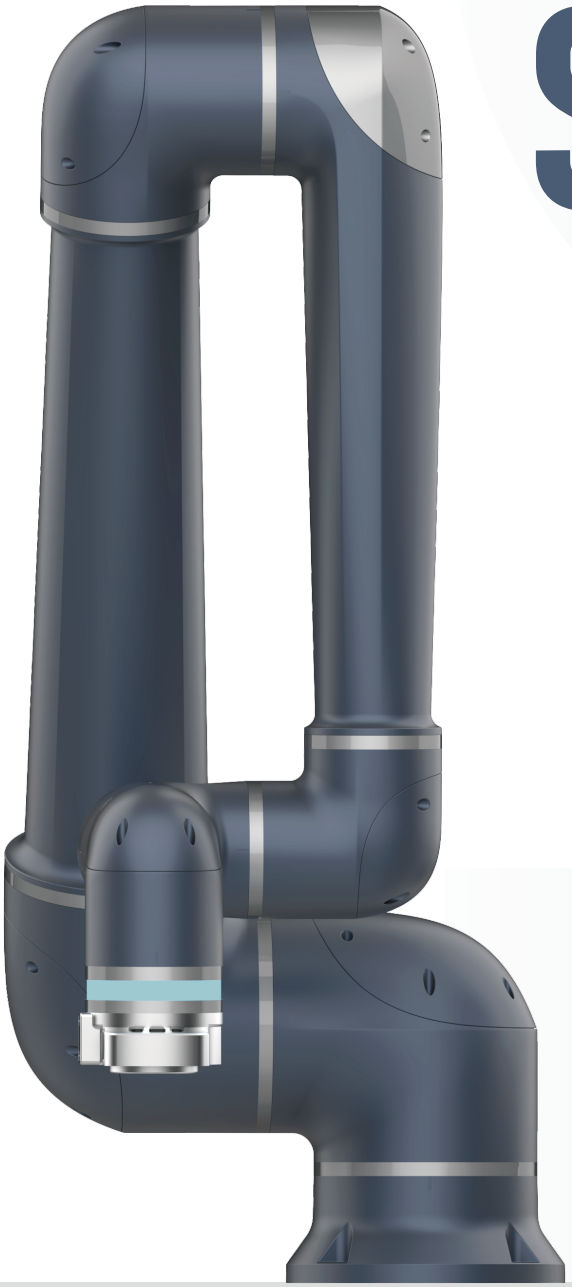
ESTUN CODROID

ESTUN CODROID S-SERIES



Collaborative Robots User Manual

ROBOTS MODEL:

S3-60 Eco	S5-90 Eco	S10-140 Eco	S20-180 Eco
S3-60 Pro	S5-90 Pro	S10-140 Pro	S20-180 Pro

CONTROL CABIENT MODEL:

COB-A03	COB-A05	COB-A10	COB-A20

NANJING ESTUN CODROID TECHNOLOGY CO. LTD

5/F, Building 1, jiangning, Double innovation Base, National University Science Park, Southeast University
No.33, Southeast University Road, jiangning District. Nanjing, 211102 jiangsu, P. R. china
400-025-3336

ORIGINAL INSTRUCTIONS

Document No.:UM202410001
Revision:V1.0
Published:2024-10-10

*Semi-finished machinery must not be put into service prior to final assembly in a complete machine that has been declared compliant with the provisions of this Directive.

Contents

1. Foreword

1.1 Thanks

Thank you for buying and using our products. This is a new generation of intelligent industrial light man-machine cooperation six-joint robot developed by our company.

1.2 Nameplate

You can find information such as the model number of the robot on the arm body.

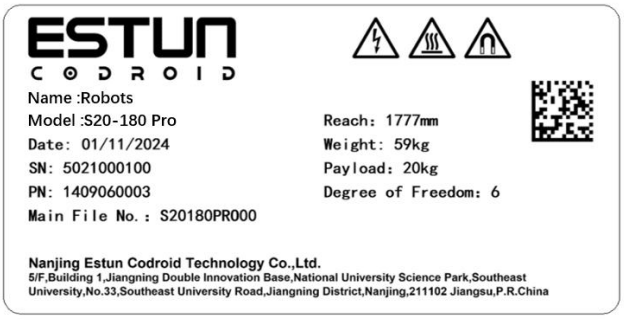


Figure 1-1 Robots body nameplate

You can find information such as the model number of the control cabinet on the control cabinet.

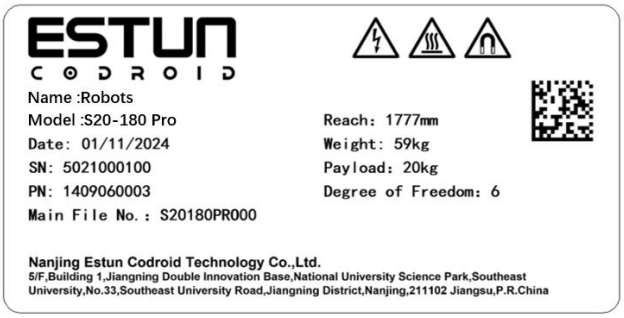


Figure 1-2 Control cabinet nameplate

1.3 How to use this manual

This manual describes the hardware composition of Kuzhuo robot and the operation of its teaching control system, which helps users to understand and master the functions, technical specifications, installation and use of Kuzhuo robot.

This document is intended for customers, sales engineers, installation and commissioning engineers, and technical support personnel.

This manual contains how to protect users and prevent machine damage methods, users should read all the

relevant descriptions in the manual and fully understand the safety precautions.

In this manual, we try to describe every situation, but because there are so many possibilities, it is impossible to record all the situations that do not need to be done or cannot be done.

1.4 Copyright Trademark Notice

Estondroid, CoDroid, CoDroid EIP, CoBrain, CoDrive, CoSense, CoSafe, CoTool are registered trademarks of Estondroid. Copyright @Nanjing EstonKudroid Technology Co., LTD.

Without the written permission of the Company, no unit or individual may extract or copy part or all of the content of this document, and may not spread it in any form.

1.5 Manual Disclaimer

Before using this product, read this manual and related technical documents published on the Internet carefully and understand the relevant information, and ensure that the robot is used under the premise of full understanding of the robot and related knowledge. We recommend that you use this manual under the guidance of professional personnel. All safety information contained in this manual is not to be regarded as a guarantee of Coolso, even if you follow this manual and related instructions, the use of the process of harm or loss may still occur.

1.6 Terms commonly used in this manual

1.6.1 Robots

A stationary or mobile automatic machine used in industrial automation that can be automatically controlled, reprogrammed, multipurpose, and programmed with three or more axes.

1.6.2 Maximum working space

The space that can be skimmed by the moving parts of the robot, plus the space that can be skimmed by the end effector and when the workpiece is moving.

1.6.3 Accuracy

Deviation of position and attitude between instruction distance and real-to-distance mean.

1.6.4 Repeatability

The consistency of the distance after repeated motion n times in the same direction for the same instruction distance.

1.6.5 Trajectory accuracy

The maximum trajectory deviation in position and attitude along the resulting trajectory.

1.6.6 Accuracy of trajectory repetition

The consistency of the actual trajectory when the robot repeats the same instruction trajectory n times.

1.6.7 Tool center point (TCP)

A point set for a specific purpose with reference to the mechanical interface coordinate system. (Refer to GB/T 12643-2013, Definition 4.9)

1.6.8 PayLoad

Refers to all loads loaded on the robot flange without the weight of the tool.

1.6.9 Protective stop

A form of interruption of operation that allows the motion to terminate in an orderly manner for safety and preserves program logic for restart.

1.6.10 Singularity

A condition in which two or more axes of a robot are collinear, resulting in uncertain robot motion and speed.

Material Number	Version	Release Date	Description
1210002200	V1.0	20241201	Initial release

1.7 Revision record

2. Safety information

2.1 Effectiveness and Liability

The information in this manual does not cover the design, installation and operation of a complete robotic application, nor does it cover all peripherals that may affect the safety of this complete system. The complete system shall be designed and installed in accordance with the safety requirements established in the standards and codes of the country in which the robot is installed.

It is the responsibility of the integrator of Estun Codroid to ensure that the applicable laws and regulations of the relevant country are followed and that there are no significant hazards in the complete robot application. This includes but is not limited to the following:

- Conduct a risk assessment of the entire robotic system
- Connect together other machinery and additional safety equipment defined by the risk assessment
- Make the appropriate safety Settings in the software
- Make sure that any security measures are not modified by the user
- Verify that the design and installation of the entire robot system is accurate
- Clear instructions for use
- Mark the relevant logo and contact information of the integrator on the robot
- Gather all documentation in the technical file; Including the risk assessment and this manual

2.2 The warning signs agreed in the manual

The following safety warning signs may appear in this manual, and they represent the following:

Warnings

This sign indicates a situation that may cause a dangerous use of electricity and, if not avoided, could result in death or serious injury or serious damage to equipment.

Warnings

This sign indicates a potentially hazardous situation that, if not avoided, could result in death or serious injury.

Warnings

This sign indicates a potentially hazardous electrical situation that, if not avoided, could result in injury or serious damage to equipment.

Warnings

This sign indicates a potentially hazardous situation that, if not avoided, could result in injury or serious damage to equipment.

Warnings

This sign indicates a potentially hazardous electrical situation that, if not avoided, could result in injury or serious damage to equipment.

Warnings

This sign indicates a hot surface that may pose a hazard and could cause injury if touched.

Be Careful

This sign indicates a condition that, if not avoided, can result in serious damage.

2.3 Safety precautions

- Make sure the robot arm and tools/end effectors are both properly and securely bolted into place. Make sure the robot's arms have enough room to move freely.
- Ensure that safety measures and/or robot safety configuration parameters have been established to protect programmers, operators and bystanders as defined in the risk assessment.
- Do not wear loose clothing and do not wear jewelry when operating the robot. Make sure long hair is tied back when operating the robot.
- Do not use the robot if it is damaged, such as when the joint cap is loose, damaged, or removed.
- Do not extend your fingers into the control box.
- Do not connect any safety devices to standard IO interfaces. Only use the secure IO interface.
- Make sure to make the correct mounting Settings (e.g. mounting Angle of robot, weight in TCP, TCP offset, security configuration).
- Use of drag and drop instruction during installation is only allowed if it has passed a risk assessment.
- Tools/end effectors and obstacles must not have sharp corners.
- Make sure to keep warning people's heads and faces out of reach of the robot being operated or about to begin operating.
- Pay attention to the robot's movement when using the teach box.
- If the risk assessment has determined, do not enter the safe area of the robot, or touch the robot while the system is operating.
- Linking different machines may aggravate the hazard or introduce new ones. Always perform a thorough risk assessment of the entire installation.
- Never tamper with the robot. Changes to the robot may cause unforeseen dangers. Robot authorization reconfiguration is subject to the latest version of all relevant service manuals.
- Ensure that the robot user is aware of the location of the emergency stop button and is instructed to activate the emergency stop in case of an emergency or abnormal situation.
- The robot and the control box generate heat during operation. Do not touch the robot when it is running or has just stopped running. You can cool the robot by turning it off and waiting for an hour.
- When the robot is attached to or working with machinery that can cause damage to the robot, it is highly recommended that all functions of the robot and the robot program be inspected separately.
- Do not constantly expose the robot to magnetic fields, combustion, potential explosions, radio interference, liquids, etc., as this may damage the robot.
- Robot systems are not allowed to be used in an explosive or potentially explosive

environment.

- The state in which the robot arm appears to have stopped while the device is in operation, as it waits for a signal to start. Should also be considered in motion, do not approach the robot arm.
- During the handling, installation, operation and maintenance of the robot, the operator should wear safety protective equipment such as safety gloves, glasses and anti-smashing shoes to avoid dangerous injuries.

2.4 Safety requirements

The safety features are generally in accordance with ISO 10218-1 and specifically meet the following requirements.

When a safety-related control system is required, the safety-related components should be designed to:

- A single failure of any one component will not result in a loss of safety function.
- As far as reasonably practicable, a single failure should be detected on or before the next time a safety function is required.
- The safety function is always in an active state when a single fault occurs and should remain in a safe state until the detected fault is fixed.
- All reasonably encountered faults should be checked out.

This requirement is considered to be equivalent to the Class 3 structure described in ISO 13849-1. Class 3 is usually accomplished by redundant circuitry. Safety functions and robot controllers comply with the performance level (PL) d specified in ISO 13849-1.

2.5 Safety Disclaimer

The information in this manual does not cover the complete design, installation and operation of the robot for use with other equipment, nor does it cover the possibility of the above use affecting peripheral equipment.

The safety of a robot installation depends on how the robot is integrated, and integrators need to conduct risk assessments for the design and installation of the system in accordance with the laws and regulations and safety codes and standards of the host country.

Risk assessment is one of the most important tasks that the integrator must complete. The integrator can carry out the risk assessment process with reference to the following standards:

- ISO 12100:2010 Safety of machinery - General Principles for design - Risk assessment and risk reduction;
- ISO 10218-2:2011 Robots and robotic equipment - Safety requirements - Part 2: Systems and integration of industrial robots;
- RIA TR R15306-2014 Technical report on industrial robots and robotic systems - Safety requirements, task-based risk assessment methods;
- ANSI B11.0-2010 - Safety of Machinery; General requirements and risk assessment.

2.6 Limitation of liability

Any safety information contained in this manual should not be taken as a guarantee of our robot, many things can not be described in all aspects, still may cause injury or damage.

We are committed to continuously improving the reliability and performance of our products and therefore reserve the right to upgrade our products without prior notice. The Company is not responsible for any errors or omissions in this manual, and reserves the right of final interpretation of this manual.

