo systems, incorrect phasing can allow motor "run-away", with devastating results. In the case of incorrect encoder phasing, use one of the following steps:

1) Invert Channel A **or** Channel B, but **not both**.

2) Connect the Channel A connection(s) to the Channel B connection(s), and the Channel B connection(s) to the Channel A connection(s).

Typical applications for some of the Tech 80 Computer boards include high resolution feedback systems such as Laser interferometer feedback, laser encoders, and high resolution glass scales.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

Chapter VI

Phasing Motors to Control Systems - With and Without Velocity Loops

Doing Applications Engineering, I speak with hundreds of people each month. Not all problems, however, are related to applications. The most often asked questions are about proper wiring of motors velocity, and position loops.

The term "Phasing" is used to describe the relationship of the motor motion with the system command. Motor phasing is perhaps the biggest nightmare a newcomer to the field can have. When phasing multiple control loops, even the most experienced engineer sometimes has problems. This chapter is devoted to outlining simple phasing techniques. The intent of this chapter is to save the many man hours of frustration, and perhaps even unforeseen catastrophes that could evolve from phasing errors.

Motor Control Phasing Defined

First, let's define the term "Phasing". Simply stated, correct Phasing means ...

The motor Direction and Rpm are coordinated with what the velocity and/or position control loops are trying to do.

In other words ...

If the control system commands the motor to remain stationary, move in a direction at a given speed, etc., and it doesn't, the control loops are "Out of Phase" (not functioning properly).

Just think of all of the out-of-phase possibilities (hard wired) there could be, and all of the symptoms that would go along with them. There's too much at stake for you to be wrong. You could damage the motor, the drive, the machine, - or worse - yourself or someone around you. A secondary effect, would be to become discouraged with the manufacturer of a given device. All of this simply because you weren't given an opportunity to secure the basic principles, or techniques, of "phasing".

In certain systems, it is possible to test the phasing of the motor when it's disconnected from the machine. But this is not always the case, as with Linear Scale systems, or split systems with tach. and encoders on opposite sides of gearing. How about startup systems with a computer reading the position feedback from an encoder board, then transferring the results to an analog driver board.

Good practice dictates that the first time startup of any

motor control system should be direct and to the point, not a symptom reading, troubleshooting, or frustrating experience.

Every day phasing situations vary widely, but the good news is, by following several basic steps and adhering to the rules, your system can be properly phased in a matter of minutes.

Basic phasing rules

Rule #1 ...

Whenever possible, phase the motor control system with the motor shaft disconnected from the mechanical load. In the event of a "run away", no damage to the system can occur.

Rule #2 ...

Always have immediate access to a motor "Kill" switch, and have it manned. In the event of a "run away", the motor can be immediately powered down.

Rule #3 ...

Connect and Phase the Lowest level control loops first, then propagate outwardly connecting and phasing each successively higher level control loop in its order.

This rule is no different than solving a simple math problem, such as ((2x2) + (3x(2x5)) + (4x4)) = ?. The problem is solved simply, by solving the groups within the lower level parenthesis first, then accumulating the total answer ....

((4) + (3x(10)) + (16)) = 4 + 30 + 16 = 50.

Note, that I talk about "loops". It's really irrelevant whether there's one, two, or ten control loops involved with the control of the motor motion. If each loop is connected and phased at its proper time, the system phasing can be accomplished quickly.

Rule #4 ...

Always have the proper tools. This includes, at least, the device schematics or interface pinouts (available from all device manufacturers), a voltmeter and/or Oscilloscope, and the system wiring diagram.

Rule #5 ...

Don't be impatient! You'll get the job done a lot faster if you do it right the first time.

Motor Control Phasing

System Style #1 Single Position loop - no velocity loop

The worst thing people do when they encounter this type of system is - hook it all up, then turn on the power. This is nothing more than a guess and has less than a 50/50 chance of working when the power is applied.

Phasing in this manner is asking for trouble. You may not think so if you've done this with flea power motors on your desk or floor and have succeeded. But try it sometime with a 20 Hp or 40 Hp motor that you can't hold in a vise grip or with a piece of tape. If the phasing is wrong, it a guarantee that the motor can do some real damage, to you, or something around you when it jumps off the floor trying to equalize the static motion of the motor housing, with the "instantaneous" dynamic force of the armature.

Remember that old law ... Action and Reaction?

Procedure #1 ...

With a DC or PWM motor, connect a voltmeter across the armature wires. By hand, rotate the motor armature recording the direction. Rotate the motor fast enough to obtain a voltage deflection on the meter (it can be quite small, since we are only interested in +/-direction, not a voltage value). Record the + or - voltage direction (not value) with respect to the direction the motor armature was rotating.

When the motor armature + wire (the wire which had the voltmeter + lead attached) is attached to the motor drive amplifier + connection, and the motor armature - wire (the wire which had the voltmeter - lead attached) is attached to the motor drive amplifier - connection, a DC signal applied to the motor drive amplifier signal input connection of the same polarity that the armature produced when you had rotated it, will cause the motor to rotate in the same direction.

The reason for this, is that PWM or DC motor drive amplifiers are supposed to be "non-inverting". That is, a + signal input yields a + voltage output from the + armature output connection, and the same for a - signal input.

The method for phasing system style #1 is ...

1) Power "OFF" to the motor, drive, and control systems.

2) Perform the test outlined in Procedure #1.

3) Connect the motor armature as described in Procedure #1 to the motor drive amplifier.

4) Connect the feedback device channels A, *A, B, *B, I, *I to the position control feedback inputs.

5) Connect the position loop DAC signal wires to the motor drive amplifier signal input connections.

6) Power "ON" to the position control system only.

7) When the position control loop device is initialized, it will send a correction DAC voltage to the motor drive amplifier to maintain 0 position (no movement).

8) Place a voltmeter across the position control system DAC + output to SIG. (or +REF.), and - to COM. (or -REF.).

