Programming: P.I.D. Filter Control

Programming: P.I.D. Filter Control:

The SmartMotor includes a very high quality, high performance brushless D.C. ServoMotor. It has a rotor with
rare earth magnets and a stator (the outside, stationary part) that is a densely wound multi-slotted electro-mag

Controlling the position of a Brushless D.C. servo's rotor with only the electro-magnetism to use as a lever is lik
a rubber band. Accurate control would seem impossible.

The magic that makes it happen is found in the "P.I.D. filter". The "P" stands for "Proportional", the "I" for "Integ
"Derivative". These are the three fundamental coefficients to a mathematical algorithm that intelligently recalcu
to the motor about 4,000 times per second. The input to the P.I.D. filter is the instantaneous actual position mi
be it at rest, or part of an ongoing trajectory. This difference is called the "error".

The proportional term of the filter creates a simple spring constant. The further the shaft is rotated away from i
more power is delivered to return it. With this as the only term the motor shaft would respond just as the end o
grabbed it and twisted. Just like a spring, if you twist it and then let go it will vibrate wildly. This kind of vibratior
mechanisms so what gets added is a shock absorber, the derivative term. If you slowly sat down on the fende
down because of your weight, based on the constant of the car's spring. You would not know if the shocks wer
hooked your toes in the bumper, however, and jumped up and down you could quickly tell whether the shock a
That is because they are not activated by position. They are activated by speed. The derivative term steals pc
of the rate of change of the overall filter output. The term gets its name from the fact that the derivative of posi
stealing power based on the magnitude of the motor shafts vibration has the same effect as putting a shock ab
the algorithm never goes bad.

Even with those two terms a situation can arise that will cause the servo to leave its target and that is created b
constant torque is applied to the end of the shaft, the shaft will comply until the deflection causes the proportior
equivalent torque. There is no speed so the derivative term has no effect. As long as that torque is there, the
of its target. That is where the integral term comes in. The integral of position is time, and the integral term mo

it back on target. There is also a separate parameter that is used to limit the scope of what the integral term ca

Each of these three parameters have their own scaling factor to taylor the overall performance of the filter to th
of any one particular application. The scaling factors are as follows:

KP	Proportional
KI	Integral
KD	Derivative
KL	Integral Limit

Tuning the Filter:

The task of tuning the filter is complicated by the fact that the terms are so interdependent. A change in one ca
settings of the others. The automatic utility relieves you of this aggravation, but you still may want to know how
When tuning the motor it is useful to have the status monitor running. This will allow you to monitor various bits
will reflect the motors performance.

KP=exp	Set Kp, proportional coeff.
KI=exp	Set Ki, time-error coefficient
KD=exp	Set Kd, damping coefficient
KL=exp	Set Kl, time-error term limit
F	Update PID filter

The main objective in tuning a servo is to get KP as high as possible, while maintaining stability. The higher KI

and the more under control it is. A good start is to simply query what to begin with (RKP) and then start increas
a time. It is a good idea to start with KI equal to zero. Keep in mind that the new settings do not take effect un
issued. Each time you raise KP, try physically to destabilize the system, by bumping or twisting it. Alternatively
program loop cycling that invokes abrupt motions. As long as the motor always settles to a quiet rest, you can

As soon as you reach the limit, you need to rediscover the appropriate derivative compensation. Move KD up a
the position that gives you the quickest stability. If KD is way too high, you will hear a grinding sound. It is not r
sign that you will want to go the other way. A good tune is not only stable, but reasonably quiet. After optimizir
you can raise KP a little more. Keep going back and forth until you have exhausted all you can do to improve t
Then it is time to take a look at KI.

KI in most cases is used to compensate for friction. Without it you will never get exactly on target. Begin with I
equal to 1000. Move the motor off target and start increasing KI and KL. Keep KL at least ten times KI during

Continue to increase KI until the motor always reaches its target, and then once that happens add about 30% t

by about 30% as well. You want the integral term to be strong enough to overcome friction, but you also want t
unruly amount of power will not be delivered if the mechanism were to jam or simply find itself against one of its

E=exp	Set maximum position error

The difference between where the motor shaft is and where it is supposed to be is appropriately called the "err
of the error is delivered to the motor in the form of torque, after it is put through the P.I.D. filter. The higher the
control the motor is. It is therefore often useful to put a limit to the allowable error, after which time the motor w
the 'E' command is for. It defaults to 1000 encoder counts, but can be set from 1 to 30,000.

a P.I.D. loop through the execution of a motion trajectory are unpredictable, but there are some that can be pre
preemptively.

KG=exp	Set Kg, gravity offset term
KGOFF	Kill Kg term during error
KGON	Maintain Kg during error

The simplest of these is gravity. Why burden the P.I.D. loop with the effects of gravity in a vertical load application, if it can simply be weeded out. If in a particular application, motion would occur with the power off due to gravity, a constant offset can be incorporated into the filter to balance the system. 'KG' is the term. KG can range from -8388608 to 8388607. To tune KG simply issue the 'KGON' command, while the motor is not servoing, and make changes to KG until the load equally favors upward and downward motion. If when there is a position error you want the motor to drop the load, then issue 'KGOFF'.

KV=exp	Set Kvff, velocity feed forward

Another predictable cause of position error is the natural latency of the P.I.D. loop itself. At higher speeds, because the calculation takes a finite amount of time, the result is somewhat "old news". The higher the speed, the more the actual motor position will slightly lag the trajectory calculated position. This can be programmed out with the 'KV' term. KV can range from zero to 65,535. Typical values range in the low hundreds. To tune KV simply run the motor at a constant speed, if the application will allow, and increase KV until the error gets reduced to near zero and stays there.

KA=exp	Set Kaff, acceleration feed forward

Force equals mass times acceleration. If the SmartMotor is accelerating a mass, it will be exerting a force during that acceleration. This force will disappear immediately upon reaching the cruising speed. This momentary torque during acceleration is also predictable and need not aggravate the P.I.D. filter. It's effects can be programmed out with the 'KA' term. It is a little more difficult to tune 'KA', especially with hardware attached. The objective, however, is to arrive at a value that will close the position error during the acceleration and deceleration phases. It is better to tune 'KA' with 'KI' set to zero because 'KI' will address this constant force in another way. It is best to have 'KA' address 100% of the forces due to acceleration, and leave the KI term to adjust for friction.

KS=exp	Set Ks, dampening sample rate

You can reduce the sampling rate of the derivative term, 'KD', with the 'KS' term. This can sometimes add stability to very high inertial loads. Useful values of KS range from 1 (the default) to 20. Results will vary from application to application.

PID1	Set normal PID update rate
PID2	Divide normal PID update rate by 2
PID4	Divide normal PID update rate by 4
PID8	Divide normal PID update rate by 8

The trajectory and P.I.D. filter calculations occur within the SmartMotor 4069 times per second. That is faster than is necessary for vary good control, especially with the larger motors. A reduction in the P.I.D. rate will result in an increase in the SmartMotor application program execution rate. The 'PID2' command will divide the P.I.D. rate by two, and the others even more.

If you do lower the P.I.D. rate, keep in mind that this is the "sample" rate that is the basis for velocity values, acceleration values, P.I.D. coefficients and wait times. If you cut the rate in half, expect to do the following to keep all else the same:

Halve all motor constants
Halve wait times
Double velocity
Increase acceleration by a factor of four.