2.7 Stop Category

stop category 0 Cat. 0	Uncontrolled shutdown, which stops the robot by immediately cutting off the power to the actuator.
stop category 1 Cat. 1	Controlled stop, where the actuator actively brakes but does not ensure that the robot stops on the motion trajectory. After the robot stops, cut off the power supply.
stop category 2 Cat. 2	Controlled shutdown, where the actuator actively brakes and ensures that the robot stops on the moving path. After the robot stops, do not cut off the power supply.

According to IEC 60204-1 standard, Kuzo robot sets three stop categories, namely stop category 0 (Cat.0), stop category 1 (Cat.1), stop category 2 (Cat.2). Among them, stop class 0 is the uncontrollable stop, stop class 1 and stop class 2 are the controllable stop.

According to IEC 60204-1 and ISO 13850, emergency equipment is not a safety guard. They are complementary protective measures and are not intended to prevent injury.

In the event of an emergency, pressing the emergency stop button can immediately stop all movement of the robot and lock it. The emergency stop cannot be used as a risk reduction measure. However, it can be considered a secondary protective device and should only be used in emergency situations.

If you need to stop the robot movement under normal circumstances, please use other methods. After the risk assessment, if the emergency stop button needs to be installed, the emergency stop button must meet the requirements of IEC-60947-5-5.

When the emergency stop button is pressed, the robot system will cut off the power supply of the robot, and the brake device between the joints of the robot will automatically lock the joints, but under the action of gravity, the slight movement of the robot body is normal, but it may also cause the risk of clamping injury or collision with the human body.

The implementation of the stop category depends on the implementation of the joint drive, and refer to IEC 61800-5-2 for further description.

For emergency stop through the safety interface, refer to Section 5.3.2 for protective stop functions.

2.8 Risk assessment

Before installing or using this product, the user must carry out the necessary risk assessment according to the conditions of use, and carefully read the company's stated value of possible residual risks. For the relevant content, please read and refer to the corresponding software and hardware version instructions.

2.9 Safety Features

The safety features of the CoDroid robot are shown in the table below.

Safety Features	Instructions
Emergency Stop	Start Stop Category 1 when the emergency Stop button is pressed.
Protective Stop	When the relevant signal input is low, start stop category 2. This feature needs to be reset manually.
Safety rated deceleration control	When the relevant signal input is low, the TCP speed will be reduced to the limit.
Joint position Limit	Set the limit range of allowable joint position.
Joint speed limit	Set a limit range of allowable joint speed.
Joint torque limit	Set the limit range of allowable joint torque.
TCP position Limits	Set the limit range of allowable TCP positions.
TCP Speed Limit	Set the maximum TCP speed.
TCP Torque Limit	Set the maximum torque for TCP.
Robot power limits	Limit the maximum power of the robot.
TCP directional restrictions	Set the directional limits allowed by the tool.
Safety level monitors downtime	Activate Stop Category 2 when the relevant signal input is low. This function can be reset when the relevant signal input signal is low.
Speed and distance monitoring	Maintain a minimum protective distance between the operator and the robot. When the separation distance decreases below the protective distance, the robot system stops. When the operator leaves the robotic system, the robot can automatically resume its motion.
Power torque Limitation	Limit the maximum power and torque of the robot.

2.10 Scram recovery

The scram button will be locked after being pressed. You need to rotate the button according to the logo on the button to release the lock. After being unlocked, the alarm can be cleared using the control software. Then, the alarm can be powered on or enabled to recover from the emergency state.

2.11 No power to forced drive

In the event of an emergency, if it is necessary to move the robot joints, but it is not possible or necessary to energize the robot, a no-power forced drive can be used.

To perform a forced drive without electricity, you must push or pull the robot arm vigorously to move the joint. Each joint brake has a holding brake that allows the joint to move while subjected to high levels of torque.

Movement without electric drive is used only in emergency situations. And has an effect on the life of the brake holding device.

2.12 Stop time and stop distance

Provide referable stopping distance and stopping time data for joint 1 (frame), joint 2 (shoulder), and joint 3 (elbow) :

• Category 0

• Category 1

• Category 2

The test for joint 1 is performed by horizontal movement, where the axis of rotation is perpendicular to the ground.

During the joint 2 and 3 tests, the robot followed a vertical trajectory, the axis of rotation was parallel to the ground, and a stop operation was performed as the robot moved down.

The robot arm was fully extended horizontally.

The general speed of the robot is set to 100% and moves at the maximum joint speed.

The maximum payload that the robot can handle.

The following table shows the stopping distance and stopping time measured under the above conditions when the 3kg robot triggers a category 1 stop. Please consult our technicians for test data of other models.

The following data is used as a reference value. The results of stopping distance and stopping time may vary according to your application scenario and operating conditions of the robot.

Position	Stopping Distance (rad)	Stop time (ms)
Joint 1 (frame)	0.30	282
Joint 2 (shoulder)	0.29	287
Joint 3 (elbow)	0.29	237

2.13 Storage, use, and transportation conditions

- The ambient temperature during storage and operation should be between 0 and 40°C;
- Low humidity, relatively dry place. Relative humidity of 10%-90%, no condensation;
- Places with less dust, dust, oil smoke and water;
- Flammable and corrosive liquids and gases are not allowed in the working area;
- The vibration or impact energy of the electric control cabinet is small (vibration is less than 0.5G);
- There should be no large electrical noise sources nearby (such as gas shielded welding TIG equipment, etc.);
- There is no potential danger of collision with mobile equipment (such as AGV);
- The control box should be installed outside the robot's range of motion (outside the safety fence);
- The control box should be at least 200mm away from the wall to keep the heat dissipation channel clear.

2.14 Identification of control cabinet and body

Following identification, nameplate. Attach to the location where specific hazards may occur. In order to avoid accidents, be sure to follow the instructions and contents of the signs when operating. Do not tear, damage or remove the sign at will. Use extreme caution when handling the parts or units attached to the sign, nameplate, and their immediate area.

A		Equipment must be operated and maintained by professionals with good personal protection. Verify that the hardware setup instructions are followed. Avoid body or other equipment injury or personal injury caused by incorrect use of the product
B		Do not open the control cabinet and body to touch the internal electronic devices and circuits to avoid electric shock. There is a risk of fire or electric shock. Be sure to use proper personal protection against arc flashover hazards, failure to follow this specification may result in personal injury or death.
C		Dangerous hot surfaces that may cause injury if touched.
D		There is a magnetic field inside the robot body, which can cause harm to the body and electronic equipment.
E		Product nameplate, confirm the basic product information

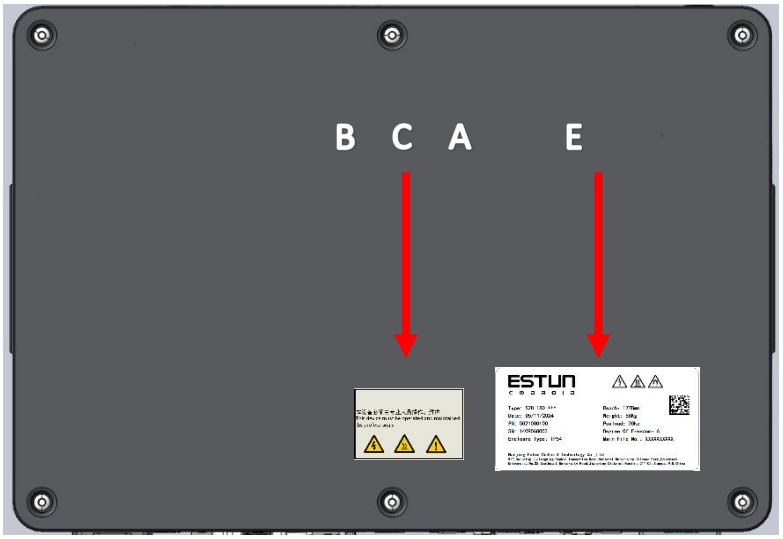


Figure 2-1 Label and nameplate location for the control cabinet

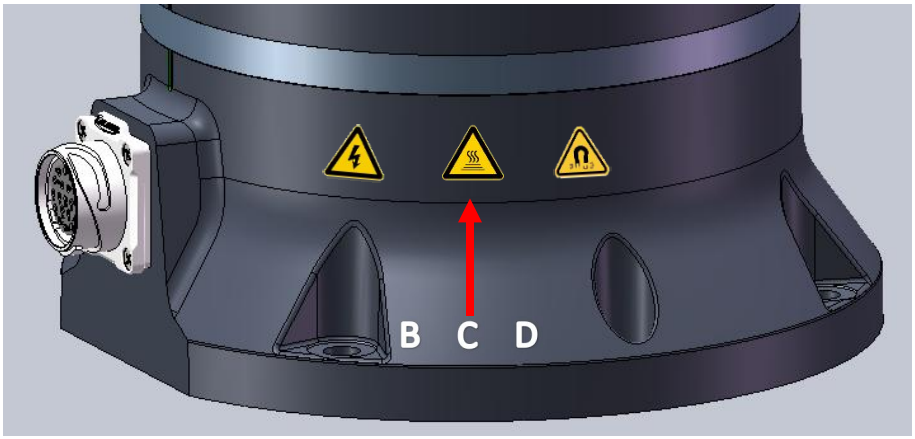


Figure 2-2 Position of the nameplate on the 10kg or smaller body

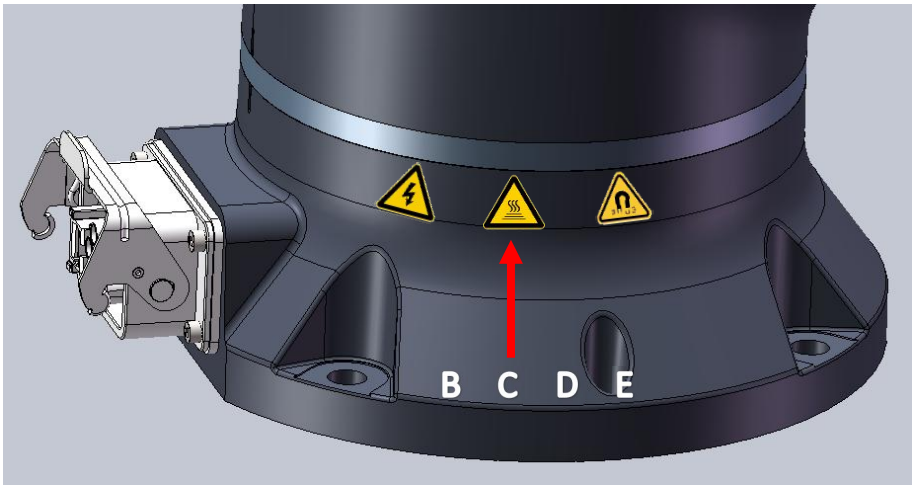


Figure 2-3 Location of the 20kg body identifier and nameplate

3. Quick use

3.1 Confirmation of product contents

Before the robot is used for the first time, the user needs to read and understand the safety information in this manual and the safety configuration parameters in the Settings.

After the product arrives, please check its shipping list, the standard shipping list includes the following 5 items (optional information will be provided separately). The robot body and the control cabinet are in two separate packages. The body package contains only the body, while the control cabinet package contains the controller body, hand operator, body and controller cables, power cables, etc.

Figure 3-1 Contents of packing-

3.2 Robot installation

3.2.1 Transportation

Keep the original packaging when shipping. Keep packing materials in a dry place; It may be necessary to wrap and move the robot later.

Move the robot from the packaging material to the installation location:

When the S3-60,S5-90,S10-140 robot arms are installed, both die cast linkages of the robot arms can be lifted at the same time. Hold the robot until all mounting bolts of the robot base are fastened. Refer to 3.2.2 Handling.

For details about how to carry the S20-180 robot arm, see 3.2.2 Carrying

Warnings

When moving the equipment, the operator should wear safety protective equipment such as safety gloves, glasses, and anti-smashing shoes to avoid dangerous injuries during the moving process.

Warnings

• Make sure you are not carrying excessive weight on your back or other body parts when lifting the equipment. Use appropriate lifting equipment.

All regional and national lifting guidelines should be followed. General Robot is not responsible for any damage caused by the transportation of the equipment.

• Ensure that the robot is installed according to the mechanical interface instructions in 3.2.3 Installation and 4.5 Installation Interface in this note.

• If the robot needs to be positioned precisely, the two reserved holes can be positioned by means of pins.

Warnings

• Ensure that the robot is installed correctly and in a position that avoids vibration.

• The power supply of the robot arm must be turned off during installation and disassembly to prevent accidents.

To turn off power:

- Return to packing position when disassembling
- Turn off the robot by pressing the power on the hand operator
- Disconnect the power plug

3.2.2 Carrying

warn

When moving the equipment, the operator should wear safety protective equipment such as safety gloves, glasses, and anti-smashing shoes to avoid dangerous injuries during the moving process.

