

## Tuning

### Setup

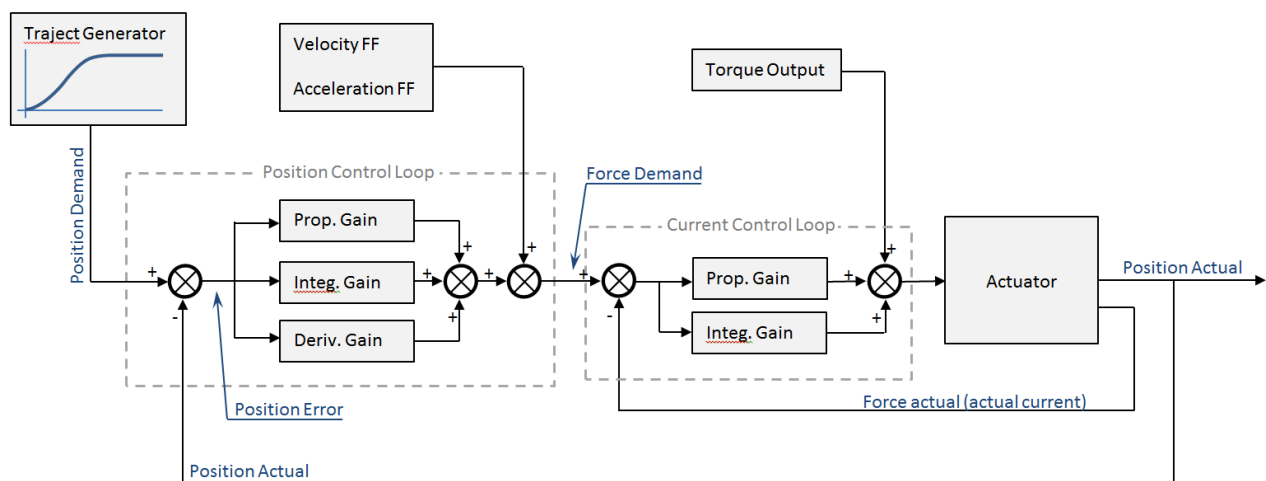
Before starting to tune, it is best to load a config file for your type of actuator into the controller. This can be done in the Tuning tab of the LCC Control Center software.

[Download config file](#)

The Tuning related parameters are displayed in this Tuning Tab as well as the responses on a move from A to B (Test Trajectory). Create a Test trajectory that is on the edge of what you expect from the actuator in your application.

Take care that you have done a (phase detect &) homing before starting the Tuning Test trajectory. You can program this in the Control Center Build Programs and Run Programs tab.

The parameters of the control loops and their locations are shown in the scheme below.



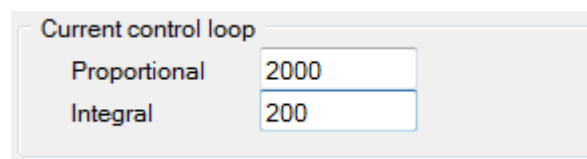
This scheme is given so you can look back on it in due time. It is not necessary to understand it fully at this moment. The scheme is a representation of how the parameters are related to each other.

Note that every parameter has a limit, for some parameters the limit value is in the config file. The integral contribution in the position loop is limited using the Integral limit. The current to the actuator is limited by several parameters: max force, max current, positive force limit and negative force limit.

## Tuning Current Loop

The first step in tuning is to check the current control loop. Select the force actual and force demand on the response plot of a movement.

The Proportional and Integral constants of the current loop are affected by the physical actuator parameters: Coil Resistance, Actuator Inductance, Supply Voltage and the dynamic bandwidth of the coil in the magnet. The Proportional and Integral gain can be adjusted in the current control loop section of the Tuning tab :



A screenshot of a software interface for tuning the current control loop. It features a title bar 'Current control loop' and two input fields. The first field is labeled 'Proportional' and contains the value '2000'. The second field is labeled 'Integral' and contains the value '200'.

