MAC CENTRE
T. 34 93 214 06 00 F. 34 93 214 06 31
Olesa 14-16 6ª floor 08027 BARCELONA

TUNING THE SMART STEP SERVODRIVER

INDEX

0.- INTRODUCTION

The present document intends to guide you about how to tune a Smart Step servodriver to fit an application. Most of the concepts and explanation are valid for all mechanical systems driven by servomotors.

1.- MONITORING

1.1. The necessity.

When we are going to tune a servodriver we need some way to know the results and to evaluate how good (or bad) are the results. Normally it is much easier to know that a system is going bad that to confirm that is going well. Some symptoms are very evident: Noisy motor, mechanical vibration, clear difference between the desired position and the real position, etc. With the experience you can create certain skills and you can detect some problems and adjust properly just by 'hearing' the system.

Sometimes there seems that the system is working properly but, if you finely look, you can find some behaviour that can be easily improved but that it is not evident to the eye or to the ear.

That's why to be able to monitor the system variables is so important. It helps to tune properly a system giving the optimum performance, it confirms that all is OK or shows the cause of the problem if there is any.

The faster the machine is and the more precision is demanded, the most important is to have a good tuning, and the monitors are who will confirm this proper behaviour.

Don't leave an application without checking all relevant monitors and confirming that they give the right values.

1.2. Different kind of monitors.

We can classify between two kinds of monitors: those that are captured in real time and those that are filtered.

Monitors that are filtered are the Monitors that you can read using the Parameter Unit (Unxxx monitor) and those you can see in the different monitoring screens in the WMON Win (Unxxx, Inputs and Output status, etc.). Those parameters are filtered so the eye can see a more or less stable value, otherwise we will see some numbers that are changing so fast that it is possible to see nothing. Besides, those monitors are viewed trough a serial link, what means a certain delay.

The conclusion is that those monitors are very useful to know about the stationary state (when there are supposed not to have big changes in the magnitude that we want to check) or to have a rough idea about the behaviour.

For example, if we are interested to know the torque that we need to move a ball screw, that is, the friction, we need to check the torque monitoring with the motor running at a constant speed. In this case, the output torque is expected to be more or less constant. If we detect that in one certain area the torque increases suddenly, that may be the indication that there is any mechanical problem in this point of the ball screw. If you try to measure the same while you are accelerating or decelerating you will see that the torque is changing all the time and not in real time, no conclusions can be obtained then.

The second kind of monitoring are the ones captured in real time. This can be done using an oscilloscope and monitoring the analogue monitor output signals or connecting to the physical inputs and outputs of the servodriver. Another option is to use the Trace function in the WMON Win to monitor in real time different signals.

Those monitors captured in real time are essential to finely tune the dynamic behaviour of the most demanding application.

1.3. Filtered monitoring

a) Using the Parameter Unit.

With the Parameter Unit it is possible to see up to four monitors or parameters at the same time. In this example we give to the motor a continuous pulse train to obtain a speed of 116 rpm, in the torque monitor we see that the motor is producing a torque that corresponds to the 13% of the motor rated torque. We know now that the friction of the machine is that 13%. This data is very useful to dimension the motor, to set the autotuning and gains and to detect problems.

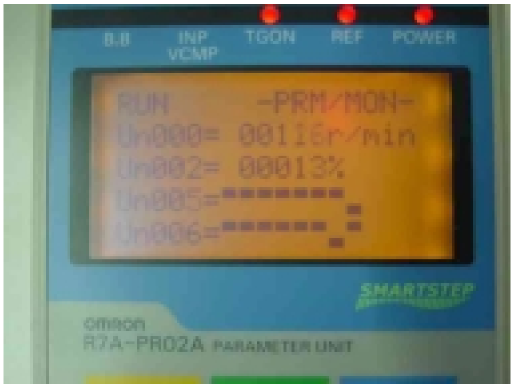
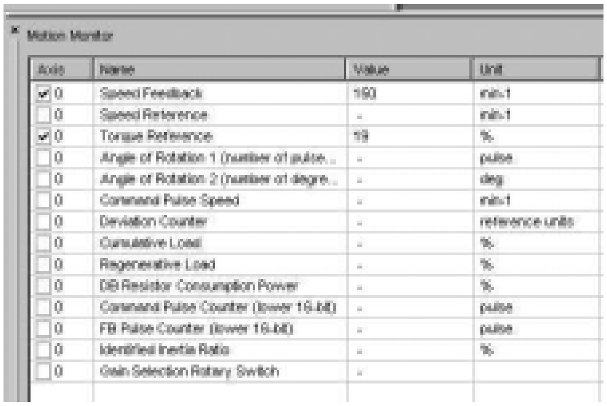


FIGURE 1. Parameter Unit

b) Using the WMON Win monitor screen.

Same monitoring can be checked using the WMON Win software. We can select which data we want to see with the compromise that the most data we monitor at the same time, the slowest refresh rate we obtain. Any monitor can be individually 'checked' to be monitored or not. In the example we update just the Speed feedback and the torque reference.



Motion Monitor

Axis	Name	Value	Unit
☑ 0	Speed Feedback	190	min-1
□ 0	Speed Reference	-	min-1
☑ 0	Torque Reference	19	%
□ 0	Angle of Rotation 1 (number of pulse...	-	pulse
□ 0	Angle of Rotation 2 (number of degre...	-	deg
□ 0	Command Pulse Speed	-	min-1
□ 0	Deviation Counter	-	reference units
□ 0	Cumulative Load	-	%
□ 0	Regenerative Load	-	%
□ 0	DB Resistor Consumption Power	-	%
□ 0	Command Pulse Counter (lower 16-bit)	-	pulse
□ 0	FB Pulse Counter (lower 16-bit)	-	pulse
□ 0	Identified Inertia Ratio	-	%
□ 0	Gain Selection Rotary Switch	-	

FIGURE 2. Monitor Screen

The equivalence between the Unxxx in the console and the Monitor in the Software is next:

Monitor n.	Monitor Item
Un000	Speed feedback
---	Speed reference. Only for W-series servodriver
Un002	Torque reference
Un003	Angle of rotation (number of pulses)
Un004	Angle of rotation (number of degrees)
Un005	--- (Input signal monitor, displayed as individual signals)
Un006	--- (Output signal monitor, displayed as individual signals)
Un007	Command Pulse Speed
Un008	Deviation Counter
Un009	Cumulative Load
Un00A	Regenerative Load
Un00B	DB Resestor Consumption Power
Un00C	Command Pulse Counter (lower 16-bit)
Un00D	Feedback pulse counter (lower 16-bit)

Fn007	Identified Inertia Ratio
Rotary SW	Gain Selection Rotary Switch

c) Comments about the most relevant monitor and guidelines for diagnostics.

Un000, Speed feedback and Un007, Command pulse speed display. As the Smart Step is a pulse commanded servodriver, the speed is a consequence of the position loop, that's why the 'speed reference monitor' is not existing and we have Un007 instead. If you apply a pulse train with a constant frequency, it is expected than the speed feedback will be constant and equal to Un007, if not, maybe the speed loop adjustment is not good.

