Chapter IX

Tying the Servo Knot

With all of the available technology, and all of the capability at hand, the world of motion control should be very well defined. It's surprising to find, however, that problems which plagued us years ago, are still with us. There's an ever increasing number of questions being asked relative to servo tuning, the servo control loop, and the how different gain structures work.

In previous chapters, methods were given for thinking about, and applying the component parts of a servo system. The intent of this chapter is to pull the system pieces together (tie the knot) by integrating "real world" outside forces into the problem. The objective is to give you more ammunition when thinking about your system requirements and selecting your control. The bottom line is to make your system functional to the required specification <u>the first time around</u>.

I) Brush, Brushless, DC, AC, or Stepper ?

When choosing a motor, the one you select has no real bearing on the choice of <u>motion controller</u>. It's the system load, inertia, profile, environment, etc. that will govern the choice of your motor package.

The electrical and physical differences of available motor types, ensure enough variety of performance to successfully choose one for your application. Only you, as the engineer of the project can judge which motor/amplifier package will satisfy your system requirement and budget.

The following information shows some simple criterion that can be used to help you determine which type of motor might be useable in your system ....

Brushless

1) High speed - short index moves
2) Heavy loads (high torque control)
3) Short duty cycle moves
4) Resolver to Quadrature converter option
   (no physical encoder requirement)
5) High power to size ratio
6) Low to medium armature inertias
7) High accel./decel. capability

PWM Motor

1) Long high speed moves
2) Low to Medium speed short moves
3) Light to Medium loaded index moves
4) Brush replacement required
5) Medium power to size ratio
6) Low to medium armatures inertias
7) Medium accel./decel. capability

Stepper

1) Light load moves
2) Speed/torque dependency
3) Positioning = +/- 3% of a full step
4) No encoder requirement
5) High power to size ratio
6) Medium armature inertias
7) Low speed operation
8) Low to medium accel./decel. capability

General purpose DC

1) Low to Medium speed
2) Light to heavy loads
3) High inertia armatures
4) Velocity mode or low accuracy positioning
5) Not designed for true servo application
6) Brute force on/off control
7) Low to medium accel./decel capability

AC

1) Low to high speed
2) Low to Medium accel./decel. capability
3) Light to Heavy load capability
4) Medium to high armature inertias
5) Brute force on/off control
6) Moderate servo positioning capability

This information is based on my experience in the field, and general information from motor manufacturers. The objective of this listing is to give you a starting point. As you gain experience, modify, or build your own list of guidelines. Knowing what you are comfortable with can save you hours of uneventful testing on future projects. You shouldn't have to begin "at the beginning" each time you need to choose a motor.

As an added item, you should always have at least two motor brands your comfortable with, as well as vendors. This will ensure you get the best pricing, and short lead times from when the order is placed (depending of course on the motor style).

II) Smart Drive Considerations

The motion control computer is the work horse for controlling move profiles which must be continuously altered (either in their velocity or position trajectories due to changing system conditions). There are many instances, however, when a reasonably "smart" drive is perfectly capable of running a system without the help of a computer control (refer to Chapter I).

With good **timing charts** (refer to Chapter XII), and **machine operation knowledge**, a smart drive could save you hundreds (and perhaps thousands) of dollars in equipment, and labor.

In many cases, the motion control computer can actually inhibit the performance you're trying to achieve. The questions you need to ask, in order to understand if your system will be able to function under simple velocity control, are outlined in the following discussion. If your system **is** capable of operating with a "simple" smart drive, don't force a computer into the problem. It could become just that, a problem!

First, if your system is to run **only** in a velocity mode with no specific stopping position, it's a very good candidate for a smart drive.

The next series of questions to answer are ...

1) What velocity stability does your system require, and can it be achieved with a tach. loop, if not with simple current control?

2) If a stopping position is required, what is its tolerance? Can a **controlled ramp** be utilized to meet this tolerance when a stop switch activation is given?