Current control loop	
Proportional	2000
Integral	200

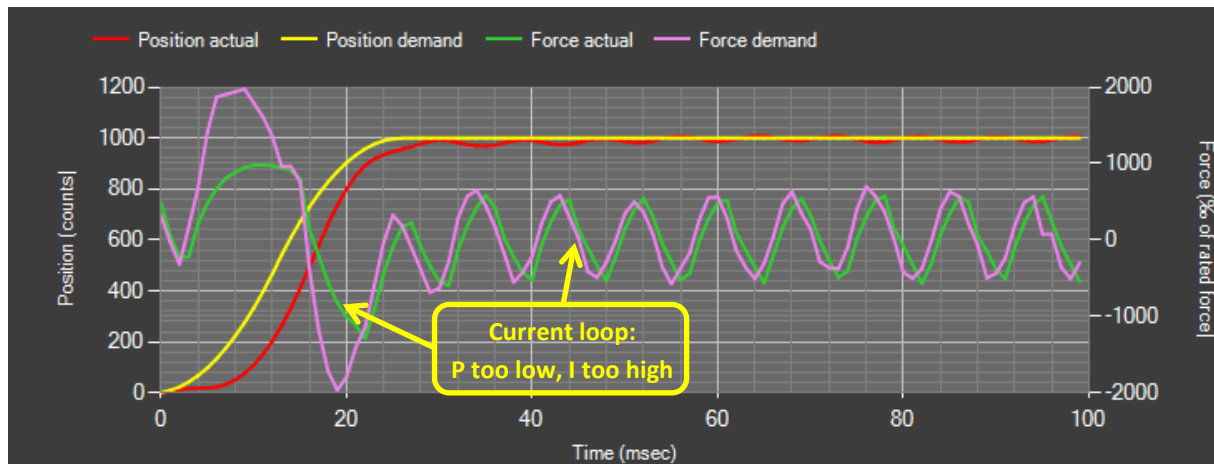
The config file is expected to have reasonable control values, not optimised to the situation but good to start with. Then you start the test and review the response plots. Look besides the Position actual and demand also at the Force actual and demand. If the responses of the Force actual and Force demand are the same or almost the same the current control loop is properly tuned.

Resonances in the current loop or limitations can cause poor performance of the actuator. There are several reasons for resonances in the current loop. There are also several reasons for limitations in the current control loop. The following section shows resonance and limitation examples and the possible solutions.

When changing PID parameters it is best to start with increasing parameters with factor of 2 each time or reduce to half the value to see the effect of it. If this doesn't result in the desired change do that up to about 5 times. If you get close to the desired response, you can use an iterative process of finding the best value.

## Current Resonance due to High Integral Gain

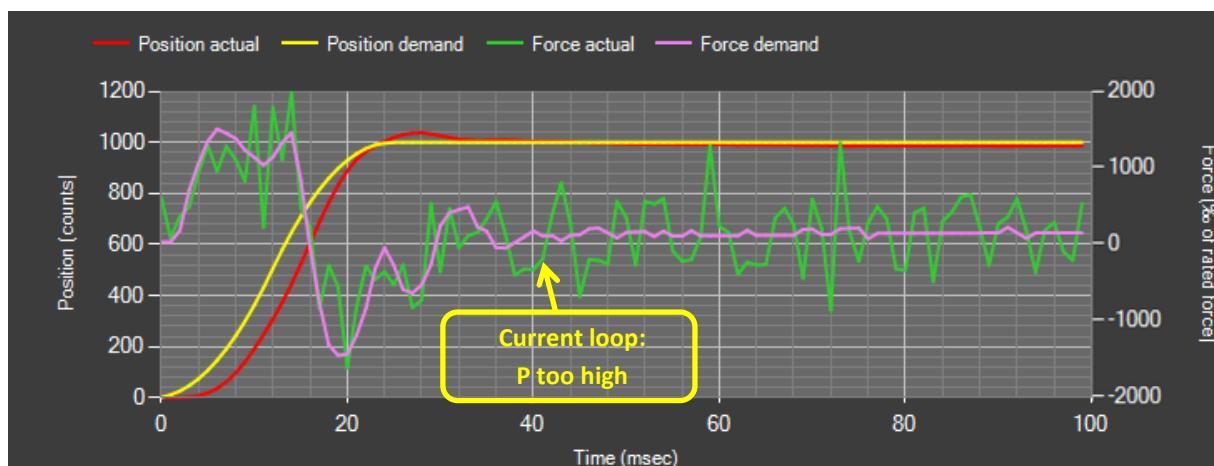
Current resonance can occur due to too high Integral gain in relation to the Proportional Gain. The resonance can be recognised by the poor response of the force actual to the force demand at the beginning of the move followed by a resonance of both force actual and force demand when position is reached. As a result the position will vibrate slightly.



The solution is to increase the Proportional gain of the current loop and/or lowering the integral gain of the current loop. This type of resonance disappears if you make the current integral gain 0. If that is not the case the resonance is caused by too high Derivative gain of the position loop !

