UNIVERSAL ROBOTS

PolyScope Manual

Version 5.3

Original instructions (en)

US version

Contents

Part II

PolyScope Manual

10 Introduction

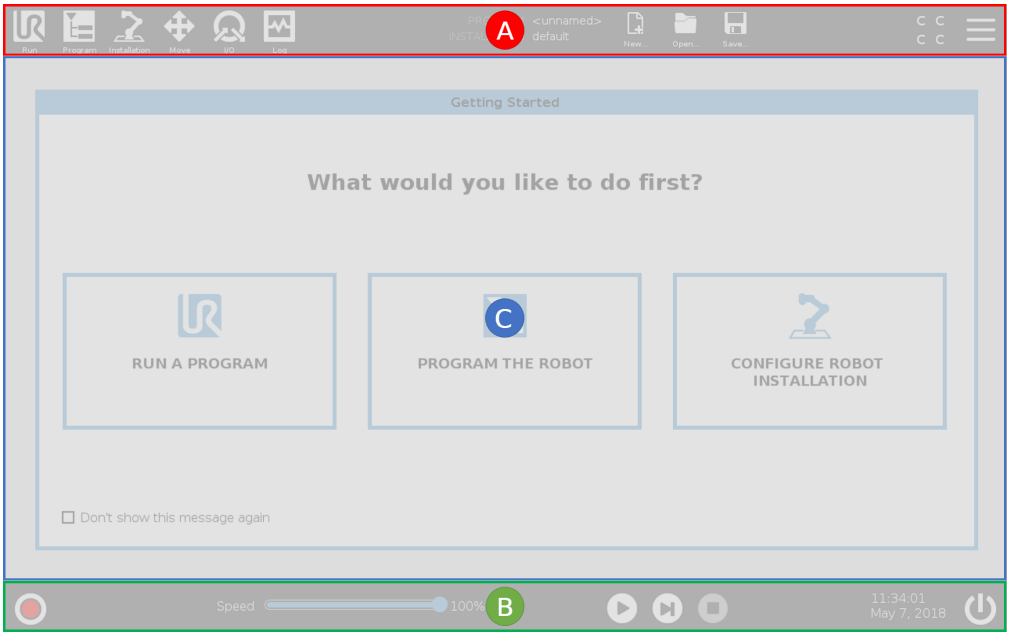
10.1 PolyScope Basics

PolyScope is the Graphical User Interface (GUI) on the **Teach Pendant** that operates the Robot Arm, Control Box and executes programs.

A : **Header** with tabs/icons that make interactive screens available to you.

B : **Footer** with buttons that control your loaded program/s.

C : **Screen** with fields that manage and monitor robot actions.



Note: On start up a Cannot Proceed dialogue can appear. You must select **Go to initialization screen** to turn on robot.

10.1.1 Header Icons/Tabs

Run is a simple means of operating the robot using pre-written programs.

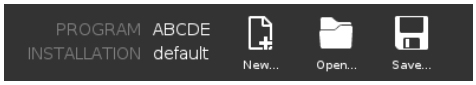
Program creates and/or modifies robot programs.

Installation configures robot arm settings and external equipment e.g. mounting and safety.

Move controls and/or regulates robot movement.

I/O monitors and sets live Input/Output signals to and from robot control box.

Log indicates robot health as well as any warning or error messages.



Program and Installation Manager selects and displays active program and installation (see 20.4). Note: File Path, New, Open and Save make up the Program and Installation Manager.

New... creates a new Program or Installation.

Open... opens a previously created and saved Program or Installation.

Save... saves a Program, Installation or both at the same time.

Note: Automatic mode and Manual mode icons only appear in the Header if you set a operational mode password.


Automatic indicates that the robot has Automatic environment loaded. Click it to switch to Manual environment.


Manual indicates that the robot has Manual environment loaded. Click it to switch to Automatic environment.


Safety Checksum displays the active safety configuration.


Hamburger Menu accesses PolyScope Help, About and Settings.

10.1.2 Footer Buttons


Initialize manages robot state. When RED, press it to make the robot operational.

Speed Slider shows in real time the relative speed at which the robot arm moves, taking safety settings into account.

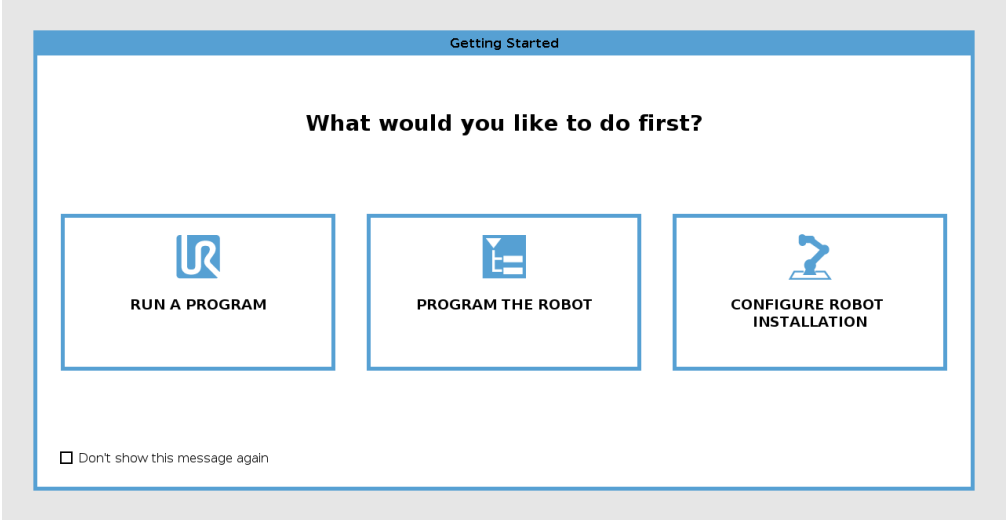
Simulation button toggles a program execution between Simulation Mode and the Real Robot. When running in Simulation Mode, the Robot Arm does not move. Therefore, the robot cannot damage itself or nearby equipment in a collision. If you are unsure what the Robot Arm will do, use Simulation Mode to test programs.

Play starts current loaded robot Program.

Step allows a Program to be run single-stepped.

Stop halts current loaded robot Program.

10.2 Getting Started Screen



Run a Program, Program the Robot or Configure Robot Installation.

11 Quick Start

11.1 Robot Arm Basics

The Universal Robot arm is composed of tubes and joints. You use the PolyScope to coordinate the motion of these joints, moving the robot and positioning its tool as desired - except for the area directly above and directly below the base.

Base is where the robot is mounted.

Shoulder and Elbow make larger movements.

Wrists 1 and 2 make finer movements.

Wrist 3 is where you attach the robot tool.

NOTE:
Before powering on the robot for the first time, your designated UR robot integrator must:

1. Read and understand the safety information in the Hardware Installation Manual.
2. Set the safety configuration parameters defined by the risk assessment (see chapter 13).

11.1.1 Installing the Robot Arm and Control Box

You can use PolyScope, once the Robot arm and Control Box are installed and switched on.

1. Unpack the **Robot arm** and the **Control box**.
2. Mount Robot arm on a sturdy surface strong enough to withstand at least 10 times the full torque of the base joint and at least 5 times the weight of the robot arm. The surface must be vibration-free.
3. Place the **Control box** on its **Foot**.
4. Connect the cable to the robot and the control box.
5. Plug in the main control box plug.

DANGER:
Tipping hazard. If the robot is not securely placed on a sturdy surface, the robot can fall over and cause injury.

See Hardware Installation Manual for detailed installation instructions.

11.1.2 Turning Control Box On/Off

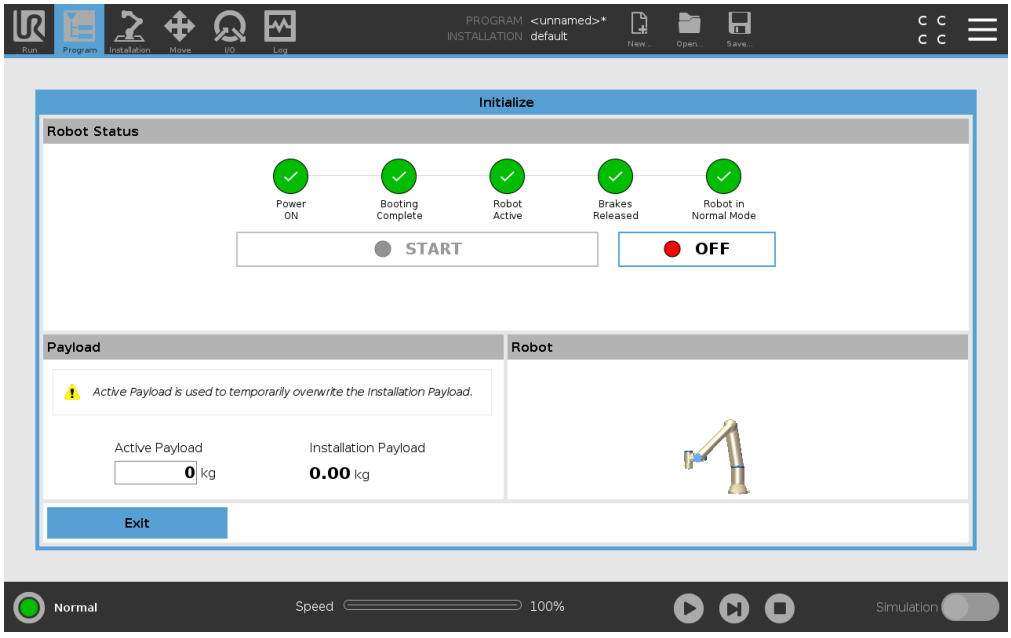
The Control Box mainly contains the physical electrical Input/Output that connects the Robot arm, the Teach Pendant and any peripherals. You must turn on the Control Box to be able to power on the Robot arm.

1. On your **Teach Pendant**, press the power button to turn on control box.
2. Wait as text from the underlying operating system, followed by buttons, appear on the screen.
3. When a Cannot Proceed dialog appears, select **Go to initialization screen** to access Initialize screen.

11.1.3 Turning Robot Arm On/Off

On the bottom left of the screen, the Initialize icon indicates the status of the robot arm using colors:

- **Red** The Robot arm is in a stopped state.
- **Yellow** The Robot arm is on, but not ready for normal operation.
- **Green** The Robot arm is on and ready for normal operation.



Note: Robot arm start up is accompanied by sound and slight movements as joint brakes are released.

11.1.4 Initializing the Robot Arm

DANGER:
Always verify the actual payload and installation are correct before starting up the Robot arm. If these settings are incorrect, the Robot arm and Control Box will not function correctly and may become dangerous to people or equipment.

CAUTION:
Ensure the Robot arm is not touching an object (e.g., a table) because a collision between the Robot arm and an obstacle might damage a joint gearbox.

To start the robot:

1. Tap the ON button with the green LED to start the initialization process. Then, the LED turns yellow to indicate the power is on and in **Idle**.
2. Tap the START button to release the breaks.
3. Tap the OFF button with the red LED to power off the Robot arm.

- When the PolyScope starts, tap the ON button once to power the Robot arm. Then, the status changes to yellow to indicate the robot is on and in idle. **Idle**.
- When the Robot arm state is **Idle**, tap the START button to start Robot arm. At this point, sensor data is checked against the configured mounting of the Robot arm. If a mismatch is found (with a tolerance of $30°$), the button is disabled and an error message is displayed below it. If the mounting is verified, tapping Start releases all joint brakes and the Robot arm is ready for normal operation.

12 Operational Mode Selection

12.1 Operational Modes

Operational Modes are enabled when you configure a 3-Postition Enabling Device, or set a password.

Automatic Mode Once activated, the robot can only perform pre-defined tasks. The Move Tab and Freedrive Mode are unavailable. You cannot modify or save programs and installations.

Manual Mode Once activated, you can program the robot using the Move Tab, Freedrive Mode and Speed Slider. You can modify or save programs and installations.

Operational mode	Manual	Automatic
Freedrive	x	*
Move robot with arrows on MoveTab	x	*
Speed Slider	x	x**
Edit & save program & installation	x	
Execute Programs	Reduced speed*	x

*Only when a 3-position enabling device is configured
** The Speed Slider on the Run Screen can be enabled in the Installation.

NOTE:

- A Universal Robots robot is not equipped with a 3-Position Enabling Device. If the risk assessment requires the device, it must be attached before the robot is used.
- If a 3-Position Enabling Device is not configured, both Freedrive and Move Tab are enabled. Speed is not reduced in Manual Mode.

WARNING:

- Any suspended safeguards must be returned to full functionality before selecting Automatic Mode.
- Wherever possible, the Manual Mode of operatio shall be performed with all persons outside the safeguard space.
- The device used to switch the robot to Operational Mode must be placed outside the safeguarded space.
- The user must not enter the safeguarded space when robot is in Automatic Mode.

The three methods for configuring Operational Mode selection are described in the following subsections. Each method is exclusive, meaning that using one method, makes the other two methods inactive.

Using Operational Mode Safety Input

1. In the Installation Tab, select Safety I/O.
2. Configure the Operational Mode Input. The option to configure appears in the drop-down menu once the 3-Position Enabling Device input is configured.
3. The robot is in Automatic Mode when the Operational Mode Input is low and in Manual Mode when the Operational Mode Input is high.

NOTE:

- The physical mode selector, if used, must completely adhere to ISO 10218-1: article 5.7.1 for selection.
- Before defining an operational input, you must define a 3-Position Enabling Device.

Using PolyScope

1. In PolyScope, select an Operational Mode.
2. To switch between modes, in the Header, select the profile icon.

See 21.3.2 for more information on setting a PolyScope password.

Note: PolyScope is automatically in Manual Mode when the Safety I/O configuration with 3-Position Enabling Device is enabled.

Using Dashboard Server

1. Connect to the Dashboard server.

2. Use the **Set Operational Mode** commands.
   - Set Operational Mode Automatic
   - Set Operational Mode Manual
   - Clear Operational Mode

See `http://universal-robots.com/support/` for more on using the Dashboard server.

12.2 3-Position Enabling Device

When a 3-Position Enabling Device is configured and the **Operational Mode** is in Manual Mode, the robot can only be moved by pressing the 3-Position Enabling Device.
When the **Operational Mode** is in Automatic Mode, the 3-Position Enabling Device has no effect.

NOTE:
The Three-Position Enabling Device complies with ISO 10218-1: article 5.8.3 for an enabling device.

12.2.1 Manual High Speed

When the Three-Position Enabling Device is not pressed, the robot is in Safeguard Stop. When the Three-Position Enabling Device is pressed the Speed Slider is set to an initial value corresponding to 250 mm/s. The speed can be increased by moving the Speed Slider in increments.

NOTE:
Use safety joint limits `(see 13.2.4)` or safety planes `(see 13.2.5)`to restrict the space the robot can move in while utilizing manual high speed.

13 Safety Configuration

13.1 Safety Settings Basics

This section covers how to access the robot safety settings. It is made up of items that help you set up the robot Safety Configuration.

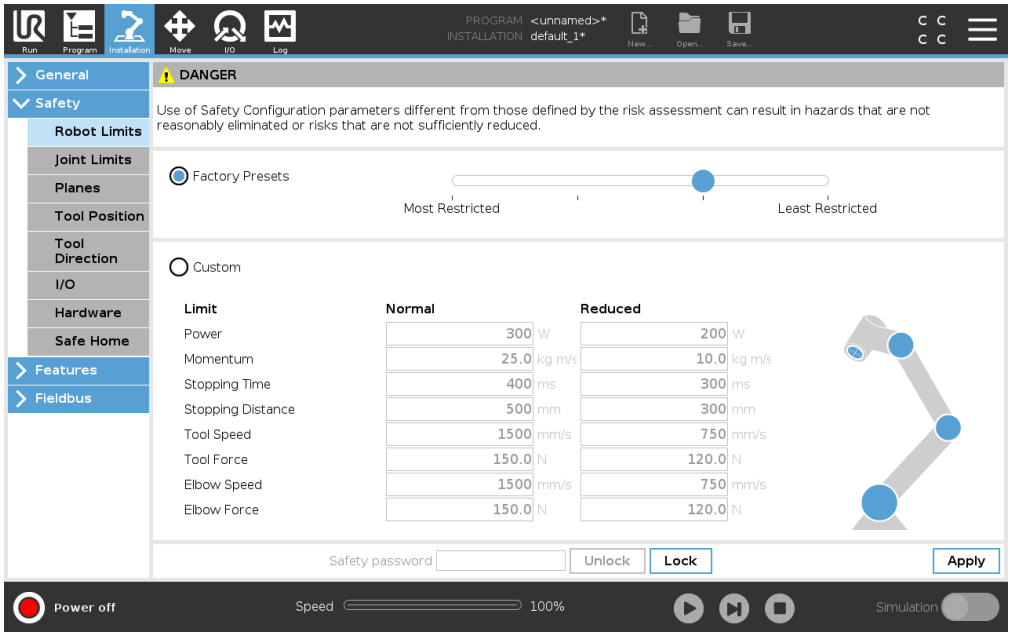
DANGER:
Before you configure your robot safety settings, your integrator must conduct a risk assessment to guarantee the safety of personnel and equipment around the robot. A risk assessment is an evaluation of all work procedures throughout the robot lifetime, conducted in order to apply correct safety configuration settings (see Hardware Installation Manual). You must set the following in accordance with the integrator's risk assessment.

1. The integrator must prevent unauthorized persons from changing the safety configuration e.g. installing password protection.
2. Use and configuration of the safety-related functions and interfaces for a specific robot application (see Hardware Installation Manual).
3. Safety configuration settings for set-up and teaching before the robot arm is powered on for the first time.
4. All safety configuration settings accessible on this screen and sub-tabs.
5. The integrator must ensure that all changes to the safety configuration settings comply with the risk assessment.

13.1.1 Accessing Safety Configuration

Note: Safety Settings are password protected and can only be configured once a password is set and subsequently used.

1. In your PolyScope header, press the **Installation** icon.
2. On the left of the screen, in the action menu, press **Safety**.
3. Observe that the **Robot Limits** screen displays, but settings are inaccessible.
4. If a **Safety password** was previously set, enter the password and press **Unlock** to make settings accessible. Note: Once Safety settings are unlocked, all settings are now active.
5. Press **Lock** tab or navigate away from the Safety menu to lock all Safety item settings again.



You can find more safety system information in the `Hardware Installation Manual`.

13.1.2 Setting a Safety Password

You must set a password to Unlock all safety settings that make up your Safety Configuration.
Note: If no safety password is applied, you are prompted to set it up.

1. In your PolyScope header right corner, press the **Hamburger** menu and select **Settings**.
2. On the left of the screen, in the blue menu, press **Password** and select **Safety**.
3. In **New password**, type a password.
4. Now, in **Confirm new password**, type the same password and press **Apply**.
5. In the bottom left of the blue menu, press Exit to return to previous screen.

Note: You can press the **Lock** tab to lock all Safety settings again or simply navigate to a screen outside of the Safety menu.

13.1.3 Changing the Safety Configuration

Changes to the Safety Configuration settings must comply with the risk assessment conducted by the integrator `(see Hardware Installation Manual)`.

Recommended procedure:

1. Verify that changes comply with the risk assessment conducted by the integrator.
2. Adjust safety settings to the appropriate level defined by the risk assessment conducted by the integrator.

3. Verify that the settings are applied.
4. Place following text in the operators' manuals:

"Before working near the robot, make sure that the safety configuration is as expected. This can be verified e.g. by inspecting the safety checksum in the top right corner of PolyScope for any changes."

13.1.4 Applying New Safety Configuration

The robot is powered off while you make changes to the configuration. Your changes only take effect after you hit the **Apply** button. The robot cannot be powered on again until you either **Apply and Restart** or **Revert Changes**. The former allows you to visually inspect your robot Safety Configuration which, for safety reasons, is displayed in SI Units in a popup. When your visual inspection is complete you can **Confirm Safety Configuration** and the changes are automatically saved as part of the current robot installation.

13.1.5 Safety Checksum

The **Safety Checksum** icon displays your applied robot safety configuration and is read from top to bottom, left to right e.g. BF4B. Different text and/or colors indicate changes to the applied safety configuration.

Note:

- The **Safety Checksum** changes if you change the **Safety Functions** settings, because the **Safety Checksum** is only generated by the safety settings.
- You must apply your changes to the **Safety Configuration** for the **Safety Checksum** to reflect your changes.

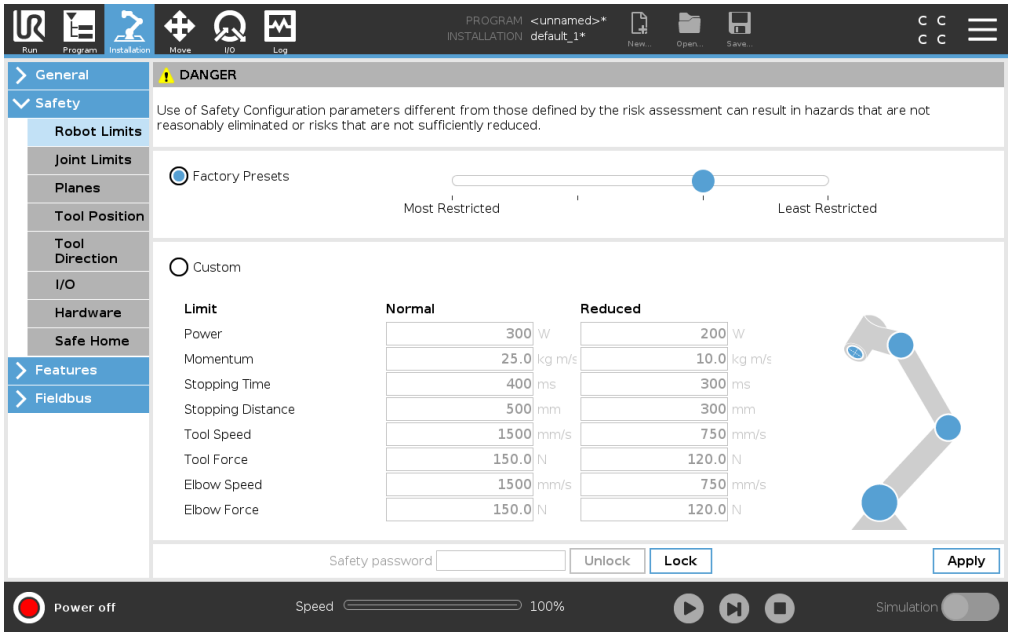
13.2 Safety Menu Settings

This section defines Safety menu settings that make up your robot Safety configuration.

13.2.1 Robot Limits

Robot Limits allow you to restrict general robot movements. The Robot Limits screen has two configuration options: **Factory Presets** and **Custom**.

1. Factory Presets is where you can use the slider to select a predefined safety setting . The values in the table are updated to reflect the preset values ranging from **Most Restricted** to **Least Restricted**
   Note: Slider values are only suggestions and do not substitute a proper risk assessment.



2. Custom is where you can set Limits on how the robot functions and monitor the associated Tolerance.

Power limits maximum mechanical work produced by the robot in the environment.
Note: this limit considers the payload a part of the robot and not of the environment.

Momentum limits maximum robot momentum.

Stopping Time limits maximum time it takes the robot to stop e.g. when an emergency stop is activated.

Stopping Distance limits maximum distance the robot tool or elbow can travel while stopping.

NOTE:
Restricting stopping time and distance affect overall robot speed. For example, if stopping time is set to 300 ms, the maximum robot speed is limited allowing the robot to stop within 300 ms.

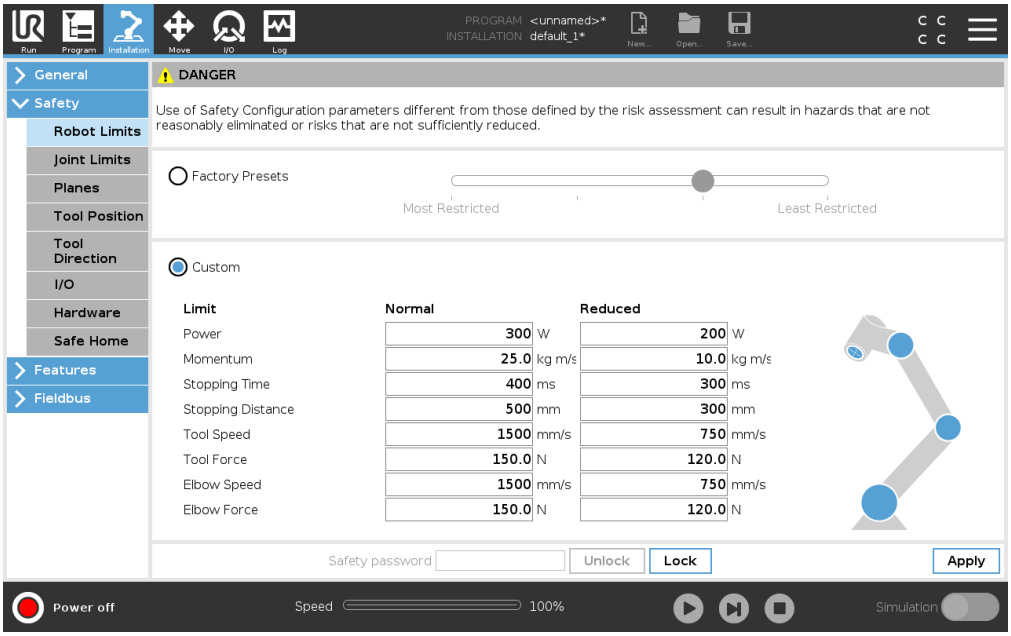
Tool Speed limits maximum robot tool speed.

Tool Force limits the maximum force exerted by the robot tool in clamping situations.

Elbow Speed limits maximum robot elbow speed.

Elbow Force limits maximum force that the elbow exerts on the environment.

The tool speed and force are limited at the tool flange and the center of the two user-defined tool positions, see 13.2.6.



NOTE:
You can switch back to **Factory Presets** for all robot limits to reset to their default settings.

13.2.2 Safety Modes

Under normal conditions, i.e. when no protective stop is in effect, the safety system operates in a Safety Mode associated with a set of safety limits:

Normal mode is the safety mode that is active by default

Reduced mode is active when the robot **Tool Center Point** (TCP) is positioned beyond a Trigger Reduced mode plane (see 13.2.5), or when triggered using a configurable input (see 13.2.8)

Recovery mode activates when a safety limit from the active limit set is violated, the robot arm performs a Stop Category 0. If an active safety limit, such as a joint position limit or a safety boundary, is violated already when the robot arm is powered on, it starts up in **Recovery** mode. This makes it possible to move the robot arm back within the safety limits. While in Recovery mode, the movement of the robot arm is restricted by a fixed limit that you cannot customize. For details about Recovery mode limits (see Hardware Installation Manual).

WARNING:
Limits for **joint position**, **tool position** and **tool orientation** are disabled in Recovery mode, so take caution when moving the robot arm back within the limits.

The menu of the Safety Configuration screen enables the user to define separate sets of safety limits for Normal and Reduced mode. For the tool and joints, Reduced mode limits for speed and momentum are required to be more restrictive than their Normal mode counterparts.

13.2.3 Tolerances

In the Safety Configuration the safety system limits are specified. The *Safety System* receives the values from the input fields and detects any violation if any these values are exceeded. The robot controller attempts to prevent any violations by making a protective stop or by reducing the speed. This means that a program might not be able to perform motions very close to a limit.

WARNING:
Tolerances are specific to Software version. Updating software may change tolerances. Consult the release notes for information about Software version changes.

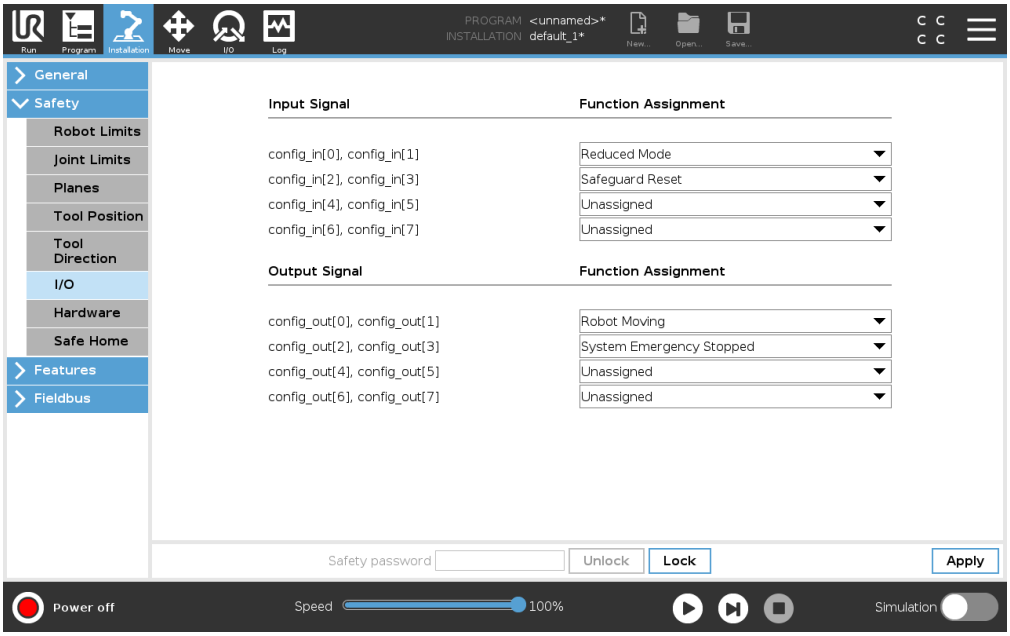
13.2.4 Joint Limits

Joint Limits allow you to restrict individual robot joint movements in joint space i.e. joint rotational position and joint rotational speed. There are two Joint Limits options: **Maximum speed** and **Position range**.

The Wrist 3 position range is unlimited by default. When using cables attached to the robot, you must first disable the **Unrestricted Range for Wrist 3** checkbox to avoid cable tension and protective stops.

1. Maximum speed is where you define the maximum angular velocity for each joint.

2. Position range is where you define the position range for each joint. Again, the input fields for Reduced mode are disabled if there is no safety plane or configurable input set to trigger it. This limit enables safety-rated soft axis limiting of the robot.



13.2.5 Planes

NOTE:
Configuring planes is entirely based on features. We recommend you create and name all features before editing the safety configuration, as the robot is powered off once the Safety Tab has been unlocked and moving the robot will be impossible.

Safety planes restrict robot workspace. You can define up to eight safety planes, restricting the robot tool and elbow. You can also restrict elbow movement for each safety plane and disable by deselecting the checkbox. Before configuring safety planes, you must define a feature in the robot installation (see 16.1.3). The feature can then be copied into the safety plane screen and configured.

WARNING:
Defining safety planes only limits the defined Tool spheres and elbow, not the overall limit for the robot arm. This means that specifying a safety plane, does not guarantee that other parts of the robot arm will obey this restriction.

Modes

You can configure each plane with restrictive **Modes** using the icons listed below.

Disabled The safety plane is never active in this state.

Normal When the safety system is in Normal mode, a normal plane is active and it acts as a strict limit on the position.

Reduced When the safety system is in Reduced mode, a reduced mode plane is active and it acts as a strict limit on the position.

Normal & Reduced When the safety system is either in Normal or Reduced mode, a normal and reduced mode plane is active and acts as a strict limit on the position.

Trigger Reduced Mode The safety plane causes the safety system to switch to Reduced mode if the robot Tool or Elbow is positioned beyond it.

Show Pressing this icon hides or shows the safety plane in the graphics pane.

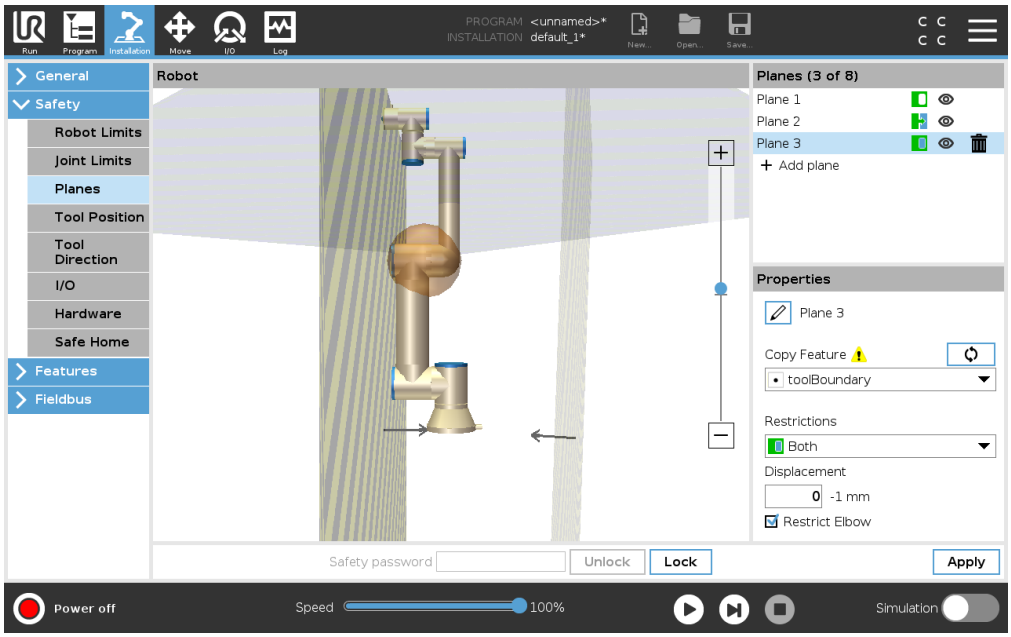
Delete Deletes the created safety plane (note: there is no undo/redo action here so if a plane is deleted by mistake, it will have to be remade)

Rename Pressing this icon allows you to rename the plane.

Configuring Safety Planes

1. In your PolyScope header, tap **Installation**.
2. On the left, in the action menu, tap Safety and select **Planes**.
3. On the top right of the screen, in the Planes field, tap **Add plane**.
4. On the bottom right of the screen, in the **Properties** field, set up Name, Copy Feature and Restrictions. Note: In **Copy Feature**, only Undefined and Base are available. You can reset a configured safety plane by selecting **Undefined**

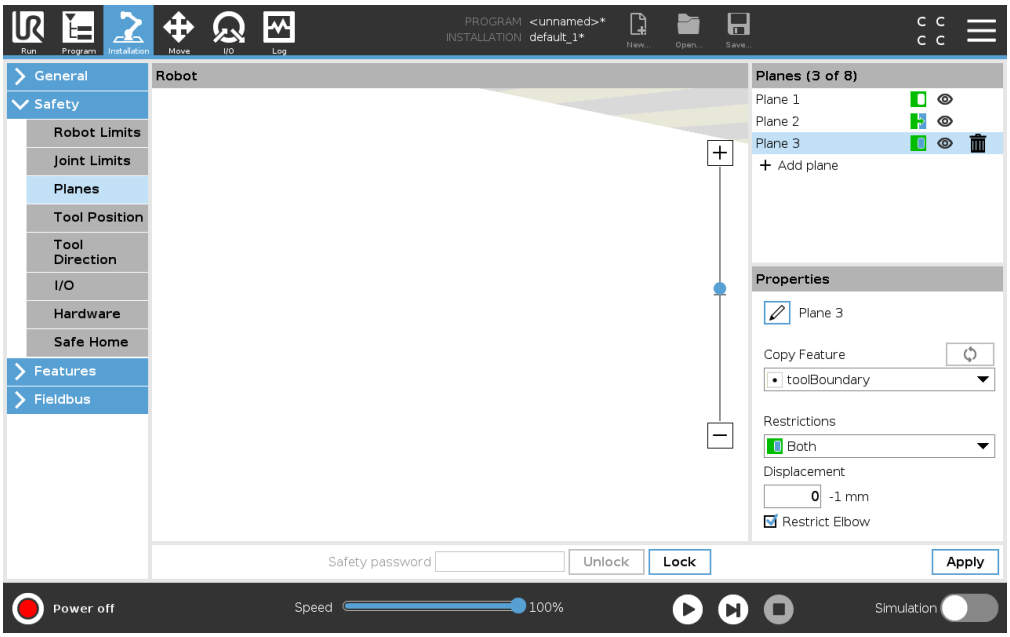
If the copied feature is modified in the Features screen, a warning icon appears to the right of the Copy Feature text. This indicates that the feature is out of sync i.e. the information in the properties card is not updated to reflect the modifications that may have been made to the Feature.