3.2.2.1 Robot handling of 10kg or less

1. Transportation and unpacking

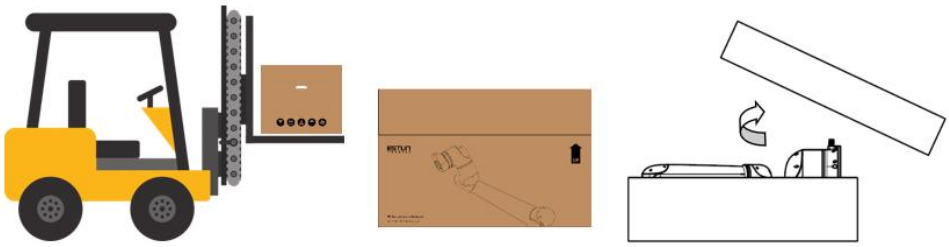


Figure 3-2 Transport and unpacking diagram of 10kg and below-

2. Install the lifting belt and use the lifting hook to lift the robot arm

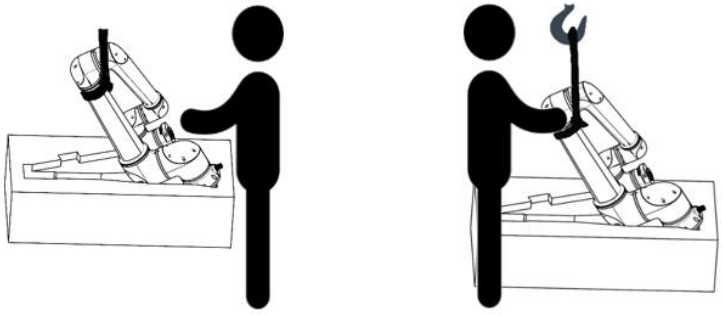


Figure 3-3 Schematic diagram of the position of the sling of 10kg and below

3. Installation

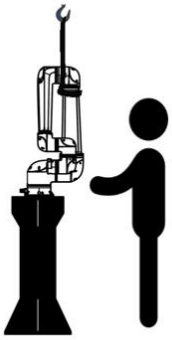


Figure 3-4 Installation diagram of 10kg or less

3.2.2.2 20kg robot handling method

1. Transportation and unpacking

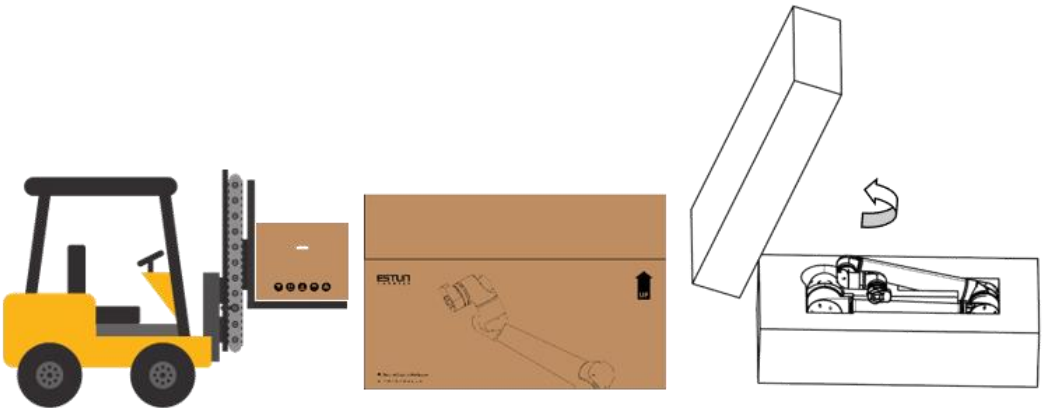


Figure 3-5 Transport and unpacking of 20kg-

2. Install the lifting belt and use the lifting hook to lift the robot arm

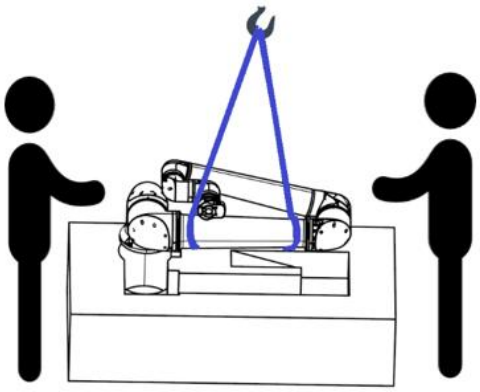


Figure 3-6 Schematic diagram of the position of the 20kg sling-

3. Installation

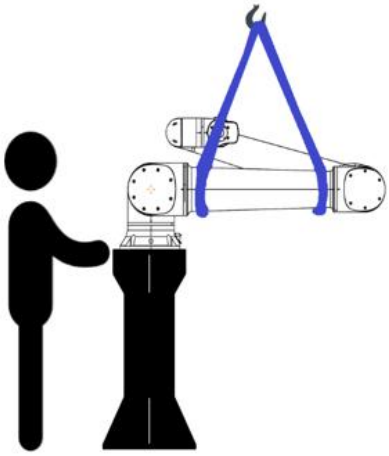


Figure 3-7 20kg installation diagram

warn

Lifting or moving heavy parts can cause injury.

• Lifting equipment/lifting AIDS may be required.

Warnings

Incorrect assembly of components and/or wiring may result in injury.

• Personal protective equipment (shoes, glasses, gloves) may be required.

• Injury and property damage may result if lifting devices appropriate for the weight of the robot are not used

• The lifting device should be able to lift up to 59kg (robot only).

• The lift unit should be able to lift 79kg of weight (robot and payload).

Hoisting belt use: Hoisting belt selection should meet the following standards under the premise of meeting the load of this product:

European Standard:

• BSEN1492-1:2000+A1:2008 Textile sling - Safety - flat webbing sling, made of man-made fiber, for general use.

• BSEN1492-2:2000+A1:2008 Textile sling - Safety - Circular sling, made of man-made fibre, for general use.

JB standard:

• JB/T8521.1-2007 Woven sling safety - Part 1: Flat lifting belts of synthetic fibres for general use

• JB/T8521.2-2007 Braided sling safety - Part 2: General purpose synthetic fibre round sling

Warnings

Using round slings without checking them can result in injury.

• Check the sling before and after each use.

• If possible, check the sling during use

Warning

Using a damaged round sling may result in injury.

• Check the sling before and after each use.

• Do not use if the sling is cracked, torn, or the stitches are loose.

• Do not use the sling if it shows signs of heat damage.

• Protect the sling from contact with sharp edges and friction.

• Do not tie knots on the sling.

• If possible, check the sling during use.

3.2.3 Install

Warning

Before feeling safe about the equipment, confirm that the operator should wear safety protective equipment such as safety gloves, glasses, and anti-smashing shoes to avoid dangerous injuries during installation.

As shown, use bolts of at least Class 12.9 strength and mounting holes in the base to mount the robot arm. See Section 4.5 for the mounting dimensions of the robot base.

Suggestions for mounting torque are as follows:

Item	S3-60	S5-90	S10-140	S20-180
Bolt Specifications	M6	M8	M8	M12
Quantity	4	4	4	4
Flat pad GB /T 97.1-2002	Φ 6	Φ 8	Φ 8	Φ 12
Dowel pin specifications	Φ 4	Φ 6	Φ 8	Φ 8
torque	>=10N·m	>=20 N·m	>=35 N·m	>=70 N·m

The robot needs to be mounted on a strong and vibrating-free bearing surface that withstands at least 10 times the full torsional force of the first joint and at least 5 times the weight of the robotic arm.

Figure 3-8 Body mounting

The robot can be installed in any position and posture, and supports various installation methods such as lifting and side mounting. In the case of non-vertical installation, the robot installation Angle needs to be set in the robot setup item. For non-vertical installation, please consult our technical personnel for the installation method of the robot body and the setting method of the installation Angle.

Cable connection

Before starting up, connect the - robot cable according to **the cable connection diagram in Figure 3-9.**

Network cable: Connect the controller and tablet computer to control the robot action;

Hand operator: used to control the robot emergency stop, enable and switch;

Control cable: used to provide power and communication for the robot body;

Power cable: to provide power to the robot system.

Warning

Before powering the robot, check that the voltage and frequency of the power supply meet the requirements, connecting the wrong voltage can cause the robot to malfunction.

Warning

A power cord is included with the package, but as single-phase power outlets are used in different countries and regions, purchase your own power cable that meets the requirements according to the customer's region.

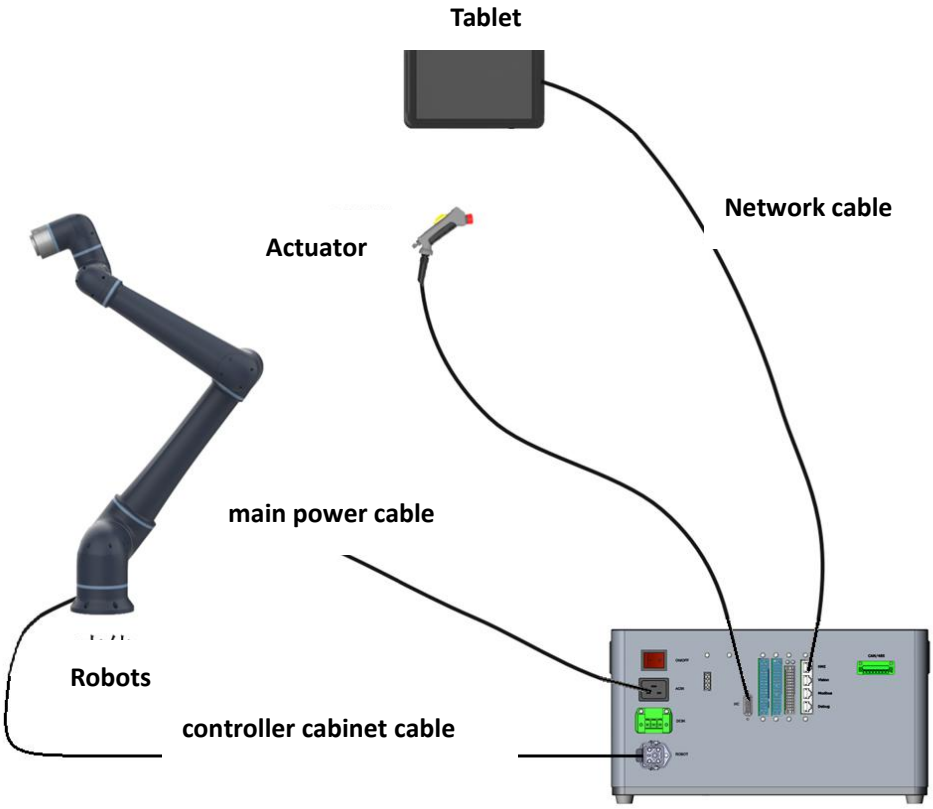


Figure 3-9 Schematic diagram of cable connection-

3.2.4 Operating position Layout

Operator and robot body, control cabinet equipment position according to **Figure 3-10 Schematic diagram of operating position layout**.

Suggestion: When the robot is running, the operator should stand outside the arm span of the robot body to ensure the safety of personnel.

This robot includes collision detection and complies with IOS/TS 15066:2016 standard

Ensure that the operator operates the robot outside the robot workspace area by referring to Section 4.2 Workspace. Do not operate the robot when a person is inside the robot workspace area.

Place the control cabinet by referring to the contents of the control cabinet in Section 4.7, and ensure that the control cabinet is installed in a ventilated, flat, and non-violent vibration environment.

Warning

Lay out the robot body, control cabinet, in the correct way. The wrong layout will cause the robot to malfunction or be damaged.

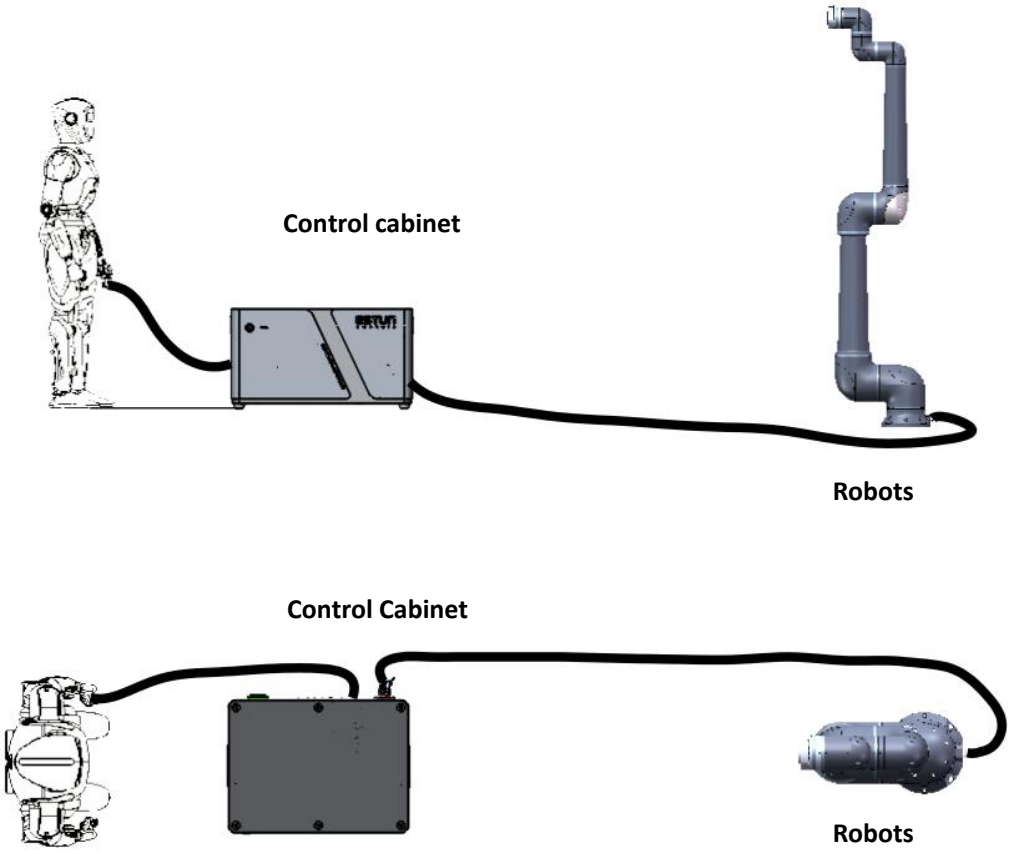


Figure 3-10 Schematic diagram of operating position layout

3.3 Start to use

Having done the above, you are ready to start using the robot.

3.3.1 Power on

1. When the robot cables are connected correctly, turn on the power and turn on the switch at the power connection of the controller. At this time, you can hear the fan of the controller start to work. At this time, press the power button on the front of the control box.

2. Wait for the end light belt of the robot to become white steady on, the small screen at the end of the arm shows communication [real time], and the operation [normal], it means that the controller has been started and the robot itself has successfully communicated with the controller. At this time, you can log in to the web page to control the robot.