## Current Resonance due to High Proportional Gain

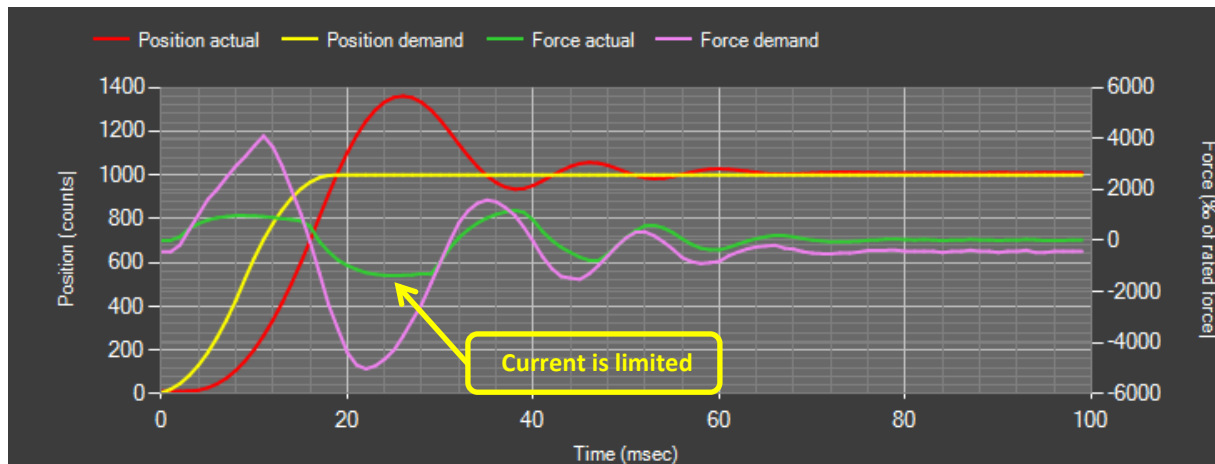
Current loop resonance can occur also due to too high Proportional gain of the current control loop. The resonance can be recognised by a noisy response (Force actual) around the Force demand. The position is hardly affected by this noise.



The solution is to decrease the current proportional gain.

## Current Limitation

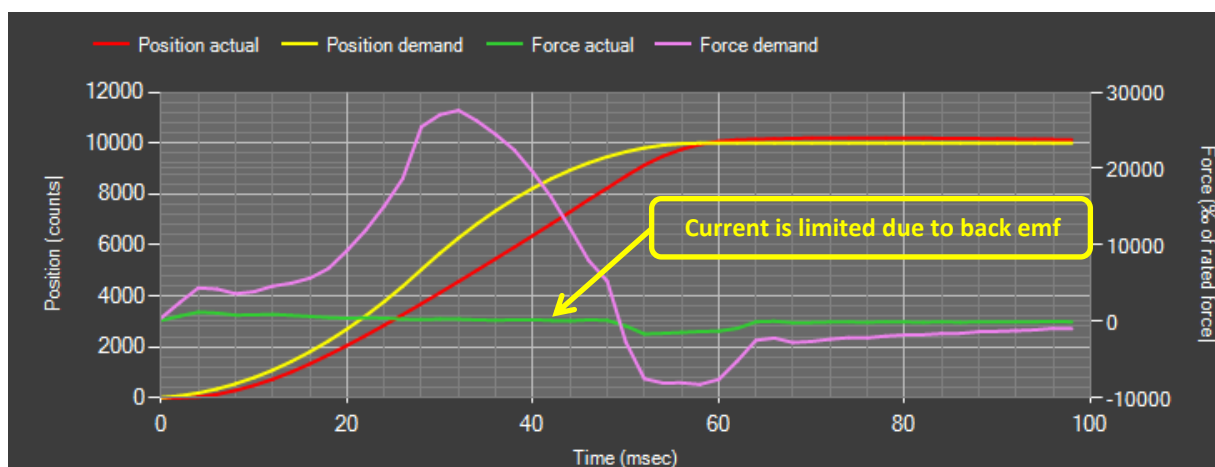
Current limitations can create poor dynamic response. The image below shows the response of a position move with a limited current. The limitation has the biggest effect on the acceleration and deceleration of the movement. The limit can be caused by either “max current limit” setting of the controller or a power supply limitation.



The solution is to take away the limiting factor or to decrease the acceleration and deceleration.

