

Force/Torque sensors CBun Software

This document is a CBun manual for Bota Systems
Force/Torque sensors



Revision

1.0

1 Table of Contents

1	Table of Contents.....	2
2	Safety.....	4
2.1	Explanation of notes	4
2.2	General safety guidelines.....	4
2.3	Safety precautions	5
3	Sensor overview	6
4	Installation	7
4.1	Mounting of the sensor.....	7
4.2	Installation of CBun software	8
4.3	Uninstalling the CBun software	8
5	Software	9
5.1	Activate Sensor	9
5.2	FT_SENSOR Methods	11
5.2.1	FT_SENSOR: tareSensor.....	11
5.2.2	FT_SENSOR: resetForce(Array<Number> forces).....	11
5.2.3	FT_SENSOR: resetTorque(Array<Number> torque).....	11
5.2.4	FT_SENSOR: resetForceSensorFrame(Array<Number> forces).....	11
5.2.5	FT_SENSOR: resetTorqueSensorFrame(Array<Number> torq.)	11
5.2.6	FT_SENSOR: resetForceRaw(Array<Number> forces).....	11
5.2.7	FT_SENSOR: resetTorqueRaw(Array<Number> forces).....	11
5.2.8	FT_SENSOR: readForce(Array<Number> forces)	11
5.2.9	FT_SENSOR: readTorque(Array<Number> torques)	11
5.2.10	FT_SENSOR: readForceSensorFrame(Array<Number> forces).....	11
5.2.11	FT_SENSOR: readTorqueSensorFrame(Array<Number> torq.)	12
5.2.12	FT_SENSOR: readFilteredForce(Array<Number> forces)	12
5.2.13	FT_SENSOR: readFilteredTorque(Array<Number> torques)	12
5.2.14	FT_SENSOR: readForceRaw(Array<Number> forces).....	12
5.2.15	FT_SENSOR: readTorqueRaw(Array<Number> torques).....	12
5.2.16	FT_SENSOR: setEnabledSystemFilters(bool hprl)	12
5.2.17	FT_SENSOR: setSystemFilterParams(Number cutoff_frequency, Number rise_slew_rate, Number fall_slew_rate).....	12
5.2.18	FT_SENSOR: addFilter(string filter_name, int filter_type, double reject_frequency, double reject_bandwidth, double cutoff_frequency, double rise_slew_rate, double fall_slew_rate)	12
5.2.19	FT_SENSOR: removeFilter(string filter_name)	13
5.2.20	FT_SENSOR: resetFilters()	13
5.2.21	FT_SENSOR: setForceControlGains(Number p_gain_trans, Number p_gain_rot, Number track_wrench_smoothing_coeff.)	13
5.2.22	FT_SENSOR: activateForceControl(bool enable_translation, bool enable_rotation)	13
5.2.23	FT_SENSOR: deactivateForceControl()	13

5.2.24	FT_SENSOR: setEnabledAxes(bool fx, bool fy, bool fz, bool tx, bool ty, bool tz).....	13
5.2.25	FT_SENSOR: setForceTrackValue(Array<Number> forces).....	13
5.2.26	FT_SENSOR: setTorqueTrackValue(Array<Number> torques)	13
5.2.27	FT_SENSOR: moveToContact(int frame, int motion_direction, Number motion_vel, Number contact_force, Number max_distance, Number retract_distance)	13
5.3	Device Dashboard	14
6	Example Programs	15
6.1	Move to contact and apply force during motion	15
7	Sensor Maintenance	16
7.1	Inspection.....	16
7.2	Calibrating	16

2 Safety

The safety section describes general safety guidelines for the product(s), an explanation of the notifications found in this manual, and the safety precautions applicable to the product(s). More specific notifications are embedded within the sections of the manual where they apply.

2.1 Explanation of notes

The notifications included here are specific to the products covered by this manual. The user should also be aware of the notifications of other components from other manufacturers installed in the system /robot.



DANGER: Indicate[s] a hazardous situation which, if not avoided, will result in death or serious injury. The signal word "DANGER" is to be limited to the most extreme situations. DANGER [signs] should not be used for property damage hazards unless personal injury risk appropriate to these levels is also involved.



WARNING: Indicate[s] a hazardous situation which, if not avoided, could result in death or serious injury. WARNING [signs] should not be used for property damage hazards unless personal injury risk appropriate to this level is also involved.



CAUTION: Indicate[s] a hazardous situation which, if not avoided, could result in minor or moderate injury. CAUTION [signs] without a safety alert symbol may be used to alert against unsafe practices that can result in property damage only.



NOTICE: Notification of specific information or instructions about maintaining, operating, installation, or setup of the product that if not followed could result in damage to equipment. The notification can emphasize but is not limited to specific grease types, good operating practices, or maintenance tips.

2.2 General safety guidelines

The user should verify that the Force/Torque sensor is rated for maximum loads and torques expected from the operation. The user should be aware of the dynamic loads caused by the robot during acceleration or deceleration of the mounted masses.

2.3 Safety precautions



WARNING: Performing maintenance or repair on the sensor, while circuits (e.g. power, water, and air) are energized could result in serious injury. Discharge and verify all circuits are deactivated in accordance with the user's safety practices and policies.



CAUTION: Modifying or disassembly of the sensor could cause damage. Use the robot or adapter mounting bolt pattern and the tool side mounting bolt pattern to mount the sensor to the robot and user tooling to the sensor. Refer to the Force/Torque sensor model drawings and specifications sheet for more information.