9) Slowly rotate the motor shaft in the same direction as was done to obtain voltage direction vs. rotation in Procedure #1.

10) Since the position loop is trying to maintain 0 movement of the motor, a voltage direction opposite to the +/- reading obtained in Procedure #1 should appear.

    If the voltage reading direction (+/-) is the same as the reading obtained in Procedure #1 ...

    a) Power "OFF" the position control loop.
    b) Swap either channel A and *A, **or** channel B and *B position feedback wires (but not both).

       Note: In the case of single ended feedback devices, swap channel A and channel B.

    c) Proceed with Step #6 above.

11) Power "OFF" the position loop.

12) Power "ON" the position loop and re-establish its position holding ability.

13) Power "ON" the motor drive amplifier.

14) Position holding should be in effect.

15) Command the position loop to **slowly** move the motor in what **you** consider the forward (+) direction.

16) If the motor turns in the correct direction, proceed to step #17.

    If the motor is turning backwards ....

    a) Power "OFF" both the position control system, and

the motor drive amplifier.
b) Swap channel A and *A, **or** channel B and *B feedback wires (but not both pairs).
c) Swap the + and - armature wires at either the motor, or motor drive amplifier (which ever is easier).
d) Proceed with step #12 above.

17) With the motor now properly phased, the next step is to verify the index marker phasing.

First check to determine if your control system requires a level triggered, or edge triggered Index signal.

Edge triggered and Simple level trigger

If the control triggers on an Index edge, or simply on an Index level, verify if the I and *I feedback inputs are correct. If they need to be reversed, simply swap the I, and *I wires, or follow the directions supplied by the position control Mfg.

Complex Level triggered

If the control triggers on an index level relative to a corresponding level on one or both of the main axis feedback channels (A/B), and it has been determined that the Index level is wrong, simply swap the index I and *I wires.

If the Index level is correct, but out of phase to channels A, and/or B ...

a) Swap channel A and *A.
b) Swap channel B and *B.
c) Swap the motor armature wires
d) Proceed to step #6

18) This completes the phasing of the position loop only style system.

System Style #2 Single Velocity loop - no position loop

Procedure #2 ...

Perform the test described in Procedure #1. When complete, connect the voltmeter across the tachometer wires. As you rotate the motor in the same direction as you did for Procedure #1, a voltage will appear on the voltmeter. Record the + or - tach. voltage direction.

With the motor armature hooked up as described in Procedure #1, the tachometer should be connected to the motor drive

amplifier so that its voltage is opposite to that of the signal input requirement.

This will produce the negative feedback the motor drive amplifier requires to maintain motor stability.

The proper method for phasing system style #2 is ...

1) Power "OFF" to the motor, drive, and control systems.

2) Perform the test outlined in Procedure #2.

3) Connect the motor armature to the motor drive amplifier.

4) Connect the tachometer wires to the motor drive amplifier tach. input connections per Procedure #2.

5) With the control DAC disconnected from the motor drive amplifier, place a jumper wire across the motor drive amplifier signal input connections SIG. to COM. (or +REF. to -REF.).

6) Since the motor may move when power is applied due to improper balance or other drive conditions, proceed with caution.

7) Power "ON" the motor drive amplifier.

8) If the motor is rotating, use the "Balance" potentiometer to stop motor movement. If the motor will not stop rotating, check for other adjustments in the drive manufacturers manual.

   Since it is possible that the Drive manufacturer has already inverted the tach. input (internal electronics)

   a) Power "OFF" the motor drive amplifier.
   b) Reverse the tach. signal wires at the motor drive amplifier.
   c) Proceed to step #7.

   If the motor still cannot be stabilized to zero motion, contact the motor drive amplifier manufacturer.

9) Power "OFF" the motor drive amplifier.

10) Remove the jumper applied in step #5 above.

11) Attach a control Voltage to the motor drive amplifier signal input.

12) Power "ON" the motor drive amplifier.

13) Increase the control voltage to slowly rotate the motor in the direction **you** consider forward (+). If the motor is rotating in the opposite direction

a) Power "OFF" the control and motor drive amplifier systems.
b) Swap the tach. + and - signal wires.
c) Swap the motor armature wires.
d) Proceed to step #12

14) This completes the phasing of the velocity loop only system.

System Style #3 Dual loop - Position and Velocity

I use this style system whenever I need high response, low speed stability, or high index rates. Typical systems would include registration, step and repeat indexing (4 to 25 shots per second), match-speed-and-cut type flying cutters (100 feet per minute +), etc.

Phasing of this type system is simply a combination of the preceding two systems with a minor variations of each.

The proper method for phasing system style #3 is ...

1) Disconnect the position loop from the system control.

2) Power "OFF" to the motor, drive, and control systems.

3) Perform the test outlined in Procedure #2.

4) Connect the motor armature to the motor drive amplifier.

5) Connect the tachometer wires to the motor drive amplifier tach. input connections per Procedure #2.

6) With the control DAC disconnected from the motor drive amplifier, place a jumper wire across the motor drive amplifier signal input connections SIG. to COM. (or +REF. to -REF.).

7) Since the motor may move when power is applied due to improper balance or other drive conditions, proceed with caution.

8) Power "ON" the motor drive amplifier.

9) If the motor is rotating, use the "Balance" potentiometer to stop motor movement. If the motor will not stop rotating, check for other adjustments in the drive manufacturer's manual.

Since it is possible that the Drive manufacturer has already inverted the tach. input (internal electronics)

a) Power "OFF" the motor drive amplifier.
b) Reverse the tach. signal wires at the motor drive amplifier.
c) Proceed to step #8.

If the motor still cannot be stabilized to zero motion, contact the motor drive amplifier manufacturer.

10) Power "OFF" the motor drive amplifier.

11) Remove the jumper applied in step #5 above.

12) Attach a control Voltage to the motor drive amplifier signal input.