3. Open the tablet computer and change the static IP address of the tablet computer to: 192.168.101.XXX through setting.

4. Open the browser, enter the IP **address of the robot in the** address **bar**: 192.168.101.100:9098**, and** press Enter to jump to the login page, as shown in Figure 3-11**. -** If it cannot be redirected, please check the IP address of the tablet; If it still doesn't redirection to the login page, contact the aftersales person.

Figure 3-11 Login page

5. Enter the account number and password to log in to the control page, the initial

account number and password can be used as follows, different accounts have different permissions, you can see the appendix for details.

Account	Password
user	123456
admin	123456

6. After login, you can go to the robot control page, and then you can power on the robot. Before powering on the robot, ensure that the emergency stop button on the hand operator is reset and no human or device is in the motion range. Click the button in the "3D Simulation" view, as shown in the robot control interface in Figure 3-12. When you hear the sound of the lock release at the joint, it indicates that the joint is powered on. 

-

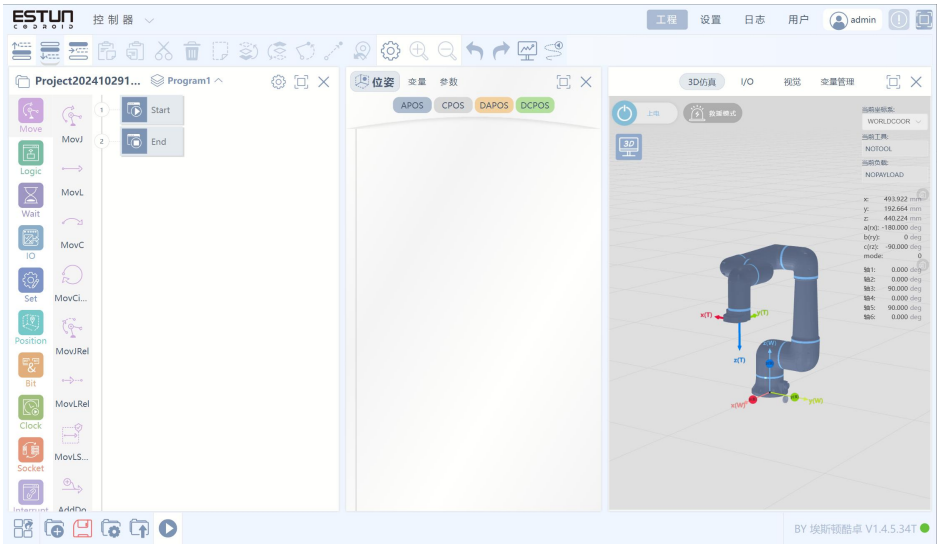


Figure 3-12 Robot control interface

7. As shown in Figure 3-13, the main interface indicates that the robot is powered on successfully, and then the robot can be controlled to move.-

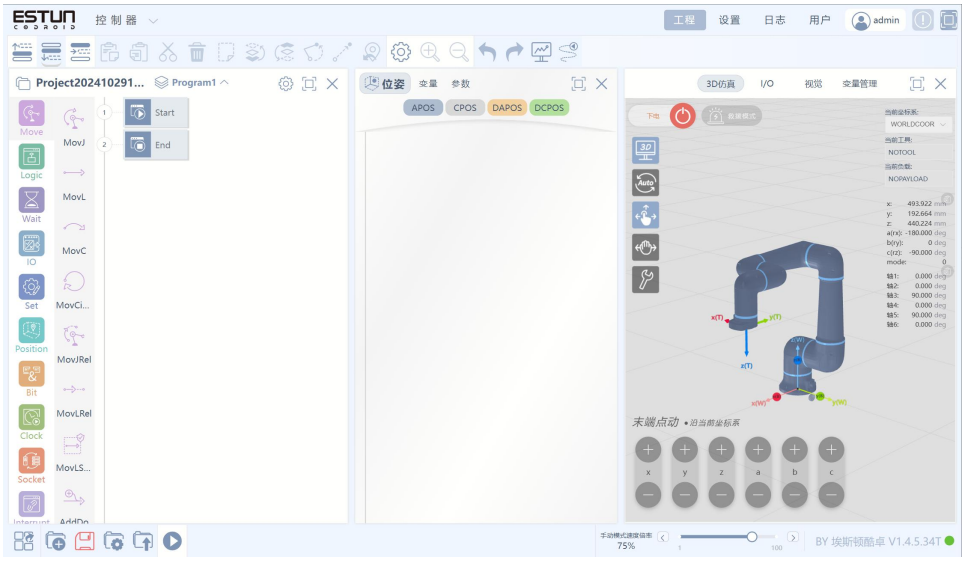


Figure 3-13 Main interface-

3.3.2 Writing programs

In manual mode, the robot can perform closing node movements and end point movements.

Close node action: The robot can be controlled for single joint movement, and the speed of the point can be adjusted by manual speed adjustment ratio. a1, a2, a3, a4, a5, a6 represent the six joints of the robot respectively.

End point motion: The robot can be controlled to move in Cartesian coordinate system, the speed of the point motion can be adjusted by manual speed adjustment ratio, and the reference coordinate system when the robot moves can be changed by switching along the current coordinate system and along the tool coordinate system. x,y, and z represent the direction of the three axes of the reference coordinate system respectively, and a,b, and c represent rotation around the reference coordinate system x,y, and z respectively.

1. In manual mode, the robot is controlled to move to the target point;

2. Click and double click to record a point P1; 

3. Move the robot to another position and repeat steps 1 and 2 to add the second point, as shown in Figure 3-14;

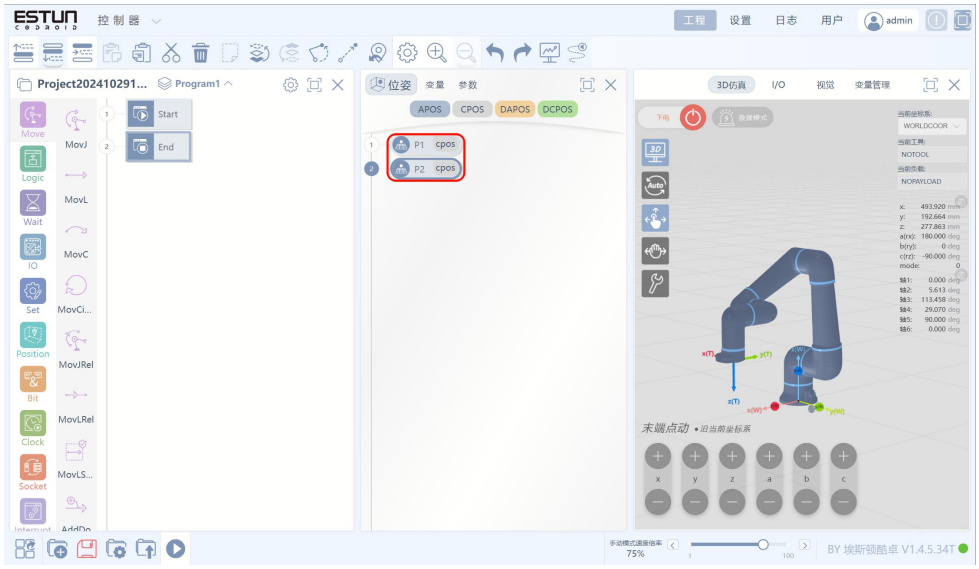


Figure 3-14

4. Select in the left command bar, double-click or drag the command directly, you can add a motion command in the right program tree; 

5. Select the program tree, click, you can appear the corresponding parameter bar of the instruction, the target position select just set point P1, the target speed select the system default V100, when the instruction parameter page does not appear the red error word, it means that the instruction has been set; 

6. Repeat step 5 to add one more instruction and set the parameters.

7. Adding an instruction in Logic and selecting the jump node as Start can make the program run continuously and repeatedly, as shown in Figure 3-15 below. 

-

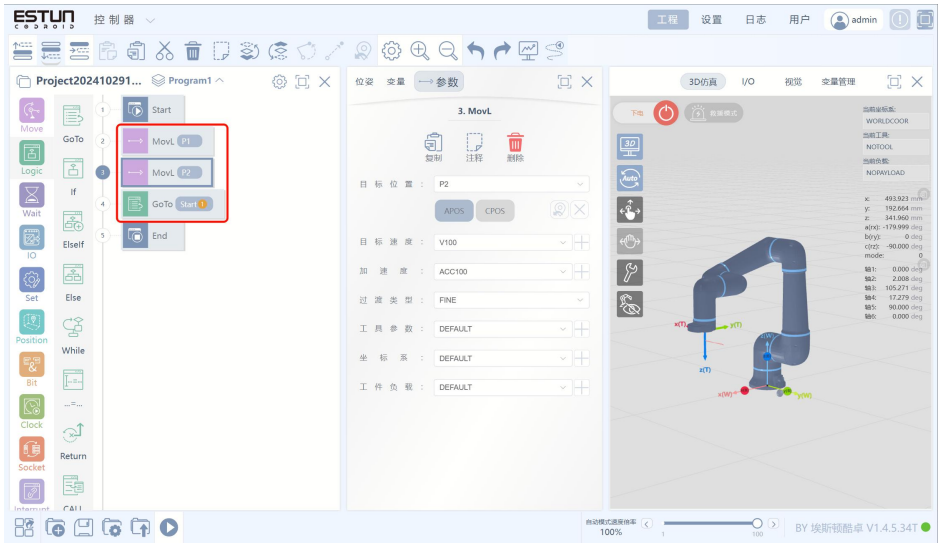


Figure 3-15

8. Click, save the program, after saving, there will be a pop-up prompt, and the icon is no longer red; 

9. Click and select OK to switch the robot into automatic mode; 

10. Click and select Auto Execute, and the robot will move from P1 to P2. 

11. Adjust the automatic mode speed multiplier to adjust the robot running speed.

12. Click to pause the robot program running, the robot will pause the movement at the same time. 

13. After the robot is paused, click to resume the robot program. 

14. Click to stop the robot program running, the robot will stop the movement at the same time. 

15. If you manually move the robot or stop the robot after the program is paused, you need to switch to the manual mode first, and then switch the robot to the automatic mode again before continuing to run the program. 

3.3.3 Power Down

Adjust the robot attitude to the right position, click, and the robot will power down, then press and hold the power button on the controller until the steady green light turns off the release button. 

4. Mechanical hardware and installation

4.1 Body composition

Kuzhuo S series robot, including 6 rotating motion joints, the upper arm and the lower arm two connecting rods. The base of the robot arm is equipped with a navigation connector, the end of the robot arm is equipped with a button and an indicator light, and the side of the tool flange is equipped with a button, a screen and a navigation connector.

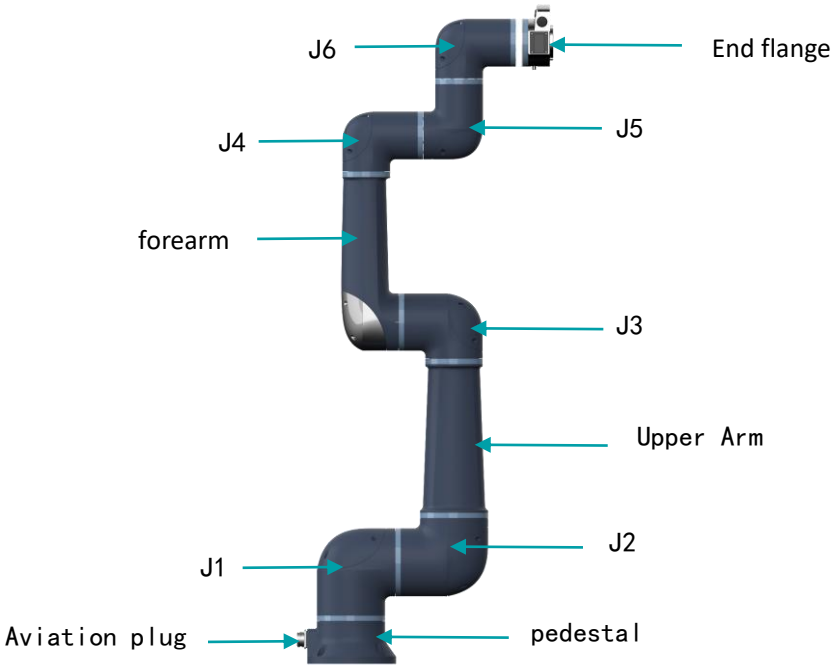


Figure 4-1 S-series robot composition

4.2 Work space

When choosing where to install the robot, be sure to consider the cylindrical space directly above and below the robot. Avoid moving the tool towards the cylinder space, as this will enter the singularity and cause the joints to move too fast during the movement, resulting in inefficient robot work and difficult risk assessment.

Warnings

When the robot is operating in manual mode (taught), the person should be outside the safety space.

The emergency stop button of the robot manual operator must be within reach in manual mode, and at least one emergency stop switch needs to be set outside the robot's range of action. When the robot does not set any action limit, the robot's action range is the maximum action range of the body. The robot action limit can be set so that all operations do not fall outside the maximum action range of the robot body.

The robot operator should be placed in an area that the robot cannot reach, and the user should also confirm that the robot movement will not reach the range of motion that can press the hand operator.