Un002 Torque command. This monitor is very useful to know the friction in a machine. Give a pulse train with a constant frequency and check this monitor, It will tell you the friction. It is not useful to see fast peaks because it is filtered.

Un003 and Un004Useful to check that the encoder signals are right and that the motor axis is in a certain angle.

Un005 and Un006 Useful to check the hardware wiring.

Un008 Position deviation, Un00C Input pulse counter, and Un00D Feedback pulse counter. Useful to check that the position loop and command is working well. You can check (in this order) that you send all the pulses (in the PLC), then that the Smart Step received all the pulses (Un00C), next that the motor moved all the pulses (Un00D) and, if necessary, than the motor is in the right angle (with Un003). Un008 is the difference between Un00C and Un00D. The units are command pulses.

Un009 Cumulative load ratio. Is quite equivalent to the torque 'rms' value. This monitor gives you an idea of the dimensioning of the motor with the application in average torque. Run the machine with the hardest cycle and, after some seconds, check this parameter. If this parameter is much smaller than 100%, let's say 50%, that means that the motor is under-dimensioned and that you can put a smaller one. If the value reach 100% the servodriver will give A71 Alarm and that means that the motor is too small for this cycle (maybe this is an indication of a mechanical problem). If the value is below but near 100% (90% for example) that means that the motor is very near its limit. This is a risky situation because, although the machine works now, when the machine will become old and the friction will increase maybe A71 start appearing. The ideal situation is to have this value between 70% and 80%. This explanation is just a guideline, in some application the limiting factor to dimension the motor is not the rms torque but the peak torque.

Un00A Useful to verify the dimensioning of the braking resistor.

1.4. Monitors captured in real time

a) Using the oscilloscope.

The biggest advantage of using an oscilloscope is that you capture the signals in real time and view they in real time too. In machines where it is difficult to make a lot of tests is the best way to adjust the servo as you can change a parameter and immediately see the response. The oscilloscope is connected usually to the analogue monitor output (CN4) where you can see a voltage proportional to the feedback speed and another voltage proportional to the torque reference.

The main problem using those signal is that the accuracy of the outputs is 15% (this is because those analogue outputs are not using D/A converter but PWM output) and the signal is quite noisy (it is a plastic connector with twisted pair cable and not screened). If you want to see a clear signal you have to add a filter in the oscilloscope. Graph in figure 3 is taken from an oscilloscope and applying a 10 KHz filter to each signal.

Of course, an oscilloscope is a tool that is very useful to check another signals not so directly related with the servodriver, as the 'quality' of the command pulses.

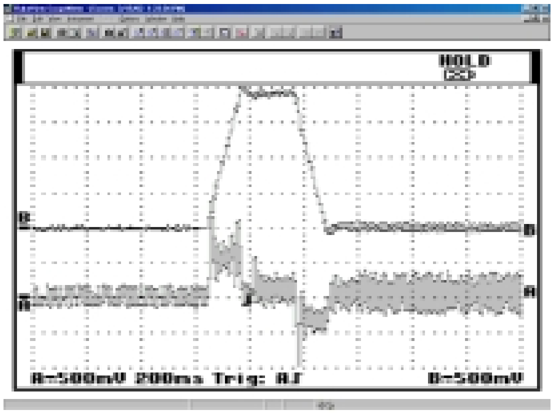


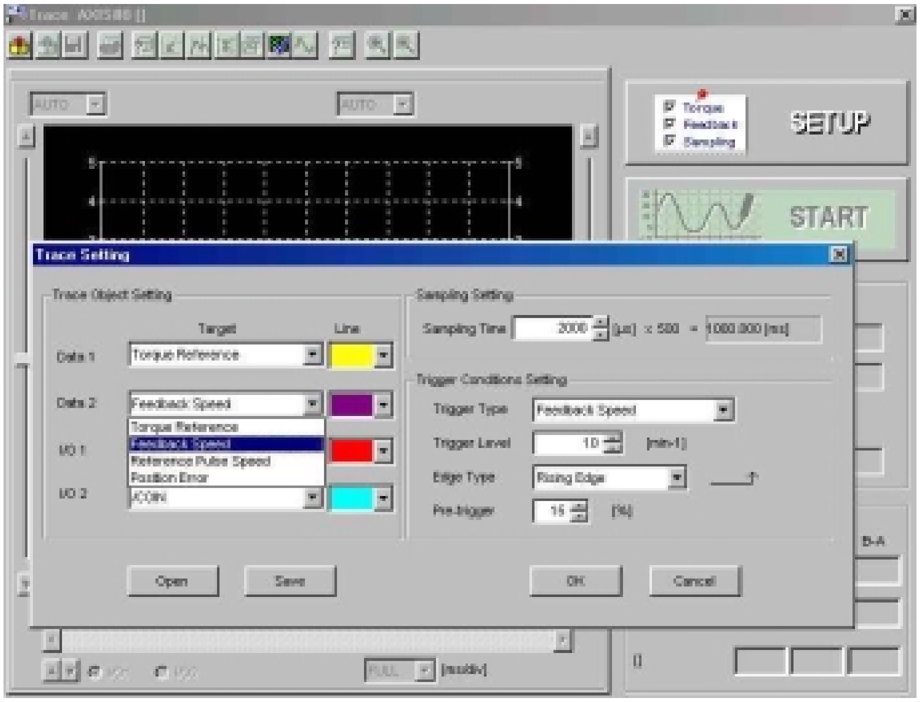
FIGURE 3. Oscilloscope graph

b) Using the Trace function in the WMON WIN

The WMON Win software has a tool (TRACE) that allows to capture signals in real time in similar way that an oscilloscope does. The data is stored in real time in a memory area inside the servodriver and, once the capture has been finished, those data is transferred to the PC.

The advantage comparing with an oscilloscope is that you just need the PC and that you can capture signals other than 'Speed Feedback' and 'Torque Command'. The inconvenient is that the time to view the captured graph is quite slow and, then, you can not change a parameter and immediately see the response.

Next screen shows a typical setting of the oscilloscope function: This window appears when selecting the 'SETUP' button in the 'TRACE' screen.



Trace Setting

Trace Object Setting

	Target	Line
Data 1	Torque Reference	🟨
Data 2	Feedback Speed	🟪
I/O 1	Torque Reference Feedback Speed Reference Pulse Speed Position Error	🟥
I/O 2	/COIN	🟦

Sampling Setting

Sampling Time 2000 [µs] x 500 = 1000.000 [ms]

Trigger Conditions Setting

Trigger Type: Feedback Speed
Trigger Level: 10 [min-1]
Edge Type: Rising Edge
Pre-trigger: 15 [%]

Open Save OK Cancel

FIGURE 4. Trace & Setup window

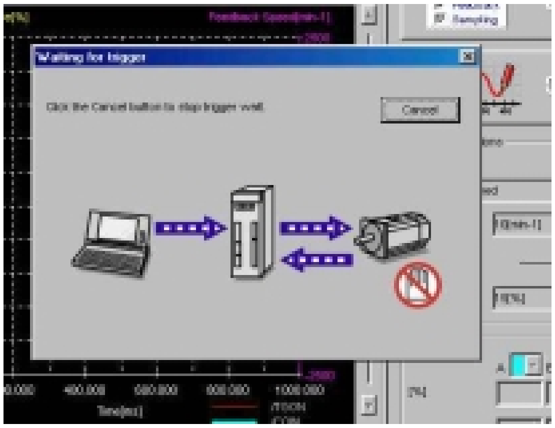
In Data 1 and Data 2 you select the analogue signal that you want to display, you can select between the next signals: ‘Torque Reference’, ‘Speed feedback’, ‘Reference pulse Speed’ and ‘Position error’

In I/O 1 and I/O 2 you select the digital signal that you want to display, you can select between: /TGON (motor rotating), /COIN (Positioning complete), /S-RDY (Servo ready), /BK (Brake command), ALM (Alarm), /S-ON (Run) and /ALM-RST (Alarm reset).