3) If your system requires multiple velocities in a given move, is the time it takes for the smart drive to make the velocity change (real world), within acceptable timing requirements of the **timing chart.** Can alternate timing, or can the acceleration or deceleration be flexed to meet the need?

4) If the system requires the acceleration and/or the deceleration to change while the move is in progress, can the change(s) can be accomplished in an allowable time window of the **timing chart**?

5) If your system is to be operated by some form of PLC (Programmable Logic Controller), could it be cost effective for you to put the motion control portion of the system on the PLC structure you're already planning. Can the PLC as an "On/Off" type of control fulfil the need of the motion(s), while allowing the drive to handle

the actual move?

6) Does your system have relatively small changes in load weight when moving?

If you answered "yes" to <u>most</u> of the questions posed, it would be well worth your time to investigate smart drive alternatives.

<u>III) Tach. or No Tach. ?</u>

In any design effort, try to accumulate enough information about the system to keep the "first time on line" success rate as high as possible. Some of the questions to ask, should be pointed toward determining if a tachometer velocity loop should be used.

The questions would have to do with velocity stability, move distance (time), as well as the accel. and decel. requirements. Although all systems need the answers to these three questions (load profiling, etc.), it isn't immediately obvious how a tach. loop could improve the positioning requirement, or the DAC resolution.

The effect the tach. loop has on position, and DAC resolution can be demonstrated in the following examples ....

<u>Example 3.1 ..... Tach. loop for an Inline System</u>

<u>A) DAC resolution, and Velocity considerations</u>

Assume the system must operate at a velocity of 1 RPM.

Load weight = 500 Lbs.
Friction Coeff. = .07

Screw Radius = 0.5 inch
Screw Pitch = 5

500 pulse quadrature encoder (10,000 counts per inch)

$$\text{Static Friction force} = \frac{(500 \times 0.7) \times 0.5}{5} = 3.5 \text{ In.Lb.}$$

Motor Torque Constant (Kt) = 5 In.Lb. / Amp.
Amplifier gain = 25 Amps / Volt

Breakaway Friction = Running Friction = 3.5 InLb.

To break friction, a current of 0.7 Amp. (3.5 / 5) must flow through the motor. This equates to a motor amplifier input voltage of 0.028 vdc. which is in turn supplied by the DAC.

If the DAC is 12 bit (4096 steps), and the full +/- 10 vdc analog signal voltage is used, then one DAC step is equivalent to 0.0048 vdc. It will, therefore, take 6 DAC steps (5.834 rounded out) to break friction, and maintain the running operation.

What this means, is if Gain parameters are set to produce 6 DAC steps with one error count, the equivalent DAC resolution will be reduced to 682 steps (4096/6), which is only a 2.6 times resolution improvement over the 8 bit DAC verses the 16 times resolution improvement it was supposed to be. The Integral parameter in the PID loop is normally used to take out the friction part of the error. Once the system "breaks" friction, the Integral term is reduced and if the Proportional and Derivative terms are set correctly, the DAC will be properly scaled. However, if an integral gain factor is not part of the gain structure you are using, it's likely that the example given will have a reduced DAC resolution. You'll need to verify this point to insure your system performance.

If a 2000 RPM motor is to be used with the 12 bit DAC, the capability of the DAC would be 0.976 RPM per DAC step (2000 / 2048). However, the resolution is now 5.865 RPM per DAC step (2000 / 341), since the velocity would constantly be interpolating between 6 DAC steps and 0 DAC steps to achieve the 1 RPM spec.'ed.

By adding a tachometer velocity loop to the motor package, and properly setting the motor amplifier gain, the motor amplifier will with the lowest DAC voltage (0.0048 vdc) applied to it, ensure that the motor is turning. Note however, that this movement could be at a velocity higher than 1 RPM due to the gain of the amplifier required to break friction. The control, however, would be interpolating between 1 DAC step, and 0 DAC steps to achieve the 1 RPM per sec. specified. Not only would there be a significant improvement in the "smoothness" of the acceleration profile, but as an added feature, the gain structure of the controller could be maintained at a more optimum setting than with no tach. loop. This in turn would increase the responsiveness, and stability of the overall system.