Elbow

You can enable **Restrict Elbow** to prevent robot elbow joint from passing through any of your defined planes. Disable Restrict Elbow for elbow to pass through planes.

Color Codes



Gray Plane is configured but disabled (A)

Yellow & Black Normal Plane (B)

Blue & Green Trigger Plane (C)

Black Arrow The side of the plane the tool and/or elbow is allowed to be on (For Normal Planes)

Green Arrow The side of the plane the tool and/or elbow is allowed to be on (For Trigger Planes)

Gray Arrow The side of the plane the tool and/or elbow is allowed to be on (For Disabled Planes)

Freedriving Robot

If the robot comes close to certain limits, while in Freedrive (see 17.2), you can experience a repelling force from the robot.

13.2.6 Tool Position

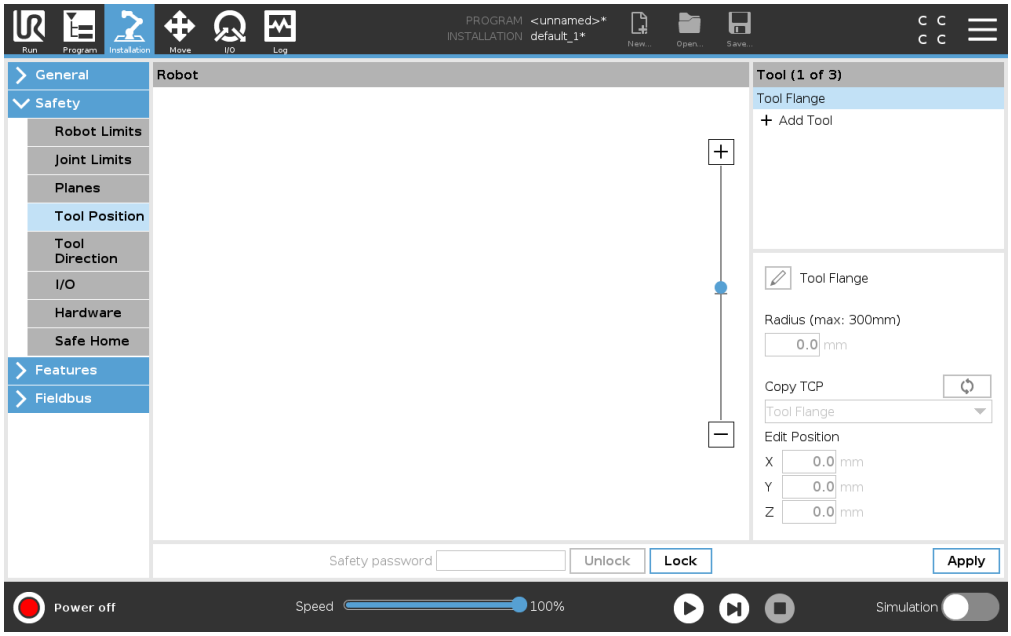
The Tool Position screen enables more controlled restriction of tools and/or accessories placed on the end of the robot arm.

Robot is where you can visualize your modifications.

Tool is where you can define and configure a tool up to two tools.

Tool_1 is the default tool defined with values x=0.0, y= 0.0, z=0.0 and radius=0.0. These values represent the robot tool flange.
Note:

- Under Copy TCP, you can also select **Tool Flange** and cause the tool values to go back to 0.
- A default sphere is defined at the tool flange.



For the user defined tools, the user can change:

Radius to change the radius of the tool sphere. The radius is considered when using safety planes. When a point in the sphere passes a reduced mode trigger plane, the robot switches to *Reduced* mode. The safety system prevents any point on the sphere from passing a safety plane (see 13.2.5).

Position to change the position of the tool with respect to the tool flange of the robot. The position is considered for the safety functions for tool speed, tool force, stopping distance and safety planes.

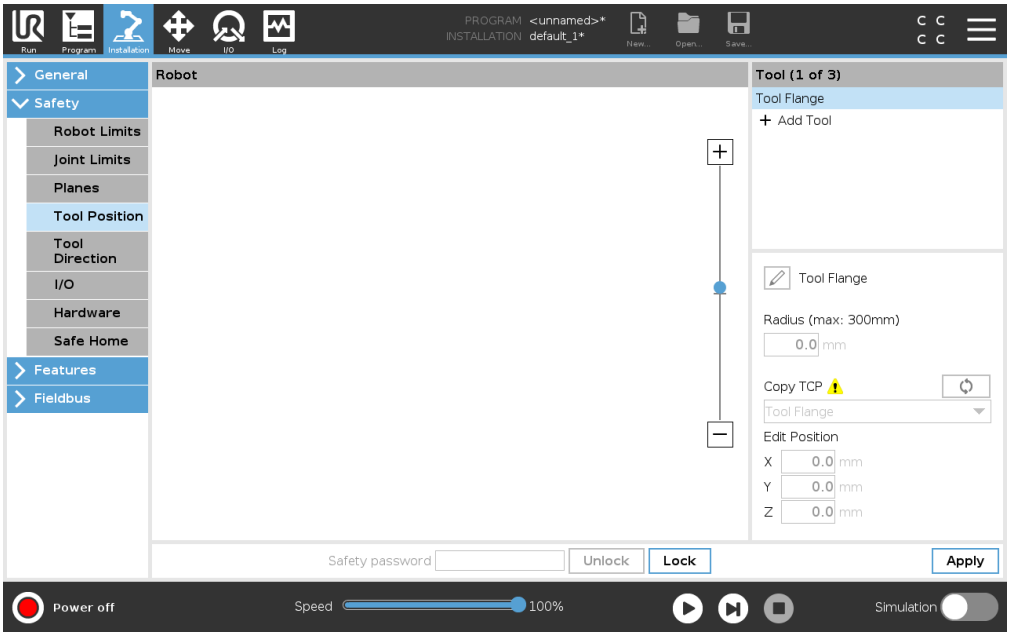
You can use an existing Tool Center Point as a base for defining new tool positions. A copy of the existing TCP, predefined in General menu, in TCP screen, can be accessed in Tool Position menu, in Copy TCP drop-down list.

When you edit or adjust the values in the **Edit Position** input fields, the name of the TCP visible in the drop down menu changes to **custom**, indicating that there is a difference between the copied TCP and the actual limit input. The original TCP is still available in the drop down list and can be selected again to change the values back to the original position. The selection in the copy TCP drop down menu does not affect the tool name.

Once you apply your Tool Position screen changes, if you try to modify the copied TCP in the TCP configuration screen, a warning icon appears to the right of the Copy TCP text. This indicates that the TCP is out of sync i.e. the information in the properties field is not updated to reflect modifications that may have been made to the TCP. The TCP can be synced by pressing the sync icon (see 16.1.1).

Note: the TCP does not have to be synced in order to define and use a tool successfully.

You can rename the tool by pressing the pencil tab next to the displayed tool name. You can also determine the Radius with an allowed range of 0-300 mm. The limit appears in the graphics pane as either a point or a sphere depending on radius size.

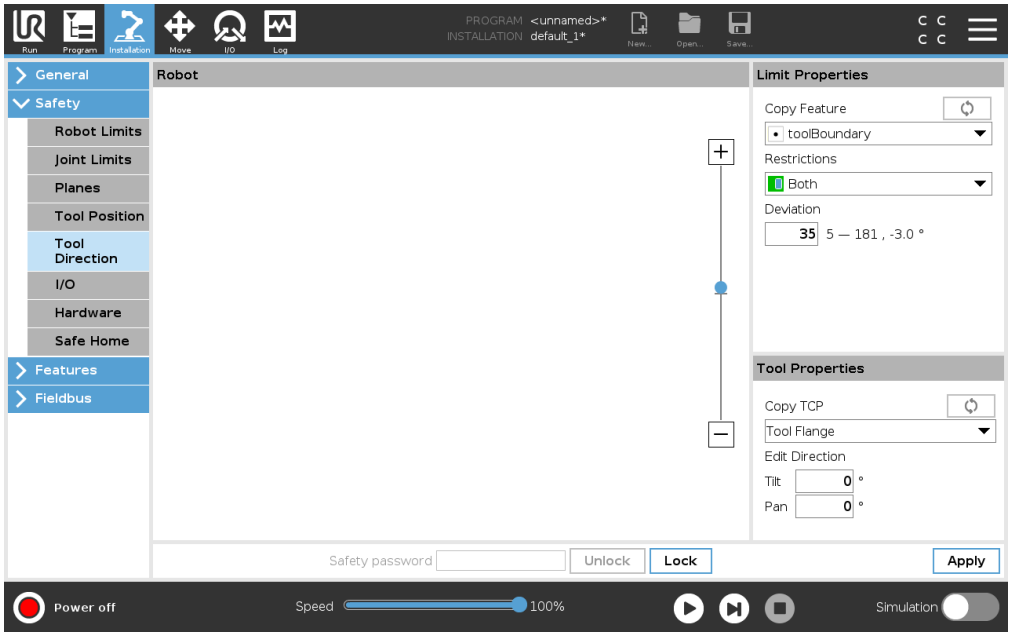


13.2.7 Tool Direction

The Tool Direction screen can be used to restrict the angle in which the tool is pointing. The limit is defined by a cone that has a fixed orientation with respect to the robot arm Base. As the robot arm moves around, tool direction is restricted so it remains within the defined cone. The default direction of the tool coincides with the Z-axis of the tool output flange. It can be customized by specifying tilt and pan angles.

Before configuring the limit, you must define a point or plane in the robot installation (see 16.3). The feature can then be copied and its Z axis used as the center of the cone defining the limit.

NOTE:
Configuration of the tool direction is based on features. We recommend you create desired feature(s) before editing the safety configuration, as once the Safety Tab has been unlocked, the robot arm powers off making it impossible to define new features.



Limit Properties

The Tool Direction limit has three configurable properties:

1. **Cone center**: You can select a point or plane feature from the drop-down menu, to define the center of the cone. The Z axis of the selected feature is used as the direction around which the cone is centred.
2. **Cone angle**: You can define how many degrees the robot is allowed to deviate from center.

Disabled Tool direction limit is never active

Normal Tool direction limit is active only when safety system is in **Normal mode**.

Reduced Tool direction limit is active only when the safety system is in **Reduced mode**.

Normal & Reduced Tool direction limit is active when the safety system is in **Normal mode** as well as when it is in **Reduced mode**.

You can reset the values to default or undo the Tool Direction configuration by setting the copy feature back to ”Undefined”.

Tool Properties

By default, the tool points in the same direction as the Z axis of the tool output flange. This can be modified by specifying two angles:

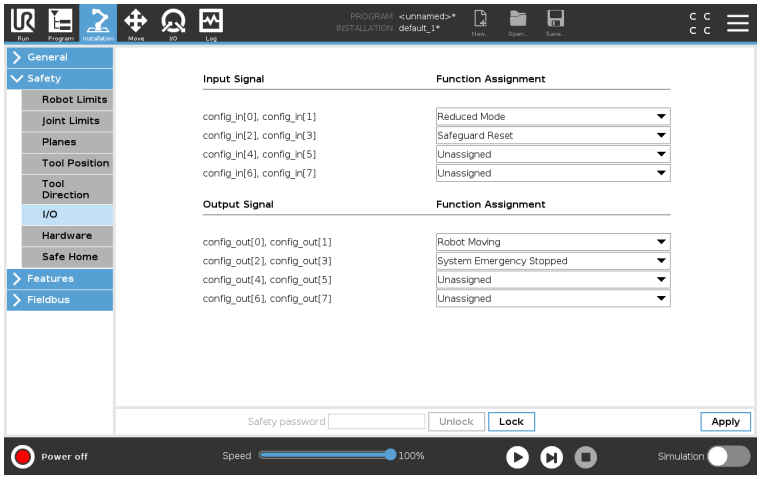
Tilt angle: How much to tilt the Z axis of the output flange towards the X axis of the output flange

Pan angle: How much to rotate the tilted Z axis around the original output flange Z axis.

Alternatively, the Z axis of an existing TCP can be copied by selecting that TCP from the drop-down menu.

13.2.8 I/O

The I/O are divided between inputs and outputs and are paired up so that each function provides a Category 3 and PLd I/O.



Input Signals

The following Safety Functions can be used with the input signals:

System Emergency Stop This is an emergency stop button alternative to the one on the Teach Pendant, providing the same functionality if the device complies with ISO 13850.

Reduced Mode All safety limits can be applied in either Normal mode or Reduced mode (see 13.2.2). When configured, a low signal sent to the inputs causing the safety system to transition to Reduced mode. The robot arm decelerates to satisfy the Reduced mode limit set. The safety system guarantees that the robot is within Reduced mode limits less than 0.5s after the input is triggered. If the robot arm continues to violate any of the Reduced mode limits, it performs a Stop Category 0. Transition back to Normal mode occurs in the same way. Note: safety planes can also cause a transition to Reduced mode.

3-Position Enabling Device Defining a **3-Position Enabling Device** safety input makes it possible to define an **Operational Mode** safety input. When defined, the **3-Position Enabling Device** must be held down for a robot in **Manual Mode** to move.

Operational Mode When defined, this input can be used to switch between **Automatic Mode** or **Manual Mode** (see 12.1).

Safeguard Reset When a Safeguard Stop is configured, this output ensures that the Safeguard Stop state continues until a reset is triggered. The robot arm will not move when in a Safeguard Stopped state.

WARNING:
By default, the Safeguard Reset input function is configured for input pins 0 and 1. Disabling it altogether implies that the robot arm ceases to be Safeguard Stopped as soon as the Safeguard Stop input becomes high. In other words, without a Safeguard Reset input, the Safeguard Stop inputs SI0 and SI1 `(see the Hardware Installation Manual)` fully determine whether the Safeguard Stopped state is active or not.

Output Signals

You can apply the following Safety functions for output signals. All signals return to low when the state which triggered the high signal has ended:

System Emergency Stop Signal is *Low* when the safety system has been triggered into an Emergency Stopped state by the Robot Emergency Stop input or the Emergency Stop Button. To avoid deadlocks, if the Emergency Stopped state is triggered by the System Emergency Stop input, low signal will not be given.

Robot Moving Signal is *Low* if the robot is moving, otherwise high.

Robot Not Stopping Signal is *High* when the robot is stopped or in the process of stopping due to an emergency stop or safeguard stop. Otherwise it will be logic low.

Reduced Mode Signal is *Low* when the robot arm is placed in Reduced mode or if the safety input is configured with a Reduced Mode input and the signal is currently low. Otherwise the signal is high.

Not Reduced Mode This is the inverse of the Reduced Mode defined above.

Safe Home Signal is *High* if the Robot Arm is stopped in the configured Safe Home Position. Otherwise, the signal is *Low*.

NOTE:
Any external machinery receiving its Emergency Stop state from the robot through the System Emergency Stop output must comply with ISO 13850. This is particularly necessary in setups where the Robot Emergency Stop input is connected to an external Emergency Stop device. In such cases, the System Emergency Stop output becomes high when the external Emergency Stop device is released. This implies that the emergency stop state at the external machinery will be reset with no manual action needed from the robot's operator. Hence, to comply with safety standards, the external machinery must require manual action in order to resume.

13.2.9 Hardware

You can use the robot without attaching the Teach Pendant. Removing the Teach Pendant requires defining another Emergency Stop source. You must specify if the Teach Pendant is attached to avoid triggering a safety violation.

Selecting Available Hardware

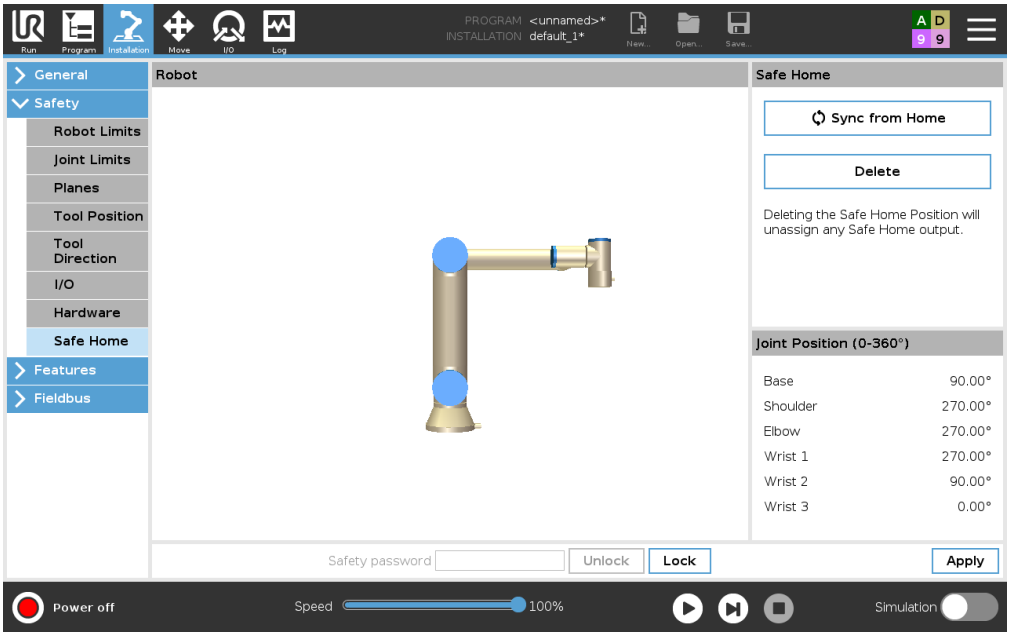
The robot can be used without PolyScope as the programming interface.

1. In the Header tap **Installation**.
2. In the action menu on left tap **Safety** and select **Hardware**.
3. Input Safety password and **Unlock** the screen.
4. Deselect **Teach Pendant** to use robot without PolyScope interface.
5. Press **Save and restart** to implement changes.

CAUTION:
If the Teach Pendant is detached or disconnected from the robot, the Emergency Stop button is no longer active. You must remove the Teach Pendant from the vicinity of the robot.

13.2.10 Safe Home Position

Safe Home is a return position defined by using the user-defined Home Position. Safe Home I/Os are active when the Robot Arm is in the Safe Home Position and a Safe Home I/O is defined. The Robot Arm is in the Safe Home Position if the joint positions are at the specified joint angles or a multiple of 360 degrees thereof.



Syncing from Home

1. In the Header, tap **Installation**.
2. In the action menu on the left, tap **Safety** and select **Safe Home**.
3. Under **Safe Home**, tap **Sync from Home**.
4. Tap **Apply** and in the dialog box that appears, select **Apply and restart**.

Safe Home Output

The Safe Home Position must be defined before the Safe Home Output (see 13.2.8).

Defining Safe Home Output

1. In the Header, tap **Installation**.
2. In the action menu on the left, under **Safety**, select **I/O**.
3. On the I/O screen in the Output Signal, under Function Assignment, in drop-down menu, select **Safe Home**.
4. Tap **Apply** and in the dialog box that appears, select **Apply and restart**.

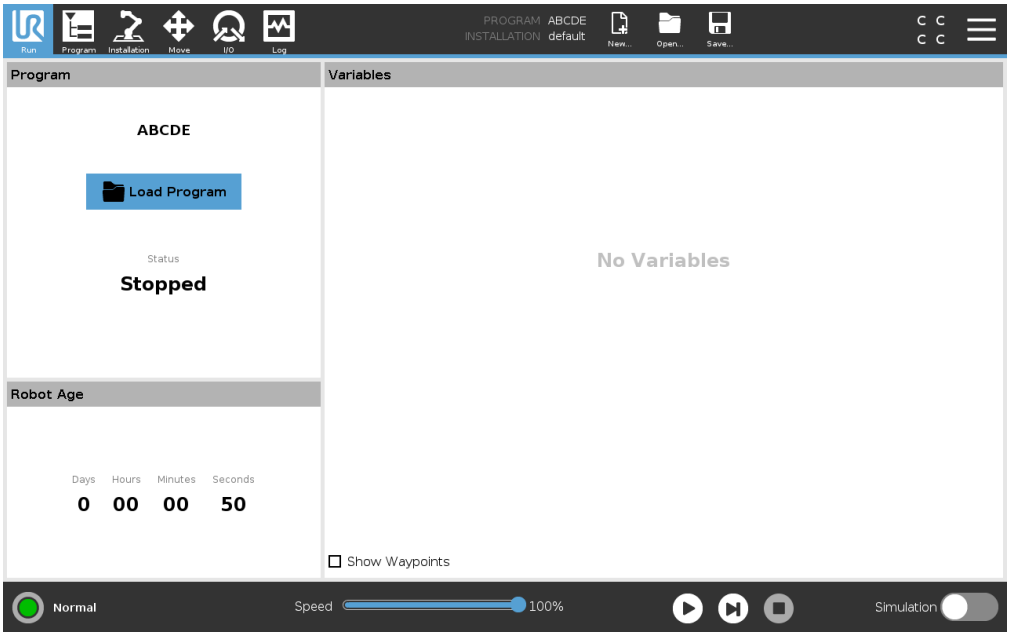
Edit Safe Home

Editing Home does not automatically modify a previously defined Safe Home position. While these values are out of sync, Home program node is undefined.

Editing Safe Home

1. In the Header, tap **Installation**.
2. In the action menu on the left, under **General**, select **Home**.
3. Tap **Edit Position** and set the new Robot Arm position and tap **OK**.
4. In the menu on the left, under **Safety**, select **Safe Home**. Note: a Safety password is required to **Unlock** the Safety Settings (See 13.1.2).
5. Under **Safe Home**, tap **Sync from Home**

14 Run Tab



The **Run** tab allows you to simply operate the robot arm and control box, using as few buttons and options as possible. You can combine simple operation with password protecting the programming part of PolyScope (see 21.3.2), to make the robot into a tool that can run exclusively pre-written programs.

On this screen you can automatically load and start default a program based on an external input signal edge transition (see 16.1.5).
Note: The combination of auto loading and starting of a default program and auto initialization on power up can, for instance, be used to integrate the robot arm into other machinery.

14.1 Program

The **Program** field, displays the name of the program that was loaded on to the robot and its current status. You can tap the **Load Program** tab to load a different program.

14.2 Variables

A robot program can make use of variables to store and update various values during runtime. Two kinds of variables are available:

Installation variables These can be used by multiple programs and their names and values are persisted together with the robot installation (see 16.1.4). Installation variables keep their

value after the robot and control box has been rebooted.

Regular program variables These are available to the running program only and their values are lost as soon as the program is stopped.

Show waypoints The robot program uses script variables to store information about waypoints. Select the **Show Waypoints** checkbox, under **Variables** to show script variables in the variables list.

Variable types

bool	A boolean variable whose value is either `True` or `False`.
int	A whole number in the range from -2147483648 to 2147483647 (32 bit).
float	A floating point number (decimal) (32 bit).
string	A sequence of characters.
pose	A vector describing the location and orientation in Cartesian space. It is a combination of a position vector (x, y, z) and a rotation vector (rx, ry, rz) representing the orientation, written `p[x, y, z, rx, ry, rz]`.
list	A sequence of variables.

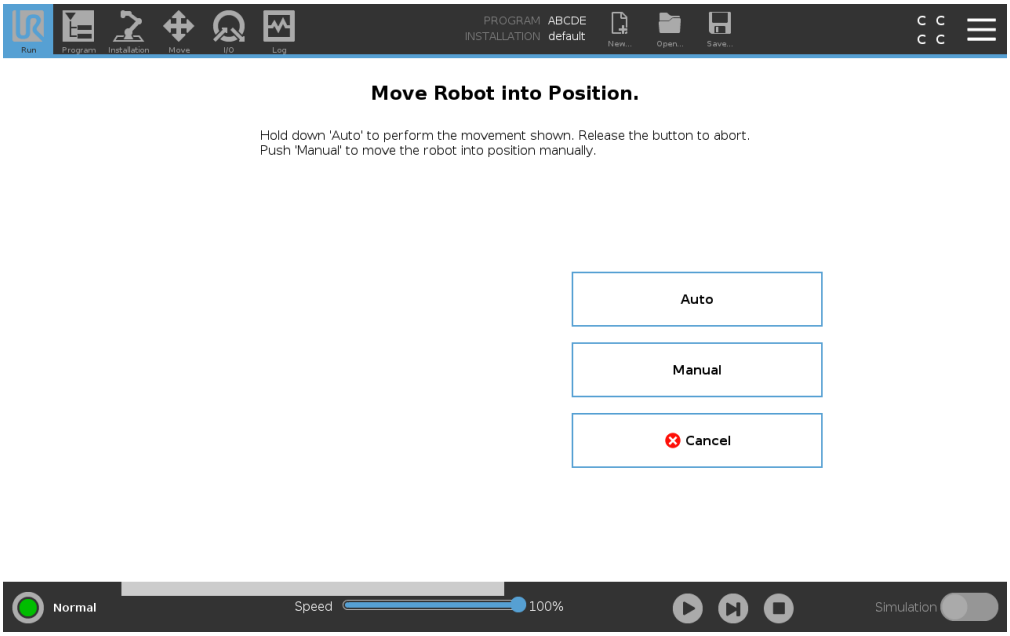
14.3 Robot Age

This field represents the length of time since the robot was initially turned on. The numbers in the field are not associated with program run time

14.4 Move Robot into Position

The **Move Robot into Position** screen appears when you tap **Play** in the **Footer**. Access the **Move Robot into Position** screen when the Robot Arm must move to a particular start position before running a program, or when the Robot Arm is moving to a waypoint while modifying a program.
In cases where the **Move Robot into Position** screen cannnot move the Robot Arm to the program start position, it moves to the first waypoint in the program tree. The Robot Arm can move to an incorrect pose if :

- The TCP, feature pose or waypoint pose of the first movement is altered during program execution before the first move is executed.
- The first waypoint is inside an If or Switch program tree node.



Auto

Hold down the **Auto** tab to move the robot arm to its start position.
Note: You can release the button to stop the motion at any time.

Animation

The animation shows the movement the robot arm is about to perform when you hold down the **Auto** tab.

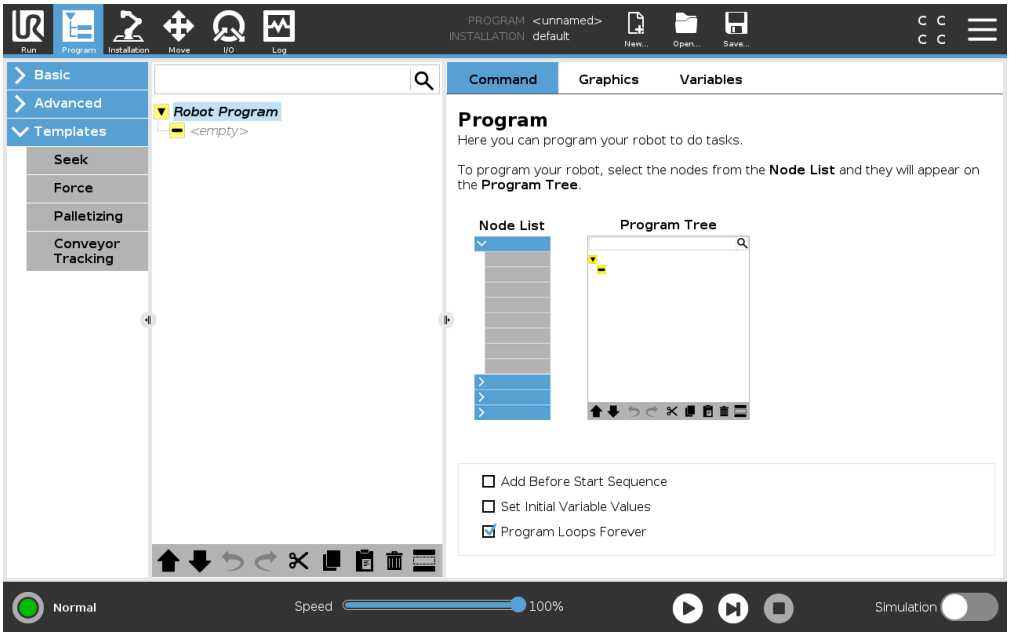
CAUTION:

1. Compare the animation with the position of the real robot arm and make sure that the robot arm can safely perform the movement without hitting any obstacles.
2. The Automove function moves the robot along the shadow trajectory. Collision might damage the robot or other equipment.

Manual

Push the **Manual** tab to access the **Move** icon screen where the robot arm can be moved manually. This is only needed if you do not prefer the animation's movement.

15 Program Tab



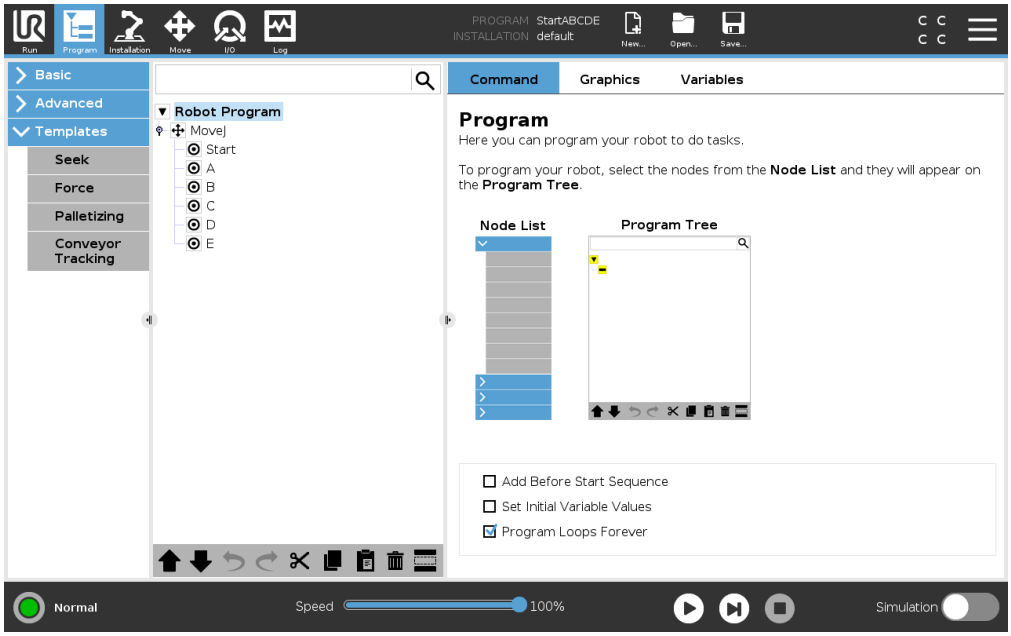
The program tab shows the current program being edited.

15.1 Program Tree

By tapping **Command** you add Program Nodes to the Program Tree. Configure the functionality of the added Program Nodes on the right side of the screen.

An empty Program Tree is not allowed to run. Programs containing mis-configured Program Nodes are also not allowed to run. Invalid Program Nodes are highlighted in yellow to indicate what should be fixed before the program is allowed to run.

15.1.1 Program Execution Indication



When the program is running, the Programe NOde currently being executed is indicated by a small icon next to the node. Furthermore, the path of execution is highlighted using a blue color. Pressing the icon at the corner of the program will make it track the command bring executed.

15.1.2 Search Button

Tap the to search in the Program Tree. Press the × icon to exit search.

15.1.3 Program Tree Toolbar

Use the toolbar at the base of the Program Tree to modify the Program Tree.

Undo/Redo Buttons

The and buttons serve to undo and redo changes to commands.

Move Up & Down

The and buttons change the position of a node.

Cut

The button cuts a node and allows it to be used for other actions (e.g., paste it on other place on the Program Tree).

Copy

button allows copies a node and allows it to be used for other actions (e.g., paste it on other place on the Program Tree).

Paste

The button allows you to paste a node that was previously cut or copied.

Delete

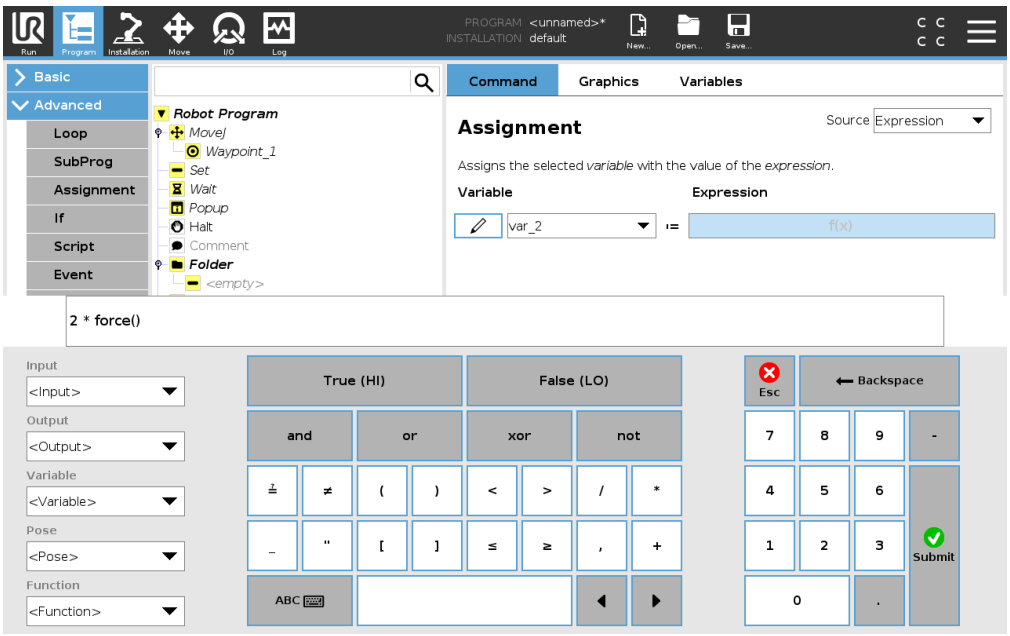
Tap the button to remove a node from the Program Tree.

Suppress

Tap the button to suppress specific nodes on the Program Tree.

Suppressed program lines are simply skipped when the program is run. A suppressed line can be unsuppressed again at a later time. This is a quick way to make changes to a program without destroying the original contents.