Next step is to select the timebase (Sampling Setting). You store 500 samples per signal. Selecting the ‘Sampling Time’, the total trace time will be 500 times this time. In the example we take a sample every 2000 µs and the total trace time will be 1 s.

At last, you have to select the trigger. The trigger determines the moment to start the capture. If you select ‘None’ the capture starts as soon you press the ‘START’ Button, if you select any trigger, the capture will start when the selected signal grows above (rising edge) or below (lowering edge) the selected threshold. The pre-trigger level allows to add an ‘offset’ in time for clearly view the desired signal.

Once the setup has been finish, press ‘OK’ and then, in the Trace screen, press ‘START’



Waiting for trigger

Click the Cancel button to stop trigger wait.

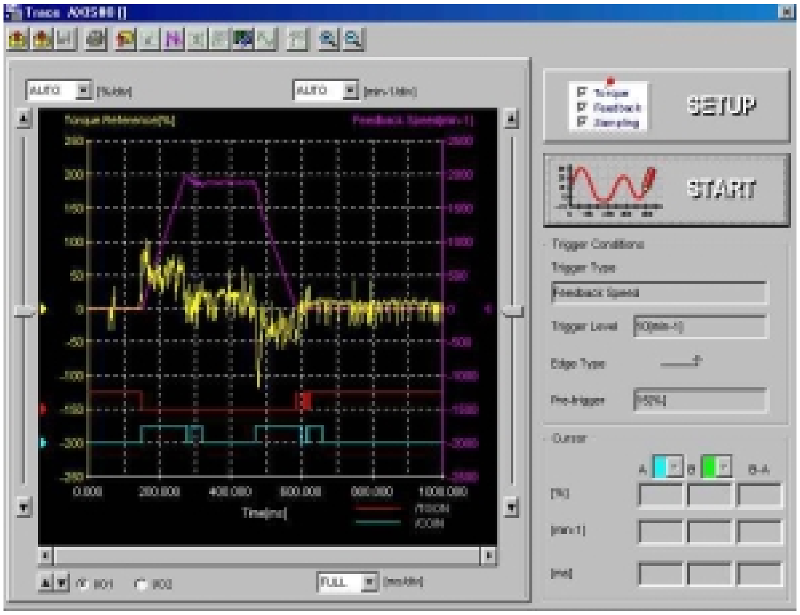
The software is waiting for the conditions for the trigger to occur.

Once those conditions are matched and the capture has been finished in the servodriver, the transfer of the information to the PC starts.

When this has finish, the taken graphs are shown as follows:

FIGURE 5. Trace capture

Figure 6 is the graph of the movement. You can compare with the signal taken with the real oscilloscope (Figure 3) and the analogue monitors. Both are graphs of the same movement.



Trace AXIS#0 []

FIGURE 6. Trace

2.- AUTOTUNING

2.1. The purpose

The purpose of the autotuning is to calculate the inertia of the load. The inertia is a characteristic of a mechanical system that depends only of the geometry and the material and is defined as the resistance of a mechanical system to change its rotational constant speed. In fact, the inertia of the system is telling you how much torque do you need to accelerate or decelerate the load.

2.2. The block diagram

This is a simplified block diagram of the Smart Step from the point it receives the position command up to it gives the torque to the motor:

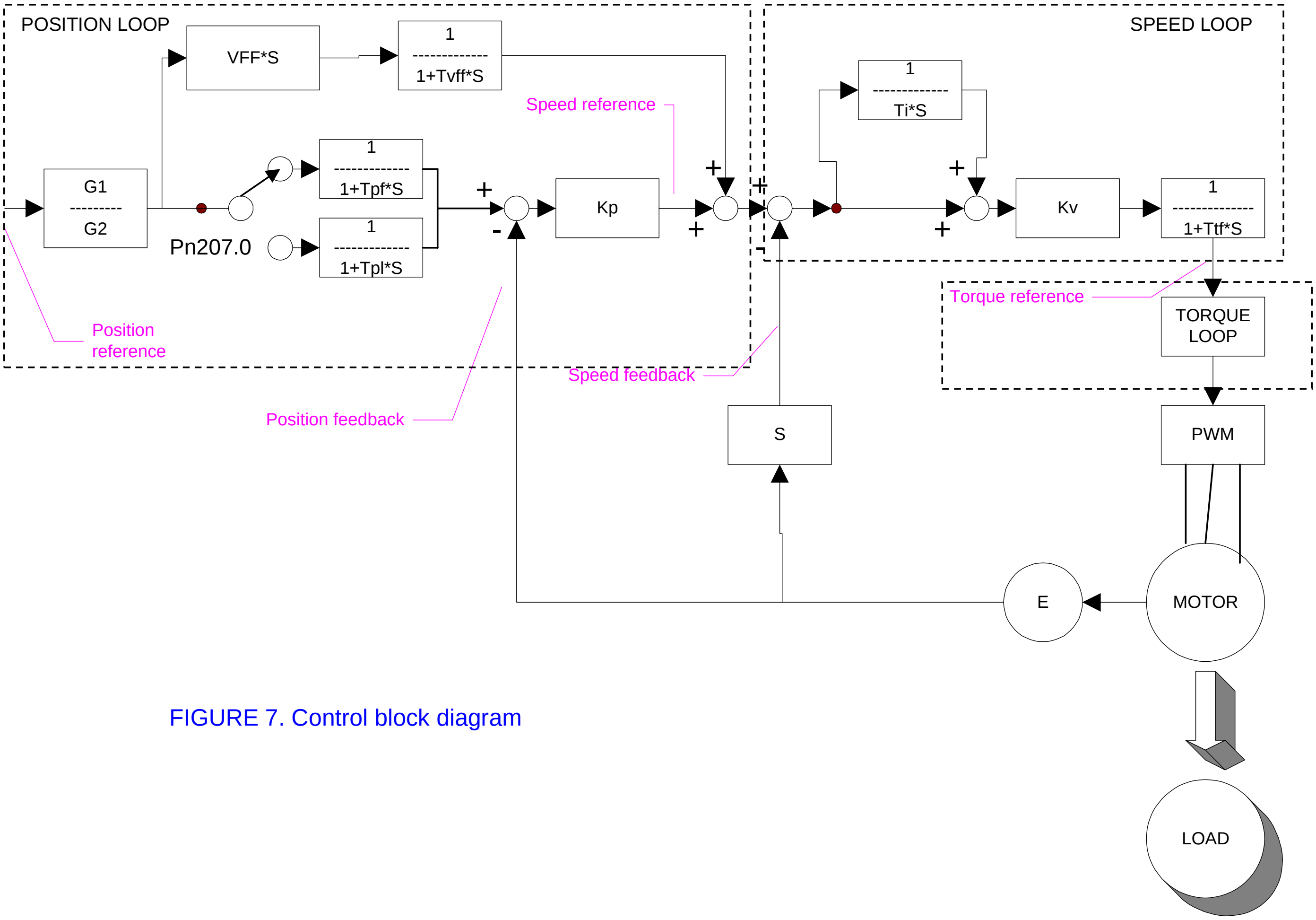


FIGURE 7. Control block diagram

Where:

