USER MANUAL

INOVO ROBOTICS MODULAR ARM

Author	Henry Wood	Doc. Ref.	URD-002
Audience	User Documentation	Date	15 Jan 2023
Classification	Public	Version	5.6.0 v19
Owner	Henry Wood	Status	PRE-RELEASE

Product version 5.6.0

Introduction

This manual describes the hardware setup and software setup and operation of the Inovo modular arm.

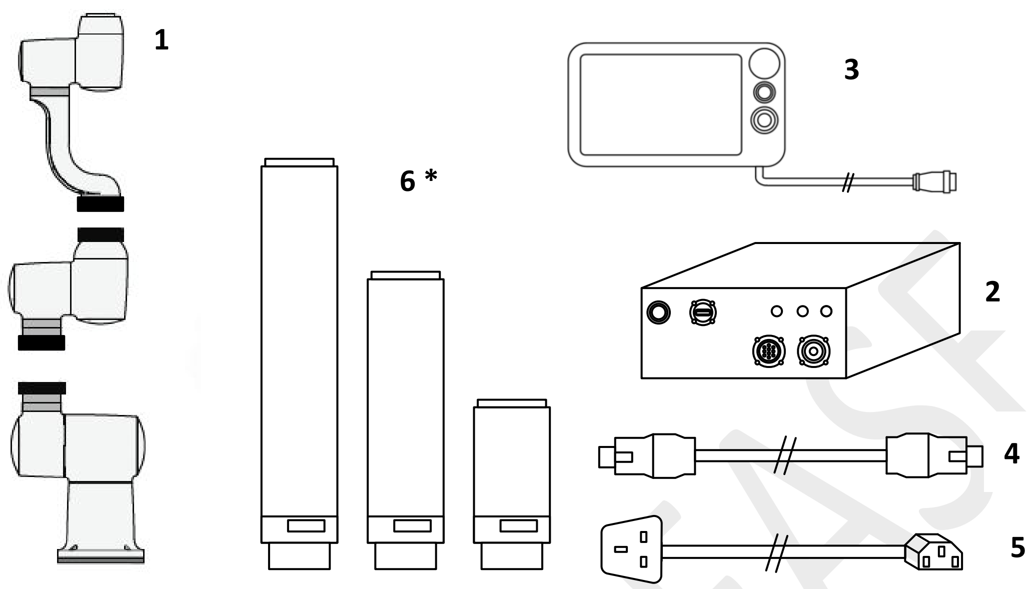
Software versions

RCU	V8.5.4
Engineering UI	V2.6.2
Pendant UI	v0.8.2
PSU Controller	v0.0.5
Joint Controllers	v2.00
Wrist Controllers	v0.1.15

CONTENTS

1. HARDWARE

1.1. ROBOT SYSTEM



1. Robot joint modules (Shoulder J1-2, Elbow J3-4, J-link wrist J5-6)
2. Control and Power Unit (RCU)
3. Pendant
4. Robot Cable
5. IEC Mains Power Cable
6. Link Tubes. (120mm, 300mm & 470mm)

*The number of link tubes included is dependant on which configuration was purchased.

** you will also need a PC or laptop with Google Chrome browser.

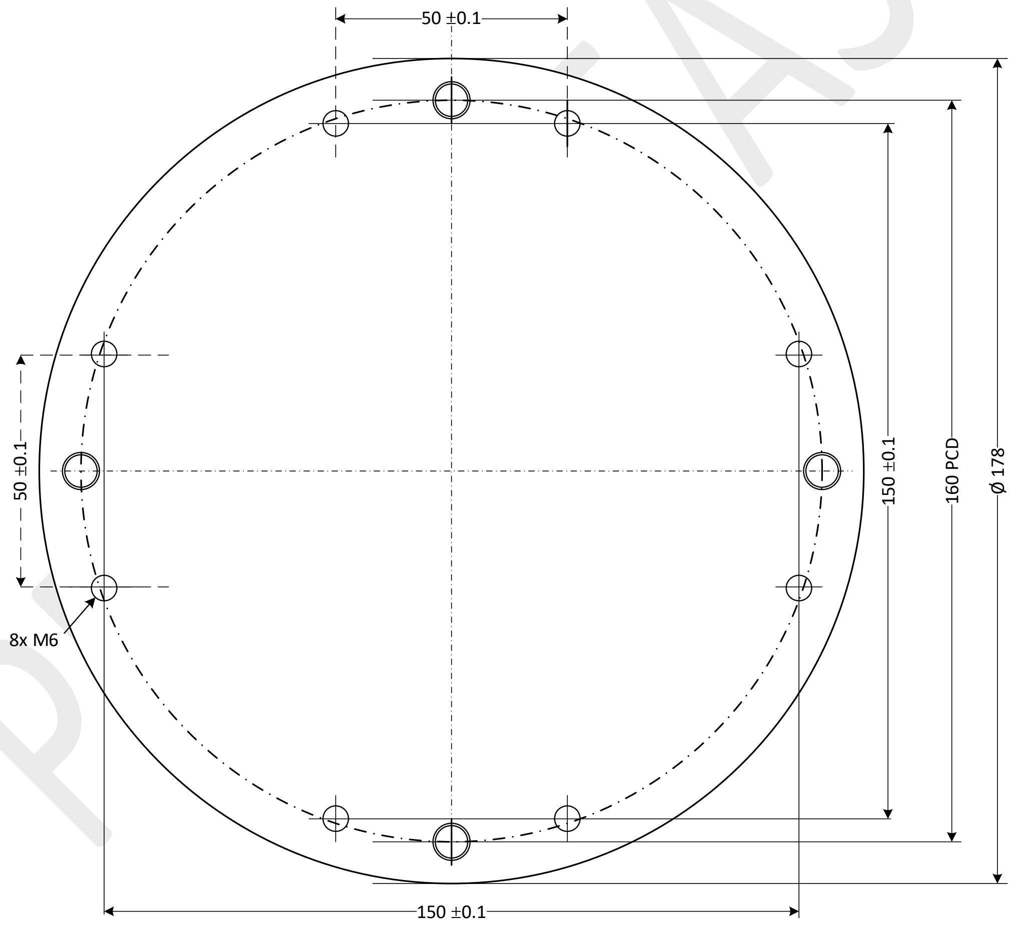
1.2. Mounting the Robot

The robot must be securely mounted to a strong, rigid surface. When lifting loads with the arm outstretched or accelerating between different positions the arm can put significant forces on the mounting surface so it should be fixed down and unable to move.

Careful consideration should be applied when choosing the point to locate the base of the robot. Make sure the space around the robot is clear and that the arm will be able to reach all the required points without being fully outstretched.

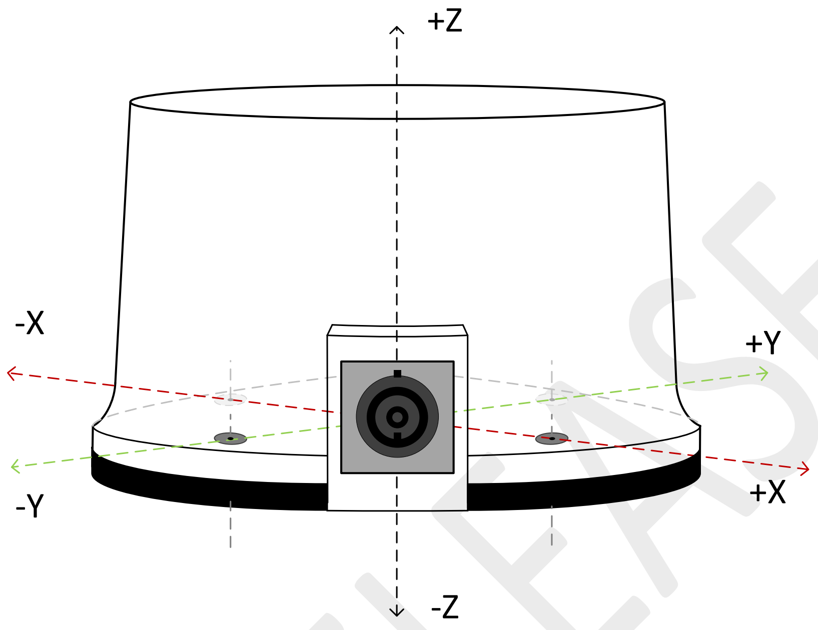
The base plate is fixed to the mounting surface by 8 x M6 cap head screws. The lower part of the first joint is then fixed to this plate using 4 x M8 cap head screws which are supplied with the robot. There should be tightened to 30Nm.

The drawing below shows the centre spacing for the M6 bolts used to secure the base plate.



1.2.1. Robot Axis relative to the mounting plate

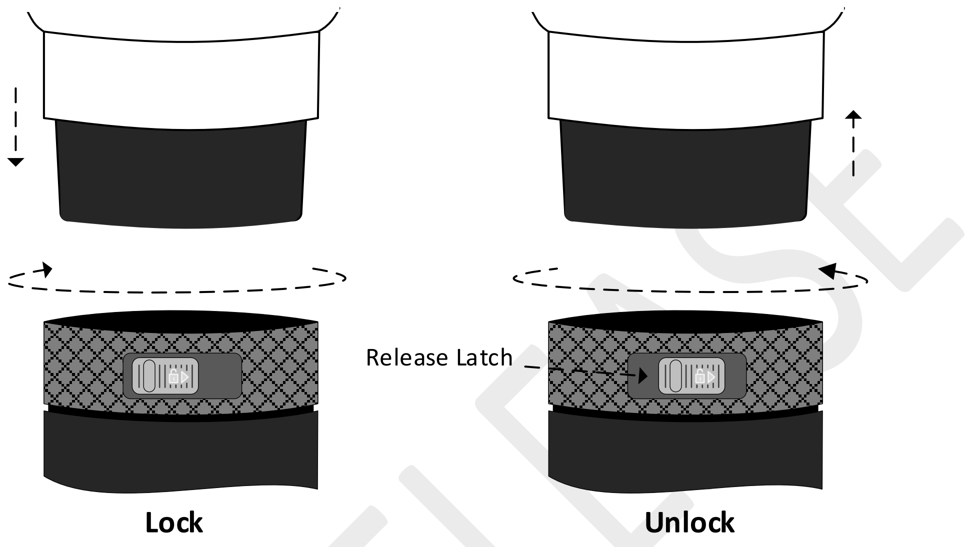
The base / world frame are defined relative to the bottom centre or the mounting plate passing through the centre of the four mounting blots as shown below:



1.3. Modular Coupling

When connecting or disconnecting the

it is recommended that the module being removed is directly above the one it is coupling or decoupling to. This will make it easier to lift in or out and limit the stress on the mechanism.

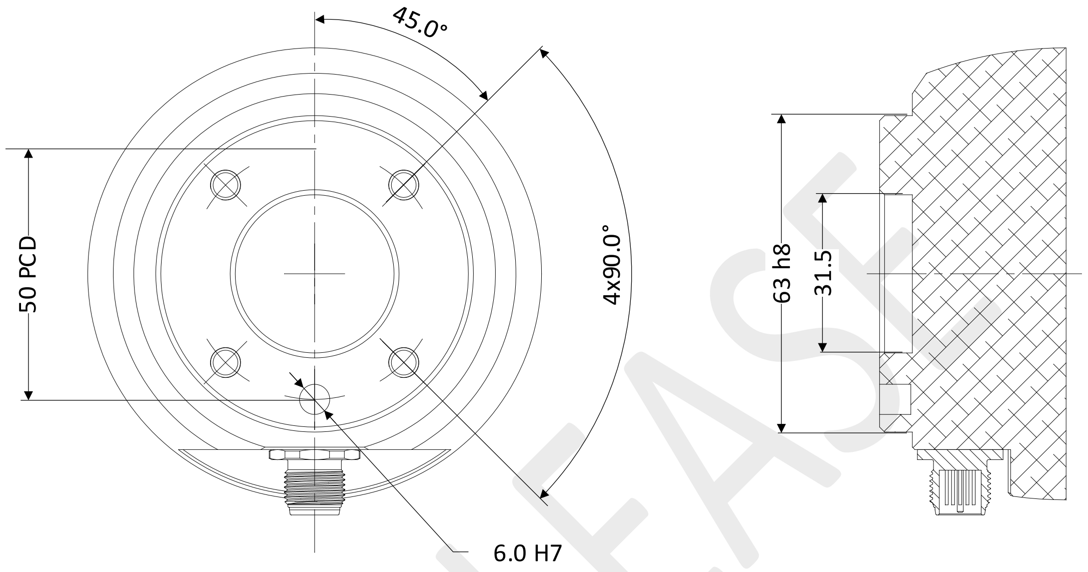


To connect the modules carefully align the locating pins and gently lower the upper module into the lower while turning the locking ring clockwise. Continue tightening until the ring cannot turn any further.

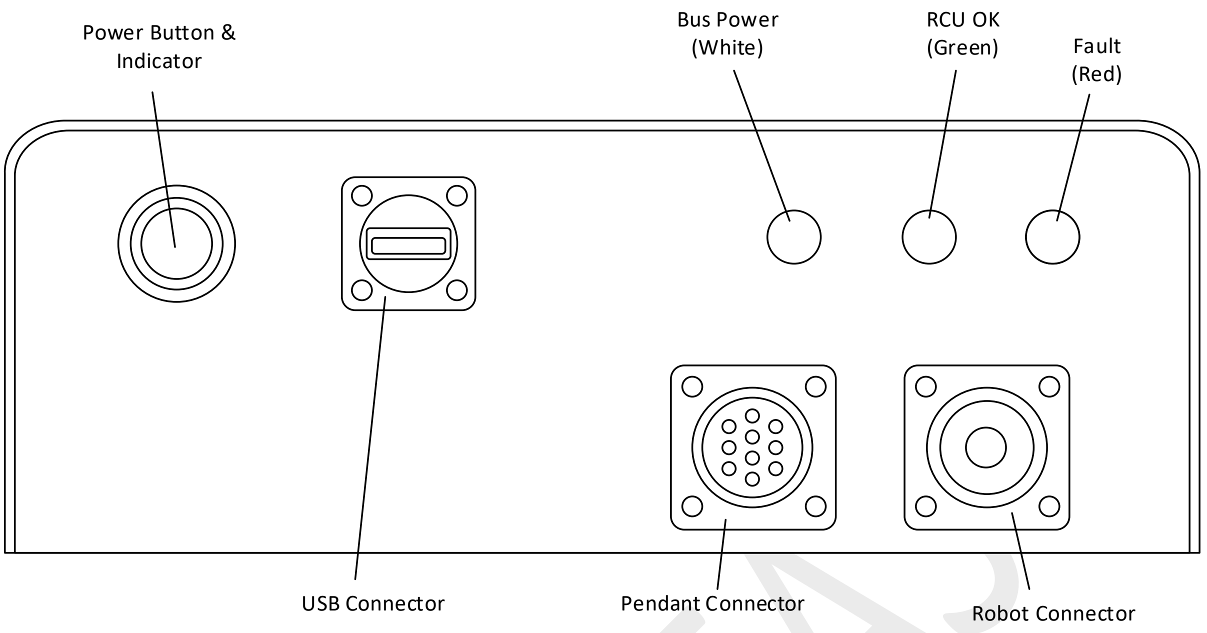
To disconnect the modules, support the upper module and turn the locking ring anticlockwise while holding the release latch to the right. As the upper module is released lift it upward supporting its weight

1.4. End effectors and Tools

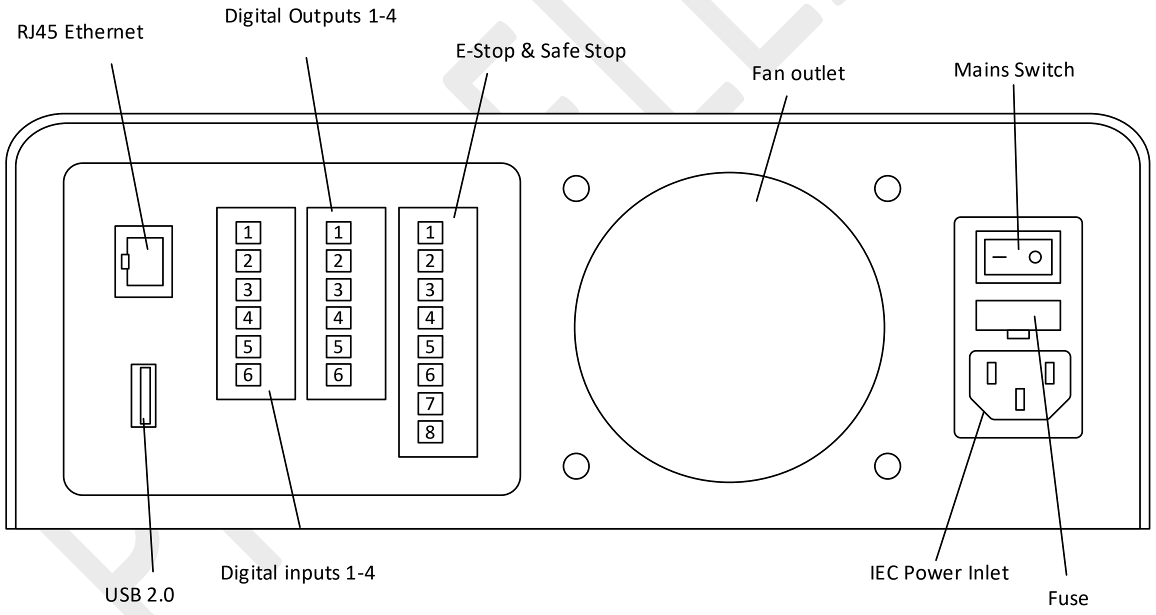
Mechanical mounting – The tool output flange is based on standard ISO 9409-1-50-4-M6. 4 x M6 screws allow common end effectors to be quickly and securely fixed to the end of the arm. A 6mm dowl pin provides accurate location and indexing. See the Wrist IO section for electrical interface details.