## Current clipping due to Back EMF

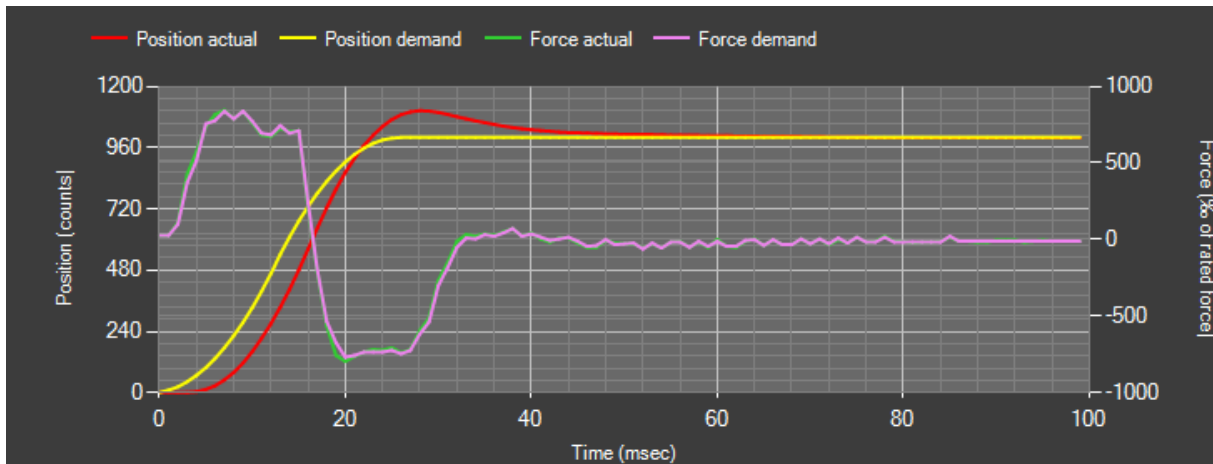
In the image below the response is shown of a motion that reaches the speed limits of the actuator. The actuator generates back EMF when moving. This back EMF (= generated voltage) is reducing the effective voltage over the coil and will therefore reduce the current.



This can be solved by using a higher voltage supply (48V is maximum for LCC10) or reduce the motion speed. Note that actuators with lower force constants have lower back EMF.

## Ideal Current Loop

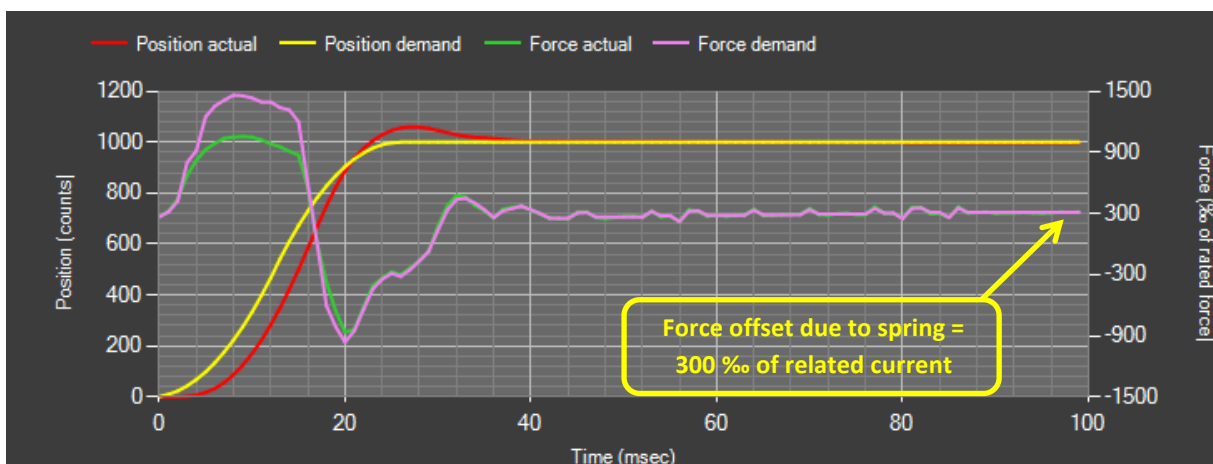
In the image below the response is shown of current control loop that is set with correct parameters.



In a properly tuned current control loop the Force actual and Force demand are close to identical

## Force offset

When the dynamic behaviour of the movement has finished the Force demand should be about zero. If that is not the case you can add Force offset. In the example below a spring was added to generate an offset of 300 % of motor rated current