G1 = Pn202	Gear Ratio numerator.
G2 = Pn203	Gear Ratio denominator
Tpf = Pn204 * 10^{-5}	Acceleration filter for the position reference (seconds)
Tpl = Pn208 * 10^{-5}	Linear acceleration for the position reference (seconds)
Kp = Pn102	Position loop gain (1/seconds)
Ti = Pn101 * 10^{-5}	Integral time (seconds)

$Kv = Pn100 * J * 2\pi$ Gain of the velocity loop

Where $J = J_m + J_l = J_m*(1+Pn103/100)$ or $J_m*(1+Fn007/100)$

$VFF = Pn109 * 10^{-2}$ Feedforward gain

$Tvff = Pn10A * 10^{-5}$ Feedforward filter time constant

The pulse (position) command is processed (filtered and scaled) and entered to the position loop. The output of the position loop is the speed reference, that goes to the input of the speed loop. Note that the speed is the derivative of the position.

The output of the speed loop is the desired acceleration (the derivative of the speed) but this value is the input to the torque loop. The relation between acceleration and torque is given by the Inertia (J) following the next formula:

$$T = J \cdot \alpha$$

T = Torque reference

J = Inertia of the system: Inertia of the motor (Jm) + Inertia of the load (Jl)

α = Desired angular acceleration

That's why to know the inertia of the system is so important. If the Inertia is the right one, when the output of the speed loop decides that the right acceleration is α_0 the output torque will produce the desired acceleration; then, the system behaviour will be as expected and the system will be stable.

The multiplication for the inertia of the system is an internal multiplication that is transparent to the user.

This value of the load inertia can be either Pn103 or Fn007 according to the next rule:

Pn110.0=2 Autotuning disabled.

- Pn103 is used. Fn007 is updated but this value has no effect.

Pn110.0=1 Online autotuning.

- Pn103 is used after power on as load inertia value.
- After the first calculation of Fn007, this value is used as load inertia value.
- Every time Fn007 is updated, the new value is used as load inertia.

Pn110.0=0 Autotuning at initial operation.

- Pn103 is used after power on as load inertia value.
- After the first calculation of Fn007, this value is used as load inertia value.
- The internal value of load inertia is not updated again with the new values of Fn007. It is not possible to monitor the internal value used as load inertia in this case..

Note that, in each option, the load inertia is always calculated in Fn007, the difference is the way this value is used. In all cases you can manually force to write the contents of Fn007 into Pn103.

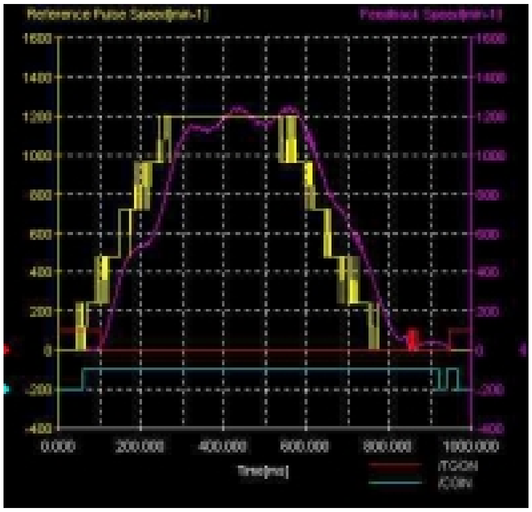
2.3. Limitation of the autotuning.

The firmware in the Smart Step servodriver uses an internal mechanical model to calculate the inertia of the system, but this model only works properly under certain conditions. If you don't match any of the conditions the results will not be reliable:

- The maximum speed is higher than 500 rpm.
- The peak torque is higher than 50% of the rated torque.
- The load inertia does not change faster than 200 ms.
- The motor doesn’t reach the torque limit of the servomotor (300% of the rated torque).
- The set rigidity is not very low (bigger than 1).
- The friction is not very high (less than 50% of nominal torque).
- There is not a force always applied (Gravity in vertical axes).

2.4. Procedure in easy mode (Using dip switch).

- Set the autotuning switch to ON.
- Apply a pulse reference in that way that you match all the requirements for a valid autotuning.



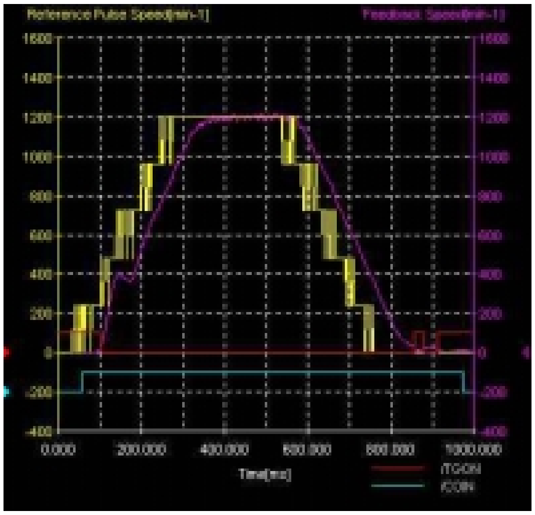
This is the response to a pulse signal of a mechanical system to a pulse train with acceleration and deceleration ramps (in yellow there is the pulse speed and in purple the feedback speed) during the first movement, that means, the load inertia has not been calculated yet.

The rigidity is set to 4 and the initial load inertia is 0.

As we can see, the performance is very poor.

FIGURE 8.

- After some movements the load inertia has been calculated and, if your load inertia is not going to change, you can turn OFF the autotuning switch. At this moment the calculated inertia is stored permanently in the EEPROM and is never calculated again.
- If your load inertia is going to change by an important amount but always inside the limits of the autotuning you can leave the autotuning switch ON all the time.



And this is the response to the same reference after two movements. The load inertia has been calculated and the performance clearly improved.

It is possible to find a discontinuity during the acceleration. This is due to the correction when the load inertia has been calculated and the value is effective.

FIGURE 9.

2.5. Procedure in advanced mode (Using parameters).

In the advanced mode we have two different ways of making the autotuning, selected by Pn110.0 The possibilities are:

=0 Autotuning is calculated once after power ON.
=1 Autotuning is calculated continuously.
=2 Autotuning is not performed.