A byproduct of the tach. loop, would be to place the motor into a speed (velocity) regulation mode instead of the current mode used in the example. This would ensure that the velocity of the motor would be more predictable, and in the case of higher speeds, more stable, and could be simulated with greater accuracy.

B) Position, and Move Distance Considerations

Use the same system parameters as in 3.1-A above ...

Requirement: Move the system at a velocity to a position of 3 encoder counts from where it currently is (relative).

In example 3.1 A, the system needed 6 DAC steps to break friction. The trajectory generator will move only 3 counts to completion (see Chapter X), and therefore, the error generator will only develop 3 error counts for the move. The two worst case gain situations would be low gain for long move stability (and no motion for the short moves), or high gain for short move capability (with possibly unstable long moves). Obviously, two different gain settings might be required (a software cure).

But what if we again, use a velocity loop (tach.) giving the motor amplifier the ability to force the motor to "break friction" with a 0.0048 volt single DAC step input. The motor amplifier would force the motion on the first DAC step insuring a move completion. As was shown previously, the unit could now have the low gain DAC requirement for meeting the long move stability, and also have the short move dimension capability.

The tach. loop would take the motor drive package out of the current mode giving the system higher velocity, and position stability with changing load requirements without having to "model" the system (to locate its "weak" points), or constantly change its gain characteristics.

In the discussion on gain structures, you'll see how the tach. loop can be the hardware counterpart of software gain structures such as Feedforward, and Feedback, and help maintain close to zero Following Error.

IV) Mechanical Considerations for Control Selection

The general machine operating requirements lead to the addition of sensors and other monitoring or controlling devices that will in some way work with the product motion.

For instance, in a tension system there will be some form of tension monitor which will feed tension information to the motion control unit. The motion controller will have to analyze the information, and adjust the velocity of the system accordingly.

On the other hand, a system that makes wood products might have multiple product sensors, limit switches, and possibly product measuring equipment. All of these signals might have to be handled in "real time" vs. polling. A smart motion controller (possibly a stand alone) might be needed to assist the host computer (or PLC) in "real time" operation.

By developing a machine (or axis) **timing chart** (refer to Chapter XII), the various sensor and interlock requirements can be determined. From this, the software can be outlined (not written) to separate the general operation from the "real time" operation. Profiling the system would then allow selection of the motor amplifier package, and finally, with all of the "creeping feature-isms" accounted for, the control can be chosen.

Creeping feature-isms are usually paid for out of project profit money. They are the after thoughts that come about once the machine has been completed. If you can determine these features "up-front" the project will be more of a dream than a nightmare.

V) Differential or Single-ended ?

The problem with interfacing equipment, is that all of the electronic knowledge that people have forgotten about, usually comes into play. Impedance matching, frequency, resistance, voltage, current, power dissipation, noise, and other considerations, must be planned for when interconnecting system components.

The selection of a Differential or Single-Ended signaling device should be done up-front, along with working out the system interface requirements. Since it is possible to obtain a differential or single ended interface for most any electronic device (such as general purpose signaling, computer I/O, encoder, motor amplifier, etc.), the choice to use becomes environment, distance, and budget oriented.

The main advantage to using the differential driver over the single ended unit is the high noise immunity achievable. This becomes very apparent (and important) in factories where the length of wire runs can exceed 100 ft. Differential signals also work well with simple twisted pairs saving the cost of shielded cable (each case must be carefully examined, however, before the shielded cable approach is tossed).

The use of differential signals, whenever possible, will help maintain that extra degree of isolation electronic systems need. In cabinets, where high power motor amplifiers, and/or switching is done, the shielded differential approach can be a lifesaver.

VI) Other than Quadrature Feedback

There is no rule that only encoders can present the feedback to the servo loop. As a matter of fact, there are many aircraft simulators around that take in a variety of strain gauge signals (force information) which are molded with various math formulas. The resulting answers are then applied to cylinders, motors, or other type of movers for adjustment of the pressure vectors sensed.

