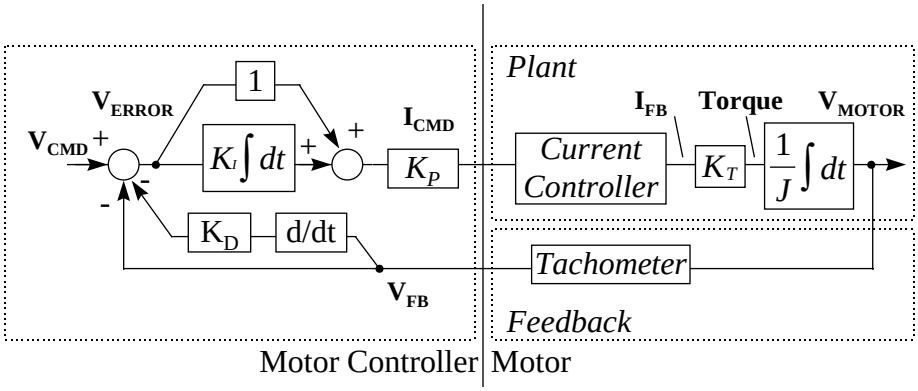
This article was the basis for the February, 1999 edition of "20-minute Tune-Up" printed in PT Design Magazine.

Closing a PID Velocity Loop

February, 1999

Closed-loop motion controllers are top performers. They are used in virtually every industry segment when applications demand the highest precision or the fastest motion. That performance comes at a cost: these controllers must be tuned, a process where various gains are adjusted so the machine achieves optimal performance. Tuning can appear mysterious, even intimidating, to engineers new to motion control. This article reviews the basics of the PID control loop including a tuning procedure. If you want to try it out yourself, you will be able to download a simulated PID controller and practice some of these principles. This article won't make you a tuning expert, but if you are new to motion control, it will give you a good start.

Every closed-loop system has three components: a controller, a "plant," and a feedback device. The plant is the mechanism that is being controlled. It might be a temperature bath, a pressure chamber, or, as in this case, motor and load. A feedback sensor is used so the system will have precise control of the plant. The controller compares the command and feedback signals, and after some processing, sends a control signal to the plant. That "processing" can take many forms, but the most popular is probably the "PID" controller. PID controllers are so named because they combine proportional, integral and derivative signals. Each of those three signals is scaled by a "gain." This is shown in Figure 1.



PID Velocity Control System
Figure 1

In Figure 1, the controller is on the left. The error is the difference between velocity command and feedback. The error is integrated and scaled by the gain K_I. A proportional gain of unity ("1") is summed with the integral term and then both are scaled by the proportional gain, K_P. The derivative is only in the feedback path and is scaled by the gain K_D. One unique characteristics of the PID controller is that the derivative is only in the feedback and not in the forward path with K_I and K_P. This is common in velocity controllers because this holds down overshoot; derivatives in the forward path usually produce excessive overshoot in response to a square wave command. The plant is shown in the upper right of Figure 1. This includes the current controller and the motor. The plant produces velocity from the current command. The feedback device is shown as a tachometer.

Each of the three gains has different functions. K_P, in the correct measure, provides stability allowing all gains to increase. But make K_P too large and the system will become noisy; larger still and the system will become unstable. K_I makes the system stiff—a large K_I will make it difficult for disturbance torques to move the shaft. But a large K_I also generates a lot of overshoot. K_D provides damping—it eliminates much of the overshoot caused by K_I. But too much K_D causes high-frequency oscillations. This behavior is shown in Table 1.

Gain	When you have enough…	When you have too much…
K_P	Stable fast response.	Noisy, even unstable.
K_I	Stiff system.	Overshoot, low-frequency oscillations.
K_D	Reduce overshoot from K_I.	High frequency oscillations.

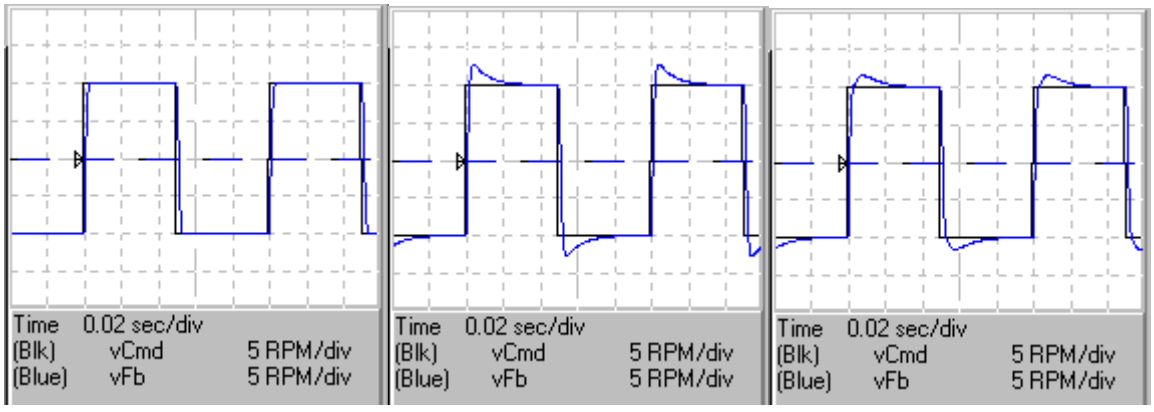
Effects of Tuning Gains
Table 1

A simple process for tuning

The process of setting K_I, K_D, and K_P can be tedious because of the problem posed by simultaneously adjusting three constants. And when the combination is wrong, the system will become unstable, generating a lot of noise and occasionally damaging the machine. The key to adjusting gains efficiently is using a process that allows you do decouple, as much as possible, the effects of one gain term to another. The process we will use is as follows:

1. Set all gains to zero.
2. Apply a square wave command.
3. Raise K_P until the system begins to overshoot.
4. Raise K_I until overshoot is approximately 15%.
5. Raise K_D to remove some of the overshoot.