CAUTION: Using fasteners that exceed the user mounting bolt pattern interface depth penetrates the body of the sensor, potentially damaging the electronics. Refer to the Force/Torque sensor model drawings and specifications for more information.



CAUTION: Do not overload the sensor. Exceeding the single-axis overload values of the sensor, causes irreparable damage.

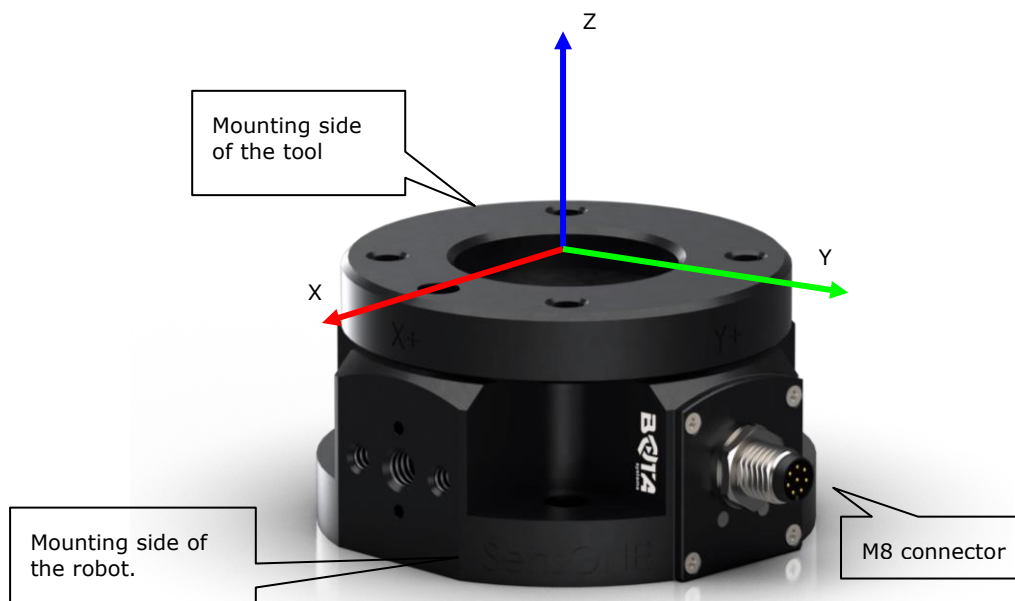


CAUTION: Overload values refer to static overloading the sensor. This shouldn't be confused with the dynamic loading. The sensor should be dynamically loaded at the rated values of force and torque.

3 Sensor overview

The Bota Systems F/T sensor system measures (6) components of force and torque (F_x , F_y , F_z , M_x , M_y , M_z) and streams data to user devices that use Serial or EtherCAT communication. The sensor has two mounting sides. These sides are the connecting mechanical interfaces of the sensor. Refer to the Force/Torque sensor model drawings and specifications for more information. The sensor is IP67 rated. An IP67 rated connector is for the cable assembly provided with the sensor. The sensor is powered through this connector. For the electrical connector pin assignments, refer to the user manual. The Bota Systems sensors provide resolved force and torque data measured in N and Nm accordingly. Each F/T sensor model has its own reference frame which is shown in its specification sheet.

Figure 1 – Example sensor Illustration : SensONE



NOTICE

NOTICE: The Sensor is powered through the cable connector. The USB version is powered directly from the USB port.

4 Installation

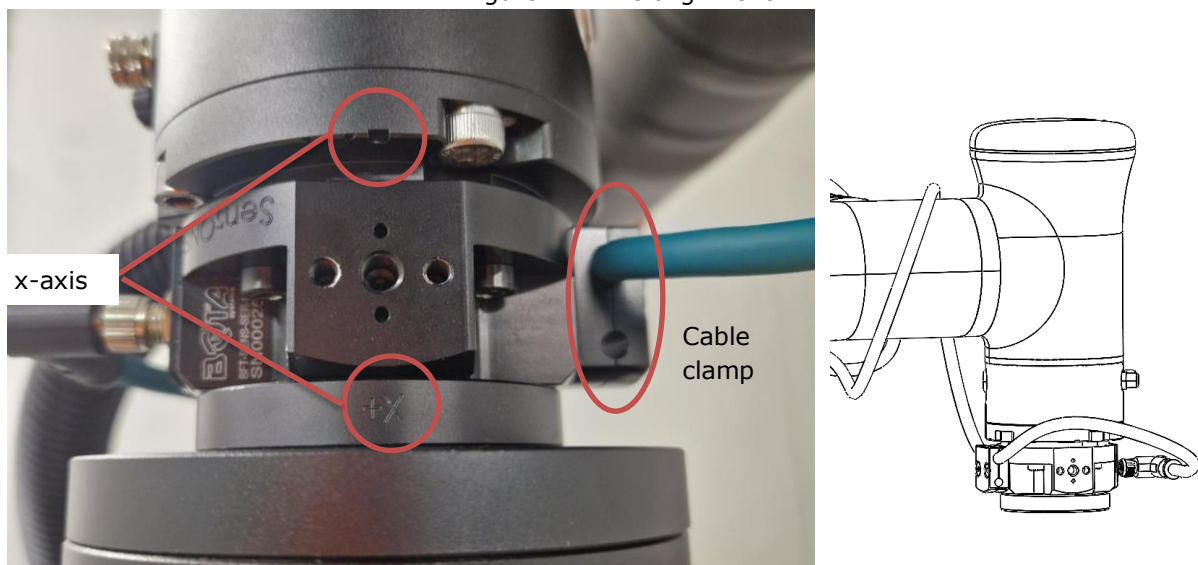
The Bota Systems CRun software is tested for Fire Fly 2 software.

