MIMO Gantry Commissioning

User Guide

Version	Date	Editor	Comment
001	2020-11-11	dg	initial version

Document AN139_MIMOGantryCommissioning_UserGuide_EP
Version 001
Source Q:\doc\ApplicationNotes\
Destination T:\Doc\ApplicationNotes
Owner dg

Triamec Motion AG
Industriestrasse 49
6300 Zug / Switzerland

Phone +41 41 747 4040
Email info@triamec.com
Web www.triamec.com

Disclaimer

This document is delivered subject to the following conditions and restrictions:

- This document contains proprietary information belonging to Triamec Motion AG. Such information is supplied solely for the purpose of assisting users of Triamec products.
- The text and graphics included in this manual are for the purpose of illustration and reference only. The specifications on which they are based are subject to change without notice.
- Information in this document is subject to change without notice.

Table of Contents

1 Introduction

This document describes the commissioning of a gantry axis equipped with two motors and two position sensors on each side of the gantry. This description can be applied to stiff or a flexible gantries.

The mechanical coupling of the tow motors requires special considerations to reach a dynamic and stable behavior of the controller. The controller topology of the *Triamec* servo drives allows to transform the two coupled motor axes x_{Axis0} and x_{Axis1} into a decoupled system, with

- a rotational axis θ indicating the angular position of the gantry, and
- a linear axis x_{Lin} indicating the location of the balance point.

The coordinates are illustrated in Figure 1. The position difference $\Delta x = x_{Axis0} - x_{Axis1}$ is used as input for the rotational controller which is proportional to the angle θ [1].

This transformation into a linear/rotational coordinate system offers the following advantages:

- With the decoupled system, a stiffer and more stable controller can be configured with less complicated tuning.
- The linear/rotational approach can be applied to either stiff or flexible gantries.
- The static or dynamic location of the balance point can be considered easily.

Triamec servo drives ordered with the option "gantry system control" (GY) support the transformation to a decoupled linear/rotational system. To activate the transformation, the parameter General.Parameters.ControllerTopology must be set to MimoGantry. This document describes the required steps for the commissioning of the drive with the MimoGantry controller topology.

1 For small values of θ compared to π the position difference can be expressed as $\Delta x \approx \theta \, \Delta y$.

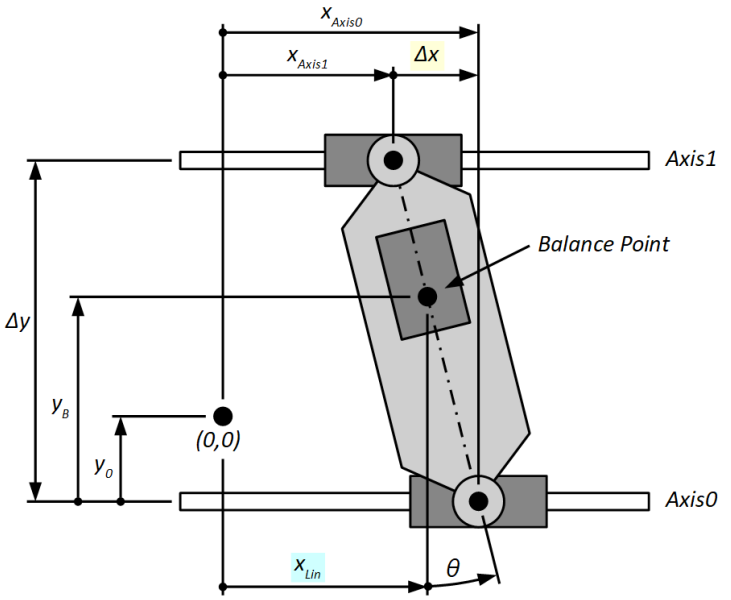

Figure 1: Gantry coordinate systems: Motor related coordinates (x_{Axis0}, x_{Axis1}) and linear/rotational coordinates (θ, x_{Lin}) .

1.1 Preconditions / Restrictions

The following preconditions have to be fulfilled and also some restrictions have to be considered before the commissioning of the gantry can start:

- The servo drive must have the software option "*gantry system control*" (*GY*). Check that register General.Signals.Identifications.Product shows the *GY* tag towards the end of the product name (e.g. *TSD80-10-TH-NONO-**GY***).
- It is possible to add the *GY* software option on an already delivered drive against an additional fee.
- The linear/rotational gantry option requires firmware 4.10.5 or higher and *TAM System Explorer* version 7.15.0 or higher. The current firmware and *TAM Software* can be downloaded from www.triamec.com.
- Only dual axis drives support the *GY* software option (TSD80, TSD130, TSD350).
- Serial encoders are not supported (e.g. *DigitalEndat*, *DigitalBiss*, *DigitalTamagawa*, *DigitalNikon*).
- Transformation of positions in the *Tama* program is not supported.
- Modulo is not supported. Both parameters ModuloPositionMaximum and ModuloPositionMinimum need to be set to zero.
- Gantry with dual-loop control (which involves both Controller[0] and Controller[1]) is not yet implemented. Please contact *Triamec Motion AG* in case you intend to setup a gantry with dual-loop control.
- For the commissioning it is assumed that the mechanical setup allows a low dynamic movement of the gantry with only one active motor.

1.2 Integration in TAM System Explorer

As explained in the introduction, the position controller of the gantry system is implemented as a linear and a rotational axis. As this topology is integrated into the same register tree as used for the indepen-

dent control of two axes, some registers have a different meaning when configuring it for a gantry.