1.5. Control box



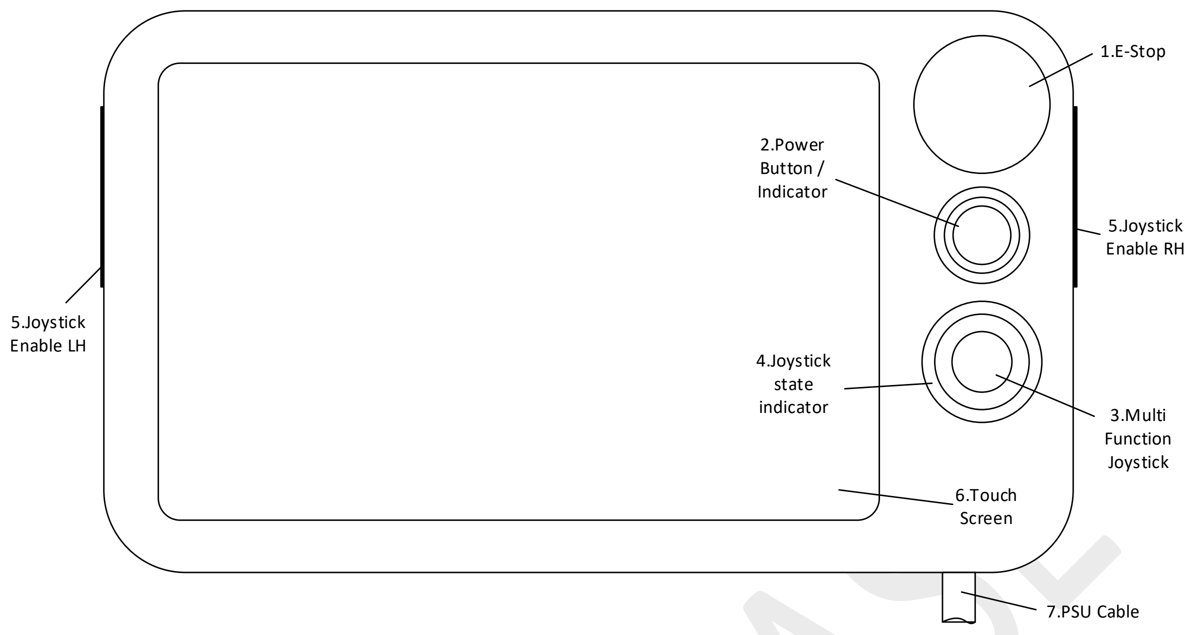
Rear Panel



Make sure the fan outlet is not covered and air can move freely from it

1.6. Pendant

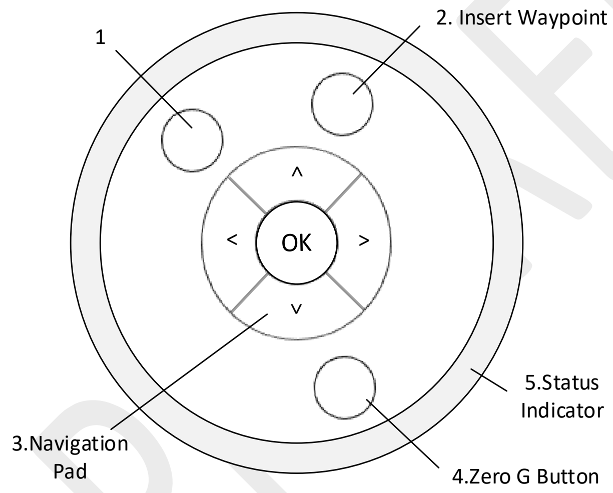
The pendant provides a simple touch screen interface to control basic functions of the robot. It includes a power button, emergency stop button and multi-function analogue joystick


1.E-Stop
2.Power Button / Indicator
5.Joystick Enable RH
5.Joystick Enable LH
4.Joystick state indicator
3.Multi Function Joystick
6.Touch Screen
7.PSU Cable

The pendant is connected directly to the robot control unit via a 3m tether cable

1.7. Wrist Interface

.



Status Indicator Colours

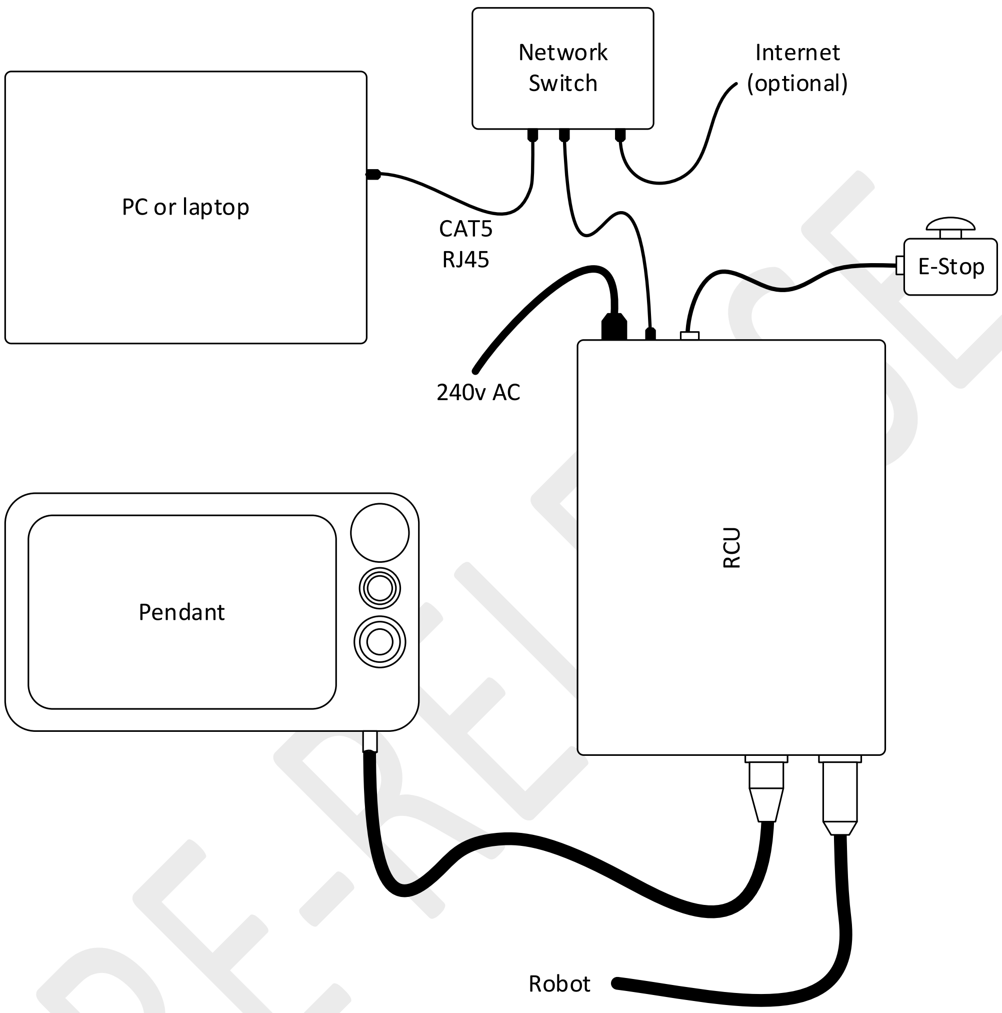
- Flashing Yellow – Initializing / Comms Lost
- Yellow – Arm Disabled
- Green – Arm Enabled
- Blue – Zero G Mode
- White – Running
- Purple – Jog mode
- Red –Safe-Stop

1.7.1. Free / Zero G Mode

With the arm enabled press and hold the Zero G button to drag the arm in free mode

1.8. Connecting the system

To connect the system, you will need a network with DHCP support. Connect the RCU to the network switch using a CAT5 RJ45 cable that is connector to a PC or Laptop with a chrome browser installed.



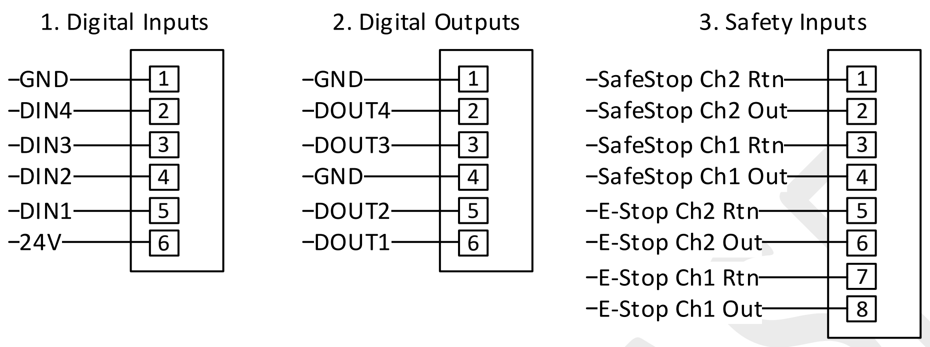
Connect a dual channel (NC NC) E-Stop to the Screw terminal J18 on the back panel. On the front panel connect the pendant or the E-Stop link if you are not using the pendant. Connect the robot to the connector on the front panel.

Connect the IEC socket on the back panel to a 240V AC Mains outlet. Do not connect any of these cable with the robot or RCU power on.

2. ELECTRICAL INTERFACE

2.1. Control Box

The control box has a set of inputs and outputs that can be monitored and controlled from the user defined software, this provides an interface to connect a wide range of third-party sensors, actuators, alarms etc.



2.2. Control Box - Digital IO

The IO available on the back panel is connected to a set of an internal EtherCAT Slices. These can be expanded as required and the rear connector panel replaced with a custom panel to meet user requirements.

The digital inputs are 24v isolated inputs provided by a Beckhoff EL1008 EtherCAT slice. They are tolerant of voltages 0-30v. A voltage below 5V will read as a low and above 11V will be read as a high signal.

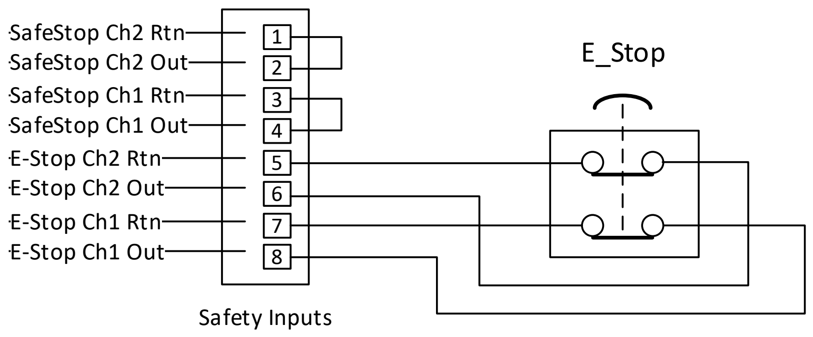
The digital outputs are provided by a Beckhoff EL2008. They provide 24v with a max current of 500mA and are short circuit protected, the internal 24v supply has a maximum current rating of 4A so the total load on all outputs combined should not exceed this. It is however possible to wire an external 24v supply to power the IO if required.

2.3. Control Box - Safety Circuits

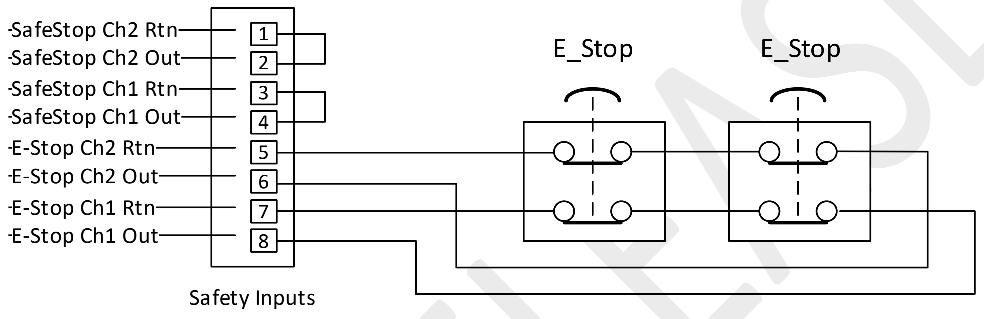
The robot controller includes dual channel E-Stop and dual channel Safe Stop circuits. If either channel of the E-Stop circuit is not complete the power will be removed from the arm and the mechanical brakes applied to all joints. If either of the Safe Stop circuits are not complete, then the motor drives will be disabled and the mechanical brakes applied but the power will remain present in the arm and to the tool connector. The Pendant forms part of the E-Stop Circuit. If an external E-Stop is installed the Pendant E-Stop will remain part of the E-Stop circuit and must be in the released position for the arm power to be switched on. Note: If the pendant is not fitted a loopback connector should be inserted in the pendant socket on the front panel of the PSU to bypass the pendant E-Stop or the arm will not enable.

External E-Stop and/or Safe Stop switches can be wired into the system via the safety input terminals on the rear panel of the control box. You must use dual channel, normally closed, latching type switches, eg. RS part number 417-5401.

Extenal E-Stop

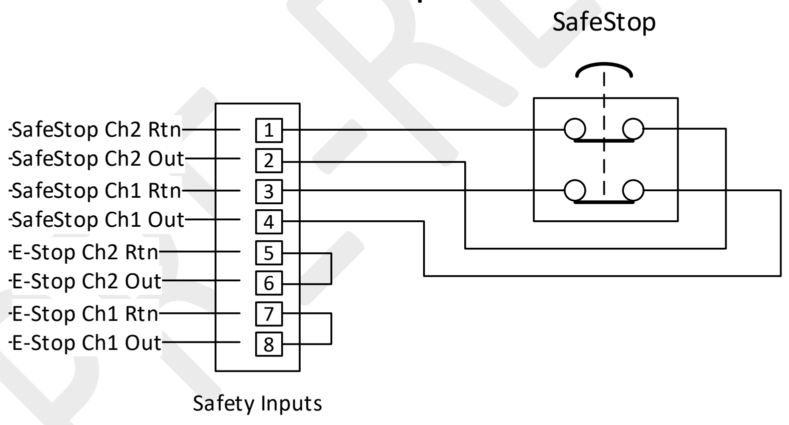


Multiple E-Stops



Multiple E-Stops can be chained in series if required. Never wire E-Stop switches in Parallel.

Extenal SafeStop



External Safe Stop switches can be wired in a similar way, connect to each channel as shown above.

If External E-Stop or Safe Stop switches are not required in your application you can use wire links to make the circuit permanently enabled.

Always conduct a safety risk assessment for your specific application installation.

2.4. Tool / Wrist Connections IO

The tool has a set of user IO available via a standard M12 connector on the side of the wrist. This provides the following:

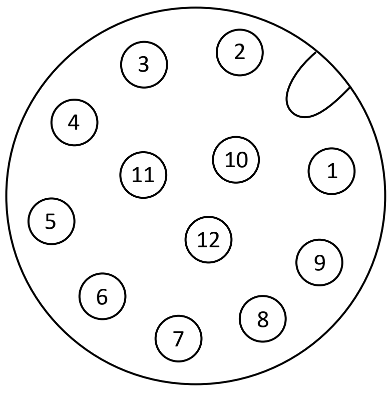
1 x 2A, 24V supply.

2 x Digital Inputs 24V.

2 x 24V Digital Outputs – 24V, 2A sink capability.

1 x RS485 port with Modbus RTU support for selected third party grippers.

2.4.1. Wrist Connector Point out



Amphenol Part #
M12A-12BFFA-SL7001

1 - Reserved	
2 - Digital Out 1	
3 - Reserved	
4 - Reserved	
5 – Digital Out 0	
6 - 24V, 2A supply	
7 – 0V, supply	
8 – Digital In 1	
9 – Digital In 0	
10 – RS485_Gnd	
11 – RS485_B	
12 – RS485_A	

3. SOFTWARE

3.1. Software Introduction

The pendant provides a simple interface to run a prebuild program and perform basic arm movements, the main software interface is accessed via a web browser running on a PC or laptop connected via a local network.

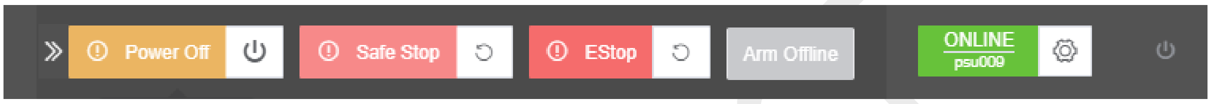
The system can be programmed to run a sequence of arm movements and interact with other hardware via electrical IO or communications interfaces. This section describes various ways to define robot motion, generate outputs and act on inputs signals. The basic principal behind programming the robot motion is to define a set of waypoints in a sequence, manually moving the robot to these positions and storing the position. The type of path, blend, velocity, and acceleration between these waypoints can then be specified in the same sequence as additional parameters.

3.2. Starting the system

Power on the robot by pressing the round power button of the front panel of the control box or pendant. The RCU (Robot control unit) takes 10-15s to boots, when the RCU has booted the power button LED will go steady blue and the green LED on the front panel will blink twice every second to show the RCU is running normally. If the RCU fails to start for any reason the red LED will blink rapidly. If this happens hold the power button for 5s to force a *'hard shutdown'*. If the problem persists, contact your supplier for support.

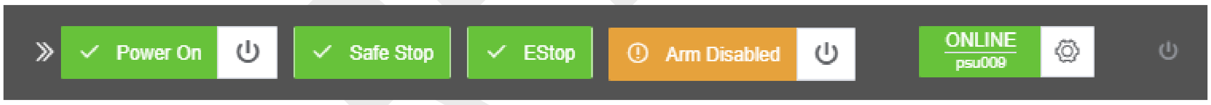
Once the RCU has booted you can connect to the user interface using a browser running on a laptop or PC that is connected to the same network subdomain. We strongly recommend that this connection is wired ethernet and not Wi-Fi as wireless signals can be intermittent and reliable communications between the RCU and UI are essential for responsive operation. Open a Google Chrome browser and enter your robots IP or hostname. Depending on your network DHCP settings the robot should be allocated an IP address associated with its hostname. The default host name for your robot is the psu[xxx] where [xxx] is the serial number of your PSU written on the rear panel, i.e. http://psu047 if your serial number is '047'. The IP address can be found by checking the bottom of the pendants home screen or by checking your routers DHCP table.

The top right area of the toolbar should show the status 'ONLINE' and the robot name underneath in smaller text.



If this shows a 'CONNECTION ERROR' check the robot and PC you are using are connected to the same network and there is no firewall or subnet restrictions.

With both the safe stop and E-Stop circuits complete, ie switches not pressed, press the Safe Stop and Estop button on the top right to clear the safety latched ,they will turn green and you are then able to Power On the arm with the button on the left. When the robot power is turned on you should hear the load dump fan run for about 5 seconds then stop, the white light will illuminate on the front panel of the PSU to indicate bus power is on and the wrist status indicator will blink yellow while the arm configuration is detected. After approximately 5 seconds the wrist indicator should go to steady red if the Safe-Stop is pressed or steady Yellow if it is released to indicate the arm is in the on but disabled state.