The key here is to insure that the resolution of the feedback device is some value tighter than what your tolerance requires. It must be remembered that all computers, counters, converters, etc., have propagation delay, math round out, and tracking errors.

All of these add up to the inability to achieve the position accuracy and/or move stability you required. Since the important thing is to be stable and repeatable, a feedback device with a tight resolution, can usually eliminate the errors caused by computer math, and general servo hunting. I generally use a 5 to

10 times rule for this scenario. For example, if I have a 0.001 inch tolerance, my feedback sensor is calibrated to 0.0002, or 0.0001 inch. By doing this, the system will have the ability (via resolution) to perform within the required tolerances.

If it's repeatable ....
it's predictable ....
and therefore, it can be controlled.

VII) System Friction and DAC Resolution

It's not very widespread knowledge, that the static friction in a given system can indirectly alter the resolution of the controller DAC. Because of this, many systems lose the velocity stability and/or the low count positioning accuracy the designer was trying for (refer to Section III). Just how the friction causes this, is the topic of this section.

In systems where a motor is used to move the load, the motor must produce enough torque to overcome the system friction force prior to motion occurring. This friction force has absolutely nothing to do with the force (torque) required to accelerate the system (system inertia times system acceleration rate). The friction force (created by the system friction times the system weight times mover Arm length) is constant, and need only be exceeded to produce motion. The motor used to generate the motion, will develop the required force (torque) when a current is passed through it. The torque (force) that a motor can produce with a given current is equal to ....

Motor Torque = Motor Current x Motor Torque Constant

When the system was "sized" and a motor was selected, the motor's torque constant was chosen to ensure that the required system torque would be met. Even if the motor's torque constant was selected only to maintain low motor currents, the selected torque constant would be considered part of the motor performance requirement. The torque constant will then govern the amount of current needed to be delivered to the motor in order to move the system load. This of course will have to be delivered to the motor by the motor drive amplifier.

The motor drive amplifier will have a gain rating in amps per volt. It's this parameter which interfaces between the DAC and the motor. The motor amplifier gain is specified after selection of the motor. With knowledge of the system static friction force (breakaway, and running), and DAC voltage resolution required, if the entire DAC resolution is to remain intact it must be verified if a motor tach. loop is necessary.

Example 7.1 ..... Inline System

Load weight = 500 Lbs.

Friction Coeff. = .07

Screw Radius = 0.5 inch
Screw Pitch = 5

$$\text{Static Friction force} = \frac{(500 \times 0.7) \times 0.5}{5} = 3.5 \text{ In.Lb.}$$

Motor Torque Constant (Kt) = 5 In.Lb. / Amp.
Amplifier gain = 25 Amps / Volt

To break friction, a current of 0.7 Amp. (3.5/5) must flow through the motor. This equates to a motor drive amplifier input voltage of 0.028 vdc. which is in turn supplied by the DAC.

If the DAC is 12 bit (4096 steps), and the full +/- 10 vdc analog signal voltage is used, then one DAC step is equivalent to 0.0048 vdc. It will, therefore, take 6 DAC steps (or 5.834 rounded out) to break friction.

What this means, is that if the Gain parameters (Kp, Ki, Kd, Feedforward, etc.) are not altered for move profiles, the equivalent DAC resolution can be reduced to 682 steps (4096/6), if the running friction is the same as the breakaway friction. This would only be a 2.6 times resolution improvement over the 8 bit DAC verses the 16 times resolution improvement it was supposed to be.

If a 2000 RPM motor was to be used with a DAC capability of 0.976 RPM per DAC step (2000/2048), the resolution is now 5.865 RPM per DAC step (2000/341). To reduce, or eliminate this phenomena, application of a tachometer loop would be reccommended (refer to Section III).

It should be apparent that system friction can do more than just increase motor size.