15.1.4 Expression Editor



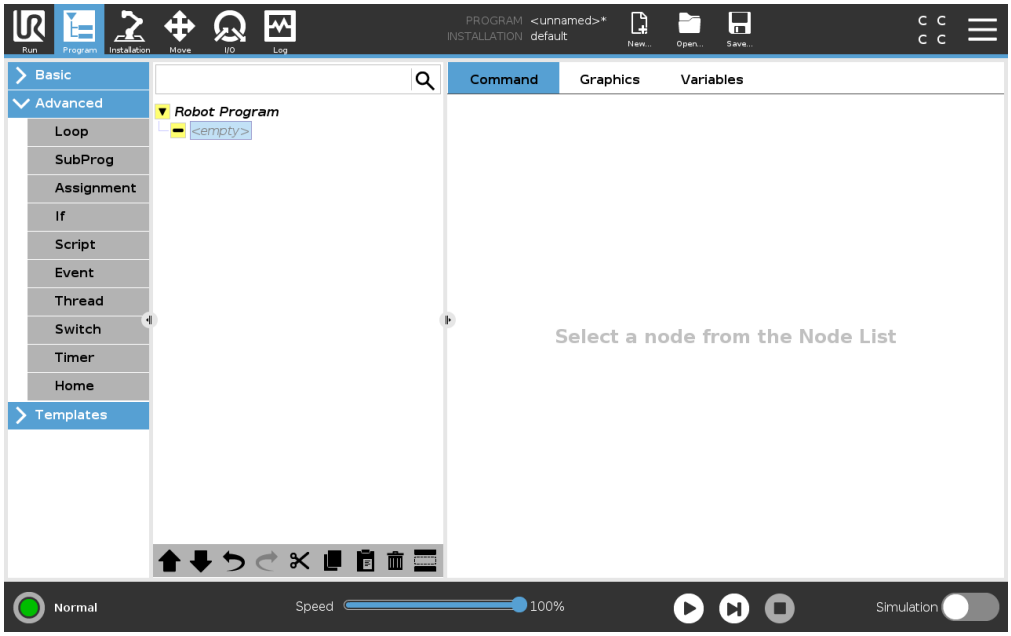
While the expression itself is edited as text, the expression editor has a number of buttons and functions for inserting the special expression symbols, such as $*$ for multiplication and $\leq$ for less than or equal to. The keyboard symbol button in the top left of the screen switches to text-editing of the expression. All defined variables can be found in the `Variable` selector, while the names of the input and output ports can be found in the `Input` and `Output` selectors. Some special functions are found in `Function`.

The expression is checked for grammatical errors when the `Ok` button is pressed. The `Cancel` button leaves the screen, discarding all changes.

An expression can look like this:

`digital_in[1]`$\stackrel{?}{=}$`True and analog_in[0]<0.5`

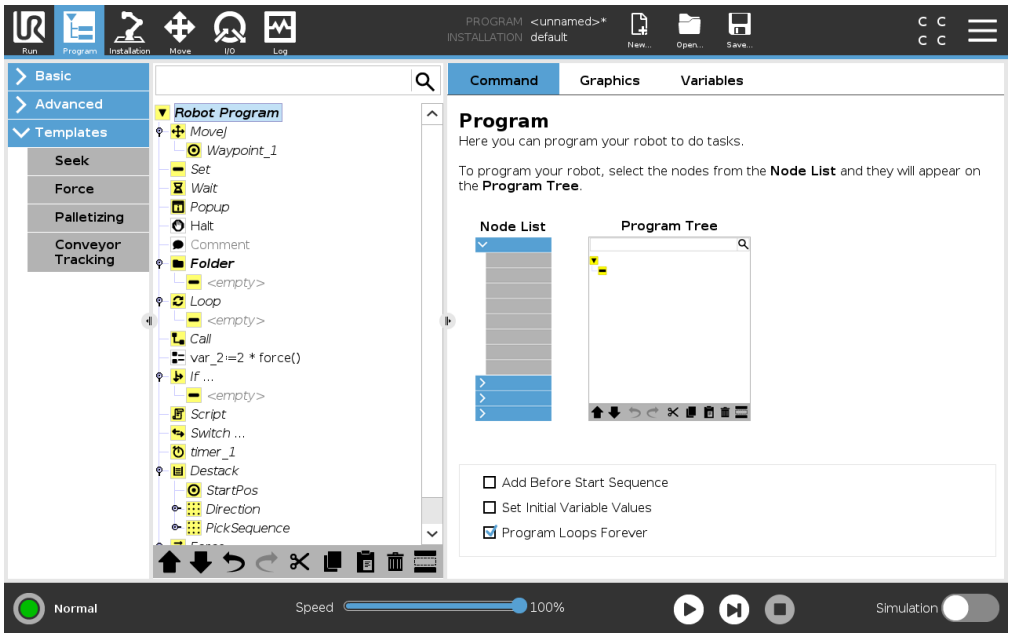
15.1.5 Empty Node



Program Nodes cannot be *empty*. All lines must be specified and defined in the Program Tree for a program to run.

15.2 Command Tab

This manual does not cover all the details about every type of Program Node. The Robot Program Node includes three check-boxes controlling the overall behavior of the program.



Add Before Start Sequence

Select this check-box to add a special section to the program which is once when the program starts.

Set Initial Variables Values

Select this to set initial values of program variables.

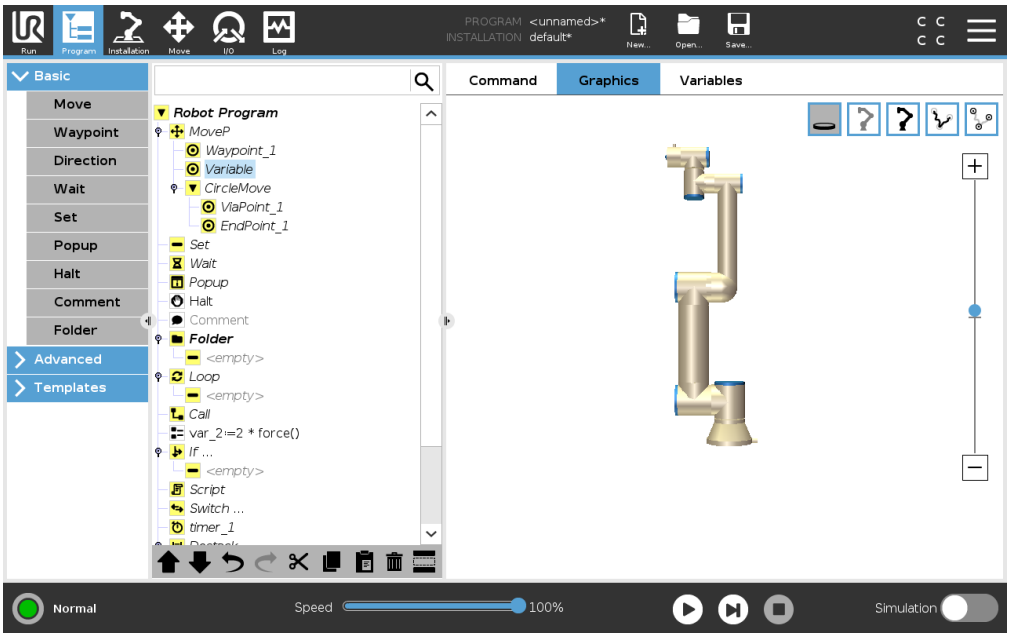
1. Select a variable from the dropdown list, or by use the variable selector box.
2. Enter an expression for that variable. This expression is used to set the variable value at program start.
3. You can select **Keep value from previous run** to initialize the variable to the value found on the **Variables** tab (see 15.4).
   This allows variables to maintain their values between program executions. The variable gets its value from the expression if the program is run for the first time, or if the value tab has been cleared.

A variable can be deleted from the program by setting its name to blank (only spaces).

Program Loops Forever

Select this to make the program continuous.

15.3 Graphics Tab



Graphical representation of the current robot program. The path of the TCP is shown in 3D view, with motion segments in black, and blend segments (transitions between motion segments) shown in green. The green dots specify the positions of the TCP at each of the waypoints in the program. The 3D drawing of the robot arm shows the current position of the robot arm, and the *shadow* of the

robot arm shows how the robot arm intends to reach the waypoint selected in the left hand side of the screen.

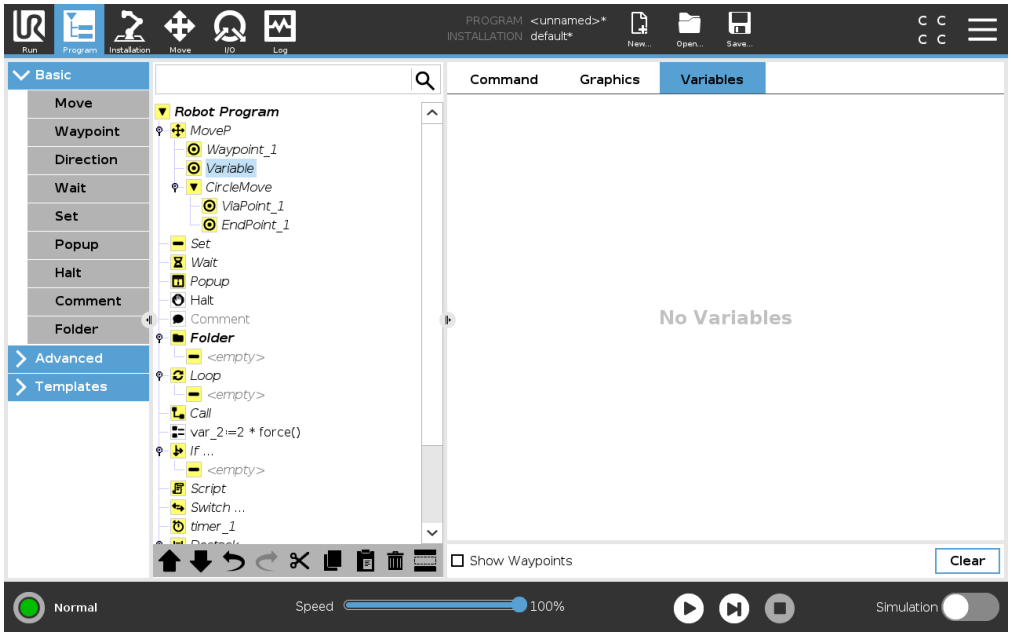
If the current position of the robot TCP comes close to a safety or trigger plane, or the orientation of robot tool is near the tool orientation boundary limit (see 13.2.5), a 3D representation of the boundary limit is shown.
Note: when the robot is running a program, the visualization of boundary limits will be disabled.

Safety planes are visualized in yellow and black with a small arrow representing the plane normal, which indicates the side of the plane on which the robot TCP is allowed to be positioned. Trigger planes are displayed in blue and green and a small arrow pointing to the side of the plane, where the **Normal** mode limits (see 13.2.2) are active. The tool orientation boundary limit is visualized with a spherical cone together with a vector indicating the current orientation of the robot tool. The inside of the cone represents the allowed area for the tool orientation (vector).

When the target robot TCP no longer is in the proximity of the limit, the 3D representation disappears. If the TCP is in violation or very close to violating a boundary limit, the visualization of the limit turns red.

The 3D view can be zoomed and rotated to get a better view of the robot arm. The buttons in the top-right side of the screen can disable the various graphical components in 3D view. The bottom button switches on/off the visualization of proximate boundary limits.

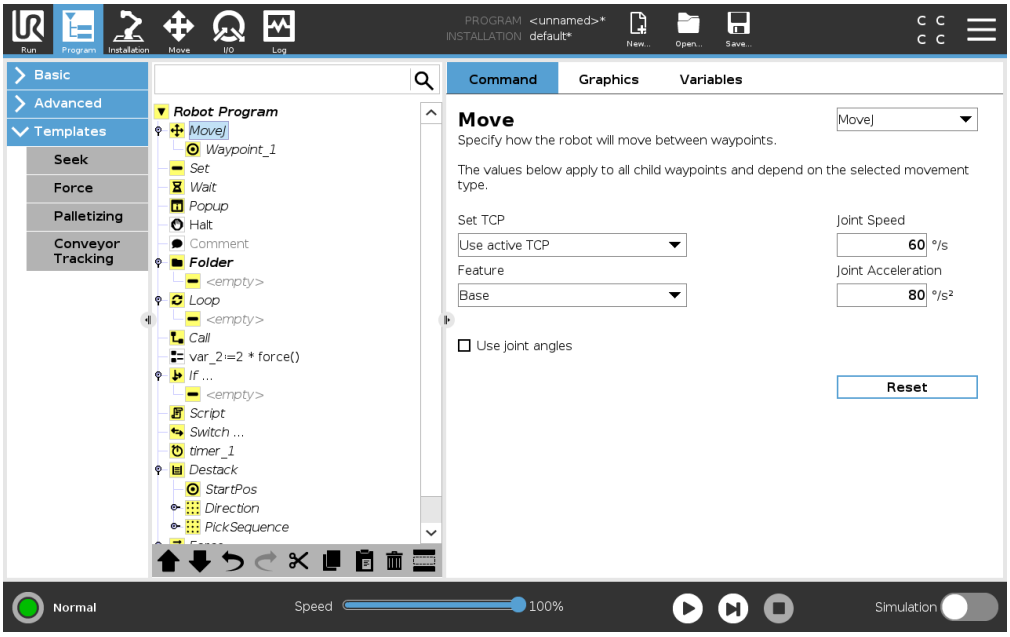
15.4 Variables Tab



The **Variables** tab shows the live values of variables in the running program, and keeps a list of variables and values between program runs. It only appears when it has information to display. Waypoint variables will be shown in the list if Show Waypoints is enabled.

15.5 Basic program nodes

15.5.1 Move



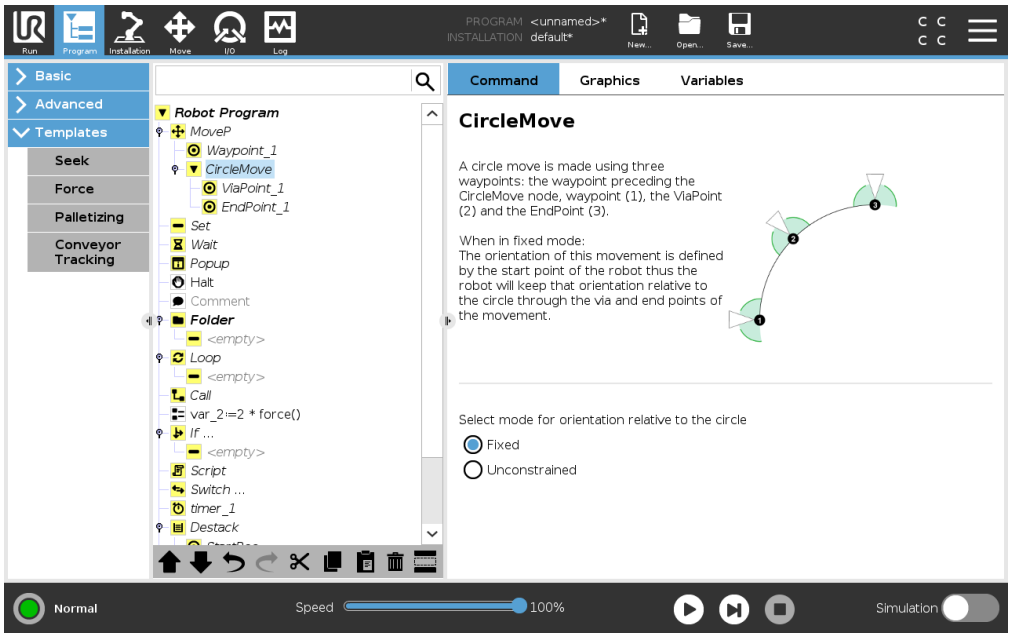
The **Move** command controls the robot motion through the underlying waypoints. Waypoints have to be under a Move command. The Move command defines the acceleration and the speed at which the robot arm will move between those waypoints.

Movement Types

You can select one of three types of movements: **MoveJ**, **MoveL** and **MoveP**. Each movement type is explained below.

- **moveJ** makes movements that are calculated in the robot arm **joint space**. Joints are controlled to finish their movements at the same time. This movement type results in a curved path for the tool. The shared parameters that apply to this movement type are the maximum joint speed and joint acceleration, specified in deg/s and deg/s^2, respectively. If it is desired to have the robot arm move fast between waypoints, disregarding the path of the tool between those waypoints, this movement type is the preferable choice.
- **moveL** moves the Tool Center Point (TCP) linearly between waypoints. This means that each joint performs a more complicated motion to keep the tool on a straight line path. The shared parameters that can be set for this movement type are the desired tool speed and tool acceleration specified in mm/s and mm/s^2, respectively, and also a feature.
- **moveP** moves the tool linearly with constant speed with circular blends, and is intended for some process operations, like gluing or dispensing. The size of the blend radius is by default a shared value between all the waypoints. A smaller value will make the path turn sharper whereas a higher value will make the path smoother. While the robot arm is moving through the waypoints with constant speed, the robot control box cannot wait for either an I/O operation or an operator action. Doing so might stop the robot arm's motion, or cause a protective stop.

- **Circle move** can be added to a **moveP** to make a circular movement. The robot starts the movement from its current position or start point, moves through a **ViaPoint** specified on the circular arc, and an **EndPoint** that completes the circular movement.
  A mode is used to calculate tool orientation, through the circular arc. The mode can be:
  - Fixed: only the start point is used to define tool orientation
  - Unconstrained: the start point transforms to the **EndPoint** to define tool orientation



Shared parameters

The shared parameters in the bottom right corner of the Move screen apply to the movement from the previous position of the robot arm to the first waypoint under the command, and from there to each of the following waypoints. The Move command settings do not apply to the path going *from* the last waypoint under that Move command.

TCP selection

The way the robot moves between waypoints is adjusted depending on whether the TCP is set using a user defined TCP or an active TCP. **Ignore Active TCP** allows this movement to be adjusted in relation to the Tool Flange.

Setting the TCP in a Move

1. Access the Program Tab screen to set the TCP used for waypoints.
2. Under Command, in the drop down menu on the right select the Move type.
3. Under Move, select an option in the **Set TCP** drop down menu.
4. Select **Use active TCP** or select **a user defined TCP**.
   You can also choose **Ignore Active TCP**.

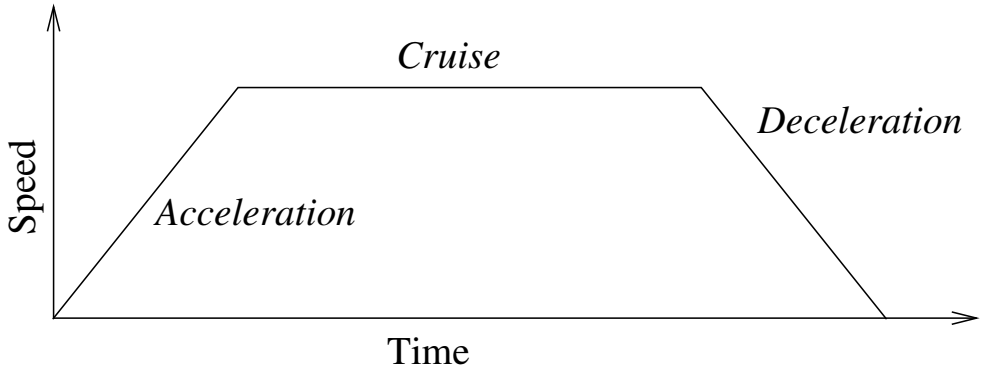


Figure 15.1: Speed profile for a motion. The curve is divided into three segments: *acceleration*, *cruise* and *deceleration*. The level of the *cruise* phase is given by the speed setting of the motion, while the steepness of the *acceleration* and *deceleration* phases is given by the acceleration parameter.

Feature selection

The feature spaces the waypoints under the Move command, that should be represented when specifying these waypoints (see section 16.3). This means that when setting a waypoint, the program will remember the tool coordinates in the feature space of the selected feature. There are a few circumstances that need detailed explanation:

Relative waypoints The selected feature has no effect on relative waypoints. The relative movement is always performed with respect to orientation of the **Base**.

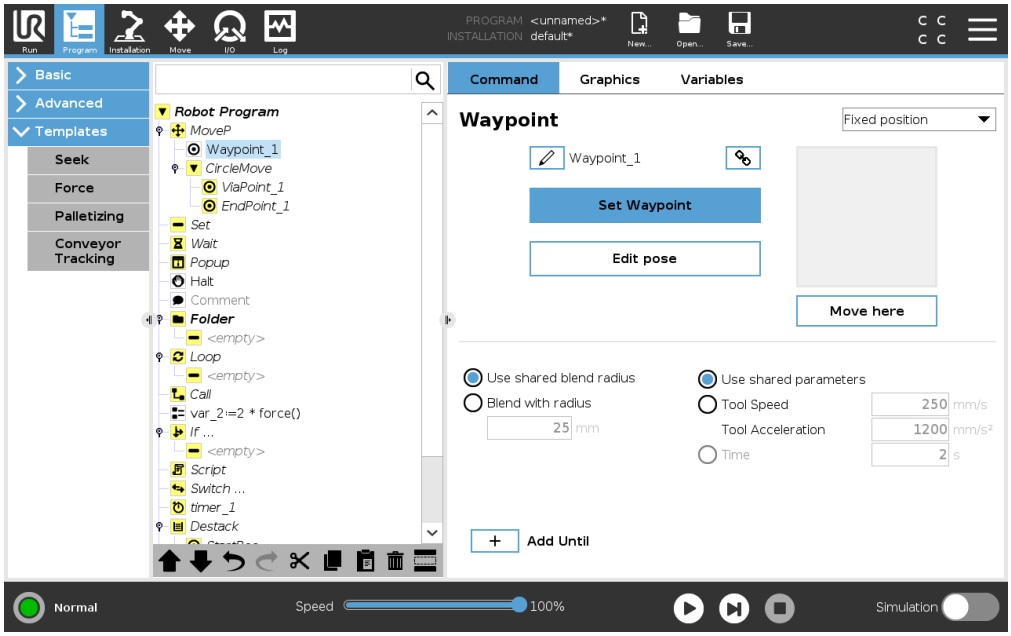
Variable waypoints When the robot arm moves to a variable waypoint, the Tool Center Point (TCP) is calculated as the coordinates of the variable in the space of the selected feature. Therefore, the robot arm movement for a variable waypoint changes if another feature is selected.

Feature variable You can change a feature's position while the program is running by assigning a pose to its corresponding variable.

Use joint angles

As an alternative to the 3D pose, you can select the **Use joint angles** checkbox when using the MoveJ to define waypoints using the robot joint angles. If **Use joint angles** is enabled, TCP and feature options are unavailable. Waypoints defined using **Use joint angles** are not adjusted when the program is moved between robots.

Fixed Waypoint



A point on the robot path. Waypoints are the most central part of a robot program, telling the robot arm where to be. A fixed position waypoint is taught by physically moving the robot arm to the position.

Teaching Waypoints

Teaching is the term used to show the robot how to position the TCP in relation to a feature for an application. To teach the robot a waypoint, follow the instructions below:

1. In the Program Tab, insert a **Move Node**.
2. On the Move Node, use the **Set TCP** drop-down menu to set the TCP.
3. On the Move Node, use the **Feature** drop-down menu to select a feature.
4. On the Waypoint Node, use **Teach Mode** or **Jog** to position the robot in a desired configuration.

Using Waypoints

Using a waypoint means applying the taught relation between the feature and the TCP in the present situation. The relation between the feature and the TCP, applied to the current selected feature, achieves the desired TCP location. Then the robot figures out how to position itself to let the currently active TCP reach that TCP position. To use a waypoint, follow the instructions below:

1. Use an existing waypoint in a Move Node, or insert the waypoint into a different Move Node (e.g. by copy and paste or use the ”Link” button on the waypoint).
2. Set the desired TCP.
3. Set the desired feature.

Setting the waypoint

Waypoint names

Waypoints automatically get a unique name. The name can be changed by the user. By selecting the link icon, waypoints are linked and share position information. Other waypoint information such as blend radius, tool/joint speed and tool/joint acceleration is configured for individual waypoints even though they may be linked.

Blending

Blending enables the robot to smoothly transition between two trajectories, without stopping at the waypoint between them.

Example Consider a pick and place application as an example (see figure 15.2), where the robot is currently at Waypoint 1 (WP_1), and it needs to pick up an object at Waypoint 3 (WP_3). To avoid collisions with the object and other obstacles (O), the robot must approach WP_3 in the direction coming from Waypoint 2 (WP_2). So three waypoints are introduced to create a path that fulfils the requirements.

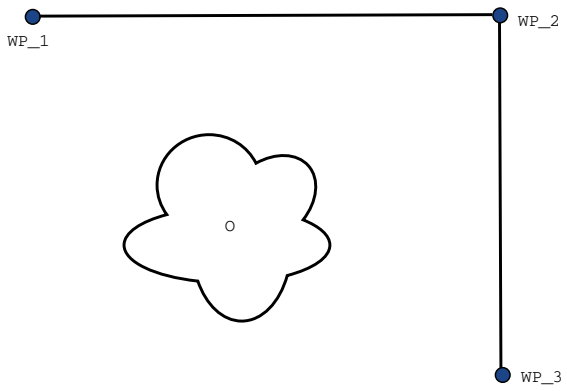


Figure 15.2: WP_1: initial position, WP_2: via point, WP_3: pick up position, O: obstacle.

Without configuring other settings, the robot will make a stop at each waypoint, before continuing the movement. For this task a stop at WP_2 is not optimal since a smooth turn would require less time and energy while still fulfilling the requirements. It is even acceptable that the robot does not reach WP_2 exactly, as long as the transition from the first trajectory to the second happens near this position.

The stop at WP_2 can be avoided by configuring a blend for the waypoint, allowing the robot to calculate a smooth transition into the next trajectory. The primary parameter for the blend is a radius. When the robot is within the blend radius of the waypoint it can start blending and deviate from the original path. This allows for faster and smoother movements, as the robot does not need to decelerate and re-accelerate.

Blend parameters Apart from the waypoints, multiple parameters will influence the blend trajectory (see figure 15.3):

- the blend radius (r)
- the initial and final speed of the robot (at positions p1 and p2, respectively)
- the movement time (e.g. if setting a specific time for a trajectory this will influence the initial/final speed of the robot)
- the trajectory types to blend from and to (MoveL, MoveJ)

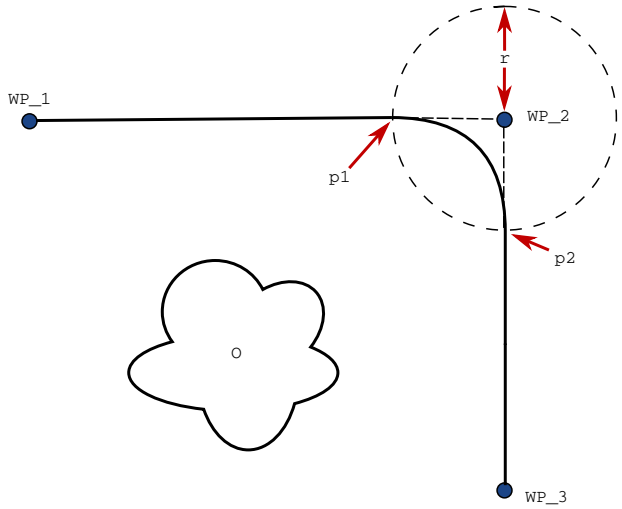


Figure 15.3: Blend over WP_2 with radius r, initial blend position at p1 and final blend position at p2. O is an obstacle.

If a blend radius is set, the robot arm trajectory blends around the waypoint, allowing the robot arm not to stop at the point.

Blends cannot overlap, so it is not possible to set a blend radius that overlaps with the blend radius of a previous or following waypoint as shown in figure 15.4.

Conditional blend trajectories The blend trajectory is affected both by the waypoint where the blend radius is set and the following one in the program tree. That is, in the program in figure 15.5 the blend around WP_1 is affected by WP_2. The consequence of this becomes more apparent when blending around WP_2 in this example. There are two possible ending positions and to determine which is the next waypoint to blend to, the robot must evaluate the current reading of the digital_input[1] already when entering the blend radius. That means the **if...then** expression (or other necessary statements to determine the following waypoint, e.g. variable waypoints) is evaluated before we actually reach WP_2 which is somewhat counter-intuitive when looking at the program sequence. If a waypoint is a stop point and followed by conditional expressions to determine the next waypoint (e.g. the I/O command) it is executed when the robot arm has stopped at the waypoint.

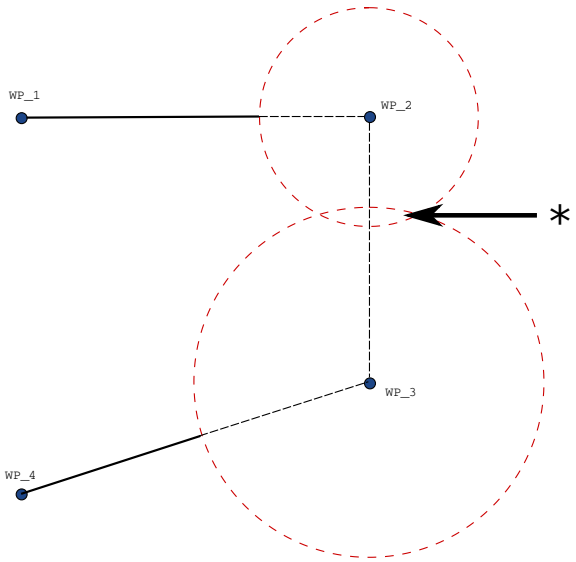


Figure 15.4: Blend radius overlap not allowed (*).

```
MoveL
    WP_I
    WP_1 (blend)
    WP_2 (blend)
    if (digital_input[1]) then
        WP_F_1
    else
        WP_F_2
```

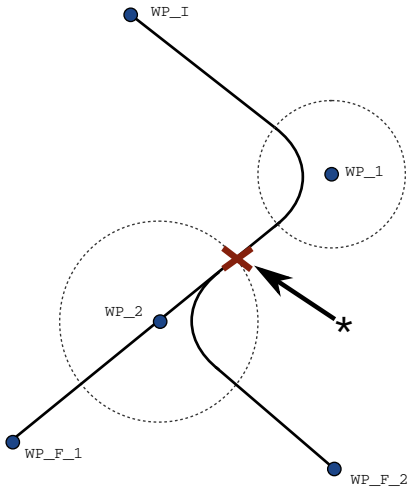


Figure 15.5: `WP_I` is the initial waypoint and there are two potential final waypoints `WP_F_1` and `WP_F_2`, depending on a conditional expression. The conditional `if` expression is evaluated when the robot arm enters the second blend (*).

Blend Trajectories Depending on the movement type (i.e., MoveL, MoveJ, or MoveP), different blend trajectories are generated.

- **Blends in MoveP** When blending in MoveP, the position of the blend follows a circle arc at constant speed. The orientation blends with a smooth interpolation between the two trajectories. You can blend a MoveJ or a MoveL into a MoveP. In such a case, the robot uses the circular

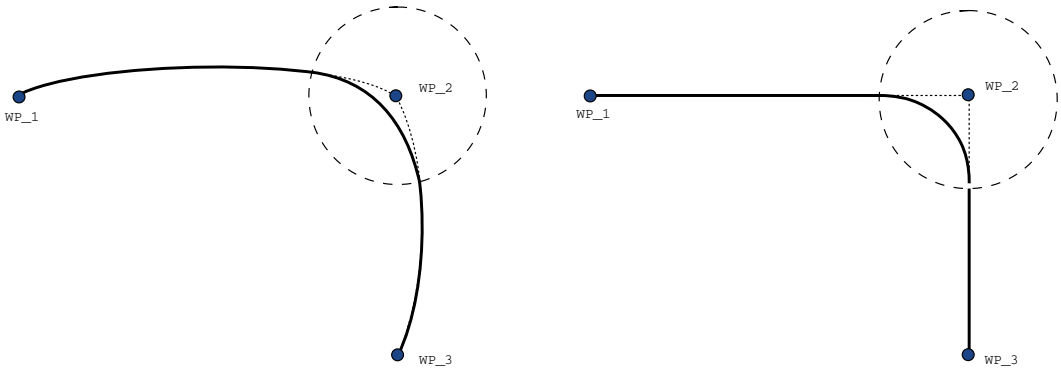
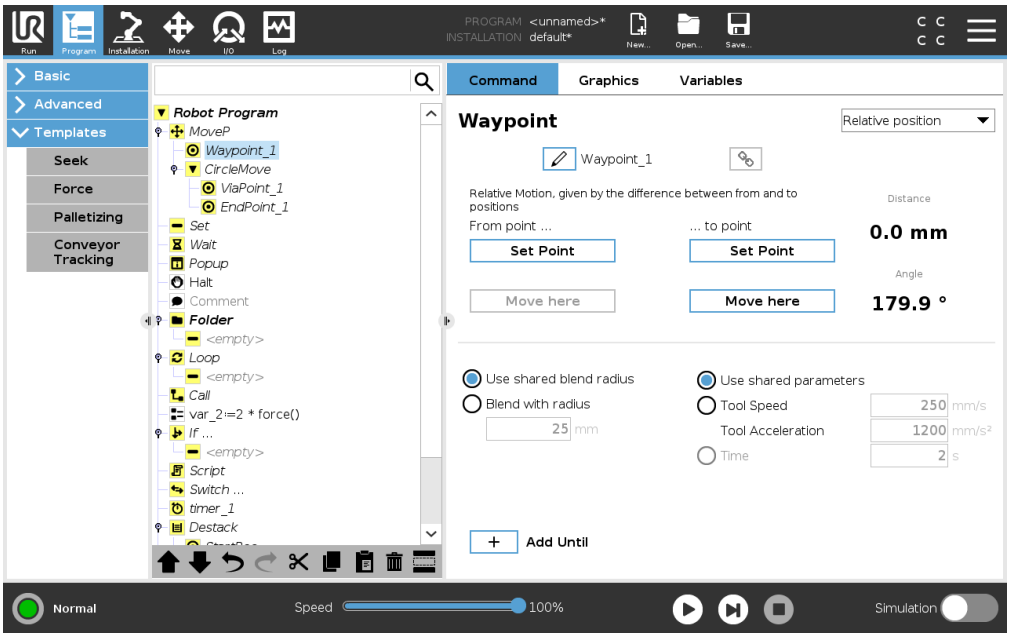


Figure 15.6: Joint space (MoveJ) vs. cartesian space (MoveL) movement and blend.

arc blend of MoveP, and interpolate the speed of the two motions. You cannot blend a MoveP to a MoveJ or a MoveL. Instead, the last waypoint of the MoveP is regarded as a stop point with no blend. You cannot perform a blend if the two trajectories are at an angle close to 180 degrees (reverse direction) because it creates a circular arc with a very small radius which the robot cannot follow at constant speed. This causes a runtime exception in the program which can be corrected by adjusting the waypoints to cause a less sharp angle.

- **Blends involving MoveJ** MoveJ blends cause a smooth curve in joint space. This goes for blends from MoveJ to MoveJ, MoveJ to MoveL and MoveL to MoveJ. The blend produces a smoother and faster trajectory than the movements without a blend (see Figure 15.6). If velocity and acceleration are used for specifying the velocity profile, the blend stays within the blend radius during the blend. If using *time* instead of *velocity* and *acceleration* for specifying the velocity profile of both motions, the blend trajectory follows the trajectory of the original MoveJ. When both motions are time constrained, using blends does not save time.
- **Blends in MoveL** When blending in MoveL, the position of the blend follows a circle arc at constant speed. The orientation blends with a smooth interpolation between the two trajectories. The robot may decelerate on the trajectory before following the circular arc to avoid very high accelerations (e.g., if the angle between the two trajectories are close to 180 degrees).