The main changes concern the position controller as shown in Figure 2. The position controller of Axis[0] controls the linear motion and the position controller of Axis[1] the rotational motion of the gantry.

Chapter 3 explains usage and properties of all registers which change their meaning when used for a gantry.

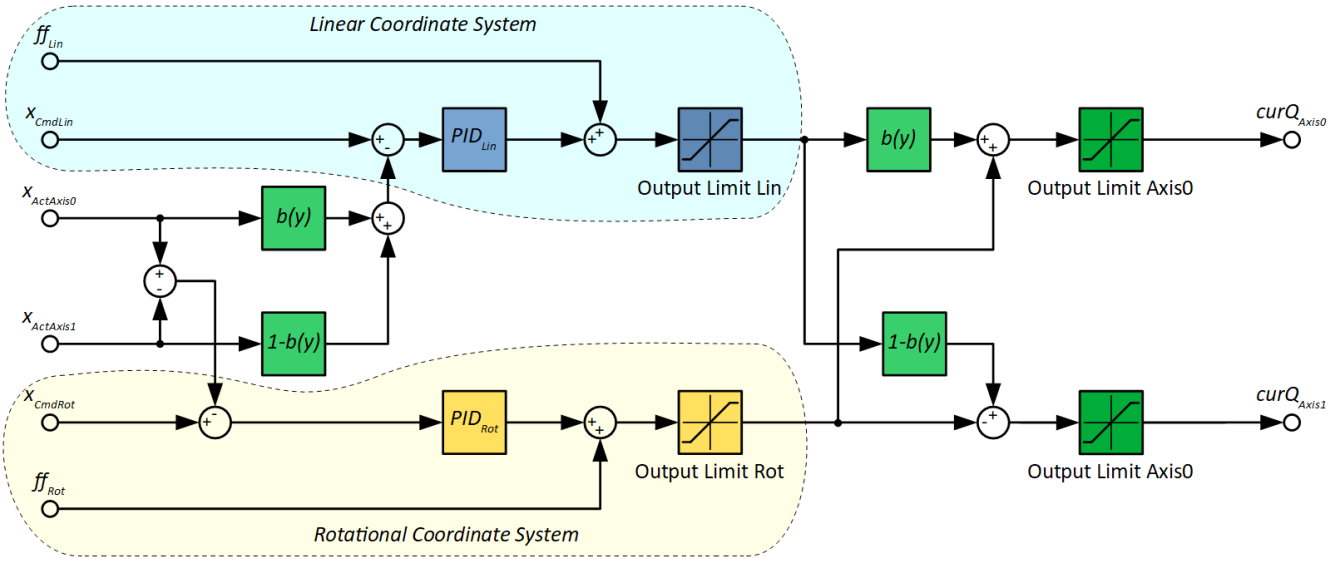

Figure 2: Structure of decoupled gantry controller.

2 Commissioning

This chapter explains the required steps to commission the gantry as a decoupled system with a linear and rotational axis. These steps reflect the best practice for a common gantry setup and may need to be adjusted for the particular system.

Important When executing the following steps, one-sided loads might be applied to the gantry. With a wrong parametrization this could potentially cause damage to the mechanics. Therefore, it is important to set the output limits carefully and to not letting the loads exceed the tolerated values.

2.1 TAM Parameter Preparations

If the servo drive was already used for a different setup, it is recommended to reset to default settings: right-click the device node in the register tree, then click – **Manage persistence...** – **Disable persistence** and reboot the drive.

First, all known parameters (e.g. motor data, encoder data, position unit, ...) need to be set in the register tree for both Axis[0] and Axis[1] – see [1] section 5.1 and 5.2 for the initial setup of the parameters.

For the preparation of the parameters the following registers need special consideration:

General.Parameters.ControllerTopology:

- Finally, this parameter will be set to MimoGantry. But as properties like the current controller and the commutation are not yet ready, the parameter ControllerTopology has to be set to Standard for the first part of the commissioning.

Gantry.Parameters:

- Set the value of BalanceOffset to the expected fraction of the total linear inertia which affects Axis[0]. If the balance point is centered, BalanceOffset is 0.5.
- As the dependency of another axis is not yet considered, BalanceSlope is set to zero.
- With CurrentLimit0 and CurrentLimit1 the desired current applied to the motors can be restricted. Typically, these parameters are set about 10% below the peak current of the motor or the drive, whichever is smaller.
- The parameter AlignAfterHoming must remain False until the homing sequence is configured and successfully tested.

Parameters.PathPlanner:

- The path planner parameters of Axis[0] define the dynamic properties of the linear motion if the internal path planner is used.
- The path planner parameters of Axis[1] define the dynamic properties of the rotational motion. It is recommended to use slow dynamics as this path planner is only used, for special tasks, such as aligning the rotational position to zero after homing.