The autotuning switch =OFF is equivalent to Pn110.0=2 and =ON is equivalent to Pn110.0=1. No equivalence for Pn110.0=0

The procedure is same than for the easy mode:

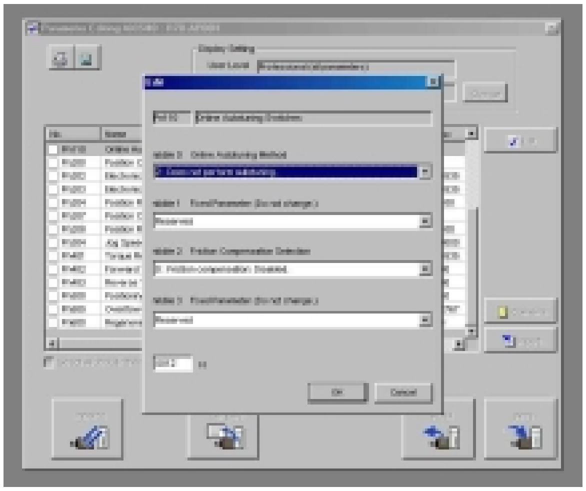

Display Setting
Online Autotuning Method
Fixed Parameter (Do not change.)
Reserved
Friction Compensation Detection
Fixed Parameter (Do not change.)
Reserved
OK
Cancel

FIGURE 10. Setting autotuning

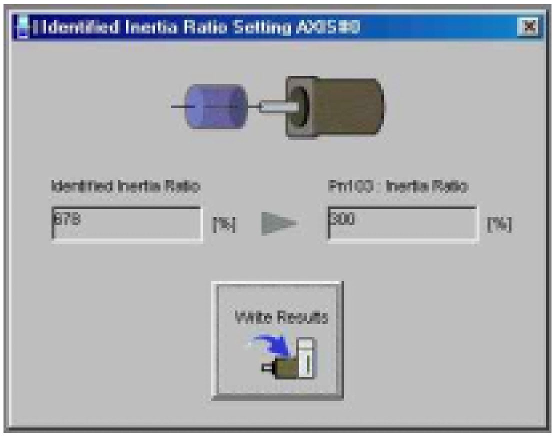
This is the autotuning parameter in the WMON WIN.

If you set the friction compensation (Pn110.2) according to the system the accuracy of the inertia calculation improves.

For friction < 10% of rated torque, no compensation, for friction between 10% and 30% set compensation small and for friction between 30% and 50% set compensation big.

Not accurate results will be obtained with frictions bigger than 50%

- Set the Pn110.0 to 0 or 1, as desired.
- Apply a valid pulse reference.
- Store the result of the autotuning using the software or the Parameter Unit.



Identified Inertia Ratio Setting AXIS#0

Identified Inertia Ratio: 678 [%]

Pn103 : Inertia Ratio: 300 [%]

FIGURE 11. Forcing inertia write to Pn103

This is the process to store the autotuning result using WMON WIN if you are working with the parameters. The identified inertia ratio is the value that the servo is using if the autotuning is enabled.

Pn103 is the value that is used if the autotuning is not enabled and the first time after power on if it is enabled (until the first inertia calculation is performed).

Storing the result has the same effect that turning ON → OFF the autotuning switch in easy mode

- Decide if you disable the autotuning or not.

NOTE 1: It is interesting to store the result of the autotuning even if you are going to leave it active all the time. Storing the result means to write the calculated value in Fn007 into Pn103. When no inertia calculation have been done yet (or the autotuning is disabled) the system is working with parameter Pn103; storing the autotuning means that you will start controlling the motor using a valid value and you avoid the transient behaviour shown in Figure 1.

This note is valid for the easy mode too but the parameter number is transparent to the user.

NOTE 2: Even if the autotuning is disabled you can see the value of ‘identified inertia Ratio (Fn007) changing some other time. That means that the load inertia calculation is done all the time but if the autotuning is disabled the calculated value is not applied and the servodriver is working with the value in Pn103.

The graphs in advanced mode are similar that the ones in easy mode.

3.- GAIN ADJUSTMENT

3.1. Introduction

Once the autotuning has been successfully done, it is time to set the gains of the system. Note that, if the inertia of the system has been calculated properly, the dynamic of the system will be defined only by the gains.

That means that, for a certain gain, the dynamic will be always same independently of the load. Eg:The motor speed profile for a certain reference will be same for a motor with no load that for a motor with a load of ten times the motor inertia if, in both cases, the gains are same.

Of course, this is valid provided the system matchs the requirements for a proper autotuning.

This fact is very good because it allows to separate the: ‘Load modelling’, adjusted with the inertia, and the ‘Dynamic adjustment’ set with the gains. In a very short time, the application engineer can create some kind of experience that shorts the commisioning time.

The name of ‘Rigidity’ is not casual, it makes reference to the Rigidity (in the mechanical meaning of the word) of the mechanical system attached to the motor axis. A system that has a big backlash (pinions) or is elastic (trapezoidal belt) tends to be unstable and does not accept high gains. On the contrary, the friction tends to stabilise the system. If you are familiar with electronics, this is quite equivalent to have a resonant R-L-C system.

On the contrary, a system that has a very rigid coupling usually accepts very high gains and, then, very high dynamics.

3.1. Easy mode (Using dip switch).

In easy mode the gains are adjusted just using the rotary switch. This adjustment is called ‘Rigidity’ and defines how high are the gains and so how close the motor follows a reference. If the setting is too high the system can become unstable.

The value of 1 means the smaller gains and A means the highest gain. Values from B to F are the same than A.

Follow the next guidelines to set the rotary switch:

SETTING	RIGIDITY	APPLICATION
0	Advanced mode	----
1	VERY LOW	Articulated robots, chain drives, belt drives, rack and pinion drives,
2	LOW	
3		
4	MEDIUM	X,Y tables, cartesian coordinate robots, general purpose machinery.
5	HIGH	Ball screws, feeders and, in general all couplings directly to the motor axis.
6		
7		
8		
9	VERY HIGH	
A to F		

The method of adjustment is very easy: Set the rotary switch according the above guidelines and apply to the servodriver the desired pulse reference. In case of doubts it is better to set a step less (lower rigidity).

See the response to the reference, if the response is too slow or the position error is too high increase the setting of the rotary switch. On the other side, if the response is vibrating or unstable or the motor becomes noisy, reduce the setting of the rotary switch.

In this figure we can see the pulse command speed (yellow) and the feedback speed (purple) with the rotary switch set to 3 (low value).

All the 'corners' are rounded and there is a big delay in the following of the reference.