13) Power "ON" the motor drive amplifier.

14) Increase the control voltage to slowly rotate the motor in any direction to verify the velocity loop stability.

If the motor is unstable refer to and perform the manufacturers recommended drive setup.

15) Power "OFF" the motor drive amplifier system.

16) Connect the position loop feedback device channels A, *A, B, *B, I, *I to the position control feedback inputs.

17) Connect the position loop DAC signal wires to the motor drive amplifier signal input connections.

18) Power "ON" to the position control system only.

19) When the position control loop device is initialized, it will send a correction DAC voltage to the motor drive amplifier to maintain 0 position (no movement).

20) Place a voltmeter across the position control system DAC + output to SIG. (or +REF.), and - to COM. (or -REF.).

21) Slowly rotate the motor shaft in the same direction as was done to obtain voltage direction vs. rotation in Procedure #1.

22) Since the position loop is trying to maintain 0 movement of the motor, a voltage direction opposite to the +/- reading obtained in Procedure #1 should appear.

If the voltage reading direction (+/-) is the same as the reading obtained in Procedure #1 ...

a) Power "OFF" the position control loop.
b) Swap either channel A and *A, **or** channel B and *B position feedback wires (but not both).

Note: In the case of single ended feedback devices, swap channel A and channel B.

c) Proceed with Step #6 above.

23) Power "OFF" the position loop.

24) Power "ON" the position loop and re-establish its position holding ability.

25) Power "ON" the motor drive amplifier.

26) Position holding should be in effect.

27) Command the position loop to **slowly** move the motor in what **you** consider the forward (+) direction.

28) If the motor turns in the correct direction, proceed to step #29.

If the motor is turning backwards ...

a) Power "OFF" both the position control system, and the motor drive amplifier.
b) Swap channel A and *A, **or** channel B and *B feedback wires (but not both pairs).

Note: In the case of single ended feedback devices swap channel A and channel B.

c) Swap the + and - armature wires at either the motor, or motor drive amplifier (which ever is easier).
d) Swap the + and - tach. signal wires at the motor drive amplifier.
e) Proceed with step #24 above.

29) With the motor now properly phased, the next step is to verify the index marker phasing.

First check to determine if your control system requires a level triggered, or edge triggered Index signal.

Edge triggered and Simple level trigger

If the control triggers on an Index edge, or simply on an Index level, verify if the I and *I feedback inputs are correct. If they need to be reversed, simply swap the I, and *I wires, or follow the directions supplied by the position control Mfg.

Complex Level triggered

If the control triggers on an index level relative to a corresponding level on one or both of the main axis feedback channels (A/B), and it has been determined that the Index level is wrong, simply swap the index I and *I wires.

If the Index level is correct, but out of phase to channels A, and/or B ...

a) Swap channel A and *A.
b) Swap channel B and *B.
c) Swap the motor armature wires
d) Proceed to step #18

30) This completes the phasing of the position loop only style system.

****

What we have just covered are the three types of control loops used in perhaps 99% of all systems. Additional control loops would be phased using combinations of the presented methods as required. If re-phasing of lower loops is necessary, it must be done following the guidelines as indicated. It's obvious that the more loops, the more carefully you must plan your approach to prevent winding up with an out of control mess.

Don't jump steps, and don't be intimidated by first-timer's worries. The steps work, and will become second nature when you've done it a few times.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

Chapter VII

A PID Analogy

Have you ever tried to think of an analogy for a PID gain structure that people could quickly relate to? The one currently in use with "shocks (Kd)", "springs (Kp)", and ??? for the Ki term hasn't really allowed good discussions to prevail. If something can't be explained clearly, then in most cases, what it affects usually suffers. System tuning has suffered from this and has always been somethine of a "seat-of-the-pants" operation (high degree of experience required), very complex to explain, and/or is generally accomplished "good enough".

The purpose of this chapter, is to explain an analogy for the PID structure in a form such that, hopefully, clears up any misconceptions as to what the PID algorithm is doing. At the end of the chapter, think about how the PID terms, your system, and the analogy relate. Once you've been dynamically put through this analogy, can you to correct the PID elements (tune) in the your specific application, according to what you're "seeing", "hearing", or "feeling"?

****

I) Activating your Kp

1) Stand up straight with your arms at your sides.

2) Have this read to you - Close your eyes.

3) Carefully lift one foot off of the floor, and place it behind the other (at ankle level).

4) Immediately you will feel the ankle of the foot supporting you reacting to changes in body balance.

****

This is "your" **Kp** error correction system reacting to the instantaneous balance changes sensed by your mind. The update period is the time it takes for you to mentally record and correct for these changes.

To clarify what is happening, your mind has a preset Kp gain value (learned over time) which is used to compensate for offsets in your balance. As you "feel" your balance change, your mind multiplies your Kp value by the balance offset (error). It then applies the adjustment value determined to your ankle. If you consciously put your "learned" Kp number to 0, your system will stop reacting and you'll fall. If on the other hand you raise your

Kp value too high, your ankle will oscillate out of control (unstable).

This action is exactly how the Kp will respond in your motor system.

****

II) Activating your Kd

5) Slowly raise your arms until they are extending outwardly to the left and right side of your body horizontally. Try to maintain them in this position for the remainder of the analogy. This position will be noted as "Zero".

6) After a few moments, you may notice that your arms have shifted (possibly in an abrupt manner) to a new position in order to help maintain your balance.

7) Also notice that your arms may maintain the new position for a period of time "longer" than the "normal" update period being observed by your ankle Kp. Your arms positions may change to "new" positions prior to you being able to return them to the zero position (if you can at all).

****

This "adjust and hold over time" method of reacting to changes in the bodies balance, is the Kd reacting to error differences at the end of given time slots (Kd update). The update time for the arm (Kd) reaction may be different than the time required to update the ankle (Kp) reaction. What you should note, is that your actual mechanical system may also require different update time periods for Kd and Kp. The Kd has an update time period which can be adjusted to the mechanical requirement. A starting point for the Kd update time period is 1/10 the system mechanical time constant.