With the Safe-Stop is released click on the 'Arm Disabled' button in orange, you will now hear a clicking sounds as the joint brakes release. The wrist status indicator will light green to show the arm is enabled and ready for motion. The robot can now be moved using ZeroG mode, Jog control or the joystick.

Click 'Arm Enabled' in the toolbar to disable the robot and engage the brakes when you do not need to move the arm. You can press the E-Stop or Safe-Stop to stop the arm moving at any time. Make sure an E-Stop or Safe-Stop button is always nearby and accessible when moving the arm or running a sequence.

3.3. Shutdown

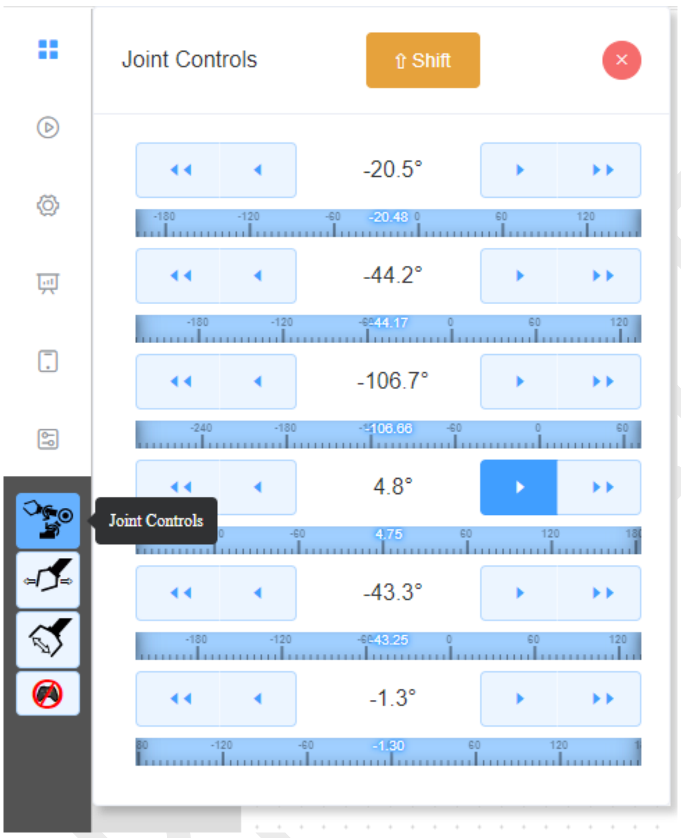
Click the 'power off' button on the far right of the toolbar to shut down the system , you will be prompted to confirm you want to shut down the system, click yes to continue, they system takes 10-15 to shutdown and can only be restarted by pressing the physical power button on the RCU or pendant.

3.4. Moving the robot

There are three main ways to move the robot, the first is by using on screen '*Jog'* arrows that allow you to move individual joints or the tool position and orientation in a desired axis. These *Jog* controls and are useful for fine control and small movements that make it easy to align the robot tool position and orientation with objects in the real-world environment around the robot. The second way to move the robot is in 'free' or 'zero gravity' mode which allows you to physically drag the arm to a desired position by simply holding the robot at the wrist and pushing or pulling it to a new pose. Free mode is useful to quickly reposition the robot when changing from a distant position and works well in combination with *jog* movements to make the fine adjustment at the end. You cannot use both at the same time, however, attempts to use the *Jog* movements when the arm is in *free* mode will be ignored. The third method is to use the joystick on the pendant. This is a analogue joystick so provides fine tactile control individual joints or the tool position depending on the mode.

3.5. Joint Jog Controls

To move the individual joints in the robot, click on the 'Joint Controls' button on the left side toolbar, this will show the 'Joint Control' panel which list the position of each joint in order. Press and hold the single arrow to rotate the joint in the desired direction slowly, the joint will move while the button is held and stop as soon as it is released. Use the double arrows to move the joint at a faster speed. You can also hold the shift key to limit the speed if you need to make fine changes.



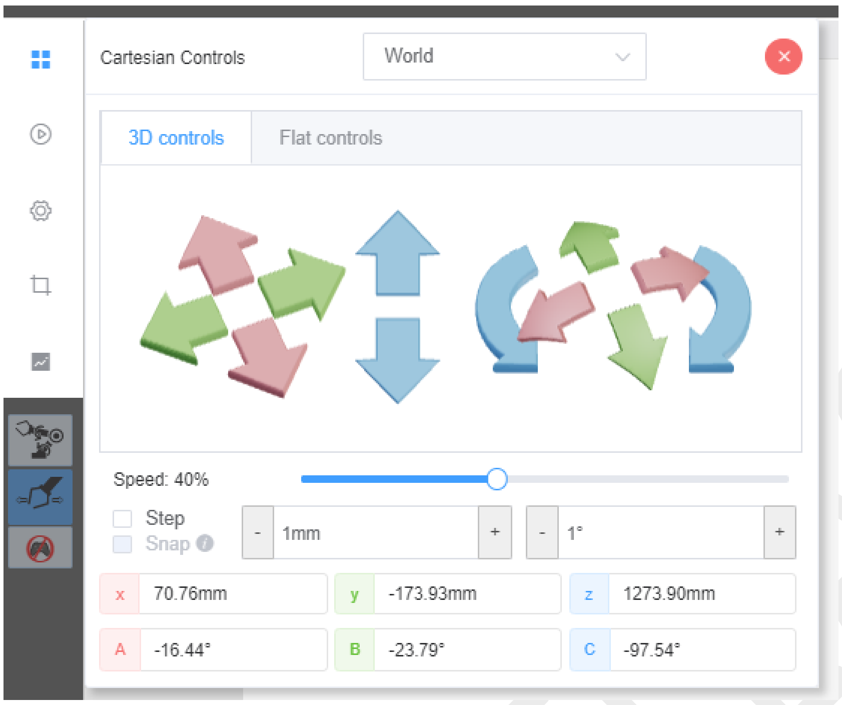
3.6. Cartesian Jog Controls

To move the robot tool centre point (TCP) in a cartesian axis X, Y or Z you can use the cartesian jog controls. Press the 'cartesian controls' button on the left side toolbar to show the cartesian controls panel. This panel has six linear arrows and six rotation arrows, the six linear arrows correspond to +X, -X, +Y, -Y, +Z, -Z, the red arrows represents the X axis as marked in the 3D visualisation, green represents Y and blue Z.

Cartesian motion are always in reference to a particular frame, by default the 'world frame is used which is the centre of the robots base plate. The drop down at the top of this panel allows you to change the reference frame between base, tool and any other user defined frames.

The speed slider controls the speed at which the TCP will move when the jog buttons are pressed.

You can define a step size and tick the step check box to limit the travel that each arrow press can cause. Set the step to 2mm and tick 'Step' for example, then press and hold the blue arrow, the robot will move a maximum of 2mm in the Z axis regardless of how long the arrow is held now.

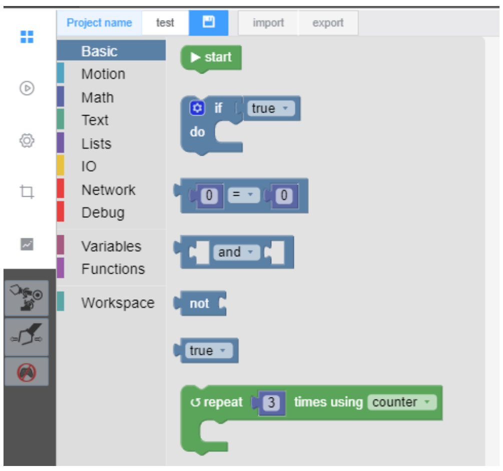


3.7. Free / Zero Gravity mode

To enter free mode where the robot can be hand guided press and hold the free mode button on the wrist keypad with the arm enabled. When the robot is in free mode you can physically drag the end of the robot to your desired position by pushing or pulling the wrist. Release the 'free' button at any time to exit free mode.

Warning : Make sure you have correctly configured the payload mass and offset in the settings before using free mode, incorrectly configured mass and offset could cause expected motion with the arm sinking or rising when the user is not intentionally dragging it.

3.8. Creating a program using function blocks



To create a program that can be replayed use the Program view, c lick the Program button on the left side bar to switch to this view.

Blockly allows you to construct programs that control the robot by dragging function block form the toolbox on the left. These function blocks plug together like jigsaw pieces to create a chain of blocks. When you run the program the block immediately below the start block gets run first then the following blocks below that in order of sequence.

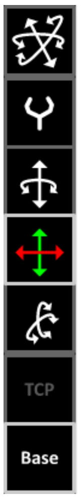
The toolbox groups functions to make it easier to navigate, drag a block from the toolbox on to the canvas to the right and place it inside the start block.

3.9. Pendant Joystick

The joystick on the pendant can operate in various modes which are selected using the buttons along the right side of the touch screen. The joystick is only active when the arm is enabled (green status indicator on the robots wrist), and one of the joystick enable buttons is held down on the side of the pendant.

The top button switches between the cartesian and joint space menu, the table below shows the cartesian menu. The four buttons below the menu mode select different joystick functions when in cartesian mode.

The three buttons at the bottom select the frame for the robot to move in. Either Tool, Base or the current 3D visualisations frame. This allows the operator to align the 3D view with their current view of the robot, so the joystick is mapped to the same axis from where they are currently standing.



Toggle Cartesian / Joint mode

Gripper Control (disabled)

Z Linear / Rotate Z

X / Y Linear

X Rotate / Y Rotate

Tool Frame

Base Frame

Z Linear / Rotate Z – This mode allows the operator to move the TCP (tool centre point) along the Z axis using Up/Down axis of the joystick in the selected frame. If the 'Base' frame is selected this will move the TCP up and down relative to the surface the robot is mounted on. If the 'TCP' frame is selected it will move the tool tip along the axis projected out of the tool in its current pose. This is useful when inserting or removing parts. Moving the joystick left and right rotates the tool relative to the select frame.

X / Y Linear – This mode moves the tool in the X plane when the joystick is deflected left and right and the Y plane when deflected Up and Down. If the 'Tool' Frame is select this will move the tool relative in the axis projected perpendicular to the tool mounting place or any customed defined user TCP. If the 'Base' frame is selected this will be the X and Y axis as the robot is mounted to the table.

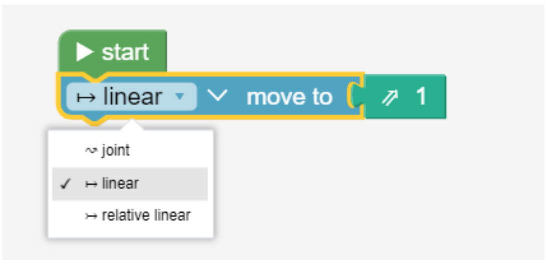
X Rotate / Y Rotate – This mode rotates the tool around the Y plane when the joystick is deflected left and right and the X plane when deflected Up and Down.

To move individual joint press the top button to toggle to Joint mode, you will then have a list of the joints (typically 1-6), select the required joint and move the joystick left to right to rotate the joint clockwise/anticlockwise.

3.10. Creating motion paths

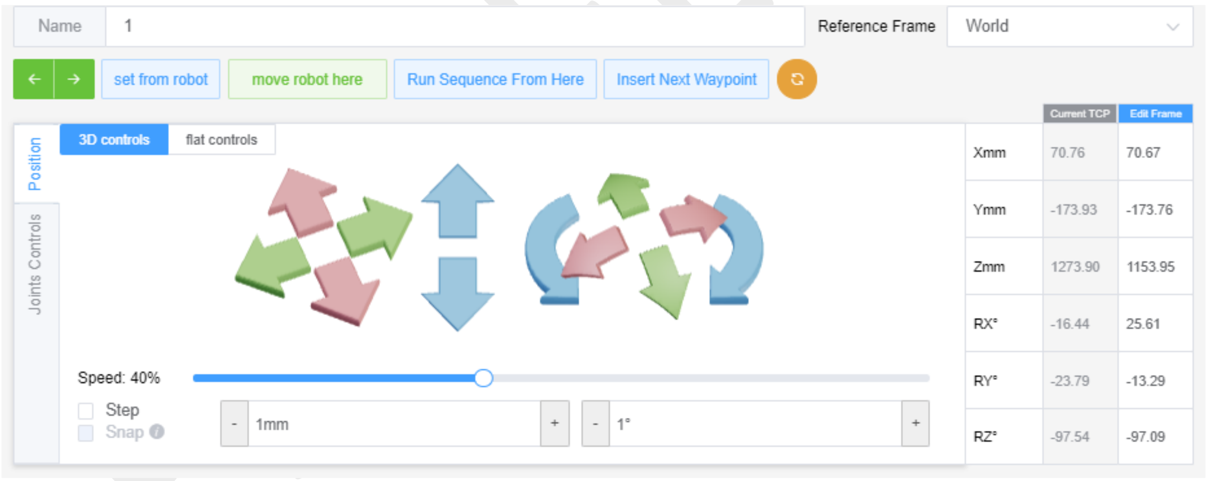
To create a motion path, you need to use one of the 'move to' blocks in the 'Motion' toolbox. Once placed on the canvas the type of motion can be selected. A **joint move** will travel to the specified target position in joint space. This means the path the tool centre point takes will be an arc rather than a straight line, but the path is more efficient so the motion will execute more quickly, and the robot will not encounter singularities.

In contrast a **linear move** will generate a straight line path to the target waypoint but will typically take more time to execute and is susceptible to singularities.



Click on the block labelled 'new waypoint' to edit the waypoint, this will open the waypoint properties panel in the top left quarter of the window.

This control offers jog control for both position (cartesian) and joint control modes selected via the tabs on the left of the panel. The field at the top labelled **Name** allows you to give the waypoint a friend name for example 'First Pick point'. We recommend you name all waypoints with a descriptive label to make the program easier to follow later.



Name: 1

Reference Frame: World

Speed: 40%

Step

Snap

1mm

1°

	Current TCP	Edit Frame
Xmm	70.76	70.67
Ymm	-173.93	-173.76
Zmm	1273.90	1153.95
RX°	-16.44	25.61
RY°	-23.79	-13.29
RZ°	-97.54	-97.09

The table on the right shows the current and target tool centre point values. The centre column is the actual, the right is the target.

The blue **Set from Robot** button captures the robot's current position as the new target for this waypoint.

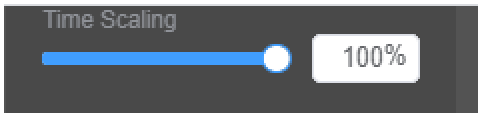
The green **Move Robot Here** button moves the real robot toward the waypoint specified in this block while the button is held. If you want to adjust a previously created waypoint it is useful to hold the 'move robot here' button until the robot is at the old waypoint then adjust the robots position using the jog controls and finally update the waypoint to the new robot position by pressing 'Set from Robot'

3.11. Modifying motion paths speed and acceleration

There are several different ways to modify the speed of a motion path:

- Time Scaling – A master scaling in the toolbar that can be changed at run time
- Modifying the speed settings of a block by placing it inside a speed constraint block

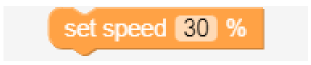
The master **time scaling** in the top tool bar can be used to slow down the program for debug and testing purposes. If you specify a motion to have a speed of 100mm/s and set the master scaling to 0.5 then the motion will playback at 50mm/s. This setting can be changed during playback.



The arm's speed and acceleration can be modified using the 'Set Speed' and 'Set TCP Speed' blocks. These blocks set the limits for the Joints and linear tool speed. They update the limit for the remainder of the sequence of until changed again with a new value by the same block.

Note: Either joint or tool speed can limit the motion at any time.

The set speed block limits each joint to a percentage of its max rated speed.

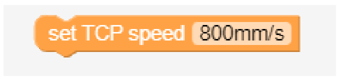


By default, the joints are limited to 50% of there rated speed so if no 'Set Speed' block is used in your sequence they will be limited to 50% of the values below

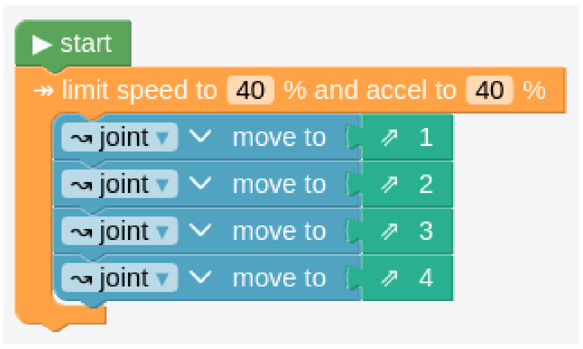
Max rated joint speed and accelerations:

Joint	Radian	Degrees	Acceleration
1	1.8 Rad/s	103 °/s	
2	1.8 Rad/s	103 °/s	
3	2.5 Rad/s	143.3 °/s	
4	2.5 Rad/s	143.3 °/s	
5	3.3 Rad/s	190 °/s	
6	3.3 Rad/s	190 °/s	

The Set TCP speed block specifies a linear TCP speed limit, this can be set from 0 - 1,000mm/s by default this is 1,000 mm/s.

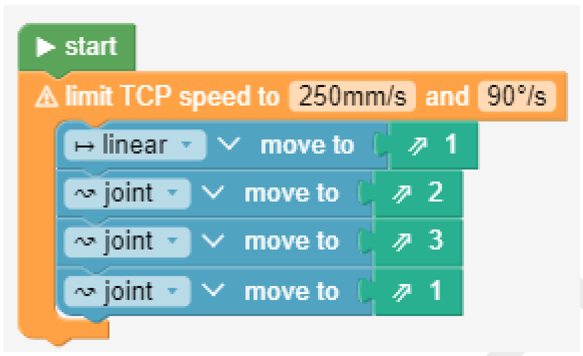


In the motion advanced toolbox there are also limit groups which apply the limitation to a subset of motion block.