Relative Waypoint

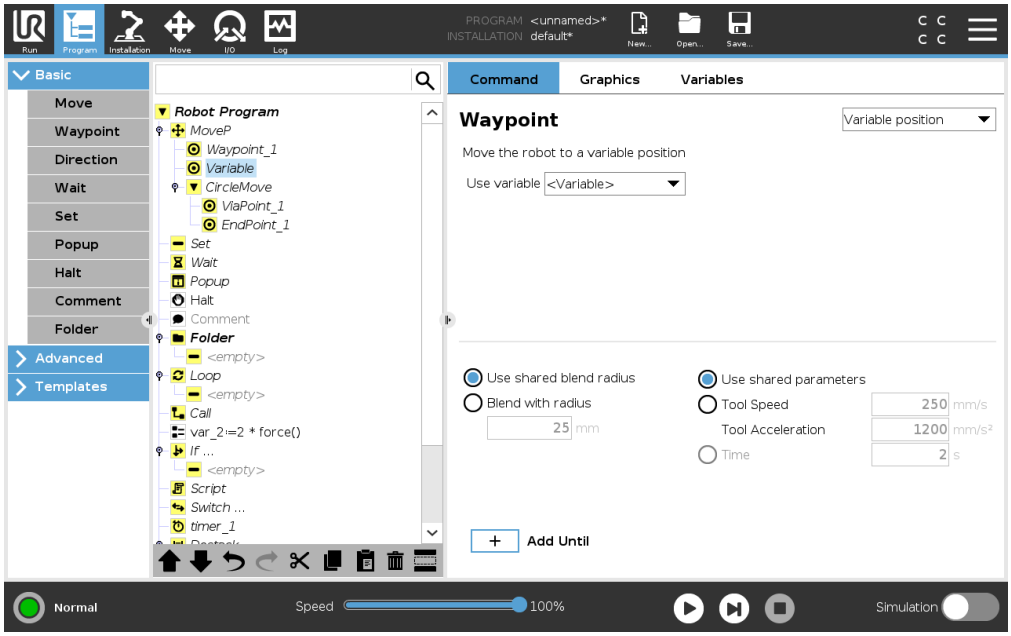


A waypoint with the position given relative to the robot arm's previous position, such as "two centimeters to the left". The relative position is defined as the difference between the two given positions (left to right).

Note: repeated relative positions can move the robot arm out of its workspace.

The distance here is the Cartesian distance between the TCP in the two positions. The angle states how much the TCP orientation changes between the two positions. More precisely, the length of the rotation vector describing the change in orientation.

Variable Waypoint



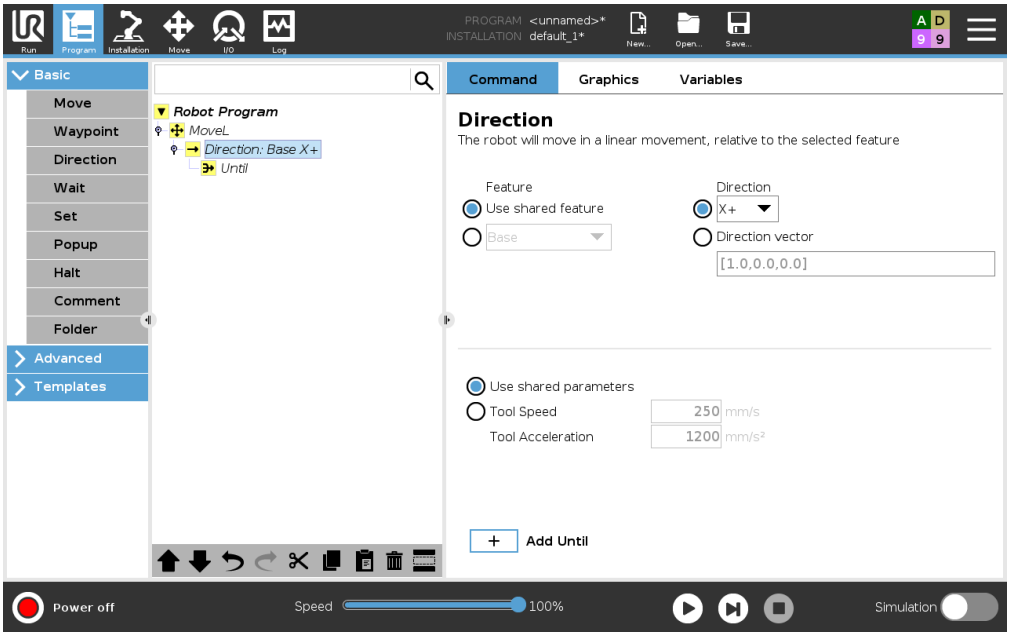
A waypoint with the position given by a variable, in this case `calculated_pos`. The variable has to be a *pose* such as `var=p[0.5,0.0,0.0,3.14,0.0,0.0]`. The first three are *x,y,z* and the last three are the orientation given as a *rotation vector* given by the vector *rx,ry,rz*. The length of the axis is the angle to be rotated in radians, and the vector itself gives the axis about which to rotate. The position is always given in relation to a reference frame or coordinate system, defined by the selected feature. If a blend radius is set on a fixed waypoint and the waypoints preceding and succeeding it are variable or if the blend radius is set on a variable waypoint, then the blend radius will not be checked for overlap `(see 15.5.1)`. If, when running the program, the blend radius overlaps a point, the robot will ignore it and move to the next one.

For example, to move the robot 20 mm along the z-axis of the tool:

```
var_1=p[0,0,0.02,0,0,0]
Movel
  Waypoint_1 (variable position):
    Use variable=var_1, Feature=Tool
```

15.5.2 Direction

The program node **Direction** specifies a motion relative to feature axes or TCPs. The robot moves in along the path specified by the Direction Program Node until that movement is stopped by an **Until** condition.



Adding a Direction Movement

1. Under Basic, tap **Direction** to add a linear movement to your Program Tree.
2. In the Direction field, under Feature, define the linear movement.

Stopping a Direction Movement

1. In the Direction field, tap the **Add Until** button to define and add stop criteria to your Program Tree.

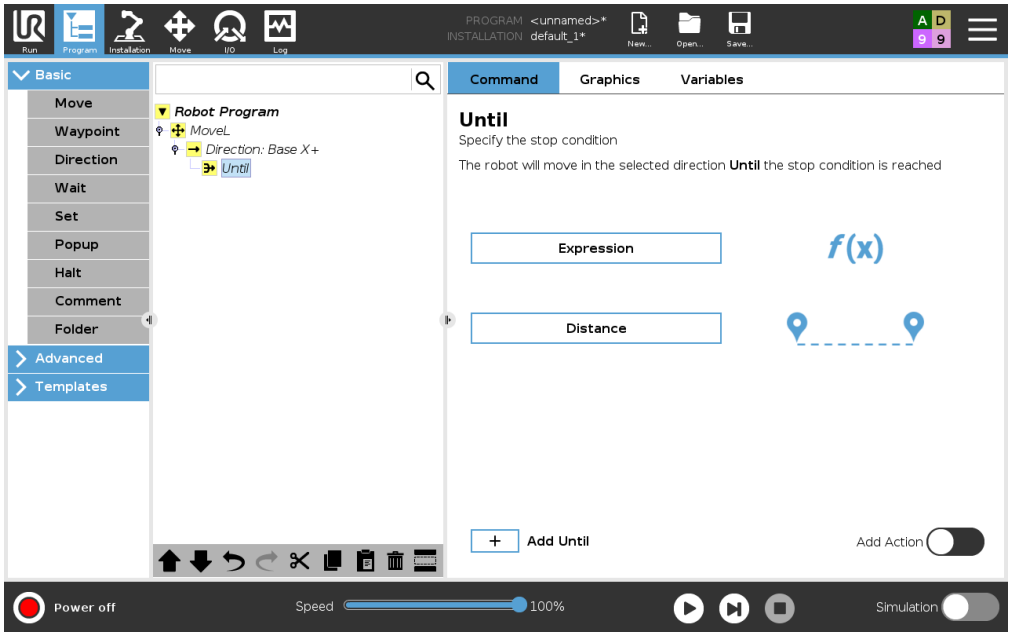
You can add Direction Vector settings, for **Tool Speed** and **Tool Acceleration**, to define the vector direction for linear motion, allowing for advanced uses as:

- defining linear motion relative to multiple feature axes
- computing the direction as a mathematical expression

The Direction Vectors defines a custom code expression that is resolved to a unit vector. For example, Direction vectors of [100,0,0] and [1,0,0] have the exact same effect on the robot; use the Speed Slider to moving along the x-axis at a desired speed. The values of the numbers in the direction vector only matter relative to each other.

Until

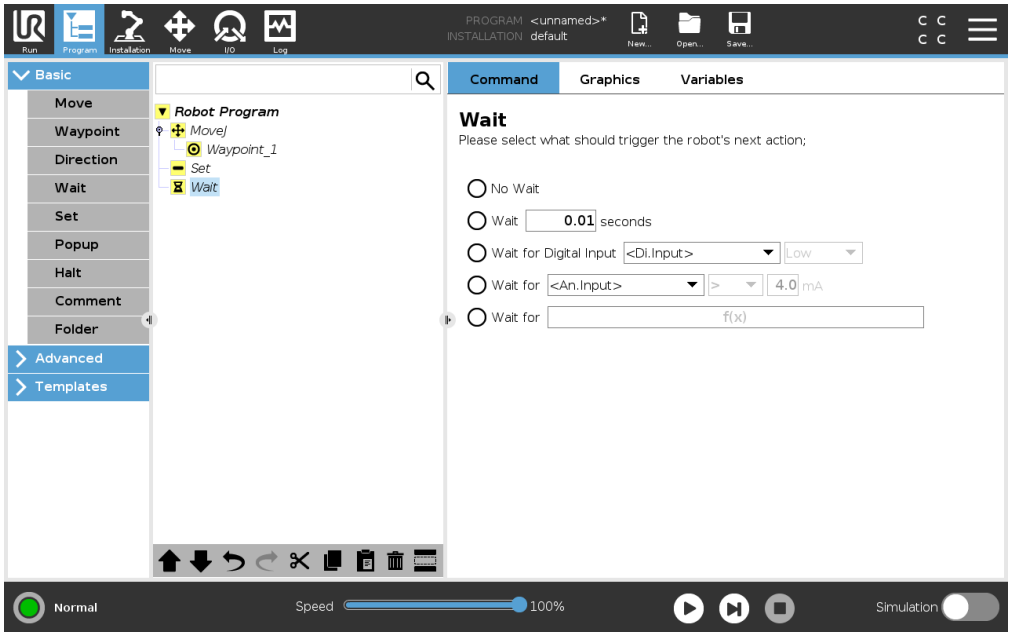
The program node **Until** defines a stop criterion for a motion. The robot moves along a path and stops when contact is detected. In the Progran Tree, you can add Until Nodes under Direction Nodes and Waypoint Nodes. You can add several stop criteria to a single movement. The motion stops when the first **Until** condition is met.



In the **Until** field, you can define the following stop criteria:

- **Distance** This node can be used to stop a Direction move when the robot has moved a certain distance. The velocity is ramped down so the robot stops exactly at the distance.
- **Expression** This node can be used to stop the motion due to a custom program expression. You can use I/Os, variables or script functions to specify the stop condition.

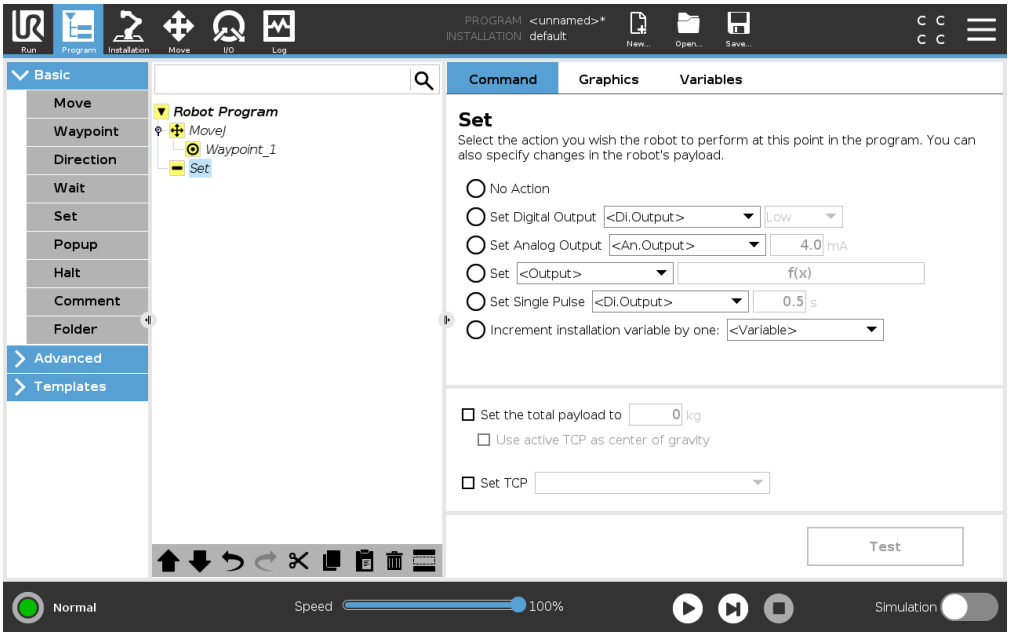
15.5.3 Wait



Wait pauses I/O signal, or expression, for a given amount of time. If **No Wait** is selected, nothing is done.

Note: Once Tool Communication Interface TCI is enabled, the tool analog input is unavailable for **Wait For** selection and expressions.

15.5.4 Set



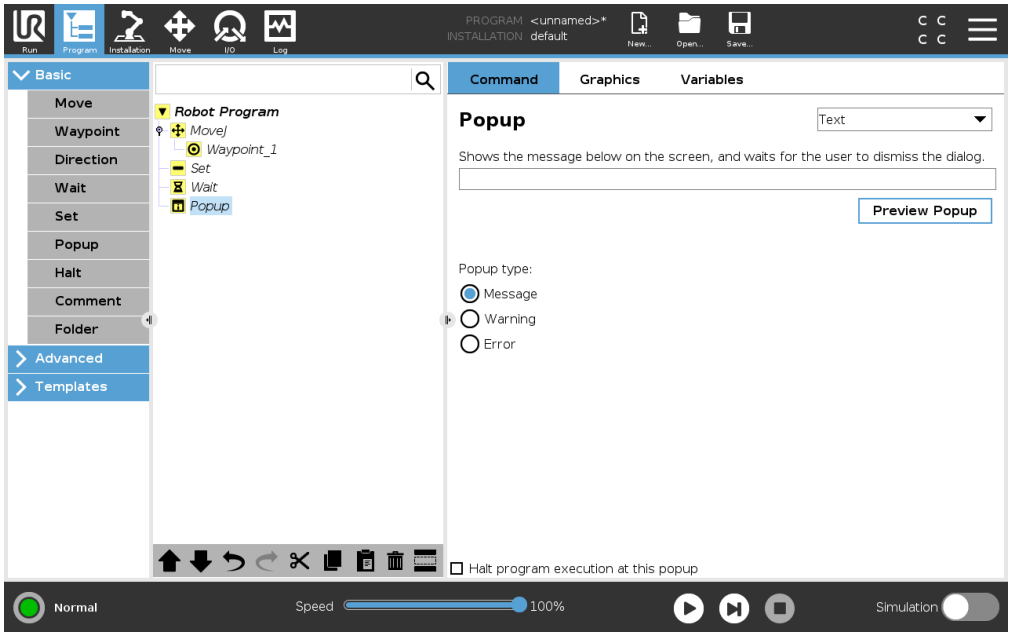
Sets either digital or analog outputs to a given value. Digital outputs can also be set to send a single pulse.

Use the Set command to set the payload of the Robot Arm. You can adjust the payload weight to prevent the robot from triggering a protective stop, when the weight at the tool differs from the expected payload. If the active TCP should not be used as the center of gravity the checkbox must be unchecked.

The active TCP can also be modified using a **Set** command, by selecting the check box and choosing one of the TCP offsets from the menu.

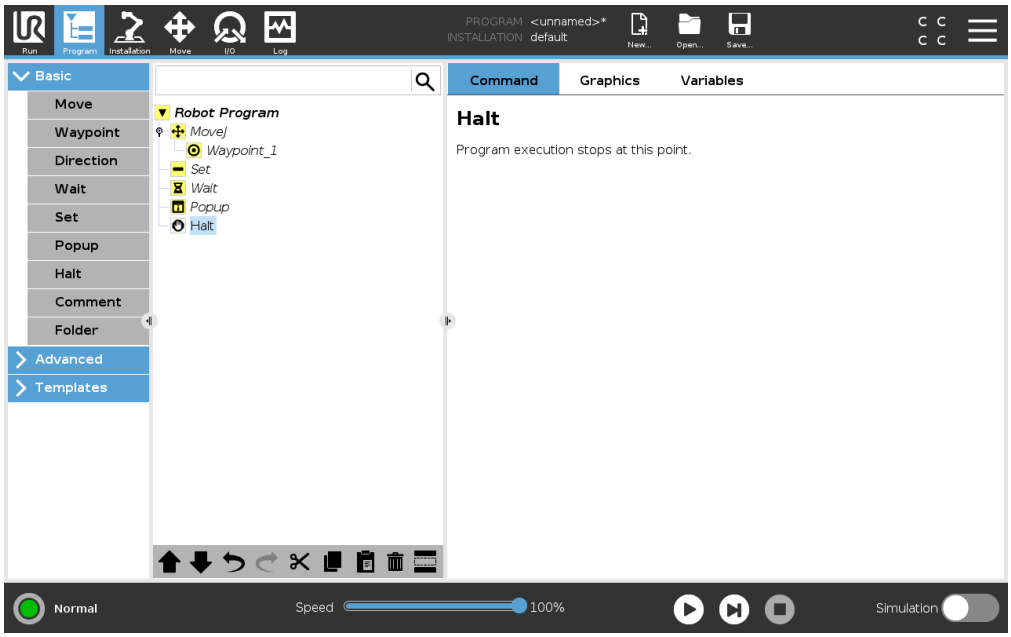
If the active TCP for a particular motion is known at the time of writing of the program, you can use the TCP selection by tapping **Move** in the Basic menu on the left (see 15.5.1). For further information about configuring named TCPs (see 16.1.1).

15.5.5 Popup



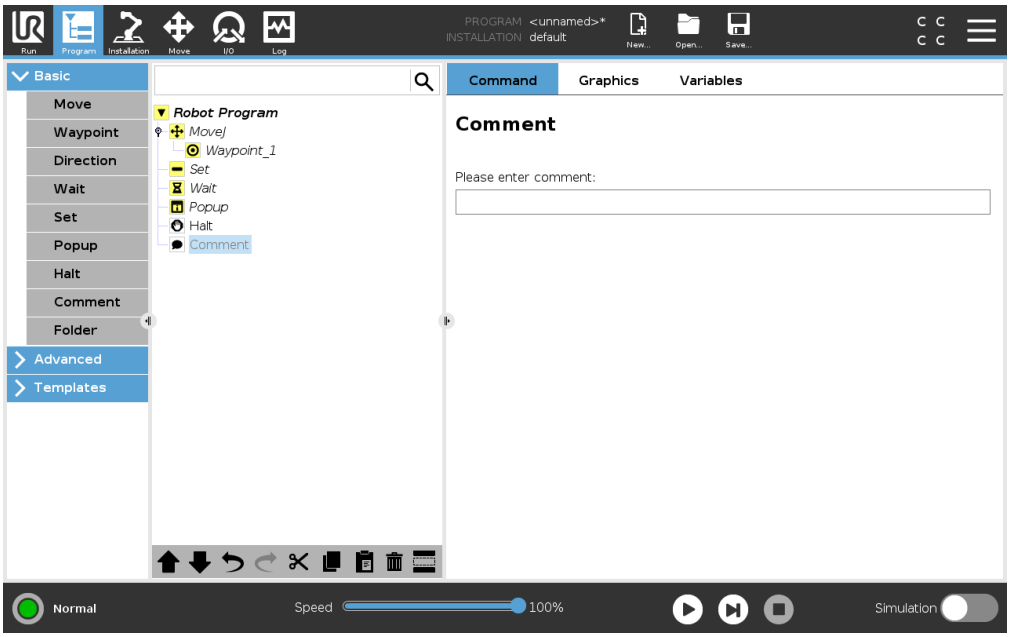
The popup is a message that appears on the screen when the program reaches this command. The style of the message can be selected, and the text itself can be given using the on-screen keyboard. The robot waits for the user/operator to press the “OK” button under the popup before continuing the program. If the “Halt program execution” item is selected, the robot program halts at this popup.
Note: Messages are limited to a maximum of 255 characters.

15.5.6 Halt



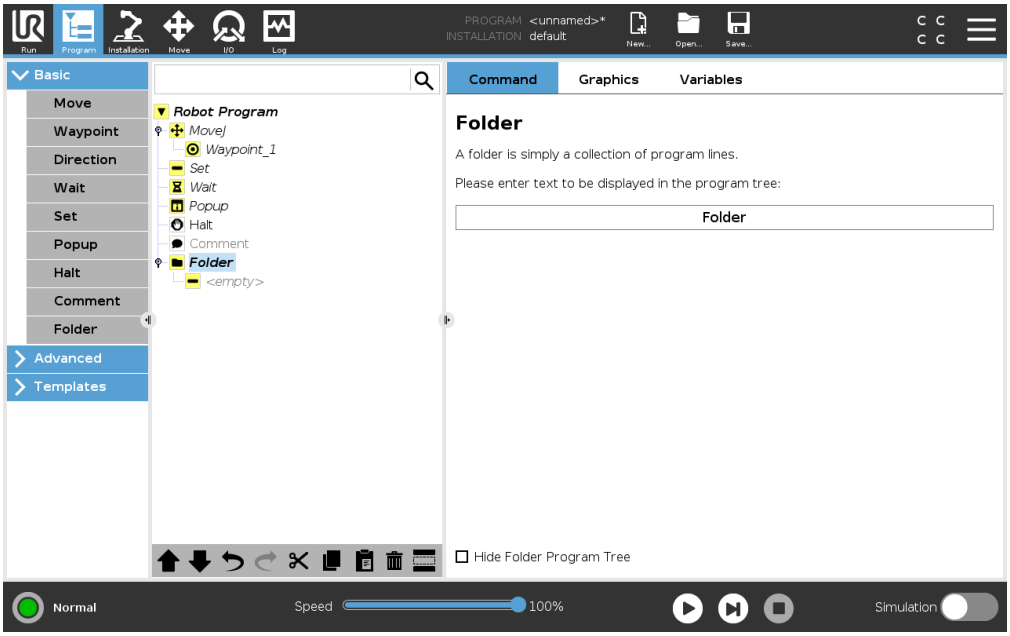
The program execution stops at this point.

15.5.7 Comment



Gives the programmer an option to add a line of text to the program. This line of text does not do anything during program execution.

15.5.8 Folder

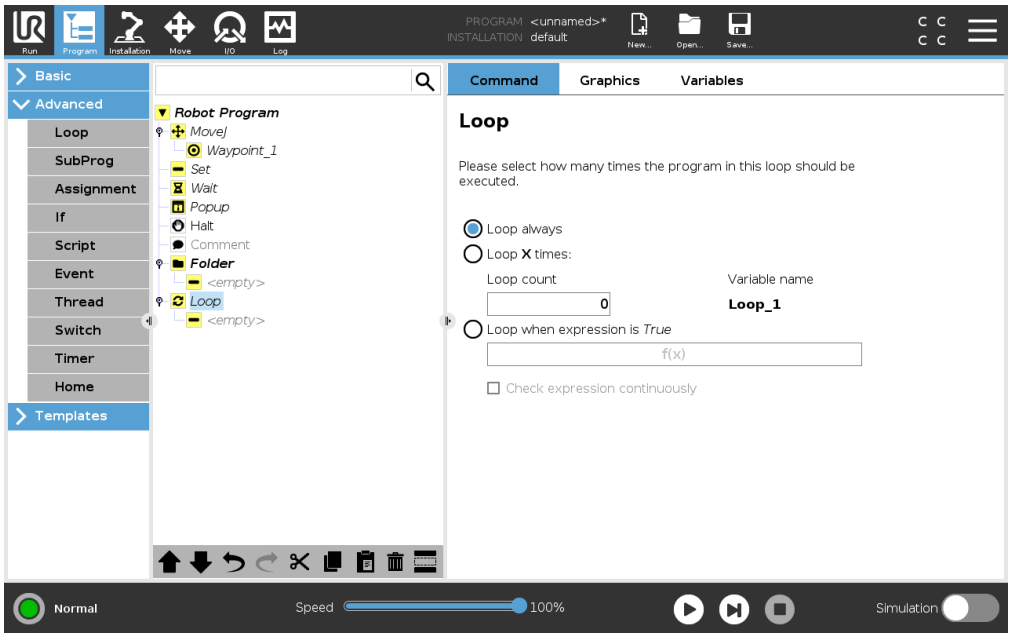


A **Folder** is used to organize and label specific parts of a program, to clean up the program tree, and to make the program easier to read and navigate.

Folders have no impact on the program and its execution.

15.6 Advanced program nodes

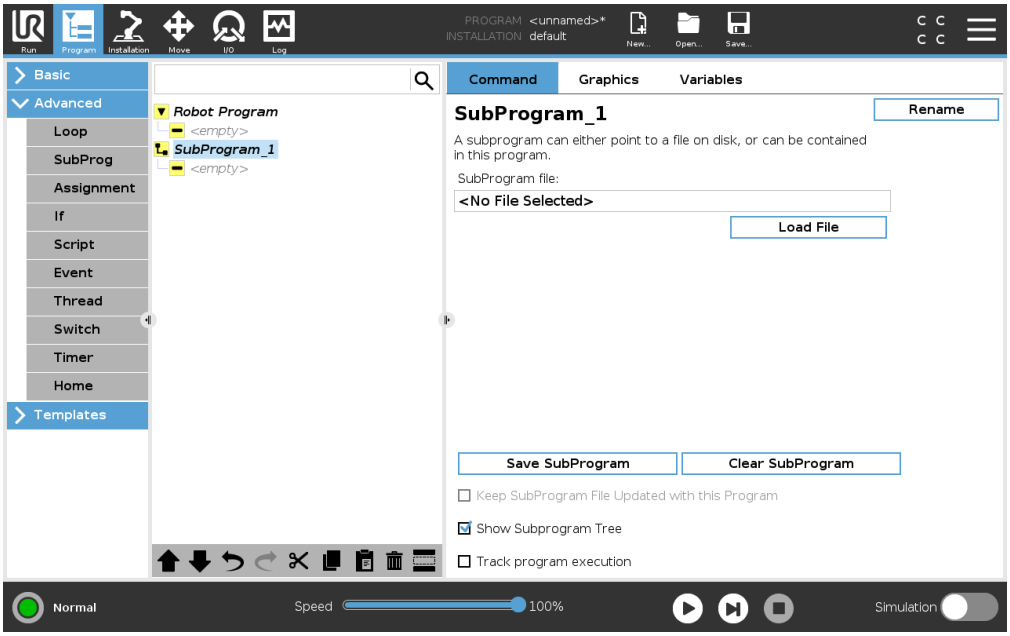
15.6.1 Loop



Loops the underlying program commands. Depending on the selection, the underlying program commands are either looped infinitely, a certain number of times or as long as the given condition is true. When looping a certain number of times, a dedicated loop variable (called `loop_1` in the screen shot above) is created, which can be used in expressions within the loop. The loop variable counts from 0 to $N - 1$.

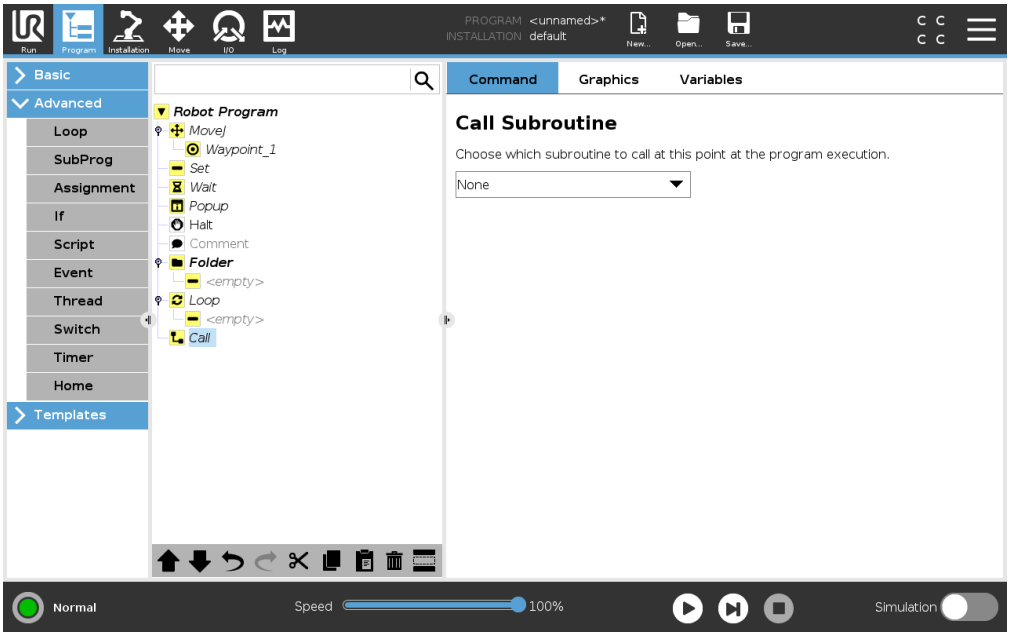
When looping using an expression as end condition, PolyScope provides an option for continuously evaluating that expression, so that the “loop” can be interrupted anytime during its execution, rather than just after each iteration.

15.6.2 SubProgram



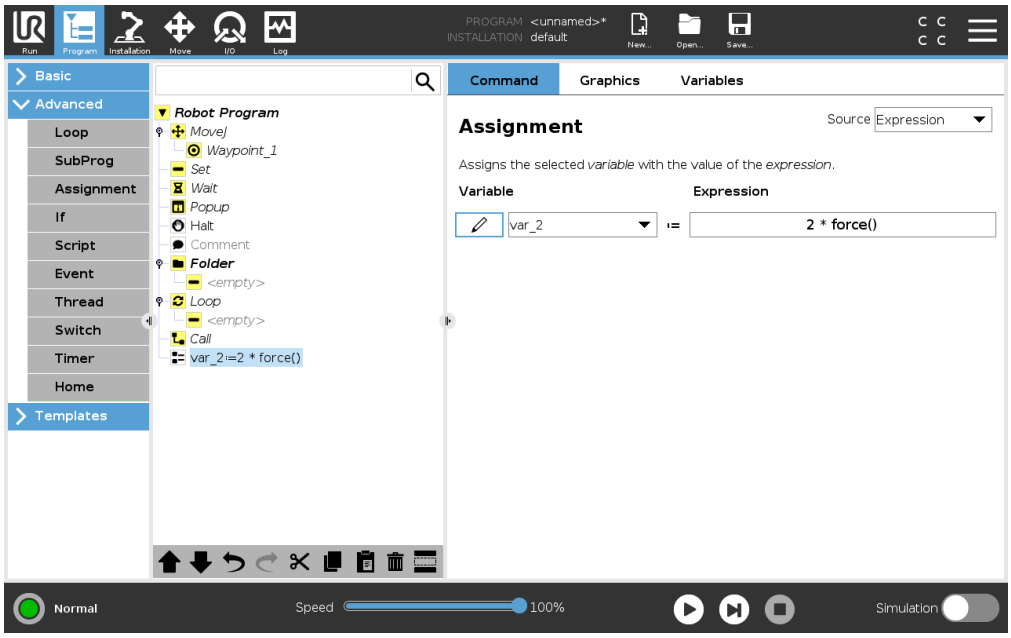
A Sub Program can hold program parts that are needed several places. A Sub Program can be a separate file on the disk, and can also be hidden to protect against accidental changes to the SubProgram.

Call SubProgram



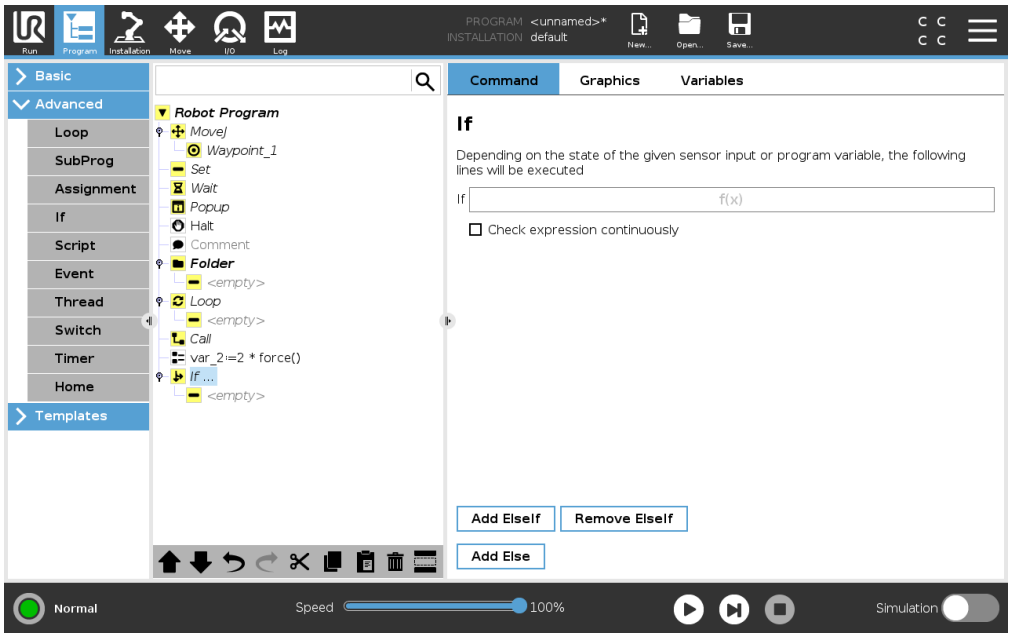
A call to a sub program will run the program lines in the sub program, and then return to the following line.

15.6.3 Assignment



Assigns values to variables. An assignment puts the computed value of the right hand side into the variable on the left hand side. This can be useful in complex programs.

15.6.4 If



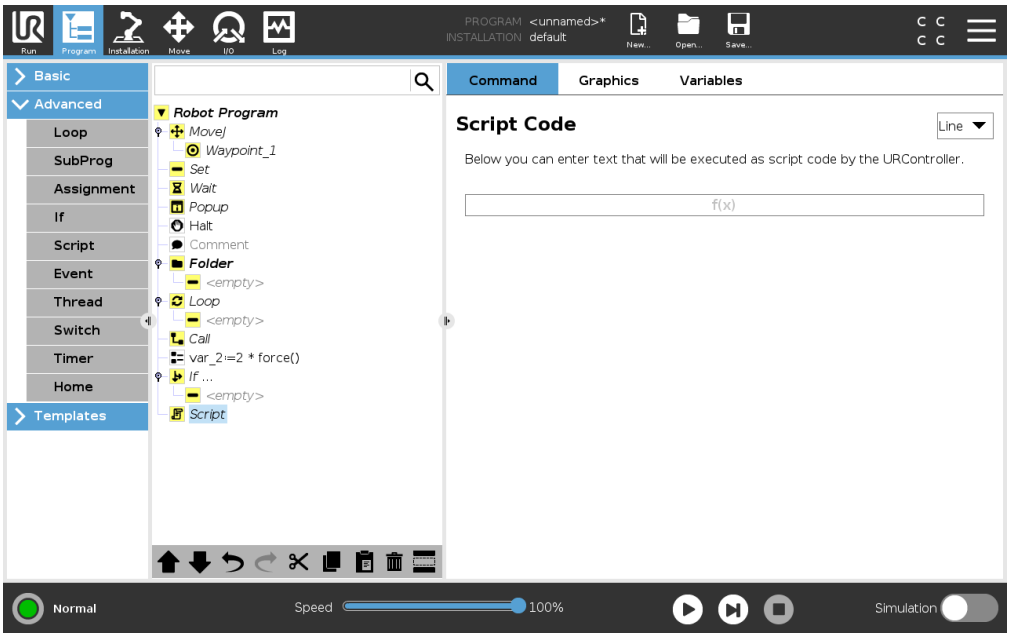
An **If...Else** command construction changes the robot's behavior based on sensor inputs or variable values. Use the Expression Editor to describe the condition under which the robot follows the

statements of this **If** command. If the condition is evaluated as True, the statements within this **If** command are executed.