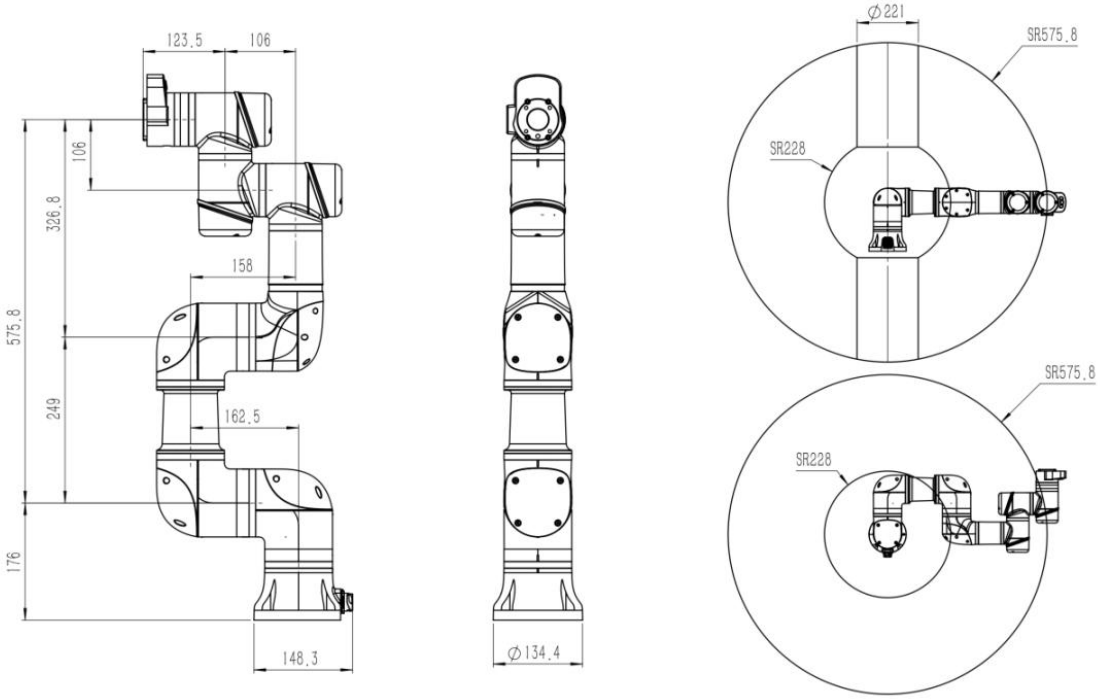


Figure 4-2- S3-60 size and working space

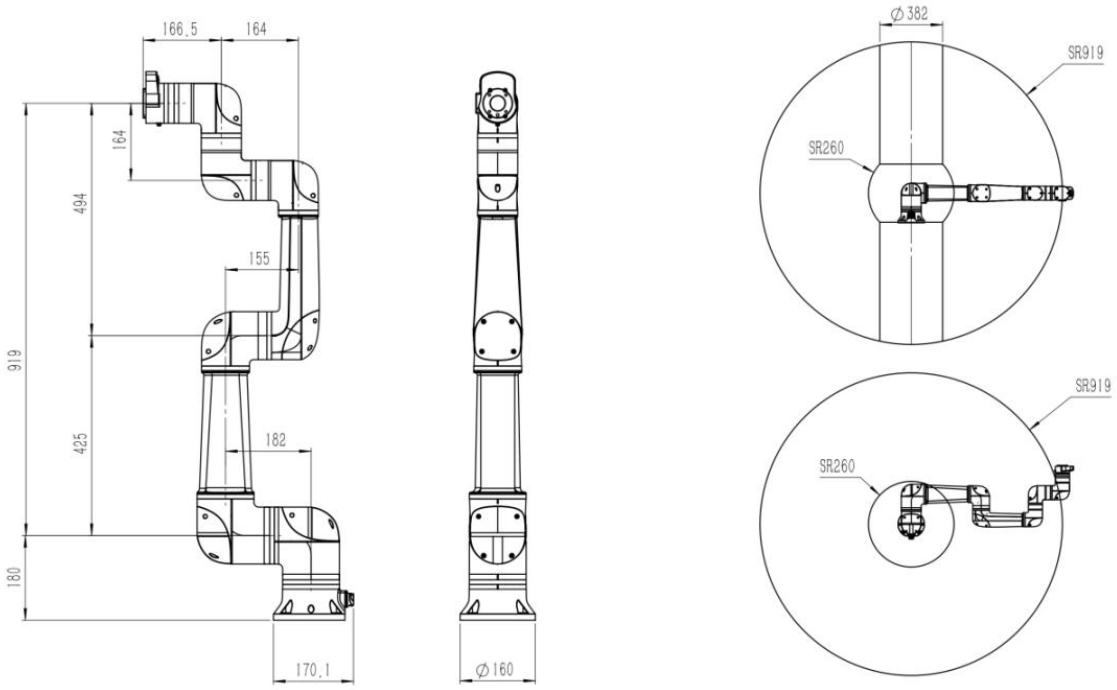


Figure 4-3 - Dimensions and **working space of S5-90**

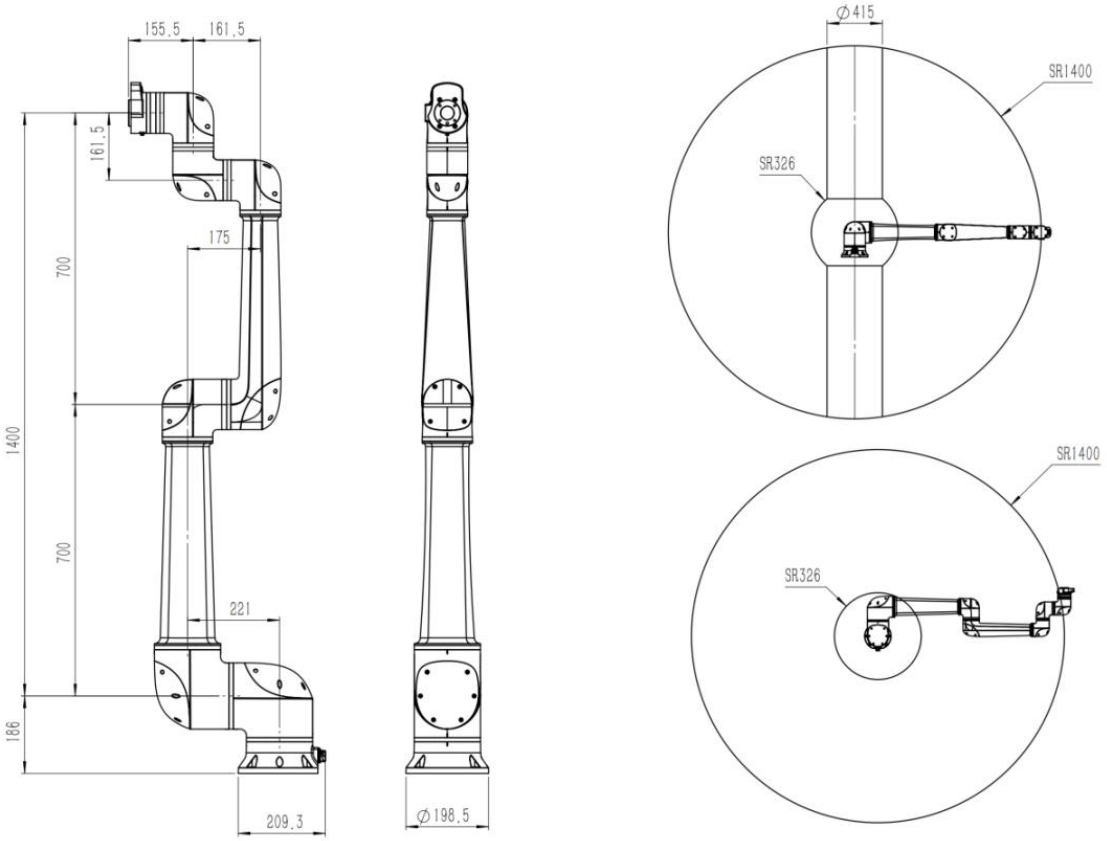


Figure 4-4 Dimensions and working space of S10-140-

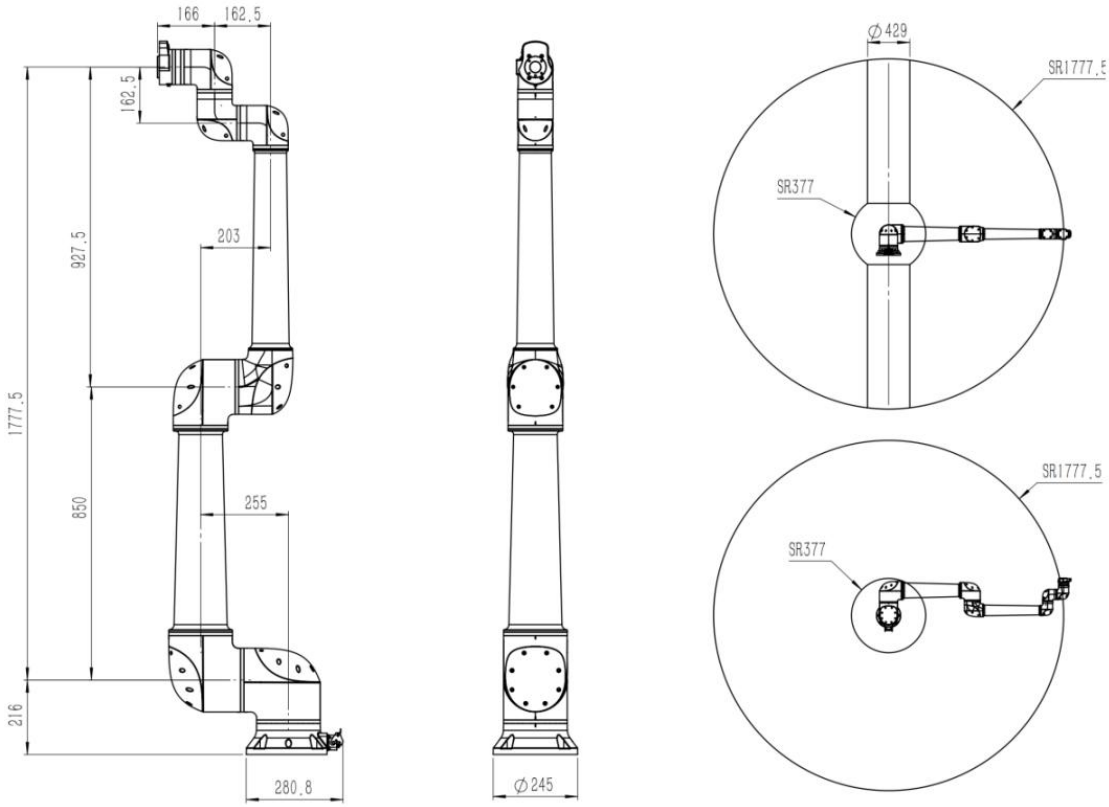


Figure 4-5 Dimensions and **working space of the S20-180**

4.3 Load curve

The maximum allowable payload of a robotic arm depends on the center of gravity shift. When the center of gravity of the load is further away, the load on the robot will be smaller. According to the eccentric distance of the load, the eccentric distance on the XY plane is taken as the vertical coordinate, and the value of Z is taken as the horizontal coordinate to find the coordinate point corresponding to the eccentric load. According to the point, observe under which curve, the load shown in the line is the maximum load that the robot can bear under the current working condition.

The total load of the tools and workpieces loaded at the end of the robot shall not exceed the maximum load.

Warnings

When calculating the load, the weight of the media flange must be taken into account and ensured to meet the load specifications of the robot. Ensure that the system never exceeds the maximum allowable load. The user should fully carry out a complete risk assessment of the media flange and the workpiece to avoid impact, vibration, crash, entanglement, puncture, puncture and other hazards. Ensure the overall safety of the system.

