Tuning Guide

Overview

This section covers tuning the velocity and position loops in the AKD. Servo tuning is the process of setting the various drive coefficients that are needed for the drive to optimally control the servo motor for your application. There are different ways to tune, and several are covered here. We will give you guidance on what the different methods of tuning are and when to use them.

The AKD works in three major operation modes: torque, velocity, and position operation mode. No servo loop tuning is required for torque mode. Velocity loop and position loop tuning are covered below.

The AKD has an auto tuner that will provide the tuning that many applications will need. This section describes the tuning process and how to tune the AKD, specifically for cases where the user does not want to use the auto tuner.

Tuning in this section will focus on tuning in the time domain. This means that we will look at the velocity or position response vs. time as the criteria we use to decide how well tuned a control loop is tuned.

Determining Tuning Criteria

Choosing the proper specifications for a machine is a prerequisite for tuning. Unless you have a clear understanding of the type of performance needed to push the machine into production, the tuning process will cause more problems and headaches than it solves. Take time to layout ALL the requirements of the machine—nothing is too trivial to consider.

- Determine what the most important criteria are. The machine was likely designed and developed with a certain performance in mind. Include ALL performance criteria in the specification. Do not concern yourself with whether or not the criteria sound scientific. (i.e. If the motion needs to visibly look smooth, put it in the specification. If it can't have any noise, put it in the specification.) At the end of the development phase, the machine's performance should match the performance previously set in the specification. This will ensure that the machine meets its performance goals and that it is ready for production.
- Test the machine with realistic motion. Do not simply tune the machine to make short linear motion, when it will make long, s-curve motions in the real world. Unless you test the machine with realistic motion, there is no way to determine if it is ready for production.
- Determine some specific, quantitative criteria for identifying unacceptable motion. It's better to be able to tell when a motion is unacceptable than to try and figure out the exact point where acceptable motion becomes unacceptable. Here are some examples of motion criteria:

1. +/- x position error counts during the entire motion.
2. Settling within +/- x position error counts, within y milliseconds.
3. Velocity tolerance of x% measured over y samples.

- It is important to focus on the things that will get the machine into production with reliable performance, based on a fundamental understanding of the system.
After you have constructed a detailed servo performance specification, you are now ready to start tuning your system.

**Before You Tune**

In the worst case, if something goes wrong during tuning, the servo can run away violently. You need to make sure that the system is capable of safely dealing with a servo run away. The drive has several features that can make a servo run away safer:

- Make sure that the limit switches turn the drive off when tripped. If a complete run away occurs, the motor can move to a limit switch very quickly.
- Make sure the max motor speed is set accurately. If a complete run away occurs, the motor can reach max speed quickly and the drive will then disable.

**Closed Loop Tuning Methods**

The closed loop control loop is responsible for the desired position and/or velocity (trajectory) of the motor and commanding the appropriate current to the motor to achieve that trajectory. The challenge in closed loop control loops is to make a system that not only follows the desired trajectory, but also is stable in all conditions and resist external forces, and do all of this at the same time.

When in velocity operation mode, only the velocity loop is tuned. When in position operation mode, both the velocity and position loops must be tuned.

**Tuning the Velocity Loop**

The velocity loop on the AKD consists of a PI (proportional, integral) in series with two anti-resonance filters (ARF) in the forward path and two anti-resonance filters in series in the feedback path.

To perform basic tuning of the velocity loop, you can use just the PI block and set ARF1 and ARF2 to unity (no effect) and set the observer to 0 (no effect). Using just the PI block simplifies the process of tuning the velocity loop. To start tuning you can adjust the PI Controller block first. A simplified velocity loop without anti-resonant filters and observer is shown below. This is how you can think of the loop before the anti resonant filters and observer is used.
Procedure for simple velocity loop tuning:

1. Set DRV.OPMODE to velocity or position, as appropriate for your application. If DRV.OPMODE is set to position, set VL.KVFF to 1.0.
2. Set VL.KP to 0.
3. Set VL.KI to 0.
4. Set service motion to make a motion that is similar to the move speeds that will be used in the real application. Do not set the service motion to a speed higher than ½ of the maximum motor speed, to allow for safe overshoot during tuning. Set acceleration to an appropriate value for your application. Set service motion to reversing. Set time1 and time2 equal to 3 times the expected settling time for the system. 1.0 second is a reasonable value for time1 and time2, if you don’t know the expected settling time.
5. Enable the drive and start the service motion. You should see no motion, as there are no velocity loop tuning gains at this point.
6. When adjusting VL.KP and VL.KI, below record VL.FB and VL.CMD. These are the traces that are used to determine the performance of the velocity loop.
7. Adjust VL.KP. Keep increasing VL.KP by a factor of 2 until you either:
   - Hear an objectionable noise from the system (buzzing, humming, etc) or
   - See velocity overshoot. No velocity overshoot should be present when using only VL.KP.
   - When you reach one of the limits above, decrease VL.KP to the value where there were no objectionable noises or overshoot.
8. Adjust VL.KI. Increase VL.KI by a factor of 1.5 until you either:
   - Hear or see objectionable noise or shuddering from the system
   - See > 15% overshoot
Here is an example of 15% overshoot. This is zoomed in view of a service motion commanded to 1000 RPM (location 1), where the overshoot peaks at 1150 RPM (location 2).