4.1 Mounting of the sensor

For instruction about mounting the sensor please have a look at our step-by-step tutorial on how [to mount the sensor to the robot flange](#). Further details can be found in the [Kassow Kit datasheet](#).

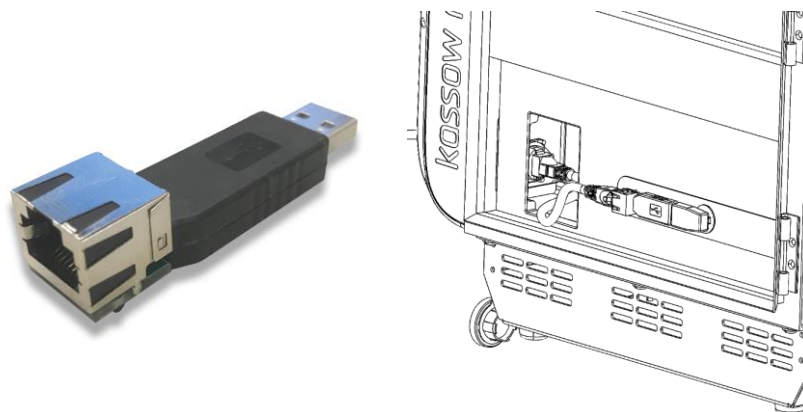
The sensor can be directly mounted to the robot flange without adapters. Please make sure to only use the provided screws for tightening the sensor. All screws are accessible from the mounting side of the tool for tightening. Make sure to align the sensors x- and y-axis with the robot's tool frame. (It might be possible that the provided pin for the alignment of the sensor is not correct). It is possible to mount an intermediate part between the robot flange and the sensor (e.g. a tool changer).

Figure 2 – Axis alignment



To avoid damage on the connector, we recommend to attach the sensor cable to the housing with the provided cable clamp (see Fig. 2). The sensor needs to be connected to the USB port of the controller with the provided USB-to-serial adapter (see Fig. 3). It is possible to have an USB extension or hub to connect multiple devices to it.

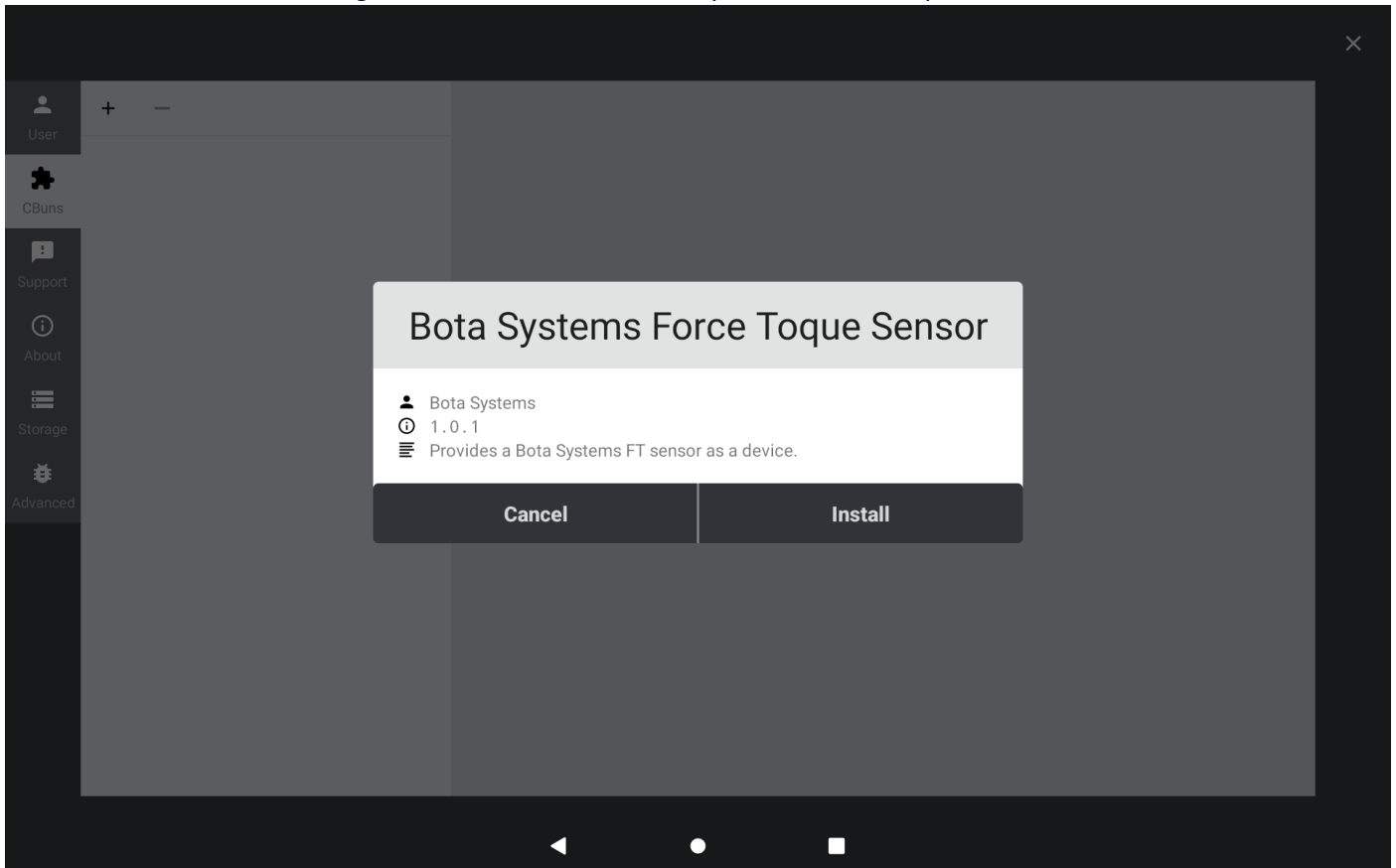
Figure 3 - USB to Serial adapter RS422 to RJ45 and Control Box mounting