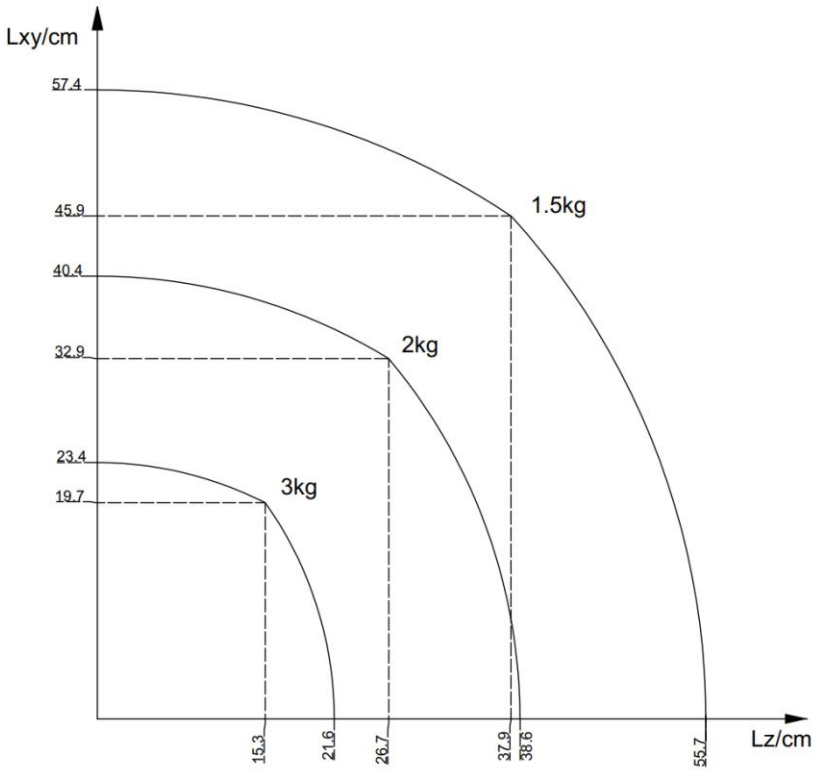


Figure 4-6 S3-60 load curve-

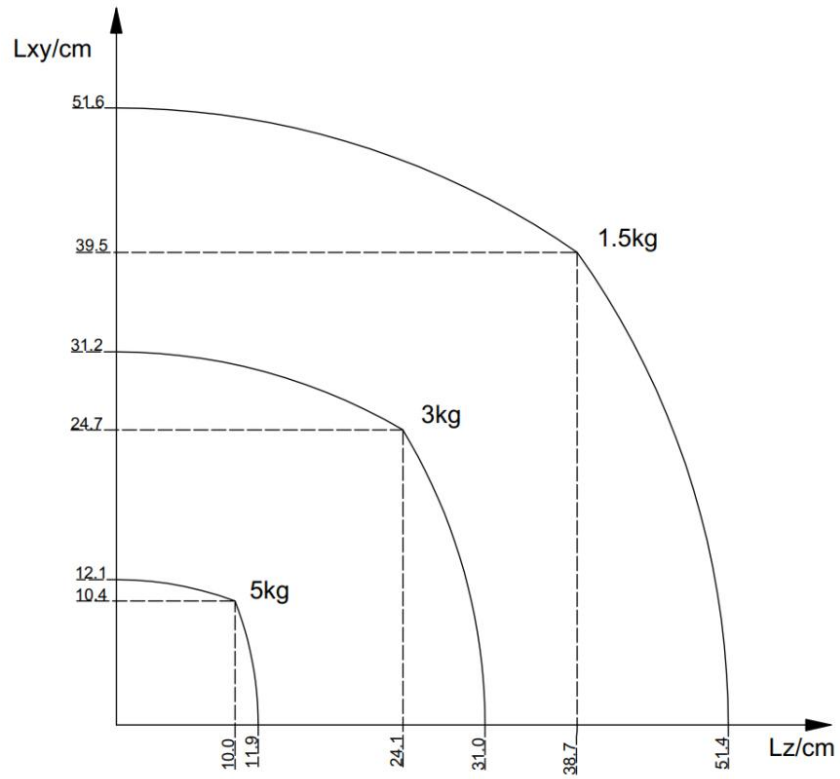


Figure 4-7 S5-90 load curve-

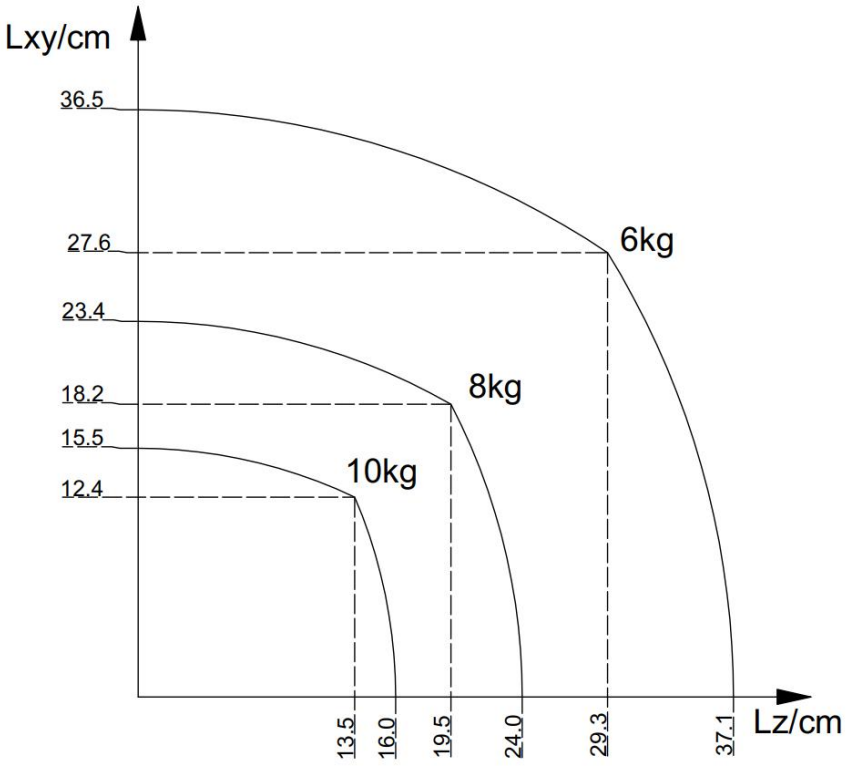


Figure 4-8 - S10-140 load curve

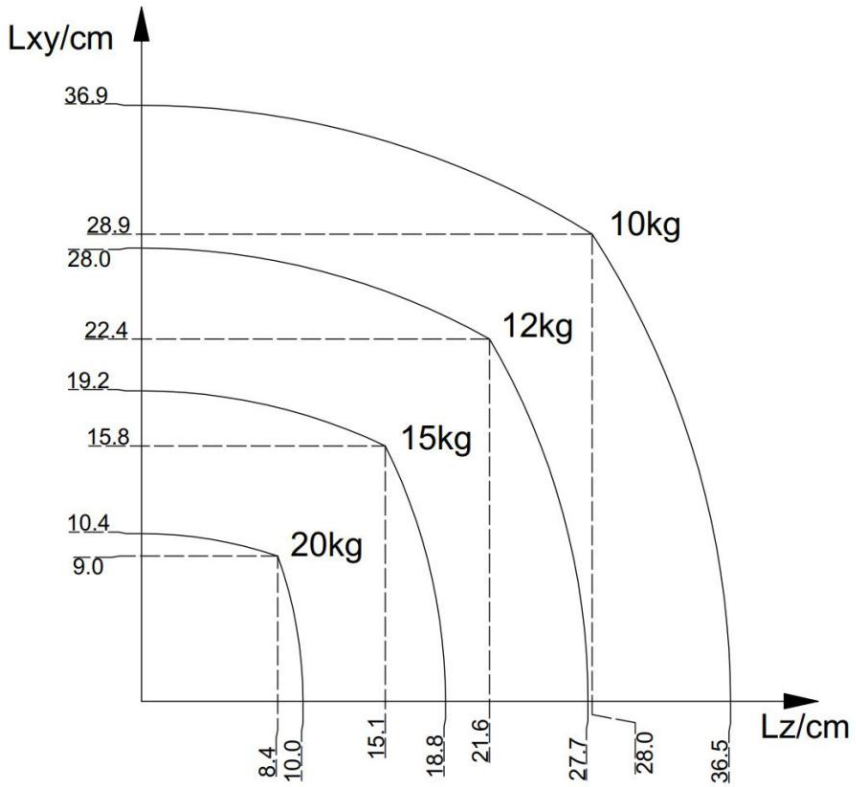


Figure 4-9 Load curve of the S20-180-

4.4 Flanged joint

The end flanges of the S-series arms are all the same size and have four M6 threaded holes for connecting tools to the robot. The flanges are designed to meet the national standard GB/T 14468.1-50-4-M6 (or ISO 9409-1-5-0-4-M6).

M6 screws must be tightened using 12Nm of torque and have a strength rating of 12.9. To accurately reposition the tool, use the dowel in the reserved Ø6 hole to maintain the exact position. Screws for mounting tools must not be inserted deeper than 8mm.

