UNIVERSAL ROBOTS

PolyScope Manual

Version 3.0 (rev. 15965)

Original instructions (en)

US version

Contents

Part II

PolyScope Manual

9 Introduction

The Universal Robot arm is composed of extruded aluminum tubes and joints. The joints with their usual names are shown in Figure 9.1. The *Base* is where the robot is mounted, and at the other end (*Wrist 3*) the tool of the robot is attached. By coordinating the motion of each of the joints, the robot can move its tool around freely, with the exception of the area directly above and directly below the base. The reach of the robot is 800 mm from the center of the base.

PolyScope is the graphical user interface (GUI) which lets you operate the robot arm and control box, execute robot programs and easily create new ones.

The following section gets you started with the robot. Afterwards, the screens and functionality of PolyScope are explained in more detail.

9.1 Getting Started

Before using PolyScope, the robot arm and control box must be installed and the control box switched on.

9.1.1 Installing the Robot Arm and Control Box

To install the robot arm and control box, do the following:

1. Unpack the robot arm and the control box.
2. Mount the robot 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 shall be vibration-free.
3. Place the control box on its foot.
4. Plug on the robot cable between the robot and the control box.
5. Plug in the mains plug of the control box.

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

Detailed installation instructions can be found in the Hardware Installation Manual. Note that a risk assessment is required before using the robot arm to do any work.

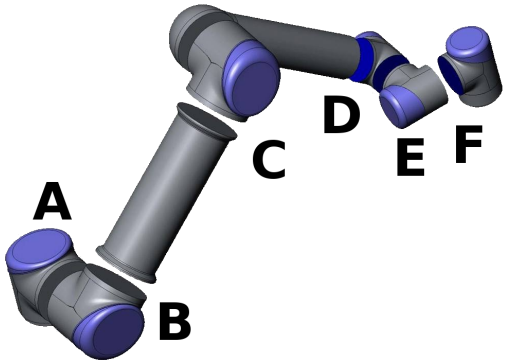


Figure 9.1: Joints of the robot. A: *Base*, B: *Shoulder*, C: *Elbow* and D, E, F: *Wrist 1, 2, 3*

9.1.2 Turning the Control Box On and Off

The control box is turned on by pressing the power button at the front side of the panel with the touch screen. This panel is usually referred to as the *teach pendant*. When the control box is turned on, text from the underlying operating system will appear on the touch screen. After about one minute, a few buttons appear on the screen and a popup guides the user to the initialization screen (see 9.4).

To shut down the control box, press the green power button on the screen, or use the `Shut Down` button on the welcome screen (see 9.3).

WARNING:
Shutting down by pulling the power cord from the wall socket may cause corruption of the robot's file system, which may result in robot malfunction.

9.1.3 Turning the Robot Arm On and Off

The robot arm can be turned on if the control box is turned on, and if no emergency stop button is activated. Turning the robot arm on is done in the initialization screen (see 9.4) by touching the `ON` button on that screen, and then pressing `Start`. When a robot is started, it makes a sound and moves a little while releasing the brakes.

The power to the robot arm can be turned off by touching the `OFF` button on the initialization screen. The robot arm is also powered off automatically when the control box shuts down.

9.1.4 Quick Start

To quickly start up the robot after it has been installed, perform the following steps:

1. Press the Emergency Stop button on the front side of the teach pendant.
2. Press the power button on the teach pendant.
3. Wait a minute while the system is starting up, displaying text on the touch screen.
4. When the system is ready, a popup will be shown on the touch screen, stating that the robot needs to be initialized.
5. Touch the button on the popup dialog. You will be taken to the initialization screen.
6. Wait for the `Confirmation of applied Safety Configuration` dialog and press the `Confirm Safety Configuration` button. This applies an initial set of safety parameters that need to be adjusted based on a risk assessment.
7. Unlock the Emergency Stop button. The robot state changes from `Emergency Stopped` to `Power off`.
8. Step outside the reach (workspace) of the robot.
9. Touch the `On` button on the touch screen. Wait a few seconds until robot state changes to `Idle`.
10. Verify that the payload mass and selected mounting are correct. You will be notified if the mounting detected based on sensor data does not match the selected mounting.
11. Touch the `Start` button on the touch screen. The robot now makes a sound and moves a little while releasing the brakes.
12. Touch the `OK` button, bringing you to the `Welcome` screen.

9.1.5 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.

Since tool motion is an important part of a robot program, a way of teaching the robot how to move is essential. In PolyScope, motions of the tool are given using a series of *waypoints*, i.e. points in the robot's workspace. A waypoint can be given by moving the robot to a certain position, or it can be calculated by software. In order to move the robot arm to a certain position, use either the Move tab (see 11.1), or simply pull the robot arm into place while holding the *teach button* at the back side 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.

To create a simple program on a robot that has been started up, do the following:

1. Touch the `Program Robot` button and select `Empty Program`.

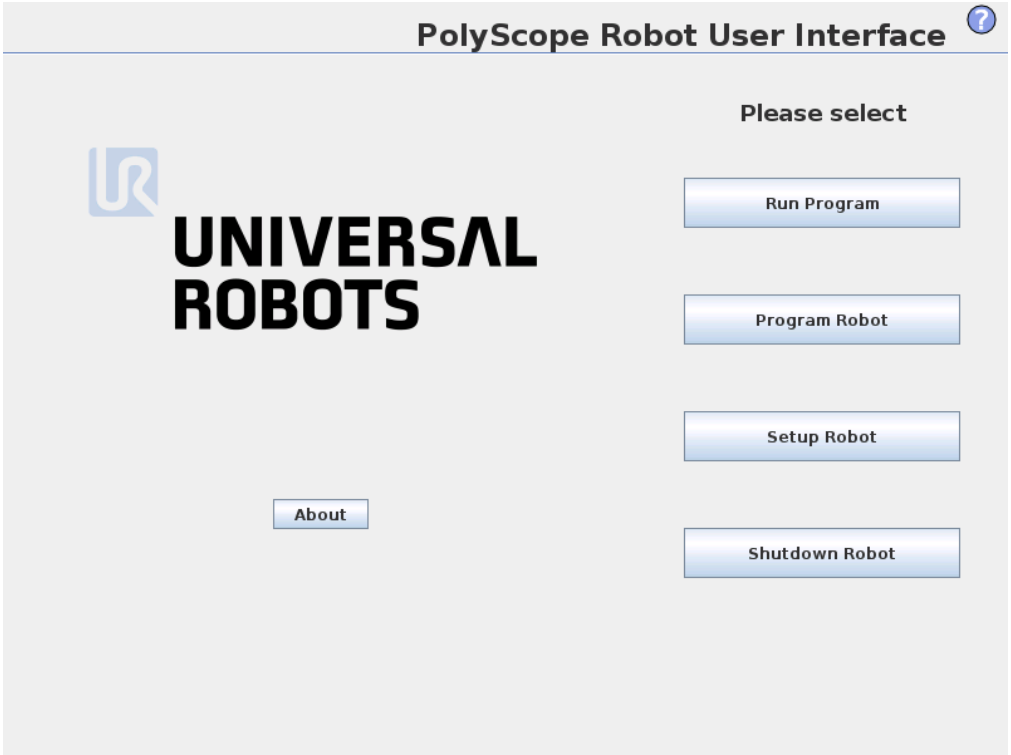
2. Touch the Next button (bottom right) so that the <empty> line is selected in the tree structure on the left side of the screen.
3. Go to the Structure tab.
4. Touch the Move button.
5. Go to the Command tab.
6. Press the Next button, to go to the Waypoint settings.
7. Press the Set this waypoint button next to the ''?'' picture.
8. On the Move screen, move the robot by pressing the various blue arrows, or move the robot by holding the Teach button, placed on the backside of the teach pendant, while pulling the robot arm.
9. Press OK.
10. Press Add waypoint before.
11. Press the Set this waypoint button next to the ''?'' picture.
12. On the Move screen, move the robot by pressing the various blue arrows, or move the robot by holding the Teach button while pulling the robot arm.
13. Press OK.with
14. Your program is ready. The robot will move between the two points when you press the "Play" symbol. Stand clear, hold on to the emergency stop button and press "Play".
15. Congratulations! You have now produced your first robot program that moves the robot between the two given waypoints.

WARNING:

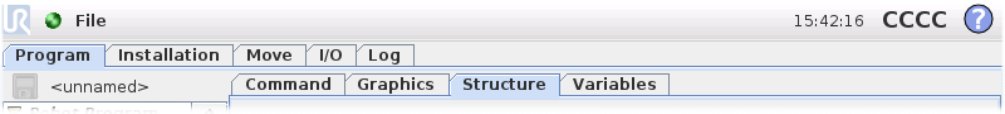
1. Do not drive the robot into itself or anything else as this may cause damage to the robot.
2. Keep your head and torso outside the reach (workspace) of the robot. Do not place fingers where they can be caught.
3. This is only a quick start guide to show how easy it is to use a UR robot. It assumes 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.

9.2 PolyScope Programming Interface

PolyScope runs on the touch sensitive screen attached to the control box.



The picture above shows the Welcome Screen. The bluish areas of the screen are buttons that can be pressed by pressing a finger or the backside of a pen against the screen. PolyScope has a hierarchical structure of screens. In the programming environment, the screens are arranged in *tabs*, for easy access on the screens.



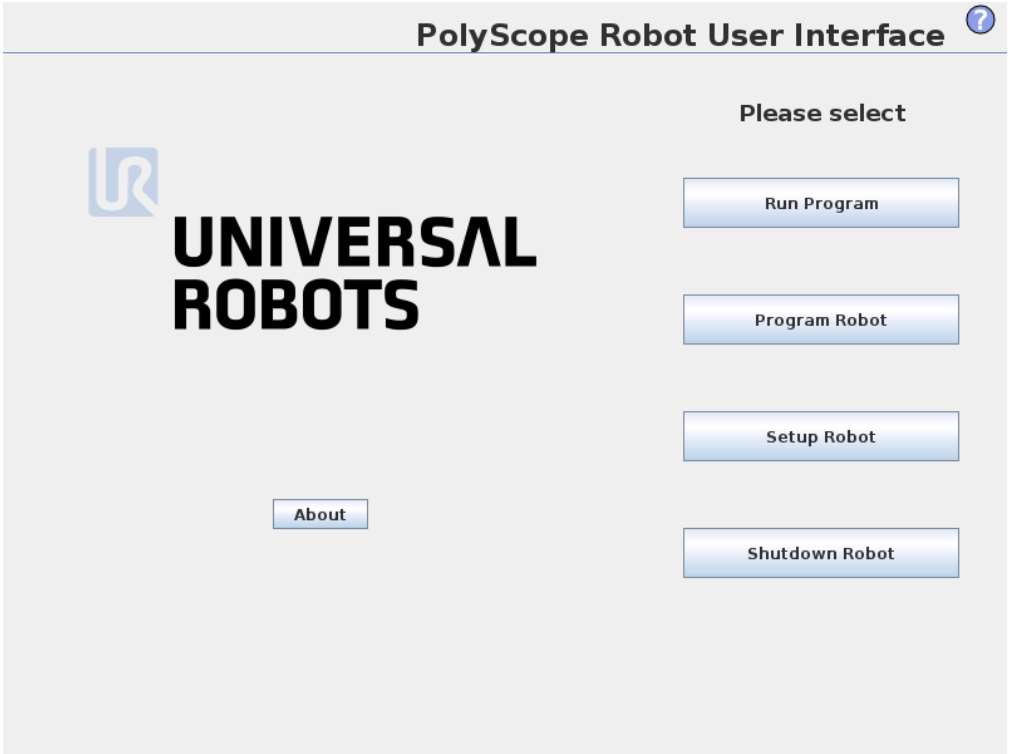
In this example, the `Program` tab is selected at the top level, and under that the `Structure` tab is selected. The `Program` tab holds information related to the currently loaded program. If the `Move` tab is selected, the screen changes to the *Move* screen, from where the robot arm can be moved. Similarly, by selecting the `I/O` tab, the current state of the electrical I/O can be monitored and changed.

It is possible to connect a mouse and a keyboard to the control box or the teach pendant; however, this is not required. Almost all text fields are touch-enabled, so touching them launches an on-screen keypad or keyboard. Non-touchable text fields have an editor icon next to them that launches the associated input editor.

The icons of the on-screen keypad, keyboard and expression editor are shown above.

The various screens of PolyScope are described in the following sections.

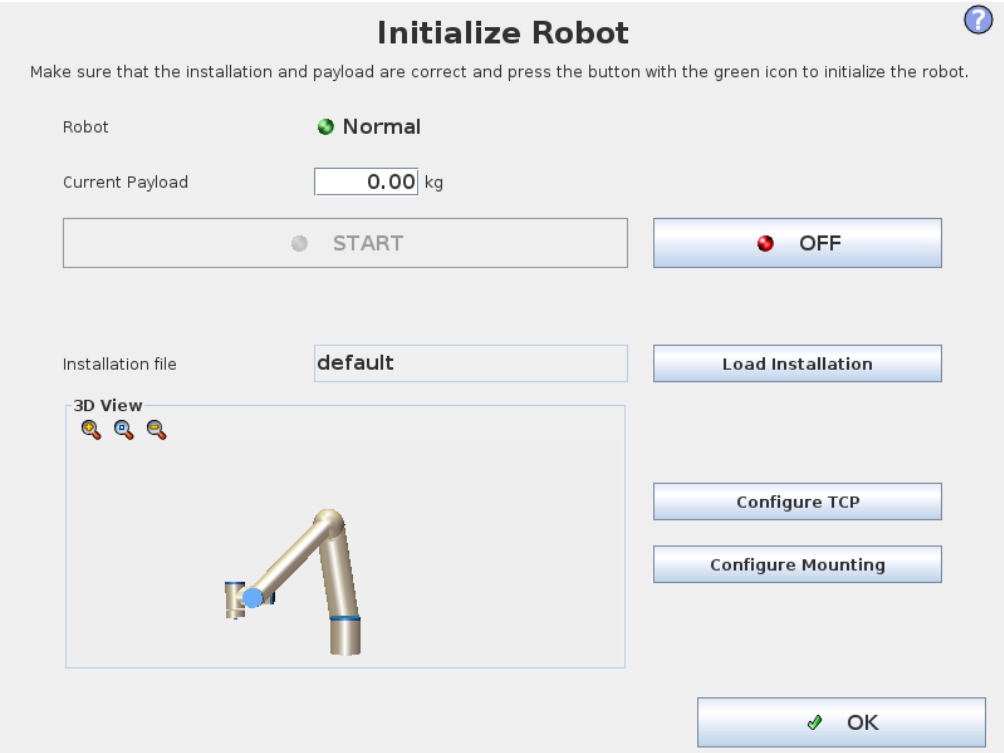
9.3 Welcome Screen



After booting up the controller PC, the welcome screen is shown. The screen offers the following options:

- **Run Program:** Choose and run an existing program. This is the simplest way to operate the robot arm and control box.
- **Program Robot:** Change a program, or create a new program.
- **Setup Robot:** Set passwords, upgrade software, request support, calibrate the touch screen, etc.
- **Shutdown Robot:** Powers off the robot arm and shuts down the control box.

9.4 Initialization Screen



On this screen you control the initialization of the robot arm.

Robot arm state indicator

The status LED gives an indicaton of the robot arm's running state:

- A bright red LED indicates that the robot arm is currently in a stopped state where the reasons can be several.
- A bright yellow LED indicates that the robot arm is powered on, but is not ready for normal operation.
- Finally, a green LED indicates that the robot arm is powered on, and ready for normal operation.

The text appearing next to the LED further specifies the current state of the robot arm.

Active payload and installation

When the robot arm is powered on, the payload mass used by the controller when operating the robot arm is shown in the small white text field. This value can be modified by tapping the text field and entering a new value. Note that setting this value does not modify the payload in the robot's installation (see 11.6), it only sets the payload mass to be used by the controller.

Similarly, the name of the installation file that is currently loaded is shown in the grey text field. A different installation can be loaded by tapping the text field or by using the `Load` button next to it. Alternatively, the loaded installation can be customized using the buttons next to the 3D view in the lower part of the screen.

Before starting up the robot arm, it is very important to verify that both the active payload and the active installation correspond to the actual situation the robot arm is currently in.

Initializing the robot arm

DANGER:
Always verify that the actual payload and installation are correct before starting up the robot arm. If these settings are wrong, the robot arm and control box will not function correctly and may become dangerous to people or equipment around them.

CAUTION:
Great care should be taken if the robot arm is touching an obstacle or table, since driving the robot arm into the obstacle might damage a joint gearbox.

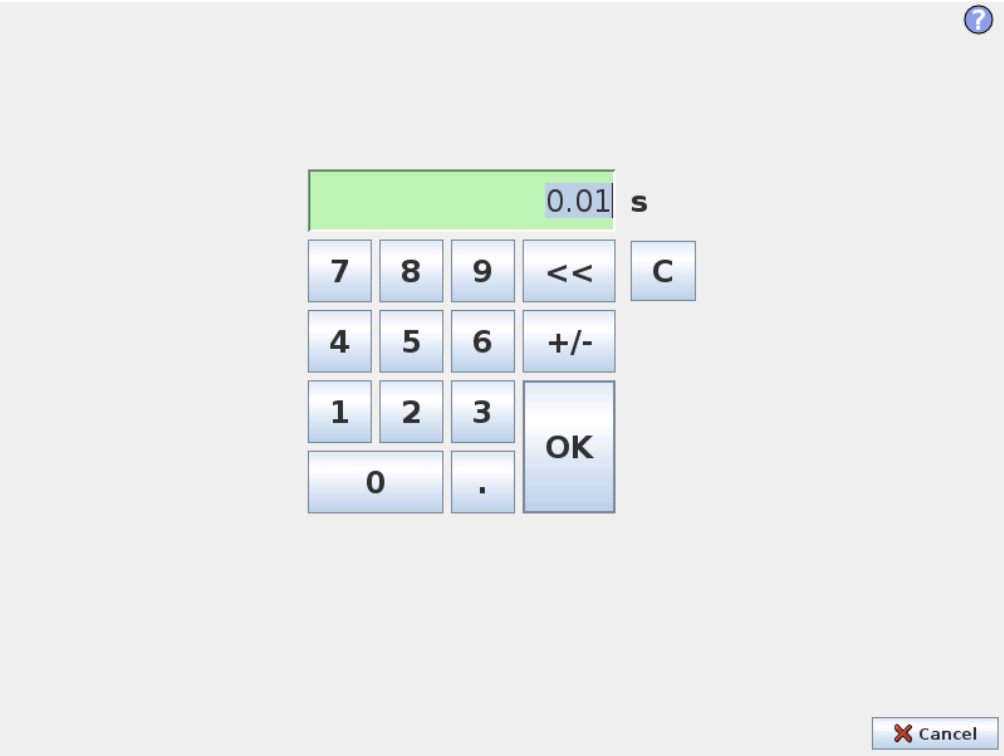
The large button with the green icon on it serves to perform the actual initialization of the robot arm. The text on it, and the action it performs, change depending on the current state of the robot arm.

- After the controller PC boots up, the button needs to be tapped once to power the robot arm on. The robot arm state then turns to *Power on* and subsequently to *Idle*. Note that when an emergency stop is in place, the robot arm cannot be powered on, so the button will be disabled.
- When the robot arm state is *Idle*, the button needs to be tapped once again to start the robot arm up. 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 verification passes, tapping the button releases all joint brakes and the robot arm becomes ready for normal operation. Note that the robot makes a sound and moves a little while releasing the brakes.
- If the robot arm violates one of the safety limits after it starts up, it operates in a special *Recovery mode*. In this mode, tapping the button switches to a recovery move screen where the robot arm can be moved back within the safety limits.
- If a fault occurs, the controller can be restarted using the button.
- If the controller is currently not running, tapping the button starts it.

Finally, the smaller button with the red icon on it serves to power off the robot arm.

10 On-screen Editors

10.1 On-screen Keypad

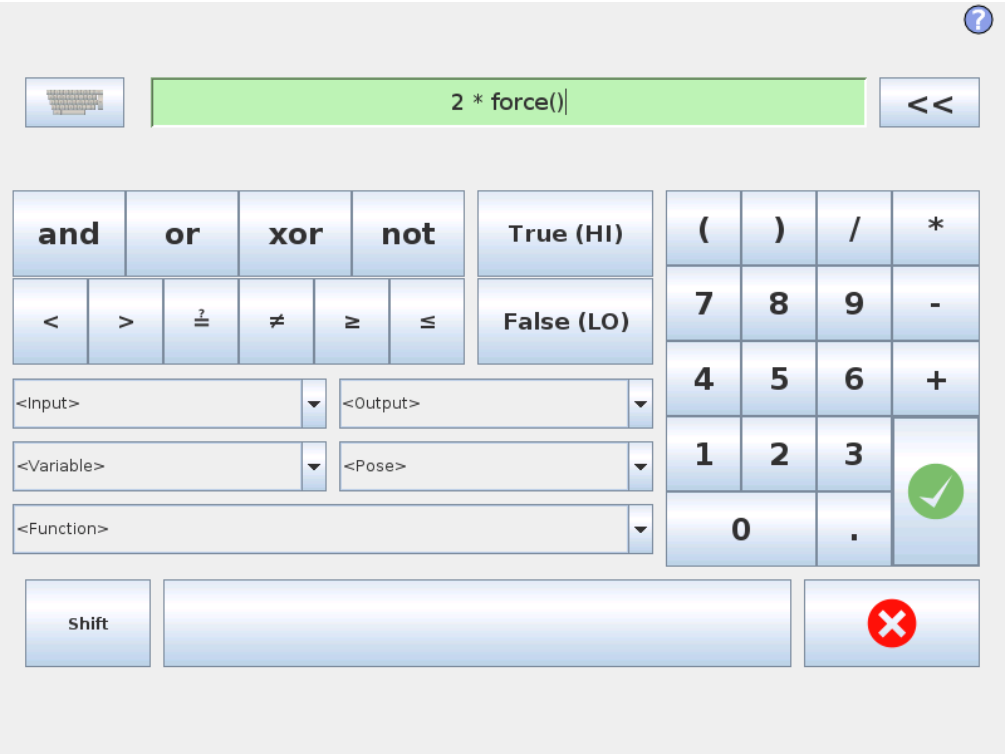


Simple number typing and editing facilities. In many cases, the unit of the typed value is displayed next to the number.

10.2 On-screen Keyboard

Simple text typing and editing facilities. The `Shift` key can be used to get some additional special characters.

10.3 On-screen 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 right 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`.

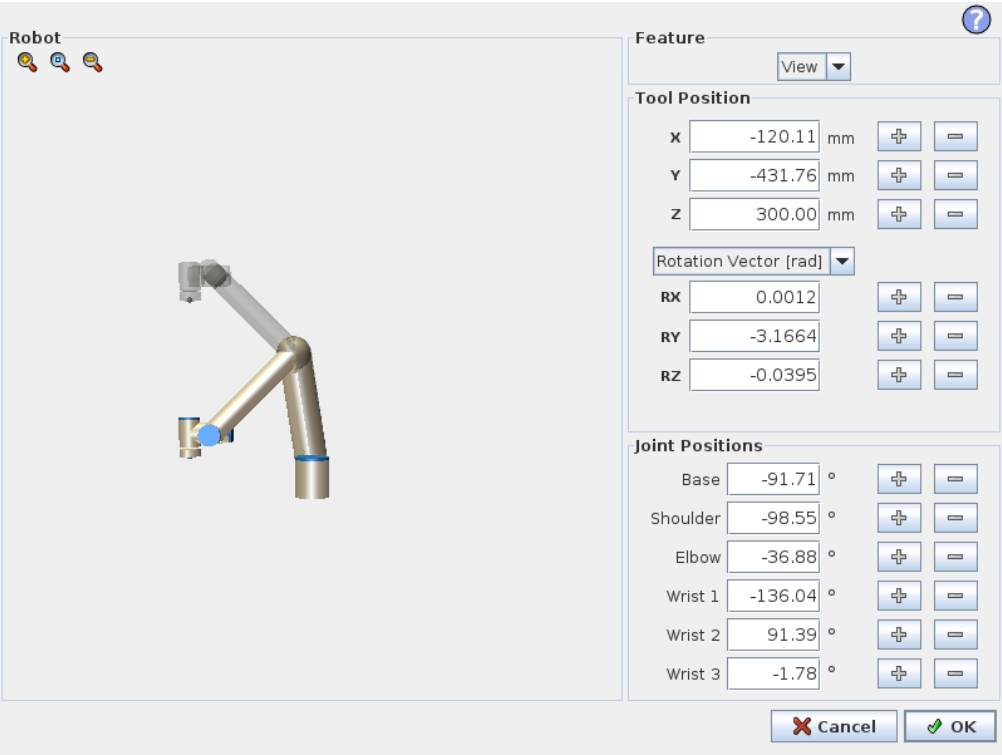
The expression is checked for grammatical errors when the `Ok` button is pressed. The `Cancel` button leaves the screen, discarding all changes.

An expression can look like this:

```
digital_in[1]≟True and analog_in[0]<0.5
```

10.4 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 14.11), 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 14.5) 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

At the top right part of the screen, the feature selector can be found. The feature selector defines which feature to control the robot arm relative to, while below it, the boxes display the full coordinate value for the tool 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 11.1), 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 12.5.

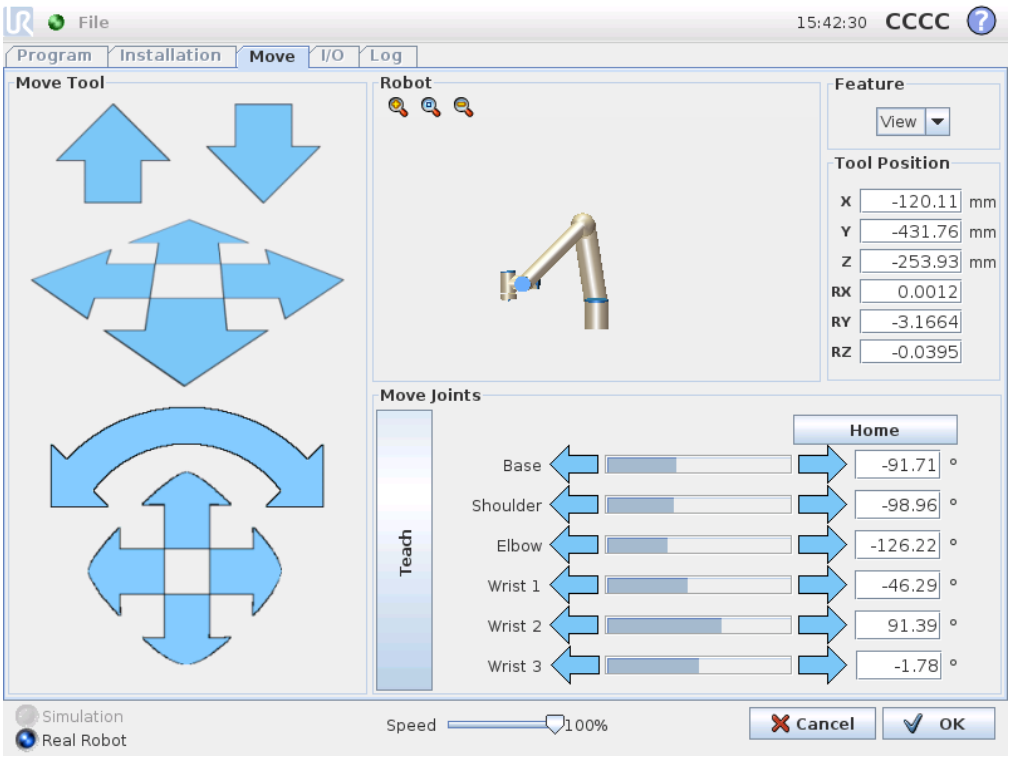
Cancel button

Clicking the `Cancel` button leaves the screen discarding all changes.

11 Robot Control

11.1 Move Tab

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



11.1.1 Robot

The current position of the robot arm is shown in 3D graphics. Push the magnifying glass icons to zoom in/out or drag a finger across to change the view. To get the best feel for controlling the robot arm, select the `View` feature and rotate the viewing angle of the 3D drawing to match your view of the real robot arm.

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 14.11), a 3D representation of the proximate boundary limit is shown. Note that 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 14.5) 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 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.

11.1.2 Feature and Tool Position

At the top right part of the screen, the feature selector can be found. It defines which feature to control the robot arm relative to, while below it, the boxes display the full coordinate value for the tool relative to the selected feature.

Values can be edited manually by clicking on the coordinate or the joint position. This will take you to the pose editor screen (see 10.4) where you can specify a target position and orientation for the tool or target joint positions.

11.1.3 Move Tool

- Holding down a `translate arrow` (top) will move the tool-tip of the robot in the direction indicated.
- Holding down a `rotate arrow` (button) will change the orientation of the robot tool in the indicated direction. The point of rotation 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.

Note: *Release the button to stop the motion at any time!*

11.1.4 Move Joints

Allows the individual joints to be controlled directly. Each joint can move from -360° to $+360^\circ$, which are the default *joint limits* illustrated by the horizontal bar for each joint. If a joint reaches its joint limit, it cannot be driven any further. If the limits for a joint have been configured with a position range different from the default (see 14.10), this range is indicated with red in the horizontal bar.

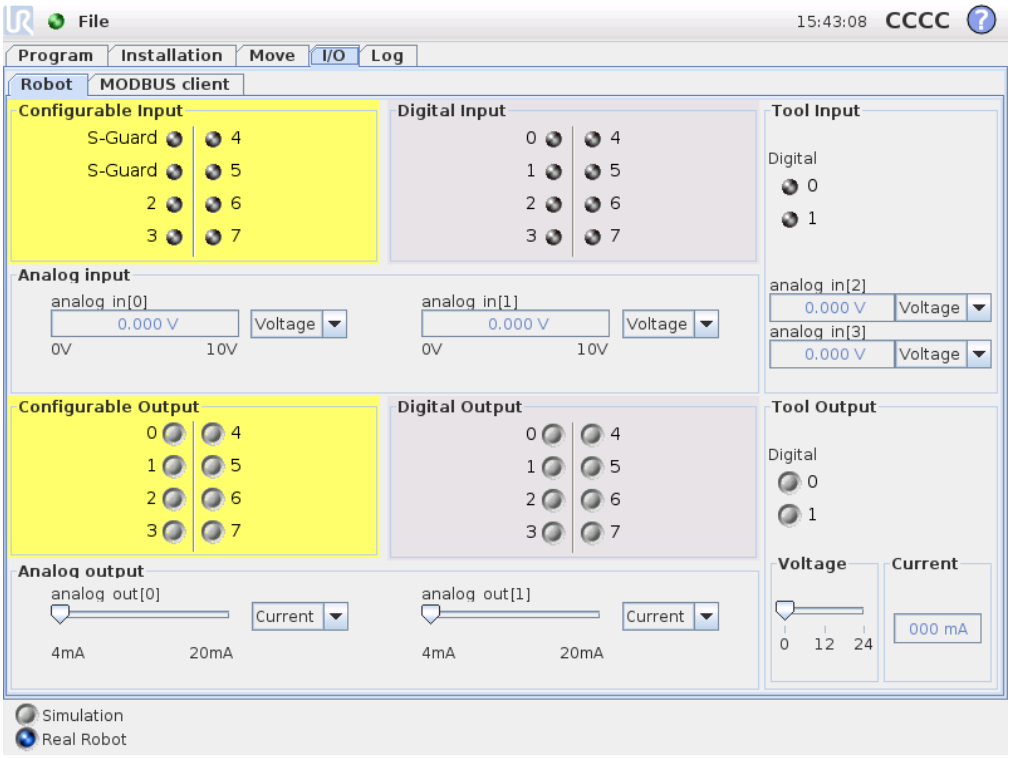
11.1.5 Teach

While the *Teach* button is held down, it is possible to physically grab the robot arm and pull it to where you want it to be. If the gravity setting (see 11.7) in the `Setup` tab is wrong, or the robot arm carries a heavy load, the robot arm might start moving (falling) when the *Teach* button is pressed. In that case, just release the *Teach* button again.