VIII) System Response and Gain loops

Gain structures, how they work, and which would best fit a given application, is probably the most discussed topic in the industry. Each time a new gain structure appears, its generally presented as the answer all - to whatever problem exists.

The real problem is the confusion created, by not presenting enough information about the gain method, to allow the user to decide if it's correct for his application.

The topic of this section, therefore, is ....
Gains structures and what they do.

The gain structures discussed are ....

1) PID and Lead/Lag (PD)
2) FeedForward, FeedBack, and Adaptive
3) Notch
4) Self optimizing

A) PID and Lead/Lag (PD)

As has been presented before, the classical PID structure consists of four independent elements (refer to Chapter VIII). The PD loop utilizes the same calculations as the PID loop, but usually incorporates some other form of gain structure into the equation for added flexibility and control.

Output = SetPoint + Proportional + Integral + Derivative

Element definitions

Output :

This is the resulting digital count applied to the DAC amplifier. The count will be converted into either a voltage, or current for use by the motor drive amplifier (generally +/- 10 Vdc.).

SetPoint :

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

Setpoint = Constant value

If used, the setpoint value is directly applied to the servo controller DAC.

Proportional :

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

Proportional = Kp x Error

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

The proportional element provides immediate position error compensation. Changes in axis attitude, which generally occur during acceleration, deceleration, or with dynamic load (or system) changes. It's the prime PID element used to develop the "stiffness" required to hold the unit stationary, when not being commanded to move.

Integral :

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

Integral = Old Integral value + (Ki x Error)

The integral value is shown to be an accumulation of error correction over time. during each update period, a new integral error correction value is calculated. Since the integral is an accumulated 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. Its real purpose, however, is to compensate for motor loading and close up following error (long term correction) as the move progresses.

Derivative :

The derivative calculation is accomplished by multiplying the derivative value against a **change** in error noted between two updates determined by the derivative sample time. The error correction calculated is then fixed until the next derivative update calculation takes place.

Derivative = Kd x (Error - Previous Error)

The Derivative element can generally be updated with sample time periods ranging from 256 usec. to 65.536 msec. This increases the Kd's dynamic range allowing the system more time to react (as in the case of large inertia systems).

Since derivative correction is only active if the position error has **changed** (between derivative update sample periods), it will not have any affect as long as the system is stable (whether stationary or moving).

B) FeedForward, FeedBack and Adaptive

It should be obvious that when using a PID gain structure, an error between the desired, and actual positions **must** exist in order to generate a DAC output. It's also known that to simulate the response of a system, you need knowledge of the total system inertia, motor parameters, the system resolution, gearing, and general profile information.

The objective of the FeedForward, and FeedBack gain structures are to estimate what the system should be doing in future updates, and to make the corrections **now** for those estimates. The corrections, however, are accomplished by changing the value of the desired position, which will increase the error value, causing the error generator to produce a larger DAC output. Note though, that the system is still working with the lower PID, PD, etc. gain structure it originally needed to keep the system at its proper level of stability.

The purpose behind this gain structure, is to maintain as close to zero following error as possible. There are a couple of inherent problems with this method ...

Problem 1 - Since the system is being simulated in order to estimate the "future", all information used for the simulation must be correct for it to work.

Problem 2 - to optimize the estimates, the controller must have the ability to record the error along the path for future comparison and use. Most systems don't have the memory capacity to retain the amount of data it would take to do this, or the CPU overhead available to continually do the update simulations required.

C) Notch

The notch filter is a software constructed filter system similar in principle to those used in the Radio/TV communication field (Low-pass, Band-pass, High-pass). The intent of the filter with relation to the servo system, is to automatically change the response of the motor (and thus the system) in mechanical areas of instability, or resonance. The objective of the Notch filter is to eliminate these instabilities by reducing (dampening) the system response in those areas.

There are certain things about the system which must be known, and fixed in time, and position, in order for the Notch filter structure to work. If the system resonant frequencies change (due to load or system mechanical changes, etc.), or if your system operation does not repeat its performance in the areas in which the filter parameters had been calculated, the Notch gain structure might not handle the system as repeatably as expected.