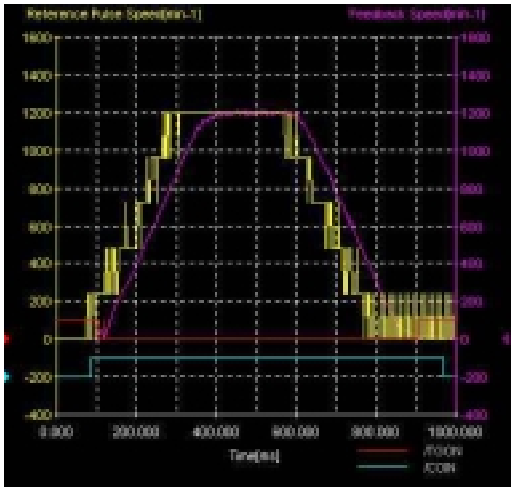
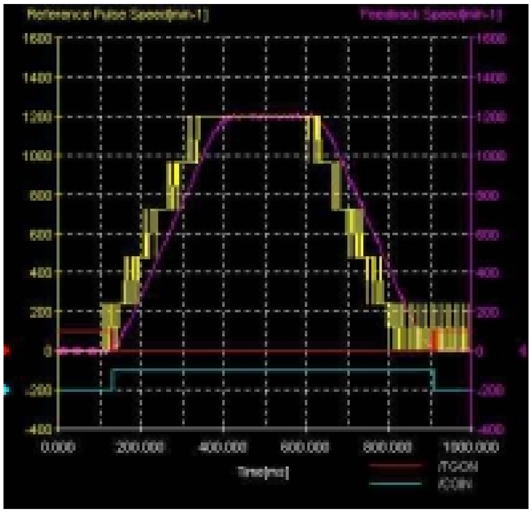


FIGURE 12.



In this figure the rotary switch is set to 5. The corners are not so rounded and the motor is following more closely to the reference.

FIGURE 13.

3.2. Advanced mode (Using parameters).

3.2.1. PI + Filter adjustments.

To allow the adjustment in advanced mode, you have to put the rotary switch to zero position, in this case the rigidity is set via Fn001. Rotary switch set to ‘X’ or Rotary switch=0 and Fn001=X have both the same effect. What the rigidity setting is doing is changing the different gains following the next table.

RIGIDITY	Pn100	Pn101	Pn102	Pn401
1	15	15	4000	250
2	20	20	3500	200
3	30	30	3000	150
4	40	40	2000	100
5	60	60	1500	70
6	85	85	1000	50
7	120	120	800	30
8	160	160	600	20
9	200	200	500	15
A to F	250	250	400	10

In the easy mode, this change is done internally in a transparent way to the user.
The function Fn007 can be changed using the Parameter Unit, but not the WMON WIN II. For the adjustment with Fn007, follow the same process than in easy mode.

Sometimes the adjustments just by the rotary switch or Fn007 is not good enough and it is necessary to modify manually each individual parameter to obtain the optimum performance. See in the manual the description of each parameter and situate it in the figure 7 for better understanding.

3.2.2. Feedforward adjustment.

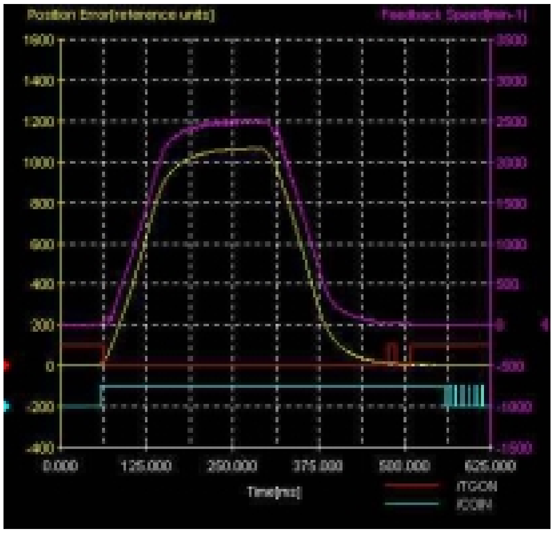
Regarding the position loop, there are two additional parameters that can be useful in order to obtain a close following: The Position feedforward gain (Pn109) and the feedforward command filter (Pn10A).

In a system with only proportional position gain you have a position (following) error that is proportional to the speed of the motor. If you increase the position loop gain, you reduce the position error in proportion but, of course, this is not always the best solution as you can not increase indefinitely the position loop gain because you will arrive to instability. The solution is to use the Position feedforward gain and filter (Pn109 and Pn10A).

If we are interested only in arriving to the target position and what happens during positioning is not of vital importance, then, those gains are not important for the application (for example, a cut-to-length application). But, if what happen in the middle is important and we want a close following during all the positioning (for example, a synchronisation with a master encoder), it is important to set correctly those gains.

The position command is differenciated then we will obtain the speed command profile. This speed command profile is added to the internal speed reference after being filtered and scaled. In the ‘perfect world’, I mean, supposing that there are not disturbances and that the reference is a good one, if Pn109=100% and Pn10A=0, the speed profile added is the best one and the position error is zero independently of the speed. In this case we will have position error just during acceleration and deceleration.

But we are not in the ‘Perfect world’ then, sometimes it necessary to filter the commanded speed to obtain a smooth profile and sometimes it is not possible a full compensation (Pn109=100) because the system becomes unstable.

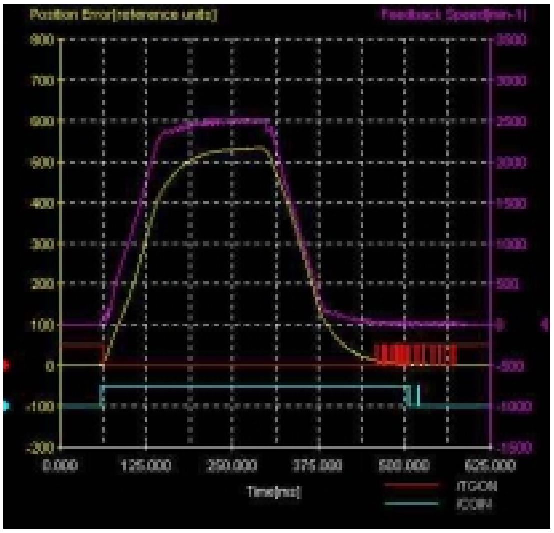


Graph of the feedback speed (purple) and the position error (yellow) with:

Pn109=0
Pn10A=0

We can see clearly that the error is proportional to the speed. A system with just proportional gain has this typical 'queue' at the final approach (slow final positioning) because if the position error is small, then, the speed reference is small too. Increasing manually Pn102 we can reduce this final approach.

FIGURE 14.

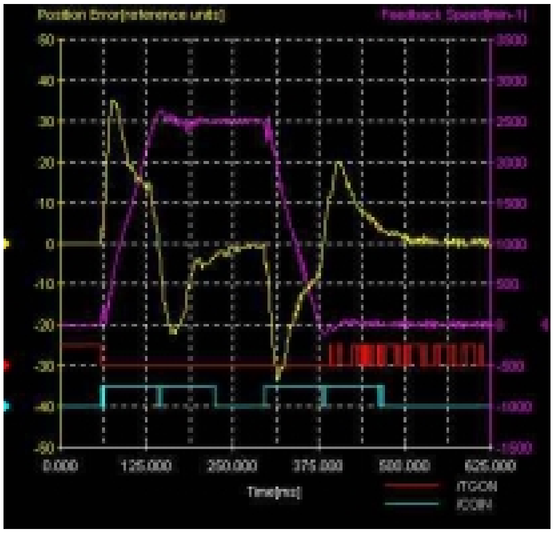


Graph of the feedback speed and the position error with:

Pn109=50
Pn10A=0

We compensate the 50%, then, the position error is half the value than without compensation. The rest of the gains are same than in the previous graph.