Parameters.PositionController:

- For the initial setup the Controller parameters Kp, Ki and Kd should be set to zero for both axes.
- To protect the mechanical setup against too much torsional tension, the value of the parameter OutputLimit of Axis[1] should be reduced to a tolerated value. Also, the value of the PositionErrorLimit of Axis[1] which corresponds to the difference between encoders should be set to a value within the tolerated rotational deviation of the two encoders.
- The parameter FeedForwardAcceleration of Axis[0] can be calculated based on the total linear inertia and the torque constant of the motors according to equation 1. For Axis[1], the parameter FeedForwardAcceleration can be set to zero.

$$FeedForwardAcceleration = \frac{Inertia}{TorqueConstant} \tag{1}$$

Parameters.Commutation:

- For the commissioning of the gantry, the PhasingMethod has to be set to RotorAlignment and EnablingMethod to ForcePhasing. After the position controller is configured, this parameter can be set to AngleSearch and Automatic in order to avoid a movement of the gantry during the phasing.
- As long as PhasingMethod is set to RotorAlignment, the RampRiseTime should be set to about 1s to 4s to not excite the gantry too much during phasing. Later this parameter will be reduced to shorten the time used for phasing.

Parameters.CurrentController:

- For the initial setup, the parameters Kr and Tn should be set to zero for both axes.

Verification of the Encoder

Before continuing with the commissioning, it is recommended to check the encoder scale and counting direction. This can be done by manually moving the axis by a certain distance in positive direction and observe the encoder position of both axes with the scope.
In case the direction or the scaling does not match, the following parameters must be adjusted:

- PositionController.Encoders[0].InvertDirection
- PositionController.Encoders[0].Pitch

In case the encoder signal shows no movement at all, the configuration and the connection of the encoders have to be verified.

2.2 Current Controller

This section describes the tuning of the current controller.

- Make sure all parameters are set as described in section 2.1. Especially General.Parameters.ControllerTopology has to be set to Standard.
- To determine the parameters Kr and Ki, execute a Bode tuning for both axes as described in [1] section 5.4.1.
- Parameters Kr and Ki of the current controller can be set for both Axis[0] and Axis[1] individually. In most cases the same parameters can be used for both axes — assuming the two motors are identical.

Warning As soon as the parameters for the current controller are committed, the axis will execute a movement in the range of one pole pair pitch when the axis is enabled! Later, this movement will be avoided by setting PhasingMethod to AngleSearch.

Verification of the Phasing

To verify if the phasing is functioning as expected, add the following signals to the scope and run the scope with a sampling time of 0.1ms during the phasing sequence.

- Signals.PositionController.Encoders[0].Position
- Signals.CurrentController.ActualCurrentQ
- Signals.CurrentController.ActualCurrentD

Check if the applied current is sufficient to align the coils with the magnets of the stator. Figure 3 shows a typical plot of the phasing sequence with RotorAlignment.

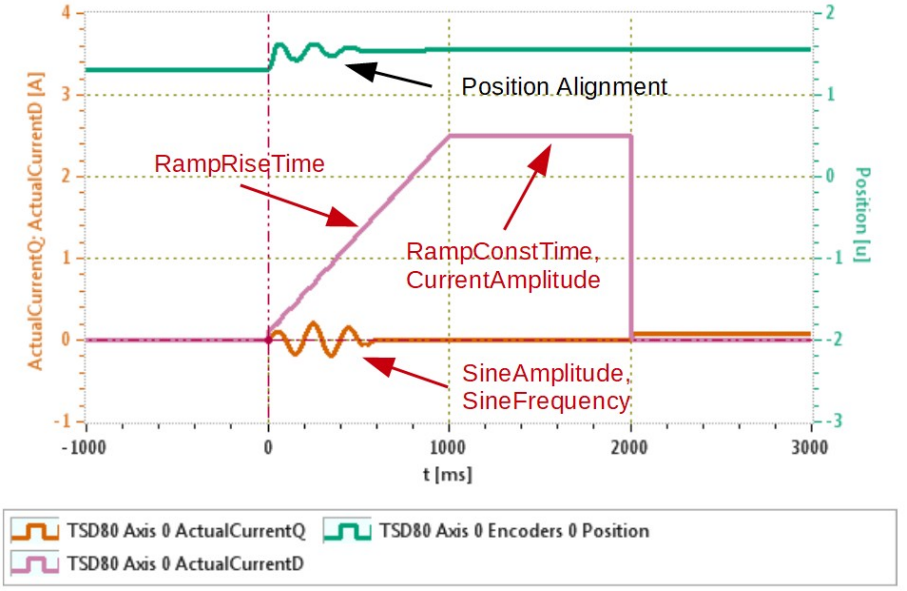

Figure 3: Phasing with RotorAlignment.

Verification of the Motor Direction and Magnetic Pitch

To verify the motor direction and the magnetic pitch, the TestGenerator is used to apply a rotating current vector, which will move the axis in positive electrical direction. The following steps are required to run the verification:

- Prepare the scope with the following signals, set the sampling time to 0.1ms and time range to 10s.
  - Axes[0].Signals.Commutation.Angle
  - Axes[0].Signals.PositionController.Encoders[0].Position
  - Axes[0].Signals.TestGenerator.Output
- Prepare the TestGenerator with the following values:
  - Frequency: This parameter defines how fast the axis will move during the test. For example, if the parameter is set to 0.5Hz, it will take 2s to move a distance of one pole pair pitch.
  - Amplitude: 0
  - Offset: 50% to 100% of the nominal current
  - RampTime: 2s
  - CommutationAngle: 0radTemporarily increase PositionController.Controllers[0].PositionErrorLimit to a value bigger than one pole-pair pitch.
- Enable Axis[0], start the scope and set Axes[0].Commands.TestGenerator.Command to StartRotatingVectorConstantCurrent and press Enter.