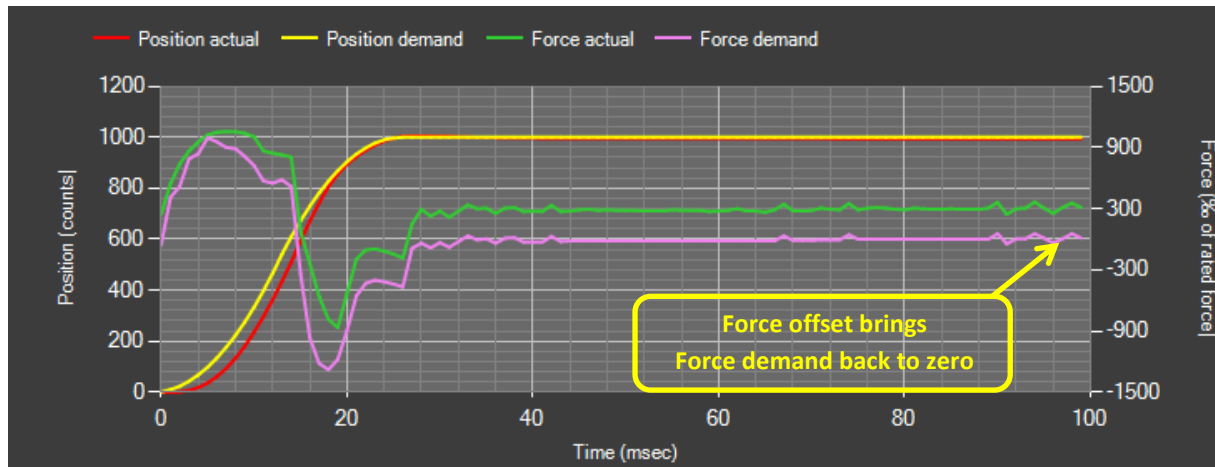
Force offset is an added current to the Target Current in the Current Control Loop.

When a constant force needs to be compensated, the force offset can generate a static force outside of the control loop. The static force can be caused by a spring or by gravity force of the moving mass of the actuator. The amount of Force offset you need to add is equal to the deviation from the Force demand to zero. In this case the output offset value is best set 300.

## Technical Note



The plot below shows the result of the output offset of 300.



## Tuning Position Loop

The second step in tuning is the position control loop. This loop is used for controlling the position in position and velocity moves. If you have loaded the config file, a default set of position control parameters is loaded for that actuator. The physical parameters that affect the position control loop values are force constant, moving mass, encoder resolution and the dynamic bandwidth of the coil/piston/guide/encoder assembly.

### Position Control Loop Parameters

Each position control loop parameter has a contribution to the Target Current (TC). The Target Current is called Force Demand in the Graphs. Note that Force, Torque and Current are the same in the control loops. The function and behaviour of each of the position loop parameters are:

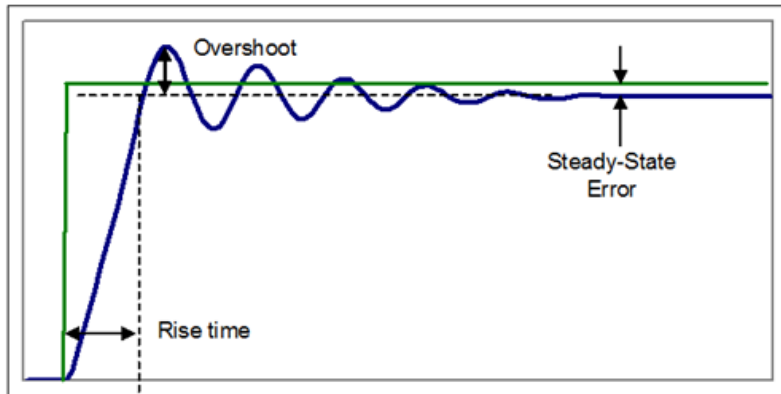
- Proportional gain :  $TC_P = \text{Position Error} * \text{Proportional gain constant}$ .  
This makes the actuator behave like a spring around the desired position (if the position error becomes bigger the generated current will become equally bigger)
- Integral gain :  $TC_I = \text{Cumulative Position Error} * \text{Integral gain constant}$ .  
It integrates the position error over time and multiplies this with integral gain value. If the position error remains a certain value its generated current will increase trying to push the actuator to the desired direction.
- Derivative gain :  $TC_D = \text{Position Error change} * \text{Derivative gain constant}$ .  
It takes the changes in position error and multiplies it with the derivative gain value. The generated current works against changes in position error and has a damping effect on the motion.
- Velocity ff:  $TC_{VFF} = \text{Target Velocity} * \text{Velocity Feed Forward value}$ .  
The feed forward for velocity generates a force proportional to the velocity to help overcome constant velocity dependent forces (friction). Normally not used for the low friction actuators of SMAC.
- Acceleration ff :  $TC_{AFF} = \text{Target Acceleration} * \text{Acceleration Feed Forward value}$ .  
Feed forward for acceleration generates a force proportional to the acceleration and deceleration to help overcome constant acceleration dependent forces (moving Mass). It is most effective in case of heavy payload.
- Integral limit : Limits the  $TC_I$  to a maximum value  
The integral gain can integrate indefinitely resulting in a very high target force. In order to limit this indefinitely high number the integral limit can limit its value.