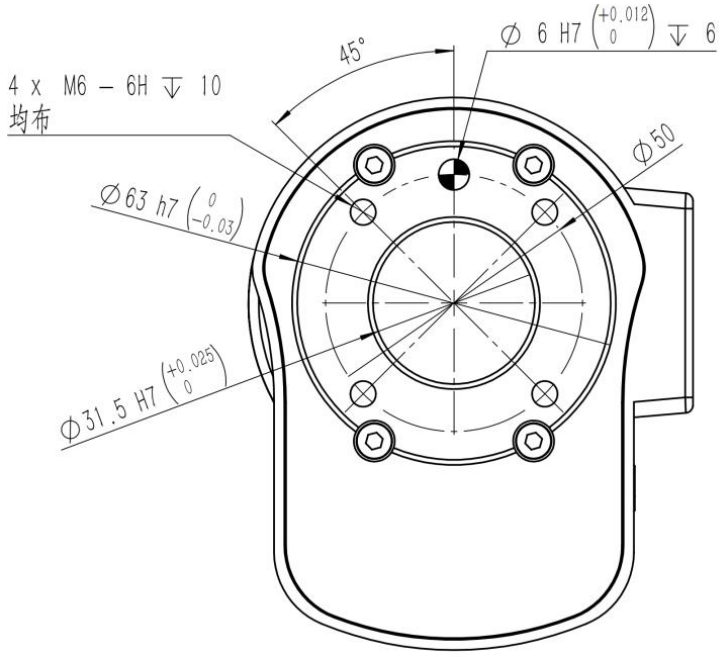


Figure 4-10 S3-60 Flange mechanical installation port

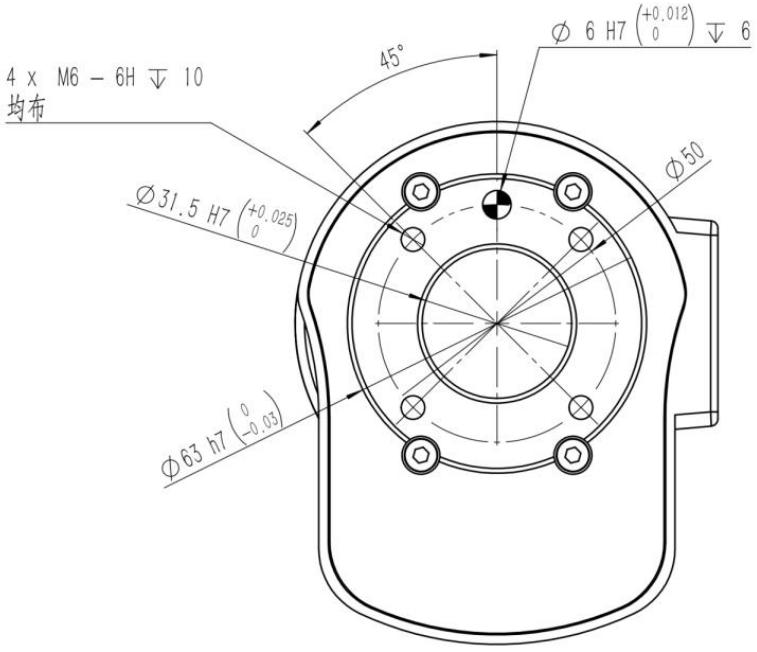


Figure 4-11 S5-90 Flanged mechanical mounting port

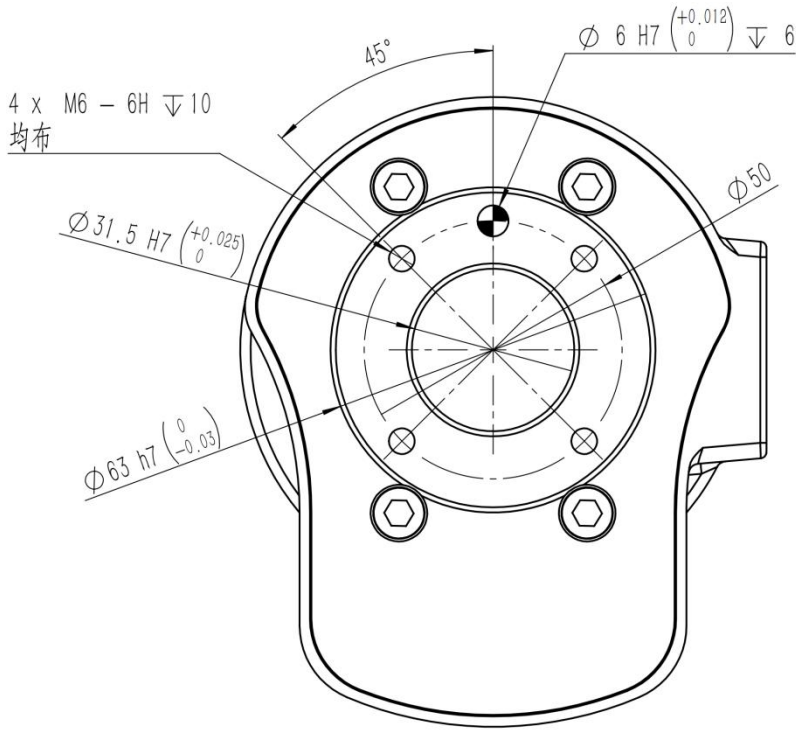


Figure 4-12 Mechanical mounting port for the - S10-140 flange

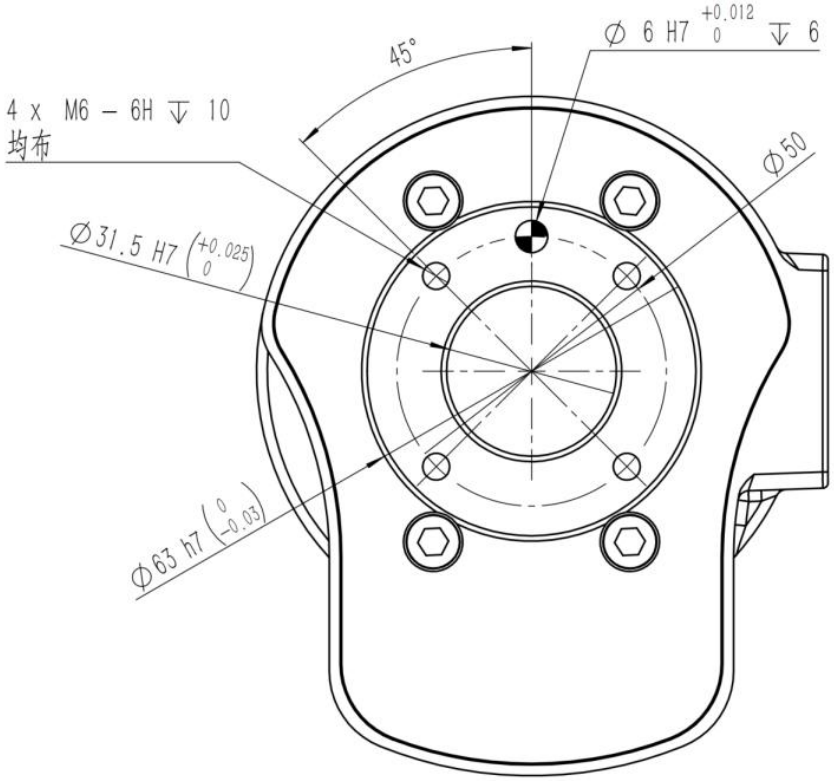


Figure 4-13 Mechanical mounting port for the S20-180 flange

4.5 Mounting interface

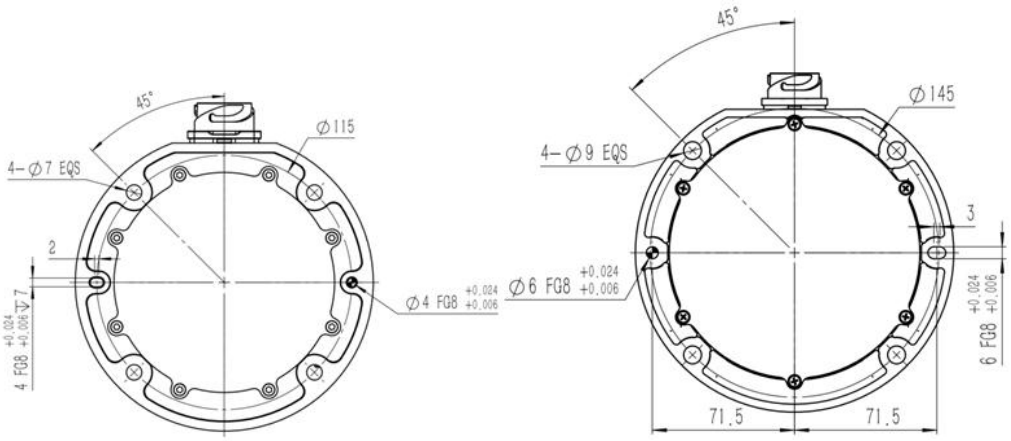


Figure 4-14 Mechanical mounting interface for the - S3-60 and S5-90 pedestals

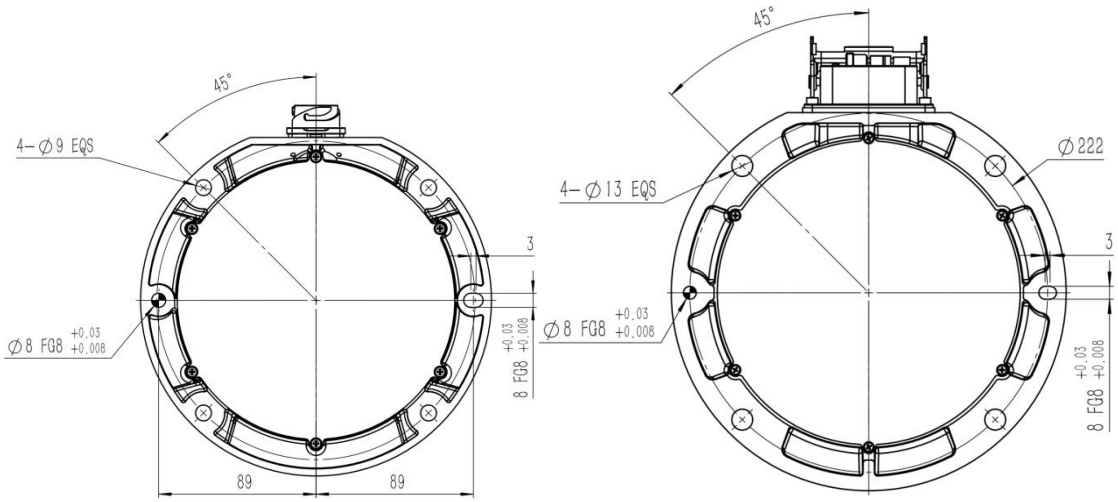


Figure 4-15 Mechanical mounting ports for the - S10-140 and S20-180 pedestals

4.6 Robot Technical Specifications

Model number	S3-60	S5-90	S10-140	S20-180
Degree of freedom	6			
Load (kg)	3	5	10	20
Arm span (mm)	575	919	1400	1777
Repeat accuracy (mm)	+ / - 0.03	+ / - 0.03	+ / - 0.05	+ / - 0.1
Body weight (kg)	18	22	38	59
Certification	EN ISO 13849-1 PLd Cat.3 & EN ISO 10218-1			
Shaft operating range	Axis 1/2/4/5/6: ±360° Axis 3: ±160°			
Maximum speed of axis	【3, 5, 10kg】 Axis 1/2/3:150 °/s Axis 4/5/6:180 °/s			
	【20kg】 Axis 1/2:110 °/s Axis 3:150 °/s Axis 4/5/6:180 °/s			
Maximum tool speed (m/s)	2	2.5	2.5	3.2
Flange interface	ISO 9409-1-50-4-M6			ISO 9409-1-50-4-M6
French Communications	2DI, 2DO, 24VDC, MODBUS RTU, RS485			
Installation method	Mounting at any Angle			
Operating temperature	0-40 ℃			
Operating humidity	70% RH			
Working noise	≤65dB			

4.7 Control cabinet

Figure 4-16 Control cabinet interface

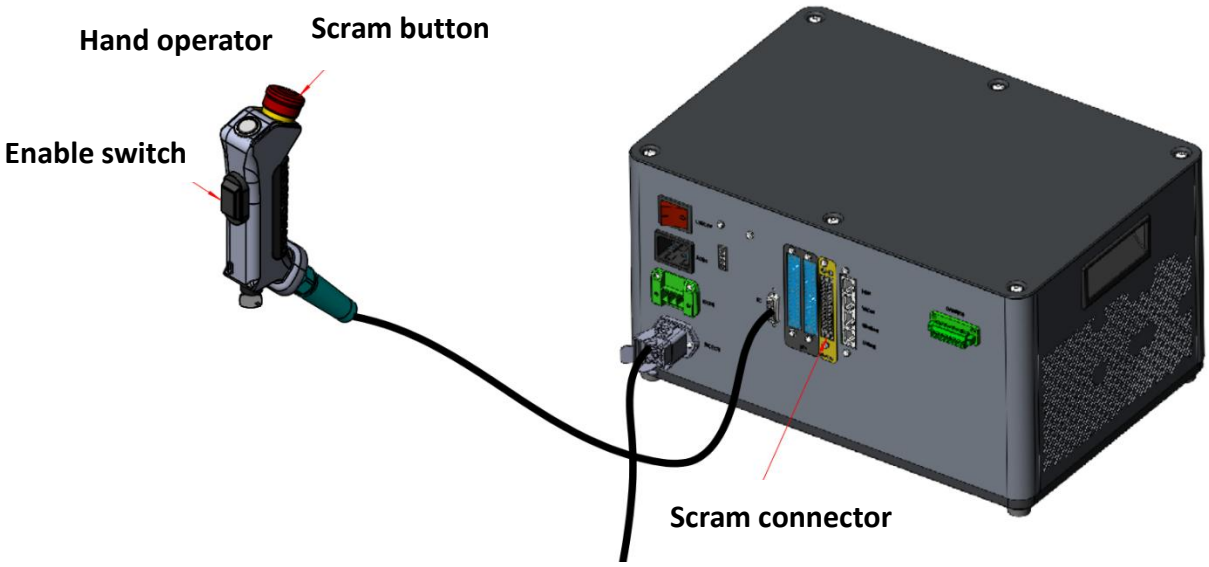


Figure 4-17 SCram and switch position diagram and safety IO interface position diagram

Safety equipment	Hand enable 1 way/Hand scram 1 way
IP level	IP20
Electric control cabinet I/O ports	16DI, 8DO, 4AI/4AO, 7 stop inputs
Electric control cabinet I/O power supply	24VDC, 2A
Operating temperature	0 ~ 40 ℃
Operating humidity	10 to 90%RH, no condensation
Operating noise	≤65dB
Altitude	Below 1000m
Power supply	AC100- 240V, 50/60Hz
Control cabinet dimensions	380mm x260mm x 200mm
Weight	14kg (20kg control cabinet), 11.8kg (10kg and below control cabinet)

4.8 Hand manipulator

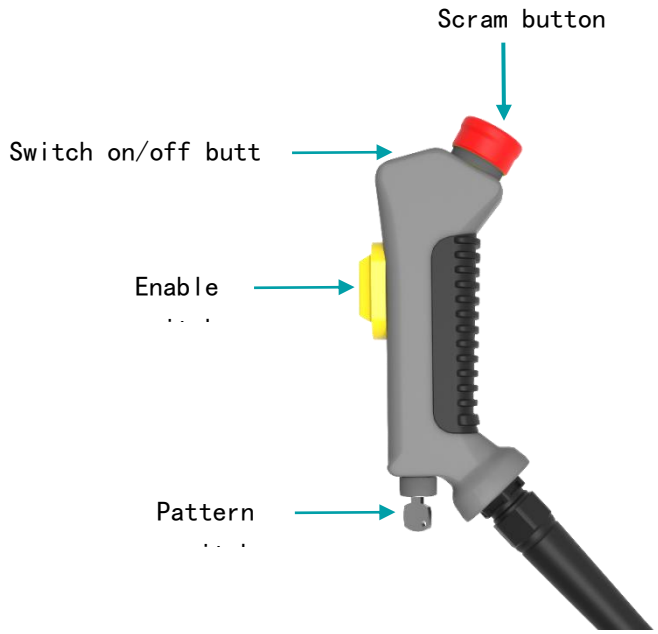


Figure 4-18 Hand operator interface-

There are four switches or buttons on the hand operator, which are the emergency stop button, the switch on and off, the enable switch and the mode switch. The position of the hand operator is shown in Figure 4-18.

In the off state, you can press the switch button to turn on the robot; In the startup state, long

press the switch on and off button to turn off the robot.

When there is an emergency, press the emergency stop switch on the hand operator, and the robot will break to stop all movement and lock up.

After pressing the emergency stop button, it will be locked, and you need to rotate the button according to the sign on the button to release the lock. After unlocking the emergency button, the alarm can be cleared using the control software. Then, enable the emergency button to recover from the emergency state.

The enable switch is a 3-stage switch, in manual mode: only when the enable switch is in the middle position, the machine can move; A Class 2 stop is triggered when the enable switch is fully released or fully pressed.

Warnings

1. Do not allow additional installation of the enable switch, which, if not avoided, could result in death or serious injury or serious damage to the equipment.
2. Do not disable the enable switch in any way which, if not avoided, could result in death or serious injury or damage to the device.
3. Do not change or modify the enable switch, which, if not avoided, could result in death or serious injury or damage to the device.
4. The enable switch takes effect only in manual mode and cannot trigger any stop function in automatic mode.

5. Electrical hardware and installation

5.1 Ontology interface

The arm base is equipped with a heavy-duty interface, the arm end is equipped with a button and indicator light, and the tool flange side is equipped with a button, screen and navigation jack. An overview of the end interface is shown in Figure 5-1-

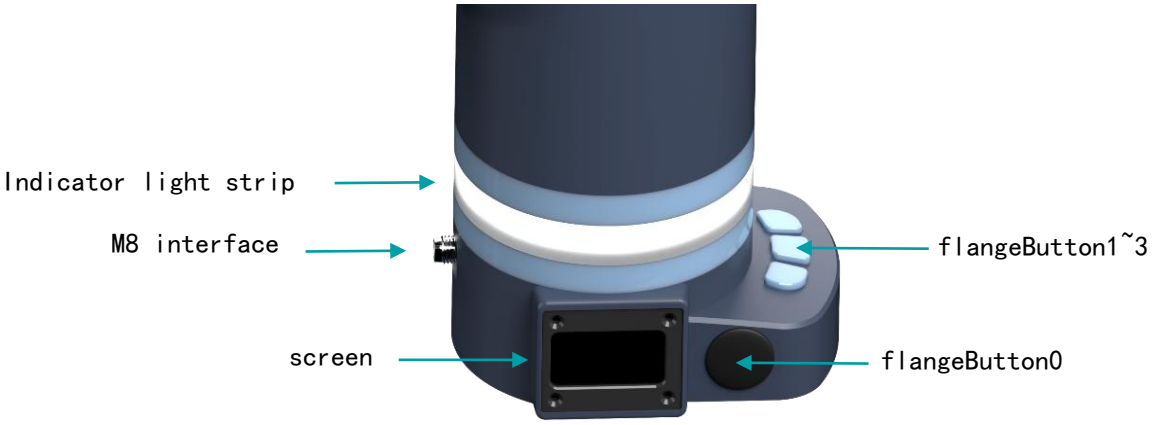


Figure 5-1 Overview of the end interface

5.1.1 Pro End Interface

Interface	Instructions
M8 interface	Power supply, input, output and communication IO
flangeButton	User function Custom button, free drag by default
flangeButton1~3	User function Custom button, can be set in the Settings interface function
Screen	Display robot status, bus communication, I/O, RS485 baud rate, user defined button status, etc
Indicator tape	Robot status indicator tape

5.1.2 Indicator tape Meaning

Strip light	Instructions
Steady blue	Initializing
white	Power on but not on
green	Manual mode

Yellow flashing	Auto run mode
Red flashing	Robot error

5.1.3 M8 interface

The flange M8 interface is located behind the flange of the robot end, and the pin distribution and definition are as follows.