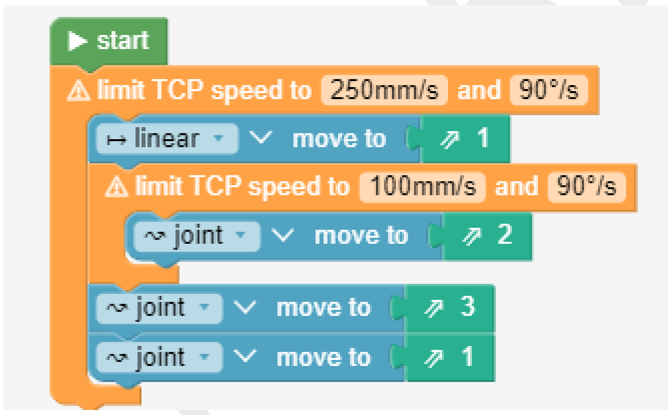
```
▶ start
  ↠ limit speed to 40 % and accel to 40 %
    joint move to 1
    joint move to 2
    joint move to 3
    joint move to 4
```

In some applications it might be necessary to set a maximum speed limit on the TCP. This can be achieved by wrapping a **limit TCP block** around a group of waypoints, which will limit the TCP velocity allowed for the motions contained within the block. This block can be found in the **Motion** toolbox category.



```
▶ start
  ⚠ limit TCP speed to 250mm/s and 90°/s
    linear move to 1
    joint move to 2
    joint move to 3
    joint move to 1
```

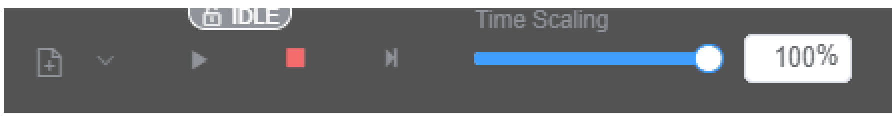
The speed settings blocks above can be nested in a hierarchy, the example below shows a sequence where waypoints 1,3,1 will be executed at a maximum of 250mm/s but the move to waypoint 2 will be executed at a maximum of 100mm/s



```
▶ start
  ⚠ limit TCP speed to 250mm/s and 90°/s
    linear move to 1
    ⚠ limit TCP speed to 100mm/s and 90°/s
      joint move to 2
    joint move to 3
    joint move to 1
```

3.12. Playback of a sequence

To playback a sequence press the Play button on the toolbar at the top of the window. The sequence always begins execution from the 'Start' block, if no start block is defined the sequence will not run anything.

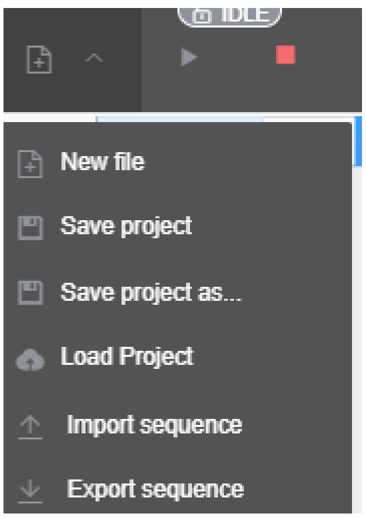


You can **Step** though a sequence one block at a time using the button.

Warning : Your sequence could contain waypoints that cause the robot to collide with obstacles around it. Make sure you check the space around the robot is clear, people nearby are aware the robot will be moving, and an E-Stop or Safe-Stop button in always in easy reach. It is good practice to run the robot with the master speed scaling set low until the sequence has been fully tested.

3.13. Load & Save a sequence

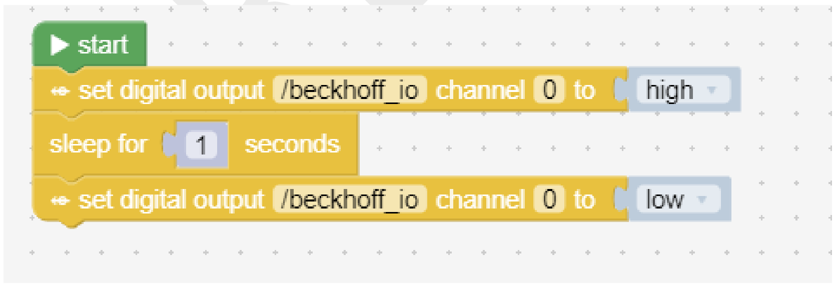
To load and save your sequences use the menu on the top left of the toolbar



You can also export or import the currently loaded sequence as a file to or from your browser which can then be saved to your local file system, network or even sent by email. This file will contain the workspace settings but any 3D meshes that you have loaded are not embedded in this file.

3.14. Setting Digital Outputs

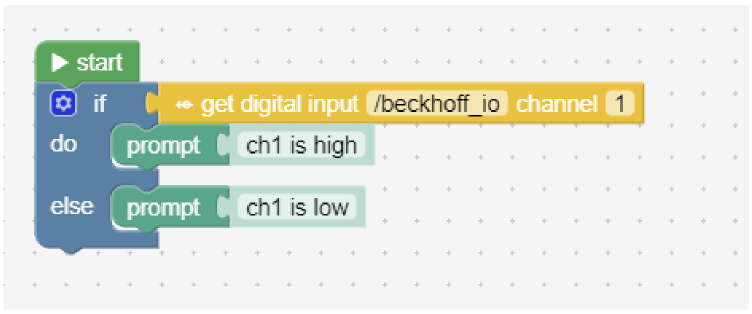
Digital outputs can be set using the 'Set Output' block found in the IO toolbox. To use this block you must specify the required location i.e. the main control box (/beckhoff_io) or Wrist connector (robot/wrist/io), the channel (or pin) number and the state. See the hardware section for pinout and wiring information.



```
start
set digital output /beckhoff_io channel 0 to high
sleep for 1 seconds
set digital output /beckhoff_io channel 0 to low
```

3.15. Reading Digital Inputs

To read the state of a digital input use the 'get digital input' block found in the IO toolbox. This block returns true or false depending on the state of the input channel. It can be assigned to a variable or use as an '*if*' or '*while*' evaluation. Specify the interface, ie. the main control box (/beckhoff_io) or wrist connector (robot/wrist/io), and input channel/pin number. See the hardware section for pinout and wiring information.

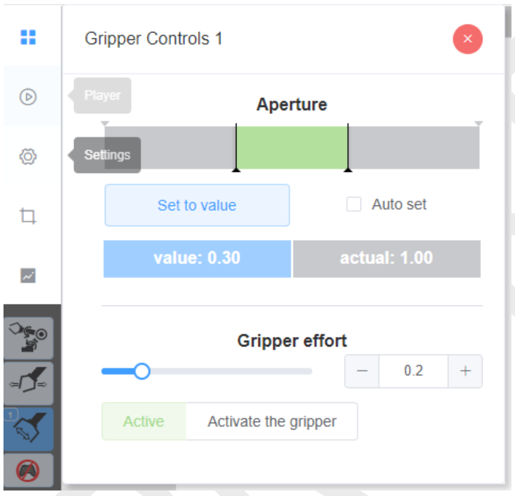


```
start
if  get digital input /beckhoff_io channel 1
do    prompt ch1 is high
else  prompt ch1 is low
```

3.16. Controlling Electric Grippers

A range of third party electric grippers can be connected directly to the wrist interface or robot control box and after the appropriate driver is loaded can then be controlled via the jog panel or in the program sequence using the grip block

To load the driver, see the setting section. Once the driver is leaded a gripper icon will be visible in the left toolbar. Click on this to show the gripper controls panel below. Some gripper such as the RobotiQ parallel finger range require an initialisation stop which will automatically drive the gripper jaws fully open and closed in order to calibrate the tool. To avoid this motion happening at an unexpected time which could cause damager or injury the user must manually active the gripper using the button at the bottom of this panel. Once activated the gripper remains active until the bus power is removed to an E-Stop event.



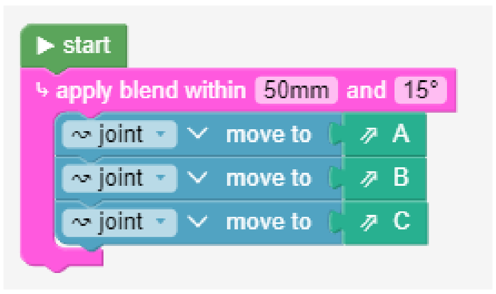
The gripper control panel allows the user to move the gripper jaws in real time. Drag the vertical bars in the aperture control to open or close the jaws. If the ‘Auto Set’ checkbox is ticked this will update the gripper jaws as soon as the control is release (mouse up), if it is not checked the ‘Set to value’ button must be pressed to move the jaws to the aperture set in the control. The effort value set with the slider below is a combined force and speed setting. Low value is slow speed and low force, a high value is a fast speed and high force. The two and not independent in most gripper because of the inertia in gearmotors.

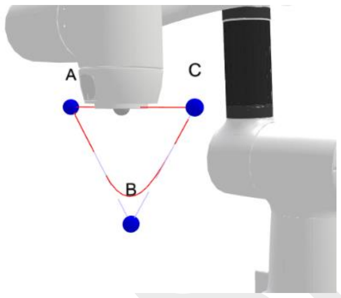
3.17. Motion Blends

Blends can be added to motions that lie between two or more other moves. Specifying a blend allows the robot to pass near a waypoint without traveling directly through it. This allows the robot to maintain TPC velocity as it passes near the waypoint which is not possible without a blend as an instantaneous change in direction could occur when passing through a waypoint with different entry and exit trajectories and would require infinite acceleration.

Blends are useful tools to speed up paths that include intermediate waypoints intended to avoid obstacles in the environment. If for example a path needs to pick an item from directly above it but is approaching from the side prior to that a blend can be used to avoid the need for the arm to stop directly above the object it is packing and instead make a small smooth arc at this point, saving cycle time.

To add a blend to a linear or joint space move block, drag a blend group block from the motion toolbox and add one or more waypoints inside it as show below

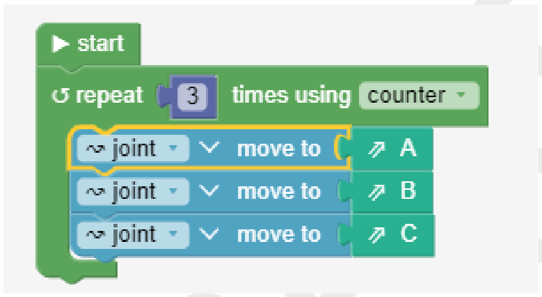




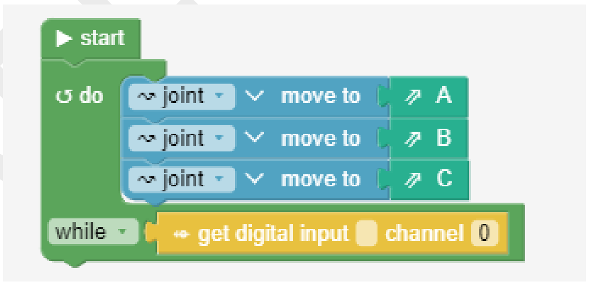
The blend radius specifies the distance from the waypoint at which the path will begin to deviate from a straight line, the angular field specifies the permissible blend in angular space.

3.18. Loops

You can create loops using *repeat* or *do* blocks. To use a **repeat** block, drag the block from the *Basics* toolbox into the sequence and set the number of times to repeat.



or



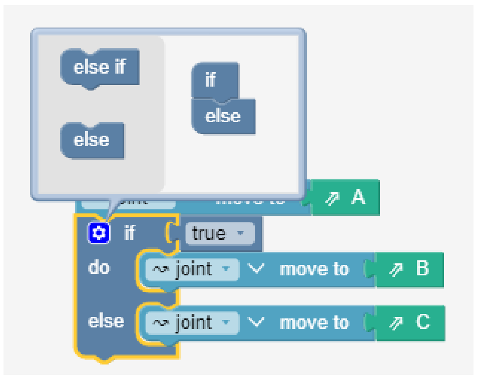
The **do** block enables a loop to be run either *while* or *until* a signal evaluates true. The example above show a *do* block that will continue to loop while the digital input channel 0 is high, if the input is low at the end of the loop cycle the program will finish.

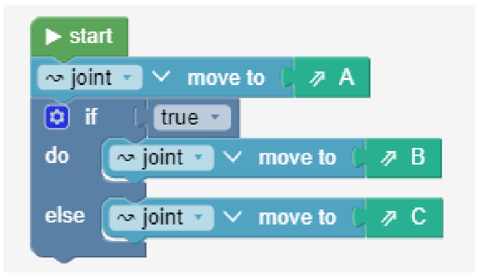
3.19. Logic If/Else

To implement logic in the sequence, use the *if* block found in the Basics toolbox.

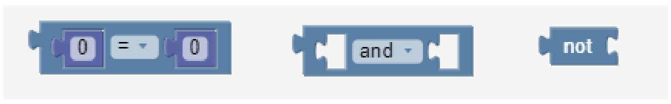
The '*if*' block evaluates an expression and runs the blocks nested inside the '*do'* section if the expression is true, if the expression is false it skips all these blocks and continues immediately after the *if* section ends.

You can add additional 'else if' and 'else' cases to the block clicking on the settings icon ⚙ at the top of the 'if' block. You can add multiple 'else if' sections but only one 'else' at the end.





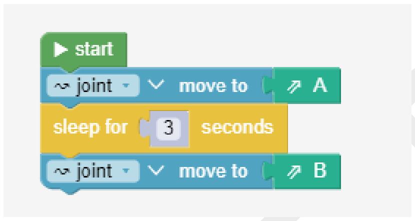
There are a range of useful Boolean evaluations in the Basics toolbox to use with the ‘if’/’else if’ inputs



The comparison block allows two inputs to be compared and return true if they are ‘=’ equal, ‘<’ greater than, ‘>’ less than, ‘≠’ not equal. The *Logical* block allow multiple comparisons to be evaluated in a single input and the *Not* block inverts a true to false and false to true.

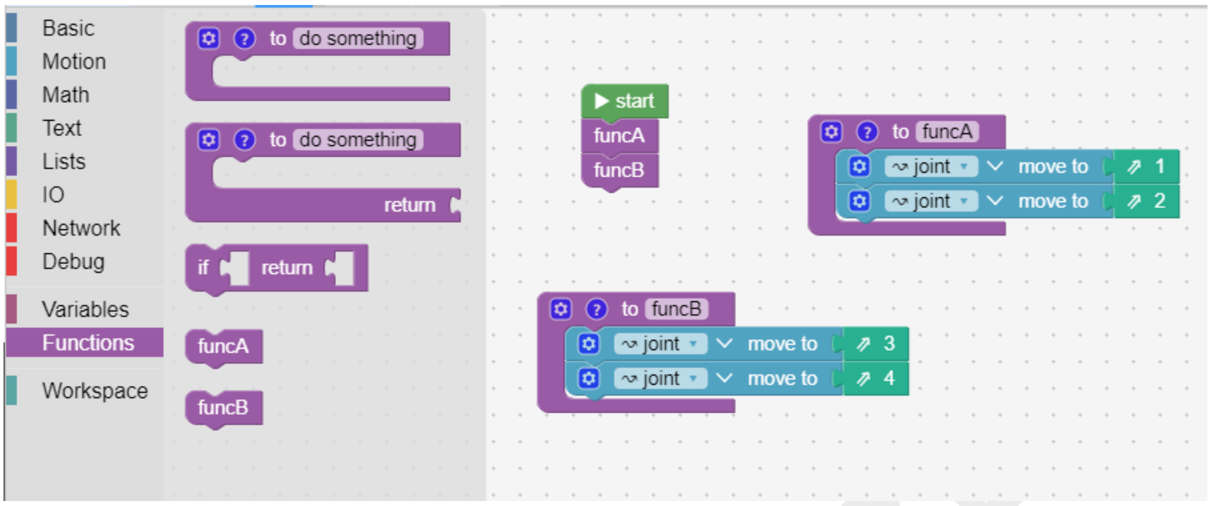
3.20. Delays

You can specify a blocking delay using the sleep block found in the ‘Basic’ toolbox



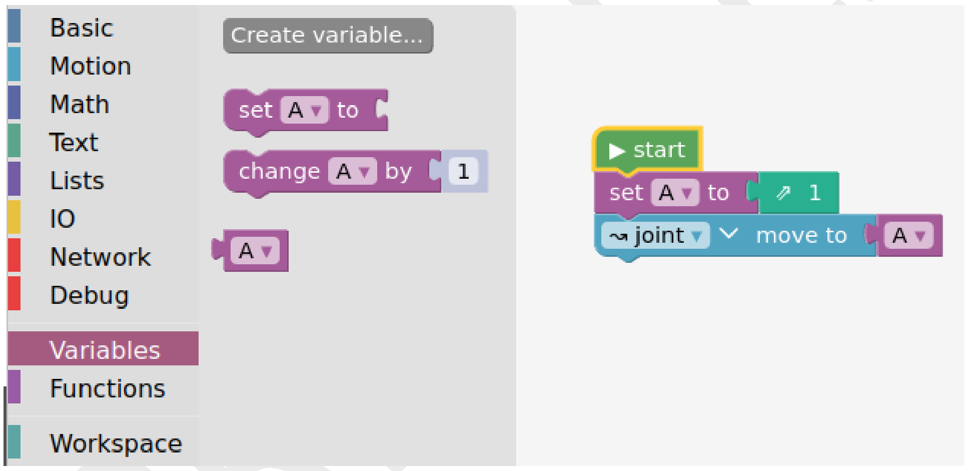
3.21. Functions

You can create custom functions by using a function block, drag an empty function onto the canvas and rename ‘do something’ with your chosen function name (‘myFunc’ in this example). Populate the function with the required blocks and you can then *call* the function from anywhere else in the sequence by dragging the new function call out of the Functions toolbox.



3.22. Variables

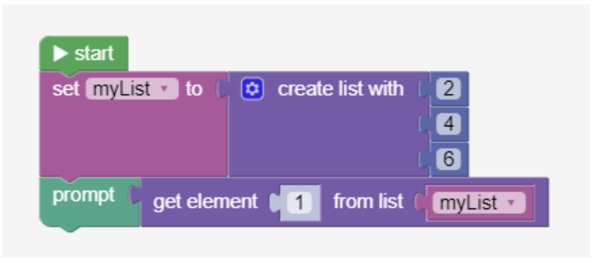
You can define user variables in the variable toolbox category, click on 'Create Variable' and give the variable a name. A new instance of the variable will then be added to the variables toolbox category which allows you to drag the variable to the sequence canvas for use as a field input or output.