4.2 Installation of CBun software

- Copy the file bota_device.cbun to a USB stick
- Insert the USB stick in the robot's USB port and install the file like a standard CBun.
 - Three dots on the top right
 - Settings
 - CBuns

Figure 4 – Install CBun "Bota Systems Force Torque Sensor"



4.3 Uninstalling the CBun software

- You can uninstall the Bota Systems CBun (Bota Systems Force Torque Sensor) like a standard CBun

5 Software

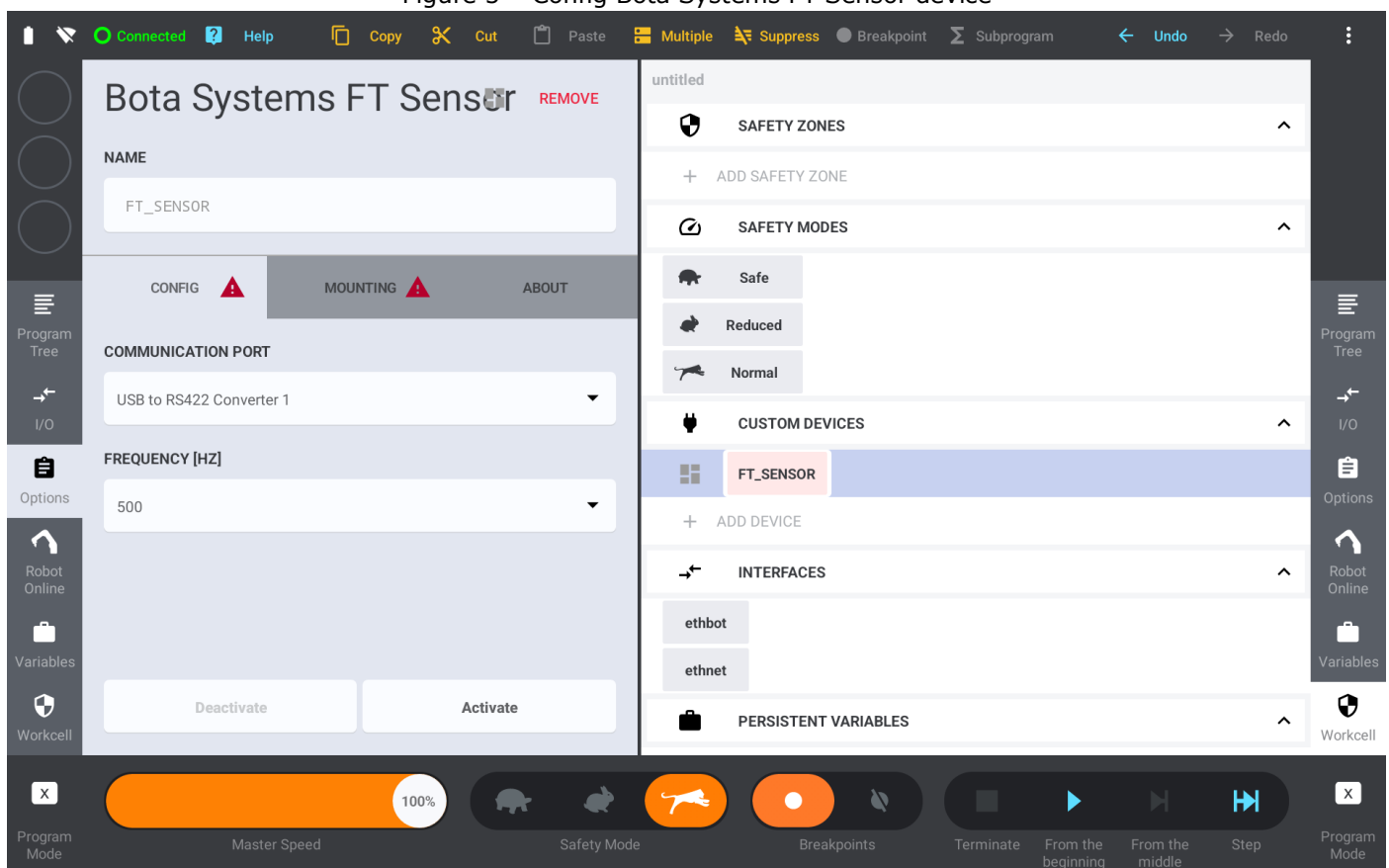
In order to write programs that are using the Bota Systems force torque sensor, the sensor (Bota Systems Force Torque Sensor) needs to be added as device (FT_SENSOR) to your configuration.