The sum of the TC's is the output of the position loop (=target current).

# Technical Note

## General Position Loop Tuning Rules

The theoretical position response on a step in target position provides a good guideline for tuning a motion. The image below shows the response (in blue) on a step response of the target position (in green).



3 characteristics of the response are important: Rise time, Overshoot and Steady State Error

**Rise Time** determines the response time on a motion and is important for fast movements. In general if the rise time needs to be as short as possible, it will be more difficult to achieve a low overshoot and a low steady state error.

To reduce the Rise Time :

- Increase Proportional Gain
- Decrease Derivative Gain + Integral Gain

**Overshoot** of a step response is higher than that of a trajectory generated curve. If it becomes too much the chance of oscillating become bigger (= not a robust control loop). If it is low, the control loop has a damped behaviour. In that case it is wise to check the current loop behaviour for too much noise or vibration in the current loop (due to too high derivative gain)

To reduce the Overshoot :

- Increase Derivative Gain
- Decrease Proportional Gain + Integral Gain

Note that for a trajectory generated curve, reducing deceleration helps to reduce overshoot also.

**Steady State error** is the following error after the move is completed and dynamic behaviour of the control loop is faded out. The use of the position control loop integral gain and the force offset are the major parameters to get the steady state error low.

To reduce the Steady State Error :

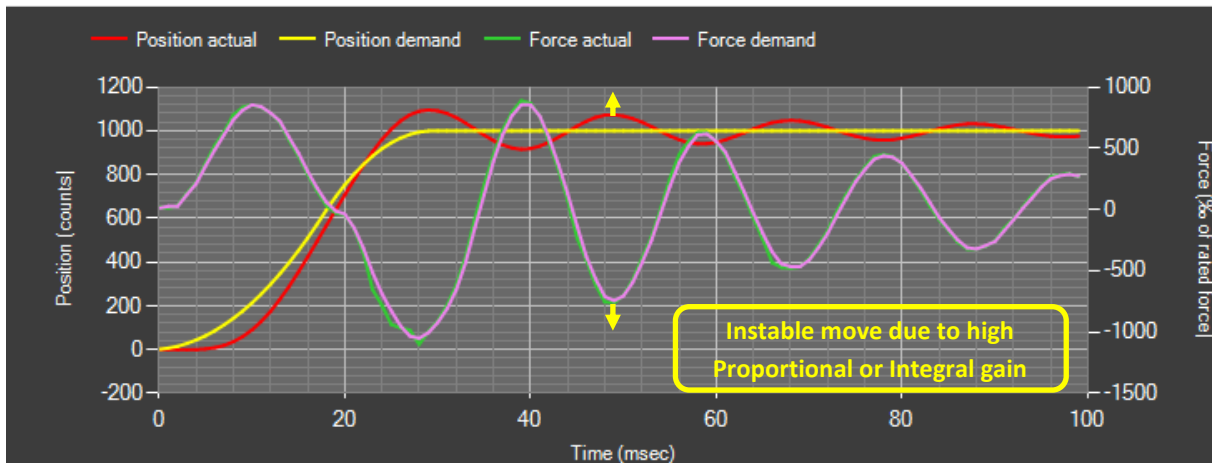
- Add Force offset
- Increase Integral Gain
- Increase Proportional Gain (but this has less effect than increasing integral gain)