Warning As soon as StartRotatingVectorConstantCurrent is committed, the axis will start to move. The value of TestGenerator.Frequency has to be set slow enough to stop the axis in time (see next item).

- If the setup is correct, the axis will now move with the defined frequency. To stop the axis, set TestGenerator.Command to Stop and press Enter. Stop the scope and disable the axis.
- To verify the motor direction, check if the signals of Commutation.Angle and PositionController.En-

coders[0].Position have the same direction (see also Figure 4). If the direction is opposite, switch the state of the parameter Motor.InvertDirection.

- To verify the magnetic pitch, check if one electrical turn (2·π) on the Commutation.Angle corresponds to the magnetic pitch measured on PositionController.Encoders[0].Position (see Figure 4).
- Repeat this sequence also for Axis[1].
- Do not forget to reset PositionController.Controllers[0].PositionErrorLimit.

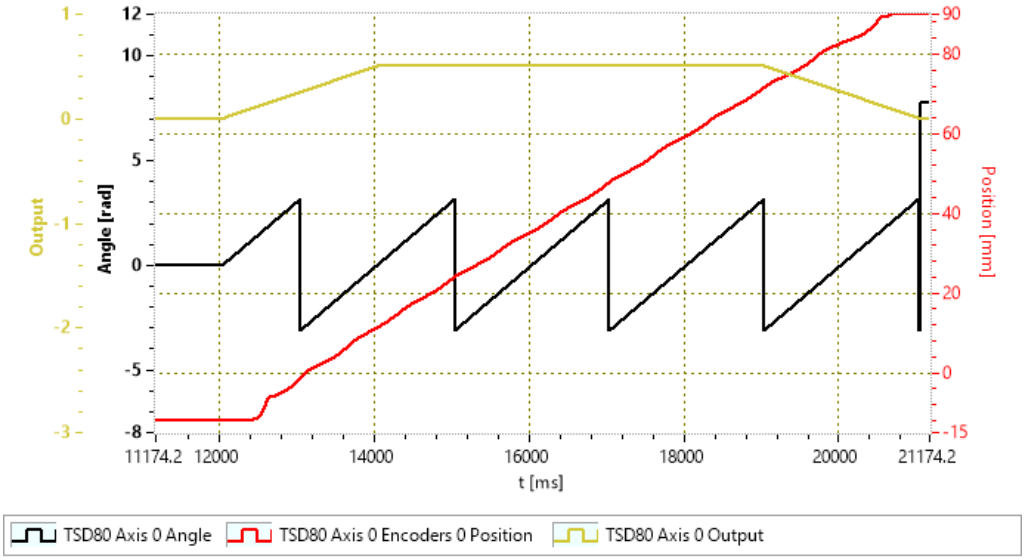

Figure 4: Verification of motor direction and magnetic pitch: The motor direction is correct as the signal of Angle and Position show the same direction. The magnetic pitch is 20 mm according to the measurement.

2.3 Trimming of the Balance Gain

The balance gain defines the load distribution to the two motors of the gantry. The parameter BalanceOffset (and in case there is a dependency with an other axis, the BalanceSlope parameter) can be used to configure the balance gain. This section describes how the value can be determined.

In many cases the balance gain can be determined in advance, for example, based on a CAD model. For a symmetric setup the balance gain is 0.5.

To verify the balance gain, or if unknown in advance, it can be determined with a Bode measurement by executing the following steps:

- Preparation: Make sure the General.Parameters.ControllerTopology is set to Standard.
- Execute a Bode measurement for Axis[0] and Axis[1] each with **Method** set to **Open Loop Bode**.
- Open both measurements in the **Bode Tuning** window and measure at the same frequency the gain g_0 for Axis[0] and g_1 for Axis[1] in the lower frequency range. With this the balance gain b can be evaluated according to the following equation:

$$b = 10^{\frac{g_1 - g_0}{20}} \tag{2}$$

- In case there is no dependency with another axis, the parameter BalanceOffset can be set to b.

- If there is a dependency to another axis (in the following called axis *y* in accordance with Figure 1), repeat the measurement above for different positions of the axis *y* and fit BalanceOffset and BalanceSlope into the measurement according to the following equation:

$$b = BalanceOffset + BalanceSlope \cdot y \quad (3)$$

Position of Axis Y

To evaluate equation 3, the servo drive needs to know the position of axis *y*. Therefore, the position of axis *y* has to be written into the following register:

- Axes[1].Commands.PathPlanner.StreamX

Depending on the control system this can be done in several ways:

- Options with *TwinCAT* and *Tria-Link*: (The position is given in SI units)
  - With *Trial-Link*, it is possible to directly cross publish data from one drive to another. See [3] for how to set up cross publishing with *TwinCAT*.
  - With the function block *TL_publishMaster2Slave*, the setpoint of *y* in *SI* units can be transmitted to Axes[1].Commands.PathPlanner.StreamX (see also [4]).
  - If the requirements regarding real-time are low, the asynchronous *TL_MC_RegisterWrite* function block can be used to transmit the position in *SI* units (see also [5]).
- With *EtherCAT*:
  - In the **I/O** interface of **Module2** (second axis of the drive), set the cyclic data *ModeOfOperation* to 8 and *TargetPosition* to the position of axis *y* in *increments*.

2.4 Position Controller

The tuning of the position controller is done in the linear/rotational coordinate system. Therefore, the parameter General.Parameters.ControllerTopology has now to be set to MimoGantry.