FIGURE 15.



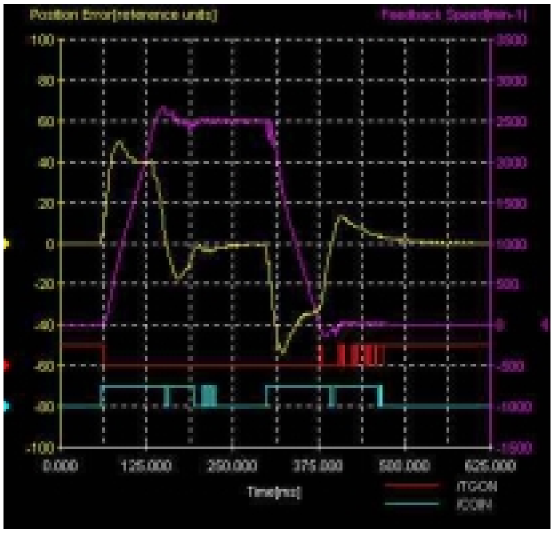
Graph of the feedback speed and the position error with:

Pn109=100
Pn10A=0

We compensate 100%, then, the position error is zero when we reach stationary state. We have position error just during acceleration and deceleration but, even there, the error is small.

As we are in the 'real world' we start seeing signs of instability.

FIGURE 16.



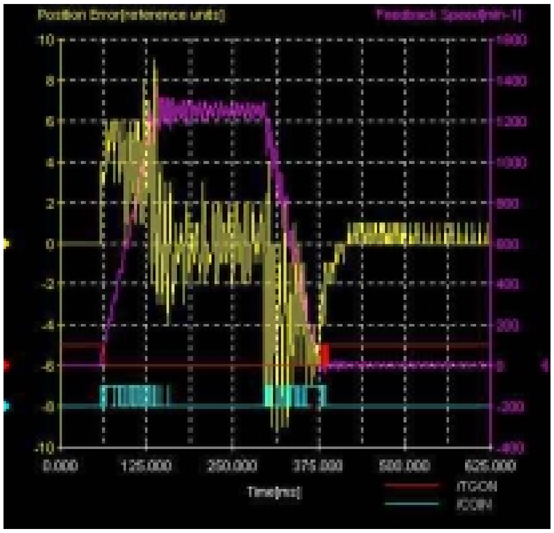
Graph of the feedback speed and the position error with:

Pn109=100
Pn10A=200

We compensate 100%, The filter reduces the disturbances and so the position error is reduced and is smaller except in the acceleration/deceleration zone. The positioning time is shortened too and there is only a small 'overshoot' at the end of the acceleration, but not in the deceleration.

FIGURE 17.

The effect of the filtering is better shown in the next graphs that shows similar adjustments but with a rigid load in the motor axis. In this case the rigidity is set to 6.

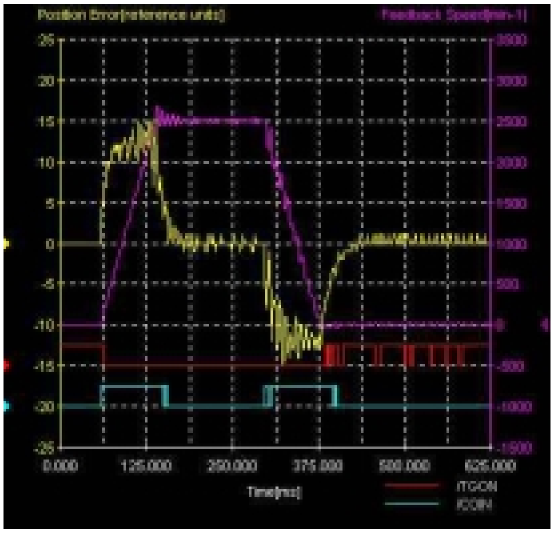


Graph of the feedback speed and the position error with:

Pn109=100
Pn10A=0

We compensate 100%, The Position error is very small and is limited to acceleration and deceleration but very clearly we see a disturbance.

FIGURE 18.



Graph of the feedback speed and the position error with:

Pn109=100
Pn10A=200

We compensate 100%, The filter reduces the disturbances and so the position error is reduced and is smaller except in the acceleration/deceleration zone. The positioning time is shortened too and there is only a small 'overshoot' at the end of the acceleration, but not in the deceleration.

FIGURE 19.

4.- REFERENCE ADJUSTMENT

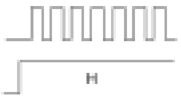
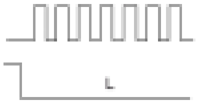
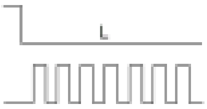
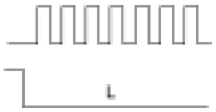
4.1. Introduction

Sometimes the pulse reference that we receive is not the most suitable one and you need to adapt this pulse reference to match the controller and application requirements. The Smart Step servodriver has certain facilities for doing this.

4.2. Easy mode

a) Pulse mode adjustment:

In easy mode you can select between two different kind of commanded pulses, depending on the position of the Switch 3. See the next table for setting.

SW 3	COMMAND	INPUT PIN	FORWARD	REVERSE
1	PLS/SIG	1:+PULS 2:-PULS 3:+SIGN 4:-SIGN		
0	CW/CCW	1:+CW 2:-CW 3:+CCW 4:-CCW		

b) Resolution adjustment:

You can select between four different resolutions with the switches 4 and 5 as follows:

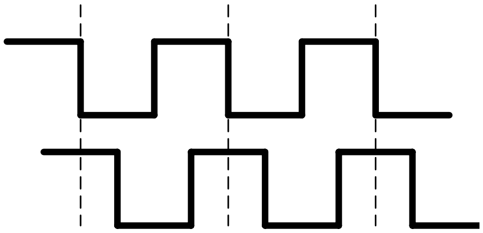
SW 4	SW5	RESOLUTION
X1	1000 P/R	1000 pulses per motor revolution
X1	500 P/R	500 pulses per motor revolution
X10	1000 P/R	10000 pulses per motor revolution
X10	500 P/R	5000 pulses per motor revolution

Note that the motor encoder is 2000 pulses per revolution, that means that we can have 8000 counts per revolution. If we select the resolution of 10000 pulses per revolution we may have a certain error. Nevertheless, this error is not accumulated.

4.3. Advanced mode

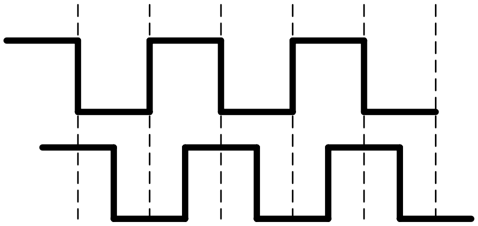
a) Pulse mode adjustment:

The adjustment is done with the parameter 200.0. Check the manual I533-E1-01 (page 2-24) for details. In this case you have more possibilities, as the quadrature pulses (like the signal that comes from a line driver quadrature encoder). In case of quadrature pulses, you can select the resolution X1, X2 or X4 as follows:



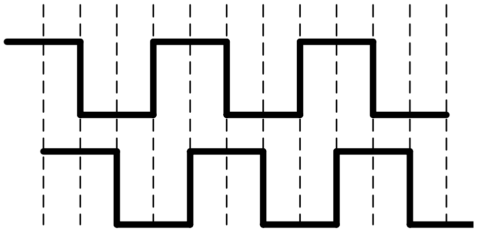
X1 Scaling. Every complete quadrature cycle is taken as a Reference Pulse.