5.1 Activate Sensor

In the config tab of the device, you can activate the sensor. Before activating the Bota Systems force torque sensor, take care to adjust the settings. You have the following parameters:

- COMMUNICATION PORT:
 - USB to RS422 Converter 1 (selecting port /dev/ttyUSB0)
 - USB to RS422 Converter 2 (selecting port /dev/ttyUSB1)
- FREQUENCY [Hz]
 - 500 Hz (default)
 - 250 Hz
 - 125 Hz

Figure 5 – Config Bota Systems FT Sensor device



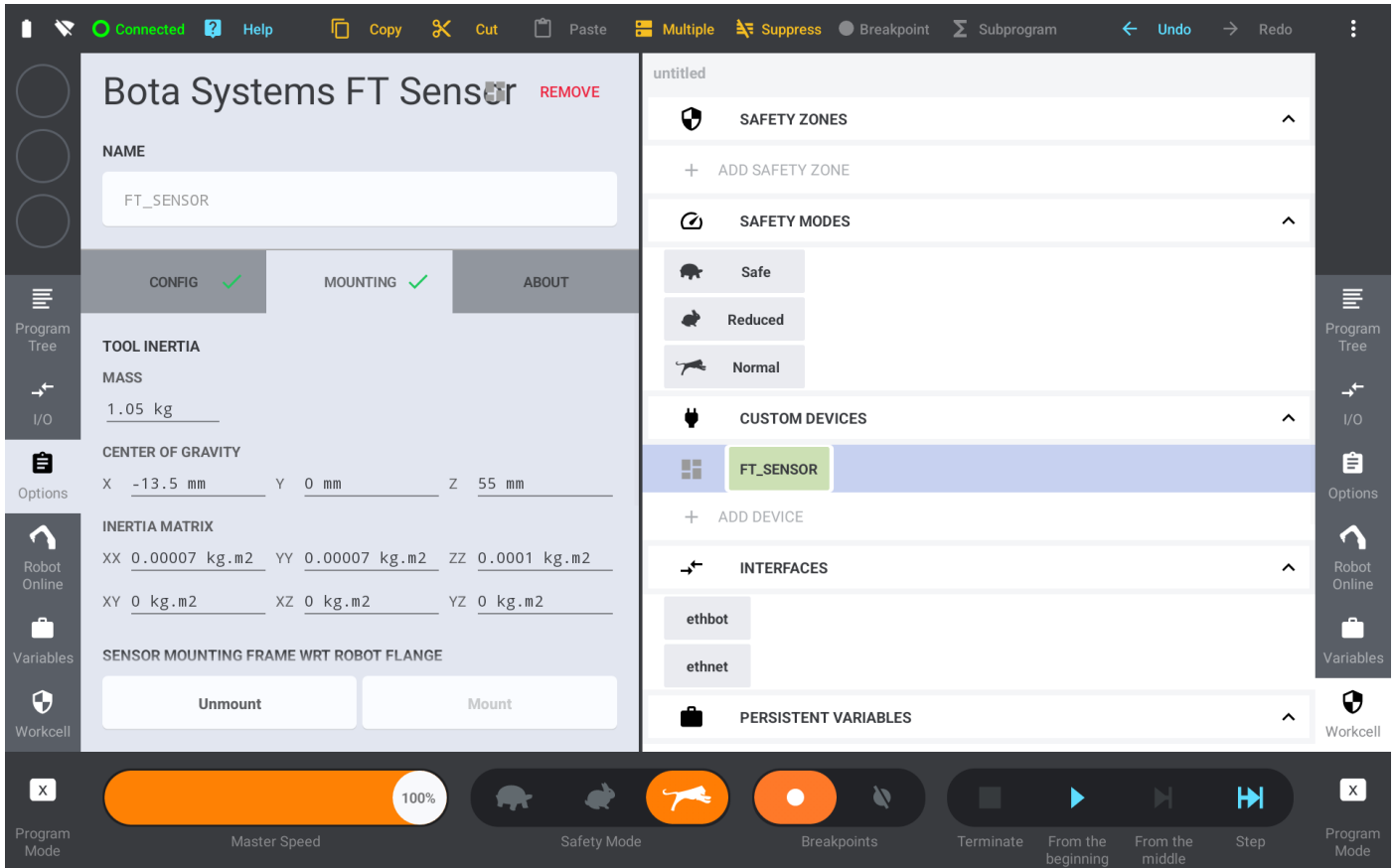
Pressing the activate button will configure the sensor, which takes ~3 seconds. During configuration, the green LED on sensor is blinking. After activation, the green LED on the sensor is running constantly. If the LED is not running, please try the other communication port or contact support.

In the mounting tab , please insert the load of the attached tool to the sensor (without the weight of the sensor). If the tool load is set through a different device, put zero weight and center of gravity.

The following parameters need to be defined on mounting the sensor:

- Tool Inertia
- Sensor frame with respect to robot flange (default values are for SensONE)

Figure 6 – Mount Bota Systems FT Sensor device



5.2 FT_SENSOR Methods

The following methods are available for the FT_SENSOR device. Note that the force control and readings are currently only available in tool frame.