Important With the change of the ControllerTopology to MimoGantry the assignment of some signals changes from the linear to the rotational coordinate system. See chapter 3 for an overview of the affected registers.

Bode Tuning

For the tuning of the position controller the following sequence is recommended:

- Set General.Parameters.ControllerTopology to MimoGantry.
- Execute a Bode measurement with Axis[0] and set **Method** to **Gantry Axis[0]: Linear** in the bode measurement window. Save the measurement with suffix "*linear*" in the file name.
- Execute a Bode measurement with Axis[1] and set **Method** to **Gantry Axis[1]: Rotational** in the bode measurement window. Save the measurement with suffix "*rotational*" in the file name.
- For the tuning of the linear position controller, open the corresponding Bode measurement using Axis[0] and setup the controller as described in [1].
- For the tuning of the rotational position controller, open the corresponding Bode measurement using Axis[1]. In case of a stiff gantry, the description in [1] may not be applied directly. See the section below for how to set up the rotational controller with a stiff gantry.

- See [1] section 5.5.2 for how to verify the controller tuning in time domain.

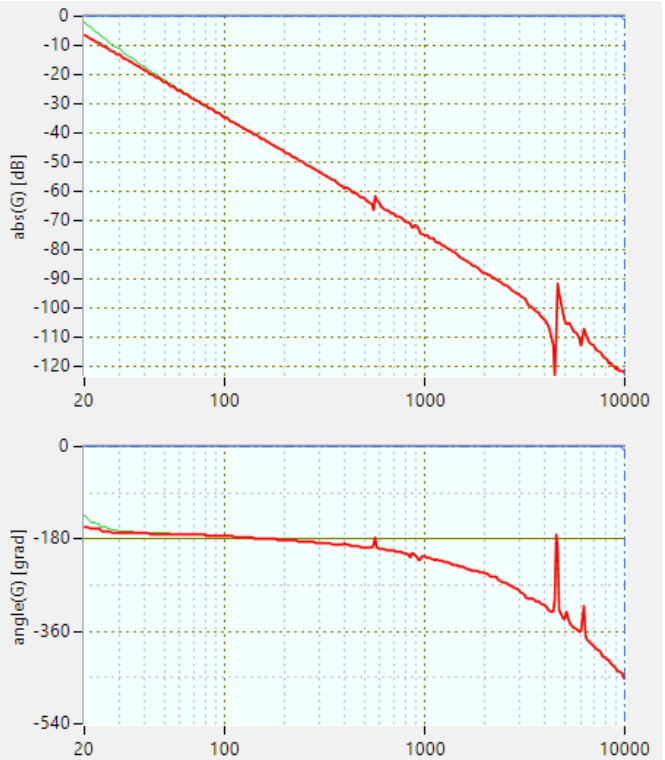

Figure 5: Typical Bode measurement of the linear axis of a gantry.

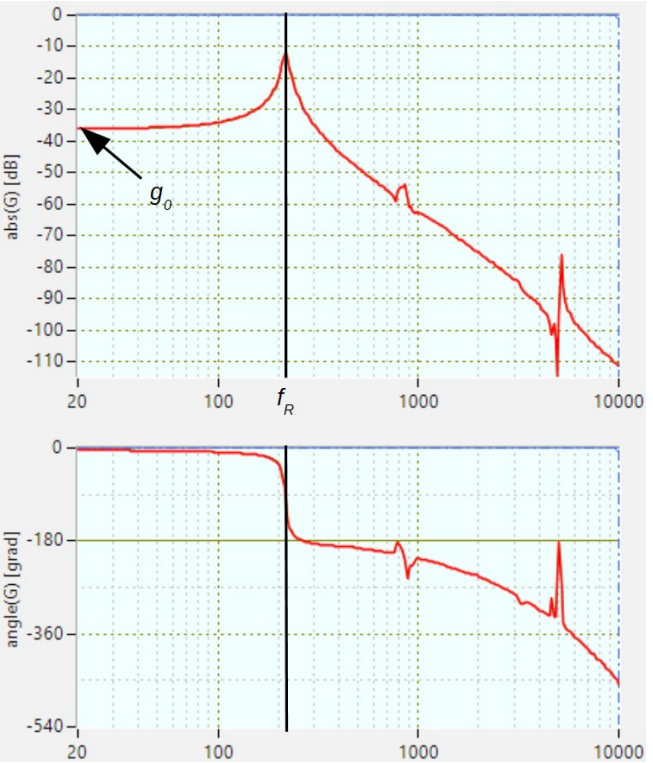

Figure 6: Typical Bode measurement of the rotational axis of a stiff gantry with a gain g0 of -36dB and resonance frequency f_R of 218Hz.

Tuning of the Rotational Axis

For soft gantries, the feedback of the rotational axis is similar to the properties of a common linear axis. Therefore, the Bode measurement shows a characteristic similar to the measurement displayed in Figure 5. The same approach for the tuning can be applied as described in [1].