****

III) Activating your Ki (Il)

8) At this time, try to maintain your body at the zero attitude position (standing straight on one foot, eyes closed, arms out horizontally from the body).

9) Notice that your torso may shift position, then recover slowly. Your mentally monitoring your balance on the way back to 0 (which by the way, actually worsens your ability to maintain zero position).

10) Notice that as you recover to zero, your ankle (Kp), arms (Kd), and torso (Ki) never stop correcting, but will not

make as large corrections as they did when you were at 0 position.

****

The long term Torso loop re-adjusts itself continually, but over multiple update periods, and with "growing or decaying" amounts, to help "pull" the system into, and maintain "zero **error**". Your torso is the Ki reaction. At the beginning of any move (0 to desired velocity) the newly computed total Ki error value is large in comparison to what it is when the system was not moving (horrizontal system).

As the move continues, the total Ki error value will grow. Newly generated individual error values will, therefore, be a progressively smaller <u>percentage change</u> of the overall Ki value.

As your body moves to "catch up" to where the controller (your mind) says it should be, the newly computed Ki values may be still as large as they first were (your body is still leaning, or bent over), but its percentage in relationship to the <u>total</u> Ki error is smaller, thus reducing its effect.

Once stability (new balance) is achieved, the Ki value will change as required to restore you to your initilized balance position (see #5 above). Also, notice that your Kp, and Kd body elements are working toward "holding you up", while the Ki element is working toward "straightening you up".

It should be obvious that if the Ki total is allowed to grow to a value which is too large for your system (body) to handle, you'll get out of control and fall. To prevent an excessively large torso reaction, the mind installs a limiting (Il) value. Your body will not tip beyond this point (that sense of falling stops further accumulation). But notice that because of this, your following error (ability to recover) got worse. Also, notice, that if your body is not limited in the way it can react (ie... Integral limit (Il) is set to high), you'll overshoot your target position, and oscillate about the position point.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

Chapter VIII

Inside the PID loop - How it works

In the field of servo systems, the PID loop has become a very popular method of gain control. Tuning the PID loop, however, still seems to present the same set of problems for most everyone. Some of the questions which always "pop-up" are ...

1) What do the PID elements do?
2) Is there an adjustment order, and if so ... which element do you adjust first?
3) How interactive are the elements?
4) Is it necessary to change the gain (Kx) values for varying loads?
5) How can I insure optimum response?

The objective of this chapter is to give the user of the PID structure a clear understanding of just how the PID loop works. With a good understanding of PID element interaction, you should have a better understanding of how the PID will affect your real time system. An added feature will be a dramatic improvement in your tuning ability.

The PID Structure

Before we proceed to the PID algorithm, it will be important to understand certain terms as used throughout this discussion.

K = A generally accepted letter used to signify "Gain", an arbitrary multiplier, or a constant. In this discussion, it will stand for a multiplier. The gain term will be indicated by a letter placed next to the K, such as Kp for Proportional gain.

Update = An arbitrary, but evenly spaced period of time, which controls when the motion registers will be "updated" with position, velocity information. The DAC error correction calculations are also accomplished in the update, along with other monitoring requirements. Update time periods are chosen by either the designer of the hardware, or the software writers (in the case of processor based motion control devices).

Some of the factors governing how small the update time can or should be are: DAC conversion time, speed and resolution of the system and processor capability.

Error = Calculated Trajectory position - Actual position

When the servo is commanded to move, the trajectory generator will perform the following calculations ...

Calculated Velocity Register value ...

= velocity register value + acceleration register value

Calculated Position Register value ...

= position register value + velocity register value

It's important to understand that the current position register is used to keep track of the actual system position, and to provide the math registers a means of calculating the **Error** during PID loop updates.

The only difference between velocity and position moves is ...

Velocity : No final stopping position will be given to the motion controller. Thus when a "Start" command is given, the motor to rotate indefinitely (or until it is commanded to stop). When a "Stop" is commanded, the motor will follow a calculated deceleration curve (slope) to a stopped motion condition.

Position : A position to move to will be loaded into the motion registers. Math work will determine the position to begin the deceleration in order to follow the deceleration slope, and ensure the stopping point will be the required position to move to.

As important as the **Error** term is, it is also important to understand the value of the DAC capability. The DAC (Digital to Analog Converter) is the interface between the computer computations, and the motor movement. The two most popular DAC sizes are 8 and 12 bit. This means that the DAC can subdivide its total Peak-to-Peak output voltage, or current swing, into 256, or 4096 steps.

Simply stated, the higher the DAC bit count, the finer the velocity control capability (well suited for very fine resolution systems). Keep in mind that the motor will require a certain friction "breakaway" current to perform a move. If, for example, the system has a 5 InLb. friction torque loading, a motor that can generate 20 InLb/Amp, and a motor drive amplifier with an input requirement of +/- 10 Vdc, and a gain of 200 Amps/Volt, it will take ...

5/20 = **0.25 Amp.** into the motor to make it move, and
10 x 0.25/200 = **0.0125 Vdc.** into the motor drive amplifier, to allow it to deliver the 0.25 amp to the motor..

An 8 bit, +/- 10 Vdc. DAC, produce's 0.039 Volt/step, and a 12 bit DAC 0.00488 Volt/step. The 8 bit DAC will produce more than enough motor current to break friction (in the above example) with a one step change in the DAC output, while the 12 bit DAC will require 3 DAC steps. If the system requires fine velocity resolution (12 bit DAC), keep in mind that the friction window is considered an "Electronic Backlash" and could be significant in moves requiring interpolation (see Chapter XI - discussion on deterioration of DAC resolution due to friction).

As has been presented before, the classical PID structure consists of four independent elements.