WARNING:

1. 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.
2. Make sure that the TCP settings and the robot mounting settings are set correctly before operating the `Teach` button. If these settings are not correct, the robot arm will move when the `Teach` button is activated.
3. The teach function (impedance/backdrive) shall only be used in installations where the risk assessment allows it. Tools and obstacles shall not have sharp edges or pinch points. Make sure that all personnel remain outside the reach of the robot arm.

11.2 I/O Tab



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, inluding 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.

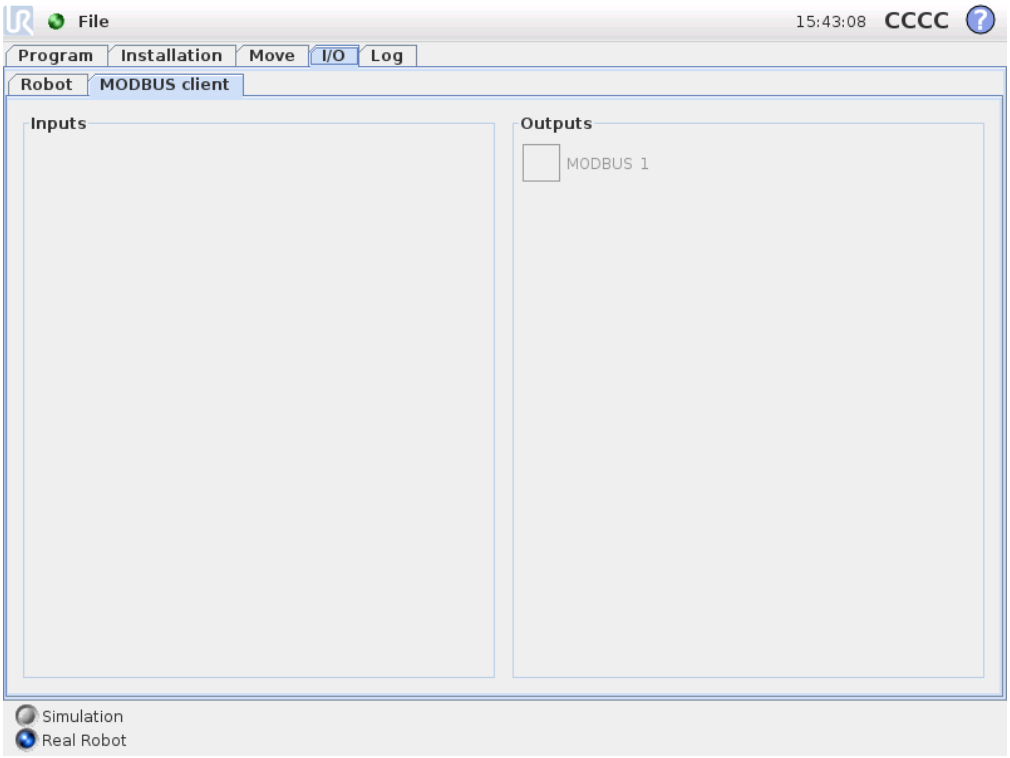
Configurable I/O's can be reserved for special safety settings defined in the safety I/O configuration section of the installaton (see 14.12); those which are reserved will have the name of the safety function in place of the default or user defined name. Configurable outputs that are reserved for safety settings are not togglable and will be displaed as LED's only.

The electrical details of the signals are described in the user manual.

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.

11.3 MODBUS client I/O

Here, the digital MODBUS client I/O signals as set up in the installation are shown. If the signal connection is lost, the corresponding entry on this screen is disabled.



Inputs

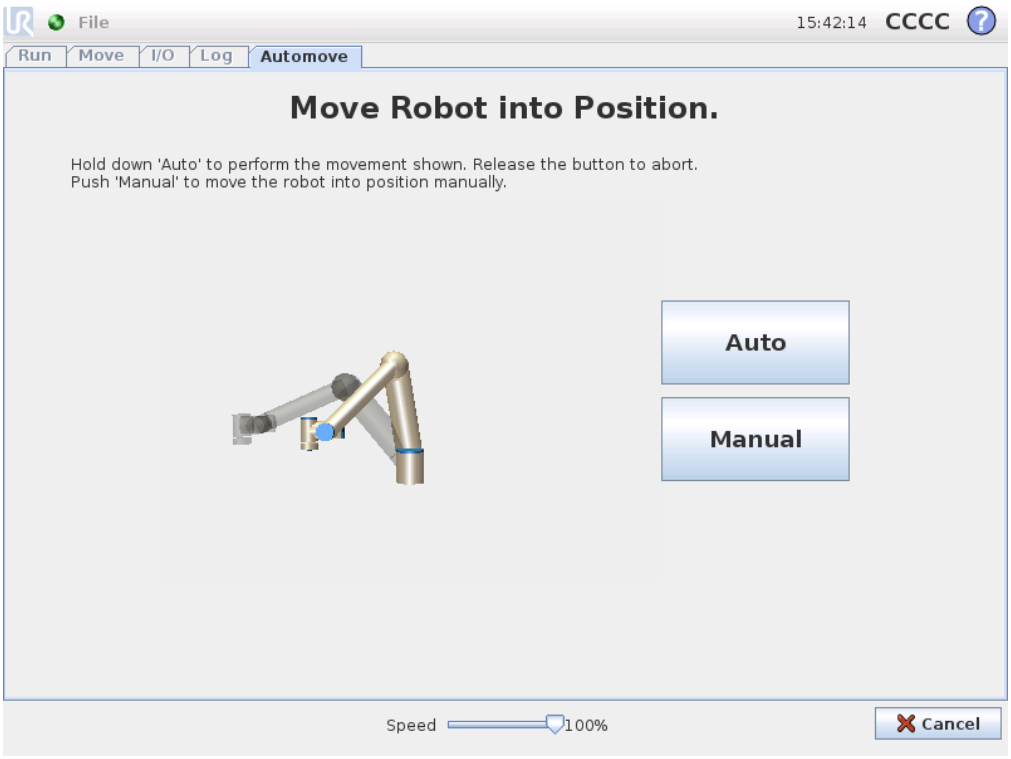
View the state of digital MODBUS client inputs.

Outputs

View and toggle the state of digital MODBUS client outputs. A signal can only be toggled if the choice for I/O tab control (described in 11.8) allows it.

11.4 AutoMove Tab

The AutoMove tab is used when the robot arm has to move to a specific position in its workspace. Examples are when the robot arm has to move to the start position of a program before running it, or when moving to a waypoint while modifying a program.



Animation

The animation shows the movement the robot arm is about to perform.

CAUTION:
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.

CAUTION:
The automove function moves in joint space, not in linear (Cartesian) space. Collision might damage the robot or other equipment.

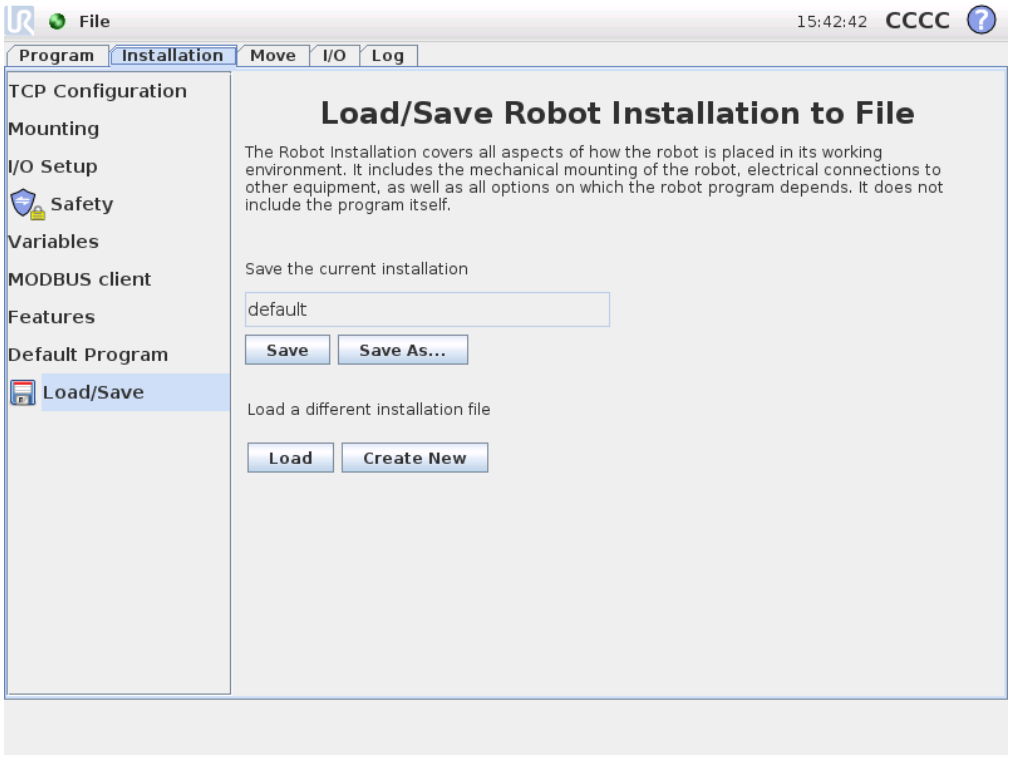
Auto

Hold down the `Auto` button to move the robot arm as shown in the animation. Note: *Release the button to stop the motion at any time!*

Manual

Pushing the `Manual` button will take you to the `MoveTab` where the robot arm can be moved manually. This is only needed if the movement in the animation is not preferable.

11.5 Installation → Load/Save



The Robot Installation covers all aspects of how the robot arm and control box are placed in the working environment. It includes the mechanical mounting of the robot arm, electrical connections to other equipment, as well as all options on which the robot program depends. It does not include the program itself.

These settings can be set using the various screens under the `Installation` tab, except for the I/O domains which are set in the `I/O` tab (see 11.2).

It is possible to have more than one installation file for the robot. Programs created will use the active installation, and will load this installation automatically when used.

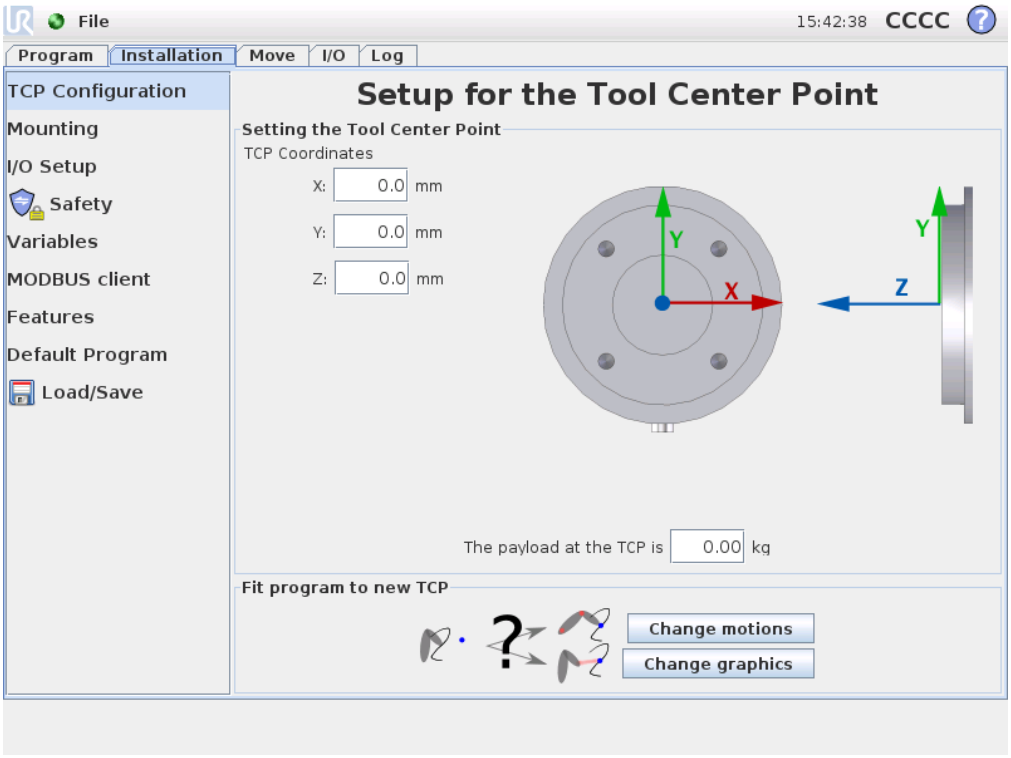
Any changes to an installation need to be saved to be preserved after power down. If there are unsaved changes in the installation, a floppy disk icon is shown next to the

Load/Save text on the left side of the Installation tab.

Saving an installation can be done by pressing the Save or Save As... button. Alternatively, saving a program also saves the active installation. To load a different installation file, use the Load button. The Create New button resets all of the settings in the Robot Installation to their factory defaults.

CAUTION:
Using the robot with an installation loaded from a USB drive is not recommended. To use an installation stored on a USB drive, first load it and then save it in the local programs folder using the Save As... button.

11.6 Installation → TCP Configuration



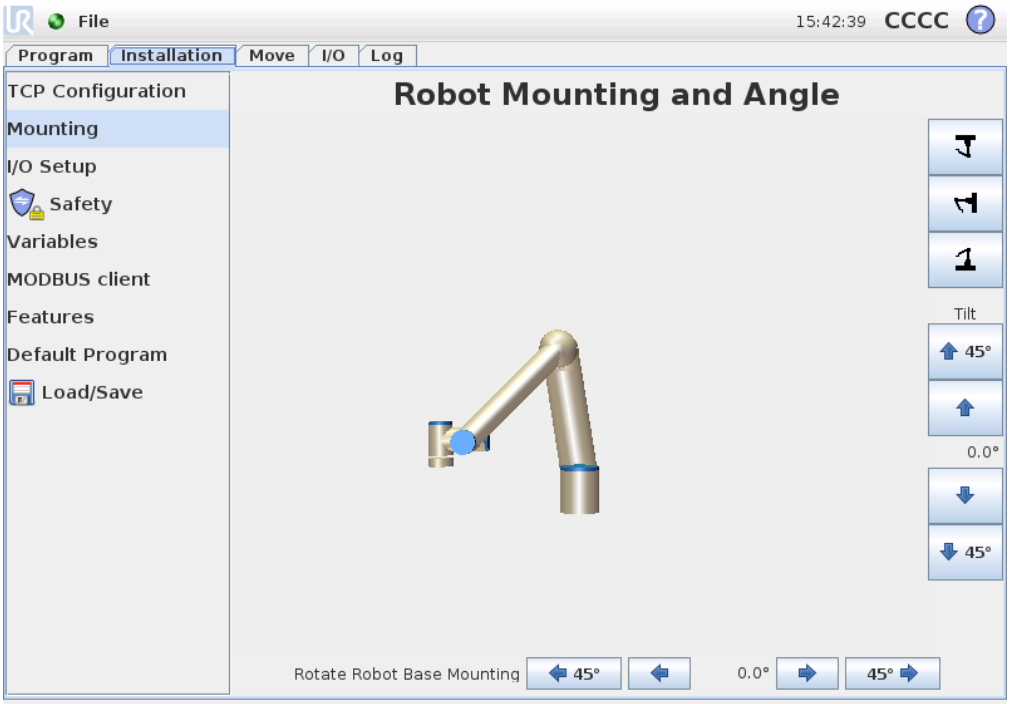
The *Tool Center Point* (TCP) is the point at the end of the robot arm that gives a characteristic point on the robot's tool. When the robot arm moves linearly, it is this point that moves in a straight line. It is also the motion of the TCP that is visualized on the graphics tab. The TCP is given relative to the center of the tool output flange, as indicated on the on-screen graphics.

WARNING:
Make sure to use the correct installation settings. Save and load the installation files along with the program.

The two buttons on the bottom of the screen are relevant when the TCP is changed.

- **Change Motions** recalculates all positions in the robot program to fit the new TCP. This is relevant when the shape or size of the tools has been changed.
- **Change Graphics** redraws the graphics of the program to fit the new TCP. This is relevant when the TCP has been changed without any physical changes to the tool.

11.7 Installation → Mounting



Here the mounting of the robot arm can be specified. This serves two purposes:

1. Making the robot arm look right on the screen.
2. Telling the controller about the direction of gravity.

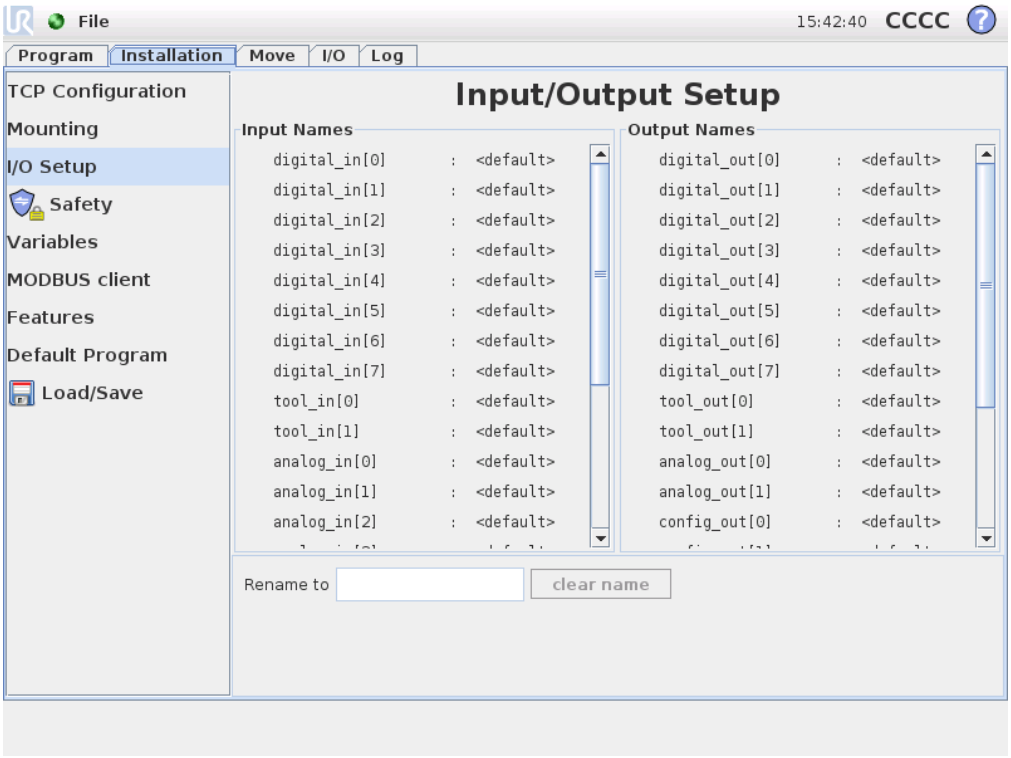
The controller uses an advanced dynamics model to give the robot arm smooth and precise motions, and to make the robot arm hold itself when in *Teach* mode. For this reason, it is very important that the mounting of the robot arm be set correctly.

WARNING:
Failure to set robot arm's mounting correctly might result in frequent protective stops, and/or a possibility that the robot arm will move when the teach button is pressed.

The default is that the robot arm is mounted on a flat table or floor, in which case 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 push-buttons. The buttons on the right side of the screen are for setting the angle of the robot arm's mounting. The three top right side buttons set the angle to *ceiling* (180°), *wall* (90°), *floor* (0°). The `Tilt` buttons can be used to 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:
Make sure to use the correct installation settings. Save and load the installation files along with the program.

11.8 Installation → I/O Setup



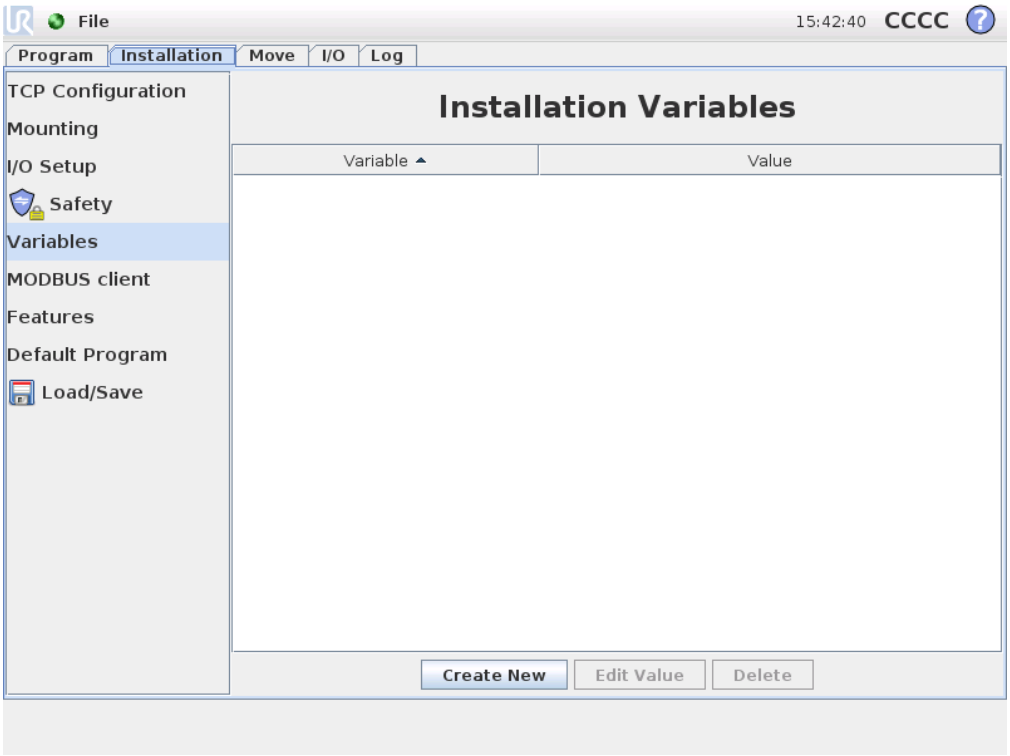
Input and output signals can be given names. This can make it easier to remember what the signal does when working with the robot. Select an I/O by clicking on it, and set the name using the on screen keyboard. You can set the name back by setting it to only blank characters.

When an output is selected, a few options are enabled. Using the check box, a default value for the output can set to either low or high. This means that the output will be set to this value when a program is not running. If the check box is not checked, the output will preserve its current state after a program ends. It is also possible to specify whether an output can be controlled on the I/O tab (by either programmers, or both operators and programmers) or if it is only robot programs that may alter the output value.

11.9 Installation → Safety

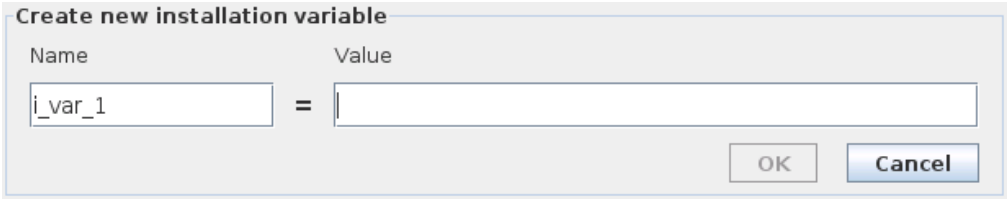
See chapter 14.

11.10 Installation → Variables



Variables created here are called installation variables and can be used just like normal program variables. Installation variables are special because they keep their value even if a program is stopped and then started 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, so it is possible to use the same variable in multiple programs.



Pressing `Create New` will bring 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 clicked if the new name is unused in this installation.

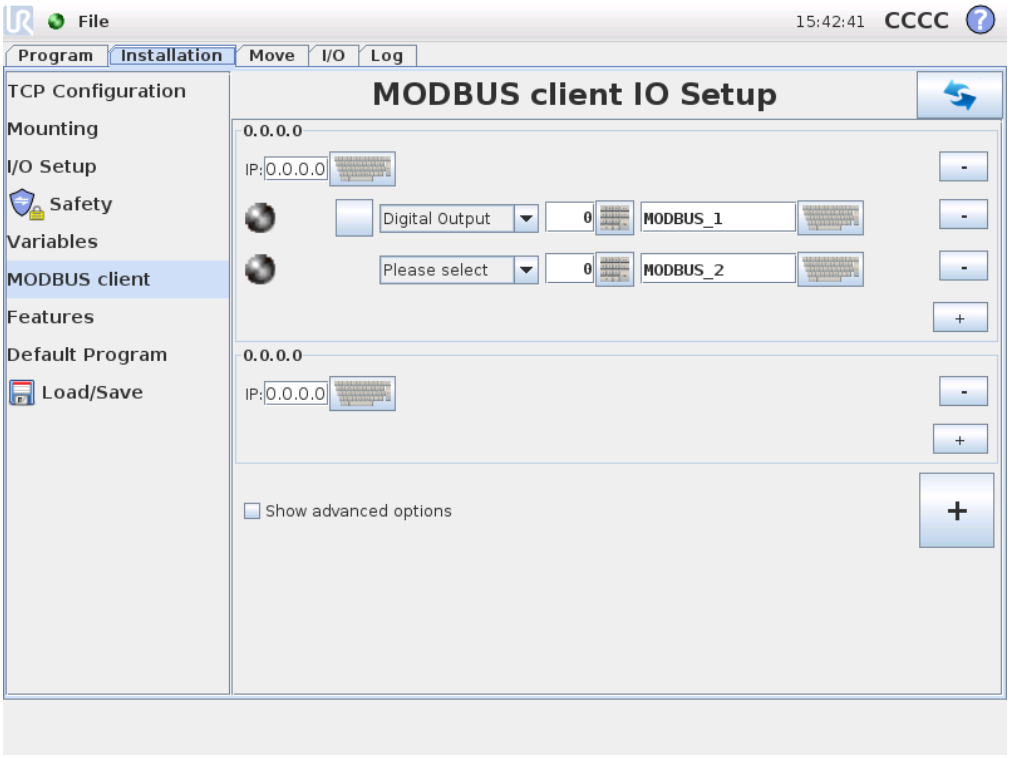
It is possible to change the value of an installation variable by highlighting the variable in the list and then clicking on `Edit Value`.

To delete a variable, select it in the list, then click `Delete`.

After configuring the installation variables, the installation itself must be saved to keep this configuration, see 11.5. The installation variables and their values are also 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 two options to resolve the issue: either use the installation variables of the same name instead of the program variable or have the conflicting variables renamed automatically.

11.11 Installation → 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.

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.

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.

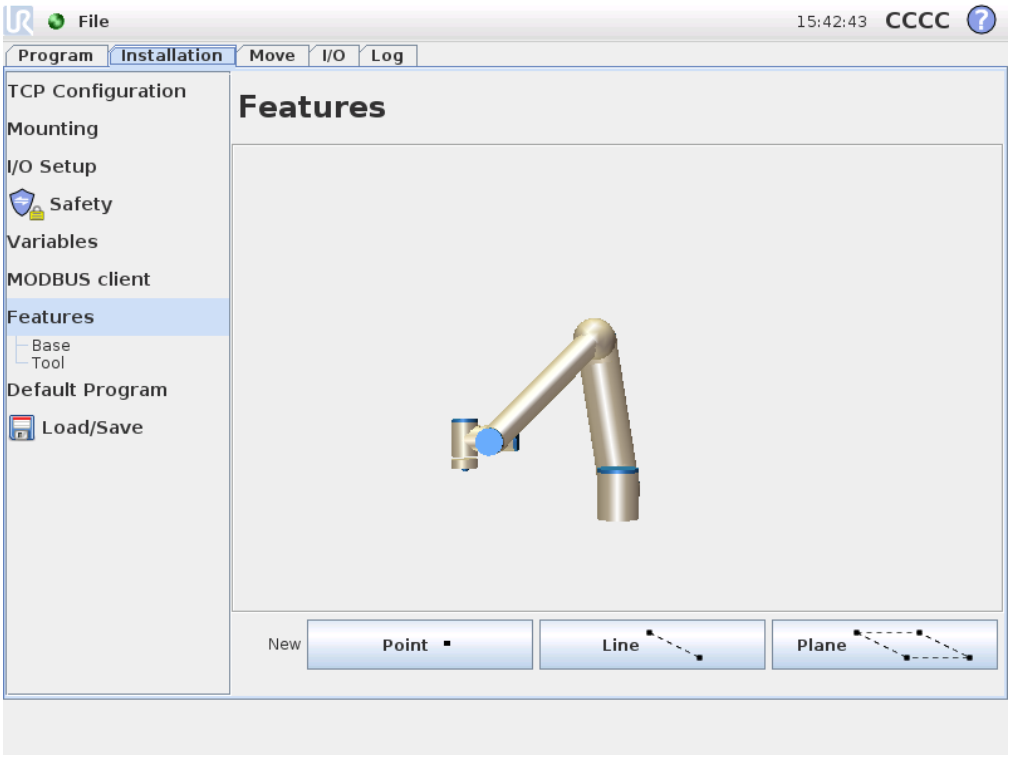
Show Advanced Options

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.
- **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.

11.12 Installation → Features



Customers that buy industrial robots generally want to be able to control or manipulate a robot arm, and to program the robot arm, relative to various objects and boundaries in the surroundings of the robot arm, such as machines, objects or blanks, fixtures, conveyers, pallets or vision systems. Traditionally, this is done by defining "frames" (coordinate systems) that relate the internal coordinate system of the robot arm (the base coordinate system) to the relevant object's coordinate system. Reference can both be made to "tool coordinates" and to "base coordinates" of the robot arm.

A problem with such frames is that a certain level of mathematical knowledge is required to be able to define such coordinate systems and also that it takes a considerable ammount of time to do this, even for a person skilled in the art of robot programming and installation. Often this task involves the calculation of 4x4 matrices. Particularly, the representation of orientation is complicated for a person that lacks the required experience to understand this problem.

Questions often asked by customers are for instance:

- Will it be possible to move the robot 4 cm away from the claw of my computerised numerically controlled (CNC) machine?
- Is it possible to rotate the tool of the robot 45 degrees relative to the table?
- Can we make the robot arm move vertically downwards with the object, let the object loose, and then move the robot arm vertically upward again?

The meaning of such and similar questions is very straightforward to an average customer who intends to use a robot arm for instance at various stations in a production plant, and it may seem annoying and incomprehensible to the customer to be told that there may not be a simple answer to such *relevant* questions. There are several complicated reasons for this being the case, and in order to address these problems, Universal Robots has developed unique and simple ways for a customer to specify the location of various objects relative to the robot arm. Within a few steps, it is therefore possible to do exactly what was asked for in the above questions.