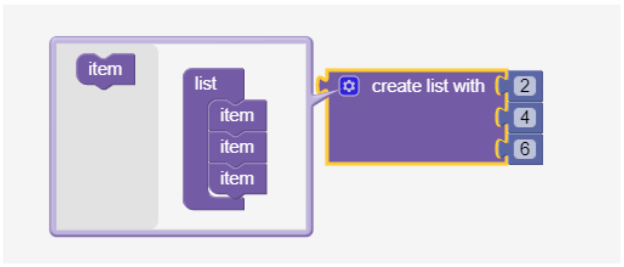


3.23. Lists

Variable can be assigned arrays or lists which can then be iterated through in loops or used in lookup functions. The 'List' toolbox contains blocks to define, manipulate and extract data in lists and arrays.

To manually create a list assigned the 'Create List with' block to a variable, adding the data you want to the list in order. You can then use 'get element [] from' to extract a specific value from the list at a given index.

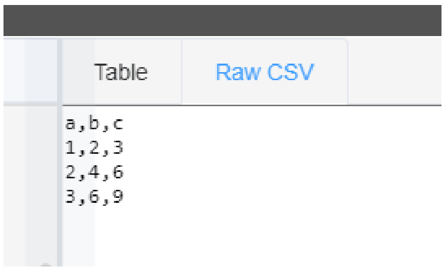

start
set myList to create list with 2 4 6
prompt get element 1 from list myList



If you need to add more elements in the list click on the gear icon in the top left and drag items to or from the list as required.

3.24. Dictionaries

Dictionaries are key-value pairs that can be stored in variables and even lists. The from CSV block allows the user to paste a table of CSV data into the form and return a list of row, with each row containing the key value pairs for each column. In this example a simple CSV file is created with the data below:



```
a,b,c
1,2,3
2,4,6
3,6,9
```

When switched into the table view in the same form you can view the data in rows and columns:

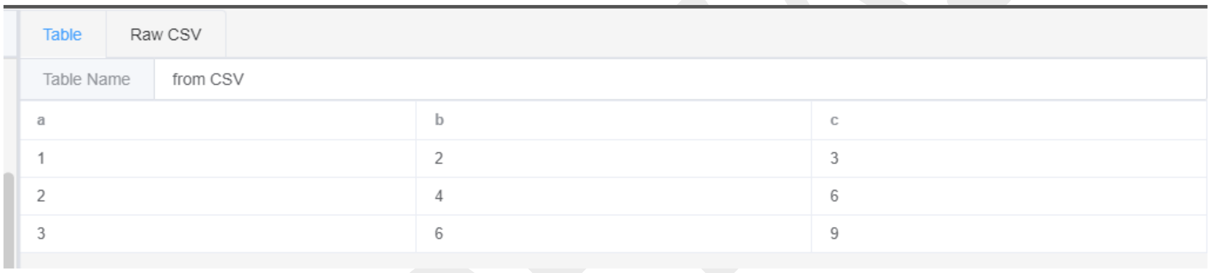
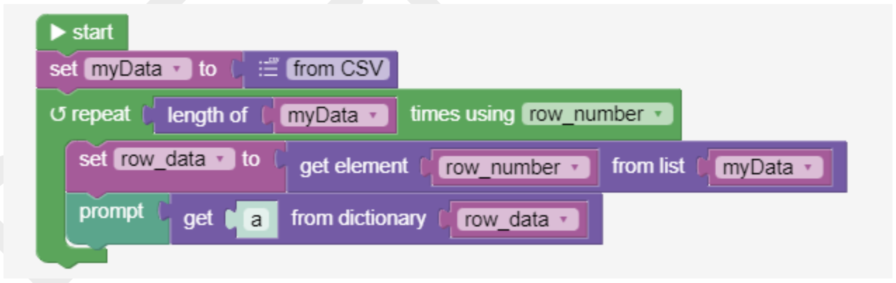


Table Name	from CSV	
a	b	c
1	2	3
2	4	6
3	6	9

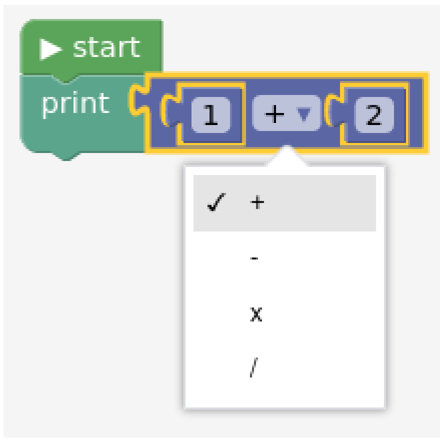
The sequence below then shows how the data can be assigned to variable called myData and iterated though using a loop, the 'get element [] from' block to extract the row and the get [] from dictionary to extract the specific value. In this case prompting the value of the 'a' column for each row.



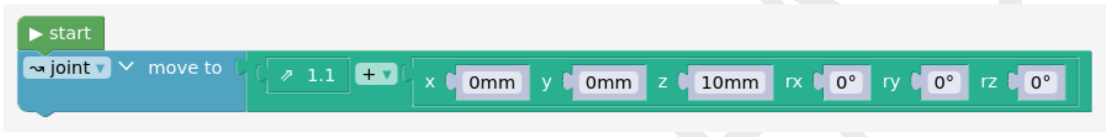
```
start
set myData to from CSV
repeat length of myData times using row_number
    set row_data to get element row_number from list myData
    prompt get a from dictionary row_data
```

3.25. Maths

Maths blocks can be found in the Math toolbox category. One main block is responsible for performing basic scalar maths operations such add, subtract, multiply and divide.

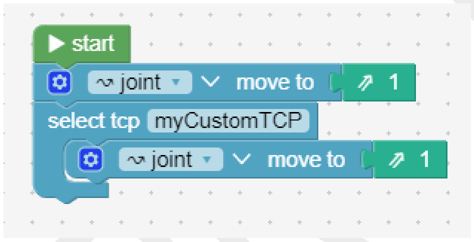


Waypoints can also be added together using the waypoint math block. This block can be found in the motion toolbox category.



3.26. TCPs

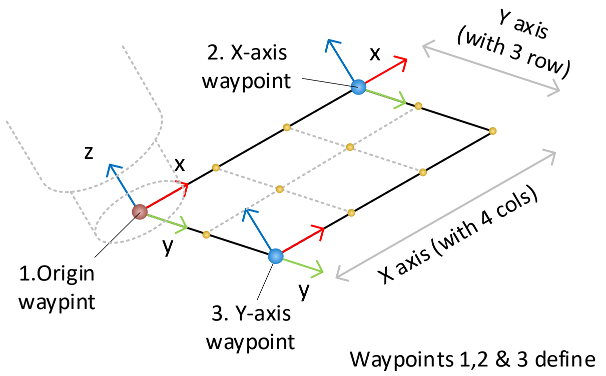
You can switch between user defined TCPs within a blockely sequence using the change tcp offset block



3.27. Grids/Palatizing

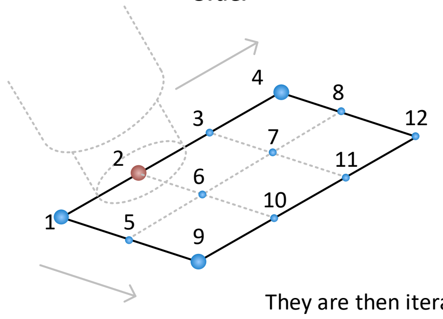
Palatizing functions can be implemented using the grid block. This block calculates on offset for each point in a grid of specified row and column count defined by three real user specified waypoints. These define the origin and the two corners which specify end of the first row and the first point on the last row.

Grid Setup Example



Waypoints 1,2 & 3 define the grid boundaries

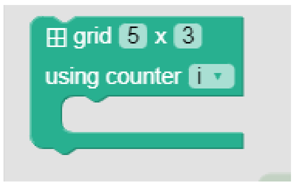
Order

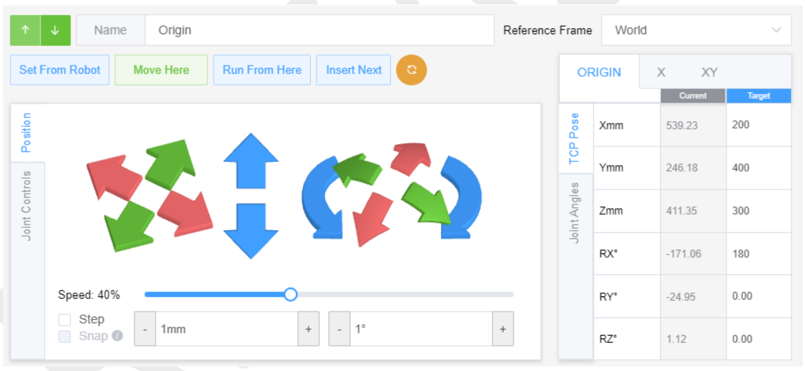


They are then iterated along the X axis first then the Y axis

The grid evaluation block does not move the robot, it simply offsets the waypoints placed within it by the amount required for each point on the grid in sequence.

To use the palatizing function drag a grid block out of the motion toolbox and place it in your sequence:

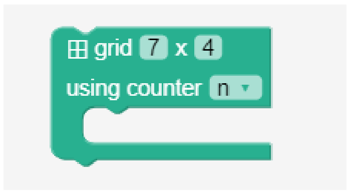


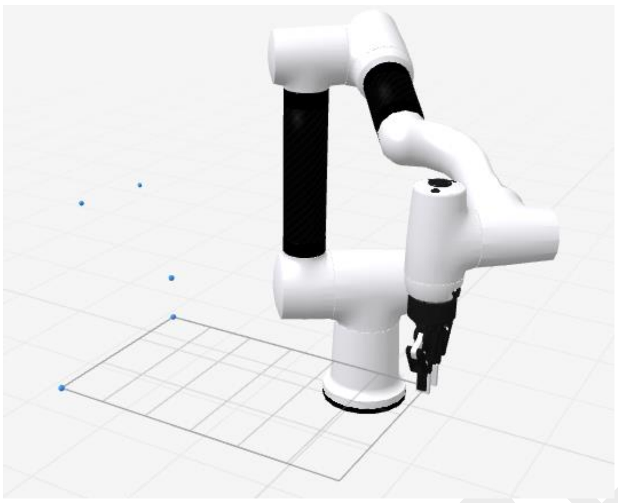


	Current	Target
Xmm	539.23	200
Ymm	246.18	400
Zmm	411.35	300
RX°	-171.06	180
RY°	-24.95	0.00
RZ°	1.12	0.00

You will then need to define the three waypoints to capture the three corners of the grid. There are called 'Origin', 'X' and 'XY'. Do this in the same way you would define a regular waypoint, by moving the arm to the position you want the jog controls, free mode or using the pendant joystick. Start with the origin by selecting the 'origin' tab on the top right and pressing 'Set From Robot' when it is in position. This will be the first point on the grid. Next set the 'X' point by selecting the tab labelled 'X' and pressing set from robot, this defines the end of the first row. Now set the 'XY' point by selecting the 'XY' tab and pressing 'Set from robot' again, this defines the end of the first column. Note the 'Insert Next' button allows you to cycle through these tabs setting the next tab to the current robot position and provided a convenient workflow.

Once you have defined the corner points you will need to set the number of rows and columns, this is done in the block itself, in this example 7 columns and 4 rows.

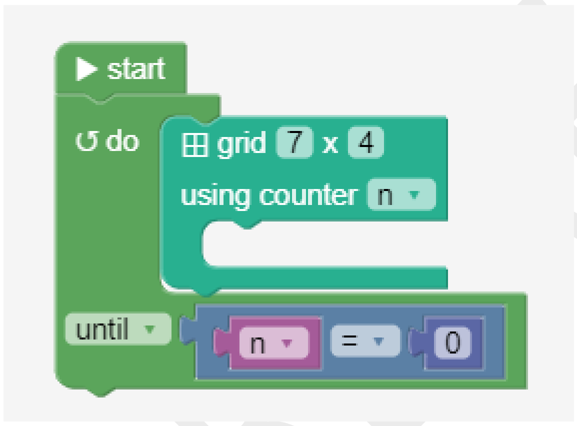




The grid block itself is passive and you will need to put waypoints within it and cycle through these with an encapsulating loop.

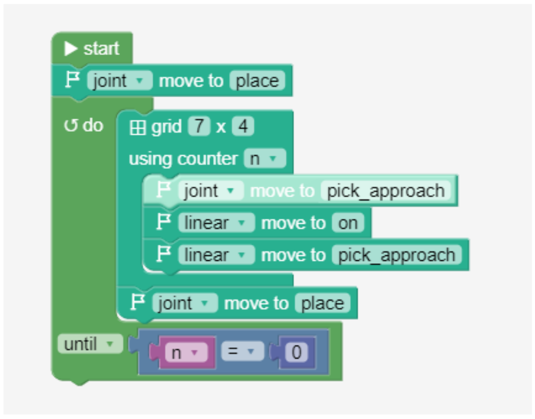
The grid block uses a variable to store its index and increments this after each time it is called.

The example below shows how a do-until loop can be used to cycle through each point in the grid once, n will be 0 on first pass but will be 1 by the time the until clause is evaluated so it will run the loop 28 times before completing.

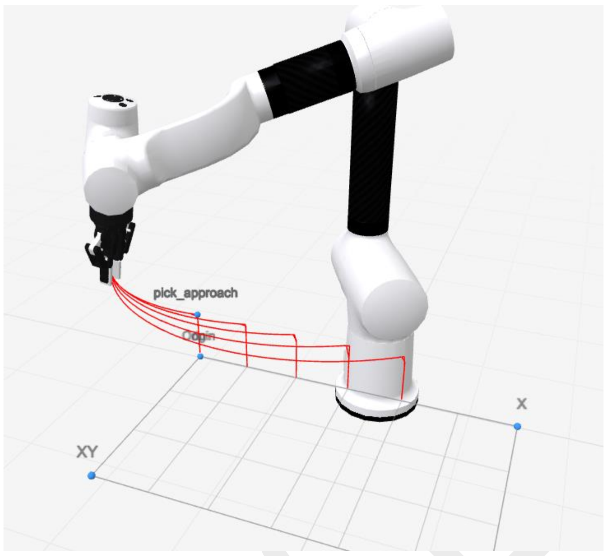


Now you have defined the grid and looping behaviour you can define the steps you want to occur each cycle.

You can put any blocks inside the 'grid' group including moves and gripper controls. Any waypoint move will be offset relative to the origin you have defined for the given index. This allows you to program the behaviour you want to occur at the origin point and it will translate this for each subsequent point on the grid. The example below shows the path with the breadcrumbs view turned on:

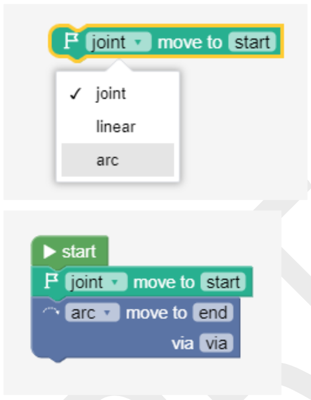


```
start
joint move to place
do  grid 7 x 4
    using counter n
        joint move to pick_approach
        linear move to on
        linear move to pick_approach
    joint move to place
until  n = 0
```



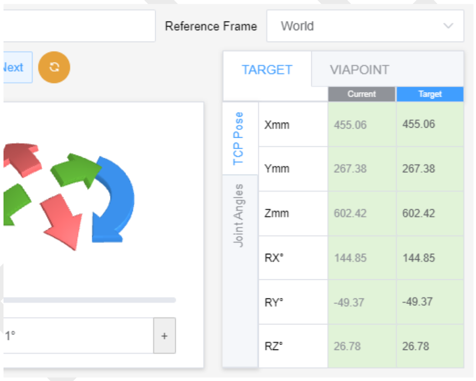
3.28. Arc Paths

The arc move path allows you to define paths with a curved trajectory. To use arc moves you will need to use a move block found in the motion toolbox, drag a move block to the workspace and then select the move type ‘arc’ on the blocks drop down:

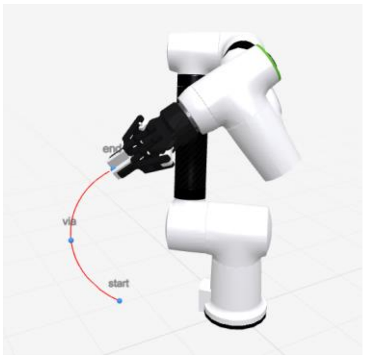


```
joint move to start
✓ joint
  linear
  arc

start
joint move to start
arc move to end
            via via
```



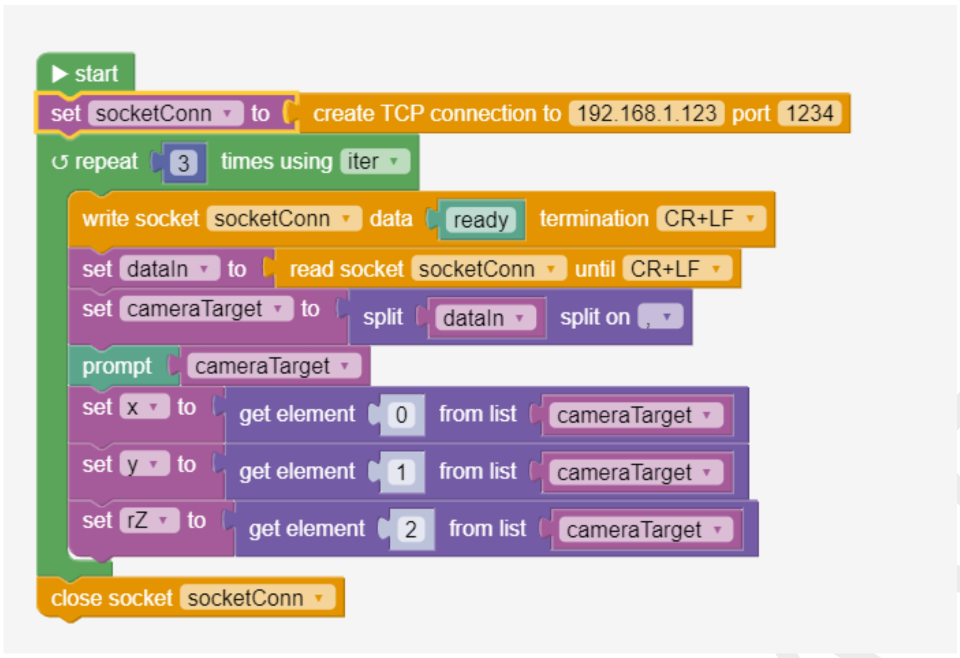
	Current	Target
Xmm	455.06	455.06
Ymm	267.38	267.38
Zmm	602.42	602.42
RX°	144.85	144.85
RY°	-49.37	-49.37
RZ°	26.78	26.78