Output = SetPoint + Proportional + Integral + Derivative

Element definition

Output : This is a count result actually applied to the DAC input. The count will be converted into either a voltage, or current to be used by the device drive amplifier (generally +/- 10 Vdc.).

SetPoint : This is a fixed count value which is usually used with analog style control (valves etc.). For servo motor control, it could be used to offset the effects of gravity in a vertical type system.

Setpoint = Constant value

If it is used, its count value is directly applied to the servo controller DAC input.

Proportional : This value is derived by directly multiplying the Kp term by the position error.

Proportional = Kp x Error

The proportional count result is directly applied to the servo controller DAC input.

Integral : This value is derived by first multiplying the Ki term by the position error obtained in the given sample time. The result is then adding to the previously computed integral total.

Integral = Old Integral value + (Ki x Error)

The integral count result is directly applied to the servo controller DAC input.

Derivative : This value is derived by multiplying the Kd term by the difference between the position error determined in this sample time - the position error

determined in the previous Kd sample.

Derivative = Kd x (Error - Previous Error)

The Derivative element can be updated in a different time than the other two elements. This will increase it's dynamic range giving the system more time to react (high inertia systems).

The derivative count result is directly applied to the servo controller DAC input.

The next step is to understand how each element is determined, and how they interact with each other.

PID Element Operation

Note: The following "Basic" program was used to produce the PID error values as will be discussed. It would be advantageous to use the program to get a "feel" of how each element reacts with various error conditions.

```
10 'PROPORTIONAL + INTEGRAL + DERIVATIVE
15 '************************************
20 '
25 KP = pp                  'Put your kp value here
30 KI = ii                  'Put your Ki value here
35 KD = dd                  'Put your Kd value here
37 IL = ll                  'Put your Il value here
40 DIFUPDTE = 3             'Num. of updates to wait
45 '                         before next differential
50 '                         calculation
55 '
60 EROR = 20: INTEG = 0: LASTERR = 0: DIFTMEBASE = 0
65 '
70 FOR X = 1 TO 20          'Number of update periods to do
75 '
80 ERRDIFF  = EROR - LASTERR        'Calculate change in error
85 PRINT X,-EROR                    'Printout any answer
90 PROP  = KP * EROR                'Calculate Kp
95 INTEG = INTEG + (KI * EROR)      'Calculate ki
97 IF INTEG > IL THEN INTEG = IL    'Chk against Integral limit
100 '
105 DIFTMEBASE = DIFTMEBASE + 1
110 IF DIFTMEBASE < DIFUPDTE GOTO 140
115 '
120 DIFF  = KD * ERRDIFF
125 DIFTMEBASE = 0                  'Reset the differential
130 '                                    update clock
135 LASTERR = EROR              'Current error to memory
140 EROR = EROR + (-((PROP+INTEG+DIFF)/100*5)+20) 'Calc Error
145 NEXT X                      'Do the next update
```

I) Setpoint (SP)

The SP is simply a fixed bias value, used to develop a fixed output voltage/current from the DAC. This value **will** sum together with the error portions of the PID structure.

To understand the use of the SP, let's assume Kp, Ki, and Kd are zero (0), and SP is some value. All of the DAC output voltage then would be derived from the SP value. Since there is no error correction applied, the servo motor will try to rotate at the SP speed (and in the SP direction).

So what could the SP be used for? Perhaps a vertical system without a counter balance. Since gravity will generally cause a downward motion of the load from the required position point, a small to moderate SP count (depending on the weight of the system) could be applied to the DAC input to counteract this effect.

It would be possible to use the servo motor drive amplifier "Balance" (or Offset) control to accomplish the same feat (assuming the drive amplifier has one). It would also be possible to allow the proportional, integral, and/or derivative gain functions to perform this operation, but as you'll see later, system stability would be less than satisfactory if it were done in this manner. It should be understood, that the SP value should be used discriminately, and in small doses.

II) Proportional

The proportional element is described as ...

Proportional = Kp x Error

Note that the proportional element is a "proportioned" error value, and its effect is immediate (the result is applied at the end of each servo loop update calculation period).

Since the proportional element deals with an immediate proportioned error value, it is used to compensate for immediate changes in servo error position. Changes in position generally occur during acceleration, deceleration, and in moves where velocity changes occur because of "on-the-fly" shifting, or changes in the system dynamics.

Anytime the system overpowers the motor's ability to "keep up", the proportional element will be there to counteract the increase (or decrease) in the new following error calculated. It's the prime element used to develop the "stiffness" desired to hold the systems position "at rest".

A method for setting the proportional gain is ...

With Ki and Kd set to zero, raise the Kp value until the motor just begins to oscillate. Reduce the Kp value until the oscillations stop. Reduce it once more to a value 15% lower.

This setup will put the proportion loop element into an "almost hair trigger" mode. The slightest deviation of the system from the commanded position will cause an immediate Kp correction to be generated.

The Kp value is not really designed to be the sole mechanism in maintaining the stability of the motor. Using the following example, it will become clear that all of the PID elements are necessary to "round-out" the control loop. All of the examples to be discussed will use the mechanical concept shown in Figure 8.1.

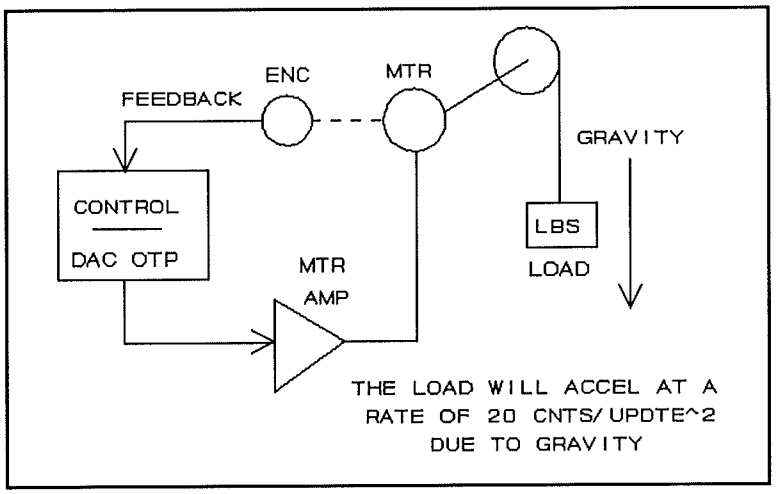

Figure 8.1

Example #1 - Vertical system at rest.