Rename

This button makes it possible to rename a feature.

Delete

This button deletes the selected feature and, if any, all sub-features.

Show Axes

Choose whether the coordinate axes of the selected feature shall be visible on the 3D graphics. The choice applies on this screen and on the Move screen.

Joggable

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

Variable

Select whether the selected feature can be used as a variable. If this option is selected a variable named the name of the feature suceeded by "_var" will then be available when editing robot programs, and this variable can be assigned a new value in a program, which can then be used to control waypoints that depend on the value of a feature.

Set or Change Position

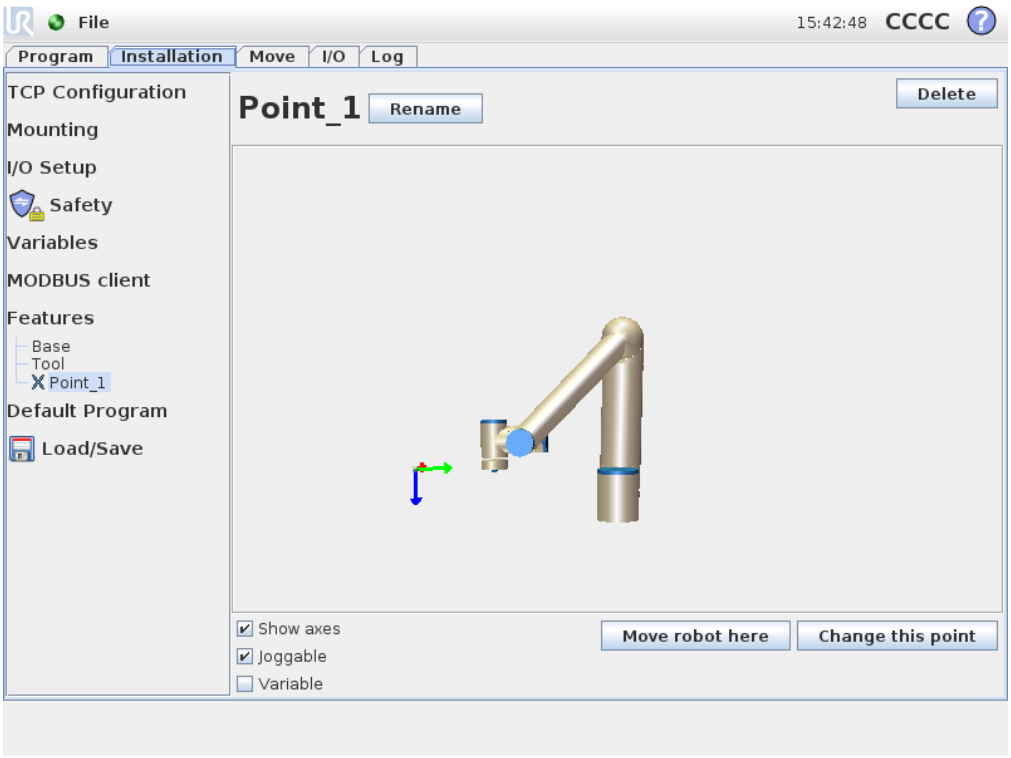
Use this button to set or change the selected feature. The Move screen will appear and a new position of the feature can be set.

Move Robot to Feature

Pressing this button will 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, except for a 180 degree rotation about the x-axis.

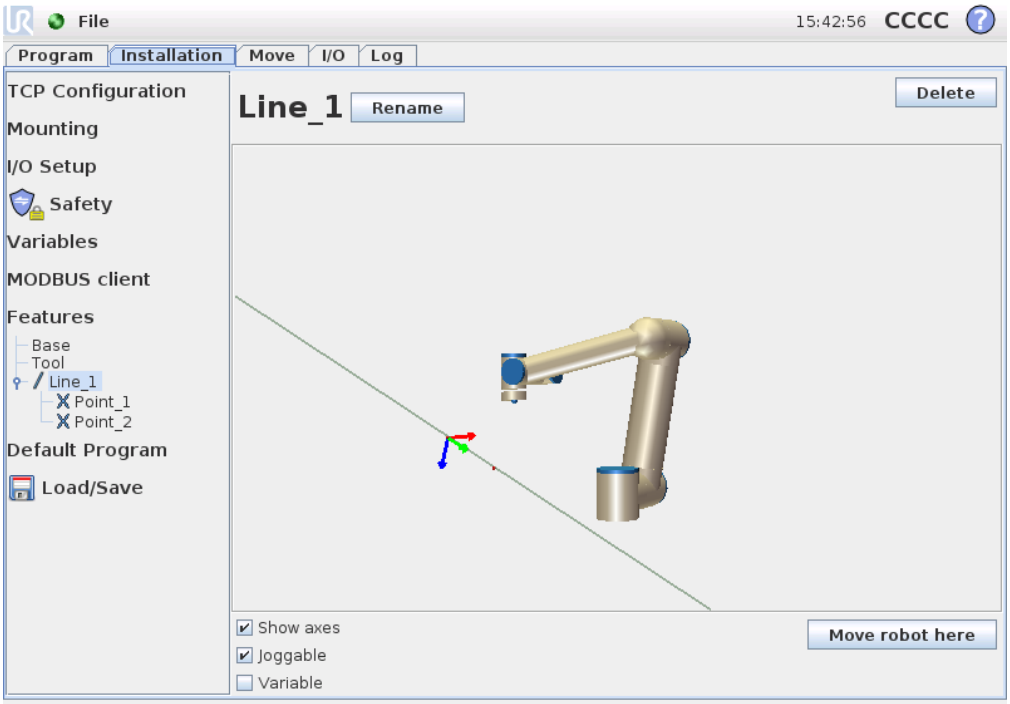
Add Point

Push this button to add a point feature to the installation. The position of a point feature is defined as the position of the TCP at that point. The orientation of the point feature is the same as the TCP orientation, except that the feature coordinate system is rotated 180 degrees about its x-axis. This makes the z-axis of the point feature directed opposite than that of the TCP at that point.



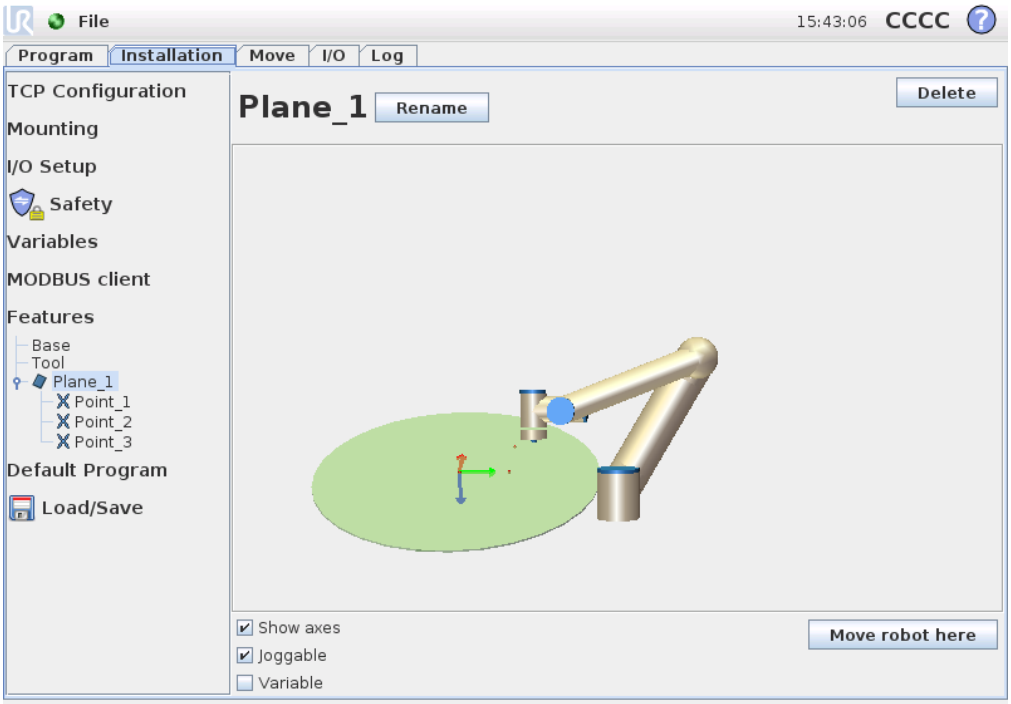
Add Line

Push this button to add a line feature to the installation. A line is defined as an axis between two point features. This axis, directed from the first point towards the second point, will constitute the y-axis of the line coordinate system. The z-axis will be defined by the projection of the z-axis of the first sub point onto the plane perpendicular to the line. The position of the line coordinate system is the same as the position for the first sub point.

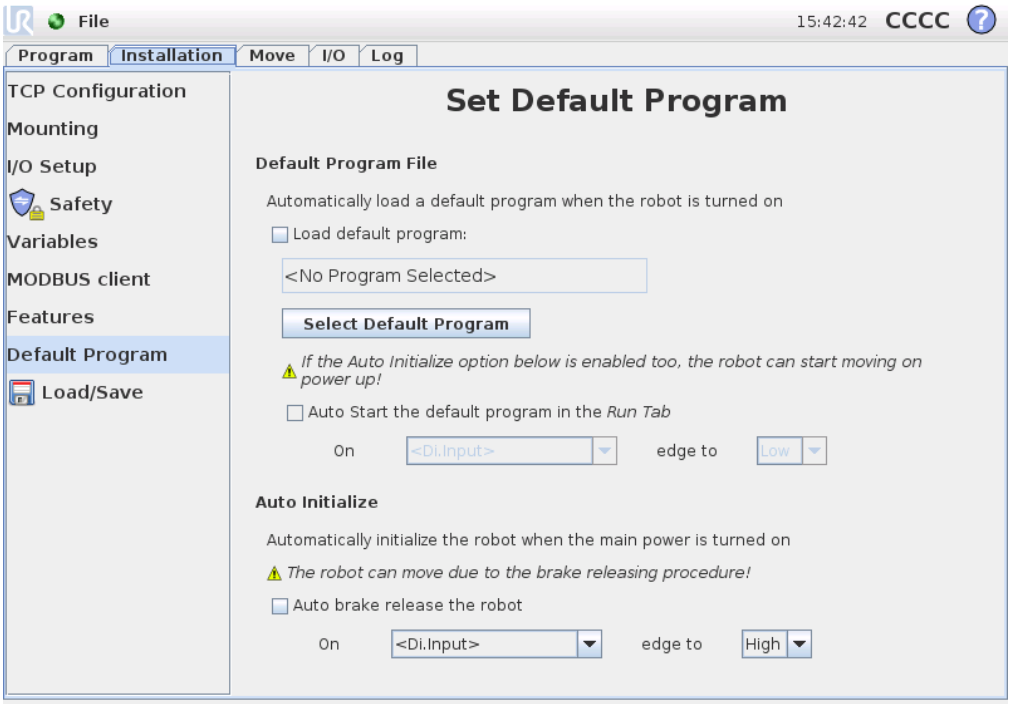


Add Plane

Push this button to add a plane feature to the installation. A plane is defined by three sub point features. The position of the coordinate system is the same as the position for the first sub point. The z-axis is the plane normal, and the y-axis is directed from the first point towards the second. The positive direction of the z-axis is set so that the angle between the z-axis of the plane and the z-axis of the first point is less than 180 degrees.



11.13 Installation → Default Program



This screen contains settings for automatically loading and starting a default program, and for auto initializing the robot arm on power up.

WARNING:
If auto load, auto start and auto initialize all three are enabled, the robot will start running the selected program as soon as the control box is powered up.

11.13.1 Loading a Default Program

A default program can be chosen to be loaded when the control box is powered up. Furthermore, the default program will also be auto loaded when the *Run Program* screen (see 9.3) is entered and no program is loaded.

11.13.2 Starting a Default Program

The default program can be 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 will be started automatically.

Note, on startup the current input signal level is undefined and chosing a transition that matches the signal level on startup will start the program immediately. Furthermore, leaving the *Run Program* screen or pressing the stop button in the *Dashboard* will disable the auto starting feature until the run button has been pressed again.

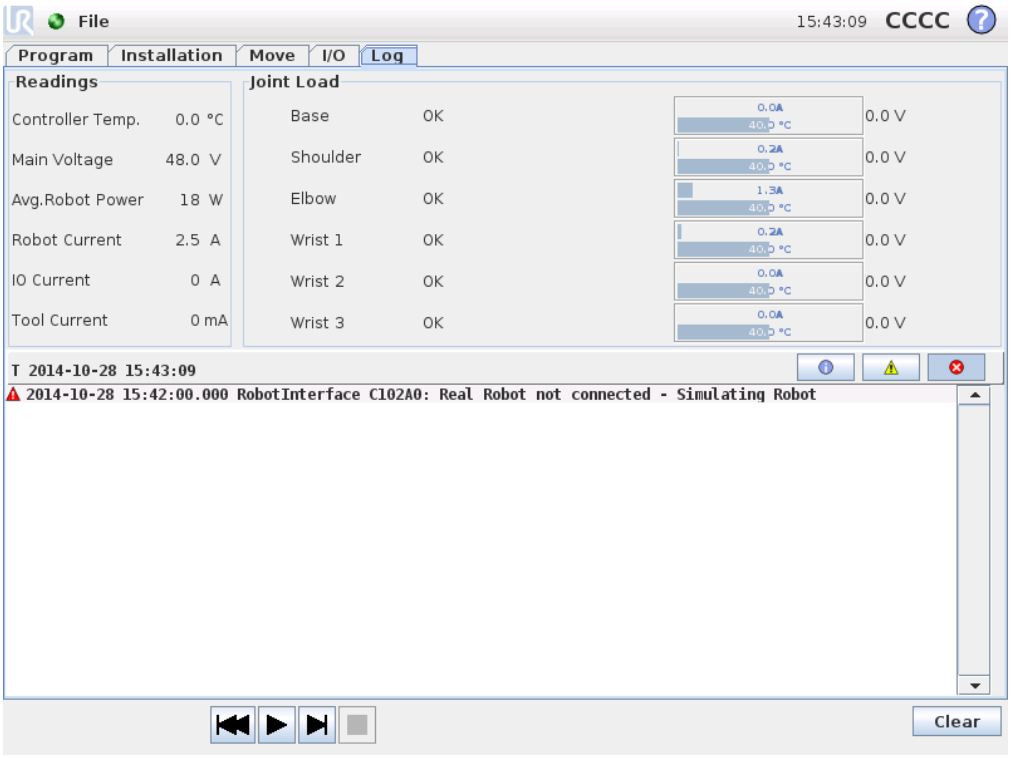
11.13.3 Auto Initialization

The robot arm can be automatically initialized, for instance when the control box is powered up. On the specified external input signal edge transition, the robot arm will be completely initialized, irrespective of the visible screen.

The final stage of initialization is *brake release*. When the robot is releasing the brakes, it moves a bit and makes a sound. Furthermore, the brakes cannot be automatically released if the configured mounting does not match the mounting detected based on sensor data. In this case, the robot needs to be initialized manually in the initialization screen (see 9.4).

Note, on startup the current input signal level is undefined and chosing a transition that matches the signal level on startup will initialize the robot arm immediately.

11.14 Log Tab



Robot Health The top half of the screen displays the health of the robot arm and control box. The left part shows information related to the control box of the robot, while the right part shows information about each robot joint. Each robot joint shows information for temperaure of the motor and electronics, the load of the joint and the voltage at the joint.

Robot Log On the bottom half of the screen log messages are shown. The first column categorizes the severity of the log entry. The second column shows the time of arrival of the message. The next column shows the sender of the message. While the last column shows the message itself. Messages can be filtered by selecting the toggle buttons which correspond to the severity. The figure above now shows that errors will be displayed while information and warning messages will be filtered. Some log messages are designed to provide more information, this can be accessed by selecting the log entry.

11.15 Load Screen

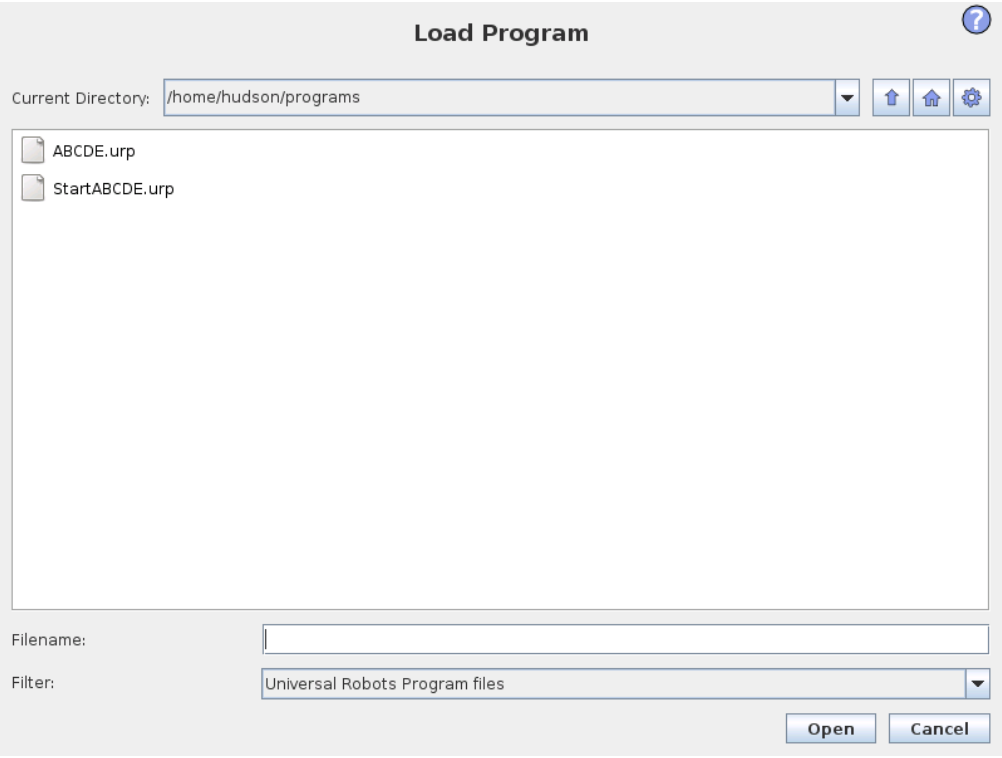
On this screen you choose which program to load. There are two versions of this screen: one that is to be used when you just want to load a program and execute it, and one that is used when you want to actually edit a program.

NOTE:
Running a program from a USB drive is not recommended. To run a program stored on a USB drive, first load it and then save it in the local `programs` folder using the `Save As...` option in the `File` menu.

The main difference lies in which actions are available to the user. In the basic load screen, the user will only be able to access files - not modify or delete them. Furthermore, the user is not allowed to leave the directory structure that descends from the `programs` folder. The user can descend to a sub-directory, but he cannot get any higher than the `programs` folder.

Therefore, all programs should be placed in the programs folder and/or sub folders under the programs folder.

Screen layout



This image shows the actual load screen. It consists of the following important areas and buttons:

Path history The path history shows a list of the paths leading up to the present location. This means that all parent directories up to the root of the computer are shown. Here you will notice that you may not be able to access all the directories above the programs folder.

By selecting a folder name in the list, the load dialog changes to that directory and displays it in the file selection area 11.15.

File selection area In this area of the dialog the contents of the actual area is present. It gives the user the option to select a file by single clicking on its name or to open the file by double clicking on its name.

In the case that the user double-clicks on a directory, the dialog descends into this folder and presents its contents.

File filter By using the file filter, one can limit the files shown to include the type of files that one wishes. By selecting "Backup Files" the file selection area will display the latest 10 saved versions of each program, where `.old0` is the newest and `.old9` is the oldest.

File field Here the currently selected file is shown. The user has the option to manually enter the file name of a file by clicking on the keyboard icon to the right of the field. This will cause an on-screen keyboard to pop up where the user can enter the file name directly on the screen.

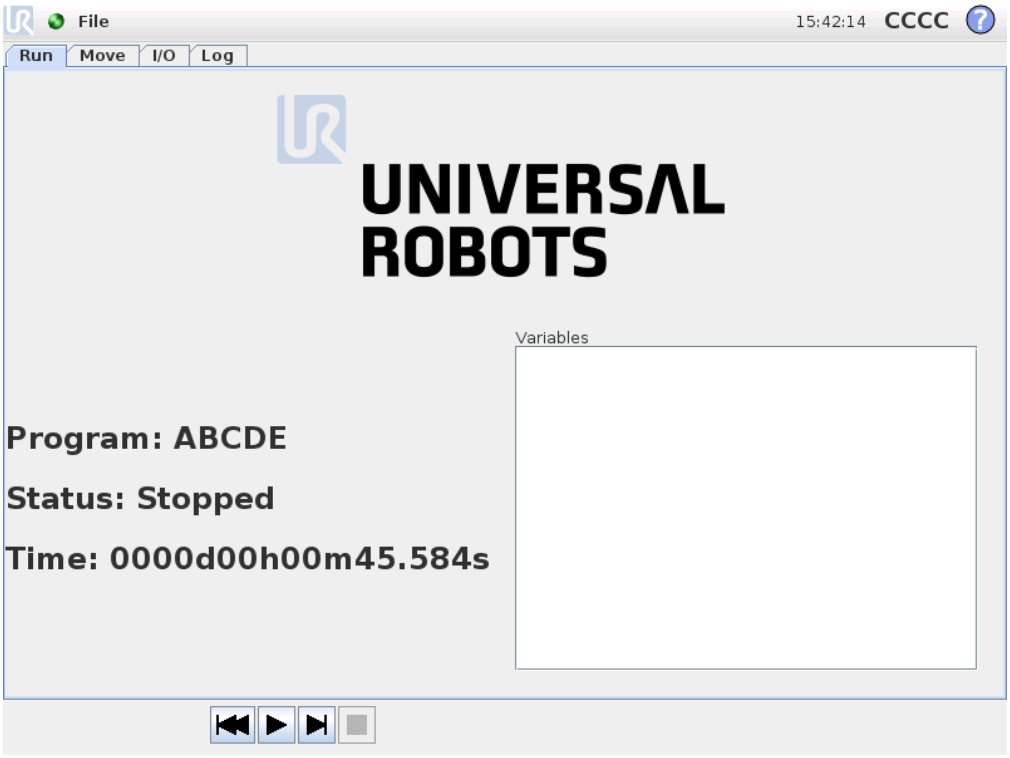
Open button Clicking on the Open button, will open the currently selected file and return to the previous screen.

Cancel button Clicking on the Cancel button will abort the current loading process and cause the screen to switch to the previous image.

Action buttons A series of buttons gives the user the ability to perform some of the actions that normally would be accessible by right-clicking on a file name in a conventional file dialog. Added to this is the ability to move up in the directory structure and directly to the program folder.

- Parent: Move up in the directory structure. The button will not be enabled in two cases: when the current directory is the top directory or if the screen is in the limited mode and the current directory is the program folder.
- Go to program folder: Go home
- Actions: Actions such as create directory, delete file etc.

11.16 Run Tab

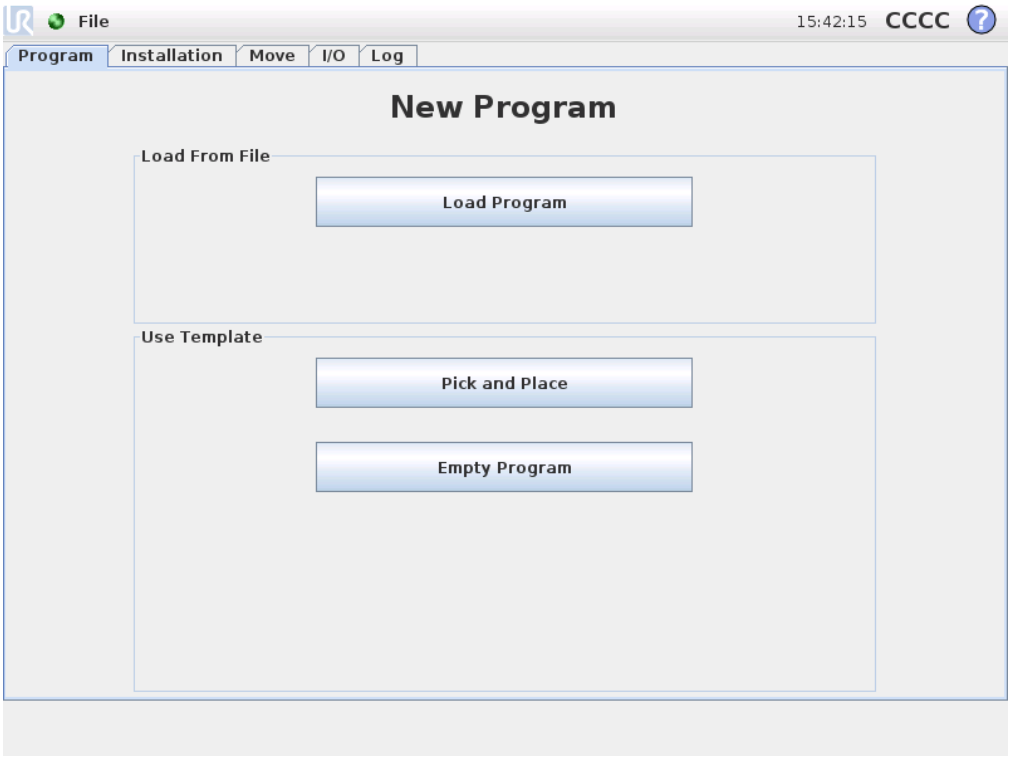


This tab provides a very simple way of operating the robot arm and control box, with as few buttons and options as possible. This can be usefully combined with password protecting the programming part of PolyScope (see 13.3), to make the robot into a tool that can run exclusively pre-written programs.

Furthermore, in this tab a default program can be automatically loaded and started based on an external input signal edge transition (see 11.13). 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.

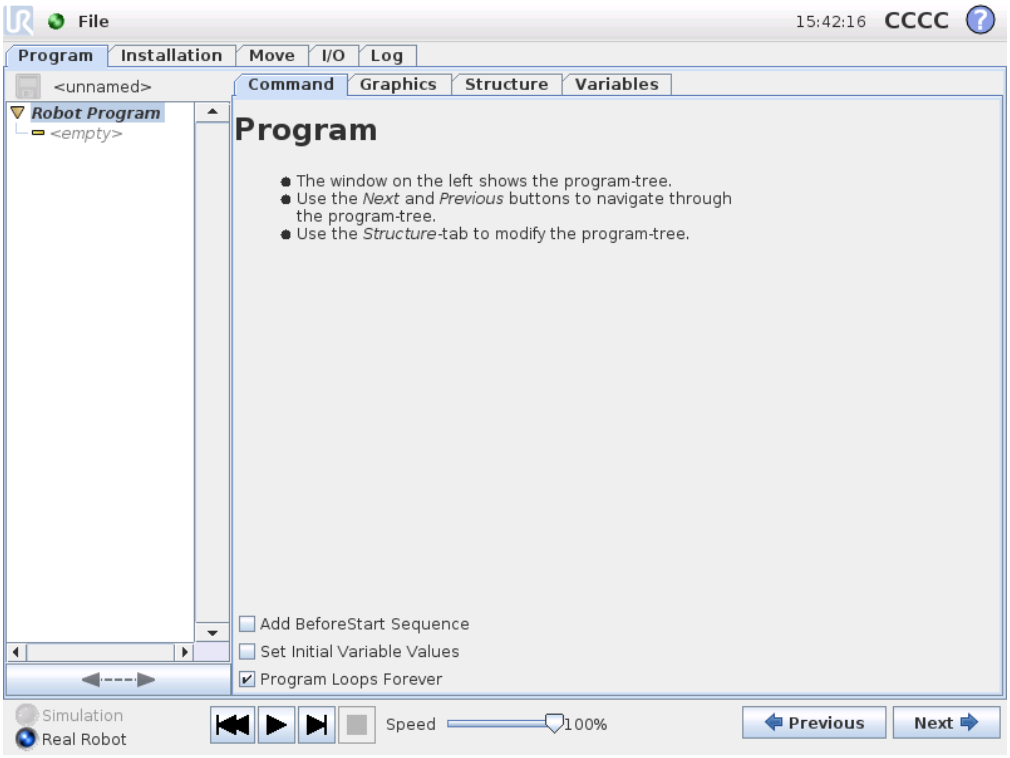
12 Programming

12.1 New Program



A new robot program can start from either a *template* or from an existing (saved) robot program. A *template* can provide the overall program structure, so only the details of the program need to be filled in.

12.2 Program Tab



The program tab shows the current program being edited.

The *program tree* on the left side of the screen displays the program as a list of commands, while the area on the right side of the screen displays information relating to the current command. The current command is selected by clicking the command list, or by using the `Previous` and `Next` buttons on the bottom right of the screen. Commands can be inserted or removed using the `Structure` tab, described in 12.29. The program name is shown directly above the command list, with a small disk icon that can be clicked to quickly save the program.

The lowest part of the screen is the *Dashboard*. The *Dashboard* features a set of buttons similar to an old-fashioned tape recorder, from which programs can be started and stopped, single-stepped and restarted. The *speed slider* allows you to adjust the program speed at any time, which directly affects the speed at which the robot arm moves. Additionally, the *speed slider* shows in real time the relative speed at which the robot arm moves taking into account the safety settings. The indicated percentage is the maximum achievable speed for the running program without faulting the safety system.

To the left of the *Dashboard* the `Simulation` and `Real Robot` buttons toggle between running the program in a simulation, or running it on the real robot. When running in simulation, the robot arm does not move and thus cannot damage itself or any nearby

equipment in collisions. Use simulation to test programs if unsure about what the robot arm will do.

DANGER:

1. Make sure to stay outside the robot workspace when the `Play` button is pressed. The movement you programmed may be different than expected.
2. Make sure to stay outside the robot workspace when the `Step` button is pressed. The function of the `Step` button can be difficult to understand. Only use it when it is absolutely necessary.
3. Make sure to always test your program by reducing the speed with the speed slider. Logic programming errors made by the integrator might cause unexpected movements of the robot arm.

While the program is being written, the resulting motion of the robot arm is illustrated using a 3D drawing on the `Graphics` tab, described in 12.28.

Next to each program command is a small icon, which is either red, yellow or green. A red icon means that there is an error in that command, yellow means that the command is not finished, and green means that all is OK. A program can only be run when all commands are green.