When applied to the simulated system, Step 3 produced a K_P of 1.3, Step 4 produced a K_I of about 100, and Step 5 produced a K_D of about 0.0005. Figure 2 below shows simulated oscilloscope curves after steps 3, 4, and 5. Note that using K_I produces overshoot which normally cannot be completely removed. However, bear in mind that most velocity controllers do not need to respond to square wave commands; usually the command is from a position controller which is usually much gentler than a square wave. So, all overshoot does not need to be removed. Note that if you don't need integral gain, don't use it. Integral gain has the benefit of removing long-term error, but if that's not important for the application, don't expose the system to the problems it causes, chiefly overshoot.



2a *2b* *2c*

PID Control System with Different Gains
a) KP = 1.3, b) KP = 1.3 and KI = 100, c) KP = 1.3, KI = 100, and KD = 0.0004
Figure 2

Two problems that arise in tuning systems are noise and resonance. Motion controllers both produce and are susceptible to noise. Resonance, a common problem caused by the motor and load connected by a compliant shaft or belt system, causes oscillation which cannot be tolerated in most applications. The level

of K_P and K_D may need to be reduced to help with these problems; K_I does not have much effect on noise or resonance.

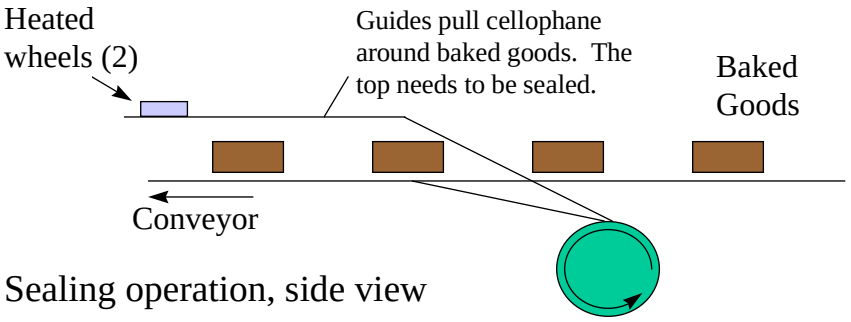
Practice tuning

The best way to learn is by doing. The simulation above was run on a stand-alone simulation package called ModelQ. ModelQ is designed to help teach principles of control systems. It runs on Windows 95, Windows 98, and Windows NT. It is available at no charge at {web site}. After installation, just click on "Run" and at top left, select the "Constants" tab and start adjusting constants. If you have questions about the operation of ModelQ, it has an integrated help manual. Finally, if you are wondering about how useful it is to learn on a simulation tool, you should know that this simulation has been compared to a real motor system and the behavior here is demonstrated in field equipment. If you are interested in practicing your controls skills, run the model and give it a try. Becoming skilled at tuning controllers takes time and experience. Like other crafts, the more you do it, the better you get.

Sidebar: Sealing cellophane in food packaging

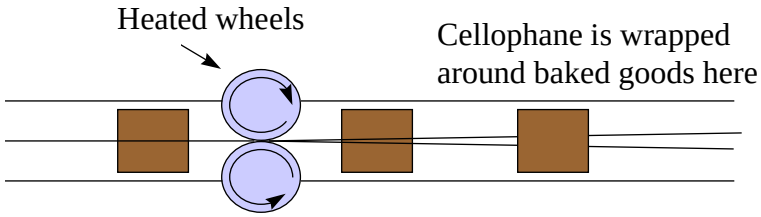
Tuning controllers requires an in-depth knowledge of the application—what the machine can do and what the customer expects. Every application is different and determining the needs is as important as setting the gains to meet those needs. One application is a sealer for a food packaging machine. On this machine, baked goods travel on a conveyor and are wrapped in a continuous roll of cellophane. The cellophane is joined across the top of the baked goods and must be sealed. This is done by two heated wheels which are turning in opposite directions at the same speed, and are synchronized to the conveyor. When the cellophane reaches the wheels, it is drawn between them and the heat on the wheel pinches the two sides together. The process can be seen in Figure A.

For this application, the most important behavior is that the wheels maintain synchronization with the belt. Since no long-term error could be tolerated, an integral gain was required. When the cellophane is drawn into the wheels, it frequently kinks and small folds are introduced. This makes the cellophane thicken and that increases the load on the motors which would otherwise maintain constant speed. However, the thicker cellophane slows the wheels. The tuning parameters K_P and K_I need to be as large as possible to keep the wheels turning at a constant speed. Because the command is near constant, overshoot to a square wave is acceptable and no derivative term is required.



Sealing operation, side view

Cellophane Roll



Sealing operation, top view