Load weight = capable of moving the system 20 encoder counts per update period in the down direction due to gravity.

12 bit DAC Kp = ? Ki = 0 Kd = 0

100 DAC input counts will try to overcome the effects of gravity by moving the system up 5 actual counts. Can Kp by itself, stabilize the system to 0? Set the Kp to 10.

Update#	Error	DAC Input (Count's)
1	-20	10 x -20 = -200
2	-30	10 x -30 = -300
3	-35	10 x -35 = -350
4	-37.5	10 x -37.5 = -375

The system appears to be stabilizing at 40 counts below zero.

Set Kp = 30.

Update#	Error	DAC Input (Count's)
1	-20	30 x -20 = -600
2	-10	30 x -10 = -300
3	-15	30 x -15 = -450
4	-12.5	30 x -12.5 = -375
5	-13.75	30 x -13.7 = -412

The system appears to be stabilizing at 12 counts below zero.

Set Kp = 50.

Update#	Error	DAC Input (Count's)
1	-20	50 x -20 = -1000
2	10	50 x 10 = 500
3	-35	50 x -35 = -1750
4	32.5	50 x 32.5 = 1625

The system is oscillating out of control.

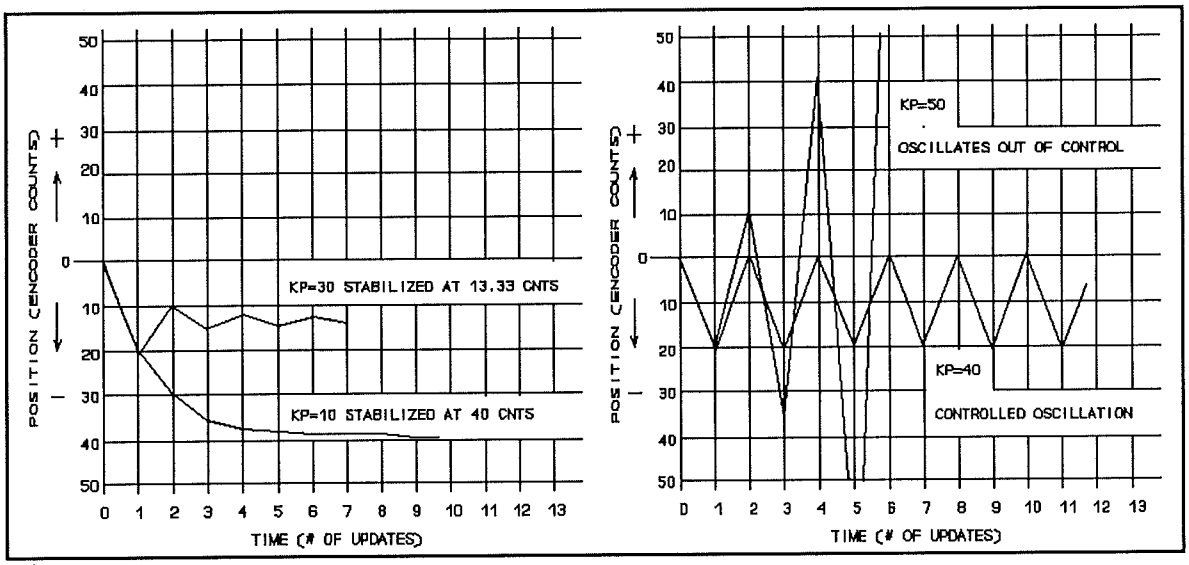

Figure 8.2

The system will oscillate indefinitely once the Kp allows the system actual position either recover to the zero position point, or crossover the zero position point. This is because the error will either disappear completly when zero position is reached (recycling the controlled oscillation), or will produce a reversed DAC output that will actually assists gravity and drive the load in the doward direction.

If you try other Kp values, you'll find that there's really no value of Kp that will simply stop the oscillations, **and** hold the system at the requested zero position point (see Figure 8.2).

III) Integral

The integral element is described as ...

Integral = Old Integral value + (Ki x Error)

The integral value is shown to be a long term accumulation of error corrections over time. The time part of the result comes from the fact that the error is summed to previously calculated Ki error values in successive update periods. When an update period has elapsed, a new integral value is calculated. Since the integral is accumulating count value over time, the integral will either grow or get smaller as succeeding calculations are made.

Note, that the integral element has more effect at the start of a move profile (instant correction), but its real purpose is to compensate for motor loading, and close up following error (long term correction).

Example #2 - Vertical system at rest.

Load weight = capable of moving the system 20 encoder counts per update period in the down direction due to gravity.

12 bit DAC Kp = 0 Ki = ? Kd = 0

100 DAC input counts will try to overcome the effects of gravity, and move the system up 5 actual counts. Can Ki by itself, stabilize the system to 0? Set Ki = 10.