This is not to say it does not work, it does, but simply to indicate that the systems it is applicable to, must be capable of being modeled. Therefore, any changes in the system using the Notch filter approach will require remodeling, thus re-tuning.

D) Self Optimizing Torque Profile

There are some gain structures which utilize the motor, and system parameters to compute optimum motor performance, while the system is operating (on-the-fly). The objective of a well tuned system, however, is to optimize to the required system (machine) profile, not the motor's capability. Therefore, this type of structure is useable in very specific applications.

Remember that ....

Optimizing the profile has nothing directly to do with the motor, it has to do with the machine timing requirements.

IX) Tuning considerations, and the Optimum Profile

Just what is the "optimum profile", and when can the tuning operation be considered complete?

Having tuned (or helped tune) hundreds of systems, I feel very comfortable in saying ...

A properly tuned system is one that performs as required.

The tuning operation begins at the specification. What is the profile requirement for your system in order to operate as planned? This is determined by the machine timing diagrams. The timing diagrams not only gives you machine timing, they also gives you action interlock requirements (monitors, limits, multiple machine interlocks, etc.), and motor duty cycle information. Duty cycle information is necessary for properly sizing the motor amplifier package. It insures the package selected will not electrically or physically fail.

Once the machine timing has been worked out, stick to it! If you want (or need) to make the system run faster, do it before it's built. A couple of months ago, I was called in on a tuning problem. I questioned them about the machine timing requirements for the axes having problems. I found that 0.5 sec. timing windows were "optimized" to 0.1 sec. or less. Needless to say, the inertia torque loading went up by square functions and the motors couldn't be brought under acceptable control (instability at the ramp down points - oscillation). However, when we tuned the axes to the timing chart requirements originally worked out by the "primary" designers, the 0.05 sec. profile worked!

You develop a timing chart to help in the design and understanding of the machine functionality. The inertia of the mechanics cannot be changed without redesign (not considering the load). The motor torque loading is a direct result of the acceleration (or deceleration) times the inertia. If, when the machine is built, you are optimizing the profile, optimize it to what your profile was specified to be, which is not necessarily the best it can be.

The ability to control a piece of equipment
- is limited only by -
the ability of that piece of equipment!

Don't try to make the system do something it wasn't "designed" to do, or that it's not capable of doing.

Stop when the system profile requirement has been met, not when the motor response capability has been peaked!

X) Closing the Servo Loop

The following discussion outlines a closed loop servo system operating with a PID gain structure (refer to Figure 9.1). As discussed in other sections, understanding how system friction can distort the DAC resolution is important to know when selecting the motor and motor amplifier package. The gain of the motor amplifier must be chosen to allow a one step DAC count to move the load if the full 2048 DAC steps are to be realized.

Once the size of the motor, and the gain of the motor amplifier are known, whether or not to put a tach. loop on the motor will be determined next. The positioning, and velocity stability requirement(s) will give you that answer. The gain values will be directly related to move velocity, and distance. By adding a tach. loop to the system, you can usually have an adequate gain structure for both fast and slow, long and short move situations.

The tach. loop also simulates, to a degree, the FeedForward and FeedBack gain structures, in that the tach. actually senses and forces a response to changes in the motor's velocity prior to the motion controller even knowing that changes occurred (update phenomenon).

The encoder resolution is then selected by multiplying the "real" resolution requirement by five to ten. Then using a PID simulator package (see Chapter X), the response curve of the trajectory generator vs. the actual system performance can be plotted. As you test the required profile, changes in the encoder resolution and PID values will either improve or worsen the "actual vs. desired" trajectory plots. When complete, the simulator will have helped verify the selected resolution, and PID entries.

Utilizing the proper tools to solve motion problems is no different than using the proper tools to change a light switch. The objective is to understand how to use the tool(s), and how to apply the numbers you getting. Proper understanding will simply enhance your profitability.