Once the move type is set to arc the block will change its colour to mauve and show two labels. Click on the block to open the form on the left and select the ‘TARGET’ tab on the top right. Set the end position for the arc by moving the arm to the required pose and pressing ‘Set from Robot’. Next define the midpoint by selecting the ‘VIAPOINT’ tab on the top right, moving the robot to the required position and pressing ‘Set From Robot’

3.29. TCP Communications Sockets

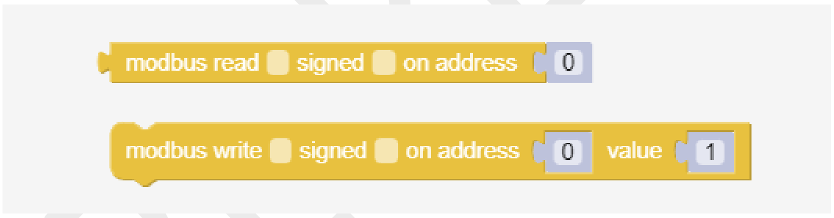
To communicate with PCs on the same network you can use a TCP Communications Socket. First create a variable for the connection, ie ‘socketConn’, then use the ‘create TCP connection’ block set this and open a TCP socket connection to a server on your local network, ie you PC, specifying the IP and port. Once the socket is open you can use the ‘read socket’ and ‘write socket’ blocks to transfer data to and from the server as a text string. When the session if finished use the ‘close socket’ block to close the connection. You can use the text ‘Split’ block to extract comma separated values from strings read over the socket connection as shown below or even the ParseJSON block for more robust data structures:


```
start
set socketConn to create TCP connection to 192.168.1.123 port 1234
repeat 3 times using iter
    write socket socketConn data ready termination CR+LF
    set dataIn to read socket socketConn until CR+LF
    set cameraTarget to split dataIn split on ,
    prompt cameraTarget
    set x to get element 0 from list cameraTarget
    set y to get element 1 from list cameraTarget
    set rZ to get element 2 from list cameraTarget
close socket socketConn
```

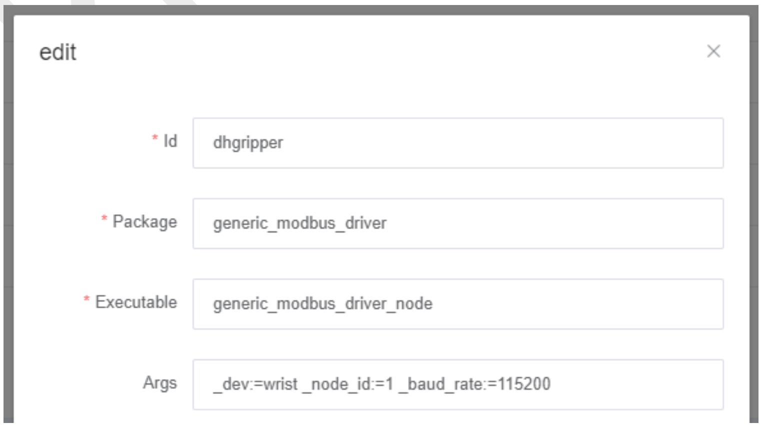
3.30. Modbus Slave Read Write

You can read and write registers in Modbus slave devices using the 'Generic Modbus driver'. This driver needs to be loaded in the same way as other third-party drivers. Please see "URD-0007 driver installation guide" for more details. Once loaded you can use the Modbus 'read' and 'write' blocks to communicate with the device which could be connected to the wrist or control box depending on the settings used in the driver installation.

These Modbus Read / Write blocks are found in the IO > Modbus toolbox.


```
modbus read signed on address 0
modbus write signed on address 0 value 1
```

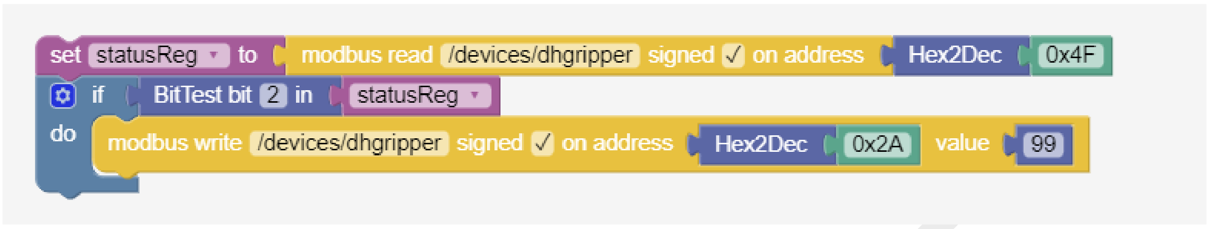
You must set the device field to match the one used in your driver settings prefixed with '/devices/', in this example "/devices/dhgripper", when the ID can be found in the Settings > driver Settings form.


edit

* Id	dhgripper
* Package	generic_modbus_driver
* Executable	generic_modbus_driver_node
Args	_dev:=wrist _node_id:=1 _baud_rate:=115200

These blocks read or write a 16-bit integer value, note the 'signed' checkbox which changes the integer range from 0 - 65,535 to -32,767 - 32,768 when checked.

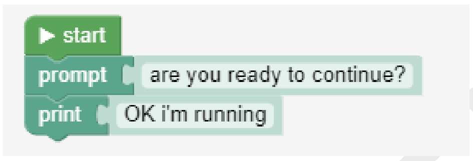
In the example below these blocks are used to read a register at address 0x4F and check if bit 2 is set, if it is set then the sequence will write back the value 99 to register 0x2A.


```
set statusReg to modbus read /devices/dhgripper signed ✓ on address Hex2Dec 0x4F
if BitTest bit 2 in statusReg
do  modbus write /devices/dhgripper signed ✓ on address Hex2Dec 0x2A value 99
```

You can have multiple Modbus devices connected to the system with multiple drivers load and reference them with different names.

3.31. User Prompt and Print

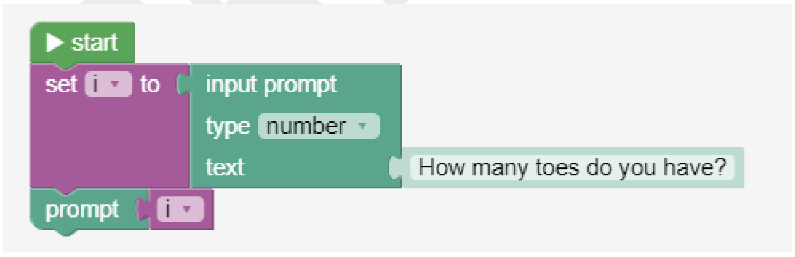
To prompt or notify the operator during the execution of a sequence you can use the prompt or print blocks


```
▶ start
prompt are you ready to continue?
print OK i'm running
```

Print send a text message to the user interface and continues with the sequence, prompt blocks the execution of the sequence until the

3.32. User Input prompt

To prompt the use to enter a number or text use the input block found in the debug toolbox, this block prompts the user, both on the pendant and engineering UI with the message set in the 'text' parameter. The value returned once the user presses enter can be assigned to a variable or evaluated in a comparison block. Set the type to number or string depending on the type of input required. If this is set to none then the block behaves in the same way as the prompt block.

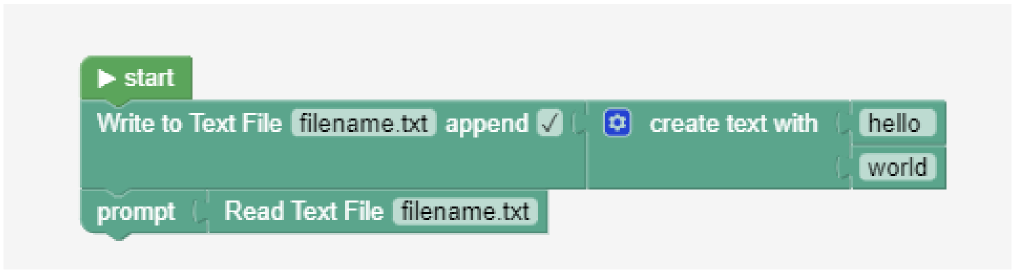

```
▶ start
set i to input prompt
         type number
         text How many toes do you have?
prompt i
```

If the user closes the prompt box or enters no text and presses submit when the type is set to 'number' then the block will throw an error. You can use the try-catch block to handle this.

3.33. Read /Write text files

You can both read and write user files in the sequence. This can provide a useful way to log information in a running sequence, to save settings or progress or for loading data from a file to be used in lists and iterators.

The Read / Write blocks are found in the 'Text' toolbox. User files are stored in the robot controller file system and can be accessed by going to the main menu > 'user files' where you can view, download or upload files.


▶ start
Write to Text File filename.txt append ✓
create text with
hello
world
prompt
Read Text File filename.txt

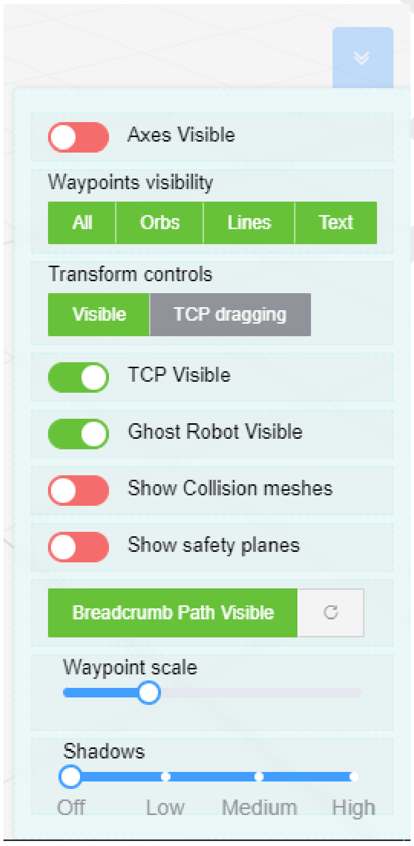
The Write File block accepts string, you can concatenate strings with the create text with block as show above. If the append checkbox is ticked the end of an existing file will be appended with the next text. If it is not checked the file will be overwritten. If no file exists a new one will be created. A Line-Feed character is added to the end of each line written with this block.

The read text block reads the entire file and assigned it to a variable. You can then use parsing tools like the 'parse from CSV' 'parse from JSON' or 'split on' blocks to convert the text data into variables or arrays or key value pairs.

3.34. Visualization

A 3D visualization of the robot arm is shown on the right pane, this can be expanded to full screen with the full button, Hidden or Shown. Left click and drag to rotate the camera view. Right click to move the camera, Roll the mouse wheel to zoom in and out.

In the bottom right click on the up arrow to show the view options pane:

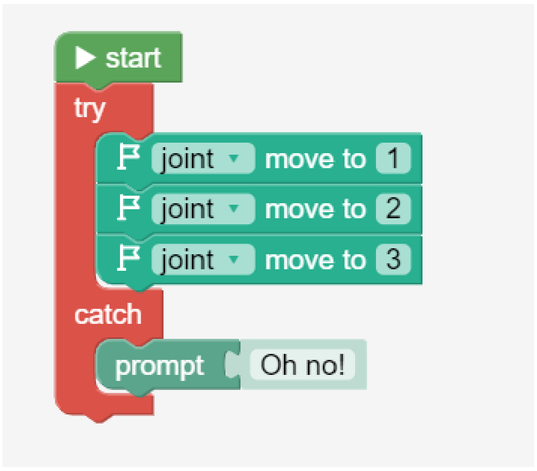


- The following options can be enabled or disabled:
- Axes visible – Show/Hide the X,Y,Z axis for each waypoint
- Waypoint visibility – Show/Hide the markers, path line and text labels for waypoints
- Transform Controls – This shows the transform arrows that allow object to the moved and rotated in the 3D view.
- TCP dragging allows you to drag the robots pose in the 3D view.
- TCP Visible – shows or hides a marker to indicate the current selected TCP position in the 3D view.
- Ghost Robot Visible – Shows / hides a preview of the robot pose when a waypoint is selected.
- Show Collision Meshes - Shows / hides the collision meshes around the robot arm.
- Show Safety Planes - Shows / hides the safety planes that have been configured in the workspace.
- Breadcrumb – This shows the path the TCP has recently travelled in the 3D graphics. This can be reset using the button to the right.
- Waypoint scale – this changes the size of the waypoint markers in the 3D view.
- Shadows – this control adjusts the visualisations of shadows in the 3D visualisation. Turn of if you are running the interface on a

lower power PC that does not have a powerful graphics card.

3.35. Try Catch

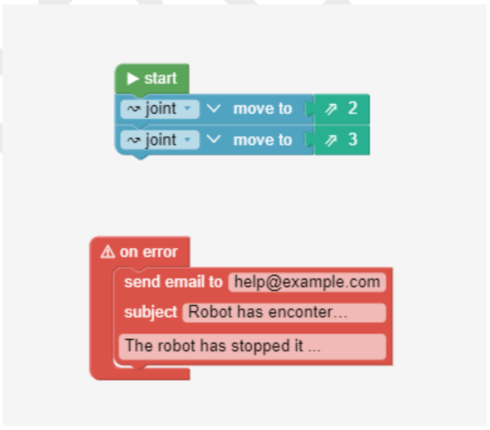
To catch errors and handle them within the sequence you can use the try catch block, this is found in the Debug toolbox.



The blocks within the try area are run as normal but if an error occurs the program will immediately jump to the catch area, in the example above running the prompt block.

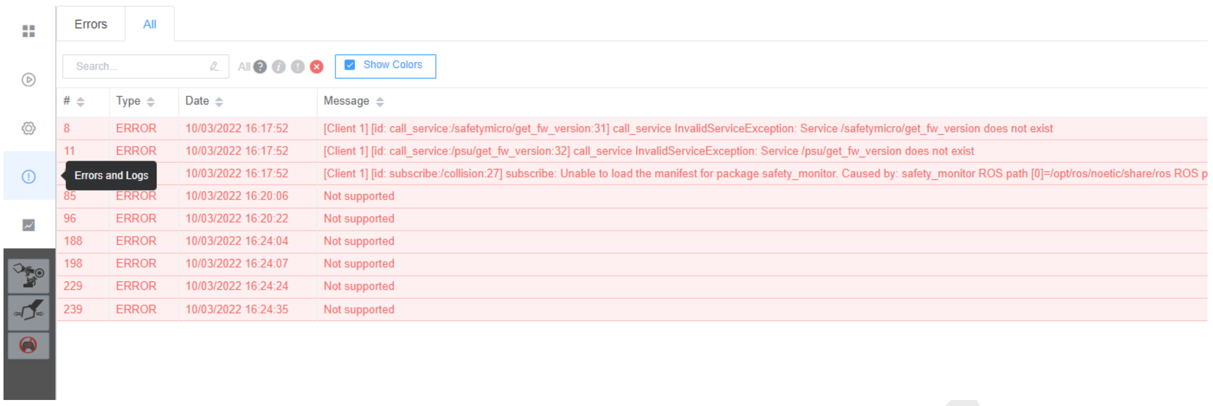
3.36. On Errors Event Handler

The robot can encounter errors for a range of different reasons. Some examples include trying to execute a motion to a waypoint that is out of reach or not possible within the kinematic constraints, a collisions between the arm and a physical object, trying to communication with a device that is not present, ie a gripper that has been removed, or using a variable that has not yet been set. You can use the 'on error' block to respond to an error. This block allows you to display and error message, send an email or SMS to notify someone that the robot has stopped.


start
joint move to 2
joint move to 3
on error
send email to help@example.com
subject Robot has enconter...
The robot has stopped it ...

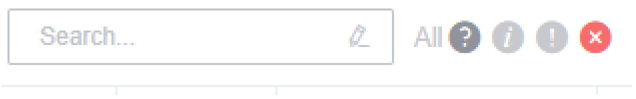
3.37. Error Log

A list of recent error can be viewed by clicking on the Errors and Log view in the left toolbar:



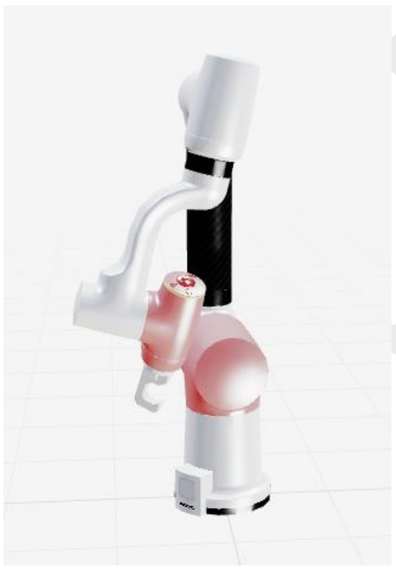
#	Type	Date	Message
8	ERROR	10/03/2022 16:17:52	[Client 1] [id: call_service:/safetymicro/get_fw_version:31] call_service InvalidServiceException: Service /safetymicro/get_fw_version does not exist
11	ERROR	10/03/2022 16:17:52	[Client 1] [id: call_service:/psu/get_fw_version:32] call_service InvalidServiceException: Service /psu/get_fw_version does not exist
		10/03/2022 16:17:52	[Client 1] [id: subscribe:/collision:27] subscribe: Unable to load the manifest for package safety_monitor. Caused by: safety_monitor ROS path [0]=/opt/ros/noetic/share/ros ROS p
85	ERROR	10/03/2022 16:20:06	Not supported
96	ERROR	10/03/2022 16:20:22	Not supported
188	ERROR	10/03/2022 16:24:04	Not supported
198	ERROR	10/03/2022 16:24:07	Not supported
229	ERROR	10/03/2022 16:24:24	Not supported
239	ERROR	10/03/2022 16:24:35	Not supported

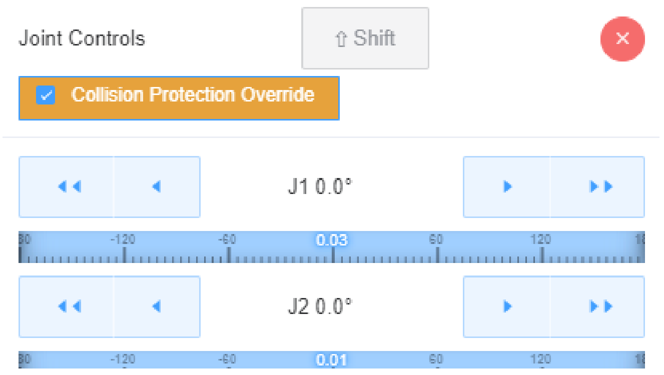
This show errors that have occurred since the client session has started. You can filter and search by using the controls at the top:



3.38. Collision protection

If the robot comes to close to another part of its arm or a safety plane the robot will stop and the relevant joint will be highlighted red in the 3D visualisation as shown below. You will not be able to move the robot when it is in this state unless you selected the ‘collision protection override’ check box in the job panel. This will then allow you to drive the arm out of the collision area using the joint jog controls.