**General:** Note that if the torque offset and acceleration feed forward could be used to optimise the behaviour.

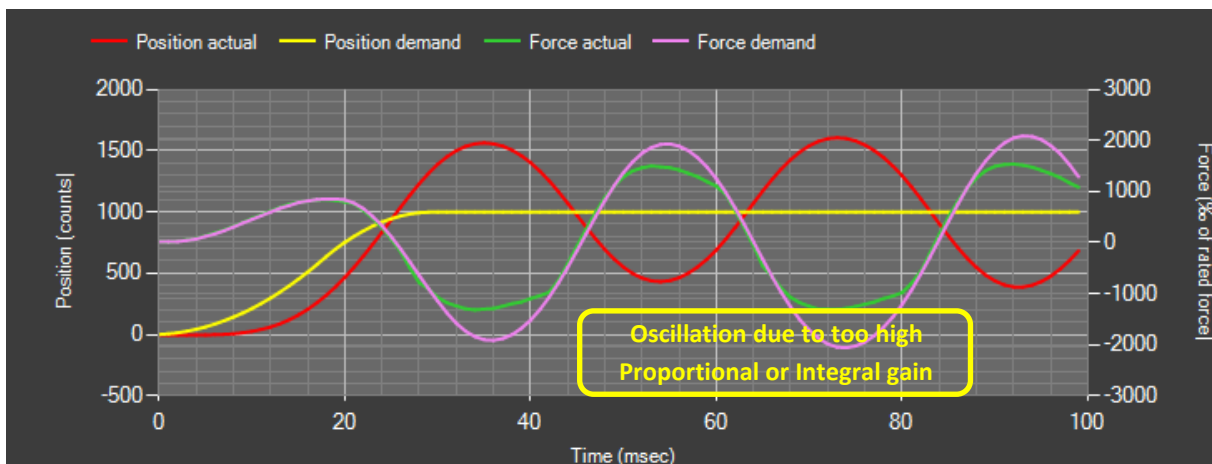


## Position Vibration due to too high Proportional or Integral Gain

As soon as a position control loop becomes close to the point of oscillation the signals look like the image shown below. You can see that the position vibration is opposite to the resonance in current.



When you increase the proportional or integral gain even more, you get an oscillation like shown below. The Force actual is not able to follow the Force demand. Due to the limit of this actual current the power of the vibration is limited.

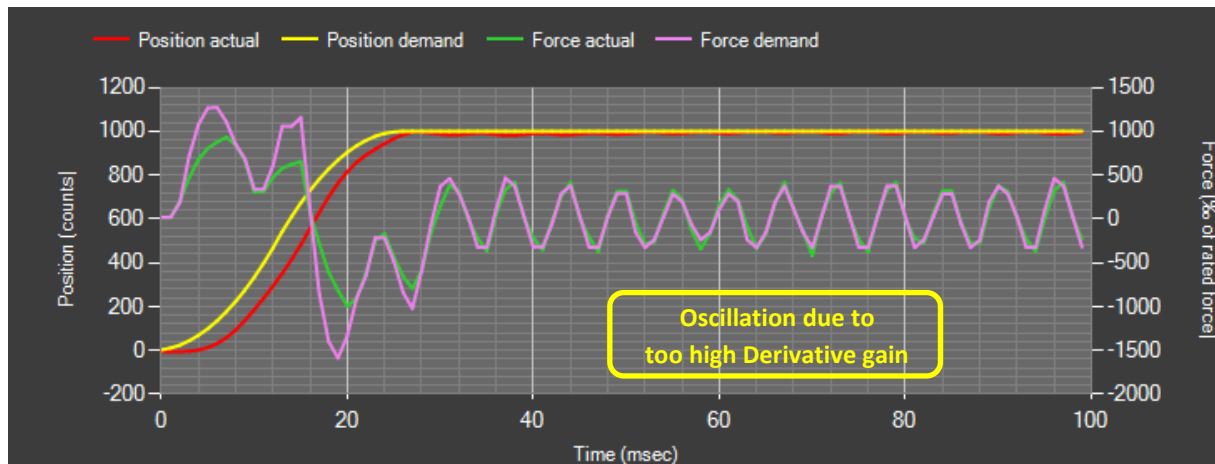


The solution to an oscillation like this is to:

- Increase Derivative gain
- Decrease Proportional gain + Integral gain

### Position Vibration due to too high Derivative Gain

A position oscillation due to a too high derivative gain will be relatively small compared to the resonance of the current. The image below shows how this can look in the response plot. It looks the same as the “*Current Resonance due to High Integral Gain*” (**page 2**) and it actually behaves the same also.