FIGURE 20.



X2 Scaling. Rising edge and falling edge of one of the quadrature signals marks the different pulses.

FIGURE 21



X4 Scaling. This is the maximum resolution that we can obtain with a quadrature signal. Every edge in each signal determine a different Reference pulse. This encoder pulses X4 are called ‘edges’ or ‘counts’.

FIGURE 22

b) Resolution adjustment.

The resolution adjustment is done via two parameters, numerator (Pn202) and denominator (Pn203), that helps to adjust the Units of the commanded pulse. Internally, the servodriver is working always using the Motor encoder pulses X4; the encoder is 2000 pulses per revolution then, the resolution is 8000 counts per revolution.

An example will help to understand better the setting of those parameters:

We are going to move a ball screw with a pitch of 5 mm with a servomotor trough a gear reduction of 1:3. We want a resolution of 1μm (this resolution is very typical in machine tool application).

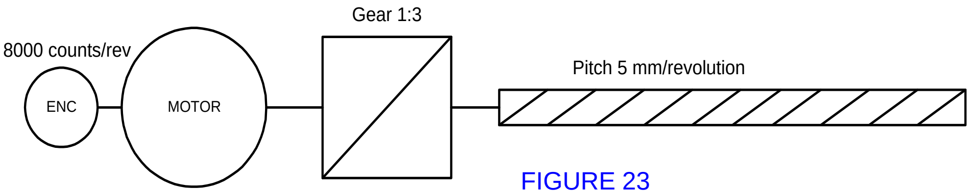


FIGURE 23

The process to obtain the right gear ratio is next:

$$1\mu m \cdot \frac{1mm}{1000\mu m} \cdot \frac{1rev\_screw}{5mm} \cdot \frac{3rev\_motor}{1rev\_screw} \cdot \frac{8000encoder\_counts}{1rev\_motor}$$

$$1\mu m = \frac{24000}{5000} encoder\_counts = \frac{24}{5} encoder\_counts$$

It is quite evident now to set:

Pn202=24
Pn203=5

It is very important to note that, even if you send a number of pulses that gives a fractional part after the gearing, the fractional part is not lost but accumulated in order of not accumulating any error.

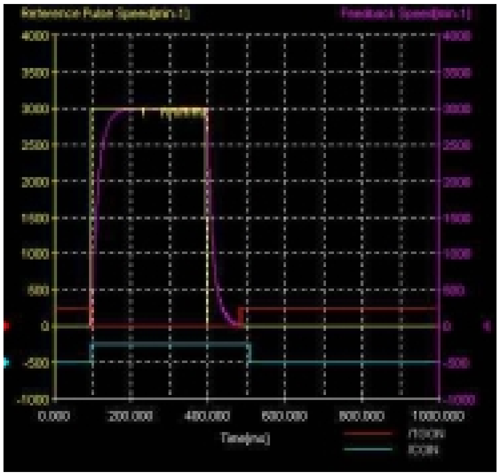
In the previous example, if you send 11 pulses, the result is 11*24/5=52.8. The servomotor will turn only 52 encoder counts and the 0.8 is stored. If a second movement of 31 command pulses is sent: 31*24/5+0.8=148.8+0.8=149.6, the motor will move 149 counts and 0.6 is stored.

A gear ratio outside the next range can give unstable behaviour: 0.01>Pn202/Pn203>100. Especially at slow speeds it is likely to have oscillation.

c) Pulse command filtering.

Sometimes the controller that is giving the pulse command reference is not very powerful and can only generate a pulse train with a certain frequency and with a number of pulses but can not generate an acceleration and deceleration ramp.

Of course, the motor can not follow a sudden change of frequency. This will create a very sharp movement with a very high current peak that is not usually desired. The bad solution is to put a 'Soft' adjustment so you will have a big following error that will soften the movement. The good solution is to filter the command in order to create a command pulse train that the motor can follow and then adjust the gains to follow this 'modified' pulse command.



This is the resultant waveform if we command a pulse train without acceleration and deceleration speed ramps (yellow).

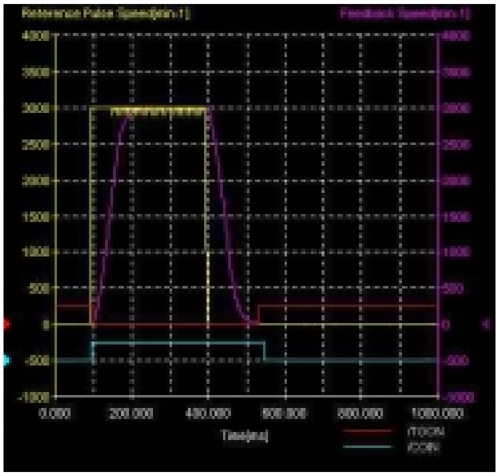
The motor try to accelerate as fast as it can. that means a following error and a sudden acceleration.

In purple there is the feedback speed.

FIGURE 24

We have two ways to filter the signal:

- Linear filter: The effect is to put linear ramps to the pulse command input speed. Use this filter when the pulse generator gives no ramps and adjust the time depending on the dynamics of your application and the torque of the motor.



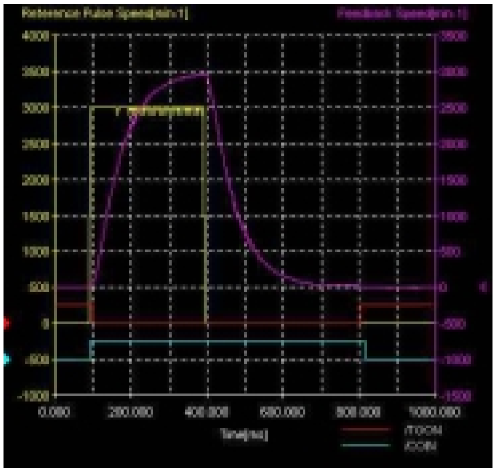
This is the resultant waveform when a linear acceleration/deceleration filter is applied.

Pn207.0=1
Pn208=6400

We generate a commanded speed profile that the motor can follow.

FIGURE 25

- First order filter: This is a standard exponential filter. It is very useful when the gear ratio is very big (near or above 100) or very small (near or below 0.01) or when the command pulse frequency is not stable.



This is the resultant waveform when a first order filter is applied.

Pn204=6400
Pn207.0=0

The normal usage for this filter is to smooth a command pulse train with an unstable frequency, not used normally to give acceleration and deceleration ramps.

FIGURE 26

NOTE: Do not use Feedforward gain (Pn109) together with those filter because the feedforward amount is taken before the filter and, then, we are mixing filtered and not filtered references (see figure 7). The result will be unexpected and unstable.