Here is an example of 11 overshoots. Each overshoot is shown by a red circle.

When you reach one of the limits above, decrease VL.KI to the value where there were no objectionable noises or overshoot.

9. Stop the service motion

**Tuning the Position Loop**

The position loop is a second loop that builds upon a correctly tuned velocity loop to provide accurate control over position. The position loop is a simple element that consists of a PI loop. It is simplest to tune the P and I terms in the velocity loop and use only the P term in the position loop.
At most, use only three non-zero P and I terms from both the velocity loop and the position loop. One combination would be VL.KP, VL.KI, and PL.KP. Another valid combination would be VL.KP, PL.KP, and PL.KI. The VL.KP, VL.KI, and PL.KP combination is shown here.

Procedure for tuning position loop:
1. Set VL.KVFF to 1
2. Increase PL.KP until either:
   - You see 25% overshoot, or
   - You see > 3 overshoots, or
   - You hear objectionable noises from the system.
   - When you reach one of the limits above, decrease PL.KP to the value where there were no objectionable noises or overshoot.

Torque Feedforward Tuning Methods

The torque based feedforward terms on the AKD effectively model the physics of your motor and allow the drive to command the appropriate current, even before the encoder has time to send data back to the drive. Torque based feedforward terms allow you to lower following error with virtually no stability penalty.

Shape Based Feedforward Tuning

To adjust IL.KAFF:

- Tune the VL.KP and VL.KI as shown above in the velocity loop tuning section. Set DRV.OPMODE to velocity (or set PL.KP and PL.KI to 0 and vl.kvff to 1).
- Set up a short, repeating service motion with accelerations that are representative of the moves you will use in your application (exact values for acceleration are not critical).
- Turn up IL.KAFF until the position error (PL.ERR) is proportional to the inverted velocity command. The adjustment of IL.KAFF will focus on removing bumps on acceleration and deceleration. The picture below has an
ideal value of IL.KAFF of 1.7.

Using Anti-Resonance Filters

The AKD has four anti-resonance filters. Two filters are in the forward path and two are in the feedback path.

Similarities

- Both types are typically used to enhance stability and performance of the system.

Differences

- Forward path filters result in higher phase lag in closed loop system response.
- Forward path filters limit spectrum from reaching the motor / feedback path filters only filter the feedback after it has been to the motor.
Types of Anti-Resonance Filters

Low Pass

A low pass filter allows signals through below a corner frequency and attenuates the signals above the same corner frequency. The behavior at the corner frequency can be specified with the low-pass Q.

![Frequency Response of Multiple Lowpass Filters](image)

To specify a lowpass filter, you must specify the frequency and Q for both the zero and pole on anti-resonance filter 1. To do this, see the following example using the terminal commands that sets:

- Filter Type = Biquad
- Zero frequency = 700 Hz (This is the Lowpass cutoff frequency)
- Zero Q = 0.707
- Pole frequency = 5000 Hz
- Pole Q = 0.707

VL.ARTYPE1 0
VL.ARZF1 700
VL.ARZQ1 0.707
VL.ARPF1 5000
VL.ARQPQ1 0.707
**Lead Lag**

A lead lag filter is a filter that has 0 dB gain at low frequencies and a gain that you specify at high frequencies. You also specify the frequency that the gain at which the transition occurs.

![Frequency Response of Multiple Lead Lag Filters](image)

To specify a Lead Lag filter, you must specify the Center Frequency and high frequency Gain (dB). To do this, see the following example by clicking on the Velocity Loop:

Click on Velocity Loop (1), then select the AR1 Tab (2), using the Filter Type drop-down, select Lead Lag (3), lastly, enter the desired Center Frequency and Gain of the Lead Lag filter (4).
Notch

A notch filter changes gain at a specific frequency. You specify the frequency at which the gain change occurs (Frequency (Hz)), how wide of a frequency range the cut occurs (Q), and how much the gain changes (Notch Depth (dB)).
To specify a notch filter, you must specify the Frequency (Hz), Depth (dB) and Width (Q) of the notch. To do this, see the following example by clicking on the Velocity Loop:

Click on Velocity Loop (1), then select the AR1 Tab (2), using the Filter Type drop-down, select Notch (3), lastly, enter the desired Frequency, Depth and Q of the Notch filter (4).
Biquad

A biquad is a flexible filter that can be thought up as being made up of two simpler filters; a zero (numerator) and a pole (denominator). In fact, the pre-defined filters mentioned above are really just special cases of the biquad.

Both the zero (numerator) and the pole (denominator) have a flat frequency response at low frequencies and a rising frequency response at high frequencies. The transition frequency and damping must be specified for both the numerator and denominator.

Analyzing the numerator and denominator, the frequency response calculation is simple:

If the numerator and denominator are plotted in dB, the biquad response is numerator – denominator. Understanding how the numerator and denominator work is crucial in understanding how a biquad frequency response is created.