12.3 Variables

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

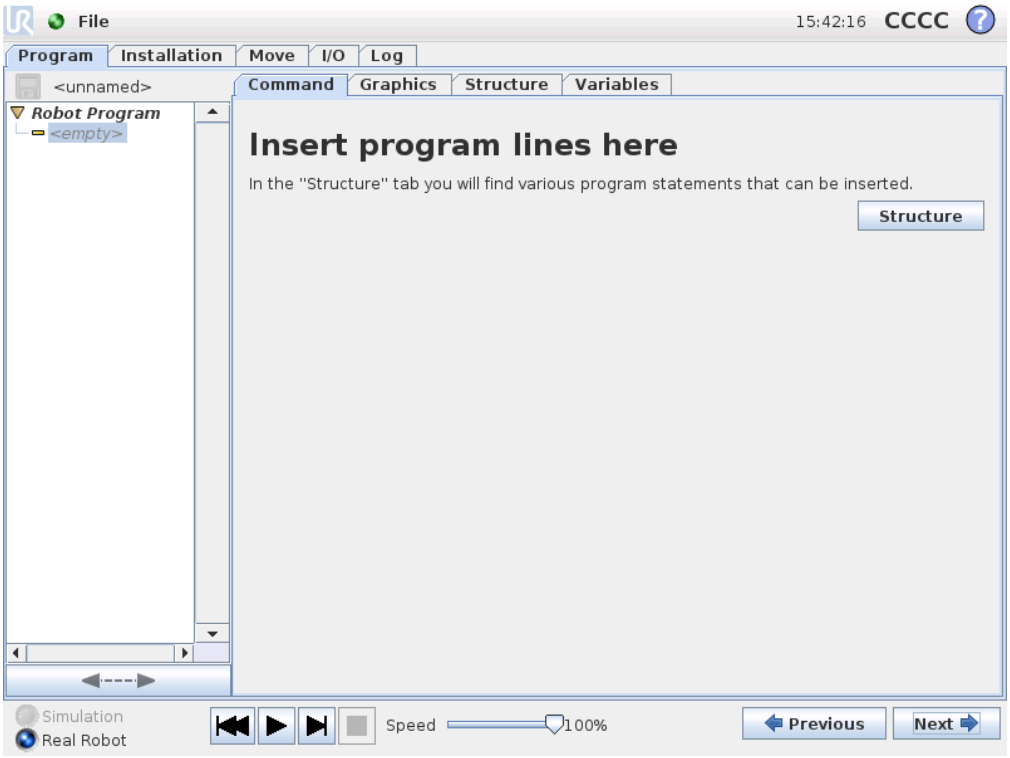
Installation variables: These can be used by multiple programs and their names and values are persisted together with the robot installation (see 11.10 for further details);

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

The following variable types are available:

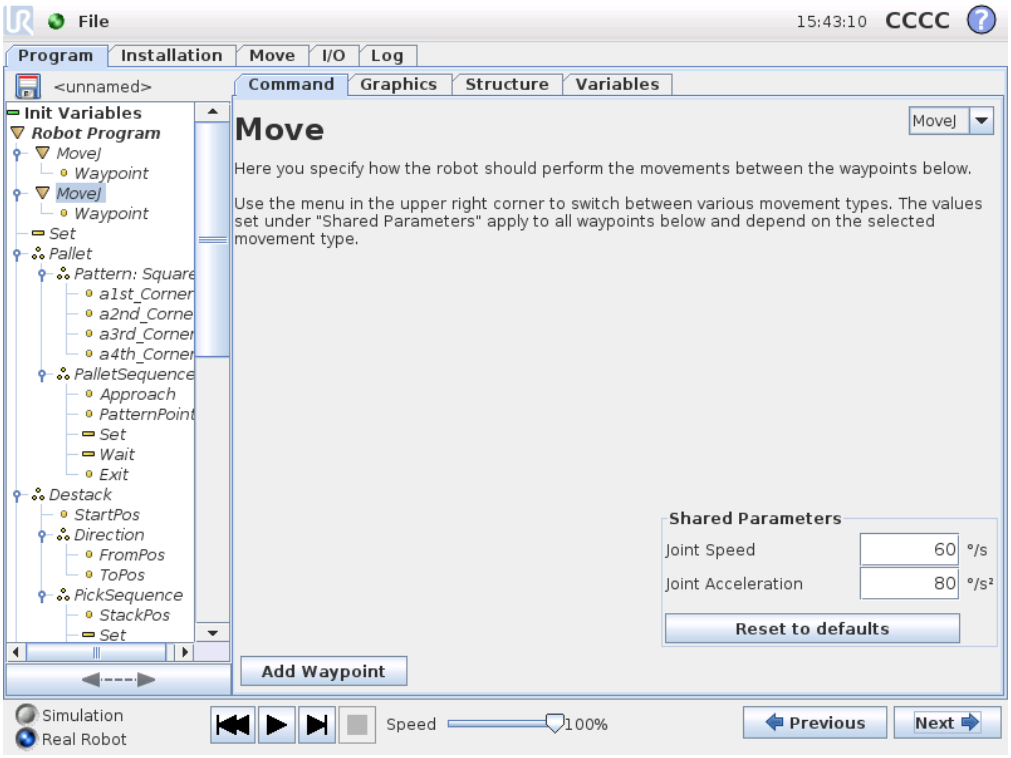
bool	A boolean variable whose value is either `True` or `False`.
int	A whole number in the range from -32768 to 32767.
float	A floating point number (decimal).
string	A sequence of characters.
pose	A vector describing the location and orientation in Cartesian space. It is a combination of a position 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.

12.4 Command: Empty



Program commands need to be inserted here. Press the `Structure` button to go to the structure tab, where the various selectable program lines can be found. A program cannot run before all lines are specified and defined.

12.5 Command: 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

It is possible to select one of three types of movements: *MoveJ*, *MoveL* and *MoveP* each explained below.

- **moveJ** will make movements that are calculated in the *joint space* of the robot arm. Each joint is controlled to reach the desired end location 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 to use for the movement calculations, 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 favorable choice.
- **moveL** will make the tool move 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. The selected feature will determine in which feature space the

tool positions of the waypoints are represented in. Of specific interest concerning feature spaces are variable features and variable waypoints. Variable features can be used when the tool position of a waypoint need to be determined by the actual value of the variable feature when the robot program runs.

- **moveP** will move 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.

  A **Circle Move** can be added to a moveP command, consisting of two waypoints: the first one specifying a via point on the circular arc, and the second one being the endpoint of the movement. The robot will start the circle movement from its current position, and then move through the two specified waypoints. The orientation change of the tool through the circle move is determined only by the starting orientation and the orientation at the endpoint, so the orientation of the via point does not influence the circle move. A Circle Move must always be preceeded by a waypoint under the same moveP.

Feature selection

For *MoveL* and *MoveP* it is possible to select in which feature space the waypoints under the Move command should be represented when specifying these waypoints. 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.

- **Fixed feature**: If a fixed feature, such as e.g. *Base*, is selected this will not have any effect on *Fixed* and Relative waypoints. The behavior for *Variable* waypoints is described below.
- **Variable feature**: If any of the features in the currently loaded installation are selected to be variable, these corresponding variables will also be selectable in the feature selection menu. If a feature variable (named by the name of the feature and proceeded by "_var") is selected, the robot arm movements (except to *Relative* waypoints) will depend on the actual value of the variable when the program is running. The initial value of a feature variable is the value of the actual feature. This means that the movements will only change if the feature variable is actively changed by the robot program.
- **Variable waypoint**: When the robot arm moves to a variable waypoint, the tool target position will always be calculated as the coordinates of the variable in the space of the selected feature. Therefore, the robot arm movement for a variable waypoint will always change if another feature is selected.

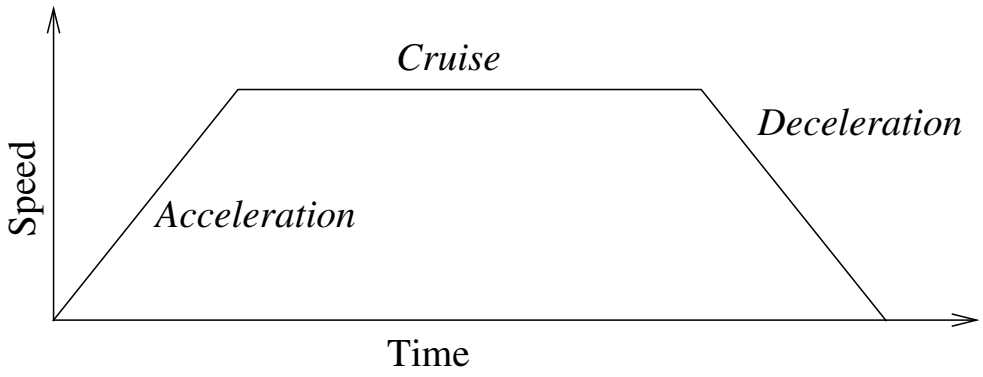
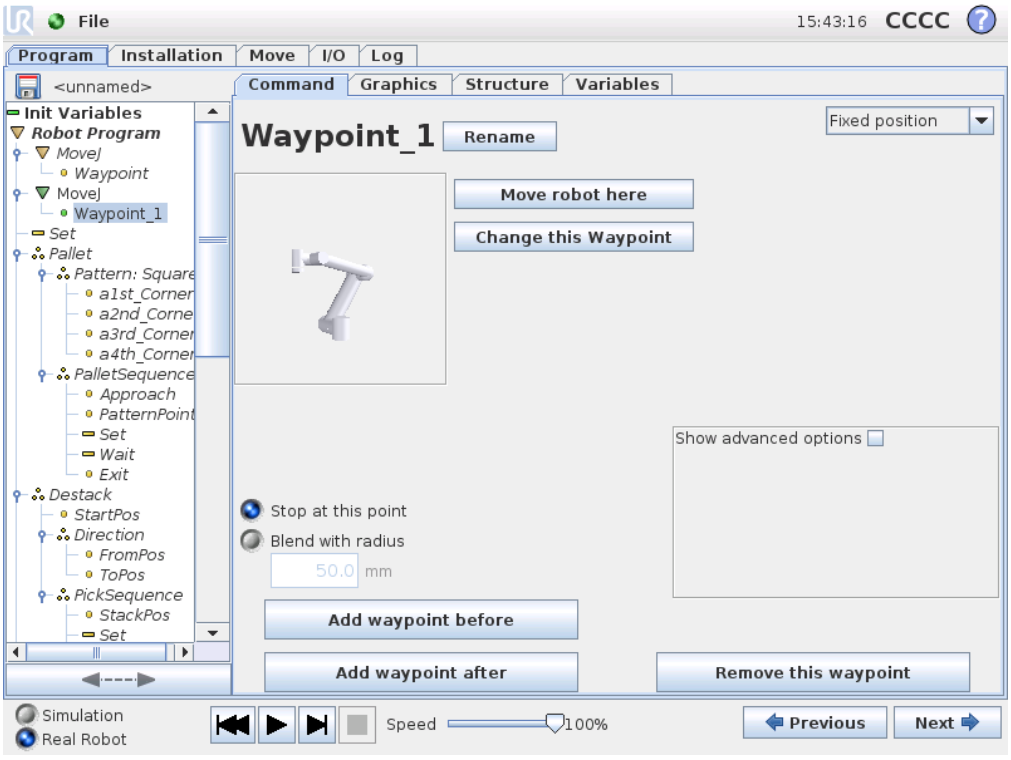


Figure 12.1: Speed profile for a motion. The curve is divided into three segments: *acceleration*, *cruise* and *deceleration*. The level of the *cruise* phase is given by the speed setting of the motion, while the steepness of the *acceleration* and *deceleration* phases is given by the acceleration parameter.

The settings of the Shared Parameters of a Move command apply to the path from the robot arm's current position 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.

12.6 Command: 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 given by physically

moving the robot arm to the position.

12.7 Setting the waypoint

Press this button to enter the Move screen where you can specify the robot arm's position for this waypoint. If the waypoint is placed under a Move command in linear space (`moveL` or `moveP`), there need to be a valid feature selected at that Move command, in order for this button to be pressable.

Waypoint names

Waypoint names can be changed. Two waypoints with the same name is always the same waypoint. Waypoints are numbered as they are specified.

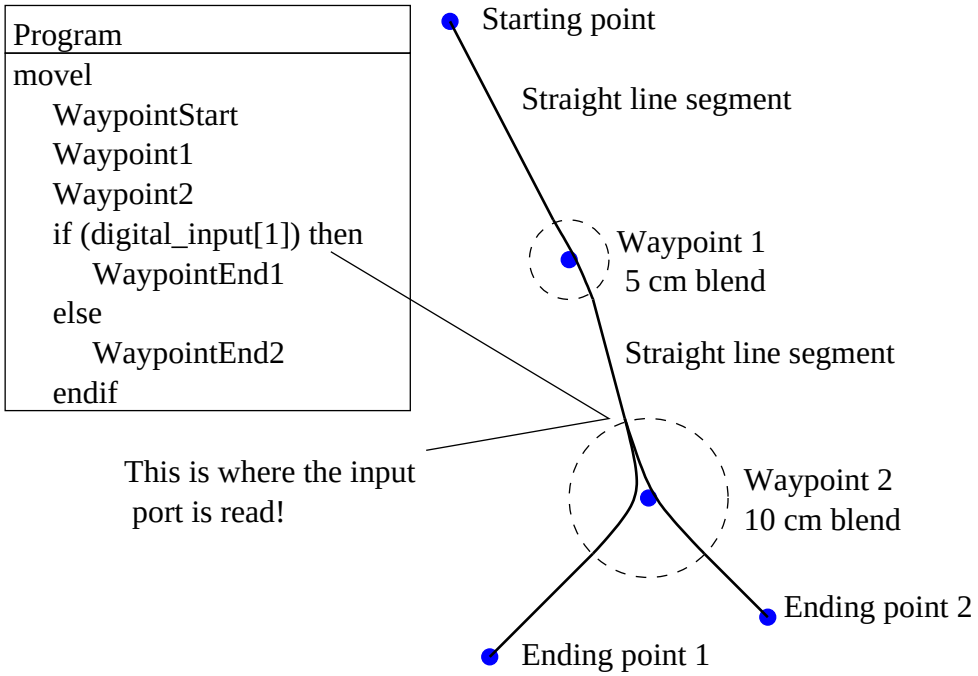
Blend radius

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 a blend radius for a previous or following waypont. A stop point is a waypoint with a blend radius of $0.0mm$.

Note on I/O Timing

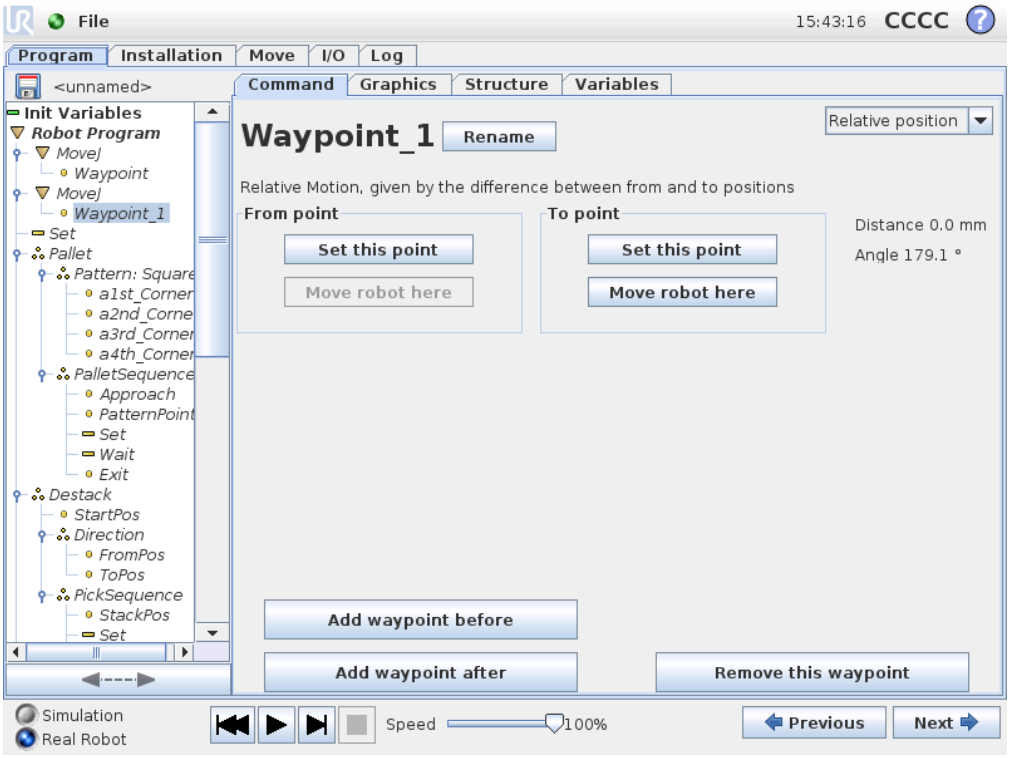
If a waypoint is a stop point with an I/O command as the next command, the I/O command is executed when the robot arm stops at the waypoint. However, if the waypoint has a blend radius, the following I/O command is executed when the robot arm enters the blend.

Example



A small example in which a robot program moves the tool from a starting position to one of two ending positions, depending on the state of `digital_input[1]`. Notice that the tool trajectory (thick black line) moves in straight lines outside the blend areas (dashed circles), while the tool trajectory deviates from the straight line path inside the blend areas. Also notice that the state of the `digital_input[1]` sensor is read just as the robot arm is about to enter the blend area around `Waypoint 2`, even though the `if...then` command is after `Waypoint 2` in the program sequence. This is somewhat counter-intuitive, but is necessary to select the right blend path.

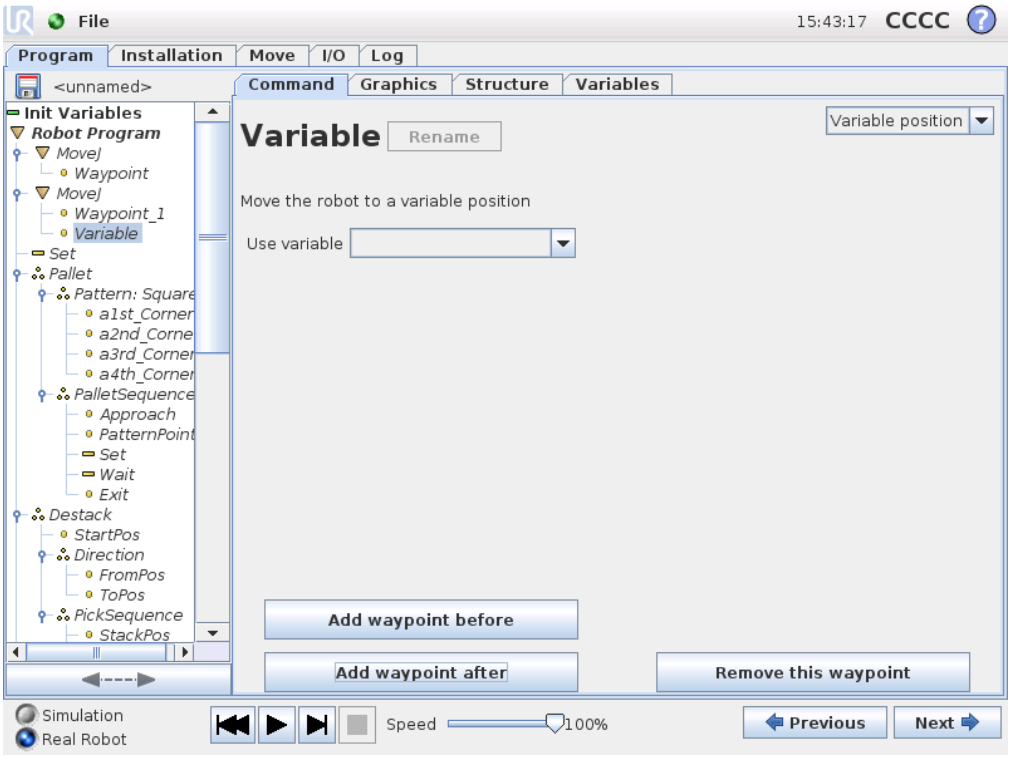
12.8 Command: Relative Waypoint



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 that 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.

12.9 Command: Variable Waypoint

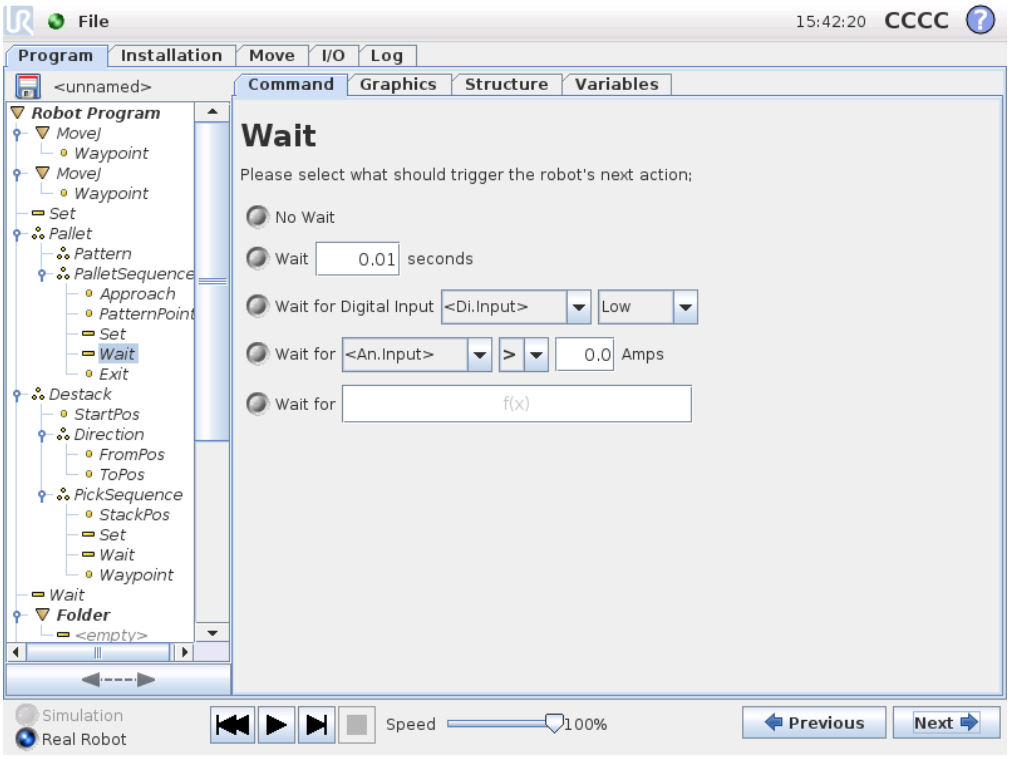


A waypoint with the position given by a variable, in this case `calculated_pos`. The variable has to be a *pose* such as `var=p[0.5,0.0,0.0,3.14,0.0,0.0]`. The first three are *x,y,z* and the last three are the orientation given as a *rotation vector* given by the vector *rx,ry,rz*. The length of the 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. The robot arm always moves linearly to a variable waypoint.

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

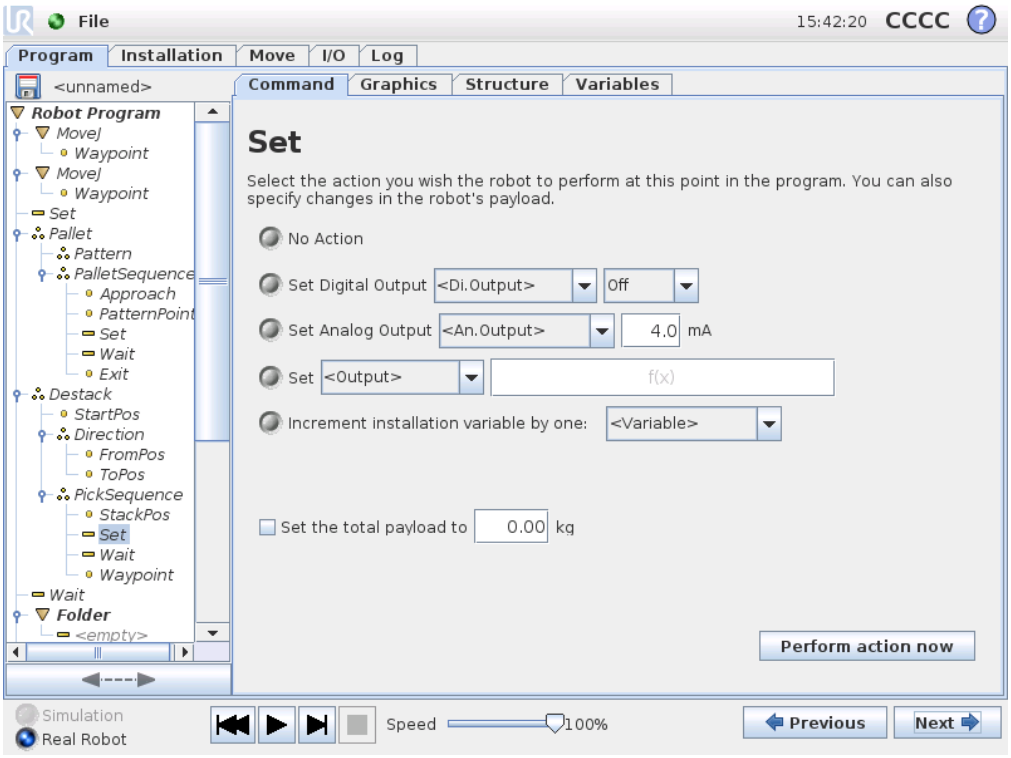
```
var_1=p[0,0,0.02,0,0,0]
Movel
  Waypoint_1 (varibale position):
    Use variable=var_1, Feature=Tool
```

12.10 Command: Wait



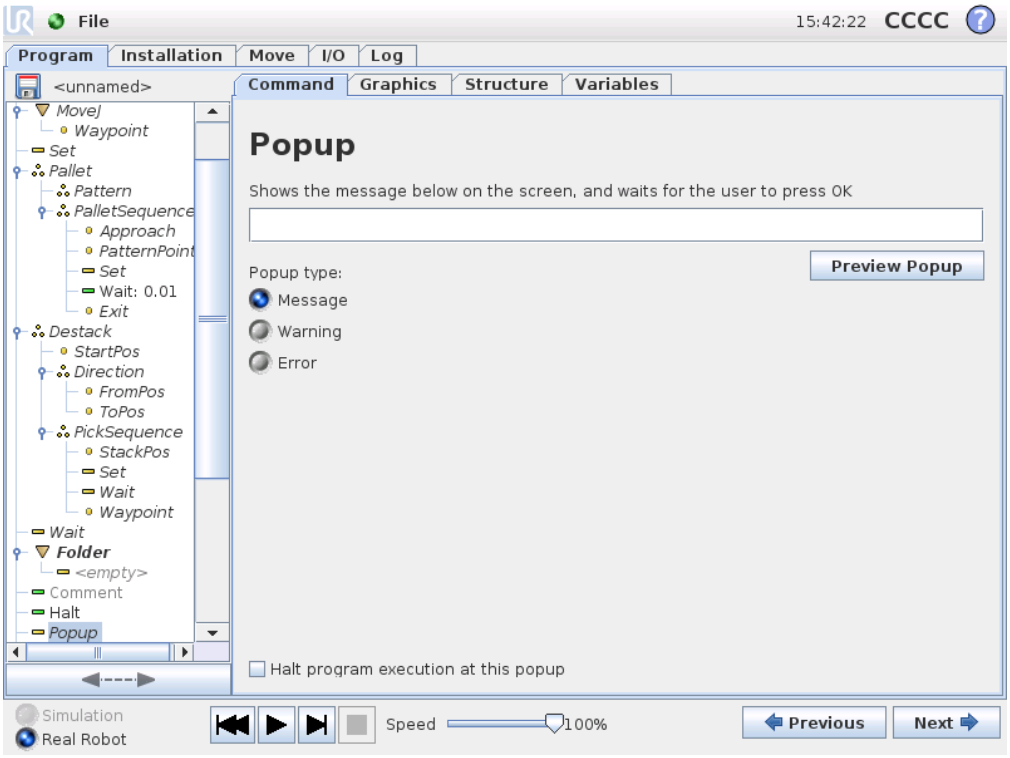
Waits for a given amount of time or for an I/O signal.

12.11 Command: Set



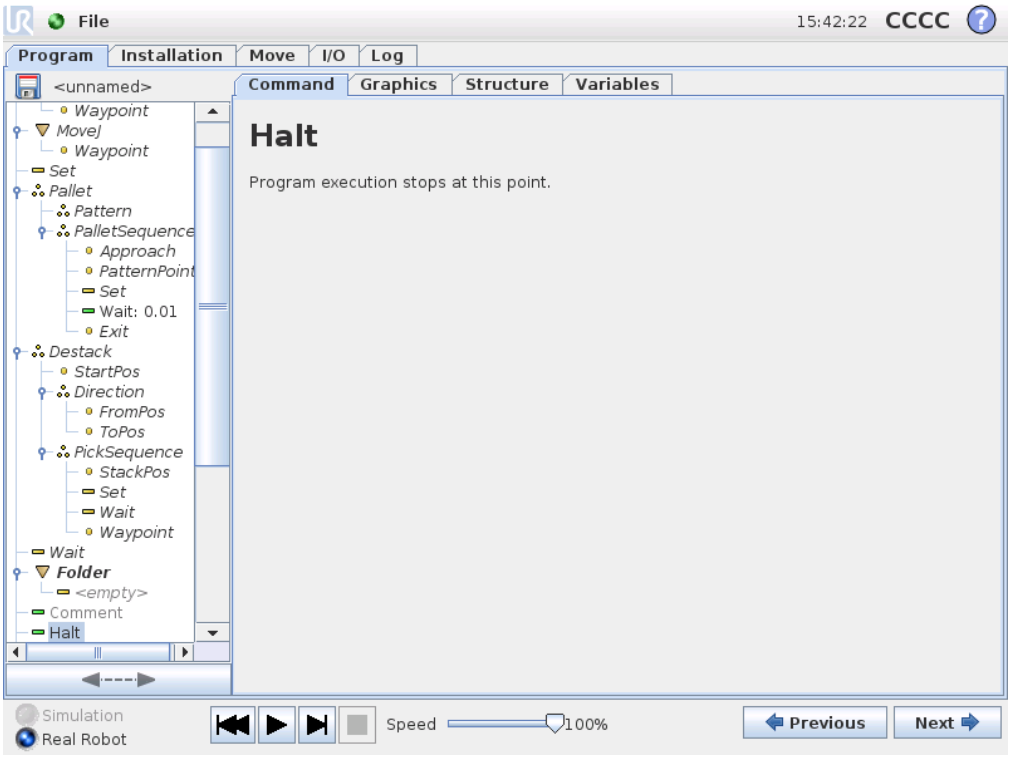
Sets either digital or analog outputs to a given value. Can also be used to set the payload of the robot arm, for example the weight that is picked up as a consequence of this action. Adjusting the weight can be neccesary to prevent the robot from protective stopping unexpectedly, when the weight at the tool is different from the expected one.