Stiff gantries show a different characteristic for the rotational axis which resembles to a spring mass system (see Figure 6). Depending on the relation between the resonance frequency f_R and the desired bandwidth of the controller f_B, a different approach for the tuning has to be used:

- If the resonance frequency f_R is smaller than the desired bandwidth f_B, the same approach for the tuning can be applied as described in [1].
- If the resonance frequency f_R is in range of the desired bandwidth f_B, the following settings according equation 4 are recommended as an initial approach. See Figure 6 for how to determine the values for the gain g_0 and resonance frequency f_R.

$$\mathrm{Kp}=0\,; \quad \mathrm{Ki}=10^{\frac{-g_0}{20}}\frac{2\pi f_B^3}{2f_B^2+f_R^2}\ldots 10^{\frac{-g_0}{20}}2\pi f_B\,; \quad \mathrm{Kd}=\frac{2\mathrm{Ki}}{(2\pi f_R)^2}\,; \quad \mathrm{T1}=0.0002\,s\,; \tag{4}$$

- If the resonance frequency f_R is much bigger than the desired bandwidth f_B, it is recommended to

apply a notch filter at f_R and then increase the integrator gain Ki until the desired bandwidth is reached.

- Further optimization of the controller is possibly required based on the criteria of Bode and Nyquist. In most cases the following condition should be maintained:

$$Ki \leqslant 2(\pi f_R)^2 Kd \tag{5}$$

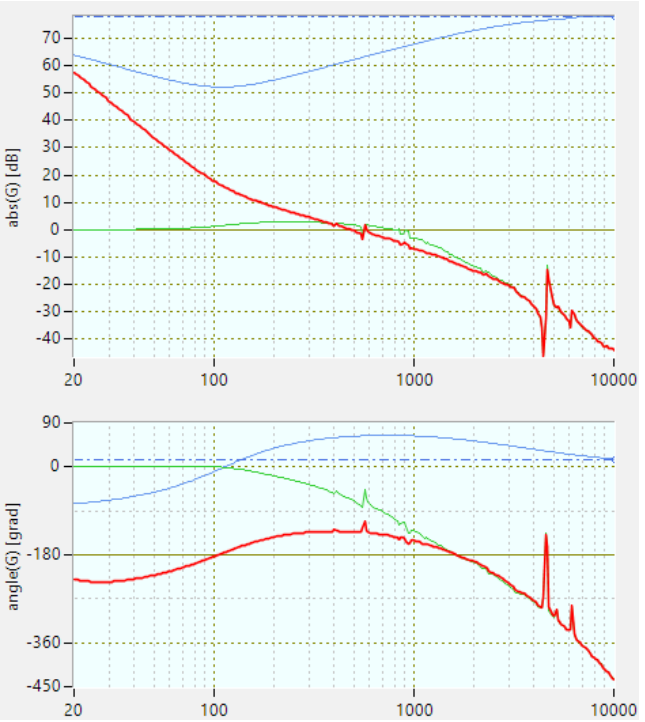

Figure 7: Bode plot of the tuned linear axis.

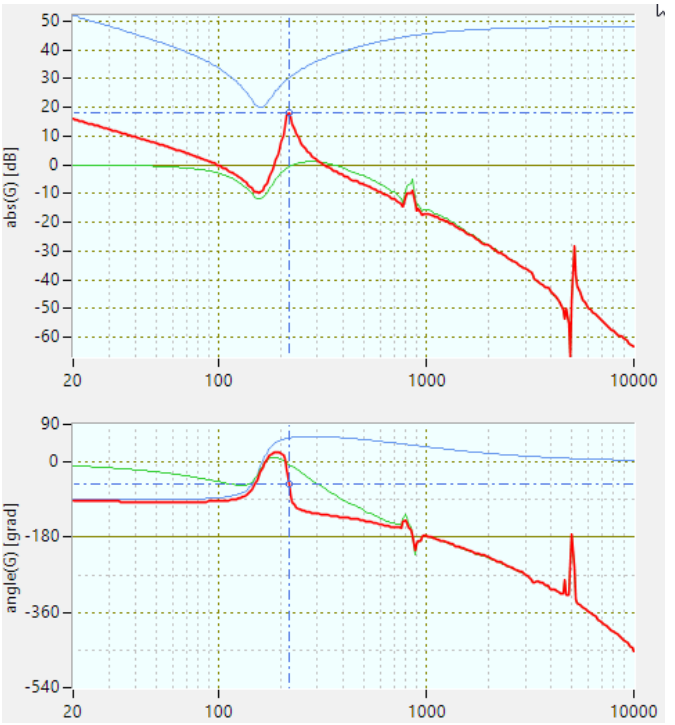

Figure 8: Bode plot of the tuned rotational axis of a stiff gantry.

Acceleration Feed Forward

The acceleration feed forward of the linear axis can also be determined or verified based on the linear Bode measurement and needs to be set in register Axes[0].Parameters.PositionController.FeedForwardAcceleration (see also [1] section 5.4.7 and 5.5.3).

As in normal case no motion is executed with the rotational axis, the FeedForwardAcceleration of Axis[1] can be set to zero.

2.5 Completion of the Commissioning

Phasing

For the commissioning of the gantry, the PhasingMethod was set to RotorAlignment. This causes a movement when the phasing is executed. With the following adjustments in node Parameters.Commutation of Axis[0] and Axis[1], this movement can be avoided:

- Set PhasingMethod to AngleSearch
- Set RampRiseTime to about 0.1s and RampRiseTime to 0.5s.

If the phasing with this new setup works as expected, the EnablingMethod can be set to Automatic. With this, the phasing will only be executed after a restart of the servo drive.

Limits

Before the gantry is used for normal operation, it is recommended to verify the value of the following parameters regarding protection of the mechanics and the dynamic requirement. Maybe some values were reduced during commissioning for safety reasons and should now be reset to the original value.