Update#	Error	DAC Input (Count's)
1	-20	(0 + (10 x -20) = -200
2	-30	(-200 + (10 x -30) = -500
3	-25	(-500 + (10 x -25) = -750
4	- 7.5	(-750 + (10 x - 7.5) = -825
5	13.75	(825 + (10 x 13.75) = 687
6	28.125	(687 + (10 x 28.12) = 406
7	28.437	(406 + (10 x 28.43) = 121
8	14.53	(121 + (10 x 14.53) = 23

Ki is too large ... Set Ki = 3

Update#	Error	DAC Input (Count's)
1	-20	(0 + (3 x -20) = - 60
2	-37	(- 60 + (3 x -37) = -171
3	-48.45	(-171 + (3 x -48.45) = -316
4	-52.63	(-316.3 + (3 x -52.63) = -474
5	-48.91	(-474.2 + (3 x -48.91) = -621
6	-37.86	(-621 + (3 x -37.86) = -734
7	-21.13	(-734.5 + (3 x -21.13) = -798
8	- 1.13	(-798.2 + (3 x - 1.13) = -801
9	18.95	(801.63 + (3 x 18.95) = 745
10	36.19	(744.77 + (3 x 36.19) = 636

It appear's as though the Integral element by itself can't do the job. Let's combine and use both the proportional and integral elements to stabilize the system position.

Set ... Kp = 30 Ki = 3 DAC = proportional + integral

Update#	Error
1	-20
2	- 7
3	-12.45
4	- 7.85
5	- 8.97
6	- 7.07

Update#	Error
7	-6.96
8	-5.97
9	-5.57
10	-4.93
11	-4.51
12	-4.04

This result is much more stable. You can see that the error isn't fluctuating as wildly as in the two previous workups. Also note that the ideal value for stabilization is 400 counts (5 counts of actual motion for 100 counts of DAC input). However, also note that the stabilization took about 8 update periods. How can this be sped up without trashing the stability?

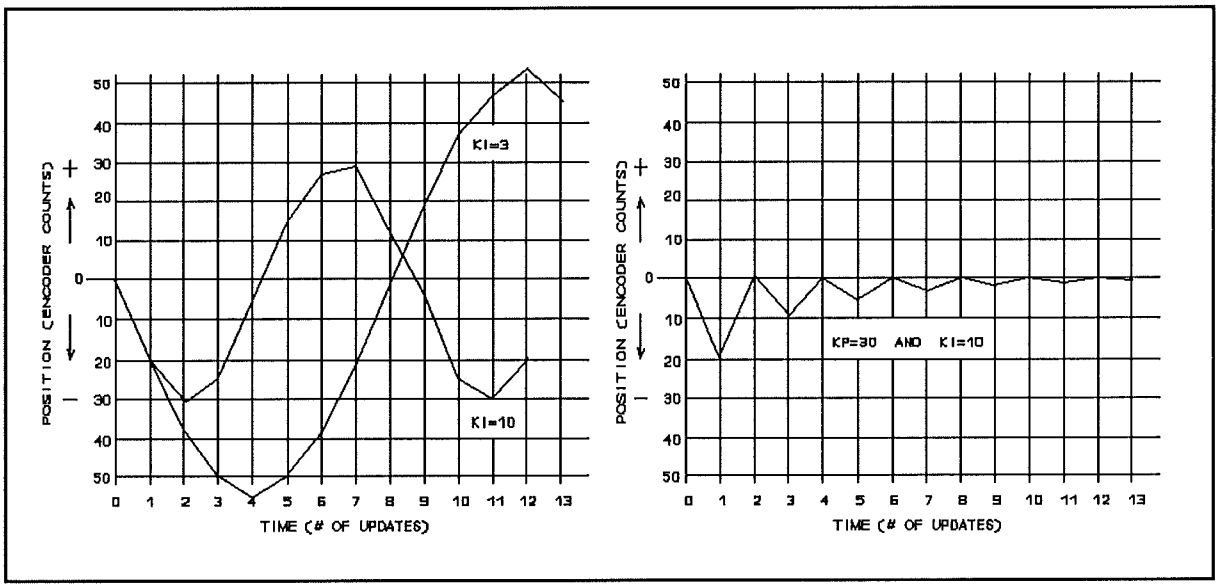

Figure 8.3

To increase the system response to error changes, increase the Ki, and to increase the stability of the Ki over time, set the Integral summation limit (**Il**). The total value of the Ki result can be restricted by setting the Integral gain limit (il) to any value that stops the growth of the summation total. This will allow larger Ki elements to respond rapidly to the calculated errors, without allowing the accumulated Ki total grow to unstable proportions. As the Ki is increased, the Kp may have to be reduced, since the initial reaction of the Ki is similar to the Kp term.

Let's repeat the last workup with ideal Kp, Ki, and Il settings .... Kp = 20 Ki = 30 Il = 400 Kd = 0

Update#	Error	Update#	Error
1	-20	7	0
2	0	8	0
3	0	9	0
4	0	10	0
5	0	11	0
6	0	12	0

Vary the Il ....
Set ... Kp = 20 Ki = 30 Il = 300 Kd = 0

Update#	Error	Update#	Error
1	-20	7	5
2	5	8	5
4	5	10	5
5	5	11	5
6	5	12	5

Set ... Kp = 20 Ki = 30 Il = 500 Kd = 0

Update#	Error	Update#	Error
1	-20	7	0.313
2	5	8	- 0.156
4	- 2.5	10	0.078
5	1.25	11	- 0.039
6	- 0.625	12	0.020

Note the following ...

1) The error originally went to 0 within one update period,

2) The system will be stable only with the right selection of PID element values.

3) When using Ki, always set the Il limiting term.

****

When tuning servo systems, each must be considered (and generally is) unique. A well learned procedure can make your tuning effort "easier", but understanding the PID elements, their interaction with the servo loop, and learning how to "read" what your system is telling you, is a must.

****

IV) Derivative

The Derivative element is described as ...

Derivative = Kd x (Error - Previous Error)

The derivative element is shown to be a term that reacts to a **change** in error over time (derivative sample time). At the end of the derivative update period, the derivative value is calculated by multiplying the Kd value by the current error **minus** the error calculated in the previous derivative update sample time. Since the derivative is only active if the error value has changed, from one update calculation to the next, it will not have any affect as long as the system is stable (whether stationary or moving).