12.12 Command: Popup



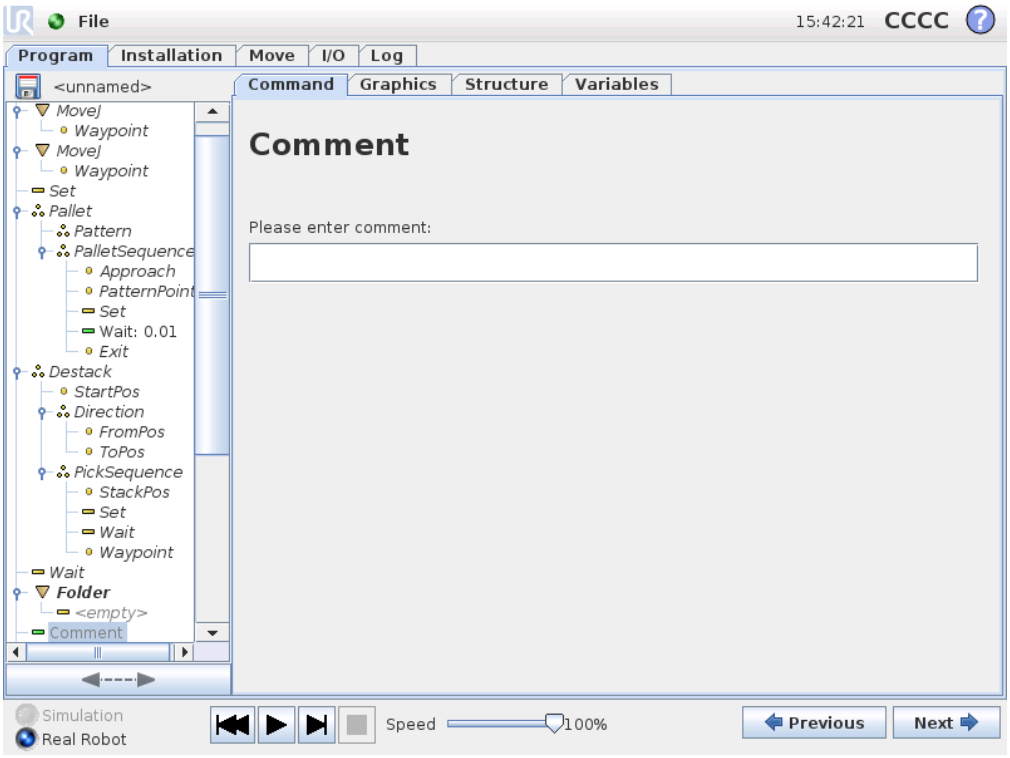
The popup is a message that appears on the screen when the program reaches this command. The style of the message can be selected, and the text itself can be given using the on-screen keyboard. The robot waits for the user/operator to press the "OK" button under the popup before continuing the program.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

12.13 Command: Halt



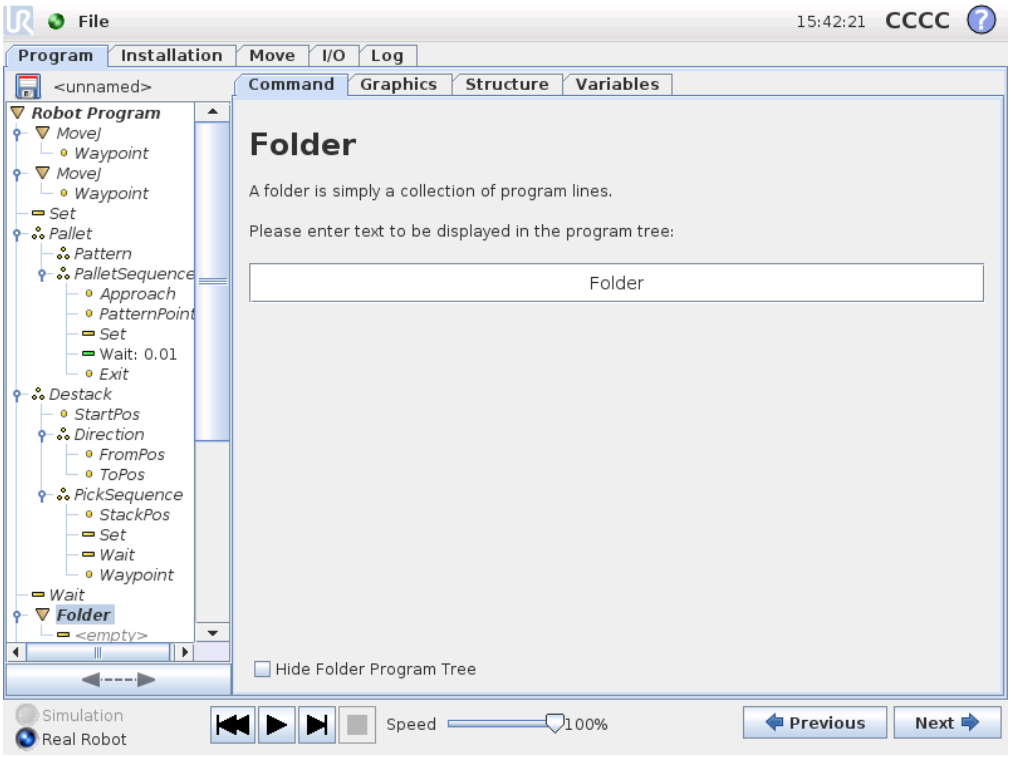
The program execution stops at this point.

12.14 Command: 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.

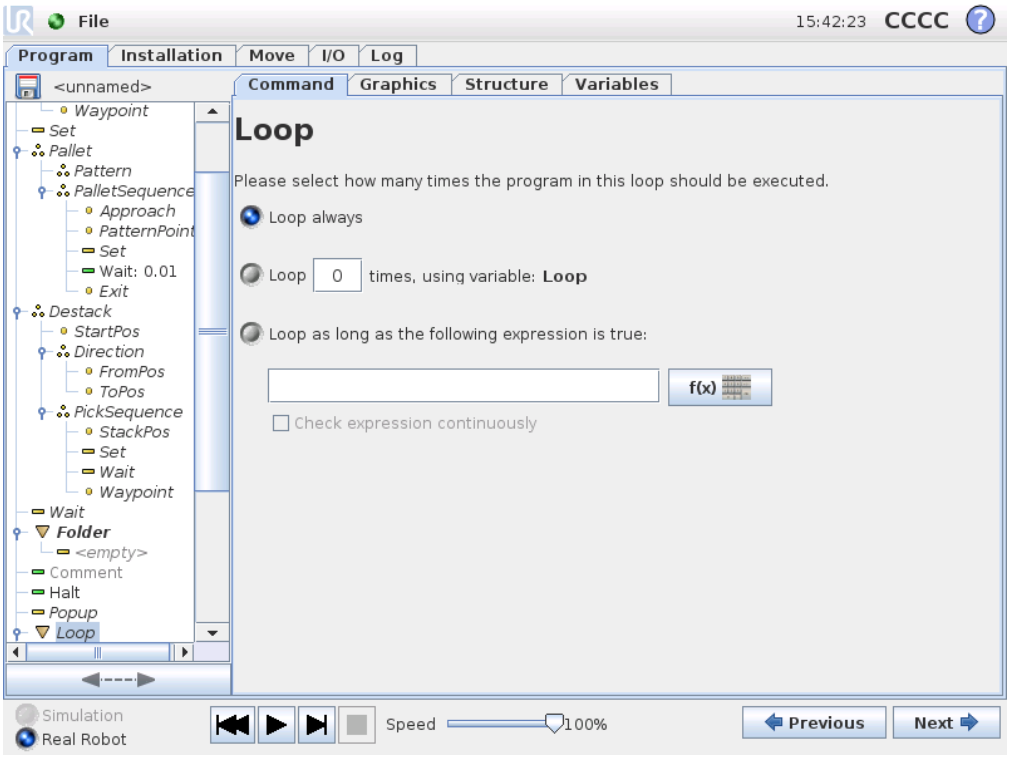
12.15 Command: Folder



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.

A folder does not in itself do anything.

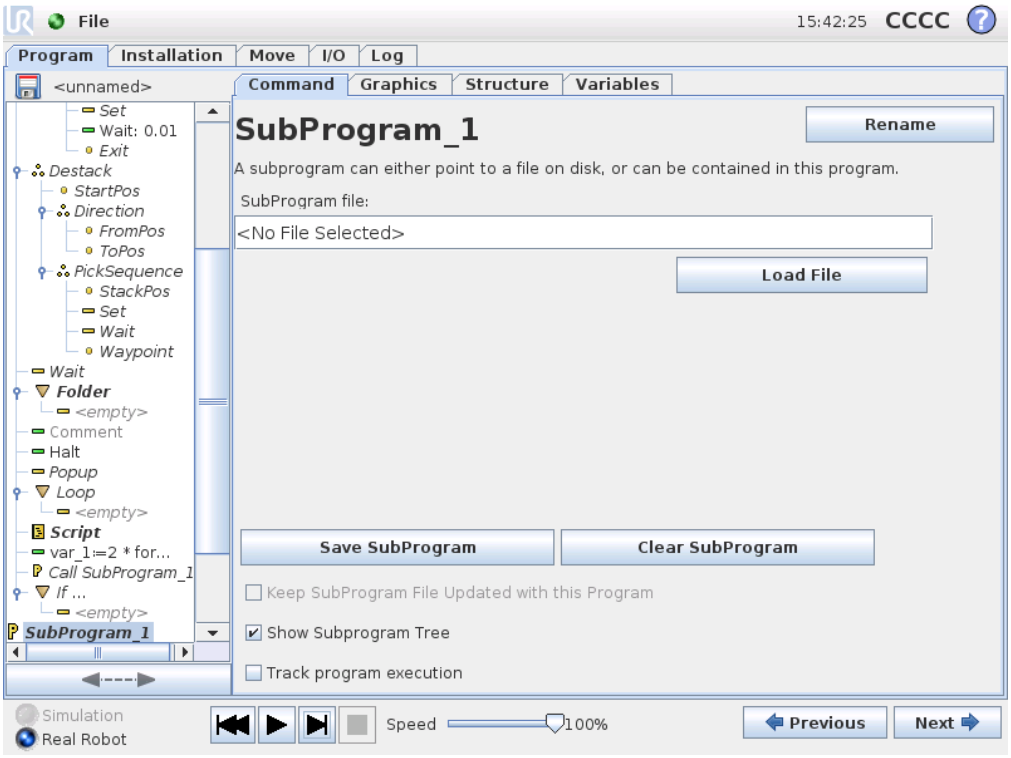
12.16 Command: Loop



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 $N - 1$.

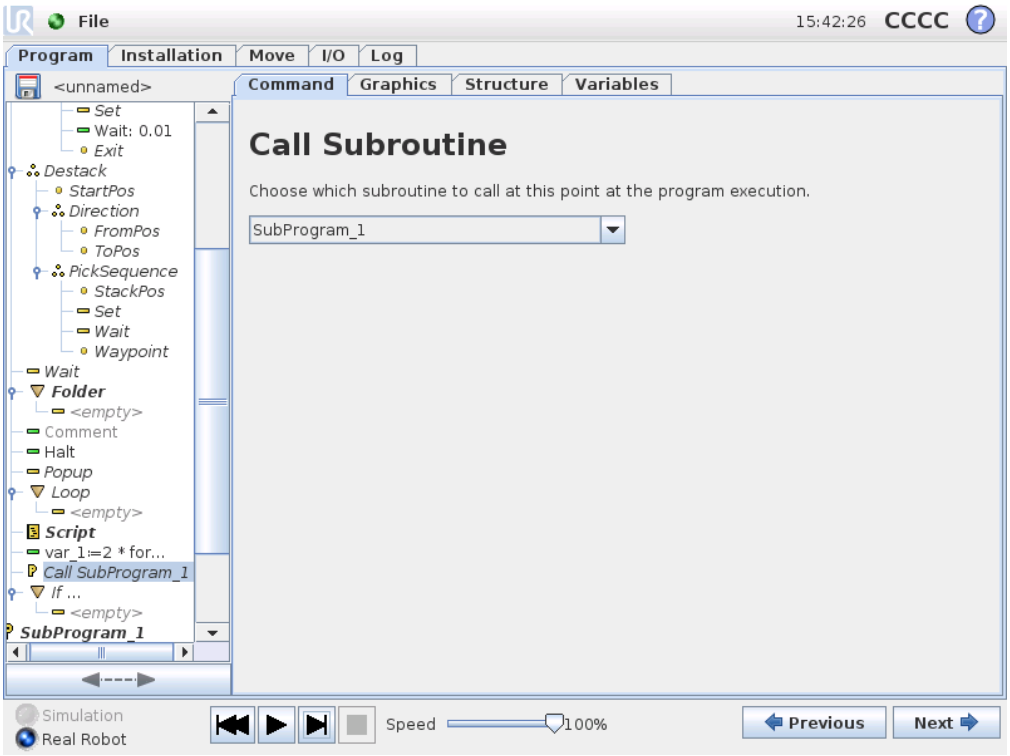
When looping using an expression as end condition, PolyScope provides an option for continuously evaluating that expression, so that the "loop" can be interrupted anytime during its execution, rather than just after each iteration.

12.17 Command: SubProgram



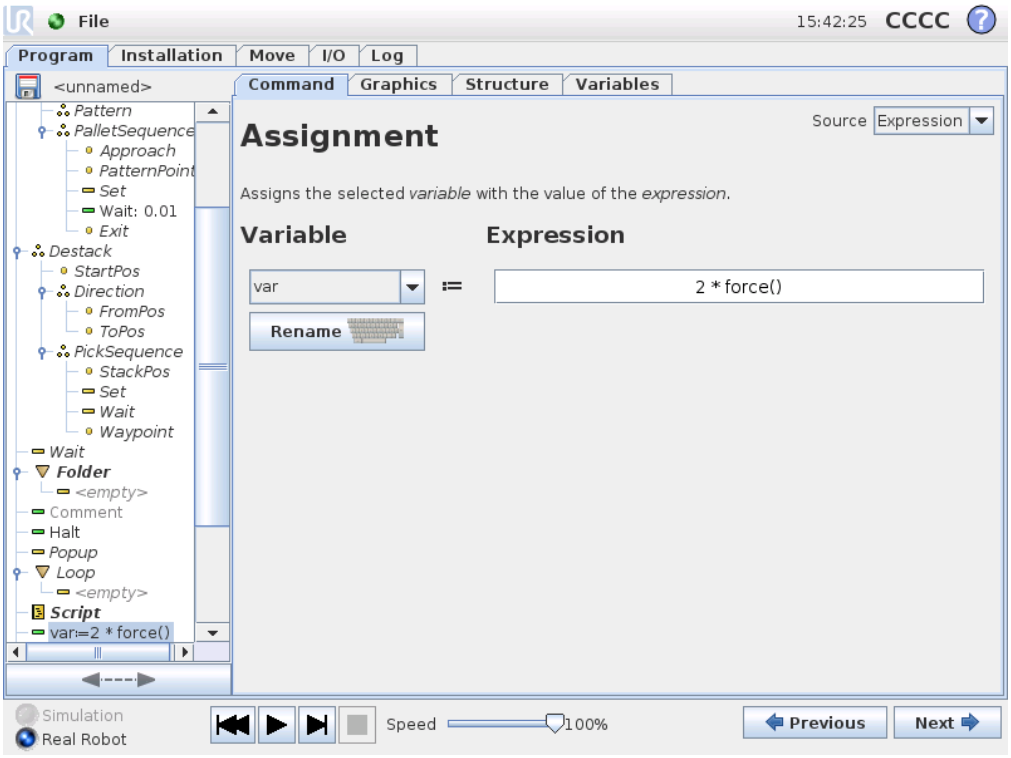
A Sub Program can hold program parts that are needed several places. A Sub Program can be a seperate file on the disk, and can also be hidden to protect against accidental changes to the SubProgram.

Command: Call SubProgram



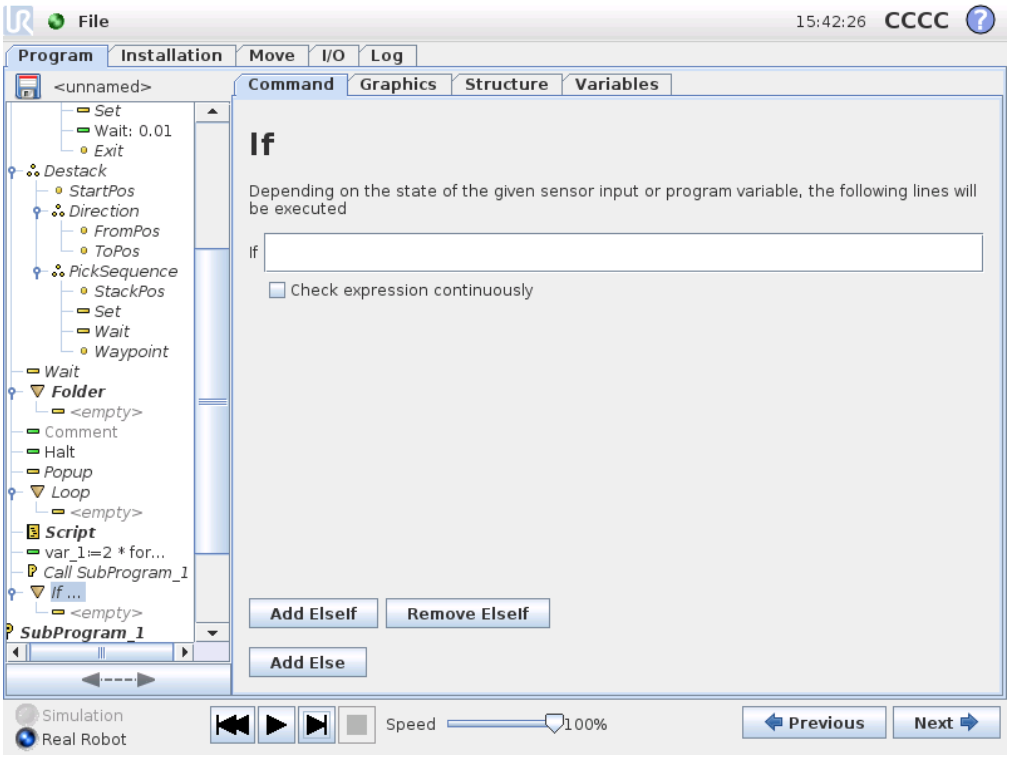
A call to a sub program will run the program lines in the sub program, and then return to the following line.

12.18 Command: 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.

12.19 Command: If

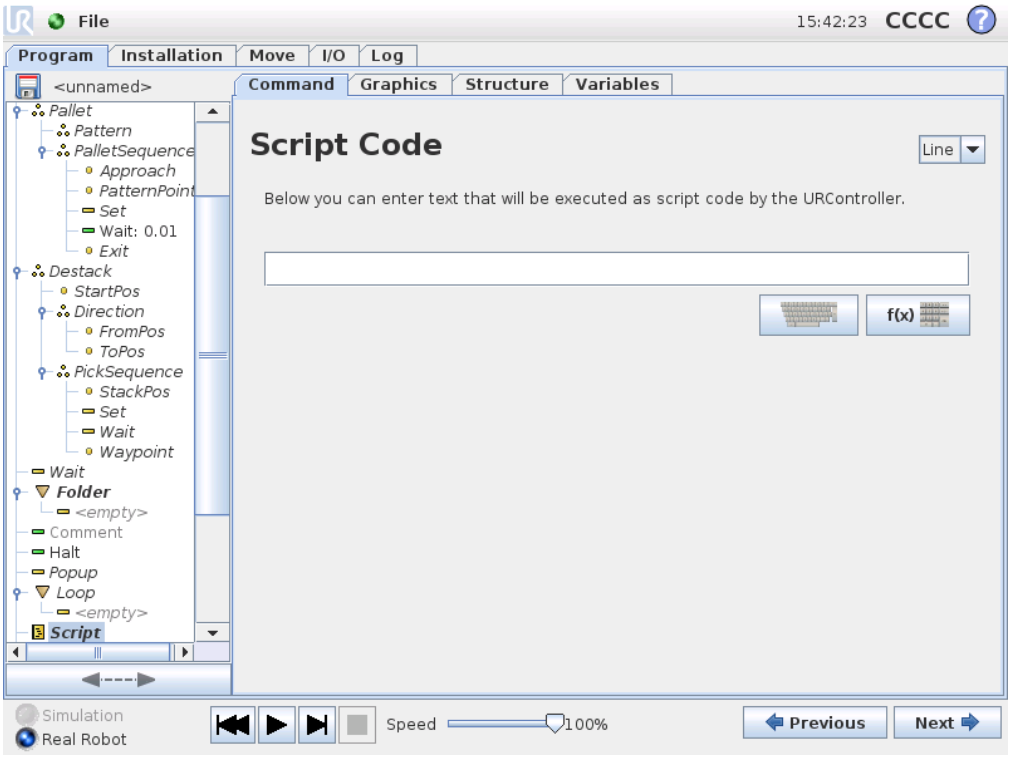


An "if...else" construction can make the robot change its behavior based on sensor inputs or variable values. Use the expression editor to describe the condition under which the robot should proceed to the sub-commands of this `If`. If the condition is evaluated to `True`, the lines inside this `If` are executed.

Each `If` can have several `ElseIf` and one `Else` command. These can be added using the buttons on the screen. An `ElseIf` command can be removed from the screen for that command.

The open `Check Expression Continuously` allow the conditions of the `If` and `ElseIf` statements to be evaluated while the contained lines are executed. If a expression evaluates to *False* while inside the body of the `If`-part, the following `ElseIf` or `Else` statement will be reached.

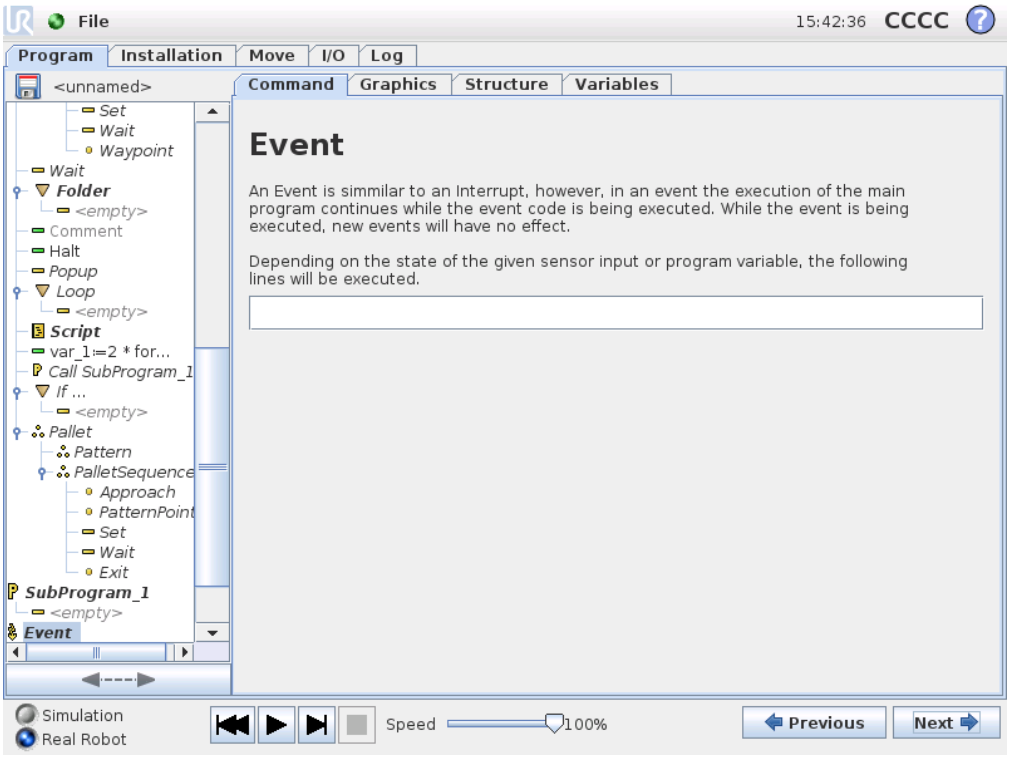
12.20 Command: Script



This command gives access to the underlying real time script language that is executed by the robot controller. It is intended for advanced users only and instructions on how to use it can be found in the Script Manual on the support website (`http://support.universal-robots.com/`). Note that only UR distributors and OEM customers have access to the website.

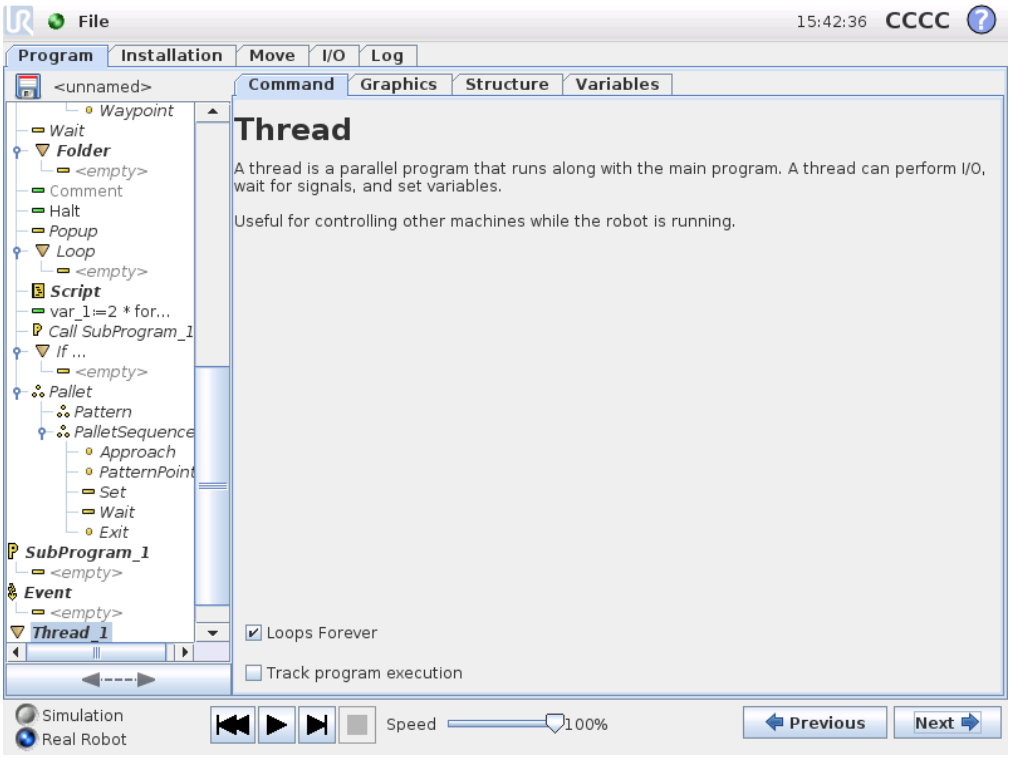
If the "File" option in the top left corner is choosen, it is possible to create and edit script programs files. This way, long and complex script programs can be used together with the operator-friendly programming of PolyScope.

12.21 Command: Event



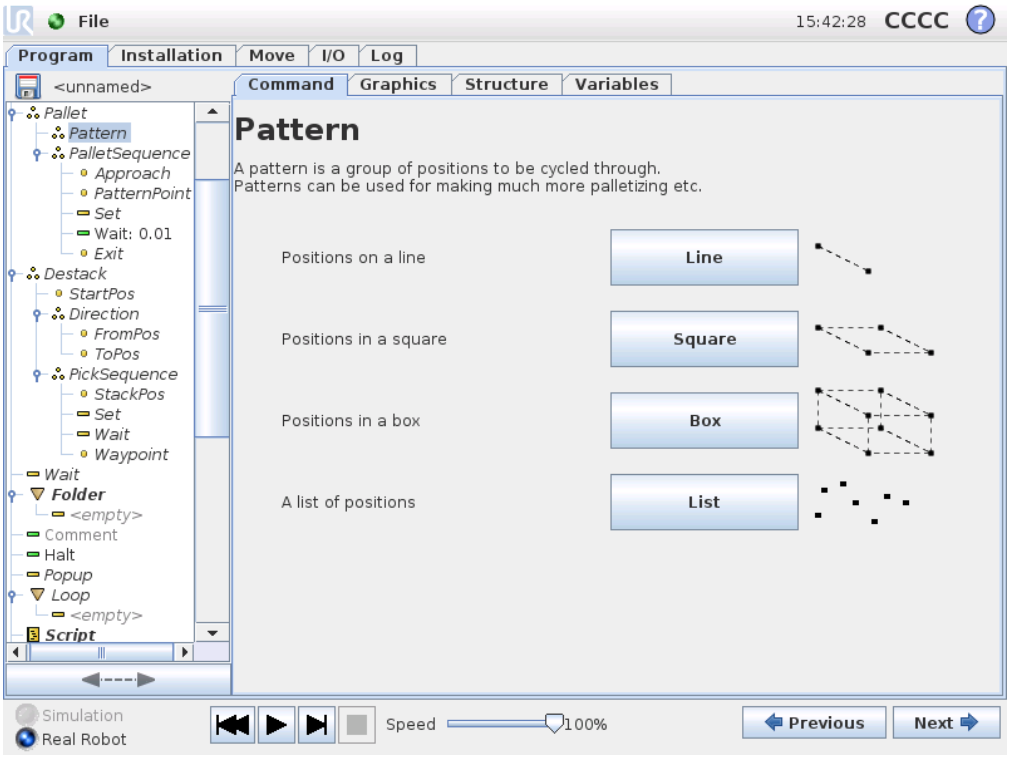
An event can be used to monitor an input signal, and perform some action or set a variable when that input signal goes high. For example, in the event that an output signal goes high, the event program can wait for 100ms and then set it back to low again. This can make the main program code a lot simpler in the case on an external machine triggering on a rising flank rather than a high input level.

12.22 Command: 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.

12.23 Command: Pattern



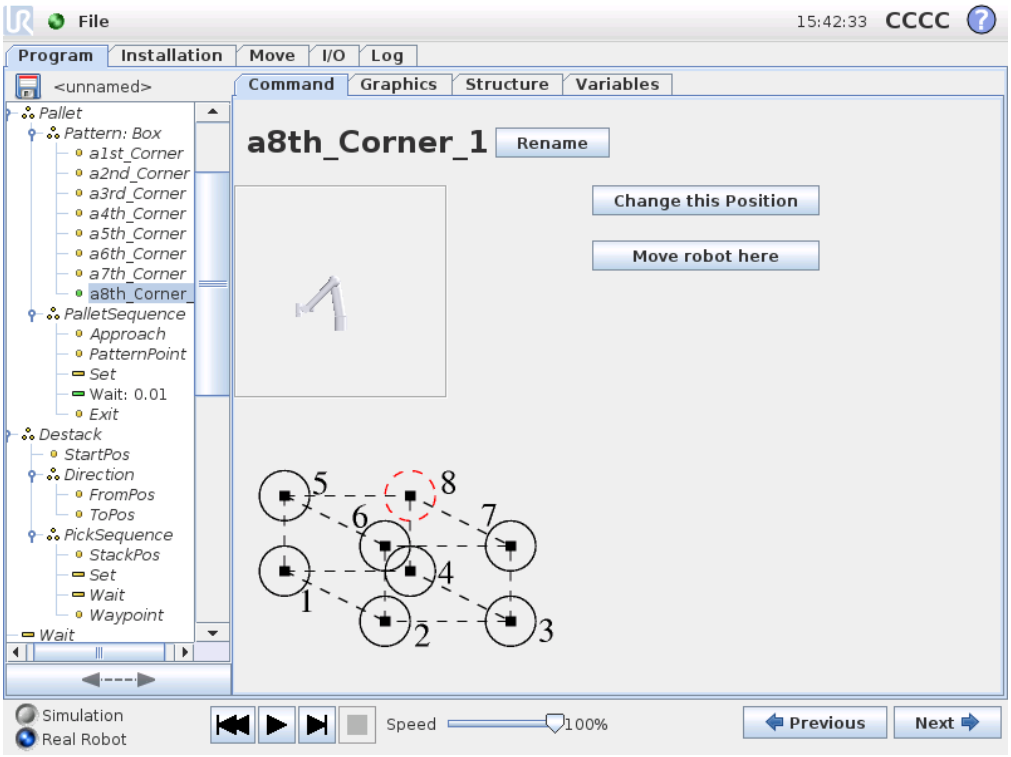
The Pattern command can be used to cycle through positions in the robot program. The pattern command corresponds to one position at each execution.