- Gantry.Parameters.CurrentLimit0
- Gantry.Parameters.CurrentLimit1
- Axis[0|1].Parameters.PositionController.OutputLimit
- Axis[0|1].PositionController.Controllers[0|1].PositionErrorLimit
- Axis[0|1].PositionController.Controllers[0|1].IntegratorOutputLimit
- Axis[0|1].CurrentController.OutputLimit

In general, these parameters should be set as small as possible to protect the environment, but with sufficient margin to not interfere with the normal operation of the machine.

Homing

The homing sequence for gantry is similar to the homing sequence of a normal axis (see [2] chapter 8 "Homing"). But the Gantry-Homing only needs to be commanded on Axis[0] , which will start both axes. The parameters for the second search can be individually set for Axis[0] and Axis[1] to support individual index marks for each axis.

- The homing parameters in Axis[0] define the general homing sequence while parameters in SecondSearchMove of Axis[0] specify the search of the home position for the encoder of Axis[0]. Parameters in SecondSearchMove of Axis[1] specify the search of the home position for the encoder of Axis[1]. These two searches run simultaneously.
- After the homing moves, the positions of both axes are set in such a way, that the position at the reference corresponds to the value of Commands.Homing.ReferencePosition.
- In case only the encoder of Axis[0] is used for the homing sequence, set Axes[1].Parameters.Homing.SecondSearchMove.EventInput to Skip. In this case, the encoder of Axis[1] is set to the same position as the encoder of Axis[0] at the home position and there will be no correction of an initial position offset.
- To start the homing sequence, select Start in the Command register of Axis[0] and press Enter.

If the parameter Gantry.AlignAfterHoming is False when homing is executed, a potential position offset between the two encoders will remain. If Gantry.AlignAfterHoming is set to True the rotational axis executes a move to zero to align the axis after the homing.

Warning Before Gantry.AlignAfterHoming is set to True the homing sequence has to be configured correctly and tested successfully. Especially for stiff gantries, false compensation of large offsets can cause large forces and potentially damage the mechanics.

Save the Configuration

When the configuration of the gantry is done and tested, it is recommended to persist the configuration on the servo drive and to save it on the PC (see section 3.5 in [1]).

3 TAM Registers with MIMO Gantry

This chapter explains the usage and meaning of all the registers which are affected, in case the parameter General.Parameters.ControllerTopology is set to MimoGantry. All other parameters can be handled the same as for two individual axes as described in [1].

3.1 The Gantry Node

If the software option "gantry system control" (*GY*) is enabled, the register tree shows a Gantry node with registers related to the gantry setup.

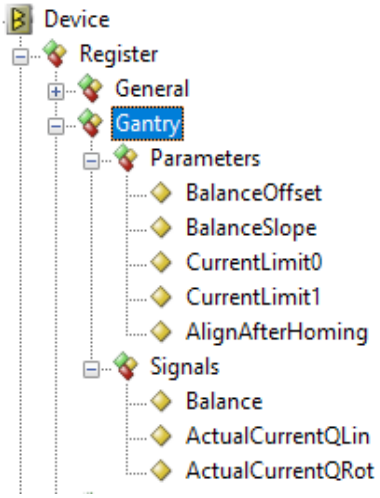

Figure 9: Gantry node in register tree.

Parameters

Gantry.BalanceOffset:

- The value of the parameter BalanceOffset determines the fraction of the total linear inertia which affects Axis[0]. If the balance point is centered, BalanceOffset has to be set to 0.5.
- The fraction of the total linear inertia which affects Axis[1] is 1 minus BalanceOffset.
- In case the position of the balance point depends on the position of another axis *y*, the value of the BalanceOffset is the fraction of the inertia when the position of *y* is zero (see also equation 3).

Gantry.BalanceSlope:

- With the parameter BalanceSlope, the effect of another axis *y* to the position of the balance point can be considered (see also Figure 1).
- The Balance factor *b* is calculated based on the position *y* according to equation 3.
- The position *y* has to be provided, for example, by the superior control system (e.g. TwinCAT) as a setpoint for Axis[1] and has to be written to register Axes[1].Commands.PathPlanner.StreamX.

- In case the balance point is not significantly impacted by the position of another axis or if the position of the other axis is not available, the value of BalanceSlope should be set to zero.

Gantry.CurrentLimit0:

- This parameter restricts the absolute setpoint current for the motor connected to Axis[0]. This can be useful to limit the maximum applied force by this motor.

Gantry.CurrentLimit1:

- This parameter restricts the absolute setpoint current for the motor connected to Axis[1]. This can be useful to limit the maximum applied force by this motor.

Gantry.AlignAfterHoming:

- If True, the rotational axis will be aligned to zero after homing.

Signals

Gantry.Balance:

- This register shows the actual balance factor evaluated according to equation 3.

Gantry.ActualCurrentQLin:

- This is the actual current in linear coordinates.

Gantry.ActualCurrentQRot:

- This is the actual current in rotational coordinates.

3.2 Parameters with New Assignment

This section describes the properties of parameters which change their assignment with the gantry setup.

Parameter.PathPlanner:

- The registers in Axis[0].Parameter.PathPlanner define the dynamic properties for the linear movement if the internal path-planner is used.
- The registers in Axis[1].Parameter.PathPlanner define the dynamic properties for the rotational movement e.g. to align the rotational axis after homing.