An **If** command can have several ElseIf statements that can be added and removed using the **Add ElseIf** and the **Remove ElseIf** buttons. However, an **If** command can have only one **Else** statement.

Note: You can select the **Check expression continuously** checkbox to allow the conditions of the **If** command and **ElseIf** statements to be evaluated while the contained lines are executed. If an expression within the **If** command is evaluated as False, the **ElseIf** or **Else** statements are followed.

15.6.5 Script



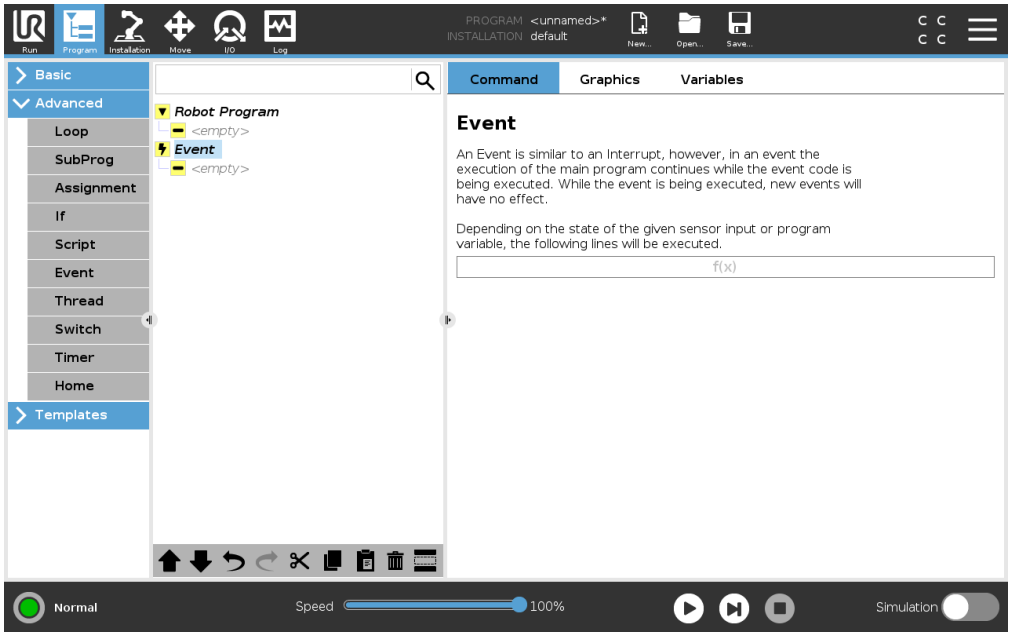
The following options are available in the drop down list under Command:

- **Line** allows you to write a single line of URscript code, using the Expression Editor (`15.1.4`)
- **File** allows you to write, edit or load URscript files.

You can find instructions for writing URscript in the Script Manual on the support website (`http://www.universal-robots.com/support`).

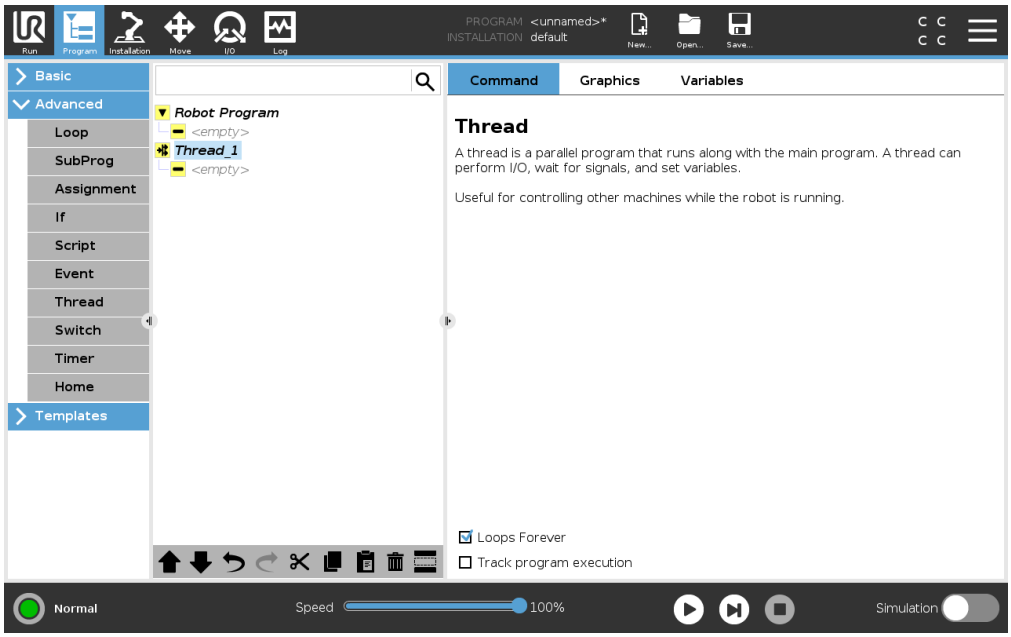
Functions and variables declared in a URscript file are available for use througout the program in the PolyScope.

15.6.6 Event



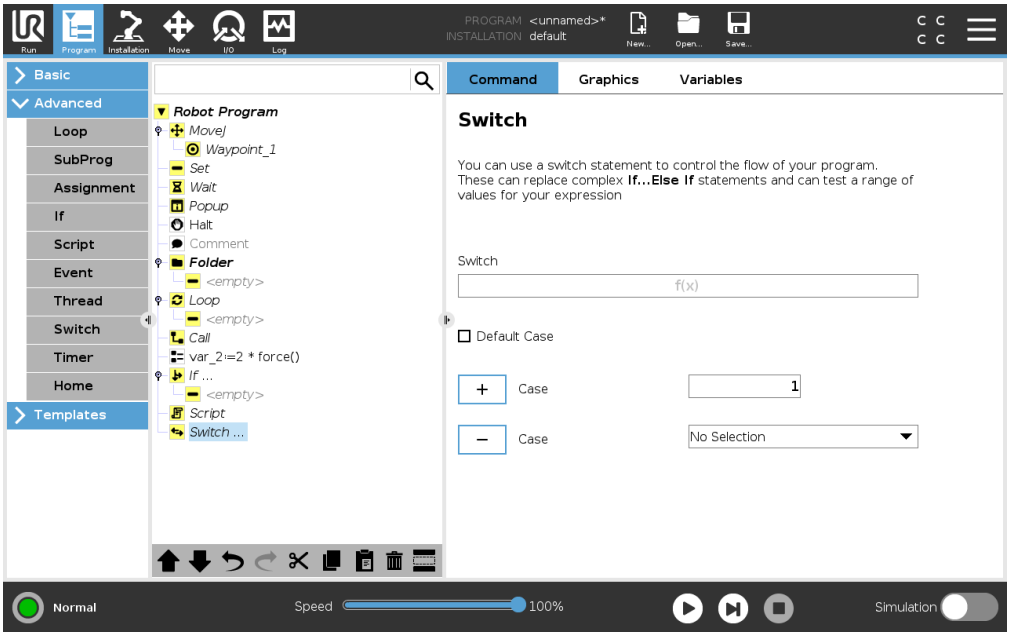
An event can be used to monitor an input signal, and perform some action or set a variable when that input signal goes high. For example, in the event that an output signal goes high, the event program can wait for 200ms and then set it back to low again. This can make the main program code a lot simpler in the case on an external machine triggering on a rising flank rather than a high input level. Events are checked once every control cycle (2ms) .

15.6.7 Thread



A thread is a parallel process to the robot program. A thread can be used to control an external machine independently of the robot arm. A thread can communicate with the robot program with variables and output signals.

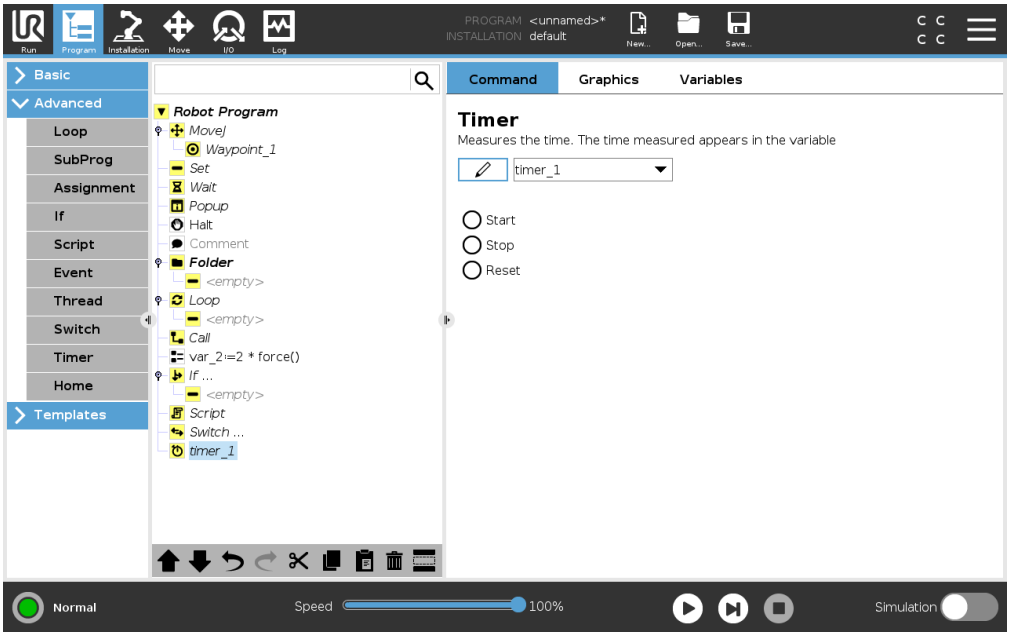
15.6.8 Switch



A **Switch Case** construction can make the robot change behavior based on sensor inputs or variable values. Use the **Expression Editor** to describe the base condition and define the cases under which the robot should proceed to the sub-commands of this `Switch`. If the condition is evaluated to match one of the cases, the lines inside the `Case` are executed. If a `Default Case` has been specified, then the lines will be executed only if no other matching cases were found.

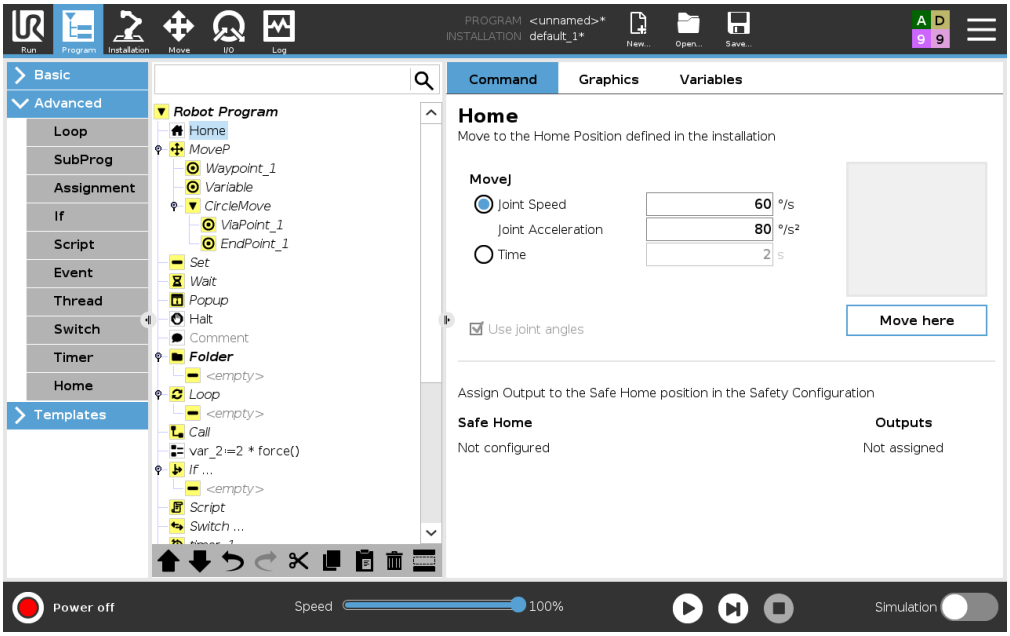
Each `Switch` can have several `Cases` and one `Default Case`. `Switches` can only have one instance of any `Case` values defined. `Cases` can be added using the buttons on the screen. A `Case` command can be removed from the screen for that switch.

15.6.9 Timer



A Timer measures the length of time it takes for specific parts of the program to run. A program variable contains the time passed since a Timer started, and can be seen in the Variables Tab and in the Run Tab.

15.6.10 Home



The Home node uses joint angles to move the robot to a predefined Home position. If defined as a Safe Home position,the Home node displays as Home(Safety) in the Program Tree. If the Home

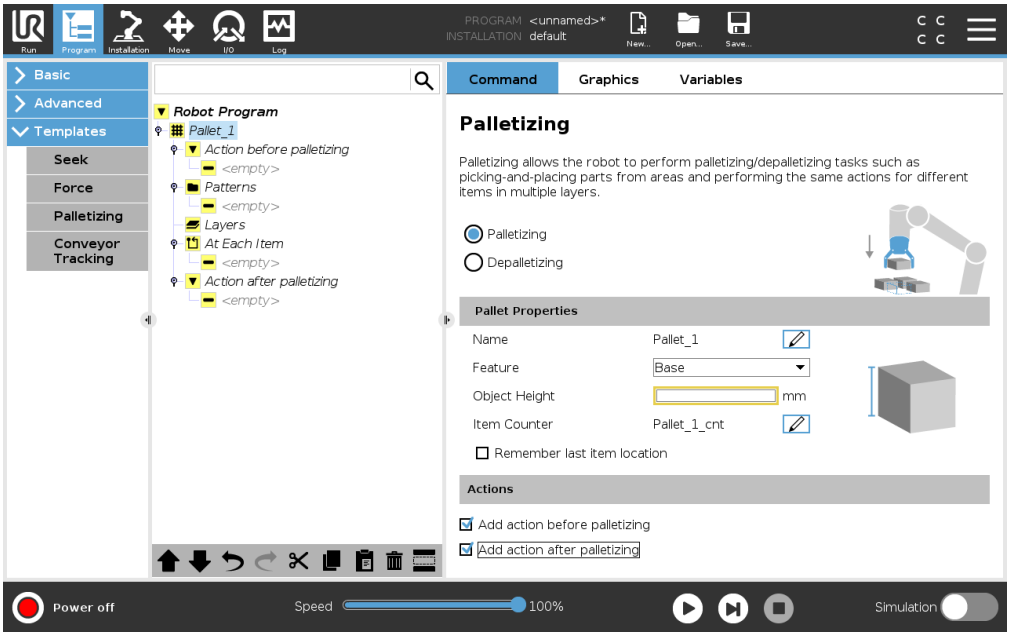
position is out of sync with Safety, the node is undefined.

15.7 Templates

15.7.1 Palletizing

Palletizing is a template to easily program palletizing and depalletizing tasks, picking-and-placing parts (i.e., from trays, fixtures, etc.), and having the robot perform repeatable actions for different items in multiple layers with different patterns. You can create different patterns and apply them to specific layers. You can also place a separator between each layer (see 15.7.1). Furthermore, you can use Features from Pallet Properties to easily adjust the placement of your pallet. To learn about Features, see 16.3. Follow the **Creating a Palletizing Program** section below to use the Palletizing template.

Creating a Palletizing Program



1. Decide if you want to teach a Feature (see 16.3) or use a Base as a reference plane.
2. In the **Program Tab**, under **Templates**, tap **Palletizing**.
3. On the Palletizing screen, select one of the following actions depending on the desired action.
   (a) Select **Palletizing** to organize items onto a pallet.
   (b) Select **Depalletizing** to remove items from a pallet.
4. Under **Pallet Properties**, specify the Name, Feature (see Step 1), Object Height, and Item Counter name for your program. Select the **Remember last item location** box if you want the robot to restart at the item it was handling when it stopped.
5. On the Palletizing screen, under **Actions**, add additional actions to be performed before or after palletizing sequence by selecting the following:
   (a) **Add Action Before Palletizing**: These actions are performed before starting to palletize.

(b) **Add Action After Palletizing**: These actions are performed after finishing palletizing.

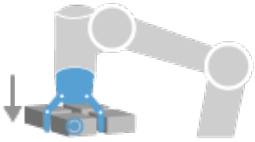
6. On the Program Tree, tap the **Patterns** node to designate patterns for your layers. You can create the following type of patterns: Line, Grid, or Irregular (see figure below). On this screen, you can select if you want to include a separator between layers (see 15.7.1).
7. Tap the pattern node(s) on the Program Tree to teach the robot layer-specific positions (e.g., start/end points, grid corners, and/or number of items). See 15.5.1 for teaching instructions. All positions must be taught at the bottom of the pallet. To duplicate a pattern, tap the **Duplicate pattern** button on the Pattern node screen that you wish to duplicate.

	**Line** To teach the positions, select each item in the Program Tree: • Start_Item_1 • End_Item_1 Insert the number of items in your sequence using the **Items** text box at the bottom of the screen.
	**Grid** To teach the positions, select each item in the Program Tree: • Corner_Item_1 • Corner_Item_2 • Corner_Item_3 • Corner_Item_4 Insert the number of rows and columns in the appropriate text boxes to set the dimensions of the pattern.
	**Irregular** To teach the positions, select each item in the Program Tree: • Item_1 • Item_2 • Item_3 Tap **Add Item** to add and identify a new item in the sequence.

8. In the Program Tree, tap the **Layers** node to configure the layers of your palletizing sequence. Use the **Choose Pattern** drop-down menu to select the pattern for each layer. Tap the **Add layer** button to add additional layers to your program. Layers must be added in the correct order, as they cannot be reordered later.
9. In the Program Tree, tap **At Each Item** node. Choose to use the default option (A) At Each Item Wizard, or (B) Manually Configure At Each Item. Instructions for each option are below.

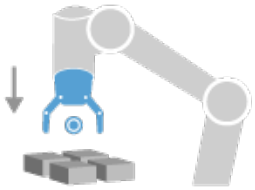
(A) At Each Item Wizard The At Each Item Wizard assists in defining the actions performed at each item on a pallet, such as the ReferencePoint, the Approach Waypoint, ToolActionPoint Waypoint, and Exit Waypoint (described in the table below). The Approach and Exit Waypoints for each item remains in the same orientation and direction regardless of the different items' orientation.

1. Tap the **At Each Item** node on the Program Tree.
2. On the At Each Item screen, tap **Next**.
3. Tap the **Move Here** button. Then, hold the **Auto** button or use the **Manual** button to move the robot to the ReferencePoint. Tap the **Continue** button. Tap **Next**.
4. Tap **Set Waypoint** to teach the Approach Waypoint (see 15.5.1). Tap **Next**.
5. Repeat Step 3.
6. Tap **Set Waypoint** to teach the Exit Waypoint (see 15.5.1). Tap **Next**.
7. Tap **Finish**.
8. You can now add appropriate gripper action nodes in the Tool Action folder in the Program Tree.



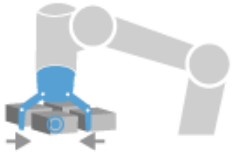
ToolActionPoint

ToolActionPoint Waypoint: The location and position you want the robot to be in when conducting an action for each item in a layer. The ToolActionPoint Waypoint is the ReferencePoint by default, but it can be edited in the Program Tree by tapping the ToolActionPoint Waypoint node.
When using the wizard, the ReferencePoint is the first position in the first defined layer on the pallet. The ReferencePoint is used to teach the robot the Approach Waypoint, ToolActionPoint Waypoint, and Exit Waypoint for each item in a layer.



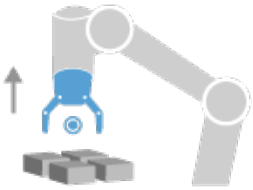
Approach

Approach Waypoint: The collision-free position and direction you want the robot to take when approaching an item in a layer.



Tool Action

Tool Action: The action you want the robot attachment to perform for each item.



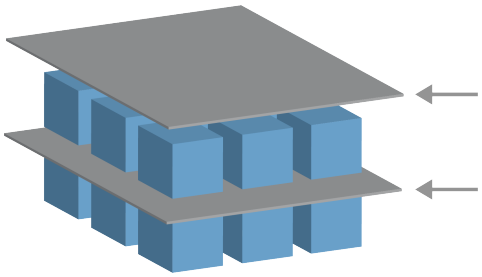
Exit

Exit Waypoint: The position and direction you want the robot to take when moving away from an item in a layer.

(B) Manual Configuration

1. Tap the **At Each Item** node on the Program Tree.
2. On the **At Each Item** start screen, tap **Manual Configuration**.
3. Use the drop-down menus to select a Pattern and a ReferencePoint item. Tap the **Use this ReferencePoint** button to set the ReferencePoint.
4. Move the robot to the ReferencePoint by tapping **Move Here**.
5. Tap the Approach node in the Program Tree to teach the robot the Approach Waypoint (see 15.5.1). The Approach Waypoint remains in the same orientation and direction regardless of the different items' orientation.
6. Tap the At Each Item node in the Program Tree. Repeat Step 4.
7. Tap the **Exit** node in the Program Tree to teach the robot the Exit Waypoint (see 15.5.1).
8. You can now add appropriate gripper action nodes in the Tool Action folder in the Program Tree.

Adding a Separator Between Layers in a Palletizing Sequence



Separators, such as paper or Styrofoam, can be placed between layers in a palletizing sequence. To add separators between layers, follow the instructions below:

1. On the Program Tree, select the **Patterns** node.
2. On the **Patterns** screen, select **Separator** and define the height using the **Separator Height** text box. If the height is not defined, the program will not run.
3. Select **Layers** in the Program Tree. On the Layers screen, select which layers you want the separators to go between (separators are automatically placed between each layer).
4. Tap the **Separator** node in the Program Tree. Tap **Set Separator** to teach the Separator Position.
5. Choose between using the default option (A) Separator Wizard, or (B) Manually Configure the Separator sequence. Instructions for each option are below.

When the wizard is complete, or if you cancel the wizard, a template appears in the Program Tree under **Separator Action**. In addition to the Tool Action folder under the Separator Action node, you can select one of the following folders:

- **Pick Up Separator** to program the robot to pick up separators for palletizing
- **Drop Off Separator** to drop off separators for depalletizing

(A) Separator Wizard

1. Tap the **Separator Action** node on the Program Tree.
2. On the Separator Action screen, tap **Next**.
3. Tap the **Move Here** button and hold the **Auto** button or use the **Manual** button to move the robot to the Separator Point. Tap the **Continue** button. Tap **Next**.
4. Tap **Set Waypoint** to teach the Approach Waypoint (see 15.5.1). Tap **Next**.
5. Repeat Step 3.
6. Tap **Set Waypoint** to teach the Exit Waypoint (see 15.5.1). Tap **Next**.
7. Tap **Finish**.
8. You can now add appropriate action nodes in the Pick Up Separator, Drop Off Separator, and Tool Action folders in the Program Tree.

(B) Manual Configuration

1. Tap the **Separator Action** node on the Program Tree.
2. On the **Separator Action** start screen, tap **Manual Configuration**.
3. Move the robot to the Separator Point by tapping **Move to Separator Point**.
4. Tap the Approach node in the Program Tree to teach the robot the Approach Waypoint (see 15.5.1).
5. Tap the Separator Action node in the Program Tree. Repeat Step 3.
6. Tap the Exit node in the Program Tree to teach the robot the Exit Waypoint (see 15.5.1).
7. You can now add appropriate action nodes in the Pick Up Separator, Drop Off Separator, and Tool Action folders in the Program Tree.

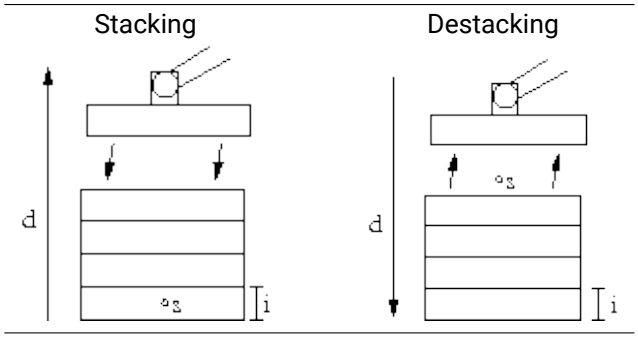
Options to Customize A Palletizing Program

You can customize your palletizing program in the following ways:

- If your pallet needs to be adjusted or re-positioned after you have created a palletizing program, you only need to re-teach the pallet Feature (see 16.3) because the palletizing sequence is fixed relative to the Feature. Thus, all other program components automatically adjust to the newly taught position.
- You can edit the properties of the move commands (see 15.5.1).
- You can change the speeds and blends radii (see 15.5.1).
- You can add other program nodes to the At Each Item sequence or the Separator Action sequence.

15.7.2 Seek

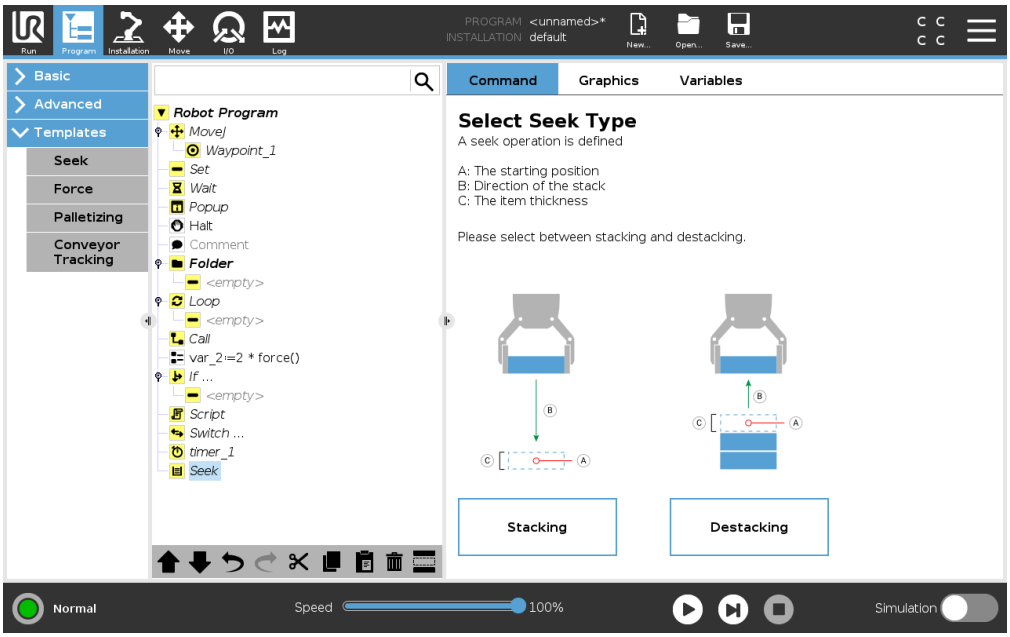
A seek function uses a sensor to determine when the correct position is reached to grab or drop an item. The sensor can be a push button switch, a pressure sensor or a capacitive sensor. This function is made for working on stacks of items with varying item thickness, or where the exact positions of the items are not known or too hard to program.



When programming a seek operation for working on a stack, one must define s the starting point, d the stack direction and i the thickness of the items in the stack.

On top of this, one must define the condition for when the next stack position is reached, and a special program sequence that will be performed at each of the stack positions. Also speed and accelerations need to be given for the movement involved in the stack operation.

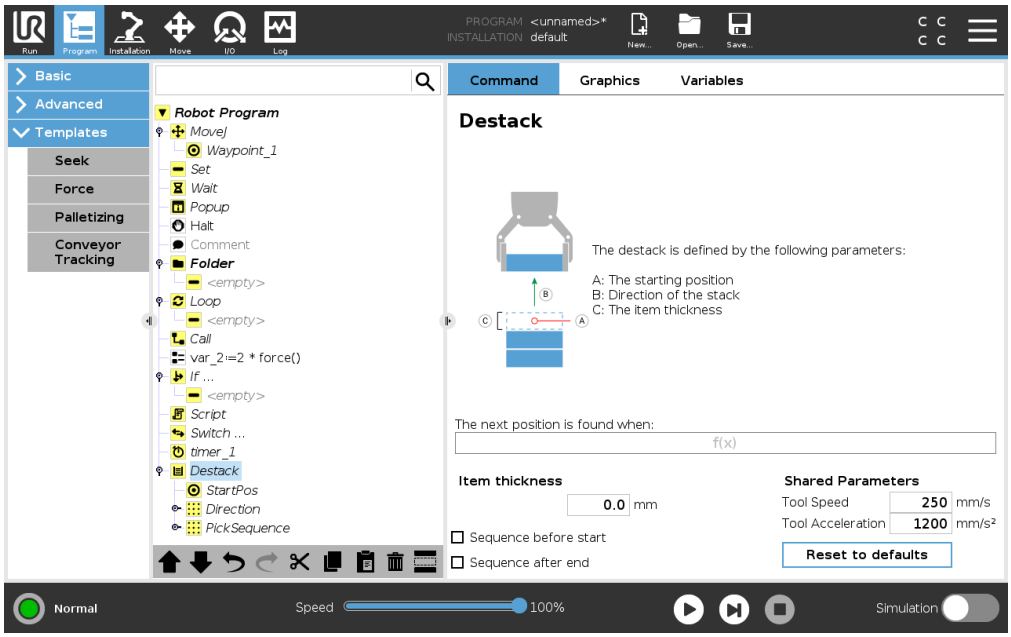
Stacking



When stacking, the robot arm moves to the starting position, and then moves *opposite* the direction to search for the next stack position. When found, the robot remembers the position and performs the special sequence. The next time round, the robot starts the search from the remembered position incremented by the item thickness along the direction. The stacking is finished when the stack

height is more than some defined number, or when a sensor gives a signal.

Destacking

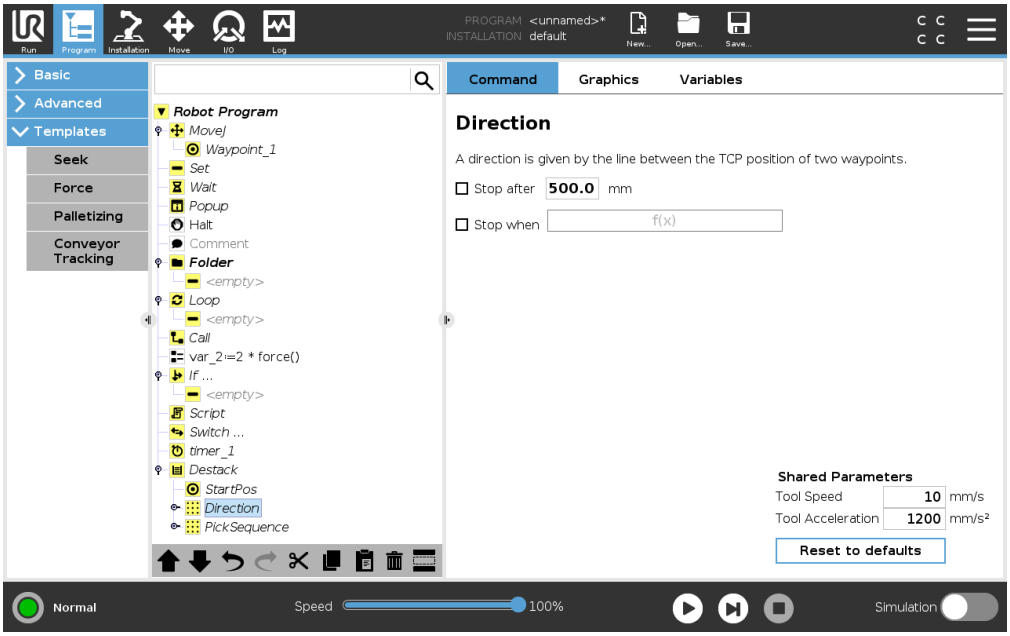


When destacking, the robot arm moves from the starting position in the given direction to search for the next item. The condition on the screen determines when the next item is reached. When the condition becomes satisfied, the robot remembers the position and performs the special sequence. The next time round, the robot starts the search from the remembered position, incremented by the item thickness along the direction.

Starting position

The starting position is where the stack operation starts. If the starting position is omitted, the stack starts at the robot arm's current position.

Direction



The direction is given by two positions, and is calculated as the position difference from the first positions TCP to the second positions TCP.
Note: A direction does not consider the orientations of the points.

Next Stacking Position Expression

The robot arm moves along the direction vector while continuously evaluating whether the next stack position has been reached. When the expression is evaluated to `True` the special sequence is executed.

"BeforeStart"

The optional `BeforeStart` sequence is run just before the operation starts. This can be used to wait for ready signals.

"AfterEnd"

The optional `AfterEnd` sequence is run when the operation is finished. This can be used to signal conveyor motion to start, preparing for the next stack.

Pick/Place Sequence

The Pick/Place Sequence is a special program sequence performed at each stack position, similar to the Pallet operation . .

15.7.3 Force

In the robot workspace **Force mode** allows for compliance and force in selectable axes. All robot arm movements under a **Force** command are in **Force mode**. When the robot arm is moving in **Force mode**, it is possible to select one or more axes there the robot arm is compliant. The robot

arm complies with the environment along a compliant axes. This means the robot arm automatically adjusts its position in order to achieve the desired force. It is also possible to make the robot arm itself apply a force to its environment, e.g. a workpiece.

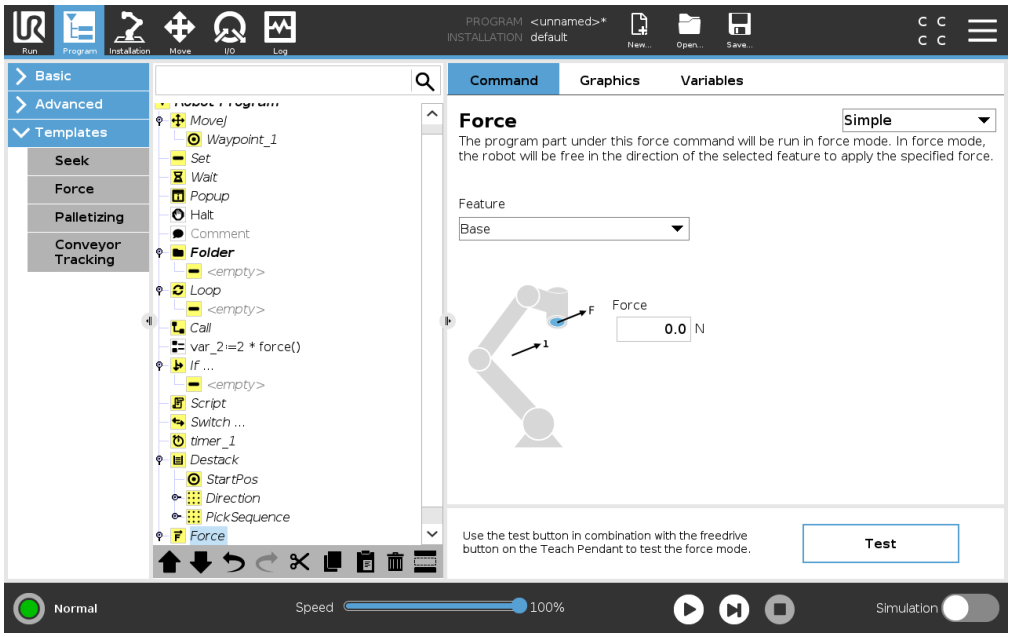
Force mode is suited to applications where the actual TCP position along a predefined axis is not important, but instead a desired force along that axis is required. For example if the robot TCP rolls against a curved surface, pushes or pulls a workpiece. **Force mode** also supports applying certain torques around predefined axes.