A pattern can be given as one of four types. The first three, “Line”, “Square” or “Box” can be used for positions in a regular pattern. The regular patterns are defined by a number of characteristic points, where the points define the edges of the pattern. For “Line” this is the two end points, for “Square” this is three of the four corner points, where as for “Box” this is four of the eight corner points. The programmer enters the number of positions along each of the edges of the pattern. The robot controller then calculates the individual pattern positions by proportionally adding the edge vectors together.

If the positions to be traversed do not fall in a regular pattern, the “List” option can be chosen, where a list of all the positions is provided by the programmer. This way any kind of arrangement of the positions can be realized.

Defining the Pattern

When the “Box” pattern is selected, the screen changes to what is shown below.



A "Box" pattern uses three vectors to define the side of the box. These three vectors are given as four points, where the first vector goes from point one to point two, the second vector goes from point two to point three, and the third vector goes from point three to point four. Each vector is divided by the interval count numbers. A specific position in the pattern is calculated by simply adding the interval vectors proportionally.

The "Line" and "Square" patterns work similarly.

A counter variable is used while traversing the positions of the pattern. The name of the variable can be seen on the `Pattern` command screen. The variable cycles through the numbers from 0 to $X * Y * Z - 1$, the number of points in the pattern. This variable can be manipulated using assignments, and can be used in expressions.

12.24 Command: Force

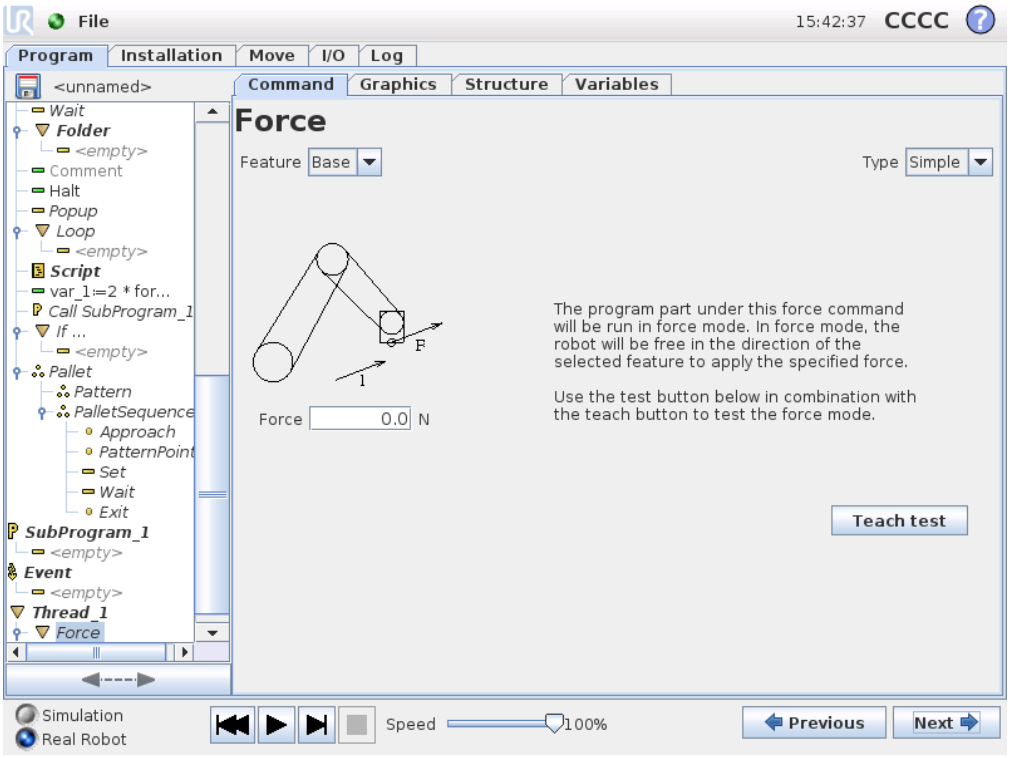
Force mode allows for compliance and forces in selectable axis in the robot's workspace. All robot arm movements under a Force command will be in Force mode. When the robot arm is moving in force mode, it is possible to select one or more axes in which the robot arm is compliant. Along/around compliant axes the robot arm will comply with the environment, which means it will automatically adjust 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 for applications where the actual tcp position along a predefined axis is not important, but in stead a desired force along that axis is required. For example if the robot TCP should roll against a curved surface, or when pushing or pulling

a workpiece. Force mode also supports applying certain torques around predefined axes. Note that if no obstacles are met in an axis where a non-zero force is set, the robot arm will try to accelerate along/about that axis.

Although an axis has been selected to be compliant, the robot program will still try to move the robot along/around that axis. However, the force control assures that the robot arm will still approach the specified force.

WARNING:
If the force function is used incorrectly, it can produce a force of more than 150N. The programmed force shall be taken into consideration during risk assessment.



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 11.12.

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 origo of the selected feature. The distance between the robot TCP and the origo 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 usefull when deburring 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 on 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 (12.28), when the robot is operating in force mode.

Force value selection

A force can be set for both compliant and non-compliant axes, but the effects are different.

- **Compliant**: The robot arm will adjust its position to achieve the selected force.
- **Non-compliant**: The robot arm will follow its trajectory set by the program while accounting for an external force of the value set here.

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

Limits selection

For all axes a limit can be set, but these have different meaning corresponding to the axes being complian or non-compliant.

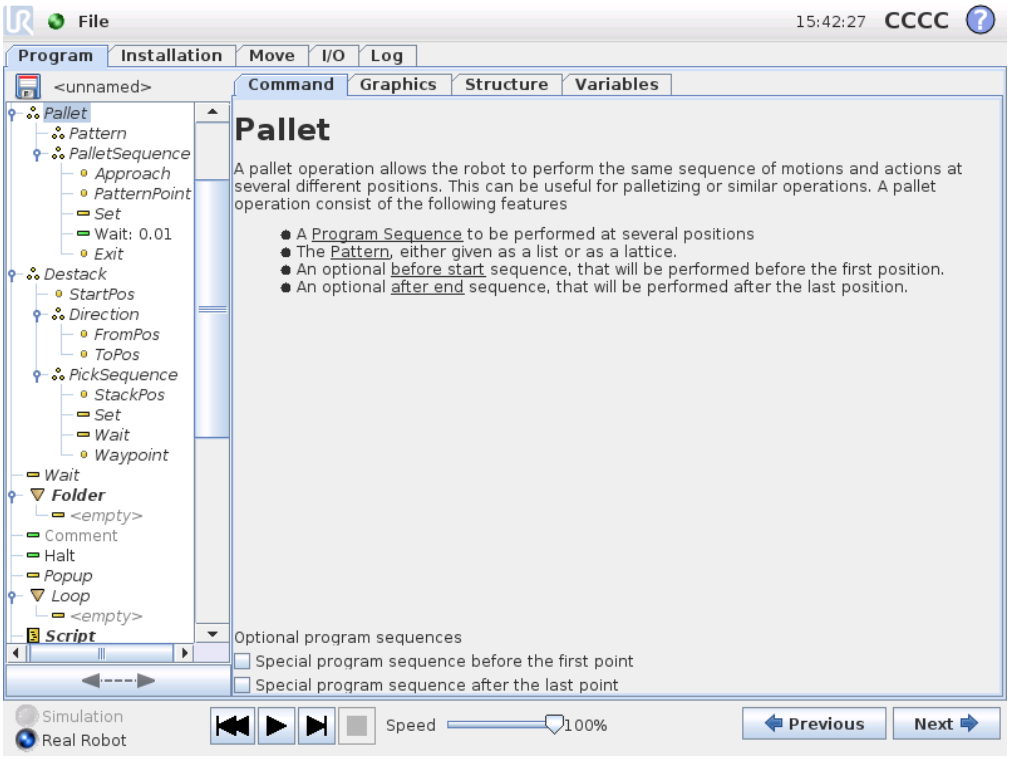
- **Compliant**: The limit is the maximum speed the TCP is allowed to attain along/about the axis. Units are [mm/s] and [deg/s].

- **Non-compliant**: The limit is the maximum deviation from the program trajectory which is allowed before the robot protective stops. Units are [mm] and [deg].

Test force settings

The on/off button, Teach Test, toggles the behavior of the Teach button on the back of the Teach Pendant from normal teaching mode to testing the force command. When the Teach Test button is on and the Teach 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 Teach button with the other, and notice in which directions the robot arm can/cannot be moved. Upon leaving this screen, the Teach Test button automatically switches off, which means the Teach button on the back of the Teach Pendant button is again used for free teach mode. Note: The Teach button will only be effectual when a valid feature has been selected for the Force command.

12.25 Command: Pallet



A pallet operation can perform a sequence of motions in a set of places given as a pattern, as described in 12.23. At each of the positions in the pattern, the sequence of motions will be run relative to the pattern position.

Programming a Pallet Operation

The steps to go through are as follows;

1. Define the pattern.
2. Make a "PalletSequence" for picking up/placing at each single point. The sequence describes what should be done at each pattern position.
3. Use the selector on the sequence command screen to define which of the waypoints in the sequence should correspond to the pattern positions.

Pallet Sequence/Anchorable Sequence

In an Pallet Sequence node, the motions of the robot arm are relative to the pallet position. The behavior of a sequence is such that the robot arm will be at the position specified by the pattern at the `Anchor Position/Pattern Point`. The remaining positions will all be moved to make this fit.

Do not use the `Move` command inside a sequence, as it will not be relative to the anchor position.

"BeforeStart"

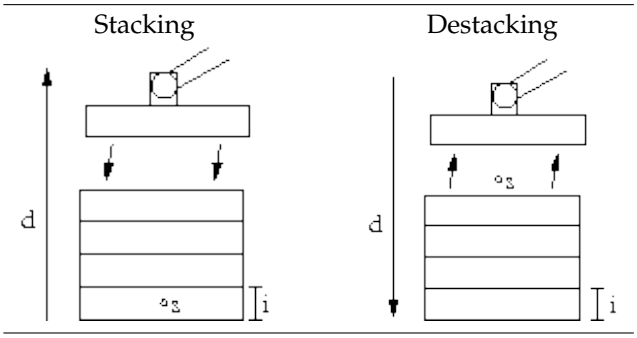
The optional `BeforeStart` sequence is run just before the operation starts. This can be used to wait for ready signals.

"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 pallet.

12.26 Command: Seek

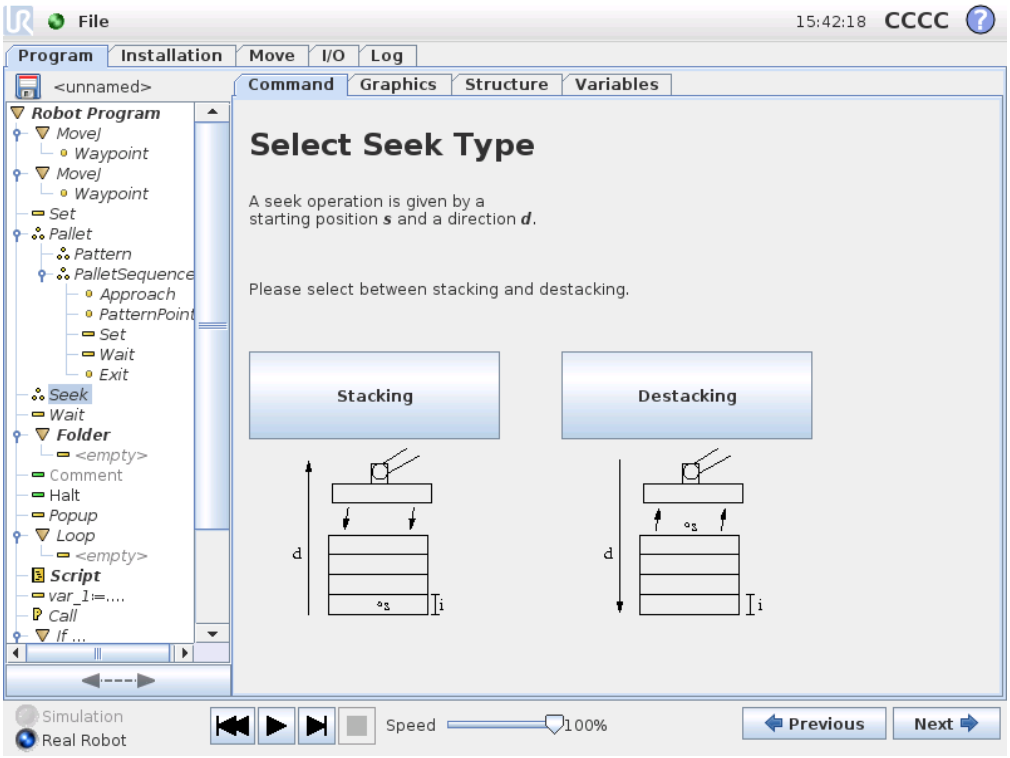
A seek function uses a sensor to determine when the correct position is reached to grab or drop an item. The sensor can be a push button switch, a pressure sensor or a capacitive sensor. This function is made for working on stacks of items with varying item thickness, or where the exact positions of 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.

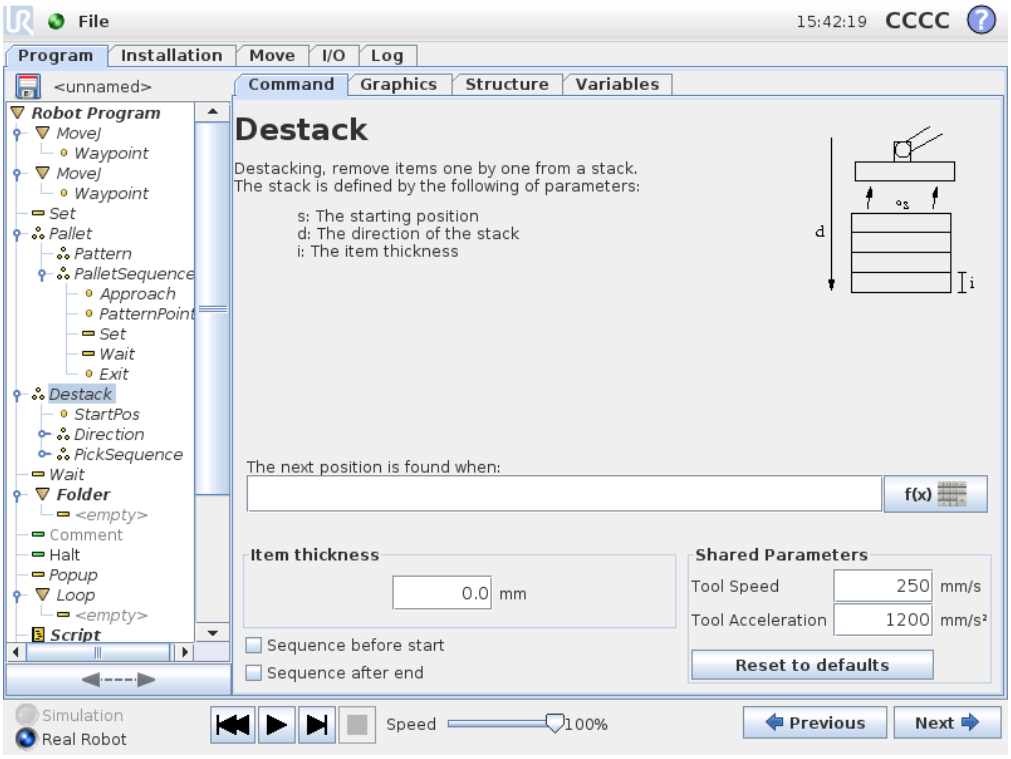
On top of this, one must define the condition for when the next stack position is reached, and a special program sequence that will be performed at each of the stack positions. Also speed and accelerations need to be given for the movement involved in the stack operation.

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

Destacking

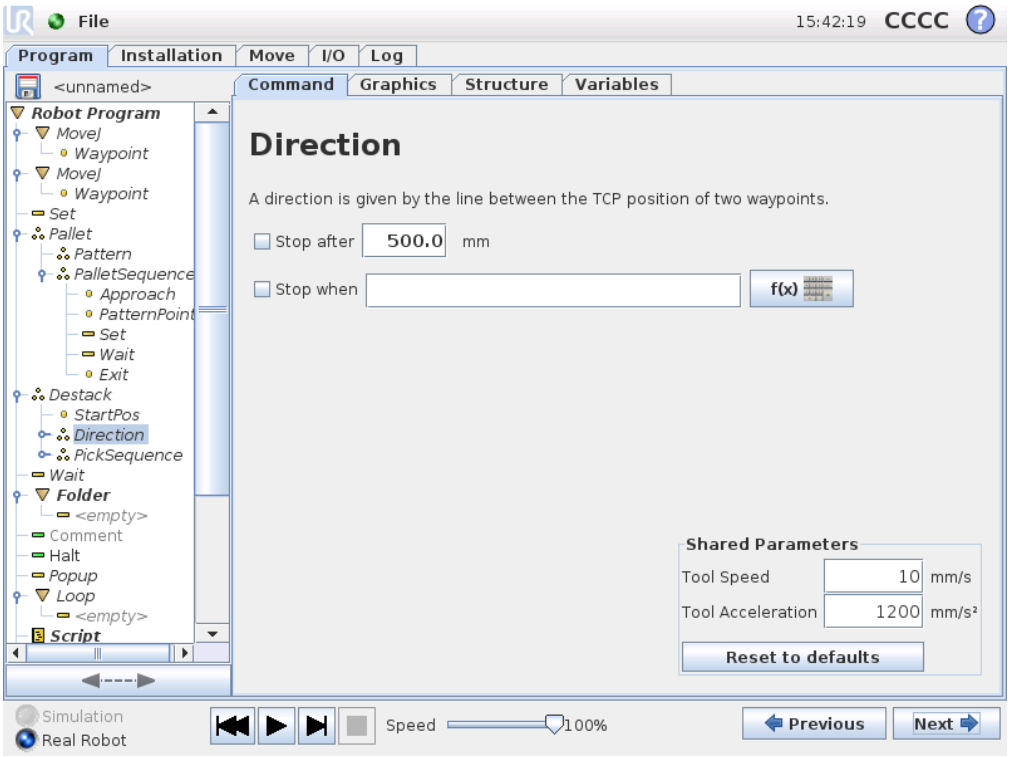


When destacking, the robot arm moves from the starting position in the given direction to search for the next item. The condition on the screen determines when the next item is reached. When the condition becomes satisfied, the robot remembers the position and performs the special sequence. The next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

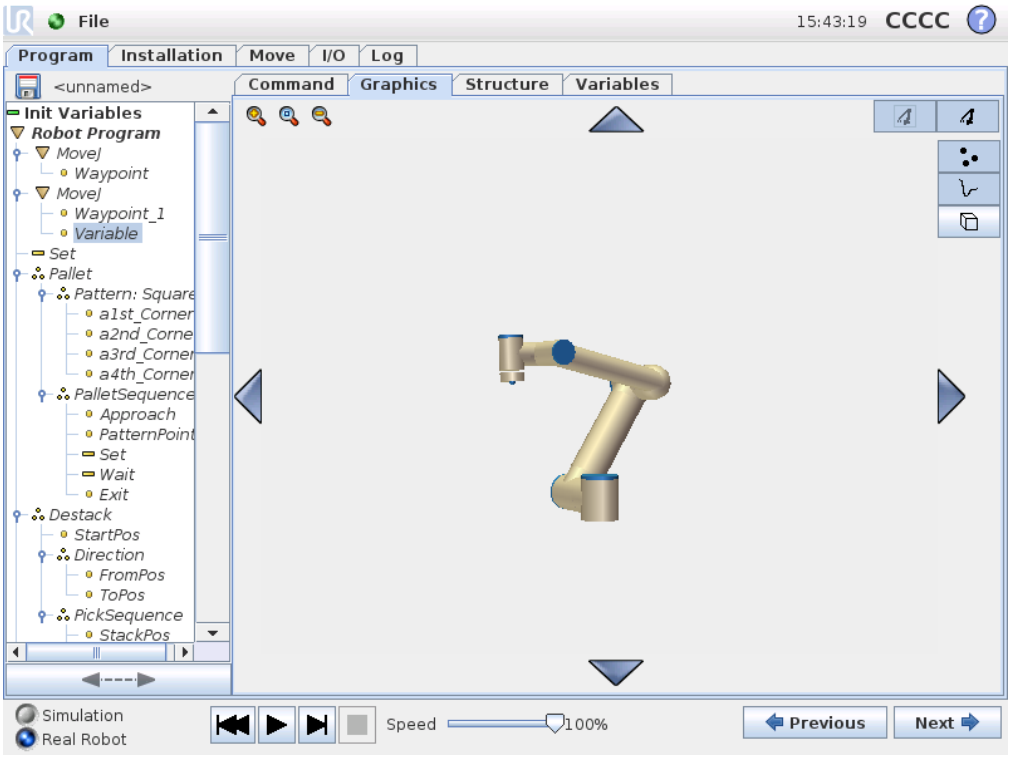
Pick/Place Sequence

Like for the Pallet operation (12.25), a special program sequence is performed at each stack position.

12.27 Command: Suppress

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.

12.28 Graphics Tab



Graphical representation of the current robot program. The path of the TCP is shown in the 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 14.11), a 3D representation of the proximate boundary limit is shown. Note that 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 14.5) are active.

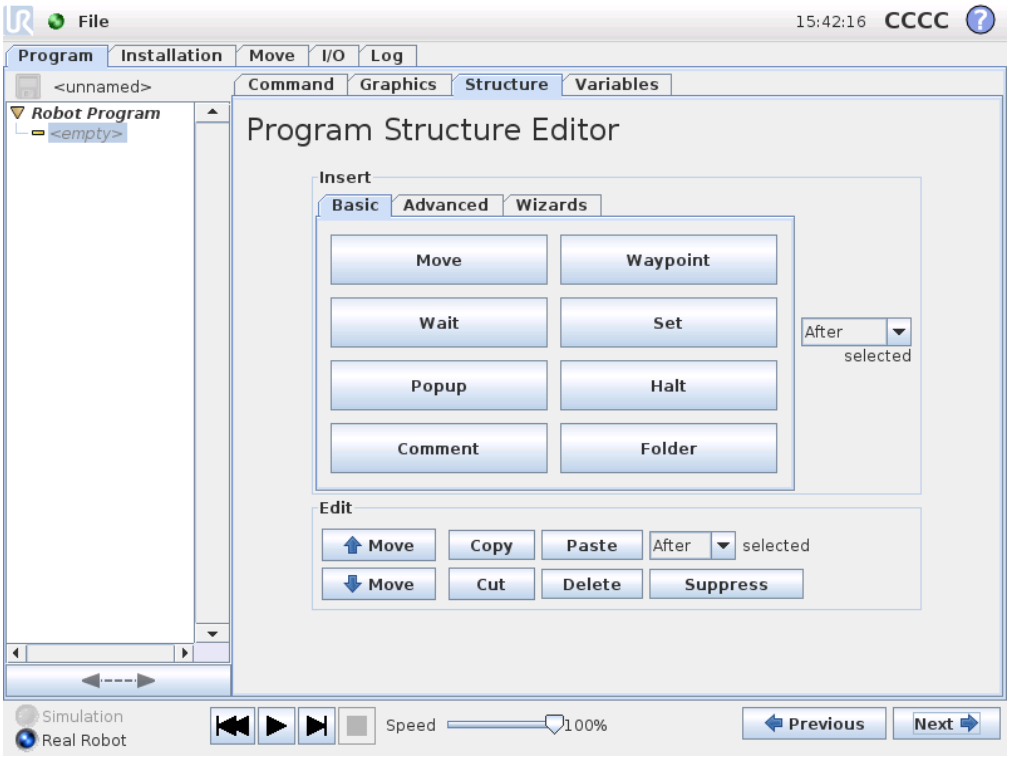
The tool orientation boundary limit is visualized with a spherical cone together with a vector indicating the current orientation of the robot tool. The inside of the cone represents the allowed area for the tool orientation (vector).

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 the 3D view. The bottom button switches on/off the visualization of proximate boundary limits.

The motion segments shown depend on the selected program node. If a `Move` node is selected, the displayed path is the motion defined by that move. If a `Waypoint` node is selected, the display shows the following $\sim$ 10 steps of movement.

12.29 Structure Tab



The program structure tab gives an opportunity for inserting, moving, copying and removing the various types of commands.

To insert new commands, perform the following steps:

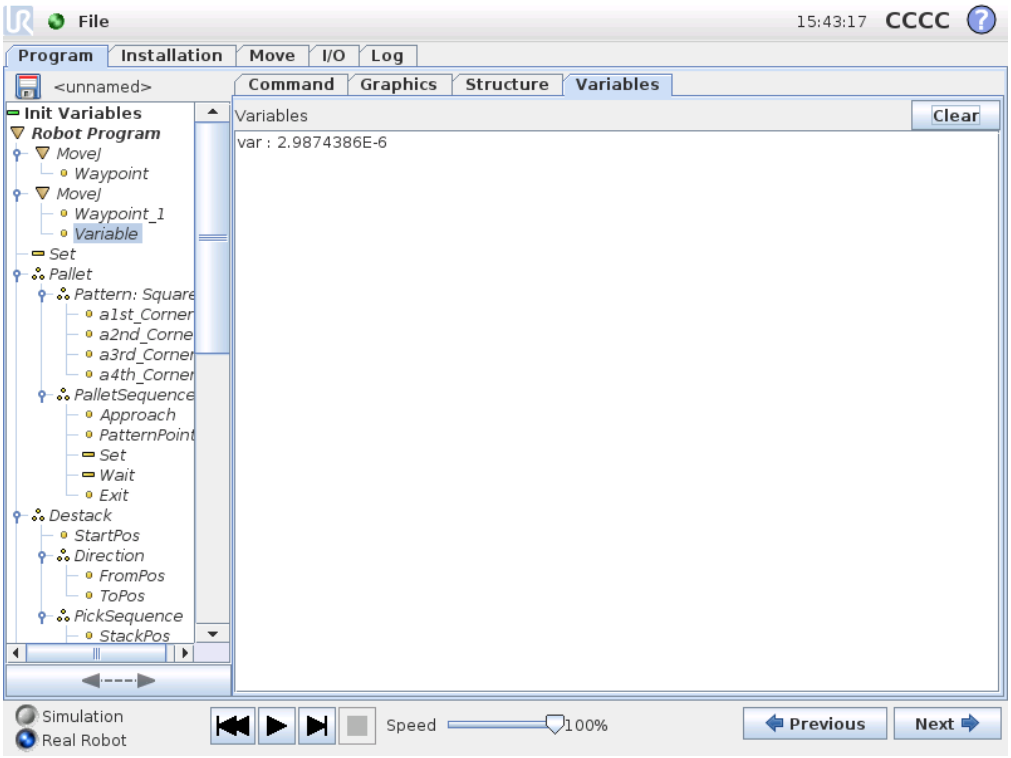
1) Select an existing program command.
2) Select whether the new command should be inserted above or below the selected command.

3) Press the button for the command type you wish to insert. For adjusting the details for the new command, go to the `Command` tab.

Commands can be moved/cloned/deleted using the buttons in the edit frame. If a command has sub-commands (a triangle next to the command) all sub-commands are also moved/cloned/deleted.

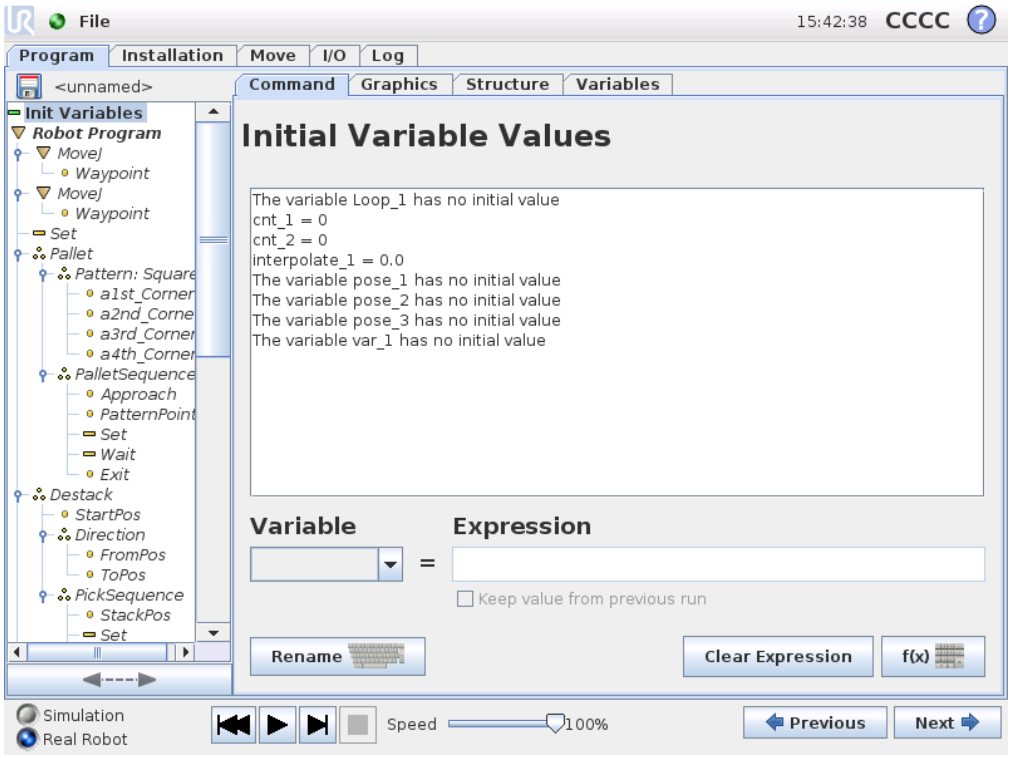
Not all commands fit at all places in a program. `Waypoints` must be under a Move command (not necessarily directly under). `ElseIf` and `Else` commands are required to be after an `If`. In general, moving `ElseIf` commands around can be messy. Variables must be assigned values before being used.

12.30 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. The variables are ordered alphabetically by their names. The variable names on this screen are shown with at most 50 characters, and the values of the variables are shown with at most 500 characters.