5.2.1 FT_SENSOR: tareSensor

Tare the sensor readings (remove the bias) and set the output to zero in the **TCP frame**. This method is recommended to call before starting any method using force torque measurements. Note that the readings are compensated with the gravitational load.

5.2.2 FT_SENSOR: resetForce(Array<Number> forces)

Set force output of sensor to the desired values in the **TCP frame**. Note that the readings are compensated with the gravitational load.

5.2.3 FT_SENSOR: resetTorque(Array<Number> torque)

Set torque output of sensor to the desired values in the **TCP frame**. Note that the readings are compensated with the gravitational load.

5.2.4 FT_SENSOR: resetForceSensorFrame(Array<Number> forces)

Set force output of sensor to the desired values in the **sensor frame**. Note that the readings are compensated with the gravitational load.

5.2.5 FT_SENSOR: resetTorqueSensorFrame(Array<Number> torq.)

Set torque output of sensor to the desired values in the **sensor frame**. Note that the readings are compensated with the gravitational load.

5.2.6 FT_SENSOR: resetForceRaw(Array<Number> forces)

Set force output of sensor to desired value. Note that the readings are NOT compensated with the gravitational load, meaning that you will measure the weight of the tool.

5.2.7 FT_SENSOR: resetTorqueRaw(Array<Number> forces)

Set torque output of sensor to desired value. Note that the readings are NOT compensated with the gravitational load, meaning that you will measure the weight of the tool.

5.2.8 FT_SENSOR: readForce(Array<Number> forces)

Read out the latest **sensor forces in the TCP frame**. Note that the readings are compensated with the gravitational load.

5.2.9 FT_SENSOR: readTorque(Array<Number> torques)

Read out the latest **sensor torques in the TCP frame**. Note that the readings are compensated with the gravitational load.

5.2.10 FT_SENSOR: readForceSensorFrame(Array<Number> forces)

Read out the latest **sensor forces in the sensor frame**. Note that the readings are compensated with the gravitational load.

5.2.11 FT_SENSOR: readTorqueSensorFrame(Array<Number> torq.)

Read out the latest **sensor torques in the sensor frame**. Note that the readings are compensated with the gravitational load.

5.2.12 FT_SENSOR: readFilteredForce(Array<Number> forces)

Read out the latest **filtered sensor forces** in the TCP frame. Note that the readings are compensated with the gravitational load. These readings are used for the force control.

5.2.13 FT_SENSOR: readFilteredTorque(Array<Number> torques)

Read out the latest **filtered sensor torques** in the TCP frame. Note that the readings are compensated with the gravitational load. These readings are used for the force control.

5.2.14 FT_SENSOR: readForceRaw(Array<Number> forces)

Read out the latest sensor forces **without gravitational load compensation** in the sensor frame.

5.2.15 FT_SENSOR: readTorqueRaw(Array<Number> torques)

Read out the latest sensor torques **without gravitational load compensation** in the sensor frame.

5.2.16 FT_SENSOR: setEnabledSystemFilters(bool hprl)

Enable /disable internal system filters. The sensor has a system filter based on the high-pass rate limiter as discussed here¹. Default value is hprl: true

5.2.17 FT_SENSOR: setSystemFilterParams(Number cutoff_frequency, Number rise_slew_rate, Number fall_slew_rate)

Set parameters for internal high-pass rate limiter. Default values are:

- cutoff_frequency: 50
- rise_slew_rate: 50
- fall_slew_rate: -50

5.2.18 FT_SENSOR: addFilter(string filter_name, int filter_type, double reject_frequency, double reject_bandwidth, double cutoff_frequency, double rise_slew_rate, double fall_slew_rate)

Add a filter from a list of filter types (low-pass filter, notch filter, rate limiter and high-pass rate limiter). Each filter added is expected to have a unique filter_name from the user. It is possible to have multiple filters of a type if they all have unique names. Filters are applied to the signal before the system filter, in the order they were added to the program. Filters are built only using parameters relevant to them; all other parameters are ignored. Relevant parameters and filters are listed as follows, cutoff_frequency for low-pass filters, reject_frequency and reject_bandwidth for notch filters, rise_slew_rate and fall_slew_rate for rate limiters, and, cutoff frequency, rise_slew_rate and fall_slew_rate for high-pass rate limiters.

Default values are:

- filter_name: Filter1
- filter_type: Low-Pass Filter
- reject_frequency: 0.0
- reject_bandwidth: 0.0
- cutoff_frequency: 0.0
- rise_slew_rate: 0.0
- fall_slew_rate: 0.0

¹ [Englsberger, Johannes & Mesesan, George & Werner, Alexander & Ott, Christian. \(2018\). Torque-Based Dynamic Walking - A Long Way from Simulation to Experiment. 10.1109/ICRA.2018.8462862.](#)

5.2.19 FT_SENSOR: removeFilter(string filter_name)

Remove any filter with the given filter_name. If filter_name is set to "all", all applied filters except for system filter are removed. In case a filter with the given name does not exist, the program continues without error.

5.2.20 FT_SENSOR: resetFilters()

Reset filter histories.