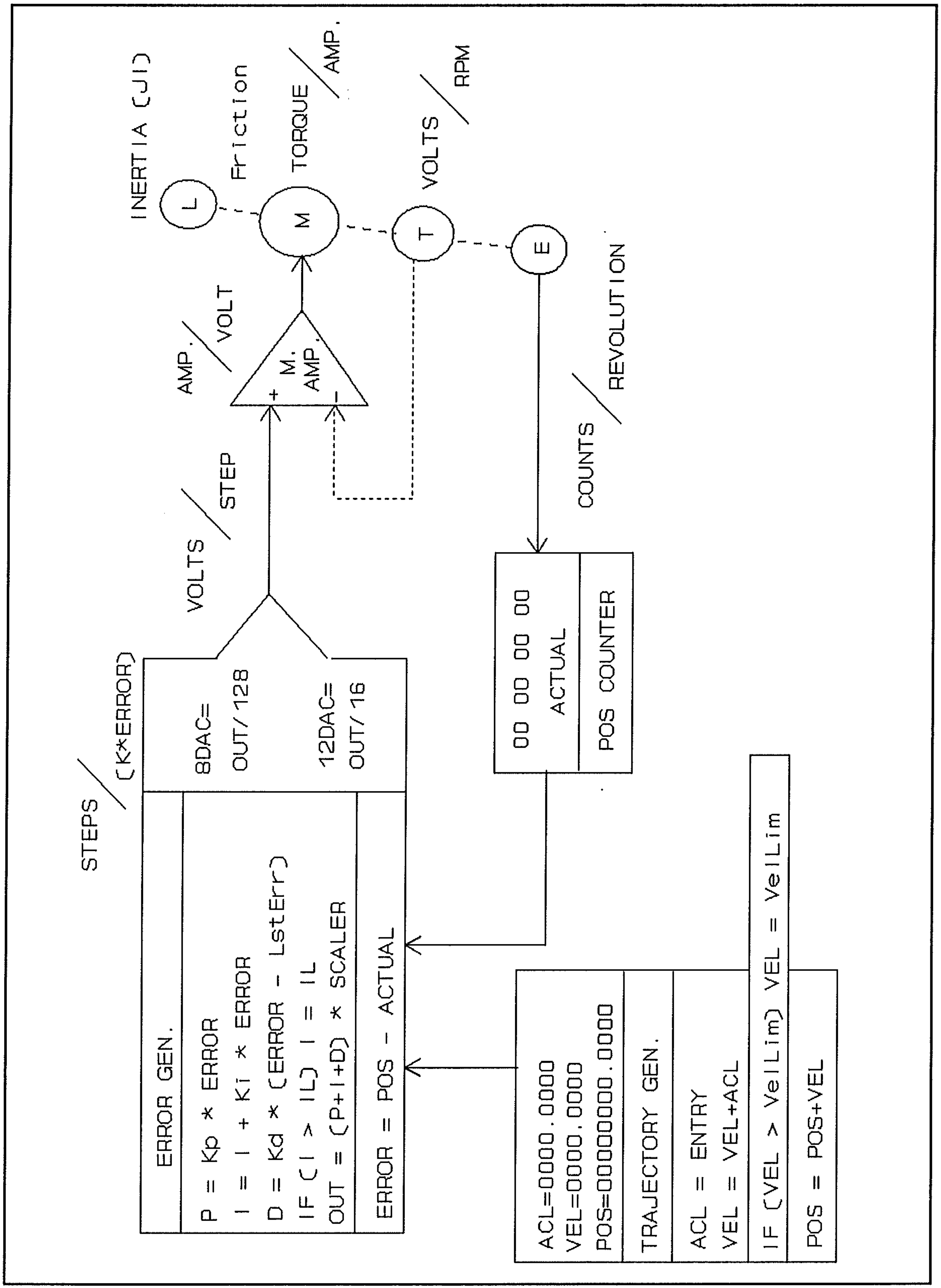

Figure 9.1 ... PID Servo Loop Control

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 ____