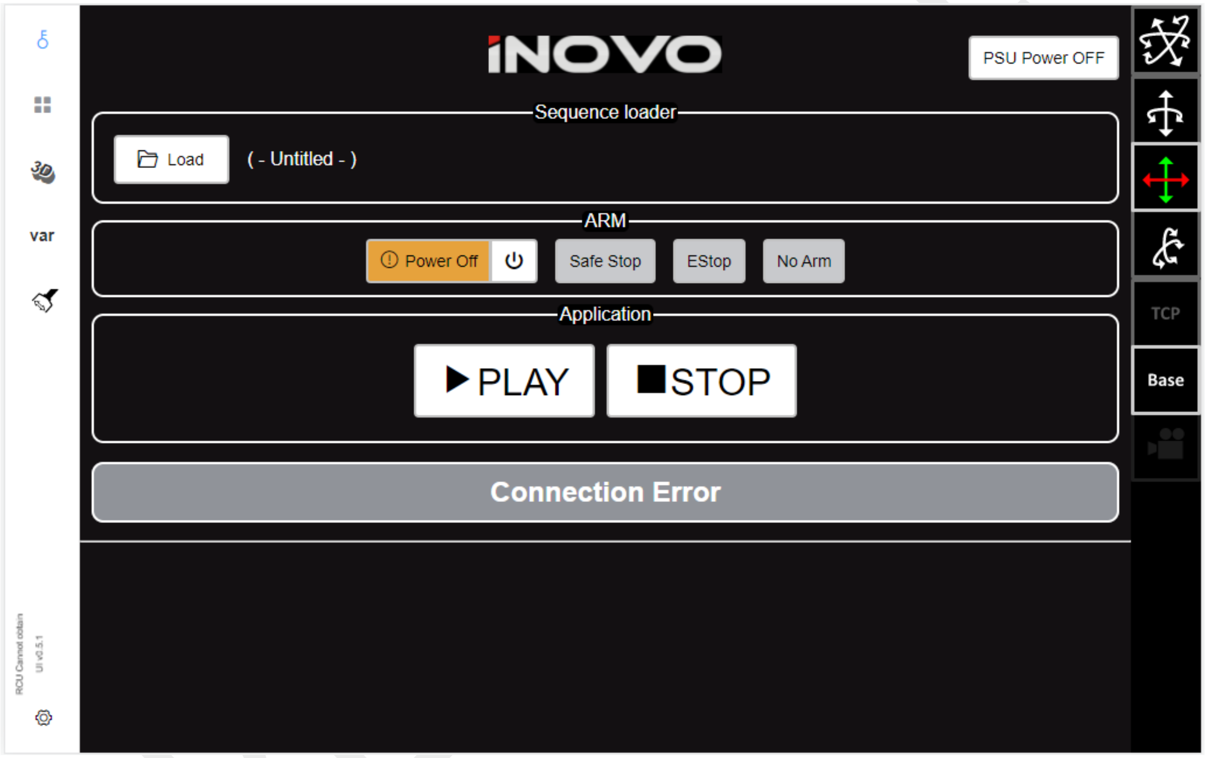
4. THE PENDANT

4.1. The Pendant purpose

The pendant is intended to be permanently connected to the robot and provide a simple control interface for daily use. It allows the robot to be powered on, enabled and a pre-existing program to be loaded and run. It does not provide an interface for programming the robot which must be done using a separate PC or Laptop. The pendant has a touch screen with several tabs that allow different screens to be selected. It also allows has an integrated joystick that allows the robot to be controller in real time. This enables the operator to move the robot to any required position to recover it from a pose that is not safe to start from, to put it in a position that provides access to itself or other areas in the workspace or to put it in a position that is convenient to reconfigure or disable the arm before powering it down.

4.2. The Home screen

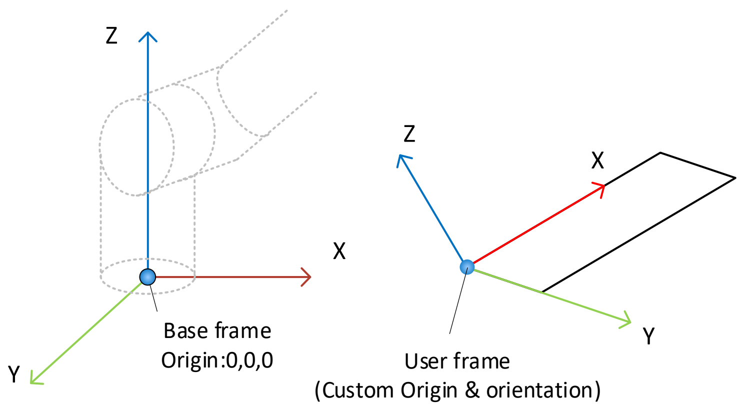
The home screen has buttons to turn the power to the arm on or off. When the RCU starts up the arm power will not be on. This must be turn on by pressing the green ‘Bus ON’ button. When the arm starts the wrist indicator will blink orange for a few seconds and then go solid orange. This indicates power is present but the arm is disabled. Press the ‘Arm Enable’ button to enable the arm. The indicator will now go green and the joint brakes are released. The arm is now able to move. If the Safe Stop is activated the indicator will turn red.



5. SETTINGS

5.1. Defining User Frames

A user frame allows you to define a custom coordinate system in the robot workspace. This is useful when you need to work with an external coordinates system from a smart camera for example.

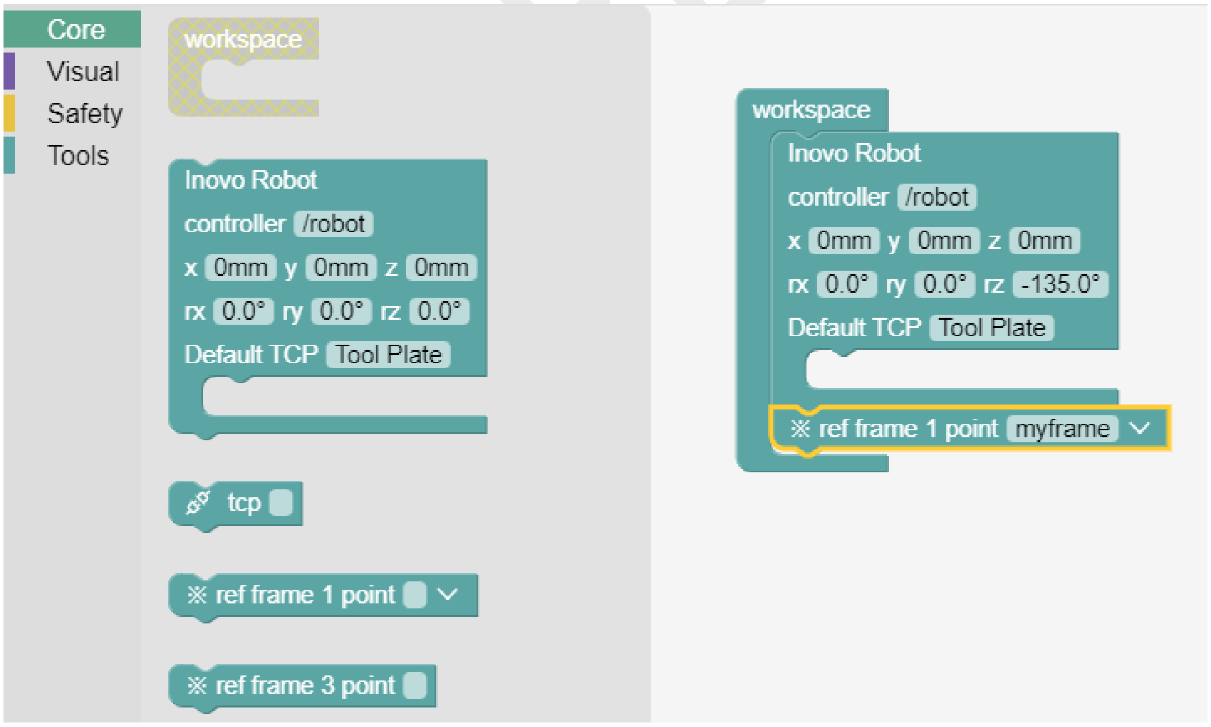


You can define multiple user frames and label them with a reference name of your choice. Once you create a user frame you can specify a waypoint using this coordinate system and move the robot with the jog control in the user frames space.

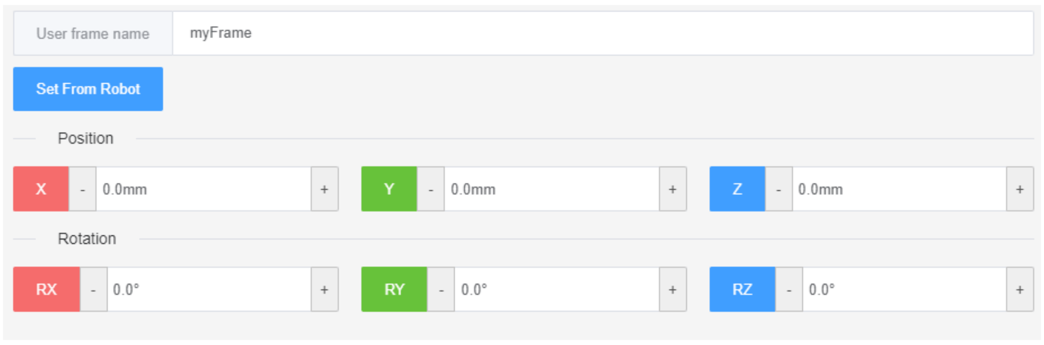
There are three ways to define a user frame, from a tool pose or single point, from a line formed of two points or from a plane defined from three points.

5.1.1. Define a user frame from a TCP pose (Single point)

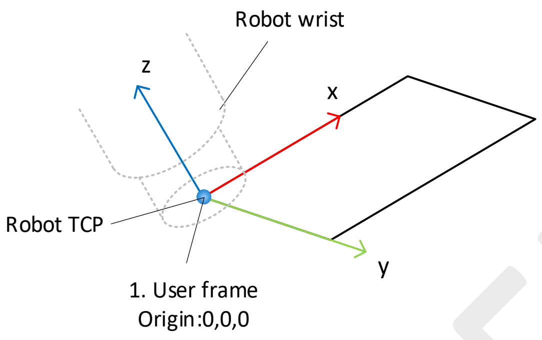
To define a new user frame drag the reference frame block out of the workspace toolbox and add it to the workspace group as shown below



Select the reference frame block so the frame form will show above the 3D visualisation:

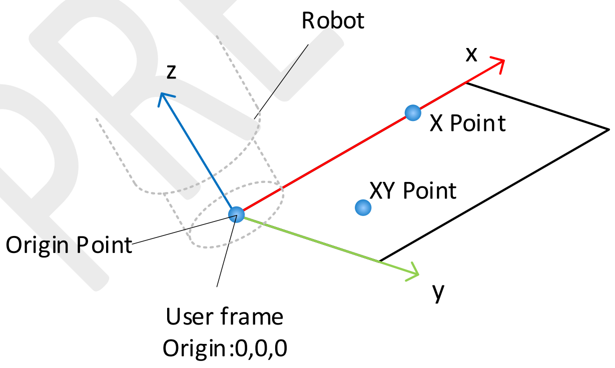


Position the robots wrist so the TCP is at the origin of the new frame you want to define and press the blue 'Set from Robot' button. This will populate the X, Y, Z and rX, rY, rZ fields with the current TCP to be used as the frame point. The frame orientation will be set to match the current TCP position and orientation at this point

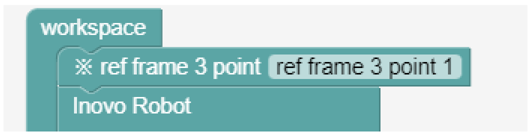


5.1.2. Define a user frame from three points

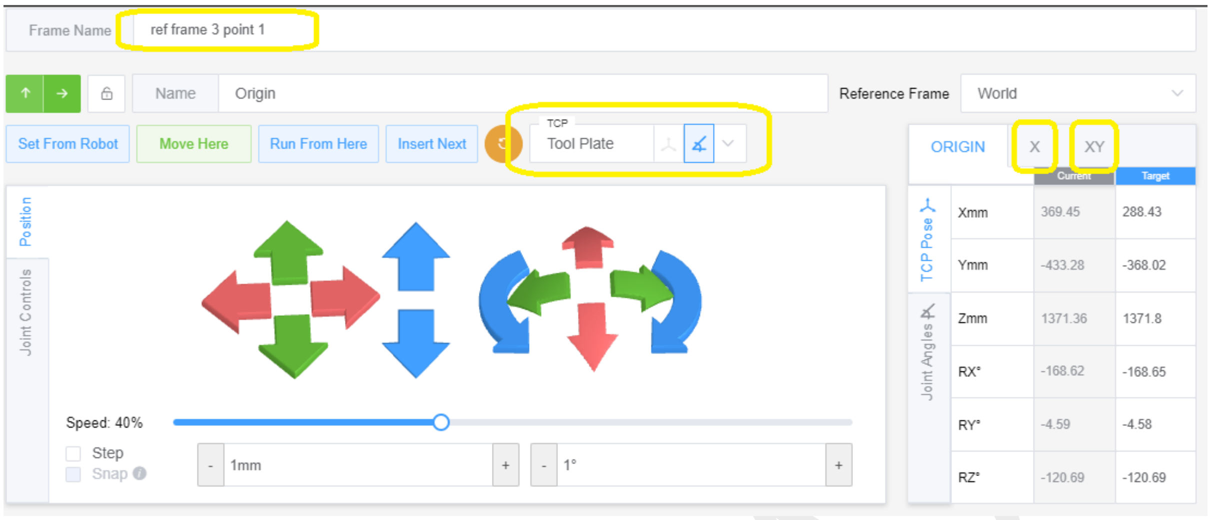
You can define a user frame by capturing three real world points using the robots TCP, one point as an origin, a second point along the X axis and a third point anywhere on the XY plane.



In your project workspace tab drag a 'ref frame 3 point' block from the 'core' toolbox into the workspace:



Click on the ‘ref frame’ block to open the form to open the form and enter a Frame name for your new frame at the top:



Frame Name: ref frame 3 point 1

Name: Origin

Reference Frame: World

TCP: Tool Plate

Speed: 40%

Step: 1mm, 1°

Snap

	Current	Target
Xmm	369.45	288.43
Ymm	-433.28	-368.02
Zmm	1371.36	1371.8
RX°	-168.62	-168.65
RY°	-4.59	-4.58
RZ°	-120.69	-120.69

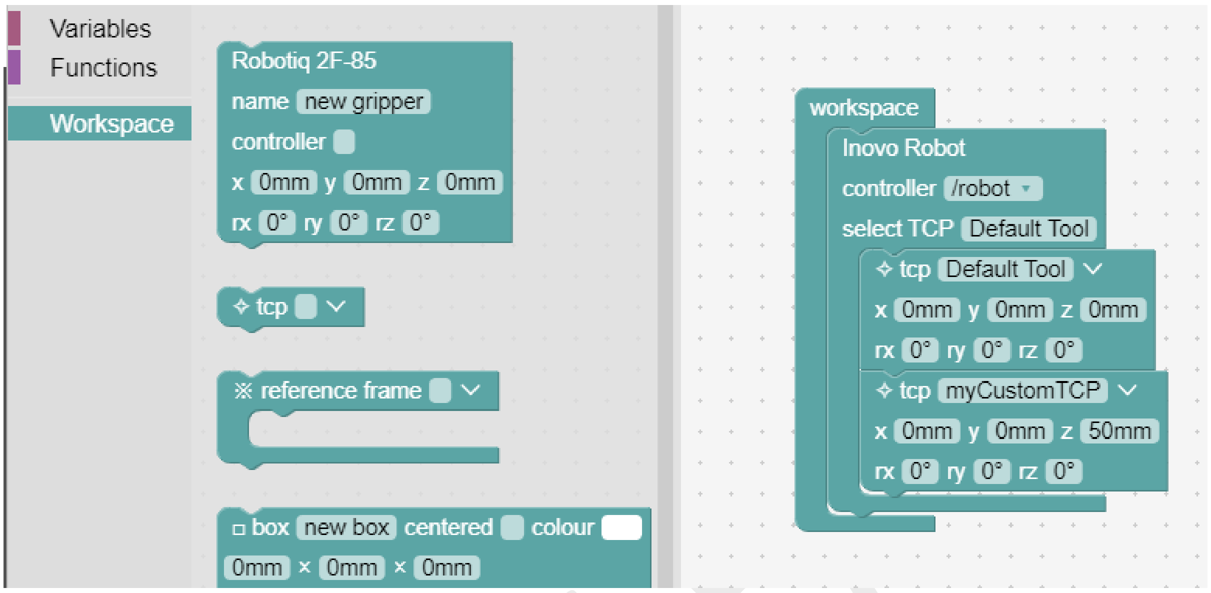
Select the TCP you want to use from the drop-down menu and move the robot so its TCP is at the position you want to be the origin for your new frame. Press ‘Set From Robot’ when in position. Note the orientation of the points does not matter in 3-point frames, only the X,Y,Z position in the selected TCP.

Next select the ‘X’ tab above the current position displayed on the right of the form and move the robot so its TCP is at a point along the X axis of your new frame. Again press ‘Set From Robot’ when in position.

Finally select the ‘XY’ tab and move the robot so its TCP is on the XY plane and press ‘Set From Robot’ when ready. Note: It is recommended to make the X and XY points at least 30cm from the origin to provide the best precision, the further these distances from the origin the better if the plane is intended for use over a large working area.