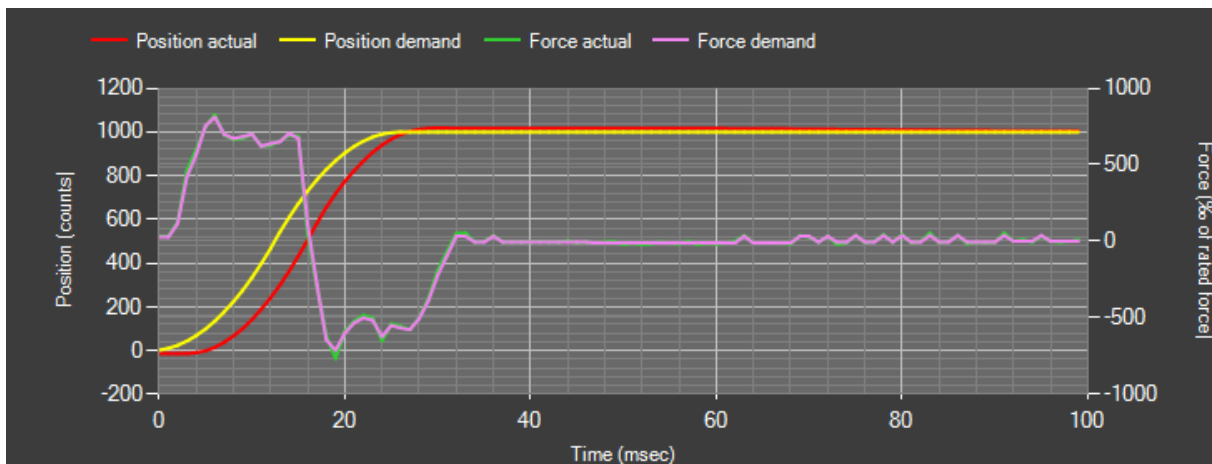
This vibration is best eliminated by reducing the position control loop Derivative Gain (or reducing the current control loop integral gain).

## Ideal Position Loop Tuning

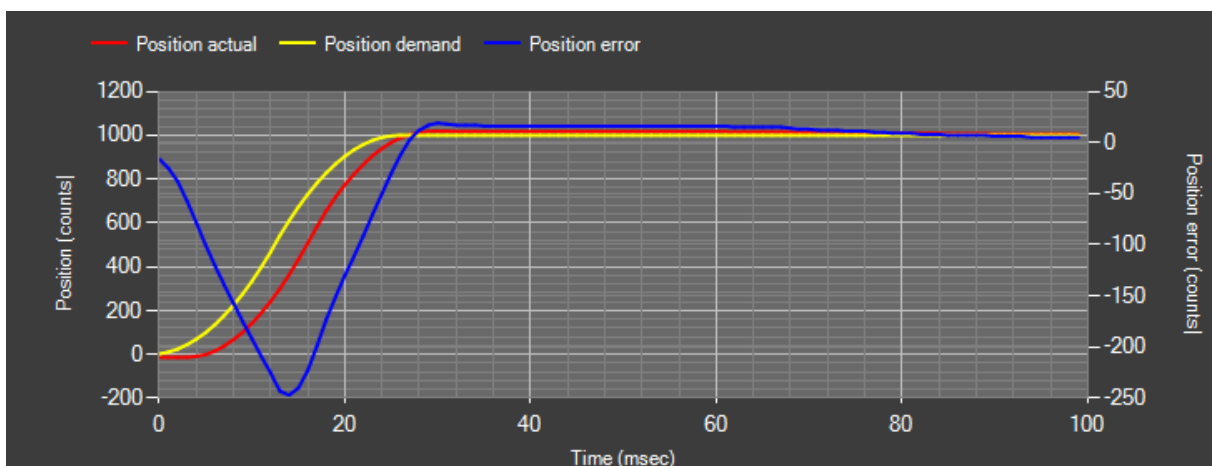
There is no such thing as “THE” ideal position loop for all applications. Each application has a different set of requirements that provide you with different boundaries for the motion like:

- As fast as possible, overshoot is not important (like bottle eject applications)
- No overshoot allowed in any case (motion application in strict dimensional boundaries)
- No or minimal position error at the end of the movement (positioning applications)
- Holding speed as constant as possible (scanning applications)

A correctly tuned position and current loop is shown in the response below.



The position error as it is shown below on the right axis provides you with the information on how close it is to the target position.



This response was achieved by only using current loop PI and position loop PID tuning including Integration limit. Be aware that that not all actuators and applications are able to achieve the behaviour as shown above. The use of Feed forward acceleration and output offset can help you get as close as possible to your requirements. If that isn't good enough you might need to lower the deceleration also.

### Robust control

When you have a properly tuned control loop, it is recommended to investigate the robustness of the system. This can be done by testing a variation to the proportional, integral and derivative gain. This way you can determine if the control loop will be robust against small changes (like friction, moving mass, external forces, temperature). In general 30 % change of the parameters should not generate resonating behaviour. In extreme applications where the process behaviour changes a lot it might be necessary to increase that tested variation larger values than 30%. If that is not the case, it is in general a good choice to lower all 3 parameters with the equal percentage.

### Saving into the controller

The parameters entered in the Tuning tab are only written in the volatile memory. This means that when you switch off the controller the changes you made are lost. In order to save them into the non-volatile memory you need to save them with the non-volatile button.

Save in non volatile

If you want to save the application settings into a new config file you can save it as a partial config file or merge the changes to an existing config file.

Save in config file

This is equally important than saving the program of the application!