Parameter.PositionController:

- Encoders: The meaning of the encoder parameters does not change with the gantry topology. They still define the properties of the assigned physical encoder.
- Controllers, FeedForward, MasterPositionSource, ExcentricityCompensation: The parameters of Axis[0] are assigned to the linear axis, the parameters of Axis[1] to the rotational axis.
- PositionUnit: Set the unit of both axes to the unit of the linear axis.

Parameter.Commutation:

- The PhasingMethod parameters of both axes need to be set to the same value.
- In case PhasingMethod is set to RotorAlignment, the phasing of the two axes is done sequentially. First, the phasing is done for Axis[0], followed by Axis[1] using the parametrization of the corresponding registers. With this phasing method, the axis *will execute a movement* during the phasing

of up to one pole-pair pitch. This phasing method only requires a proper setup of the current controller for both axes.

- In case PhasingMethod is set to AngleSearch, the phasing of the two axes is done synchronously within the linear-rotational coordinate system. With this method, the axis *will not move* during the phasing. Nevertheless, it requires the configuration of the linear and rotational position controller before it can be used.
- In most cases, the method RotorAlignment should only be used during configuration of the gantry system and changed to AngleSearch when the position controller is configured.
- In general, the homing parameters for Axis[0] and Axis[1] should be identical.

Parameter.Homing:

- The homing is defined by the parameters Axes[0].Parameters.Homing and Axes[1].Parameters.Homing.SecondSearchMove.
- The homing parameters in Axis[0] define the general homing sequence while parameters in SecondSerachMove of Axis[0] specify the search of the home position for the encoder of Axis[0]. Parameters in SecondSearchMove of Axis[1] specify the search of the home position for the encoder of Axis[1].
- In case the encoder of Axis[1] is not used for the homing sequence, set Axes[1].Parameters.Homing.SecondSearchMove.EventInput to Skip.

3.3 Commands with New Assignment

This section describes the functionality of commands of Axis[0] and Axis[1] which change their assignment with the gantry setup.

Commands.General

- For both Axis[0] and Axis[1], the EnableAxis and DisableAxis commands enable or disable both motors of the gantry.

Commands.PathPlanner:

- The commands of Axis[0] can be used to control the linear motion and the commands of Axis[1] to control the rotational motion e.g. to align the rotational axis after homing.

Commands.PositionController:

- The commands of Axis[0] can be used to handle the linear position controller and the commands of Axis[1] to handle the rotational position controller.
- The Active command can be used to deactivate or activate the linear or the rotational position controller.

Commands.TestGenerator

- The following commands apply to the linear gantry axis in case the command is executed on Axis[0], or to the rotational gantry axis in case the command is executed on Axis[1]:
  - StartPositionSin, StartPositionSquare, StartCurrentSin, StartCurrentSquare.
- The following commands apply to the motor of Axis[0] or Axis[1] respectively:
  - StartVoltageSin, StartCurrentSinStaticVector, StartVoltageSinStaticVector, StartRotatingVectorConstantCurrent, StartRotatingVectorConstantVoltage.

Commands.Homing:

- To start the homing sequence, Start has to be selected in the Command register of Axis[0]. The Start command on Axis[1] is not supported and must not be used.
- The commands used with absolute encoders (Invalidate, SaveEncoder, InvalidateEncoder) apply to the encoder of the corresponding axis.
- ReferencePosition: After homing, the position of both encoders is set in such a way that the position at the reference corresponds to the value of ReferencePosition. This value is persistently stored on the drive, but *not* in the **TAM Configuration**.

3.4 Signals with New Assignment

This section describes the signals of Axis[0] and Axis[1] which change their assignment with the gantry setup.

Signals.PathPlanner:

- The signals of Axis[0] show the state of the linear path planning and the signals of Axis[1] the state rotational path planning.

Signals.PathInterpolator:

- The signals of Axis[0] show the state of the linear interpolator and the signals of Axis[1] the state of the rotational interpolator.

Signals.PositionController:

- Encoders: The meaning of the encoder signals does not change with the gantry topology and they do still show the states of the assigned physical encoder.
- Controllers, FeedForward, MasterPosition, MasterError, MasterVelocity, DesiredCurrentQ, DesiredCurrentD: The parameters of Axis[0] are assigned to the linear axis, the parameters of Axis[1] to the rotational axis.

Signals.TestGenerator:

- The assignment depends on the selected command. See Commands.TestGenerator in section 3.3.

References

[1] “Servo Drive Setup Guide, TSD and TSP Series”, ServoDrive-SetupGuide_EP007.pdf, Triamec Motion AG, 2021.

[2] “Encoder configuration for the TSD drive series”, AN107_Encoder_EP009.pdf, Triamec Motion AG, 2020.

[3] "Twincat Library: Cross Publishing data, Application Note", AN106_TwinCAT-CrossPublishing_EP002.pdf, Triamec Motion AG, 2016.

[4] "Twincat Library: Defining cyclic telegrams, Application Note", AN105_TwinCAT-CyclicTelegrams_EP004.pdf, Triamec Motion AG, 2020.

[5] "Twincat Library: Accessing Drive Registers, Application Note", AN109_TwinCAT-AccessingDriveRegisters_EP006.pdf, Triamec Motion AG 2019.

[6] "Twincat Library: Defining cyclic telegrams, Application Note AN104", AN104_EtherCAT-CyclicTelegrams_EP002.pdf, Triamec Motion AG, 2020.