5.2.21 FT_SENSOR: setForceControlGains(Number p_gain_trans, Number p_gain_rot, Number track_wrench_smoothing_coeff.)

Set parameters for the admittance control. **Important:** call this method before activating force control. Default values are p_gain_trans: 5, p_gain_rot: 5, track_wrench_smoothing_coefficient: 0.01

5.2.22 FT_SENSOR: activateForceControl(bool enable_translation, bool enable_rotation)

Start the admittance control (in tool frame) through ARTO. You can separately activate compliance in translation or rotation or both. The admittance control is active as soon as the trajectory (motion) starts.

5.2.23 FT_SENSOR: deactivateForceControl()

Stop the admittance control through ARTO.

5.2.24 FT_SENSOR: setEnabledAxes(bool fx, bool fy, bool fz, bool tx, bool ty, bool tz)

Enable/disable compliance of single axis in the admittance control (in tool frame)

5.2.25 FT_SENSOR: setForceTrackValue(Array<Number> forces)

Set the desired reference force of the admittance control in the TCP frame.

5.2.26 FT_SENSOR: setTorqueTrackValue(Array<Number> torques)

Set the desired reference torque of the admittance control in the TCP frame.

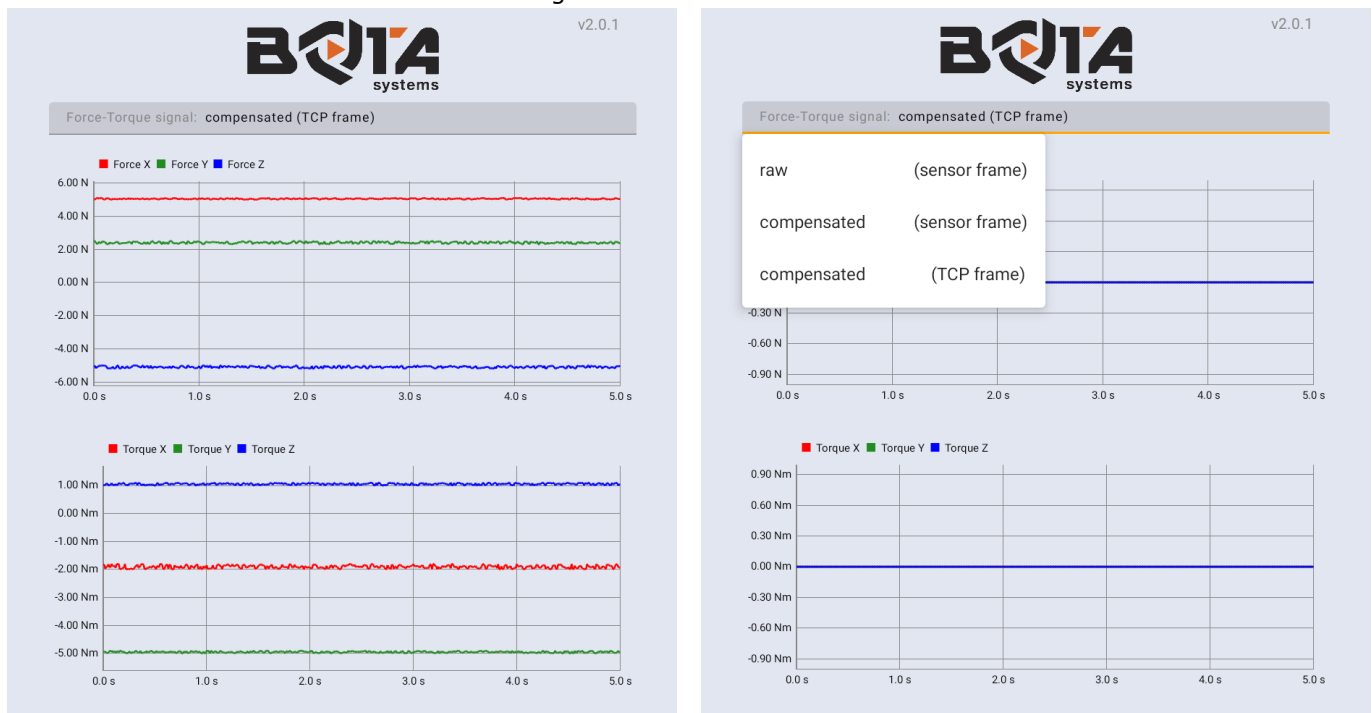
5.2.27 FT_SENSOR: moveToContact(int frame, int motion_direction, Number motion_vel, Number contact_force, Number max_distance, Number retract_distance)

Move linearly until the contact force is reached. Retract by the retract distance afterwards in the opposite motion direction. This method is intended to move the robot into contact with the environment. The robot can be moved linearly either in TCP frame or in World frame in all motion directions.

5.3 Device Dashboard

Attention! The device dashboard is only available in the Beta version of the Cbun. The Cbun for Bota Systems sensors comes with a device dashboard for online monitoring. Clicking the dashboard icon launches the device dashboard on the opposite side of the screen.