Note: if no obstacles are met in an axis where a non-zero force is set, the robot arm attempts to accelerate along that axis.

Although an axis is selected to be compliant, the robot program still tries to move the robot along that axis. However, force control assures that the robot arm still approaches the specified force.

WARNING:

1. Avoid high deceleration just before entering force mode.
2. Avoid high acceleration in force mode, since it decreases force control accuracy.
3. Avoid movements parallel to compliant axes before entering force mode.



Feature selection

The **Feature menu** is used to select the coordinate system (axes) the robot will use while it is operating in force mode. The features in the menu are those which have been defined in the installation (see 16.3).

Force mode type

The are four different types of force mode each determining the way in which the selected feature will be interpreted.

- **Simple**: Only one axis will be compliant in force mode. The force along this axis is adjustable. The desired force will always be applied along the z-axis of the selected feature. However, for Line features, it is along their y-axis.
- **Frame**: The Frame type allows for more advanced usage. Here, compliance and forces in all six degrees of freedom can be independently selected.
- **Point**: When Point is selected, the task frame has the y-axis pointing from the robot TCP towards the origin of the selected feature. The distance between the robot TCP and the origin of the selected feature is required to be at least 10 mm. Note that the task frame will change at runtime as the position of the robot TCP changes. The x- and z-axis of the task frame are dependent on the original orientation of the selected feature.
- **Motion**: Motion means that the task frame will change with the direction of the TCP motion. The x-axis of the task frame will be the projection of the TCP movement direction onto the plane spanned by the x- and y-axis of the selected feature. The y-axis will be perpendicular to the robot arm's motion, and in the x-y plane of the selected feature. This can be useful when de-burring along a complex path, where a force is needed perpendicular to the TCP motion.
  Note: when the robot arm is not moving: If force mode is entered with the robot arm standing still, there will be no compliant axes until the TCP speed is above zero. If later, while still in force mode, the robot arm is again standing still, the task frame has the same orientation as the last time the TCP speed was larger than zero.

For the last three types, the actual task frame can be viewed at runtime on the graphics tab (see 15.3), when the robot is operating in force mode.

Force value selection

- Force or torque value can be set for compliant axes, and robot arm adjusts its position to achieve the selected force.
- For non-compliant axes robot arm will follow the trajectory set by the program.

For translational parameters, the force is specified in Newtons [N] and for rotational the torque is specified in Newton meters [Nm].

NOTE:
You must do the following:

- Use `get_tcp_force()` script function in separate thread, to read actual force and torque.
- Correct wrench vector, if actual force and/or torque is lower than requested.

Speed limits

Maximum Cartesian speed can be set for compliant axes. The robot moves at this speed in force control, as long as it does not come into contact with an object.

Test force settings

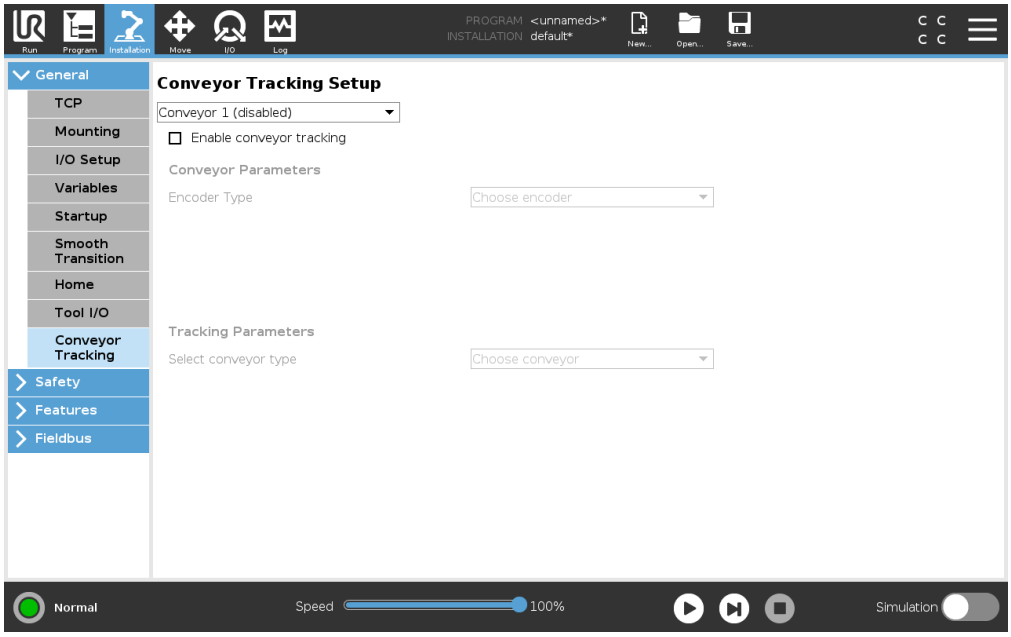
The on/off button, labelled **Test**, toggles the behavior of the **Freedrive** button on the back of the Teach Pendant from normal Freedrive mode to testing the force command.

When the **Test button** is on and the **Freedrive** button on the back of the Teach Pendant is pressed, the robot will perform as if the program had reached this force command, and this way the settings can be verified before actually running the complete program. Especially, this possibility is useful for verifying that compliant axes and forces have been selected correctly. Simply hold the robot TCP using one hand and press the **Freedrive** button with the other, and notice in which directions the robot arm can/cannot be moved. Upon leaving this screen, the Test button automatically switches off, which means the **Freedrive** button on the back of the Teach Pendant is again used for regular **Freedrive** mode.

Note: The **Freedrive** button will only be effectual when a valid feature has been selected for the Force command.

15.7.4 Conveyor Tracking

Conveyor Tracking allows the Robot Arm to track the movement of up to two conveyors. Conveyor Tracking is defined in the Installation Tab (see section 16.1.9).



The Conveyor Tracking program node is available in the Program Tab under Templates. All movements under this node are allowed while tracking the conveyor, but they are relative to the motion of the conveyor belt. Blends are not allowed when exiting Conveyor Tracking, so the robot stops completely before making the next motion.

Tracking a Conveyor

1. In the Header, tap **Program**.
2. Tap **Templates** and select **Conveyor Tracking** to add a Conveyor Tracking node to the Program Tree. Any movements listed under the Conveyor Tracking node tracks the movement of the conveyor.
3. Under Conveyor Tracking, in the Select Conveyor dropdown list, select **Conveyor 1** or **Conveyor 2** to define which conveyor must be tracked.

15.8 The First Program

A program is a list of commands telling the robot what to do. PolyScope allows people with only little programming experience to program the robot. For most tasks, programming is done entirely using the touch panel without typing in any cryptic commands.

Tool motion is the part of a robot program that teaches the Robot Arm how to move. In PolyScope, tool motions are set using a series of **waypoints**. The combined waypoints form a path that the Robot Arm follows. A waypoint is set by using the Move Tab, manually moving (teaching) the robot to a certain position, or it can be calculated by software. Use the Move tab (see 17) to move the Robot Arm to a desired position, or teach the position by pulling the Robot Arm into place while holding the Freedrive button at the top of the Teach Pendant.

Besides moving through waypoints, the program can send I/O signals to other machines at certain points in the robot's path, and perform commands like **if...then** and **loop**, based on variables and I/O signals.

The following is a simple program that allows a Robot arm that has been started up, to move between two waypoints.

1. In the PolyScope Header **File Path**, tap **New...** and select **Program**.
2. Under Basic, tap **Waypoint** to add a waypoint to the program tree. A default MoveJ is also added to the program tree.
3. Select the new waypoint and in the Command tab, tap **Waypoint**.
4. On the Move Tool screen, move the Robot arm by pressing the move arrows.
   You can also move the Robot arm by holding down the Freedrive button and pulling the Robot arm into desired positions.
5. Once the Robot arm is in position, press **OK** and the new waypoint displays as Waypoint_1.
6. Follow steps 2 to 5 to create Waypoint_2.
7. Select Waypoint_2 and press the Move Up arrow until it is above Waypoint_1 to change the order of the movements.
8. Stand clear, hold on to the emergency stop button and in the PolyScope Footer, press **Play** button for the Robot arm to move between Waypoint_1 and Waypoint_2.
   Congratulations! You have now produced your first robot program that moves the Robot arm between the two given waypoints.

WARNING:

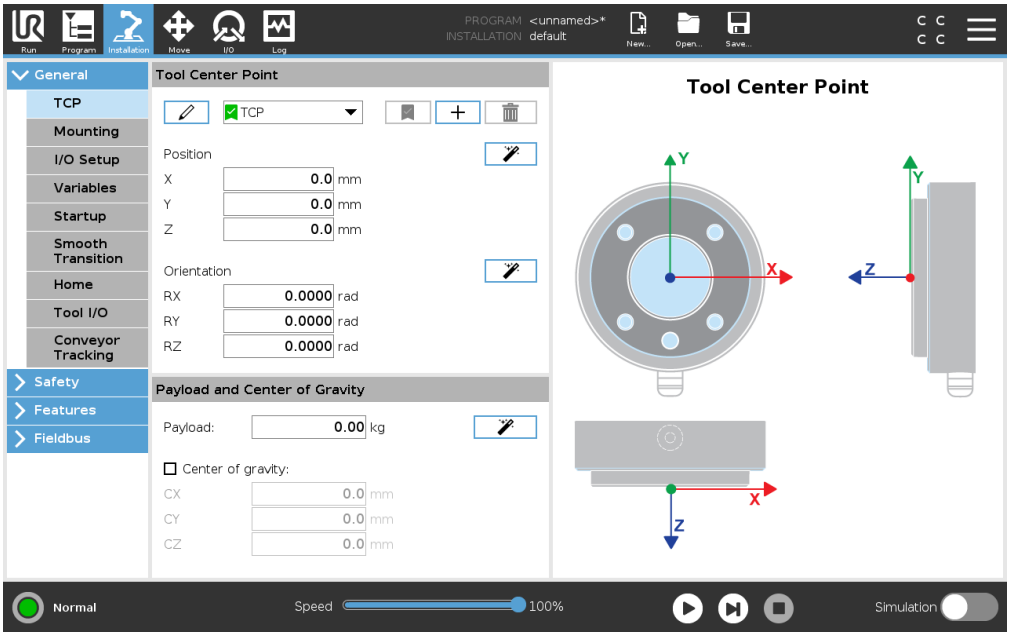
1. Do not drive the robot into itself or anything else as this may cause damage to the robot.
2. Keep your head and torso outside the reach (workspace) of the robot. Do not place fingers where they can be caught.
3. This is only a quick start guide to show how easy it is to use a UR robot. It assumes a harmless environment and a very careful user. Do not increase the speed or acceleration above the default values. Always conduct a risk assessment before placing the robot into operation.

16 Installation Tab

16.1 General

The Installation Tab allows you to configure the settings which affect the overall performance of the robot and PolyScope.

16.1.1 TCP Configuration



A **Tool Center Point** (TCP) is a point on the robot's tool. The TCP is defined and named in the Installation Tab **Setup for the Tool Center Point** screen (shown above). Each TCP contains a translation and a rotation relative to the center of the tool output flange.

When programmed to return to a previously stored waypoint, a robot moves the TCP to the position and orientation saved within the waypoint. When programmed for linear motion, the TCP moves linearly.

The `X`, `Y` and `Z` coordinates specify the TCP position, while the`RX`, `RY` and `RZ` coordinates specify its orientation. When all values are zero, the TCP coincides with the center point of the tool output flange and adopts the coordinate system depicted on the screen.

Adding, Renaming, Modifying and Removing TCPs

To define a new TCP, tap the **New** button. The created TCP automatically receives a unique name and becomes selectable in the drop-down menu. To rename a TCP, tap the **Pencil** button next to the **TCP** drop-down menu. To remove the selected TCP, tap the **Remove** button. The last TCP cannot be removed.

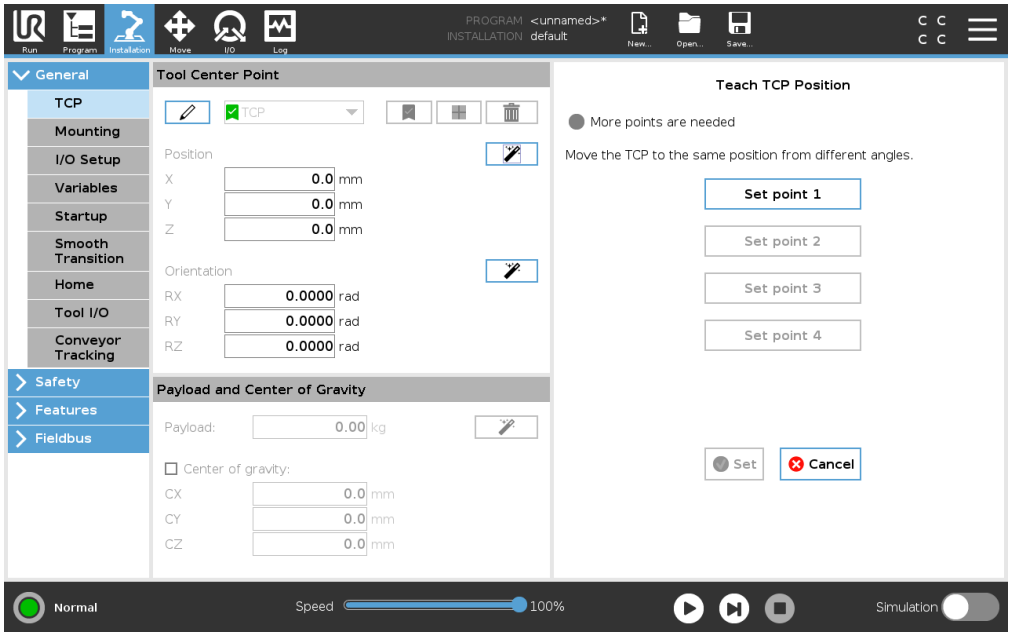
The translation and rotation of the selected TCP can be modified by tapping the respective white text fields and entering new values.

The default and the active TCP

There is one default configured TCP, marked by a green checkmark icon to the laft of its name in the **Available TCPs drop-down** menu. To set a TCP as the default, select the desired TCP and tap **Set as default**.

A TCP offset is designated as *active* to determine all linear motions in Cartesian coordinate system space. The motion of the active TCP is visualized on the Graphics Tab (see 15.3). Before a program runs, the default TCP is set as the active TCP. Within a program, any of the specified TCPs can be set as *active* for a particular movement of the robot (see 15.5.1 and 15.5.4).

Teaching TCP position

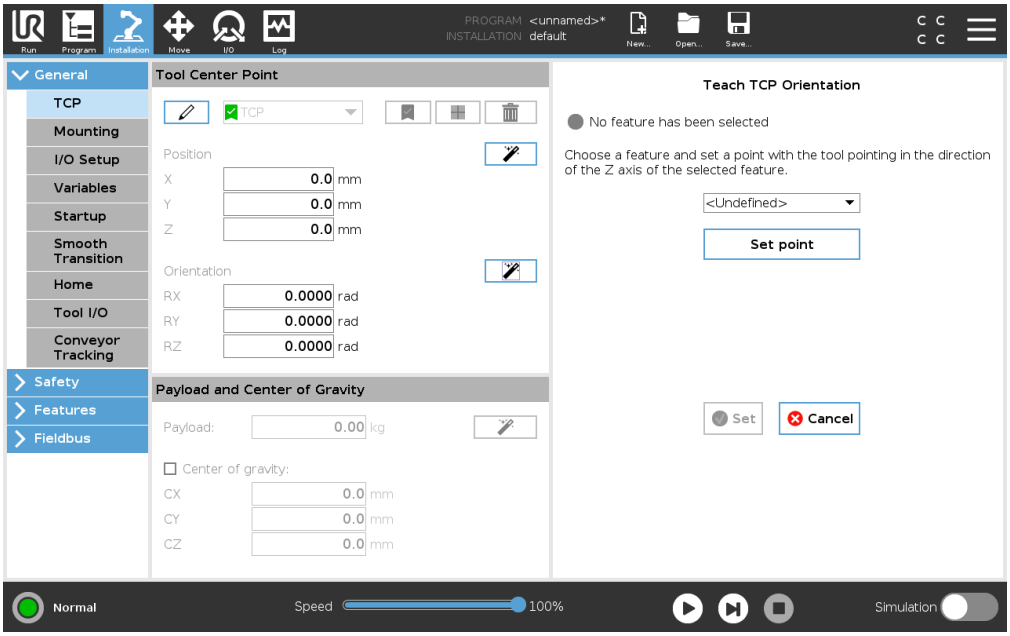


TCP position coordinates can be calculated automatically as follows:

1. Tap the **TCP Position Wizard**.
2. Choose a fixed point in the workspace of the robot.
3. Use the position arrows on the right side of the screen to move the TCP from at least three different angles and to save the corresponding positions of the tool output flange.
4. Use the **Set** button to apply the verified coordinates to the appropriate TCP. The positions must be sufficiently diverse for the calculation to work correctly. If they are not sufficiently diverse, the status LED above the buttons turns red.

Though three positions are sufficient to determine the TCP, a fourth position can be used to further verify the calculation is correct. The quality of each saved point, with respect to the calculated TCP, is indicated using a green, yellow, or red LED on the corresponding button.

Teaching TCP orientation



1. Tap the **TCP Orientation Wizard**.
2. Select a feature from the drop-down list. (See 16.3) for additional information on defining new features
3. Tap **Select point** and use **Move tool arrows** to a position where the tool's orientation and the corresponding TCP coincide with the selected features's coordinate system.
4. Verify the calculated TCP orientation and apply it to the selected TCP by tapping **Set**.

Payload

The weight of the robot's tool is specified in the lower part of the screen.To change this setting, simply tap the white text field and enter a new weight. The setting applies to all defined TCPs. For details about the maximum allowed payload, see the Hardware Installation Manual.

Payload Estimation

This feature allows the robot to help set the correct payload and Center of Gravity.

Using Payload Estimation Wizard

1. In the Installation Tab, under General, select **TCP**
2. On the TCP screen, under Payload and Center of Gravity, tap the **Payload and Center of Gravity Wizard**.
3. In the Payload Estimation Wizard tap **Next**
4. Follow the steps to set the four positions.
   Setting the four positions requires moving the robot arm into four different positions. Each position is measured. Individual measurements can be modified by tapping the center of gravity fields and entering values.

5. Once all measurements are complete, tap **Finish**

NOTE:
Follow the these guidelines for best Payload Estimation results:

- Ensure the four TCP positions are as different as possible from each other
- Perform the measurements within a short timespan

WARNING:

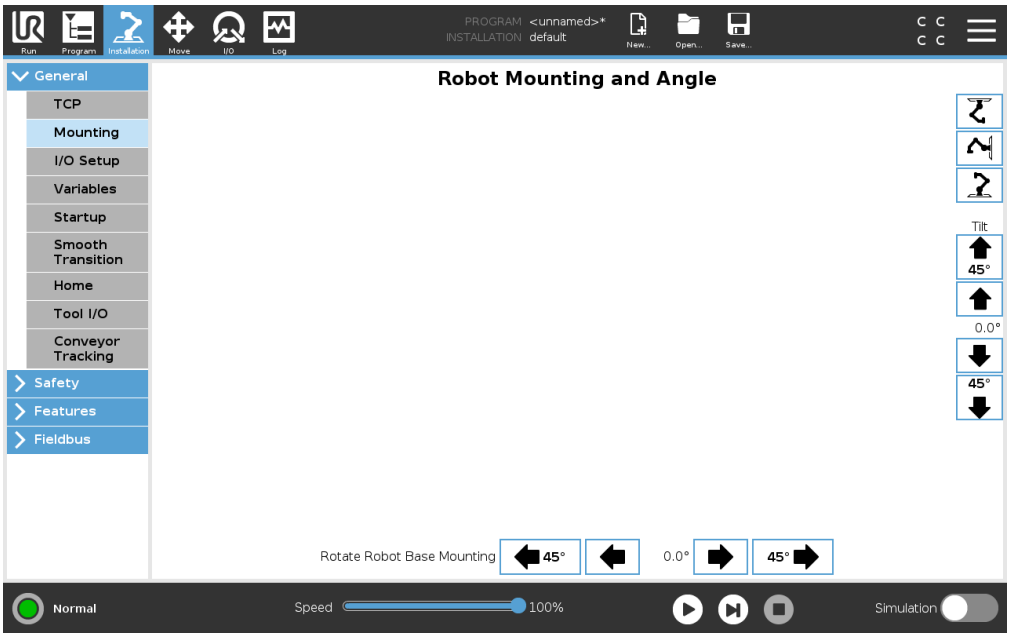
- Avoid pulling on the tool and/or attached payload before and during estimation
- Robot mounting and angle must be correctly defined in the installation

Center of gravity

The tool's center of gravity is specified using the fields CX, CY and CZ. The settings apply to all defined TCPs. Installations created before version 5.2 support the center of gravity being set to the TCP if they were previously set. If the center of gravity is manually set, in 5.2 or higher, the ability to set the center of gravity for the TCP is permanently removed.

WARNING:
Use the correct installation settings. Save and load the installation files with the program.

16.1.2 Mounting



Specifying the mounting of the Robot arm serves two purposes:

1. Making the Robot arm appear correctly on screen.
2. Telling the controller about the direction of gravity.

An advanced dynamics model gives the Robot arm smooth and precise motions, as well as allows the Robot arm to hold itself in **Freedrive Mode**. For this reason, it is important to mount the Robot arm correctly.

WARNING:
Failure to mount the Robot's arm correctly may result in frequent Protective Stops, and/or the Robot arm will move when pressing the **Freedrive** button.

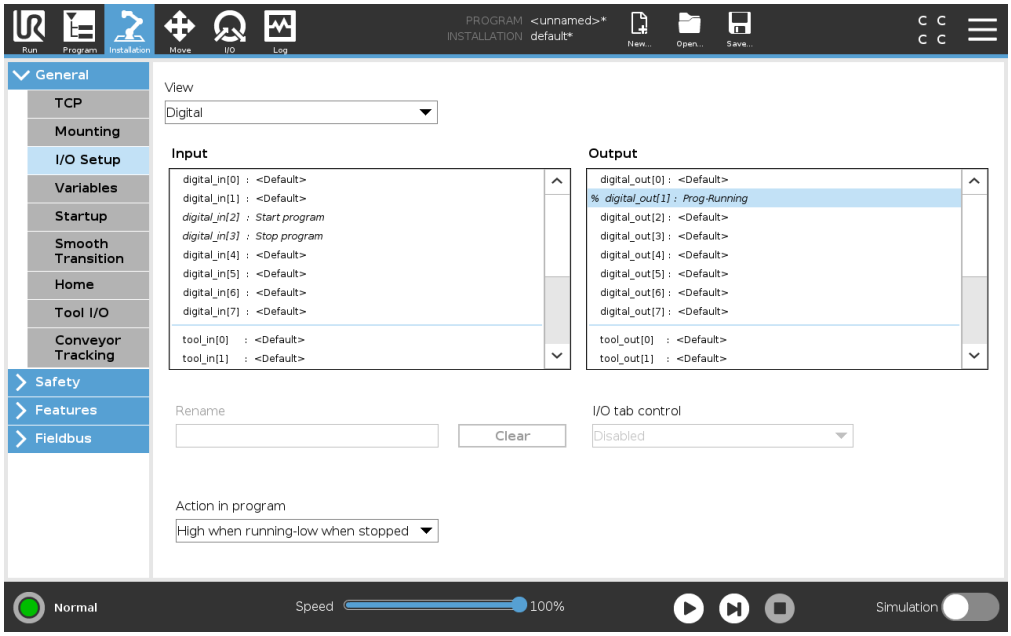
If the Robot arm is mounted on a flat table or floor, no change is needed on this screen. However, if the Robot arm is **ceiling mounted**, **wall mounted**, or **mounted at an angle**, this needs to be adjusted using the buttons.

The buttons on the right side of the screen are for setting the angle of the Robot arm's mounting. The top three right side buttons set the angle to **ceiling** ($180°$), **wall** ($90°$), **floor** ($0°$). The **Tilt** buttons set an arbitrary angle.

The buttons on the lower part of the screen are used to rotate the mounting of the Robot arm to match the actual mounting.

WARNING:
Use the correct installation settings. Save and load the installation files with the program.

16.1.3 I/O Setup



On the I/O Setup screen, users can define I/O signals and configure actions with the I/O tab control. Note: When the Tool Communication Interface (TCI) is enabled, the tool analog input becomes unavailable.

The **Input** and **Output** sections list types of I/O signals such as:

- Digital standard general purpose, configurable and tool
- Analog standard general purpose and tool
- MODBUS
- General purpose registers (boolean, integer and float) The general purpose registers can be accessed by a fieldbus (e.g., Profinet and EtherNet/IP).

I/O Signal Type

To limit the number of signals listed in the **Input** and **Output** sections, use the **View** drop-down menu at the top of the screen to change the displayed content based on signal type.

Assigning User-defined Names

To easily remember what the signals do while working with the robot, users can associate names to Input and Output signals.

1. Select the desired signal
2. Tap the text field in the lower part of the screen to set the name.
3. To reset the name to default, tap **Clear**.

A general purpose register must be given a user-defined name to make it available in the program (i.e., for a **Wait** command or the conditional expression of an **If** command) The **Wait** and **If** commands are described in (15.5.3) and (15.6.4), respectively. Named general purpose registers can be found in the **Input** or **Output** selector on the **Expression Editor** screen.

I/O Actions and I/O Tab Control

Input and Output Actions Physical and Fieldbus digital I/Os can be used to trigger actions or react to the status of a program.

Available Input Actions:

- Start: starts or resumes the current program on a rising edge. This function is only enabled in Remote Control (see 21.3.3).
- Stop: Stops the current program on a rising edge.
- Pause: Pauses the current program on a rising edge.
- Freedrive: When the input is high, the robot is in freedrive (similar to the freedrive button). The input is ignored if a program is running or other conditions disallow freedrive.

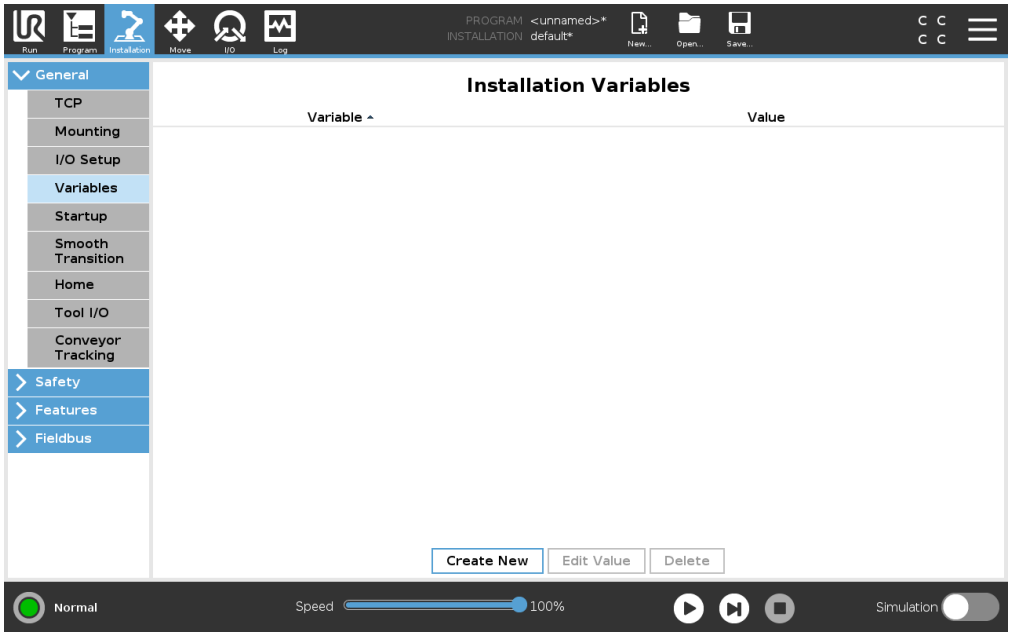
WARNING:
If the robot is stopped while using the Start input action, the robot slowly moves to the first waypoint of the program before executing that program. If the robot is paused while using the Start input action, the robot slowly moves to the position from where it was paused before resuming that program.

Available Output Actions:

- Low when not running: Output is low when the program state is “stopped” or “paused”.
- High when not running: Output is high when the program state is “stopped” or “paused”.
- High when not running, low when stopped: Output is low when the program state is “stopped” or “paused” and high when it is running.
- Continuous Pulse: Output alternates between high and low for a specified number of seconds, while the program is running. Pause or stop the program to maintain the pulse state.

I/O Tab Control Specify whether an output is controlled on the I/O tab (by either programmers, or both operators and programmers), or if it is controlled by the robot programs.

16.1.4 Variables



Variables created on the Variables screen are called Installation Variables and are used like normal program variables. Installation Variables are distinct because they keep their value even if a program stops and then starts again, and when the Robot arm and/or Control Box is powered down and powered up again. Their names and values are stored with the installation, therefore it is possible to use the same variable in multiple programs.

Pressing **Create New** brings up a panel with a suggested name for the new variable. The name may be changed and its value may be entered by touching either text field. The **OK**-button can only tapped if the new name is unused in this installation.

It is possible to change the value of an installation variable by highlighting the variable in the list and then clicking on **Edit Value**.

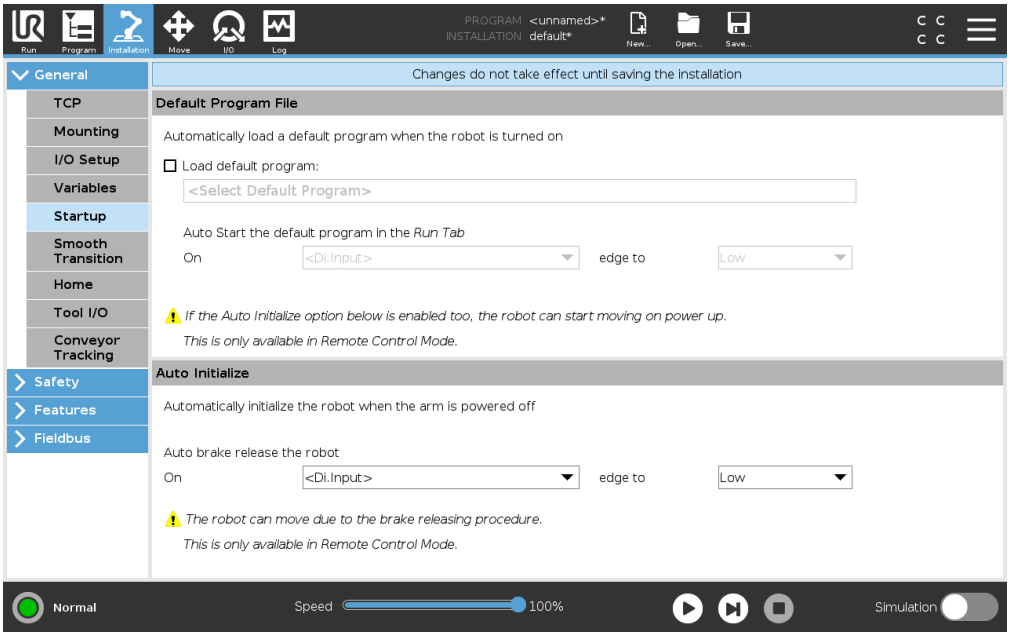
To delete a variable, select it and tap **Delete**.

After configuring the installation variables, the installation itself must be saved to keep the configuration.

The installation variables and their values are saved automatically every 10 minutes.

If a program or an installation is loaded and one or more of the program variables have the same name as the installation variables, the user is presented with options to either resolve the issue using the installation variables of the same name instead of the program variable or rsolve the issue by having the conflicting variables renamed automatically.

16.1.5 Startup



The Startup screen contains settings for automatically loading and starting a default program, and for auto-initializing the Robot arm during power up.

WARNING:

1. When autoload, auto start and auto initialize are enabled, the robot runs the program as soon as the Control Box is powered up as long as the input signal matches the selected signal level. For example, the edge transition to the selected signal level will not be required in this case.
2. Use caution when the signal level is set to LOW. Input signals are low by default, leading the program to automatically run without being triggered by an external signal.
3. You must be in **Remote Control** Mode before running a program where auto start and auto initialize are enabled.

Loading a Startup Program

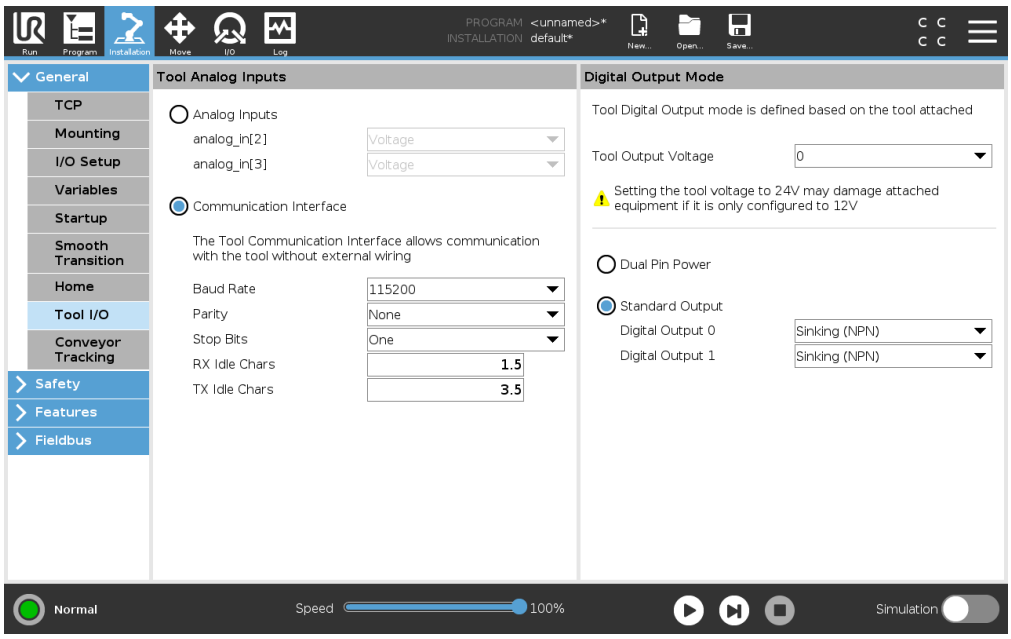
A default program is loaded after the Control Box is powered up. Furthermore, the default program is auto loaded when the **Run Program** screen `(see 14)` is entered and no program is loaded.

Starting a Startup Program

The default program is auto started in the **Run Program** screen. When the default program is loaded and the specified external input signal edge transition is detected, the program is started automatically.

On Startup, the current input signal level is undefined. Choosing a transition that matches the signal level on startup starts the program immediately. Furthermore, leaving the **Run Program** screen or tapping the Stop button in the Dashboard disables the auto start feature until the Run button is pressed again.

16.1.6 Tool I/O



Tool Analog Inputs

Tool Communication Interface

The Tool Communication Interface (TCI) enables the robot to communicate with an attached tool via the robot tool analog input. This removes the need for external cabling.

Once the Tool Communication Interface is enabled, all tool analog inputs are unavailable.

Configuring the Tool Communication Interface (TCI)

1. Tap the Installation tab and under General tap **Tool I/O**.
2. Select **Communication Interface** to edit TCI settings.
   Once the TCI is enabled, the tool analog input is unavailable for the I/O Setup of the Installation and does not appear in the input list. Tool analog input is also unavailable for programs as Wait For options and expressions.
3. In the drop-down menus under Communication Interface, select required values.
   Any changes in values are immediately sent to the tool. If any installation values differ from what the tool is using, a warning appears.