12.31 Command: Variables Initialization



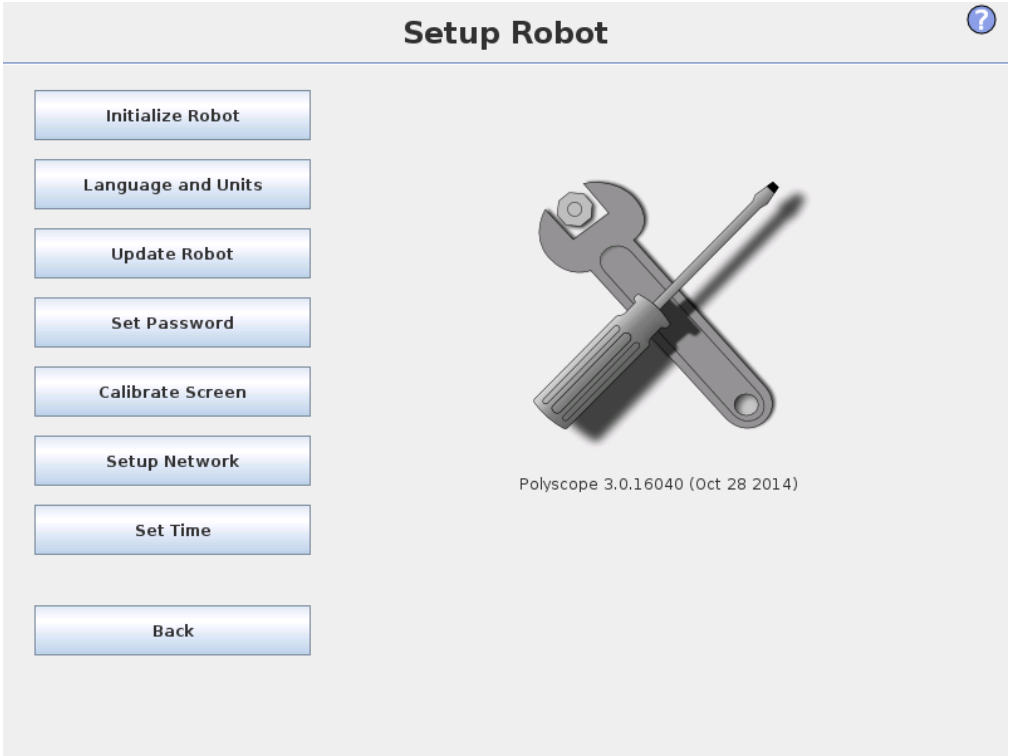
This screen allows setting variable values before the program (and any threads) start executing.

Select a variable from the list of variables by clicking on it, or by using the variable selector box. For a selected variable, an expression can be entered that will be used to set the variable value at program start.

If the “Prefers to keep value from last run” checkbox is selected, the variable will be initialized to the value found on the `Variables` tab, described in 12.30. This permits variables to maintain their values between program executions. The variable will get its value from the expression if the program is run for the first time, or if the value tab has been cleared.

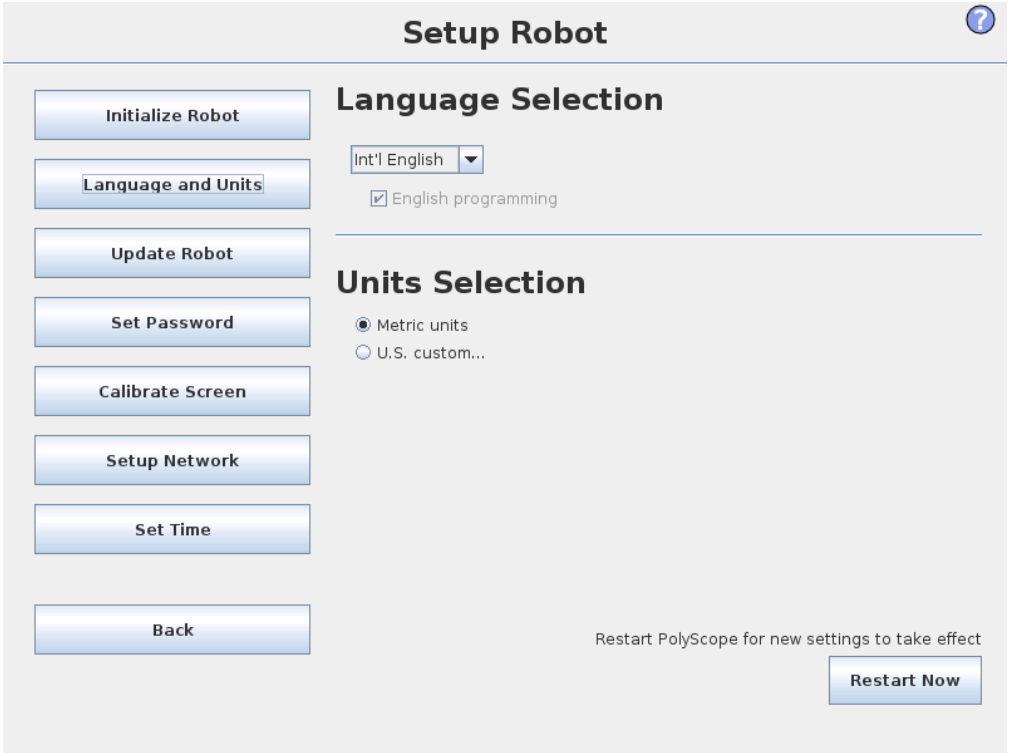
A variable can be deleted from the program by setting its name to blank (only spaces).

13 Setup Screen



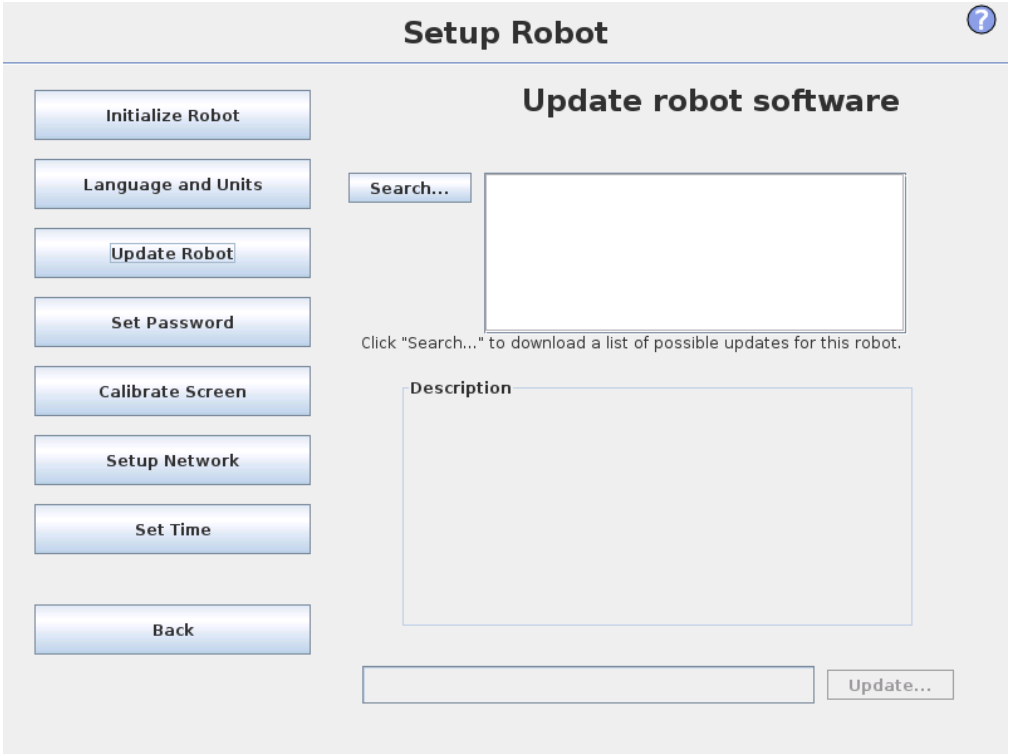
- **Initialize Robot** Goes to the initialization screen, see 9.4.
- **Language and Units** Configure the language and units of measurements for the user interface, see 13.1.
- **Update Robot** Upgrades the robot software to a newer version, see 13.2.
- **Set Password** Provides the facility to lock the programming part of the robot to people without a password, see 13.3.
- **Calibrate Screen** Calibrates the "touch" of the touch screen, see 13.4.
- **Setup Network** Opens the interface for setting up the Ethernet network for the robot control box, see 13.5.
- **Set Time** Set the time and date for the system and configure the display formats for the clock, see 13.6.
- **Back** Returns to the Welcome Screen.

13.1 Language and Units



Language and units used in PolyScope can be selected on this screen. The selected language will be used for the text visible on the various screens of PolyScope as well as in the embedded help. Tick off "English programming" to have the names of commands within robot programs written in English. PolyScope needs to be restarted for changes to take effect.

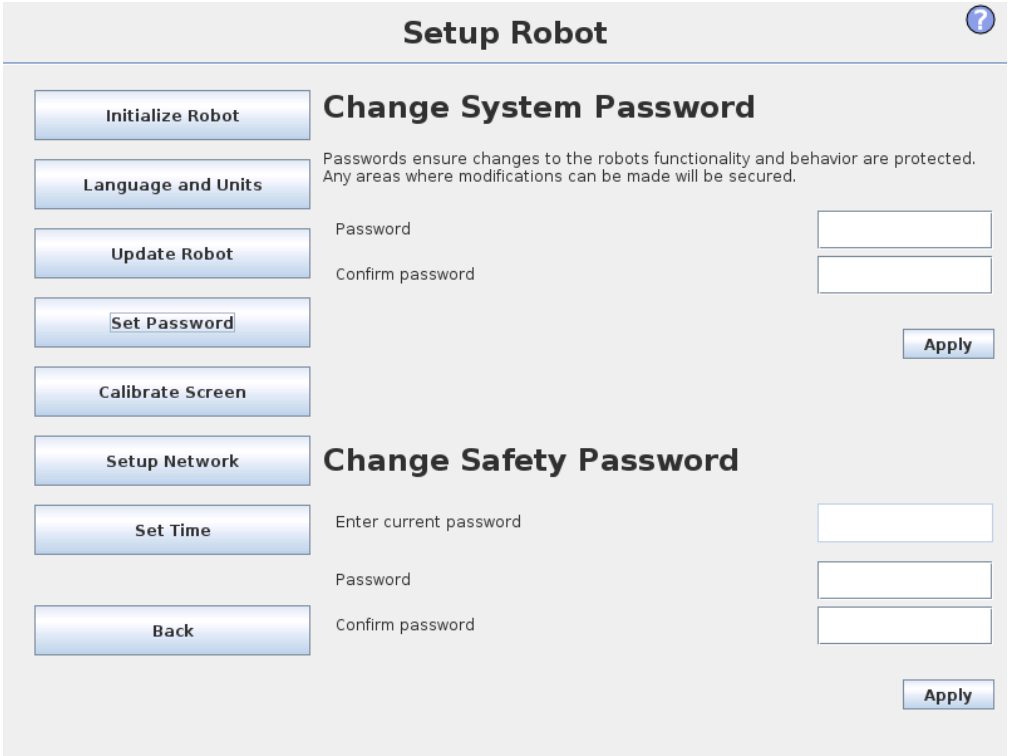
13.2 Update Robot



Software updates can be installed from USB flash memory. Insert an USB memory stick and click **Search** to list its contents. To perform an update, select a file, click **Update**, and follow the on-screen instructions.

WARNING:
Always check your programs after a software upgrade. The upgrade might change trajectories in your program. The updated software specifications can be found by pushing the "?" button located at the top right corner of the GUI. Hardware specifications remain the same and can be found in the original manual.

13.3 Set Password



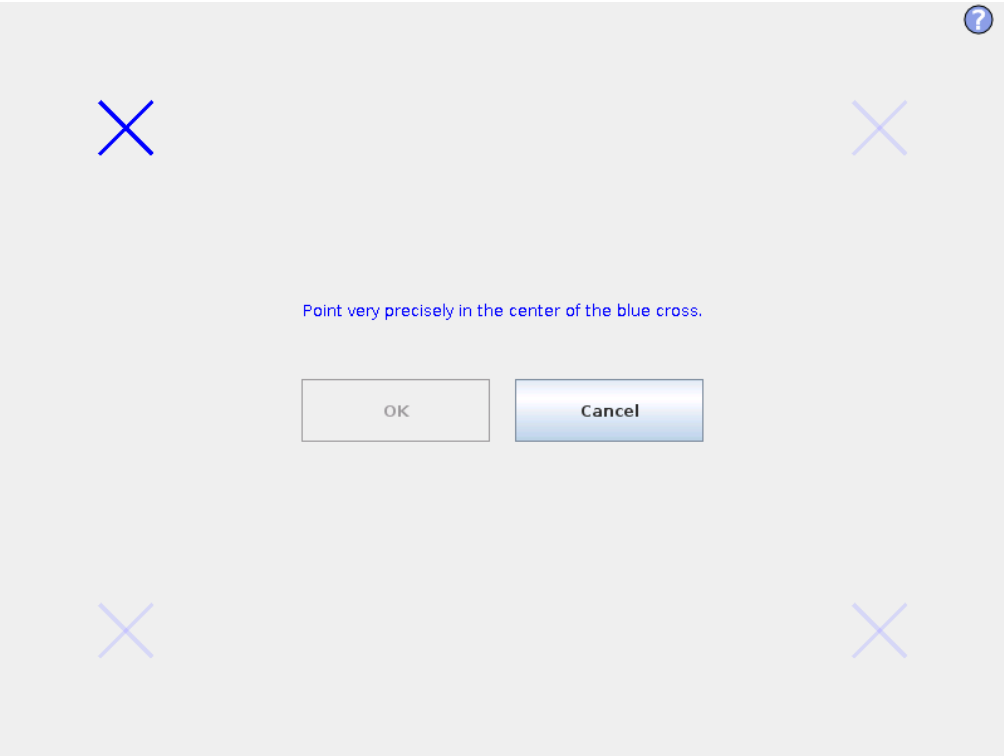
Two passwords are supported. The first is an *optional* System password which prevents unauthorized modification of the setup of the robot. When the System password is set, programs can be loaded and executed without the password, but the user must enter the correct password in order to create or change programs.

The second is a *required* Safety password which must be entered correctly in order to modify the safety configuration.

NOTE:
In order to change the safety configuration, the Safety password must be set.

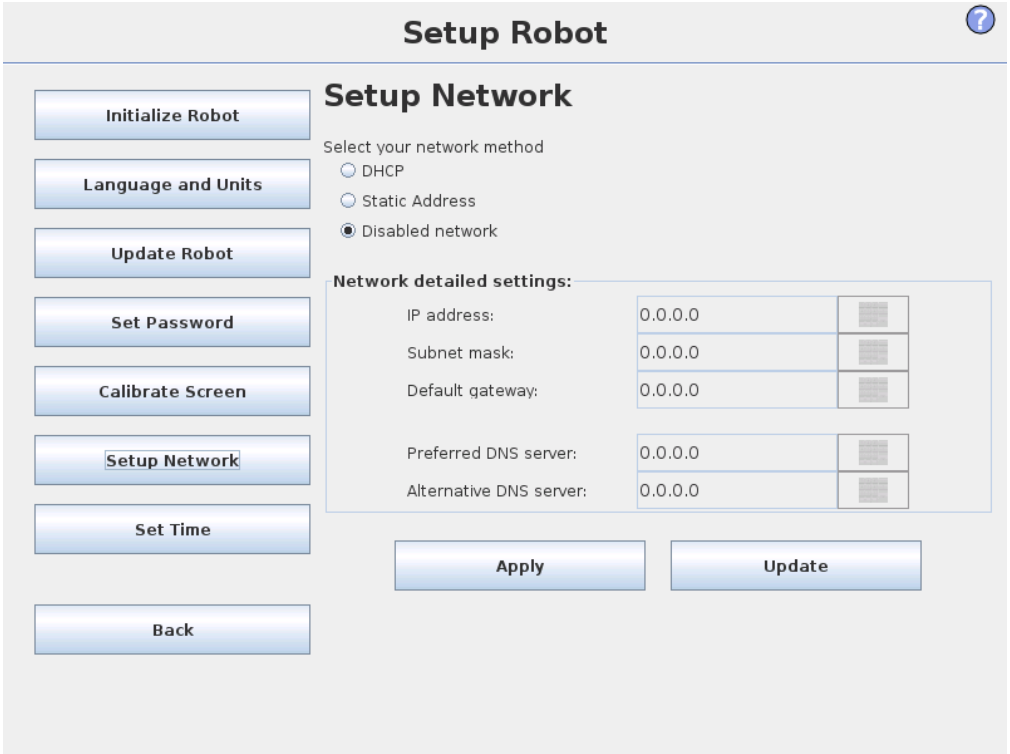
WARNING:
Add a System password to prevent non-authorized personnel from changing the robot installation.

13.4 Calibrate Screen



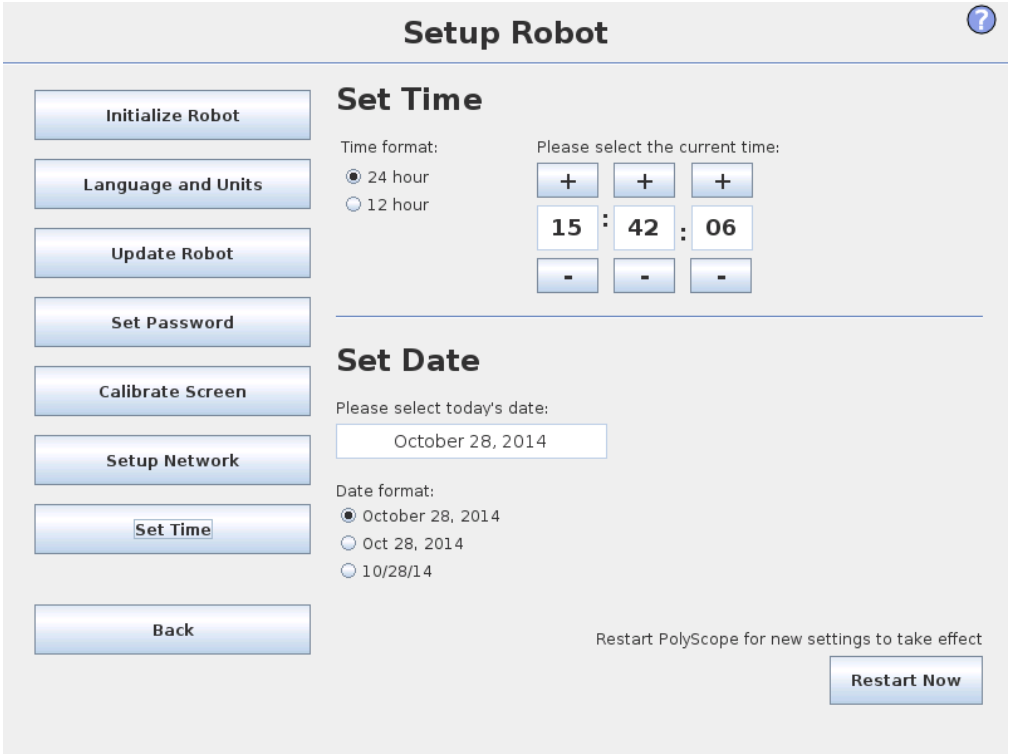
Calibrating the touch screen. Follow the on-screen instructions to calibrate the touch screen. Preferably use a pointed non-metallic object, such as a closed pen. Patience and care help achieve a better result.

13.5 Setup Network



Panel for setting up the Ethernet network. An Ethernet connection is not necessary for the basic robot functions, and is disabled by default.

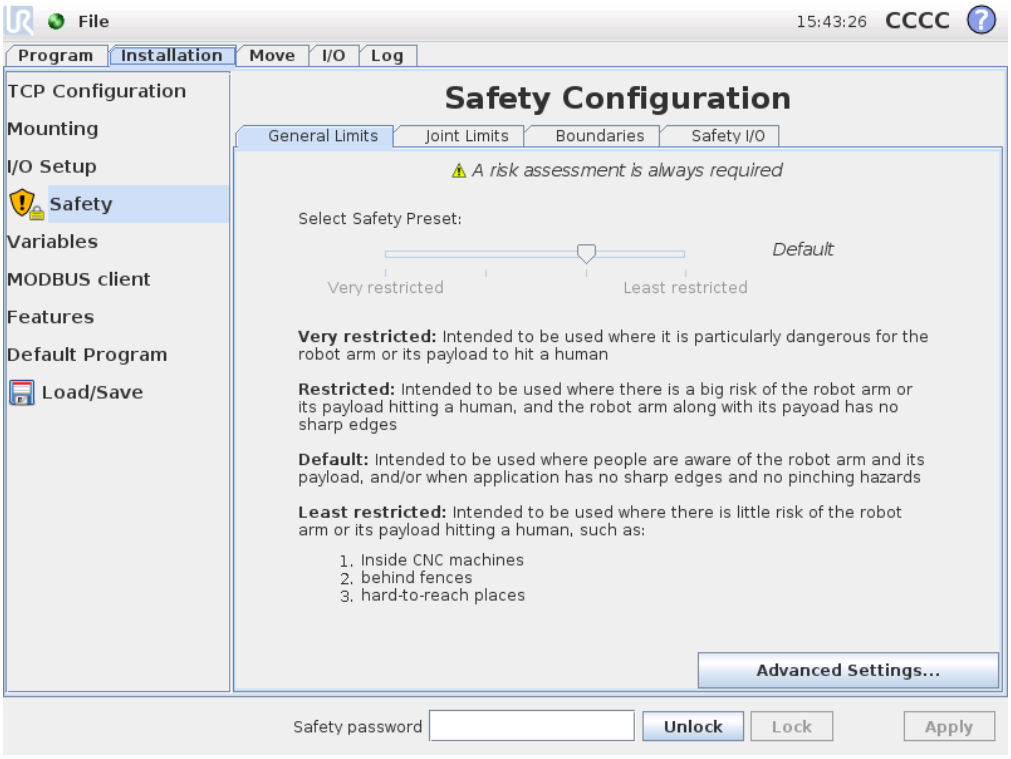
13.6 Set Time



Set the time and date for the system and configure the display formats for the clock. The clock is displayed at the top of the *Run Program* and *Program Robot* screens. Tapping on it will show the date briefly. The GUI needs to be restarted for changes to take effect.

14 Safety Configuration

The robot is equipped with an advanced safety system. Depending on the particular characteristics of its workspace, the settings for the safety system must be configured to guarantee the safety of all personnel and equipment around the robot. For details on the safety system, see the Hardware Installation Manual. The `Safety Configuration` screen can be accessed from the `Welcome` screen (see 9.3) by pressing the `Program Robot` button, selecting the `Installation` tab and tapping `Safety`. The safety configuration is password protected, see 14.7.



WARNING:

1. A risk assessment is always required.
2. All safety settings accessible on this screen and its subtabs are required to be set according to the risk assessment.
3. The integrator is required to ensure that all changes to the safety settings are done in agreement with the risk assessment.

The safety settings consist of a number of limit values used to constrain the move-

ments of the robot arm, and of safety function settings for the configurable inputs and outputs. They are defined in the following subtabs of the safety screen:

- The `General Limits` subtab defines the maximum *force*, *power*, *speed* and *momentum* of the robot arm. When the risk of hitting a human or colliding with a part of its environment is particularly high, these settings need to be set to low values. If the risk is low, higher general limits enable the robot to move faster and exert more force on its environment. For further details, see 14.9.
- The `Joint Limits` subtab consists of *joint speed* and *joint position* limits. The *joint speed* limits define the maximum angular velocity of individual joints and serve to further limit the speed of the robot arm. The *joint position* limits define the allowed position range of individual joints (in joint space). For further details, see 14.10.
- The `Boundaries` subtab defines safety planes (in Cartesian space) and a tool orientation boundary for the robot TCP. The safety planes can be configured either as hard limits for the position of the robot TCP, or triggers for activating the *Reduced* mode safety limits (see 14.5)). The tool orientation boundary puts a hard limit on the orientation of the robot TCP. For further details, see 14.11.
- The `Safety I/O` subtab defines safety functions for configurable inputs and outputs (see 11.2). For example, *Emergency Stop* can be configured as an input. For further details, see 14.12.

14.1 Changing the Safety Configuration

NOTE:
The recommended procedure for changing the safety configuration is as follows:

1. Make a risk assessment.
2. Adjust safety settings to the appropriate level (refer to relevant directives and standards from our manual on how to set the safety limits).
3. Test the setting on the robot.
4. Put the 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 checksum in the top right corner of the PolyScope (see 14.4 in the PolyScope Manual)."

14.2 Safety Synchronization and Errors

The state of the applied Safety configuration in comparison to what robot installation the GUI has loaded, is depicted by the shield icon next to the text `Safety` on the left side of the screen. These icons provide a quick indicator to the current state. They are defined below:


Configuration Synchronized: Shows the GUI installation is identical to the currently applied Safety configuration. No changes have been made.


Configuration Altered: Shows the GUI installation is different from the currently applied Safety configuration.

When editing the Safety configuration, the shield icon will inform you whether or not the current settings have been applied.

If any of the text fields in the `Safety` tab contain any invalid input, the Safety configuration is in an error state. This is indicated in several ways:

- A red error icon is displayed next to the text `Safety` on the left side of the screen.
- The subtab(s) with errors are marked with a red error icon at the top.
- Text fields containing errors are marked with a red background.

When errors exist and attempting to navigate away from the `Installation` tab, a dialog appears with the following options:

1. Resolve the issue(s) so that all errors have been removed. This will be visible when the red error icon is no longer displayed next to the text `Safety` on the left side of the screen.
2. Revert back to the previously applied Safety configuration. This will disregard all changes and allow you to continue to the desired destination.

If no errors exist and attempting to navigate away, a different dialog appears with the following options:

1. Apply changes and restart the system. This will apply the Safety configuration modifications to the system and restart. Note: This does not imply that any changes have been saved; shutdown of the robot at this point will lose all changes to the robot installation including the Safety configuration.
2. Revert back to the previously applied Safety configuration. This will disregard all changes and allow you to continue to the desired selected destination.

14.3 Tolerances

In the *Safety Configuration*, physical limits are set. The input fields for these limits are excluding the tolerances: where applicable tolerances are displayed next to the field.

The *Safety System* receives the values from the input fields, and detects any violation of these values. The *Robot Arm* attempts to prevent any violations of the safety system and gives a protective stop by stopping the program execution when the limit minus the tolerance is reached. Note, that this means that a program might not be able to perform motions very close to a limit, e.g. the robot may not be able to obtain the exact maximum speed specified by a joint speed limit or the TCP speed limit.

WARNING:
A risk assessment is always required using the limit values without tolerances.

WARNING:
Tolerances are specific to the version of the software. Updating the software may change the tolerances. Consult the release notes for changes between versions.

14.4 Safety Checksum

The text in the top right corner of the screen gives a shorthand representation of the safety configuration currently used by the robot. When the text changes, this indicates that the current safety configuration has changed as well. Clicking on the checksum displays the details about the currently active safety configuration.

14.5 Safety Modes

Under normal conditions (i.e. when no protective stop is in effect), the safety system operates in one of the following *safety modes*, each with an associated set of safety limits:

Normal mode: The safety mode that is active by default;

Reduced mode: Active when the robot TCP is positioned beyond a *Trigger Reduced mode* plane (see 14.11), or when triggered using a configurable input (see 14.12).

Recovery mode: When the robot arm is in violation of one of the other modes (i.e. *Normal* or *Reduced* mode) and a category 0 stop has occurred, the robot arm will start up in *Recovery* mode. This mode allows the robot arm to be manually adjusted until all violations have been resolved. It is not possible to run programs for the robot in this mode.

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

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

When a safety limit from the active limit set is violated, the robot arm performs a category 0 stop. 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 limited by a fixed limit set that is not customizable by the user. For details about *Recovery* mode limits, see **??** in the Hardware Installation Manual.

14.6 Teach Mode

When in *Teach* mode (see 11.1.5) and the movement of the robot arm comes close to certain limits, the user will feel a repelling force. This force is generated for limits on the position, orientation and speed of the robot TCP and the position and speed of the joints.

The purpose of this repelling force is to inform the user that the current position or speed is close to a limit and to prevent the robot from violating that limit. However, if enough force is applied by the user to the robot arm, the limit can be violated. The magnitude of the force increases as the robot arm comes closer to the limit.

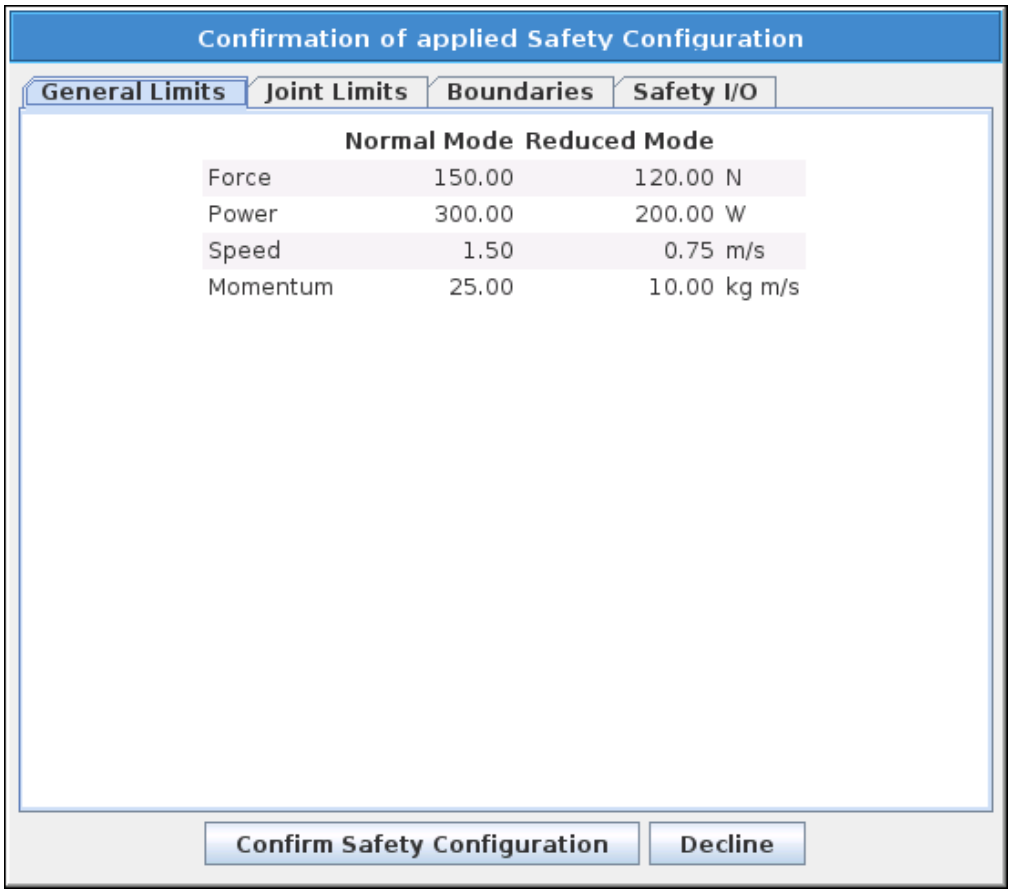
14.7 Password Lock

All settings on this screen are locked until the correct Safety password (see 13.3) is entered in the white text field at the bottom of the screen and the `Unlock` button is pressed. The screen can be locked again by clicking the `Lock` button. The `Safety` tab is automatically locked when navigating away from the Safety Configuration screen. When the settings are locked, a lock icon is visible next to the text `Safety` on the left side of the screen. An unlock icon is shown when the settings are unlocked.

NOTE:
Note that the robot arm is powered off when the Safety Configuration screen is unlocked.

14.8 Apply

When unlocking the safety configuration, the robot arm will be powered off while changes are being made. The robot arm cannot be powered on until the changes have been applied or reverted, and a manual power on is performed from the initialization screen. Any changes to the safety configuration must be applied or reverted, before navigating away from the Installation tab. These changes are *not* in effect until after the `Apply` button is pressed and confirmation is performed. Confirmation requires visual inspection of the changes given to the robot arm. For safety reasons, the information shown is given in SI Units. An example of the confirmation dialog is shown in figure 14.8.