Figure 7 – Device Dashboard



The dashboard shows the live view of the force torque readings. Different signals can be selected:

- Raw sensor measurements (for debugging)
- Measurement in sensor frame with gravitational load compensation
- Measurement in TCP frame with gravitational load compensation

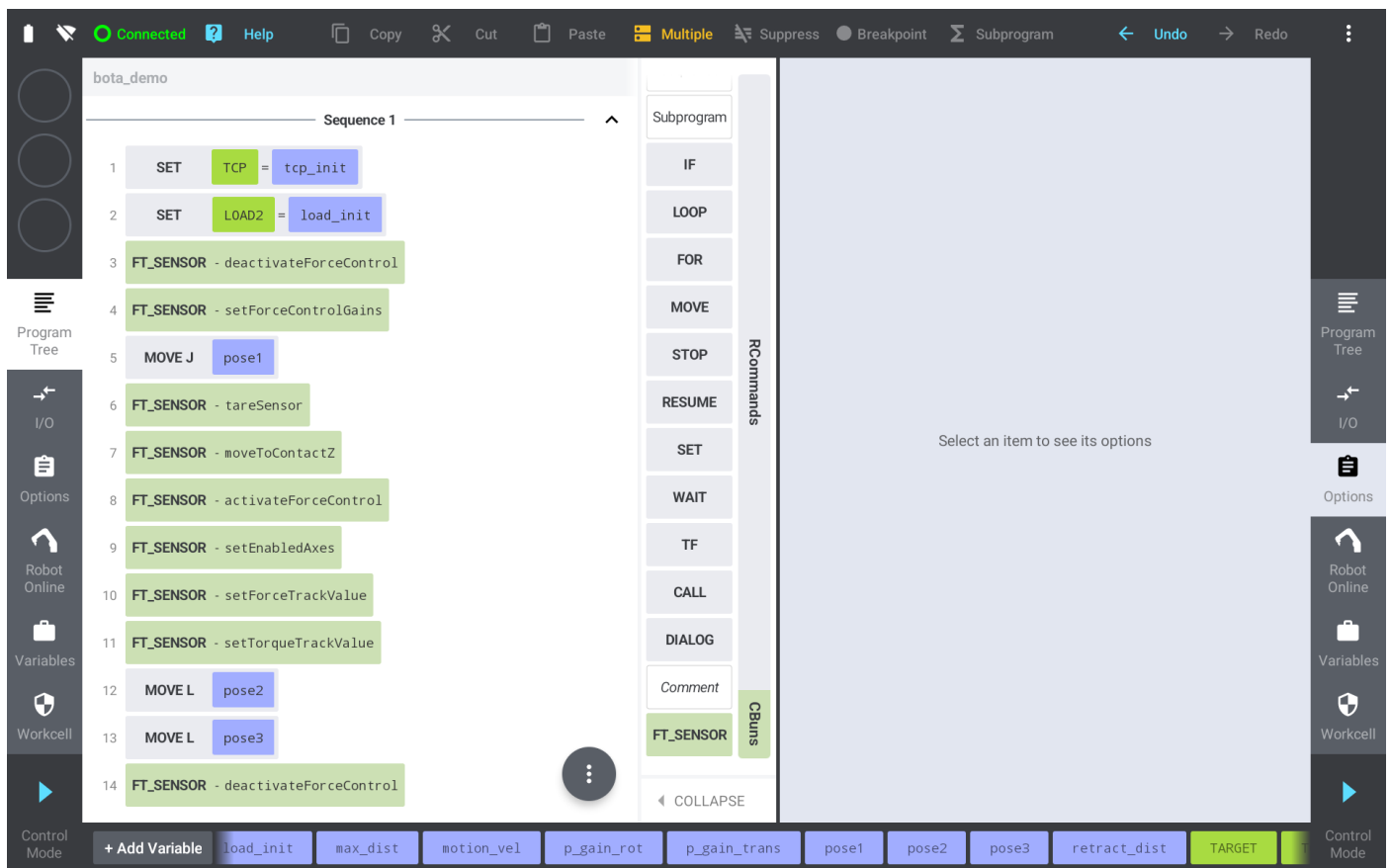
6 Example Programs

There are example programs distributed together with the CBun. They provide an overview of how to use the with activated force control (bota_simple_example.kr2) or how to filter and monitor the sensor data (bota_filter_example.kr2).

6.1 Move to contact and apply force during motion

This section gives an outline how a program with force control can be structured. Generally a program involving force control should involve the following steps in this order or similarly:

FT_SENSOR – deactivateForceControl	// make sure force control is disabled at the beginning of the program
FT_SENSOR – setForceControlGains	// Set the gains of the force control
MOVE J/L	// Move close to the contact point
WAIT 0.5 seconds	// Wait until oscillations stop
FT_SENSOR – tareSensor	// Set sensor to zero
FT_SENSOR – moveToContactZ	// Move to contact
FT_SENSOR – activateForceControl	// Enable admittance control
FT_SENSOR – setEnabledAxes	// Allow/restrict compliance in certain directions
FT_SENSOR – setForceTrackValue	// Set desired tracking force
FT_SENSOR – setTorqueTrackValue	// Set desired tracking torque
MOVE L	// Add motion under force control
...	
MOVE L	
FT_SENSOR – deactivateForceControl	// stop admittance control



7 Sensor Maintenance

7.1 Inspection

If the sensor is used in applications that frequently move the system's cabling and dismount the connector, the cable jacket and connector pins should be checked for signs of wear. The Bota Systems sensors are IP68 equivalent. However, debris and dust should be kept from accumulating on the sensor. The sensor can be cleaned with a wet towel.

7.2 Calibrating

Periodic calibration of the sensor and its electronics is required to maintain traceability to international standards. Applicable ISO-9000-type standards for calibration should be followed. There is no fixed re-calibration interval suggested, but the calibration should be done if reference measurements with weights show degeneration of the signal accuracy. The best practice would be to send it back to us for recalibration.