Note that the Kd term has more effect at the start of a move profile, or when there's a change in the load, since the error usually gets smaller as the profile continues. To show the differential effect, let's continue with another example of the vertical system analogy.

Example #3 - Vertical system at rest.

Load weight = capable of moving the system 20 encoder counts per update period down due to the effect of gravity.

12 bit DAC Kp = 30 Ki = 3

100 DAC input counts will try to overcome the effects of gravity, and move the system up 5 actual counts. Set Kd = 7

Update#	Error	Update#	Error
1	-20	11	-1.37
2	- 3	12	-0.44
3	-10.45	13	-1.03
4	- 1.95	14	-0.38
5	- 5.51	15	-0.57
6	- 1.80	16	-0.11
7	- 4.7	17	-0.30
8	- 1.6	18	-0.10
9	- 2.5	19	-0.23
10	- 0.48	20	-0.08

Reviewing the results, the system appears to have become stable at the sixth update whereas the previous example without the derivative element, it took over ten. Although the system is still coming into 0 position, it's evident that by choosing correct PID gain values, this could happen much sooner. Also, if you use higher gain values and allow the error to cross over the desired position point, the system will oscillate. The values you pick for the PID elements should be set to yield fast system stabilization.

It should be apparent that because the differential element

reacts immediately (similar to the proportional element) it will also help to stabilize the move similar to Kp. The Differential term, however, will retain its applied correction value for a number of update periods. This feature allows the Kd to act as a setpoint. Instead of reacting to every error presented in every update period, the Kd update time allows the system to "respond" prior to sampling and applying another Kd correction value. If the Kd update time is set to 0, then the Kd is treated the same as Kp. A starting point for the Kd update time is 1/10 of the system mechanical time constant (see Chapter X - lines 3060, and 3070).

The Kd analogy generally accepted is to think of the derivative term as a car shock absorber. The "shock" is designed to react to major changes in the cars attitude. The reactive shock absorber applied force is "held" for a period of time to prevent violent wheel oscillations.

To restate, the elements are interactive, and depending on what you are trying to accomplish in tuning your system, you must know how the elements interact.

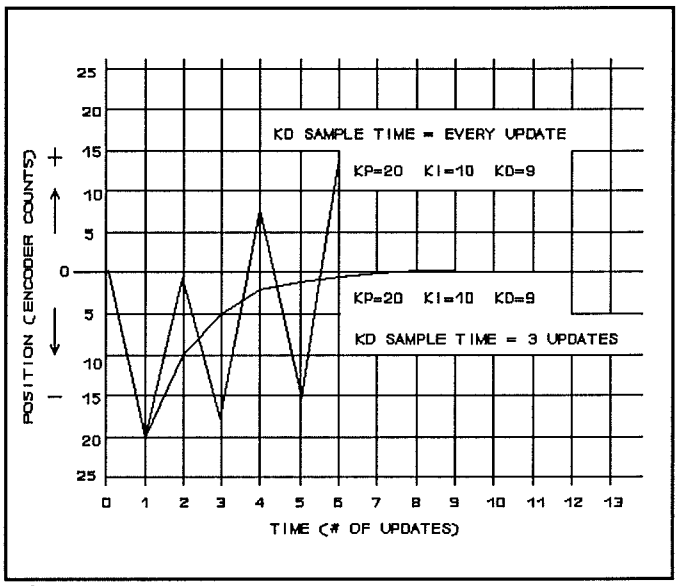

Figure 8.4

Discussion

The normal PID filter operation (as previously discussed) could have acceleration take-offs too severe for the load, due to the linear calculations of the velocity loop, and the reactiveness of the of the PID elements when a move begins. With some systems, it's desirable to produce an "S" curve acceleration/deceleration profile to soften the inertial loading off the motor system.

If this situation exists in your system, there are three possible cures ....

1) Use Gain Break Accel/Decel Profiles.

Therefore, use the PID values determined to maintain good loop following response while running at the 60% (or greater) velocity point. Prior to achieving the 60% velocity point, use reduced (or different) PID loop values. This will allow the system inertia to develop some the following error on take-off to soften the loading. A gain break is generally placed at 30%, and 60% ot the required velocity.

2) Use an "S" curve Acceleleration, Deceleration Profile.

An "S" curve approach is given in Chapter X "A PID Simulator". Also, a graph explaining "S" curve vs Trapezoidal Acceleration is shown in Figure 11.11. The objective is to gradually increase the commanded velocity (acceleration ramp) until it is "safe" to apply the full velocity (acceleration rate) without damaging, shifting, or losing control of the load.

When tuning any system, first determine what the system has to do. Optimizing response is nothing more than insuring the system can do the necessary profile, in the allotted time, without compromising system performance, and without damaging the product.

Why run the unit at 100 feet per minute when it can perform the required specification at 50 feet per minute? You might gain product throughput, but you could possibly have some side-effects like prematurely wearing out the mechanics. If you need to run at 100 feet per minute, make that a design specification, don't just arbitrarily run it faster.

If the unit will be doing multiple axis interpolation, the resolution and loading of the axes must be taken into account. The Following Error generated by each axis, for identical profiles, should be identical.

When optimizing the profile, the tool you should use is an oscilloscope. With a scope probe connected to the DAC output (make sure your scope is not grounded when doing this test), first set the Kp value as described in (section II) above. You'll "see" the DAC oscillate prior to "feeling or hearing" it. Next run a low speed profile. As the system is moving back and forth, adjust the Ki and Il values for the best flat line stability when at speed, and good rounding when coming off the acceleration ramp. Continue to raise the velocity (while under load) in small increments re-adjusting the Kp, Ki, and Il as necessary. When you've completed this operation, try varying the load. If noticeable oscillations occur, start applying some Kd. A starting point for the Kd time base should be approximately 1/10 the system mechanical time constant (refer to Chapter X (PID Simulator) for a description of the Electrical, and Mechanical time constant calculations).

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28