Furthermore, on confirmation the changes are automatically saved as part of the current robot installation. See 11.5 for further information on saving the robot installation.

14.9 General Limits

The general safety limits serve to limit the linear speed of the robot TCP as well as the force it may exert on the environment. They are composed of the following values:

Force: A limit for the maximum force that the robot TCP exerts on the environment.

Power: A limit for the maximum mechanical work produced by the robot on the environment, considering that the payload is part of the robot and not of the environment.

Speed: A limit for the maximum linear speed of the robot TCP.

Momentum: A limit for the maximum momentum of the robot arm.

There are two means available for configuring the general safety limits within the installation; *Basic Settings* and *Advanced Settings* which are described more fully below.

Defining the general safety limits only defines the limits for the tool, and not the overall limits of the robot arm. This means that although a speed limit is specified, it does *not* guarantee that other parts of the robot arm will obey this same limitation.

When in *Teach* mode (see 11.1.5), and the current speed of the robot TCP is close to the *Speed* limit, the user will feel a repelling force which increases in magnitude the closer the speed comes to the limit. The force is generated when the current speed is within approximately 250 $^{mm}/_{s}$ of the limit.

Basic Settings The initial general limits subpanel, shown as the default screen, features a slider with the following predefined sets of values for the general limits in both *Normal* and *Reduced* modes:

Very Restricted: Intended to be used where it is particularly dangerous for the robot arm or its payload to hit a human.

Restricted: Intended to be used where there is a high risk of the robot arm or its payload hitting a human, and the robot arm along with its payload has no sharp edges.

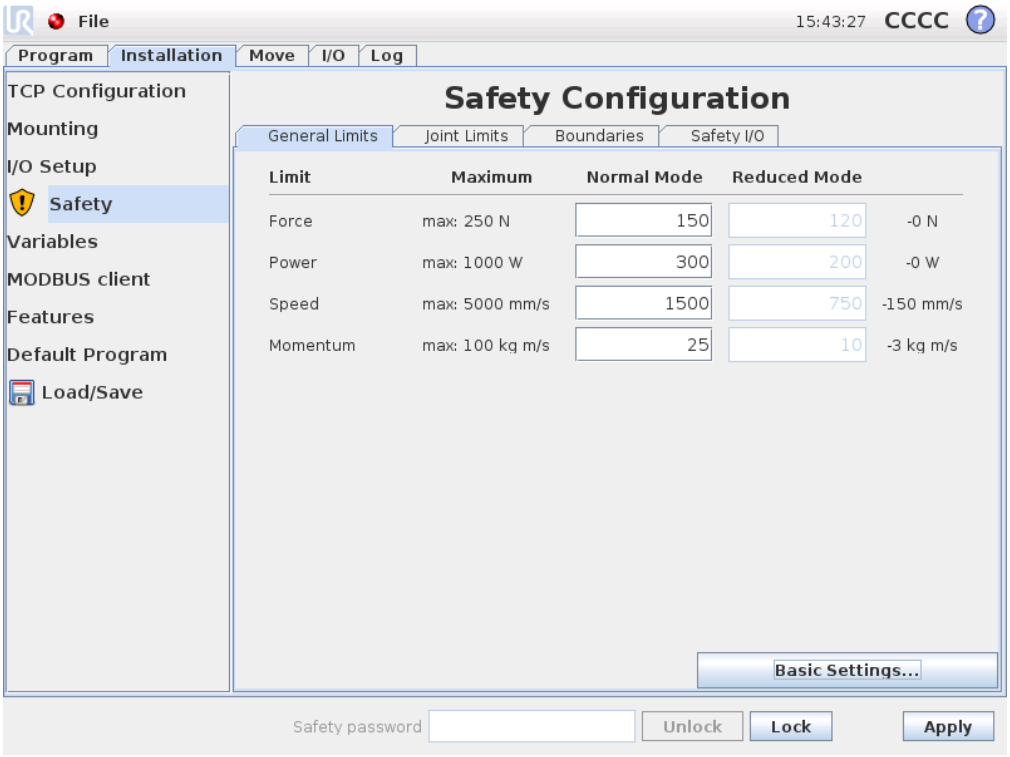
Default: Intended to be used where people are aware of the robot arm and its payload, and/or when application has no sharp edges and no pinching hazards.

Least Restricted: Intended to be used where there is little risk of the robot arm or its payload hitting a human, such as inside CNC machines, behind fences or in hard-to-reach places.

These modes are merely suggestions and a proper risk assessment is always required.

Switching to Advanced Settings Should *none* of the predefined sets of values be satisfactory, the `Advanced Settings...` button can be pressed to enter the advanced general limits screen.

Advanced Settings



Here, each of the general limits, described in 14.9, can be modified independently of the others. This is done by tapping the corresponding text field and entering the new value. The highest accepted value for each of the limits is listed in the column titled `Maximum`. The force limit can be set to a value between 100N and 250N, and the power limit can be set to a value between 80W and 1000W.

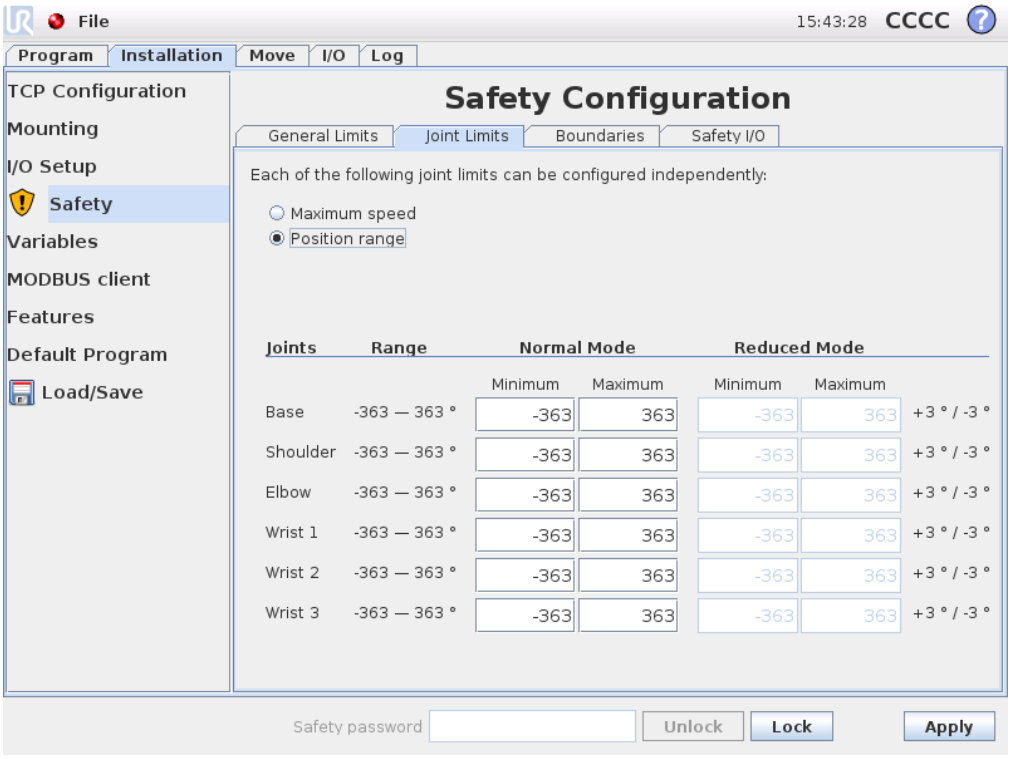
Note that the fields for limits in *Reduced* mode are disabled when neither a safety plane nor a configurable input is set to trigger it (see 14.11 and 14.12 for more details). Furthermore, the *Speed* and *Momentum* limits in *Reduced* mode must not be higher than their *Normal* mode counterparts.

The tolerance and unit for each limit are listed at the end of the row that corresponds to it. When a program is running, the speed of the robot arm is automatically adjusted in order to not exceed any of the entered values minus the tolerance (see 14.3). Note that the minus sign displayed with the tolerance value is only there to indicate that the tolerance is subtracted from the actual entered value. The safety system performs a category 0 stop, should the robot arm exceed the limit (without tolerance).

WARNING:
The speed limit is imposed only on the robot TCP, so other parts of the robot arm may move faster than the defined value.

Switching to Basic Settings Pressing the `Basic Settings...` button switches back to the basic general limits screen and all general limits are reset to their *Default* preset. Should this cause any customized values to be lost, a popup dialog is shown to confirm the action.

14.10 Joint Limits



Joint limits restrict the movement of individual joints in joint space, i.e. they do not refer to Cartesian space but rather to the internal (rotational) position of the joints and their rotational speed. The radio buttons in the upper portion of the subpanel make it possible to independently set up `Maximum Speed` and `Position Range` for the joints.

When in *Teach* mode (see 11.1.5), and the current position or speed of a joint is close to the limit, the user will feel a repelling force which increases in magnitude as the joint approaches the limit. The force is generated when joint speed is within approximately 20 °/s of the speed limit or joint position is within approximately 8 ° of the position limit.

Maximum Speed This option defines the maximum angular velocity for each joint. This is done by tapping the corresponding text field and entering the new value. The highest accepted value is listed in the column titled `Maximum`. None of the values can be set below the tolerance value.

Note that the fields for limits in *Reduced* mode are disabled when neither a safety plane nor a configurable input is set to trigger it (see 14.11 and 14.12 for more details). Furthermore, the limits for *Reduced* mode must not be higher than their *Normal* mode counterparts.

The tolerance and unit for each limit are listed at the end of the row that corresponds to it. When a program is running, the speed of the robot arm is automatically adjusted in order to not exceed any of the entered values minus the tolerance (see 14.3). Note that the minus sign displayed with each tolerance value is only there to indicate that the tolerance is subtracted from the actual entered value. Nevertheless, should the angular velocity of some joint exceed the entered value (without tolerance), the safety system performs a category 0 stop.

Position Range This screen defines the position range for each joint. This is done by tapping the corresponding text fields and entering new values for the lower and upper joint position boundary. The entered interval must fall within the values listed in the column titled `Range` and the lower boundary cannot exceed the upper boundary.

Note that the fields for limits in *Reduced* mode are disabled when neither a safety plane nor a configurable input is set to trigger it (see 14.11 and 14.12 for more details).

The tolerances and unit for each limit are listed at the end of the row that corresponds to it. The first tolerance value applies to the minimum value and the second applies to the maximum value. Program execution is aborted when the position of a joint is about to exceed the range resulting from adding the first tolerance to the entered minimum value and subtracting the second tolerance from the entered maximum value, if it continues moving along the predicted trajectory. Note that the minus sign displayed with the tolerance value is only there to indicate that the tolerance is subtracted from the actual entered value. Nevertheless, should the joint position exceed the entered range, the safety system performs a category 0 stop.

14.11 Boundaries

In this tab you can configure boundary limits consisting of safety planes and a limit on the maximum allowed deviation of the robot tool orientation. It is also possible to define planes that trigger a transition into *Reduced* mode.

Safety planes can be used to restrict the allowed workspace of the robot by enforcing that the robot TCP stay on the correct side of the defined planes and not pass through them. Up to eight safety planes can be configured. The constraint on the orientation of tool can be utilized to ensure that the robot tool orientation does not deviate more than a certain specified amount from a desired orientation.

WARNING:
Defining safety planes only limits the TCP and not the overall limit for the robot arm. This means that although a safety plane is specified, it does *not* guarantee that other parts of the robot arm will obey this restriction.

The configuration of each boundary limit is based on one of the features defined in the current robot installation (see 11.12).

NOTE:
It is highly recommended, that you create all features needed for the configuration of all the desired boundary limits and assign them appropriate names before editing the safety configuration. Note that since the robot arm is powered off once the *Safety* tab has been unlocked, the `Tool` feature (containing the current position and orientation of the robot TCP) as well as *Teach* mode (see 11.1.5) will not be available.

When in *Teach* mode (see 11.1.5), and the current position of the robot TCP is close to a safety plane, or the deviation of the orientation of the robot tool from the desired orientation is close to the specified maximum deviation, the user will feel a repelling force which increases in magnitude as the TCP approaches the limit. The force is generated when the TCP is within approximately 5 cm of a safety plane, or the deviation of the orientation of the tool is approximately 3 ° from the specified maximum deviation.

When a plane is defined as a *Trigger Reduced mode* plane and the TCP goes beyond this boundary, the safety system transitions into *Reduced* mode which applies the *Reduced* mode safety settings. Trigger planes follow the same rules as regular safety planes except they allow the robot arm to pass through them.

14.11.1 Selecting a boundary to configure

The `Safety Boundaries` panel on the left side of the tab is used to select a boundary limit to configure.

To set up a safety plane, click on one of the top eight entries listed in the panel. If the selected safety plane has already been configured, the corresponding 3D representation of the plane is highlighted in the `3D View` (see 14.11.2) to the right of this panel. The safety plane can be set up in the `Safety Plane Properties` section (see 14.11.3) at the bottom of the tab.

Click the `Tool Boundary` entry to configure the orientation boundary limit for the robot tool. The configuration of the limit can be specified in the `Tool Boundary Properties` section (see 14.11.4) at the bottom of the tab.

Click the / button to toggle the 3D visualization of the boundary limit on/off. If a boundary limit is active, the *safety mode* (see 14.11.3 and 14.11.4) is indicated by one of the following icons / / / .

14.11.2 3D visualization

The `3D View` displays the configured safety planes and the orientation boundary limit for the robot tool together with the current position of the robot arm. All configured boundary entries where the visibility toggle is selected (i.e. showing icon) in the `Safety Boundaries` section are displayed together with the current selected boundary limit.

The (active) safety planes are shown 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. A small arrow illustrates the side of the plane that does *not* trigger the transition into *Reduced* mode. If a safety plane has been selected in the panel on the left side of the tab, the corresponding 3D representation is highlighted.

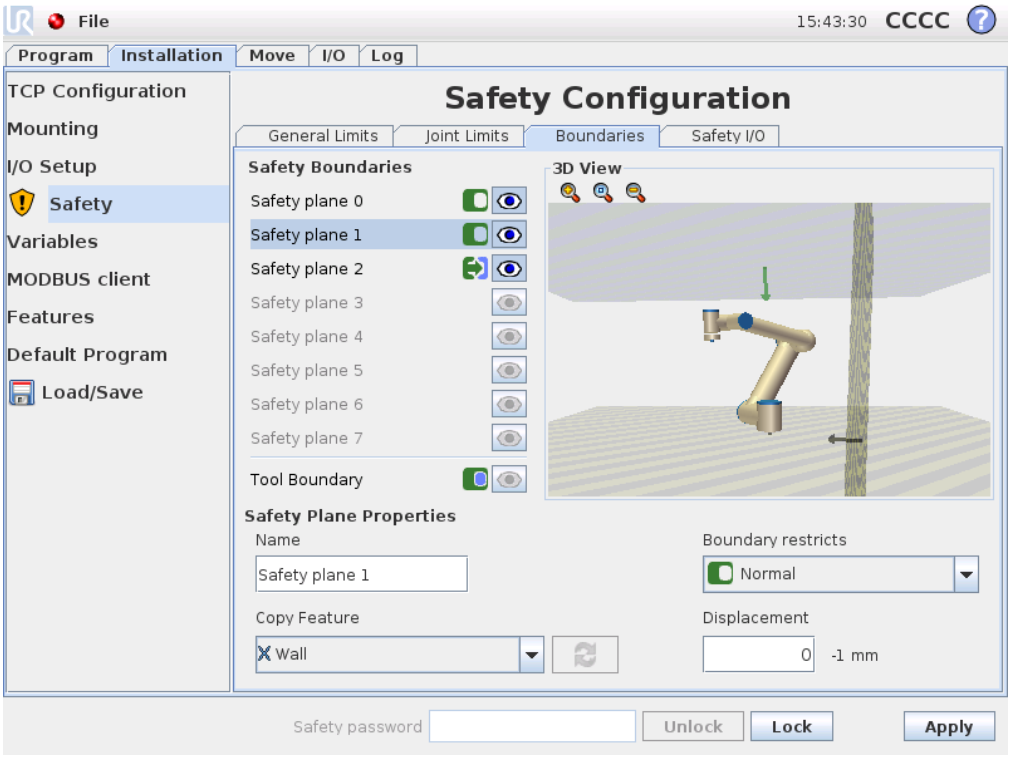
The tool orientation boundary limit is visualized with a spherical cone together with a vector indicating the current orientation of the robot tool. The inside of the cone represents the allowed area for the tool orientation (vector).

When a plane or the tool orientation boundary limit is configured but not active, the visualization is gray.

Push the magnifying glass icons to zoom in/out or drag a finger across to change the view.

14.11.3 Safety plane configuration

The `Safety Plane Properties` section at the bottom of the tab defines the configuration of the selected safety plane in the `Safety Boundaries` panel in the upper left portion of the tab.



Name The `Name` text field allows the user to assign a name to the selected safety plane. Change the name by tapping the text field and entering a new name.

Copy Feature The position and normal of the safety plane is specified using a feature (see 11.12) from the current robot installation. Use the drop-down box in the lower left portion of the `Safety Plane Properties` section to select a feature. Only the point and plane type features are available. Choosing the `<Undefined>` item clears the configuration of the plane.

The z-axis of the selected feature will point to the disallowed area and the plane normal will point in the opposite direction, except when the `Base` feature is selected, in which case the plane normal will point in the same direction. If the plane is configured as a *Trigger Reduced mode* plane (see 14.11.3), the plane normal indicates the side of the plane that does *not* trigger transition into *Reduced* mode.

It should be noted that when the safety plane has been configured by selecting a feature, the position information is only *copied* to the safety plane; the plane is *not* linked to that feature. This means that if there are changes to the position or orientation of a feature which has been used to configure a safety plane, the safety plane is not automatically updated. If the feature has changed, this is indicated by a ⚠ icon positioned over the feature selector. Click the 🔄 button next to the selector to update the safety plane with the current position and orientation of the feature. The ⚠ icon is also displayed if the selected feature has been deleted from the installation.

Safety mode The drop down menu on the right hand side of the `Safety Plane Properties` panel is used to choose the *safety mode* for the safety plane, with the following modes available:

`Disabled`	The safety plane is *never active*.
`Normal`	When the safety system is in *Normal* mode, a `Normal` mode plane is *active* and it acts as a *strict limit* on the position of the robot TCP.
`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 of the robot TCP.
`Normal & Reduced`	When the safety system is either in *Normal* or *Reduced* mode, a `Normal & Reduced` mode plane is *active* and it acts as a *strict limit* on the position of the robot TCP.
`Trigger Reduced mode`	When the safety system is either in *Normal* or *Reduced* mode, a *Trigger Reduced mode* plane is *active* and it causes the safety system to switch to *Reduced* mode for as long as the robot TCP is positioned beyond it.

The selected *safety mode* is indicated by an icon in the corresponding entry in the `Safety Boundaries` panel. If the *safety mode* is set to `Disabled`, no icon is shown.

Displacement When a feature has been selected in the drop down box in the lower left portion of the `Safety Plane Properties` panel, the safety plane can be translated by tapping the `Displacement` text field in the lower right portion of this panel and entering a value. Entering in a positive value increases the allowed workspace of the robot by moving the plane in the opposite direction of the plane normal, while entering a negative value decreases the allowed area by moving the plane in the direction of the plane normal.

The tolerance and unit for the displacement of the boundary plane are shown to the right of the text field.

Effect of *strict limit* planes Program execution is aborted when the TCP position is about to cross an active, strict limit safety plane minus the tolerance (see 14.3), if it continues moving along the predicted trajectory. Note that the minus sign displayed with the tolerance value is only there to indicate that the tolerance is subtracted from the actual entered value. The safety system will perform a category 0 stop, should the TCP position exceed the specified limit safety plane (without tolerance).

Effect of *Trigger Reduced mode* planes When no protective stop is in effect and the safety system is not in the special *Recovery* mode (see 14.5), it operates either in *Nor-*

mal or *Reduced* mode and the movements of the robot arm are limited by the respective limit set.

By default, the safety system is in *Normal* mode. It transitions into *Reduced* mode whenever one of the following situations occurs:

a) The robot TCP is positioned beyond some *Trigger Reduced mode* plane, i.e. it is located on the side of the plane that is *opposite to* the direction of the small arrow in the visualization of the plane.

b) The `Reduced Mode` safety input function is configured and the input signals are low (see 14.12 for more details).

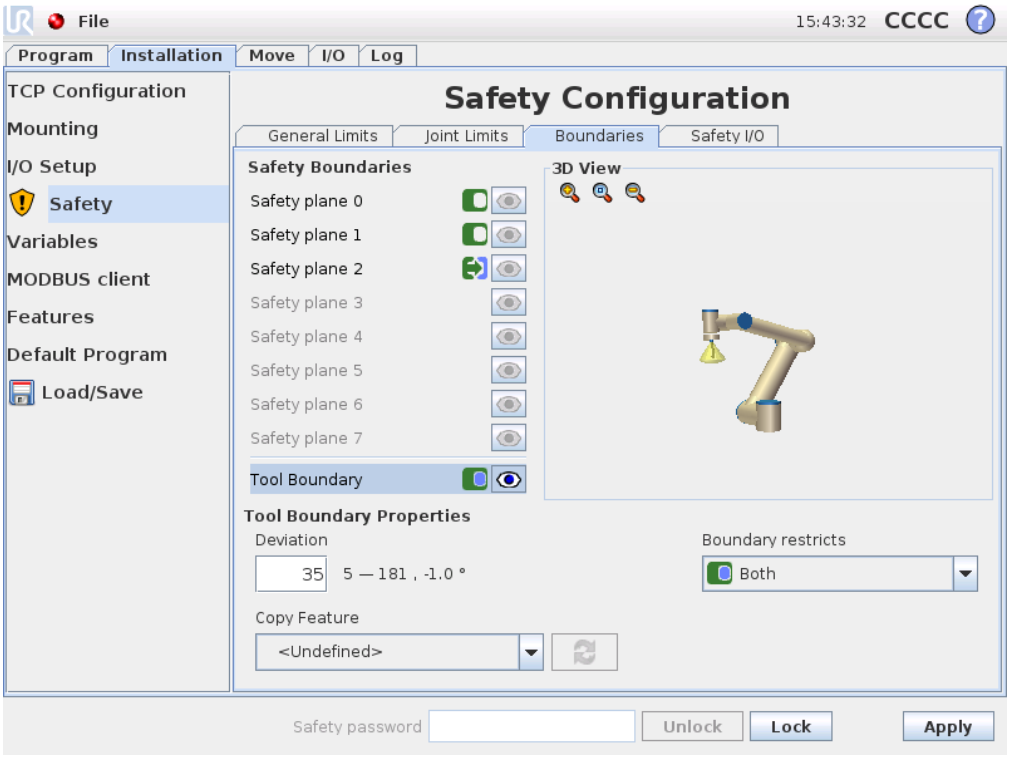
When none of the above is the case any longer, the safety system transitions back to *Normal* mode.

When the transition from *Normal* to *Reduced* mode is caused by passing through a *Trigger Reduced mode* plane, a transition from the *Normal* mode limit set to the *Reduced* mode limit set occurs. As soon as the robot TCP is positioned 20 mm or closer to the *Trigger Reduced mode* plane (but still on the *Normal* mode side), the more permissive of the *Normal* and *Reduced* mode limits is applied for each limit value. Once the robot TCP passes through the *Trigger Reduced mode* plane, the *Normal* mode limit set is no longer active and the *Reduced* mode limit set is enforced.

When a transition from *Reduced* to *Normal* mode is caused by passing through a *Trigger Reduced mode* plane, a transition from the *Reduced* mode limit set to the *Normal* mode limit set occurs. As soon as the robot TCP passes through the *Trigger Reduced mode* plane, the more permissive of the *Normal* and *Reduced* mode limits is applied for each limit value. Once the robot TCP is positioned 20 mm or further from the *Trigger Reduced mode* plane (on the *Normal* mode side), the *Reduced* mode limit set is no longer active and the *Normal* mode limit set is enforced.

If the predicted trajectory takes the robot TCP through a *Trigger Reduced mode* plane, the robot arm will start decelerating even before passing through the plane if it is about to exceed joint speed, tool speed or momentum limit in the new limit set. Note that since these limits are required to be more restrictive in the *Reduced* mode limit set, such premature deceleration can occur only when transitioning from *Normal* to *Reduced* mode.

14.11.4 Tool Boundary configuration



The `Tool Boundary Properties` panel at the bottom of the tab defines a limit on the orientation of robot tool composed of a desired tool orientation and a value for the maximum allowed deviation from this orientation.

Deviation The `Deviation` text field shows the value for the maximum allowed deviation of the orientation of the robot tool from the desired orientation. Modify this value by tapping the text field and entering the new value.

The accepted value range together with the tolerance and unit of the deviation are listed next to the text field.

Copy Feature The desired orientation of the robot tool is specified using a feature (see 11.12) from the current robot installation. The z-axis of the selected feature will be used as the desired tool orientation vector for this limit.

Use the drop down box in the lower left portion of the `Tool Boundary Properties` panel to select a feature. Only the point and plane type features are available. Choosing the `<Undefined>` item clears the configuration of the plane.

It should be noted that when the limit has been configured by selecting a feature, the orientation information is only *copied* to the limit; the limit is *not* linked to that feature. This means that if there are changes to the position and orientation of a feature, which has been used to configure the limit, the limit is not automatically updated. If the

feature has changed, this is indicated by a ⚠ icon positioned over the feature selector. Click the 🔄 button next to the selector to update the limit with the current orientation of the feature. The ⚠ icon is also displayed if the selected feature has been deleted from the installation.

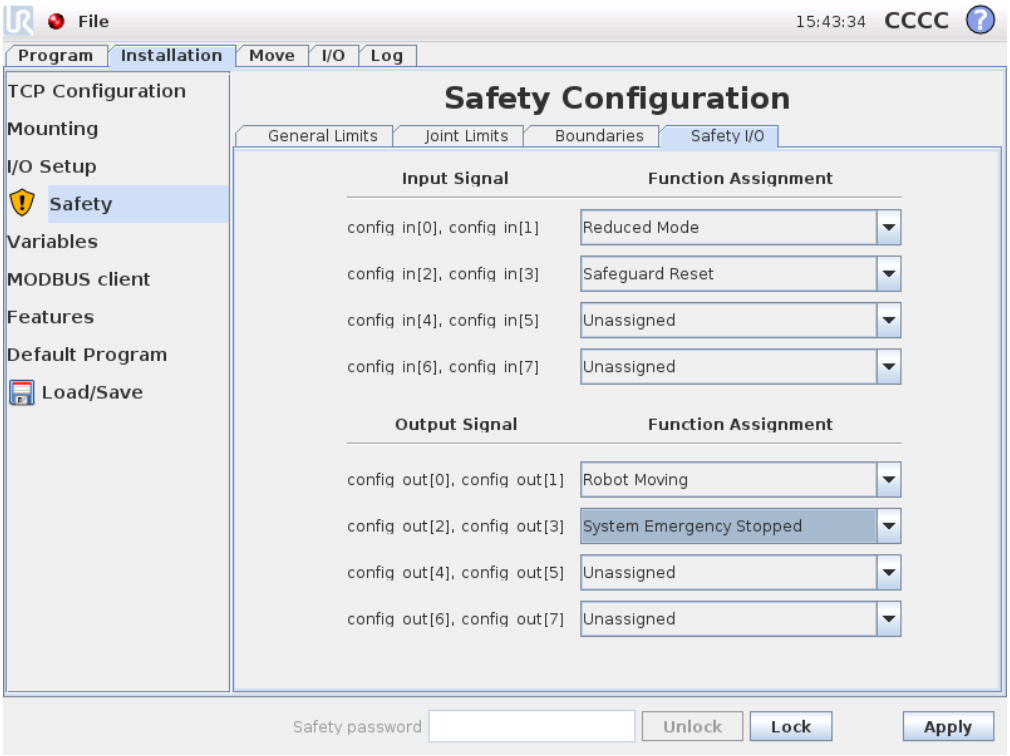
Safety mode The drop down menu on the right hand side of the `Tool Boundary Properties` panel is used to choose the *safety mode* for the tool orientation boundary. The available options are:

`Disabled`	The tool boundary limit is never active.
`Normal`	When the safety system is in *Normal* mode, the tool boundary limit is active.
`Reduced`	When the safety system is in *Reduced* mode, the tool boundary limit is active.
`Normal & Reduced`	When the safety system is either in *Normal* or *Reduced* mode, the tool boundary limit is active.

The selected *safety mode* is indicated by an icon in the corresponding entry in the `Safety Boundaries` panel. If the *safety mode* is set to `Disabled`, no icon is shown.

Effect Program execution is aborted when the deviation of the tool orientation is about to exceed the entered maximum deviation minus the tolerance (see 14.3), if it continues moving along the predicted trajectory. Note that the minus sign displayed with the tolerance value is only there to indicate that the tolerance is subtracted from the actual entered value. The safety system will perform a category 0 stop, should the deviation of the tool orientation exceed the limit (without tolerance).

14.12 Safety I/O



This screen defines the *Safety functions* for configurable inputs and outputs (I/Os). The I/Os are divided between the inputs and outputs, and are paired up so that each function is provided a Category 3 and PLd I/O for safety (in the event one of the I/Os are to be no longer reliable).

Each *Safety function* can only control one pair of I/Os. Trying to select the same safety function a second time removes it from the first pair of I/Os previously defined. There are 3 *Safety functions* for input signals, and 4 for output signals.

Input Signals For input signals, the following *Safety functions* can be selected:

- `Emergency Stop`: When selected, this allows the option of having an alternative `Emergency Stop` button in inclusion of the one that is on the Teach Pendant. This will provide the same functionality that the `Emergency Stop` button provides on the Teach Pendant when a device complying with ISO 13850:2006 is attached.
- `Reduced Mode`: All safety limits have two modes in which they can be applied: *Normal* mode, which specifies the default safety configuration, and *Reduced* mode (see 14.5 for more details). When this input safety function is selected, a low signal given to the inputs causes the safety system to transition to *Reduced* mode. If necessary, the robot arm then decelerates to satisfy the *Reduced* mode limit set.

Should the robot arm still violate any of the *Reduced* mode limits, it performs a category 0 stop. The transition back to *Normal* mode happens in the same manner. Note that safety planes can also cause a transition to *Reduced* mode (see 14.11.3 for more details).

- `Safeguard Reset`: If `Safeguard Stop` is wired in the safety I/Os, then `Safeguard Reset` is used to ensure the Safeguard Stopped state continues until a reset is triggered. The robot arm will not move when in 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 For the output signals the following *Safety functions* can be applied. All signals return to low when the state which triggered the high signal has ended:

- `System Emergency Stop`: Low signal is given when the safety system has been triggered into an `Emergency Stopped` state. It is in a high signal state otherwise.
- `Robot Moving`: A low signal is given whenever the robot arm is in a mobile state. When the robot arm is in a fixed position, a high signal is given.
- `Robot Not Stopping`: When the robot arm has been requested to stop, some time will pass from the request until the arm stops. During this time the signal will be high. When the robot arm is moving and has not been requested to stop, or when the robot arm is in a stopped position, the signal will be low.
- `Reduced Mode`: Sends a low signal 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.