Below is an example of a biquad filter similar to a Lead Lag filter type. To help understand how to determine the frequency response of the biquad, the numerator and denominator response have been plotted. If the denominator is subtracted from the numerator, the biquad response is the result.
The biquad filter is very flexible, which allows custom filters to be designed. Below is an example of a resonance filter using a biquad. Notice how the high Q values affect the numerator and denominator. This gives a biquad frequency response similar to a mechanical resonance.
The previous two examples used a numerator frequency lower than the denominator frequency, yielding a positive gain in high frequencies. If the denominator frequency is lower than the numerator frequency, then high frequencies will have a negative gain.

Below is an example where the numerator frequency is higher than the denominator. Notice the high frequencies have a negative gain.
To specify a biquad filter, you must specify the frequency and Q for both the zero and the pole on anti-resonance filter 3. To do this, see the following example using the terminal commands that sets:

- Filter Type = Biquad
- Zero frequency = 100 Hz
- Zero Q = 0.7
- Pole frequency = 1000 Hz
- Pole Q = 0.8

VL.ARTYPE3 0
VL.ARZF3 100
VL.ARZQ3 0.7
VLARPF3 1000
VLARPQ3 0.8

### Biquad Calculations

In the s-domain, the linear biquad response is calculated:
Biquad Frequency

To convert from idealized s-domain behavior to a more realistic z-domain behavior, we convert using a pole/zero transform. To calculate the frequency response for an individual frequency:
\[ t = 62.5 \mu \text{sec} \]

\[ N_{\text{Rad}} = 1 - (2Q_N)^{-2} \]

\[ N_{2\text{ Unscaled}} = 1 \]

\[ \text{if}(N_{\text{Rad}} > 0): N_{1\text{ Unscaled}} = -2e^{-\omega_{N\zeta} N t} \cos(\omega_N t \sqrt{1 - \zeta^2}) \]

\[ \text{if}(N_{\text{Rad}} \leqslant 0): N_{1\text{ Unscaled}} = -2e^{-\omega_{N\zeta} N t} \cosh(\omega_N t \sqrt{1 - \zeta^2}) \]

\[ N_{0\text{ Unscaled}} = e^{-\omega_{N\zeta} N t} \]

\[ D_{\text{Rad}} = 1 - (2Q_D)^{-2} \]

\[ D_2 = 1 \]

\[ \text{if}(D_{\text{Rad}} > 0): D_1 = -2e^{2\omega_{D\zeta} D t} \cos(\omega_D t \sqrt{1 - \zeta^2}) \]

\[ \text{if}(D_{\text{Rad}} \leqslant 0): D_1 = -2e^{-2\omega_{D\zeta} D t} \cosh(\omega_D t \sqrt{1 - \zeta^2}) \]

\[ D_0 = e^{-2\omega_{D\zeta} D t} \]
$$N_{Scale} = \frac{N_{0Unscaled} + N_{1Unscaled} + N_{2Unscaled}}{D_0 + D_1 + D_2}$$

$$N_2 = \frac{N_{2Unscaled}}{N_{Scale}}$$

$$N_1 = \frac{N_{1Unscaled}}{N_{Scale}}$$

$$N_0 = \frac{N_{0Unscaled}}{N_{Scale}}$$

$$\angle_z = \omega t = 2\pi \times \text{freq} \times t$$

$$\text{Num}_{Re} = N_2 \cos(2\angle_z) + N_1 \cos(\angle_z) + N_0$$

$$\text{Num}_{Im} = N_2 \sin(2\angle_z) + N_1 \sin(\angle_z)$$

$$\text{Den}_{Re} = D_2 \cos(2\angle_z) + D_1 \cos(\angle_z) + D_0$$

$$\text{Den}_{Im} = D_2 \sin(2\angle_z) + D_1 \sin(\angle_z)$$

$$\text{Gain}_{dB} = 20 \log_{10} \left( \frac{\sqrt{\text{Num}_{Re}^2 + \text{Num}_{Im}^2}}{\sqrt{\text{Den}_{Re}^2 + \text{Den}_{Im}^2}} \right)$$

$$\text{Phase}_{\text{deg}} = \frac{180}{\pi} \left( \tan^{-1} \left( \frac{\text{Num}_{Re}}{\text{Num}_{Im}} \right) - \tan^{-1} \left( \frac{\text{Den}_{Re}}{\text{Den}_{Im}} \right) \right)$$
Common Uses Of Anti-Resonance Filters

Low pass filters in the feedback path. This is a common way to deal with noisy feedback sensors. When used in combination with noisy feedback sensors, significant reduction in audible noise can result.

Lead / lag filters in the forward path. This is a common way to achieve phase lead for control loops without exciting high frequency resonances.

Low pass filters in the forward path. This is a common way to limit high frequency energy from reaching a system that can not productively use energy at these high frequencies. This is also used to lower the effect of system resonances over a wide range of frequencies.

Notch filters are used to cancel system resonances. Notch filters are designed to be the opposite in amplitude of system resonances. Notch filters are applied to very specific frequencies, and therefore you must know your system resonance frequencies accurately to use them effectively.