Digital Output Mode

The tool communication interface allows two digital outputs to be independently configured. In PolyScope, each pin has a drop-down menu that allows the output mode to be set. The following options are available:

- Sinking: This allows the pin to be configured in an NPN or Sinking configuration. When the output is off, the pin allows a current to flow to the ground. This can be used in conjunction with the PWR pin to create a full circuit (see **??**).
- Sourcing: This allows the pin to be configured in a PNP or Sourcing configuration. When the output is on, the pin provides a positive voltage source (configurable in the IO Tab). This can be used in conjunction with the GND pin to create a full circuit (see **??**).
- Push / Pull: This allows the pin to be configured in a Push / Pull configuration. When the output is on, the pin provides a positive voltage source (configurable in IO Tab). This can be used in conjunction with the GND pin to create a full circuit (see **??**). When the output is off, the pin allows a current to flow to the ground.

After selecting a new output configuration, the changes take effect. The currently loaded installation is modified to reflect the new configuration. After verifying the tool outputs are working as intended, make sure to save the installation to prevent losing changes.

Dual Pin Power

Dual Pin Power is used as a source of power for the tool. Enabling Dual Pin Power disables default tool digital outputs.

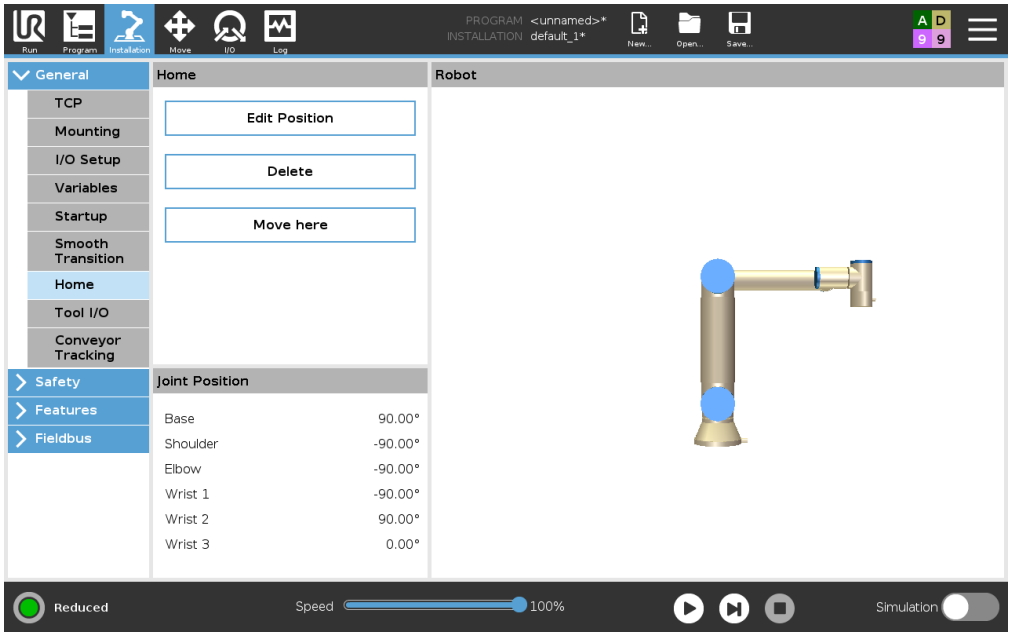
16.1.7 Smooth Transition Between Safety Modes

When switching between safety modes during events (i.e., Reduced Mode Input, Reduced Mode Trigger Planes, Safeguard Stop, and Three-Position Enabling Device), the Robot Arm aims to use 0.4s to create a “soft” transition. Existing applications have unchanged behavior which corresponds to the “hard” setting. New installation files default to the “soft” setting.

Adjusting Acceleration/Deceleration Settings

1. In the Header, tap **Installation**.
2. In the menu on the left, under **General**, select **Smooth Transition**.
3. Select **Hard** to have a higher acceleration/deceleration
   or select **Soft** for the smoother default transition setting.

16.1.8 Home



Home is a user-defined return position for the Robot Arm. Once defined, the Home Position is available when creating a robot program. You can use the Home Position to define a Safe Home Position.(See 13.2.10)

Defining Home

1. In the Header, tap **Installation**.
2. Under **General**, select **Home**.
3. Tap **Set Position**.
4. Teach robot using either **Freedrive** or **Transition buttons**.

16.1.9 Conveyor Tracking Setup

The Conveyor Tracking Setup allows the movement of up to two separate conveyors to be configured. The Conveyor Tracking Setup provides options for configuring the robot to work with absolute or incremental encoders, as well as linear or circular conveyors.

Defining a Conveyor

1. In the Header, tap Installation.
2. Under General, select **Conveyor Tracking**.
3. Under Conveyor Tracking Setup, in the dropdown list select **Conveyor 1** or **Conveyor 2**. You can only define one conveyor at a time.
4. Select **Enable Conveyor Tracking**
5. Configure **Conveyor Parameters** (section 16.1.9) and **Tracking Parameters** (section 16.1.9).

Conveyor Parameters

Incremental encoders can be connected to Digital Inputs 8 to 11. Decoding of digital signals runs at 40kHz. Using a **Quadrature** encoder (requiring two inputs), the robot can determine the speed and direction of the conveyor. If the direction of the conveyor is constant, a single input can be used to detect *Rising*, *Falling*, or *Rise and Fall* edges which determine conveyor speed.

Absolute encoders can be connected through a MODBUS signal. This requires a Digital MODBUS Output register preconfigured in (section 16.4.1).

Tracking Parameters

Linear Conveyors When a linear conveyor is selected, a line feature must be configured in the **Features** part of the installation to determine the direction of the conveyor. Ensure accuracy by placing the line feature parallel to the direction of the conveyor, with a large distance between the two points that define the line feature. Configure the line feature by placing the tool firmly against the side of the conveyor when teaching the two points. If the line feature's direction is opposite to the conveyor's movement, use the **Reverse direction** button.

The **Ticks per meter** field displays the number of ticks the encoder generates when the conveyor moves one meter.

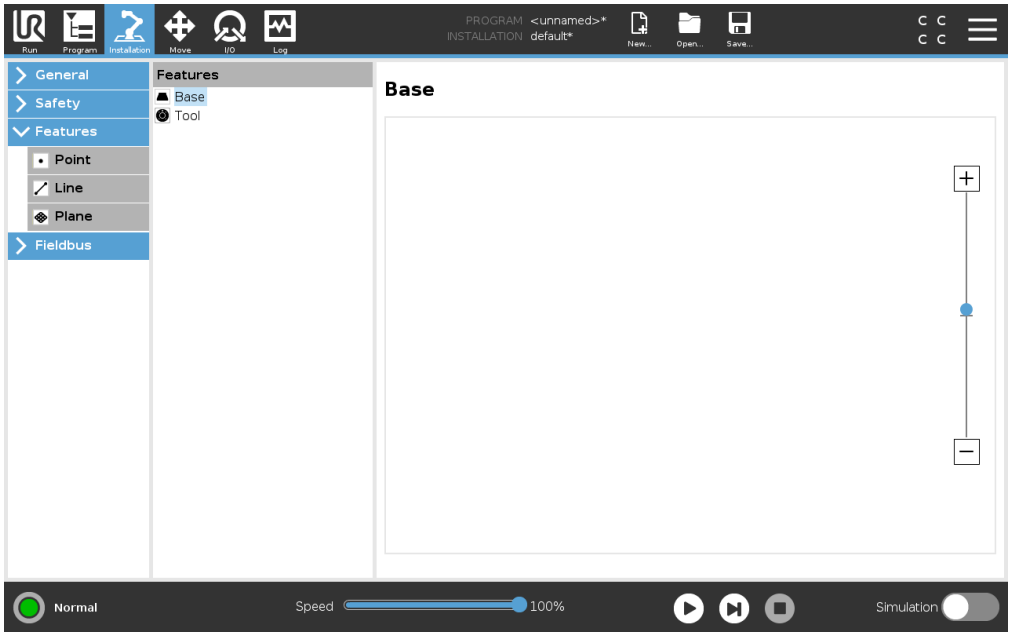
Circular Conveyors When tracking a circular conveyor, the conveyor center point must be defined.

1. Define the center point in the **Features** part of the installation. The value of **Ticks per revolution** must be the number of ticks the encoder generates when the conveyor rotates one full revolution.
2. Select the **Rotate tool with conveyor** checkbox for the tool orientation to track the conveyor rotation.

16.2 Safety

See chapter 13.

16.3 Features



The **Feature**, is a representation of an object that is defined with a name for future reference and a six dimensional pose (position and orientation) relative to the robot base.
Some subparts of a robot program consist of movements executed relative to specific objects other than the base of the Robot arm. These objects could be tables, other machines, workpieces, conveyors, pallets, vision systems, blanks, or boundaries which exist in the surroundings of the Robot arm. Two predefined features always exist for the robot. Each feature has its pose defined by the configuration of the Robot arm itself:

- The Base feature is located with origin in the centre of the robot base (see figure 16.1)
- The Tool feature is located with origin in the centre of the current TCP (see figure 16.2)

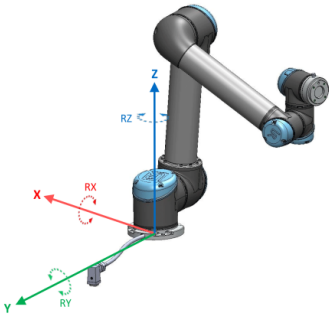


Figure 16.1: Base feature

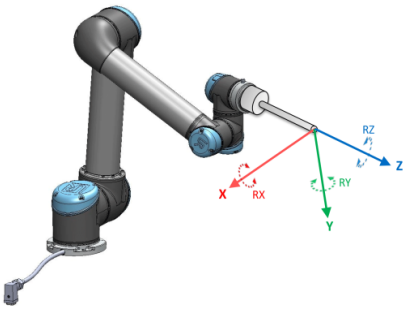


Figure 16.2: Tool (TCP) feature

User-defined features are positioned through a method that uses the current pose of the TCP in the work area. This means the users can teach feature locations using Freedrive Mode or ”jogging” to

move the robot to the desired pose.

Three different strategies exist (**Point**, **Line** and **Plane**) for defining a feature pose. The best strategy for a given application depends on the type of object being used and the precision requirements. In general a feature based on more input points (**Line** and **Plane**) is be preferred if applicable to the specific object.
To accurately define the direction of a linear conveyor, define two points of a Line feature with as much physical separation as possible. The Point feature can also be used to define a linear conveyor, however, the user must point the TCP in the direction of the conveyor movement.
Using more points to define the pose of a table means that the orientation is based on the positions rather than the orientation of a single TCP. A single TCP orientation is harder to configure with high precision.
To learn about the different methods to define a feature see `(sections: 16.3.2)`, `(16.3.3)` and `(16.3.4)`.

16.3.1 Using a feature

When a feature is defined in the installation, you can refer to it from the robot program to relate robot movements (e.g. **MoveJ**, **MoveL** and **MoveP** commands) to the feature `(see section 15.5.1)`. This allows for easy adaptation of a robot program (e.g., when there are multiple robot stations, when an object is moved during program runtime, or when an object is permanently moved in the scene). By adjusting the feature of an object, all program movements relative to the object is moved accordingly. For further examples, see `(sections 16.3.5)` and `(16.3.6)`.

Features configured as joggable are also useful tools when manually moving the robot in the Move Tab `(section 17)` or the **Pose Editor** screen `(see 17.3.1)`. When a feature is chosen as a reference, the Move Tool buttons for translation and rotation operate in the selected feature space `(see 17.3)` and `(17.1)`, reading of the TCP coordinates. For example, if a table is defined as a feature and is chosen as a reference in the Move Tab, the translation arrows (i.e., up/down, left/right, forward/backward) move the robot in these directions relative to the table. Additionally, the TCP coordinates will be in the frame of the table.

- In the Features tree you can rename a Point, Line or Plane by tapping the pencil button.
- In the Features tree you can delete a Point, Line or Plane by tapping the Delete button.

Joggable

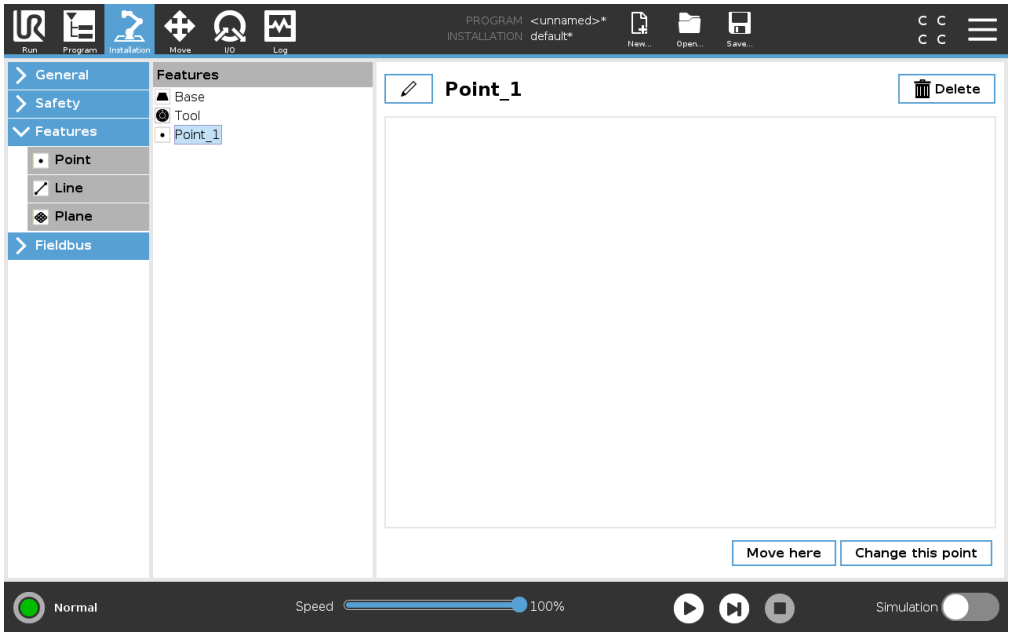
Choose whether the selected feature should be joggable. This determines whether the feature will appear in the feature menu on the Move screen.

Using Move robot here

Push the **Move robot here** button to move the Robot arm towards the selected feature. At the end of this movement, the coordinate systems of the feature and the TCP will coincide.

16.3.2 Adding a Point

Push the **Point**button to add a point feature to the installation. The point feature defines a safety boundary or a global home configuration of the Robot arm. The point feature pose is defined as the position and orientation of the TCP.



16.3.3 Adding a Line

Push the **Line** button to add a line feature to the installation. The line feature defines lines that the robot needs to follow. (e.g., when using conveyor tracking). A line *l* is defined as an axis between two point features *p1* and *p2* as shown in figure 16.3.

In figure 16.3 the axis directed from the first point towards the second point, constitutes the y-axis of the line coordinate system. The z-axis is defined by the projection of the z-axis of *p1* onto the plane perpendicular to the line. The position of the line coordinate system is the same as the position of *p1*.

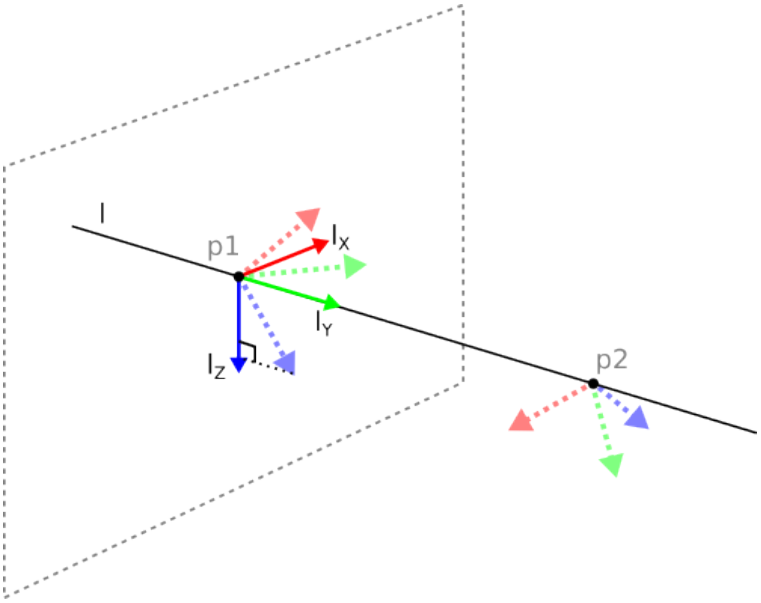
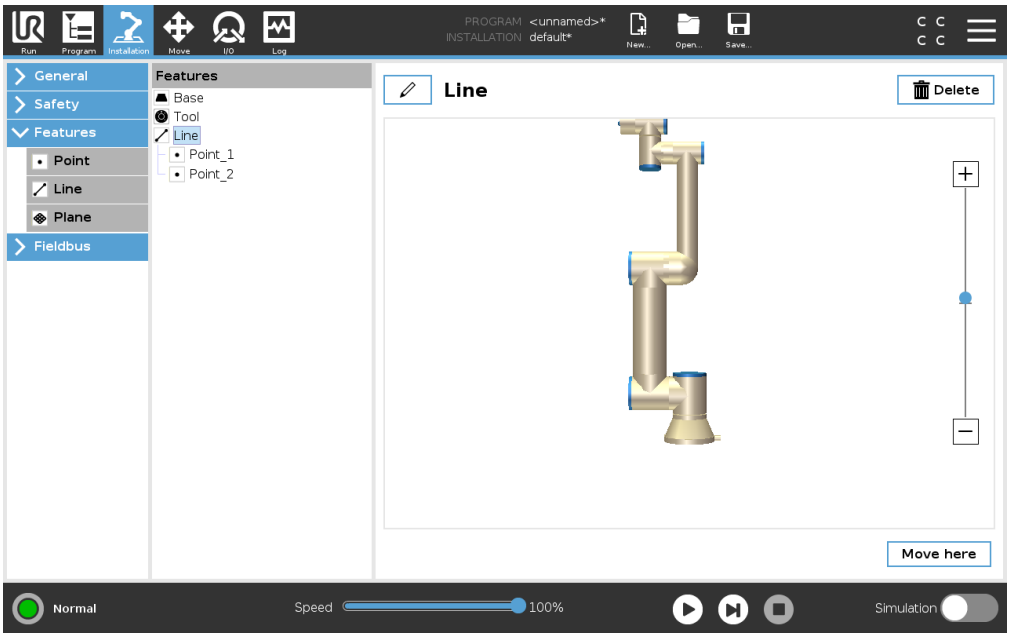


Figure 16.3: Definition of the line feature



16.3.4 Plane Feature

Select the plane feature when you need a frame with high precision: e.g., when working with a vision system or doing movements relative to a table.

Adding a plane

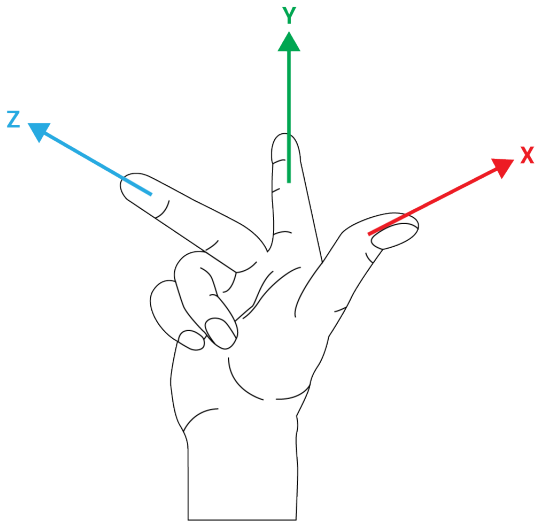
1. In Installation, select **Features**.
2. Under Features select **Plane**.

Teaching a plane

When you press the plane button to create a new plane, the on-screen guide assists you creating a plane.

1. Select Origo
2. Move robot to define the direction of the positive x-axis of the plane
3. Move robot to define the direction of the positive y-axis of the plane

The plane is defined using the right hand rule so the z- axis is the cross product of the x-axis and the y-axis, as illustrated below.



NOTE:
You can re-teach the plane in the opposite direction of the x-axis, if you want that plane to be normal in the opposite direction.

Modify an existing plane by selecting Plane and pressing Modify Plane. You will then use the same guide as for teaching a new plane.

16.3.5 Example: Manually Updating a Feature to Adjust a Program

Consider an application where multiple parts of a robot program is relative to a table. Figure 16.4 illustrates the movement from waypoints `wp1` through `wp4`.

The application requires the program to be reused for multiple robot installations where the position of the table varies slightly. The movement relative to the table is identical. By defining the table position as a feature *P1* in the installation, the program with a *MoveL* command configured relative to the plane can be easily applied on additional robots by just updating the installation with the actual position of the table.

```
Robot Program
    MoveJ
        S1
    MoveL # Feature: P1_var
        wp1
        wp2
        wp3
        wp4
```

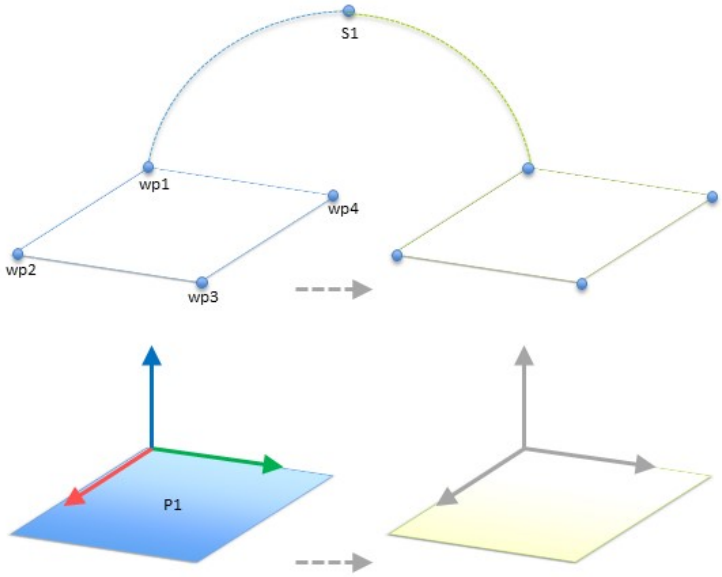


Figure 16.4: Simple program with four waypoints relative to a feature plane manually updated by changing the feature

The concept applies to a number of Features in an application to achieve a flexible program can solve the same task on many robots even though if other places in the work space varies between installations.

16.3.6 Example: Dynamically Updating a Feature Pose

Consider a similar application where the robot must move in a specific pattern on top of a table to solve a particular task (see 16.5).

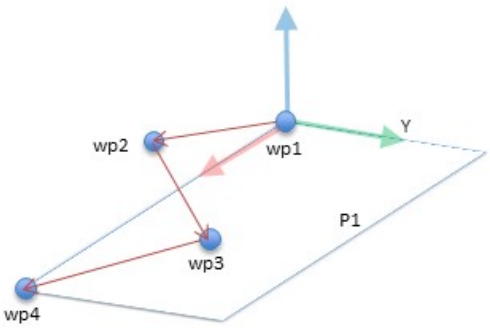


Figure 16.5: A *MoveL* command with four waypoints relative to a plane feature

The movement relative to *P1* is repeated a number of times, each time by an offset *o*. In this example the offset is set to 10 cm in the Y-direction (see figure 16.6, offsets *O1* and *O2*). This is achieved using *pose_add()* or *pose_trans()* script functions to manipulate the variable.

It is possible to switch to a different feature while the program is running instead of adding an offset. This is shown in the example below (see figure 16.7) where the reference feature for the

```
Robot Program
    MoveJ
        wp1
    y = 0.01
    o = p[0,y,0,0,0,0]
    P1_var = pose_trans(P1_var, o)
    MoveL # Feature: P1_var
        wp1
        wp2
        wp3
        wp4
```

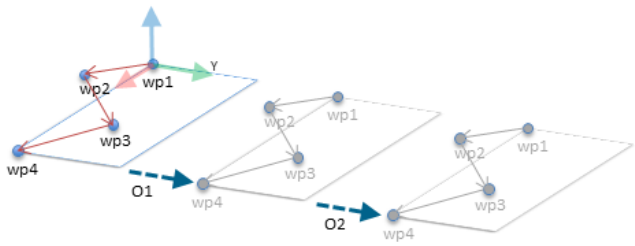


Figure 16.6: Applying an offset to the plane feature

MoveL command *P1_var* can switch between two planes *P1* and *P2*.

```
Robot Program
    MoveJ
        S1
    if (digital_input[0]) then
        P1_var = P1
    else
        P1_var = P2
    MoveL # Feature: P1_var
        wp1
        wp2
        wp3
        wp4
```

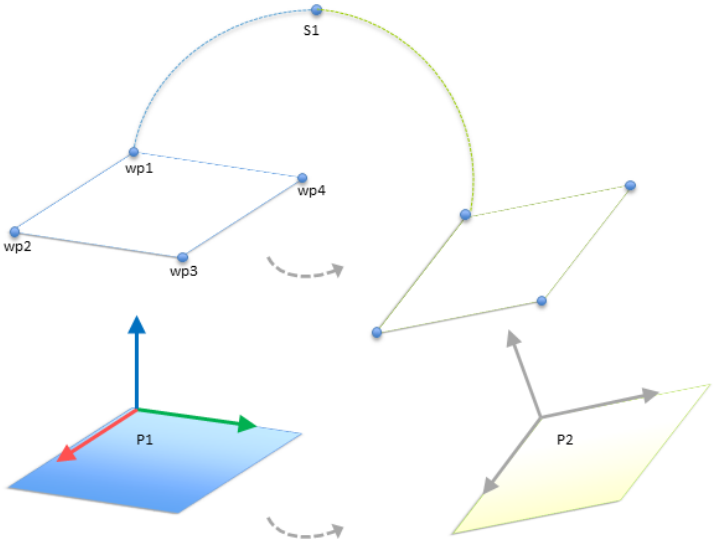
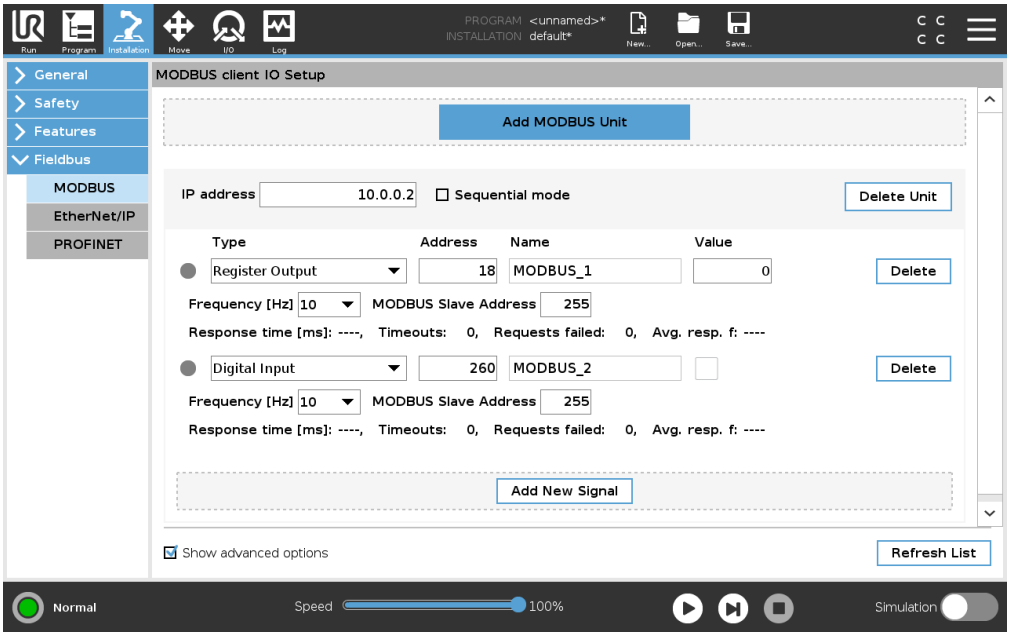


Figure 16.7: Switching from one plane feature to another

16.4 Fieldbus

Here you can set the family of industrial computer network protocols used for real-time distributed control accepted by PolyScope: MODBUS and Ethernet/IP

16.4.1 MODBUS client I/O Setup



Here, the MODBUS client (master) signals can be set up. Connections to MODBUS servers (or slaves) on specified IP addresses can be created with input/output signals (registers or digital). Each signal has a unique name so it can be used in programs.

Refresh

Push this button to refresh all MODBUS connections. Refreshing disconnects all modbus units, and connects them back again. All statistics are cleared.

Add unit

Push this button to add a new MODBUS unit.

Delete unit

Push this button to delete the MODBUS unit and all signals on that unit.

Set unit IP

Here the IP address of the MODBUS unit is shown. Press the button to change it.

Sequential mode

Available only when Show Advanced Options (see 16.4.1) is selected. Selecting this checkbox forces the modbus client to wait for a response before sending the next request. This mode is required by some fieldbus units. Turning this option on may help when there are multiple signals, and increasing request frequency results in signal disconnects. Note that the actual signal frequency may be lower than requested when multiple signals are defined in sequential mode. Actual signal frequency can be observed in signal statistics (see section 16.4.1). The signal indicator will turn yellow if the actual signal frequency is less than half of the value selected from the ”Frequency” drop-down list.

Add signal

Push this button to add a signal to the corresponding MODBUS unit.

Delete signal

Push this button to delete a MODBUS signal from the corresponding MODBUS unit.

Set signal type

Use this drop down menu to choose the signal type. Available types are:

Digital input A digital input (coil) is a one-bit quantity which is read from the MODBUS unit on the coil specified in the address field of the signal. Function code 0x02 (Read Discrete Inputs) is used.

Digital output A digital output (coil) is a one-bit quantity which can be set to either high or low. Before the value of this output has been set by the user, the value is read from the remote MODBUS unit. This means that function code 0x01 (Read Coils) is used. When the output has been set by a robot program or by pressing the **set signal value** button, the function code 0x05 (Write Single Coil) is used onwards.

Register input A register input is a 16-bit quantity read from the address specified in the address field. The function code 0x04 (Read Input Registers) is used.

Register output A register output is a 16-bit quantity which can be set by the user. Before the value of the register has been set, the value of it is read from the remote MODBUS unit. This means that function code 0x03 (Read Holding Registers) is used. When the output has been set by a robot program or by specifying a signal value in the **set signal value** field, function code 0x06 (Write Single Register) is used to set the value on the remote MODBUS unit.

Set signal address

This field shows the address on the remote MODBUS server. Use the on-screen keypad to choose a different address. Valid addresses depends on the manufacturer and configuration of the remote MODBUS unit.

Set signal name

Using the on-screen keyboard, the user can give the signal a name. This name is used when the signal is used in programs.

Signal value

Here, the current value of the signal is shown. For register signals, the value is expressed as an unsigned integer. For output signals, the desired signal value can be set using the button. Again, for a register output, the value to write to the unit must be supplied as an unsigned integer.

Signal connectivity status

This icon shows whether the signal can be properly read/written (green), or if the unit responds unexpected or is not reachable (gray). If a MODBUS exception response is received, the response code is displayed. The MODBUS-TCP Exception responses are:

E1 ILLEGAL FUNCTION (0x01) The function code received in the query is not an allowable action for the server (or slave).

E2 ILLEGAL DATA ADDRESS (0x02) The function code received in the query is not an allowable action for the server (or slave), check that the entered signal address corresponds to the setup of the remote MODBUS server.

E3 ILLEGAL DATA VALUE (0x03) A value contained in the query data field is not an allowable value for server (or slave), check that the enterd signal value is valid for the specified address on the remote MODBUS server.

E4 SLAVE DEVICE FAILURE (0x04) An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.

E5 ACKNOWLEDGE (0x05) Specialized use in conjunction with programming commands sent to the remote MODBUS unit.

E6 SLAVE DEVICE BUSY (0x06) Specialized use in conjunction with programming commands sent to the remote MODBUS unit, the slave (server) is not able to respond now.

Show Advanced Options

This check box shows/hides the advanced options for each signal.

Advanced Options

Update Frequency This menu can be used to change the update frequency of the signal. This means the frequency with which requests are sent to the remote MODBUS unit for either reading or writing the signal value. When the frequency is set to 0, then modbus requests are initiated on demand using a *modbus_get_signal_status*, *modbus_set_output_register*, and *modbus_set_output_signal* script functions.

Slave Address This text field can be used to set a specific slave address for the requests corresponding to a specific signal. The value must be in the range 0-255 both included, and the default is 255. If you change this value, it is recommended to consult the manual of the remote MODBUS device to verify its functionality when changing slave address.

Reconnect count Number of times TCP connection was closed, and connected again.

Connection status TCP connection status.

Response time [ms] Time between modbus request sent, and response received - this is updated only when communication is active.

Modbus packet errors Number of received packets that contained errors (i.e. invalid lenght, missing data, TCP socket error).

Timeouts Number of modbus requests that didn't get response.

Requests failed Number of packets that could not be sent due to invalid socket status.

Actual freq. The average frequency of client (master) signal status updates. This value is recalculated each time the signal receives a response from the server (or slave).

All counters count up to 65535, and then wrap back to 0.

16.4.2 Ethernet/IP

EtherNet/IP is where you can enable or disable the connection of the robot to a EtherNet/IP. If Enable, you can select which action should occur to a program when there is a loss of EtherNet/IP scanner connection. Those actions are:

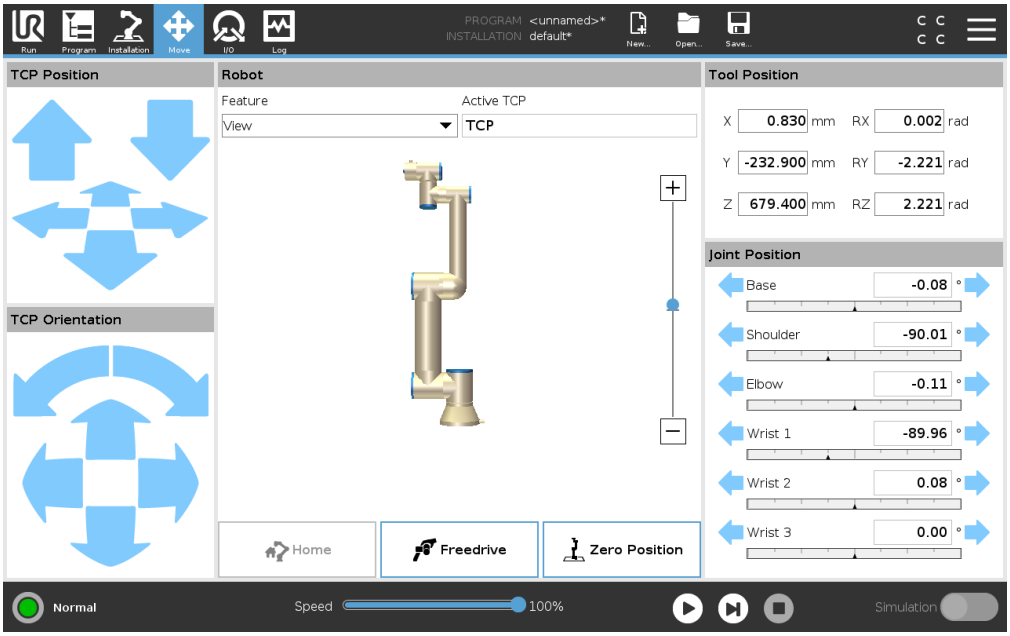
None: PolyScope will ignore the loss of EtherNet/IP connection and continue normally with the program.

Pause: PolyScope will pause the current program. Program will resume where it stopped.

Stop: PolyScope will stop the current program.

17 Move Tab

On this screen, you can move (jog) the robot arm directly, either by translating/rotating the robot tool, or by moving robot joints individually.