5.2. Defining User Tool Centre Points (TCPs)

You can define user TCPs in the workspace group, go to the 'Workspace' toolbox on the left and drag a tcp block into the Inovo Robot group in the 'workspace' block or edit an existing one. Enter a name for your custom TCP and set the offset values X,Y,Z and rx,ry,rz.



Robotiq 2F-85
name new gripper
controller
x 0mm y 0mm z 0mm
rx 0° ry 0° rz 0°

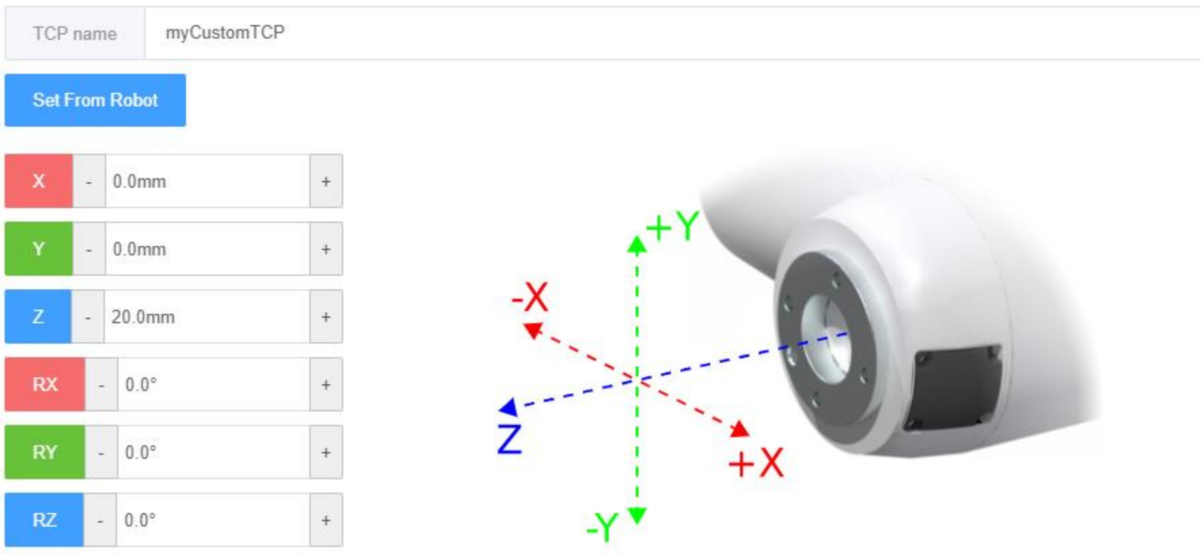
tcp

reference frame

box new box centered colour
0mm × 0mm × 0mm

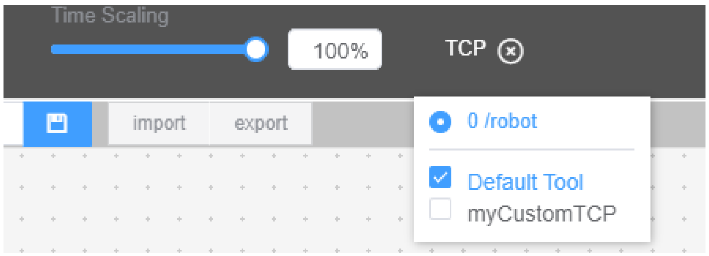
workspace
Inovo Robot
controller /robot
select TCP Default Tool
tcp Default Tool
x 0mm y 0mm z 0mm
rx 0° ry 0° rz 0°
tcp myCustomTCP
x 0mm y 0mm z 50mm
rx 0° ry 0° rz 0°

The form to edit the TCP will appear on the right when you select the required block.



TCP name	myCustomTCP
X	0.0mm
Y	0.0mm
Z	20.0mm
RX	0.0°
RY	0.0°
RZ	0.0°

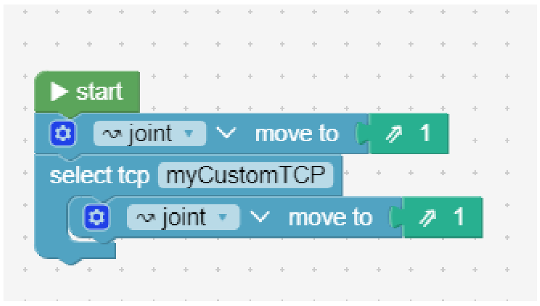
You can now select any of the TCPs from a dropdown on the top toolbar



This will affect the linear jog motion when the ‘Tool’ Frame is selected and any linear moves used in a blockly sequence.

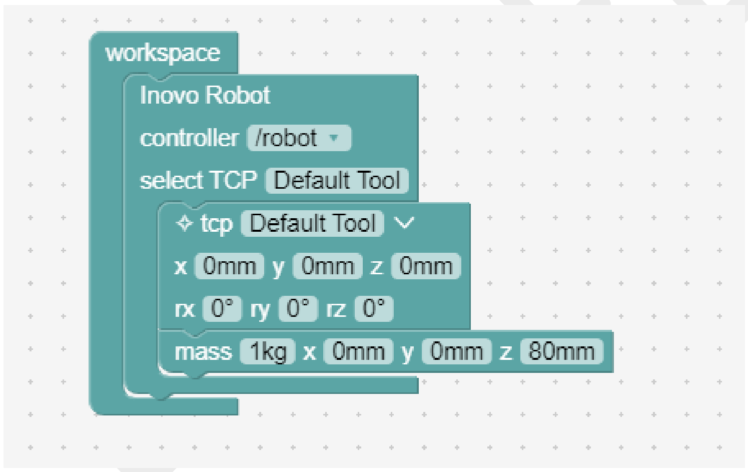
You can also switch between user defined TCPs within a blockely sequence using the change tcp offset block

In the example below the arm moves to the first waypoint using the default TCP, it then selects the custom TCP and moves the arm to the same waypoint in the context of the new TCP, this caused the arm to reposition itself to the new TCP is as this waypoint.



5.3. Defining Payload Mass and Centre of Mass

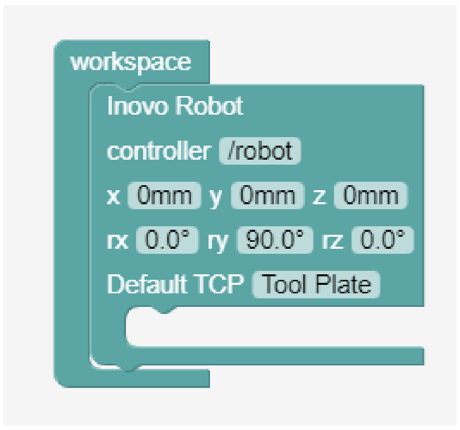
You can define a custom payload and centre or mass using the ‘mass’ block found in the workspace toolbox. Drag the ‘mass’ block into the Inovo Robot group block found in the workspace area of the blockly canvas. Set the mass and x,y,z parameters to define the centre of mass.



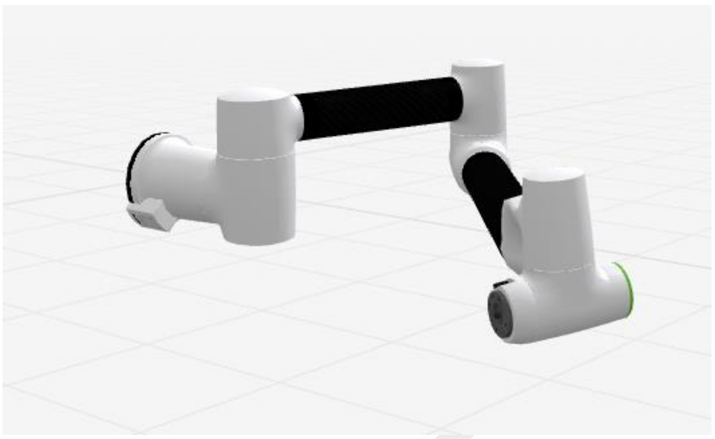
5.4. Configuring the robots mounting

The dynamic motion compensation accounts for the effects of gravity on the arm as it moves. This means that the robot’s orientation must be configured properly. This is set in the workspace for your project, when you create a new project the default orientation is assumed to be with the robot mounted on a flat level surface. If you mount the robot with the base on a wall or the ceiling you must configure this in the Inovo Robot block.

The example below shows a wall mounted configuration. Note the orientation of the inlet connector can be used to confirm the rotation of joint 1 is correct which is critical in a wall mounted setup.

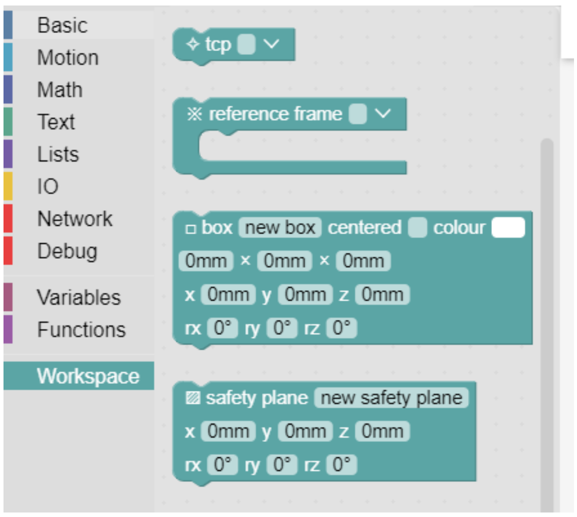


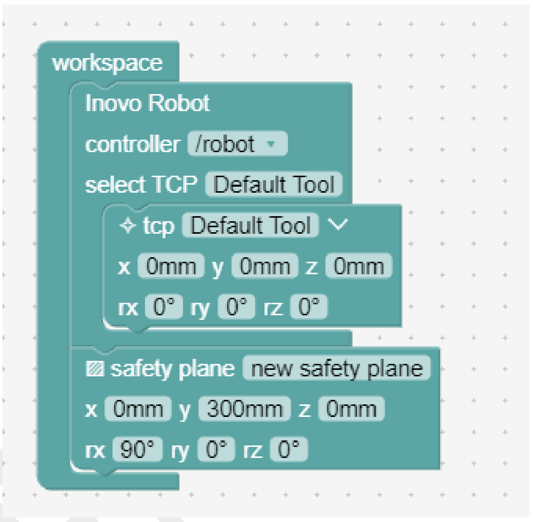
workspace
Inovo Robot
controller /robot
x 0mm y 0mm z 0mm
rx 0.0° ry 90.0° rz 0.0°
Default TCP Tool Plate



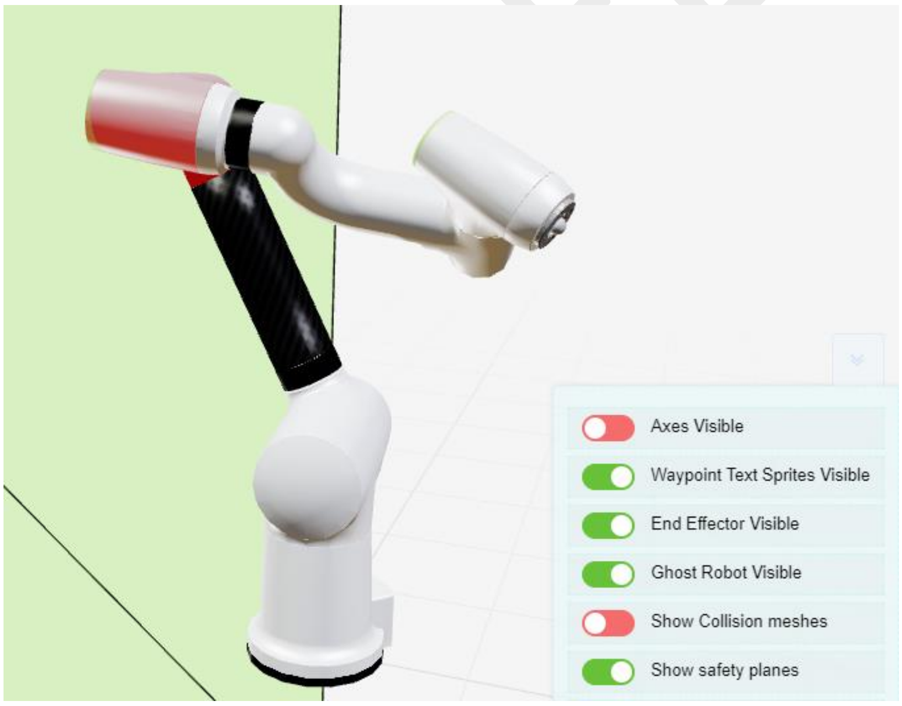
5.5. Defining Safety Planes

The safety planes feature allow you to define half planes which the robot can't cross without throwing and error and stopping the arm. These are useful if you are operating the robot in a confined space near walls or other surfaces that it could otherwise collide with. To define a safety plane drag a 'safety plane' block from the workspace toolbox on the left into the workspace group on the canvas as show below:





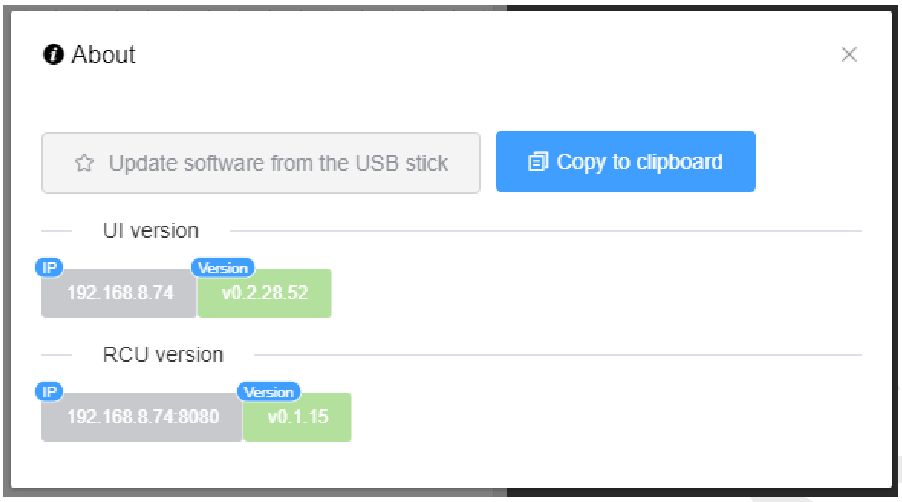
In the example below the safety plane can be seen in the visualisation because the switch is set to show collision meshes, it is seen as a translucent green surface to the left of the arm, the elbow has collided with the plane so the relevant joint is highlighted red and the arm has been disabled.



6. SOFTWARE UPDATE

6.1. Checking the current software version

Go to the File menu and select 'About' this will display the software versions of the various subsystems



UI version

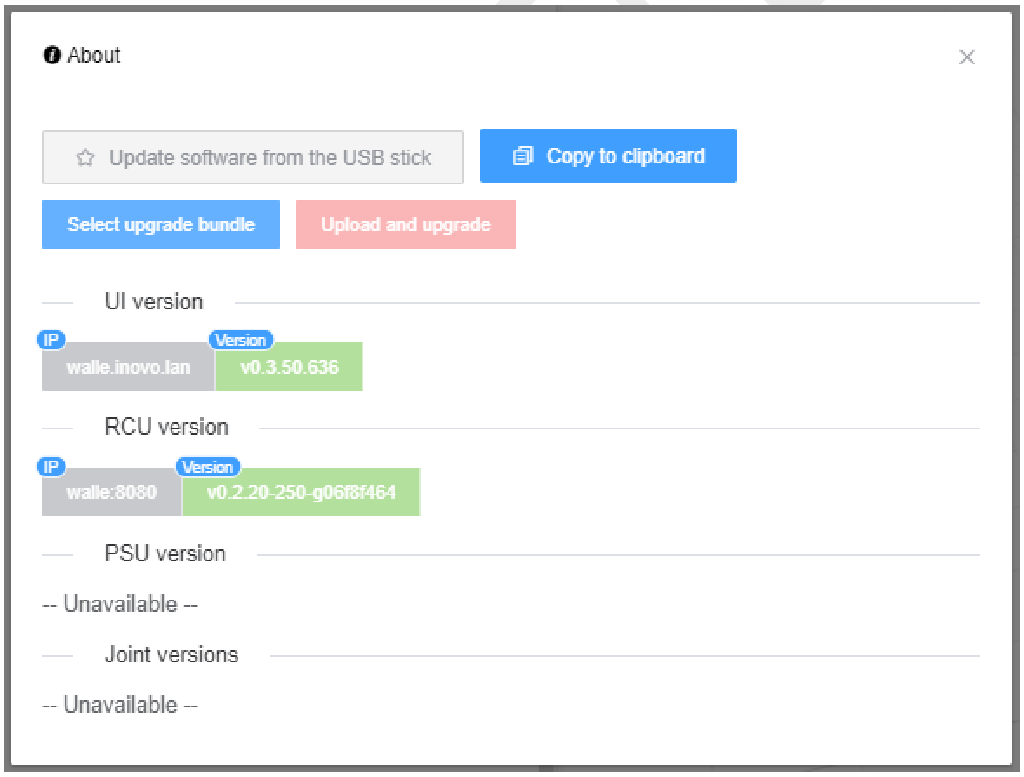
IP 192.168.8.74 Version v0.2.28.52

RCU version

IP 192.168.8.74:8080 Version v0.1.15

6.2. Software update from via the browser

Disable and power down the arm. Go to the file menu and select 'about', click on the 'select upgrade bundle' button, browse to the file provided by Inovo and click open, click 'Update and upgrade'. This process can take 2-3 minute and will reboot the controller.



UI version

IP walle.inovo.lan Version v0.3.50.636

RCU version

IP walle:8080 Version v0.2.20-250-g06f8f464

PSU version

-- Unavailable --

Joint versions

-- Unavailable --

7. Acronyms

Term	Description
Tool Centre Point (TCP)	A reference point projected on or around the wrist
Free Mode	A mode where the robot can be moved by hand but will remain static when releases
Waypoint	A position and orientation in space usually defined by six values X,Y,Z,rX,rY,rZ
Frame	A frame defines the position and rotation of the coordinate system
Tool Frame	A frame that moves with the wrist of the robot
World Frame	Usually a centre point at the base of the robot
End Effector	A tool attached to the wrist of the robot

8. Index