17.1 Move Tool

Hold down any of the **Move Tool** arrows to move the robot arm in a particular direction.

- The **Translate arrows** (upper) move the robot tool-tip in the direction indicated.
- The **Rotate arrows** (lower) change the orientation of the robot tool in the indicated direction. The rotation point is the Tool Center Point (TCP), i.e. the point at the end of the robot arm that gives a characteristic point on the robot's tool. The TCP is shown as a small blue ball.

17.2 Robot

If the current position of the robot TCP comes close to a safety or trigger plane, or the orientation of robot tool is near the tool orientation boundary limit `(see 13.2.5)`, a 3D representation of the proximate boundary limit is shown.
Note: when the robot is running a program, the visualization of boundary limits is disabled.

Safety planes are visualized in yellow and black with a small arrow representing the plane normal, which indicates the side of the plane on which the robot TCP is allowed to be positioned. Trigger

planes are displayed in blue and green and a small arrow pointing to the side of the plane, where the **Normal** mode limits (see 13.2.2) are active. The tool orientation boundary limit is visualized with a spherical cone together with a vector indicating the current orientation of the robot tool. The inside of the cone represents the allowed area for the tool orientation (vector).

When the robot TCP is no longer in proximity of the limit, the 3D representation disappears. If the TCP is in violation or very close to violating a boundary limit, the visualization of the limit turns red.

Feature

In the top left corner of the **Robot** field, under **Feature**, you can define how to control the robot arm relative to **View**, **Base** or **Tool** features.
Note: For the best feel for controlling the robot arm you can select the **View** feature, then use **Rotate arrows** to change the viewing angle of the 3D image to match your view of the real robot arm.

Active TCP

In the to right corner of the **Robot** field, under **Active TCP**, the name of the current active Tool Center Point (TCP) is displayed.

Home

The **Home** button accesses the **Move Robot into Position** screen, where you can hold down the **Auto** button (see 14.4) to move robot into position previously defined under Installation (see 16.1.8).

Freedrive

The on-screen **Freedrive** button allows the Robot Arm to be pulled into desired positions/poses.

Zero Position

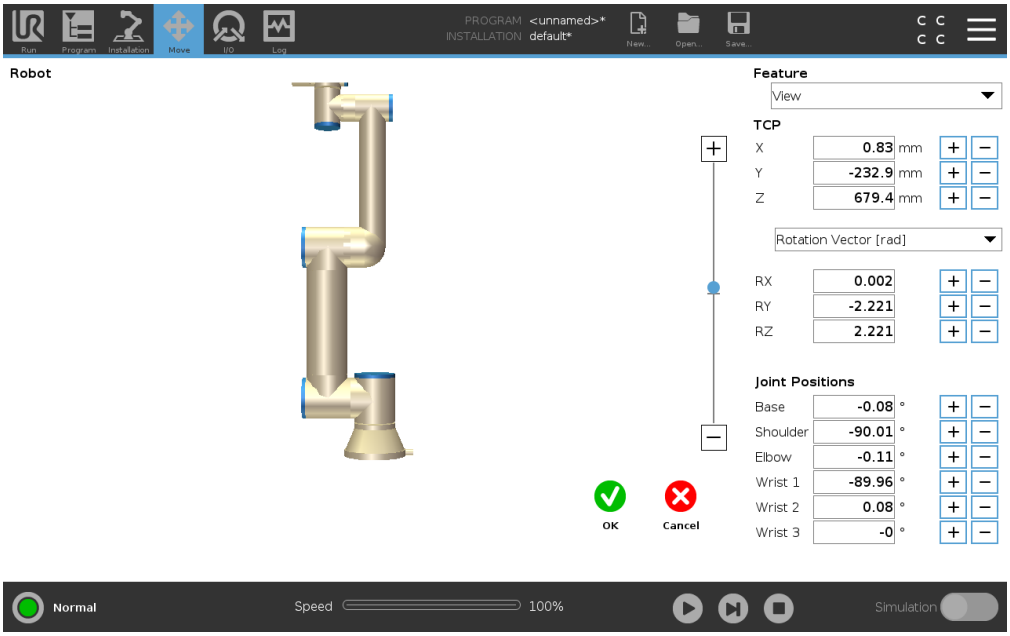
The **Zero Position** button allows the Robot Arm to return to an upright position.

17.3 Tool Position

The text boxes display the full coordinate values of the TCP relative to the selected feature.
Note: You can configure several named TCPs (see 16.1.1). You can also tap **Edit pose** to access the **Pose Editor** screen.

17.3.1 Pose Editor Screen

On this screen you can specify target joint positions, or a target pose (position and orientation) of the robot tool. This screen is “offline” and does not control the robot arm directly.



Robot

The current position of the robot arm and the specified new target position are shown in 3D graphics. The 3D drawing of the robot arm shows the current position of the robot arm, and the “shadow” of the robot arm shows the target position of the robot arm controlled by the specified values on the right hand side of the screen. Push the magnifying glass icons to zoom in/out or drag a finger across to change the view.

If the specified target position of the robot TCP is close to a safety or trigger plane, or the orientation of robot tool is near the tool orientation boundary limit (see 13.2.5), a 3D representation of the proximate boundary limit is shown.

Safety planes are visualized in yellow and black with a small arrow representing the plane normal, which indicates the side of the plane on which the robot TCP is allowed to be positioned. Trigger planes are displayed in blue and green and a small arrow pointing to the side of the plane, where the *Normal* mode limits (see 13.2.2) are active. The tool orientation boundary limit is visualized with a spherical cone together with a vector indicating the current orientation of the robot tool. The inside of the cone represents the allowed area for the tool orientation (vector).

When the target robot TCP no longer is in the proximity of the limit, the 3D representation disappears. If the target TCP is in violation or very close to violating a boundary limit, the visualization of the limit turns red.

Feature and tool position

In the top right corner of the screen, the feature selector can be found. The feature selector defines which feature to control the robot arm relative to

Below the feature selector, the name of the currently active Tool Center Point (TCP) is displayed. For further information about configuring several named TCPs, see 16.1.1. The text boxes show the

full coordinate values of that TCP relative to the selected feature. X, Y and Z control the position of the tool, while RX, RY and RZ control the orientation of the tool.

Use the drop down menu above the RX, RY and RZ boxes to choose the orientation representation. Available types are:

- **Rotation Vector [rad]** The orientation is given as a *rotation vector*. The length of the axis is the angle to be rotated in radians, and the vector itself gives the axis about which to rotate. This is the default setting.
- **Rotation Vector [°]** The orientation is given as a *rotation vector*, where the length of the vector is the angle to be rotated in degrees.
- **RPY [rad]** *Roll*, *pitch* and *yaw* (*RPY*) angles, where the angles are in radians. The RPY-rotation matrix (X, Y', Z" rotation) is given by:

$$R_{rpy}(\gamma, \beta, \alpha) = R_Z(\alpha) \cdot R_Y(\beta) \cdot R_X(\gamma)$$

- **RPY [°]** *Roll*, *pitch* and *yaw* (*RPY*) angles, where angles are in degrees.

Values can be edited by clicking on the coordinate. Clicking on the + or − buttons just to the right of a box allows you to add or subtract an amount to/from the current value. Pressing and holding down a button will directly increase/decrease the value. The longer the button is down, the larger the increase/decrease will be.

Joint positions

Allows the individual joint positions to be specified directly. Each joint position can have a value in the range from $-360°$ to $+360°$, which are the *joint limits*. Values can be edited by clicking on the joint position. Clicking on the + or − buttons just to the right of a box allows you to add or subtract an amount to/from the current value. Pressing and holding down a button will directly increase/decrease the value. The longer the button is down, the larger the increase/decrease will be.

OK button

If this screen was activated from the Move tab (see 17), clicking the OK button will return to the Move tab, where the robot arm will move to the specified target. If the last specified value was a tool coordinate, the robot arm will move to the target position using the *MoveL* movement type, while the robot arm will move to the target position using the *MoveJ* movement type, if a joint position was specified last. The different movement types are described in 15.5.1.

Cancel button

Clicking the Cancel button leaves the screen discarding all changes.

17.4 Joint Position

The **Joint Position** field allows you to directly control individual joints. Each joint moves along a default joint limit range from $-360°$ to $+360°$, defined by a horizontal bar. Once the limit is reached

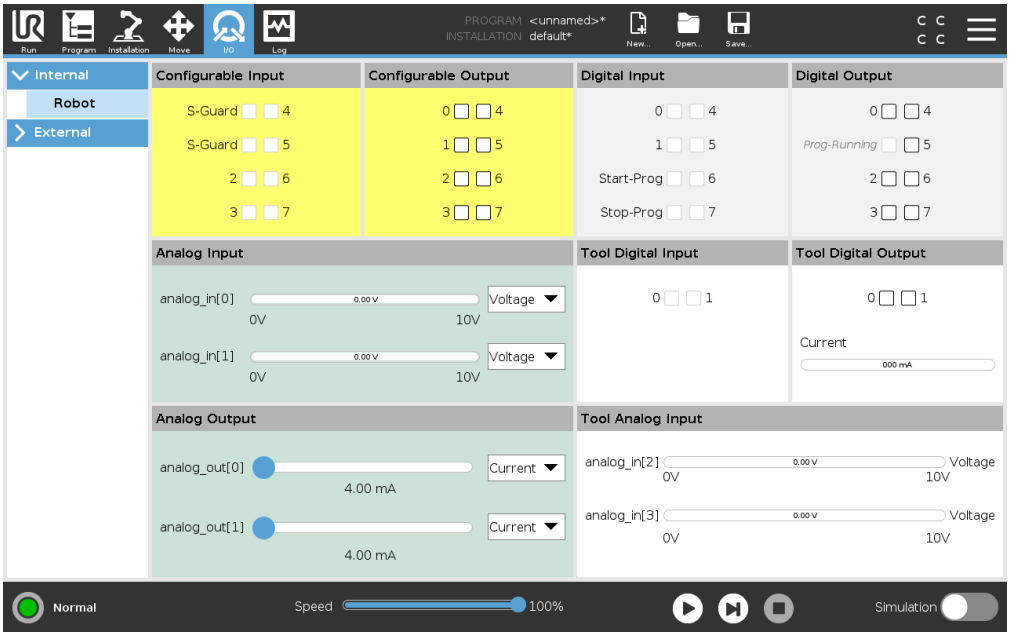
you cannot move a joint any further.
Note: You can configure joints with a position range different from the default (see 13.2.4), this new range is indicated with red zone inside the horizontal bar.

WARNING:

1. In the **Setup** tab, if the gravity setting (see 16.1.2) is wrong, or the robot arm carries a heavy load, the robot arm can start moving (falling) when you press the **Freedrive** tab. In that case, release **Freedrive**again.
2. Make sure to use the correct installation settings (e.g. Robot mounting angle, weight in TCP, TCP offset). Save and load the installation files along with the program.
3. Make sure that the TCP settings and the robot mounting settings are set correctly before operating **Freedrive** tab. If these settings are not correct, the robot arm will move when you activate **Freedrive**.
4. The **Freedrive** function (Impedance/Backdrive) must only be used in installations where the risk assessment allows it. Tools and obstacles must not have sharp edges or pinch points. Make sure that all personnel remain outside the reach of the robot arm.

18 I/O Tab

18.1 Robot



On this screen you can always monitor and set the live I/O signals from/to the robot control box. The screen displays the current state of the I/O, including during program execution. If anything is changed during program execution, the program will stop. At program stop, all output signals will retain their states. The screen is updated at only 10Hz, so a very fast signal might not display properly.

Configurable I/O's can be reserved for special safety settings defined in the safety I/O configuration section of the installation (see 13.2.8); those which are reserved will have the name of the safety function in place of the default or user defined name. Configurable outputs that are reserved for safety settings are not togglable and will be displayed as LED's only.

The electrical details of the signals are described in chapter **??**.

Analog Domain Settings The analog I/O's can be set to either current [4-20mA] or voltage [0-10V] output. The settings will be remembered for eventual later restarts of the robot controller when a program is saved.

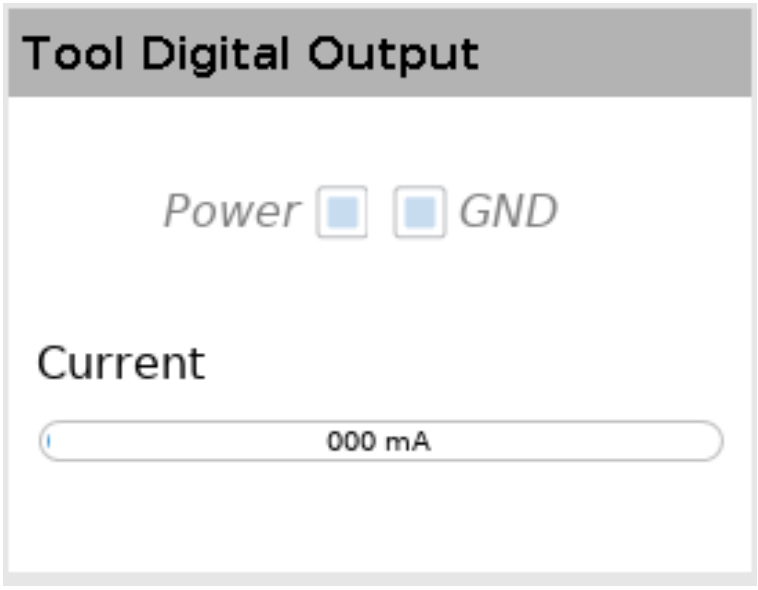
Tool Communication Interface When the **Tool Communication Interface TCI** is enabled, the tool analog input becomes unavailable. On the **I/O** screen, the **Tool Input** field changes as illustrated below.

Tool Analog Input	
Baud Rate	115200
Parity	None
Stop Bits	One
RX Idle Chars	1.50
TX Idle Chars	3.50

NOTE:
When the **Powered Dual Pin** is enabled, the tool digital outputs must be named as follows:

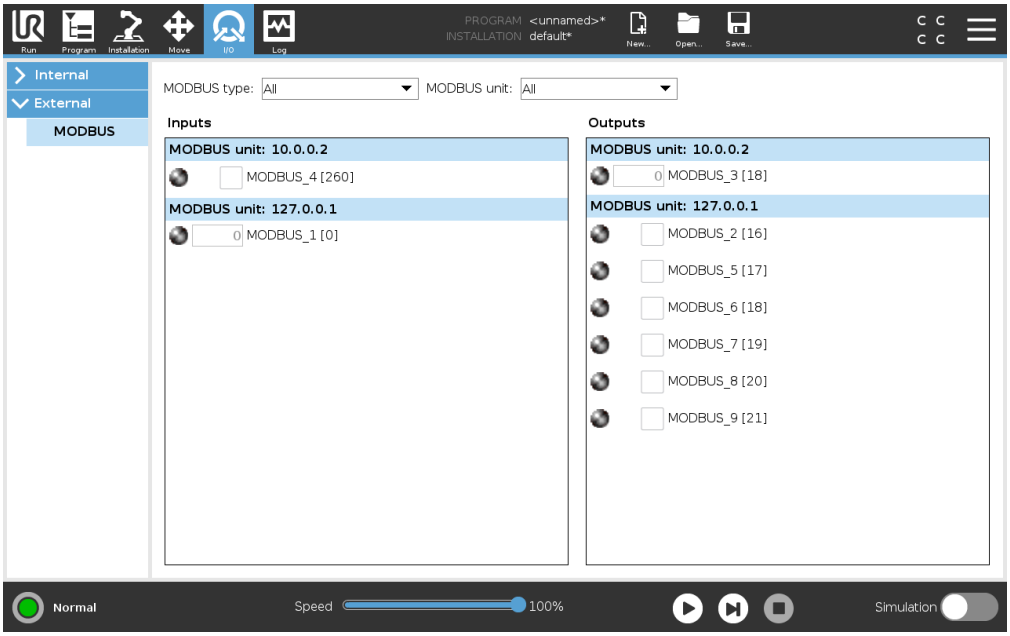
- tool_out[0] (Power)
- tool_out[1] (GND)

The **Tool Output** field is illustrated below.

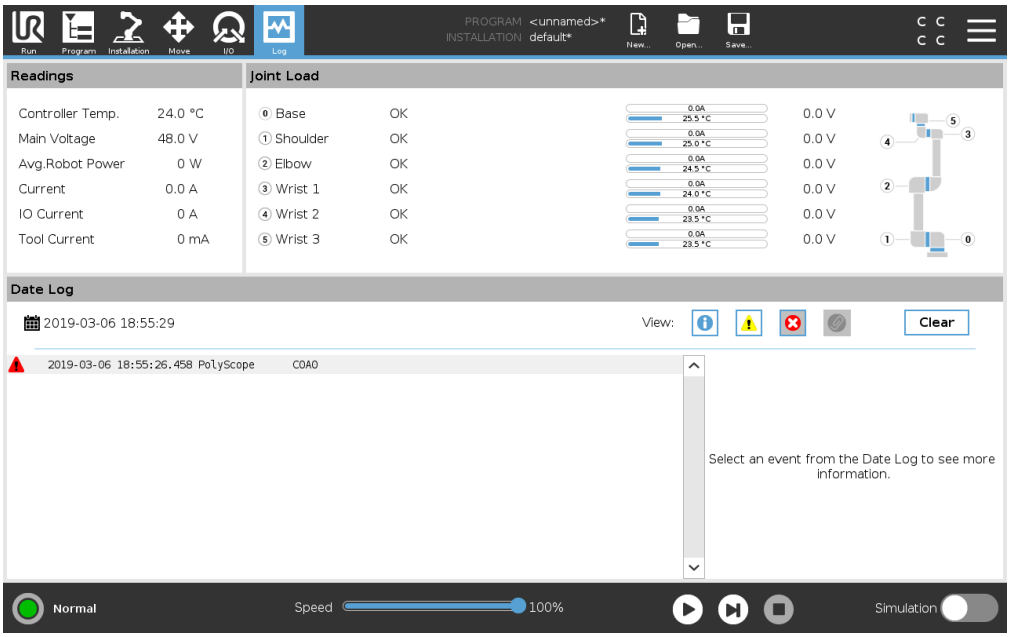


18.2 MODBUS

The screenshot below displays the MODBUS client I/O signals as they are set up in the installation. Using the drop-down menus at the top of the screen, you can change the displayed content based on signal type and MODBUS unit if more than one is configured. Each signal in the lists contains its connections status, value, name, and signal address. The output signals can be toggled if the connection status and the choice for I/O tab control allows it (see 16.1.3).



19 Log Tab



19.1 Readings and Joint Load

The top half of the screen displays the health of the Robot Arm and Control Box.
The left side of the screen shows information related to the Control Box, while the right side of the screen displays robot joint information. Each joint displays the temperature of the load of the joint, and the voltage.

19.2 Date log

The first column categorizes the severity of the log entry. The second column shows the messages' time of arrival. The next column shows the sender of the message. The last column shows the message itself. Messages can be filtered by selecting the toggle buttons which correspond to the severity. The figure above shows errors will be displayed while information and warning messages will be filtered. Some log messages are designed to provide more information that is displayed on the right side, after selecting the log entry.

19.3 Saving Error Reports

A detailed status report is available when a paper clip icon appears on the log line.

- Select a log line and tap the Save Report button to save the report to a USB drive.

- The report can be saved while a program is running.

NOTE:
The oldest report is deleted when a new one is generated. Only the five most recent reports are stored.

The following list of errors can be tracked and exported:

- Fault
- Internal PolyScope exceptions
- Protective Stop
- Unhandled exception in URCap
- Violation

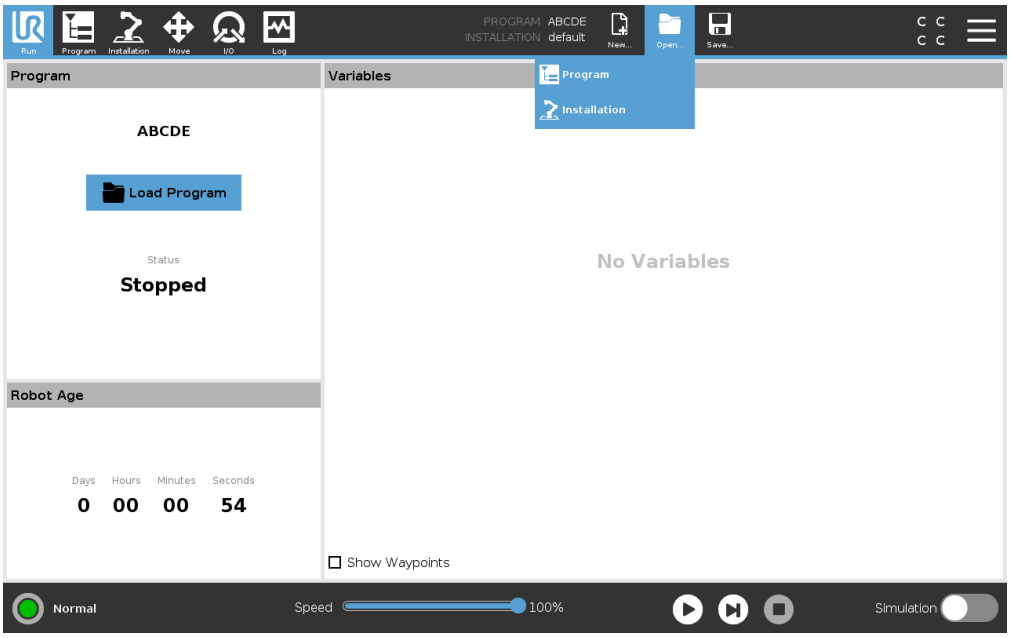
The exported report contains: a user program, a history log, an installation and a list of running services.

20 Program and Installation Manager

The Program and Installation Manager refers to three icons that allow you to create, load and configure Programs and Installations: **New...**, **Open...** and **Save...**. The File Path displays your current loaded Program name and the type of Installation. File Path changes when you create or load a new Program or Installation. You can have several installation files for a robot. Programs created load and use the active installation automatically.

20.1 Open...

Allows you to load a Program and/or Installation.



Opening a Program

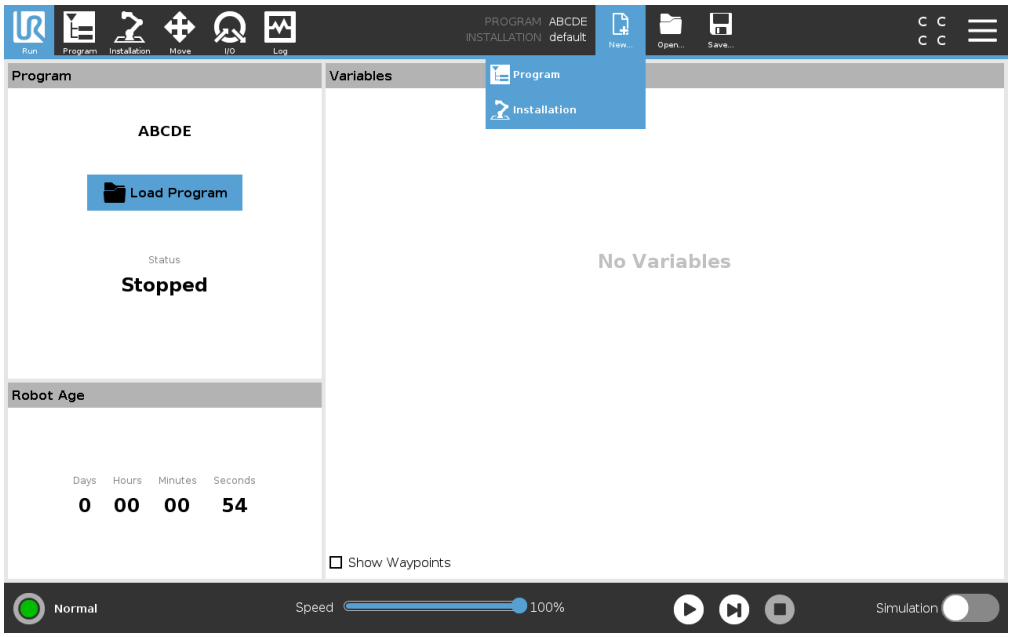
1. In the Program and Installation Manager, tap **Open...** and select Program.
2. On the Load Program screen, select an existing program and tap Open.
3. In the File Path, verify that the desired program name is displayed.

Opening an Installation.

1. In the Program and Installation Manager, tap **Open...** and select Installation.
2. On the Load Robot Installation screen, select an existing installation and tap Open.
3. In the Safety Configuration box, select Apply and restart to prompt robot reboot.
4. Select Set Installation to set installation for the current Program.
5. In the File Path, verify that the desired installation name is displayed.

20.2 New...

Allows you to create a new Program and/or Installation.



Creating a new Program

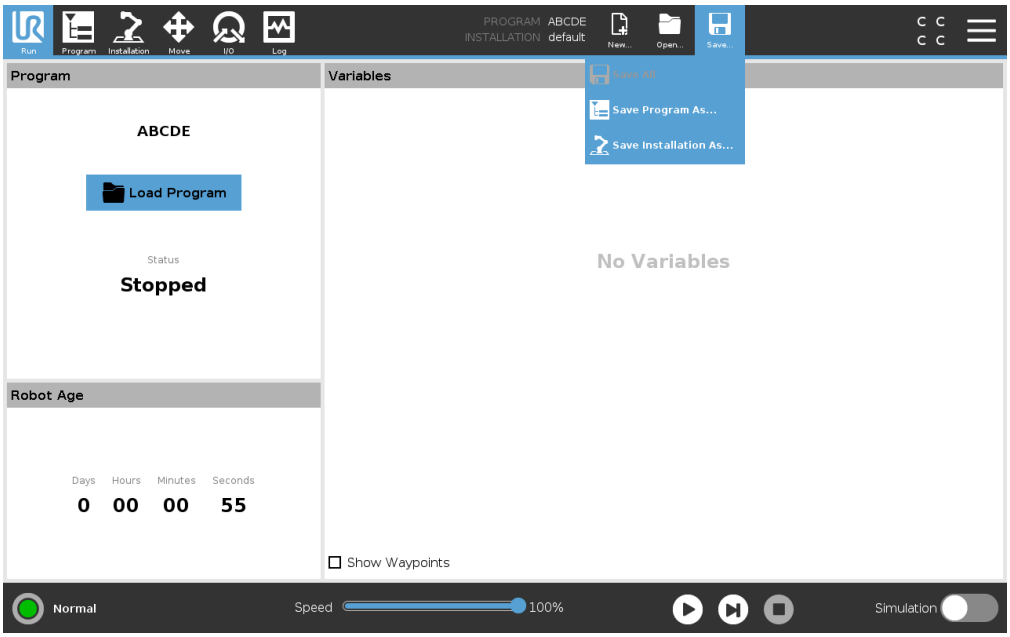
1. In the Program and Installation Manager, tap **New...** and select Program.
2. On the Program screen, configure your new program as desired.
3. In the Program and Installation Manager, tap **Save...** and select Save All or Save Program As...
4. On the Save Program As screen, assign a file name and tap Save.
5. In the File Path, verify that the new program name is displayed.

Creating a new Installation
Note: You must save an installation for use after robot power down.

1. In the Program and Installation Manager, tap **New...** and select Installation.
2. Tap Confirm Safety Configuration.

3. On the Installation screen, configure your new installation as desired.
4. In the Program and Installation Manager, tap **Save...** and select Save Installation As...
5. On the Save Robot Installation screen, assign a file name and tap Save.
6. Select Set Installation to set installation for the current Program.
7. In File Path, verify that the new installation name is displayed.

20.3 Save...



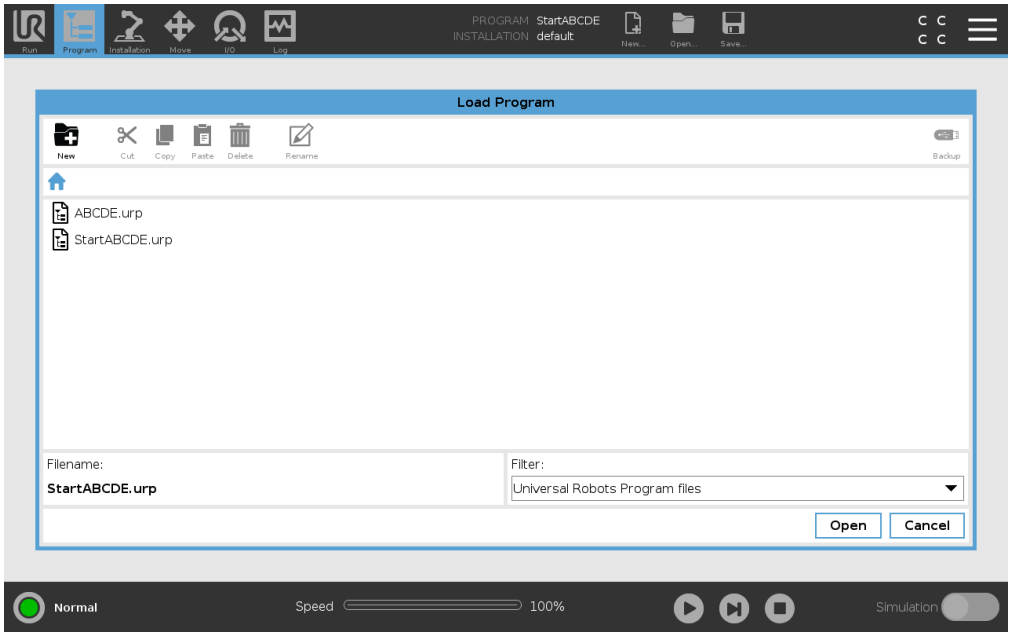
Save... proposes three options. Depending on the program/installation you load-create, you can:

Save All to save the current Program and Installation immediately, without the system prompting to save to a different location or different name. Note: If no changes are made to the Program or Installation, the Save All... button appears deactivated.

Save Program As... to change the new Program name and location. Note: the current Installation is also saved, with the existing name and location.

Save Installation As... to change the new Installation name and location. Note: the current Program is saved, with the existing name and location.

20.4 File manager



This image shows the load screen which consists of the following buttons:

Breadcrumb Path The breadcrumb path shows a list of directories leading to the present location. By selecting a directory name in the breadcrumb, the location changes to that directory and displays it in the file selection area.

File Selection Area Tap the name of a file to open it. Directories are selected by pressing their name for half a second.

File Filter You can specify the file types shown. After selecting Backup Files, this area displays the 10 most recently saved program versions, where '.old0' is the newest and '.old9' is the oldest.

Filename The selected file is shown here. When saving a file, use the text field to manually enter the file name.

Action buttons The action bar consists of a series of buttons that enable you to manage files. The 'Backup' action to the right of the action bar supports backing up the currently selected files and directories to the location and to a USB. The 'Backup' action is only enabled when an external media is attached to the USB port.

21 Hamburger menu

21.1 Help

You can find definitions for all elements that make up PolyScope capabilities.

1. In the right corner of the **Header**, tap the **Hamburger** menu and select **Help**.
2. Tap one of the red question marks that appears, to define desired element.
3. In the top right corner of element definition screen, tap the red X to exit Help.

21.2 About

You can display Version and Legal data.

1. Tap the **Hamburger** menu and select **About**.
2. Tap either **Version** or **Legal** to display data.
3. Tap Close to return to your screen.

21.3 Settings

Personalizing PolyScope Settings

1. In the Header, tap the Hamburger menu and select **Settings**.
2. On the left, in the Settings screen action menu, select an item to personalize. Note: If an operational mode password was set, in the action menu, **System** is only available to the programmer.
3. On the bottom right, tap **Apply and Restart** to implement your changes.
4. On the bottom left, tap **Exit** to close Settings screen without changes.

21.3.1 Preferences

Language

You can change PolyScope language and measurement unit (Metric or Imperial).

Time

You can access and/or adjust the current time and date displayed on the PolyScope.

1. In the Header, tap the Hamburger menu icon and select **Settings**.
2. Under Preferences, select **Time**.
3. Verify and/or adjust **Time** and/or **Date** as desired.
4. Tap **Apply and Restart** to apply your changes.

Date and Time are displayed in the Log tab (see 19.3) under **Date Log**.

Hiding Speed Slider

Located at the base of the Run tab screen, the Speed Slider allows the operator to change the speed of a running Program.

1. In the Header, tap the Hamburger menu icon and select **Settings**.
2. Under Preferences, tap **Run Screen**.
3. Select check box to show or hide **Speed Slider**.

21.3.2 Password

Mode

The operational mode password prevents unauthorized modification of robot setup, by creating two different user roles on PolyScope: Automatic and Maual. When you set operational mode password, programs or installations can only be created and loaded in manual mode. Any time you enter manual mode, PolyScope prompts for the password that was previously set on this screen.

Safety

The Safety password prevents unauthorized modification of the Safety settings.

21.3.3 System

Update

You can use a USB to apply updates to ensure the robot software is up-to-date.

1. Insert an USB and tap **Search** to list update files.
2. Select desired update and tap **Update** to install.
3. Follow the on-screen instructions to complete update.

WARNING:
Always check your program/s after a software upgrade. The upgrade might change the trajectories in your program.

Network

You can configure robot connection to a network by selecting one of three available network methods:

- DHCP
- Static Address
- Disabled network (if you don't wish to connect your robot to a network)

Depending on the network method you select, configure your network settings:

- IP Address

- Subnet Mask
- Default Gateway
- Preferred DNS Server
- Alternative DNS Server

Note: Press **Apply** to apply changes.

URCaps

You can manage your existing **URCaps** or install a new one in your robot.

1. In the Header, press the Hamburger menu and select **Settings**.
2. Under System, select **URCaps**.
3. Press the **+** button, select the **.urcap** file and press **Open** Note: Check more details about the new URCaps by selecting it in **Active URCaps** field. More information appears below in **URCaps Information** field below.
4. If you wish to proceed with the installation of that URCap, press **Restart**. After that step, the URCaps is installed and ready to be used.
5. To remove an installed URCaps, select it from Active URCaps, press the **-** button and press **Restart** so changes can take effect.

Remote Control

A robot can either be in Local Control (controlled from the Teach Pendant) or Remote Control (controlled externally).

Local Control does not allow	Remote Control does not allow
Power on and brake release sent to the robot over network	Moving the robot from Move Tab
Receiving and executing robot programs and installation sent to the robot over network	Starting from Teach Pendant
Autostart of programs at boot, controlled from digital inputs	Load programs and installations from the Teach Pendant
Auto brake release at boot, controlled from digital inputs	Freedrive
Start of programs, controlled from digital inputs	

Control of the robot via network or digital input is, by default, restricted. Enabling and selecting the Remote Control feature removes this restriction. Enable Remote Control by switching to the Local Control profile (PolyScope control) of the robot, allowing all control of running programs and executing scripts to be performed remotely.
Note: Enable the Remote Control feature in Settings to access Remote mode and Local mode in the profile.

1. In the Header, press the Hamburger menu and select **Settings**.
2. Under System, select **Remote Control**.
3. Press **Enable** to make the Remote Control feature available. PolyScope remains active. Note: Enabling Remote Control does not immediately start the feature. It allows you to switch from Local Control to Remote Control.
4. In the profile menu, select **Remote Control** to alter PolyScope. Note: You can return to Local Control by switching back in the profile menu, or selecting Operator or Programmer if a password is specified.

NOTE:

- Although Remote Control limits your actions in PolyScope, you can still monitor robot state.
- When a robot system is powered off in Remote Control, it starts up in Remote Control.

21.4 Shutdown Robot

The **Shutdown Robot** button allows the robot to be powered off or restarted.

Shutting Down the Robot

1. In the Header, tap the Hamburger menu and select **Shutdown Robot**.
2. When the Shutdown Robot dialog box appears, tap **Power Off**.