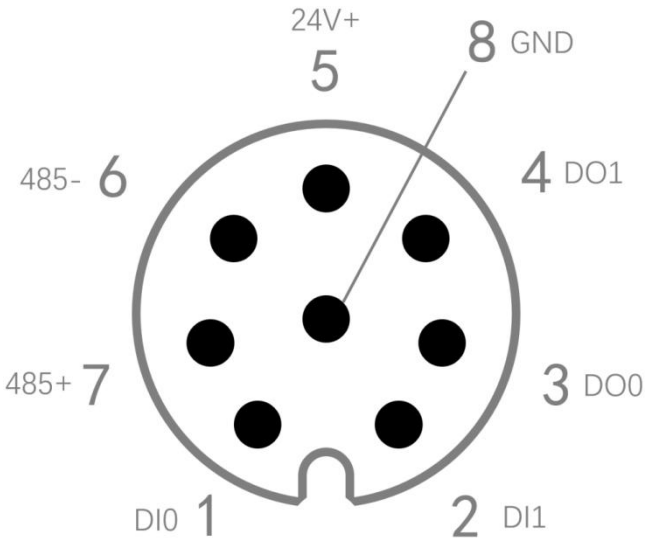


Figure 52 - Pin distribution of the M8 interface at the end of the flange

Pin serial number	name	Definition
1	DI0	Numeric input 0 (flangeDI 0)
2	DI1	Digital input 1 (flangeDI 1)
3	DO0	Digital output 0 (flangeDO 0)
4	DO1	Digital Output 1 (flangeDO 1)
5	24V+	Provide 24V power positive to the outside
6	485 -	485 - Communication for MODBUS -(A)
7	485 +	485 communication +(B) for MODBUS
8	GND	Flange inside ground; 24V power output negative

The cable model for the M8 interface is Lumberg KKMV 8-354 or Lutronic FP-222460. 24V provides a maximum 2A current externally.

The digital output is PNP type, the output can provide a maximum of 5mA current, only provides a level signal, cannot be used to drive the device.

When the digital input is PNP and the switch is used as the DI input source, the connection method is as follows.

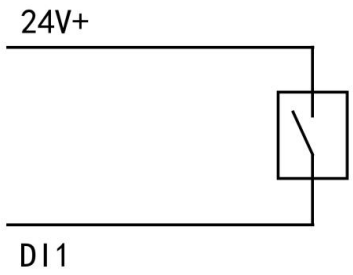


Figure 5-3 Connecting the PNP DI switch at the end of the flange

5.2 Screen Information

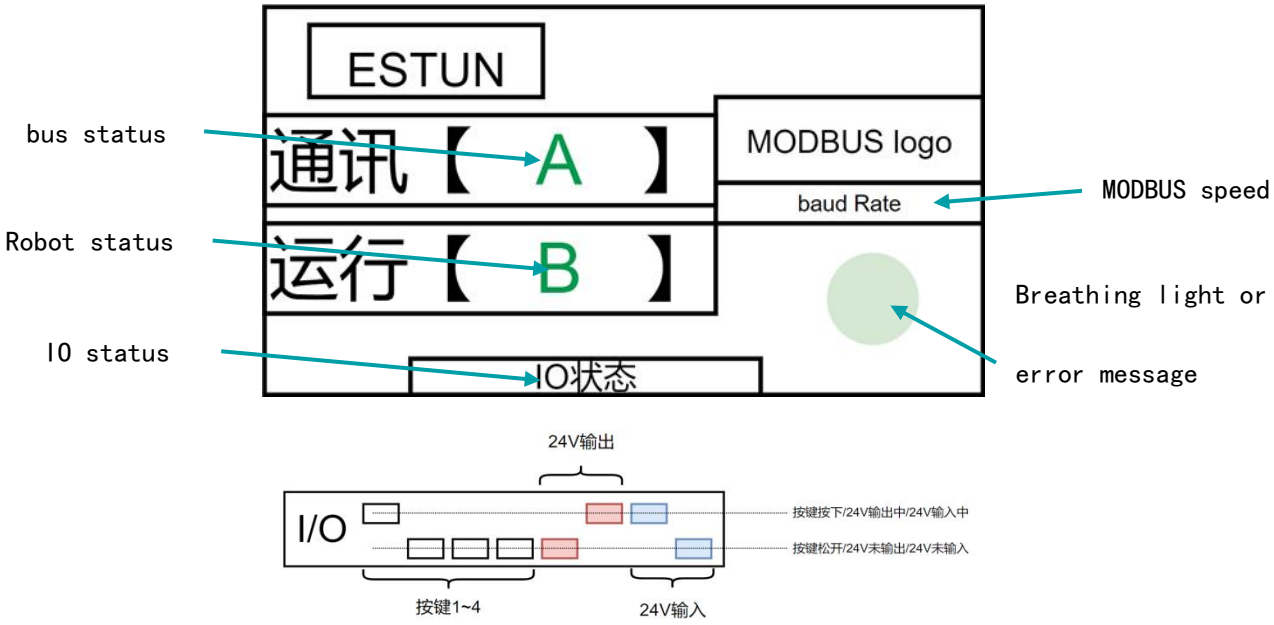


Figure 5-4 Flange end screen information

Info	Instructions
Bus status	"Offline" (red) : indicates the EtherCAT communication state machine is in INIT. This usually occurs just after startup, or somewhere in the communication cable between the controller and the flange is disconnected "Mailbox" (yellow) : indicates that the EtherCAT communication state machine is in PreOP or BOOT. Occurs when Flange firmware is updated, or when PDO is not established "Live" (green) : indicates that the EtherCAT communication state machine is in SafeOP, OP. Indicates that the communication cable is connected properly and the communication with the controller is normal "Error" (red) : Indicates that the EtherCAT communication state machine is accidentally switching from OP to INIT. Usually occurs in the cable break in the communication, suddenly loses power controller or soft reset
Robot state	"Normal" (green) : The robot has no errors. Error (red) : An error occurs when the robot is running.
Terminal IO status	If the cursor is high, the corresponding item is active; Low indicates that the corresponding item is not active
MODBUS Rate	The MODBUS baud rate includes 115200, 57600, 28400, 19200, 9600, 4800, 2400, 1200, and 600. The Modbus baud rate can be set by parameter
Breath light or error message	Green slow blinking (2s) : Communication not fully established (INIT, BOOT, PreOP) Blinking green fast (0.5s) : Real-time communication is connected (SafeOP, OP) Blinking red (2S) : Communication is disconnected. OP->INIT Error message: The robot reports an error, displaying an error alert

5.3 Control cabinet interface

Control cabinet positive only a power button, the robot system, which can be opened, long shutdown condition according to the boot state, long press can close the robot system.

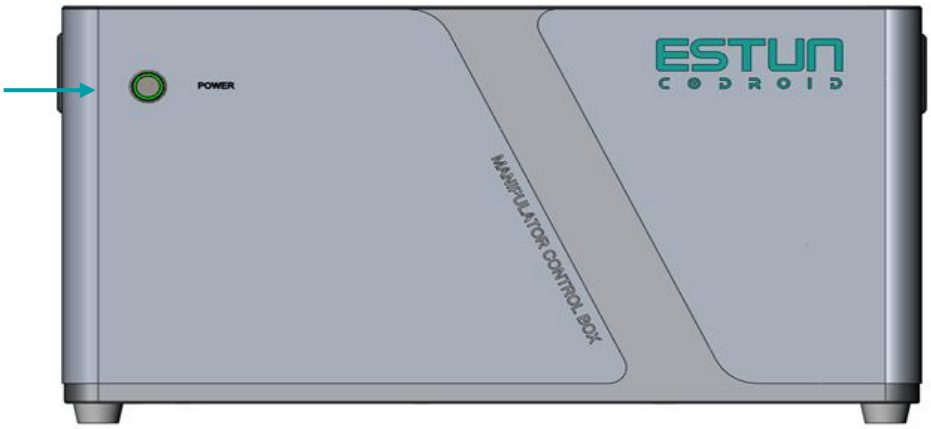


Figure 5-5 Power button for the control cabinet

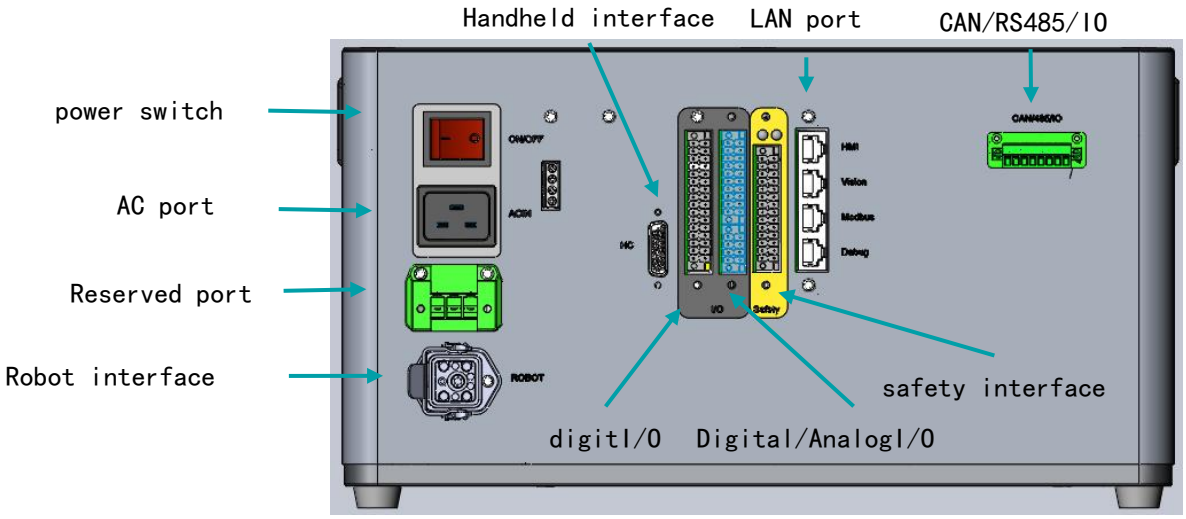


Figure 5-6 Overview of ports on the control cabinet

5.3.1 Electrical interface overview

Interfaces	Instructions
Ac power supply interface	For connecting AC100-240V 50/60Hz AC power supply
Power switch	Power supply power switch
Hand operator interface	Hand operator interface
Robot navigation jack	It is used to connect the robot and the control cabinet to provide power supply and communication for the robot
LAN network port	Used to connect demonstrator, vision, bus, and development debugging
CAN/485/IO interface	CAN/RS485/IO interface
Security interface	Security function related interfaces
Analog/digital I/O	Analog input/output interface; Digital input/output interface
Digital I/O	Digital input/output interface

5.3.2 Security interface

The security interfaces are seven groups of security dual-channel interfaces, groups 1 to 3 are protective stop interfaces, and groups 4 to 7 are emergency stop interfaces. The internal safety relay of the protective stop interface and the emergency stop interface is two independent channels. The factory muzzle connector uses the yellow short wire for lateral short connection, otherwise it can not release the emergency stop state.

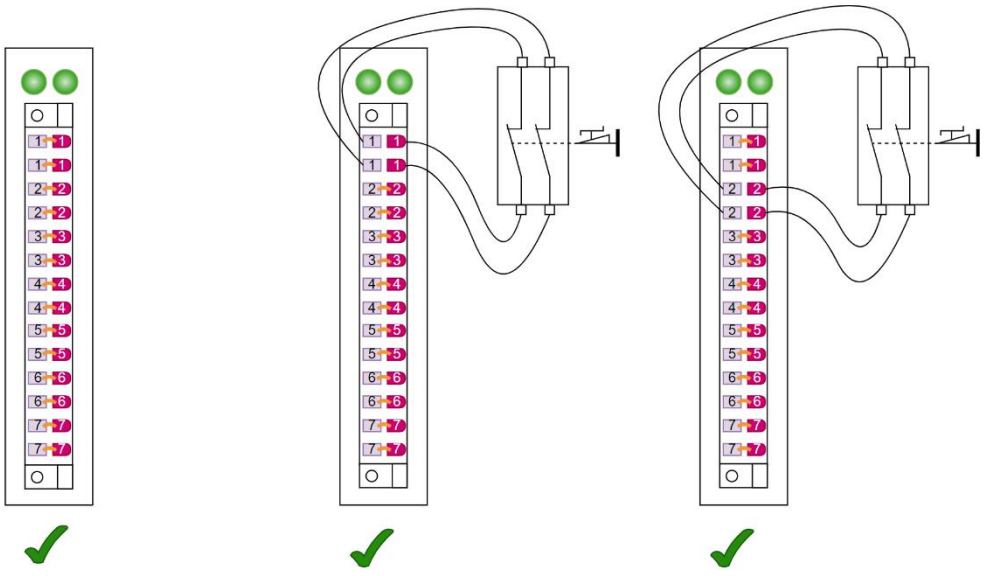


Figure - 5-7-1 Example of a correct cable to be connected to a protective stop

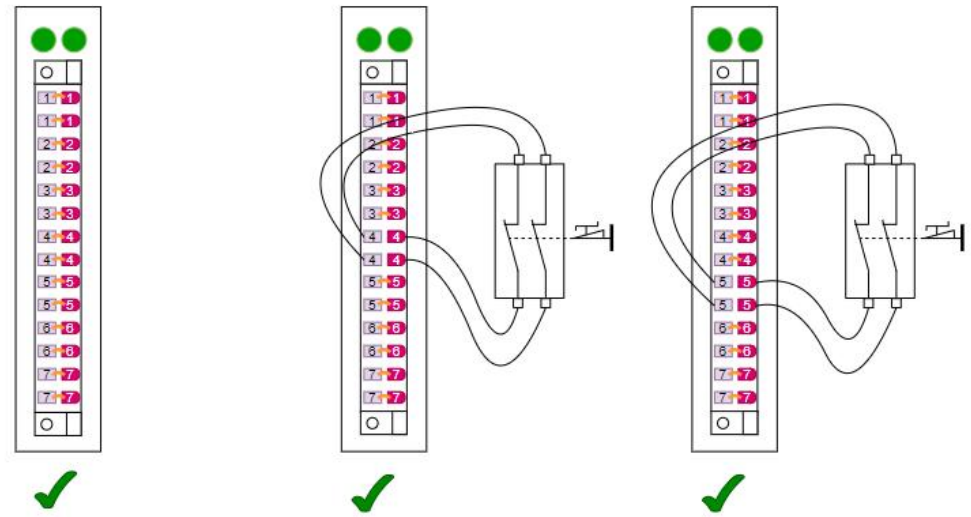


Figure 5-7-2 Safety scram cable connection-

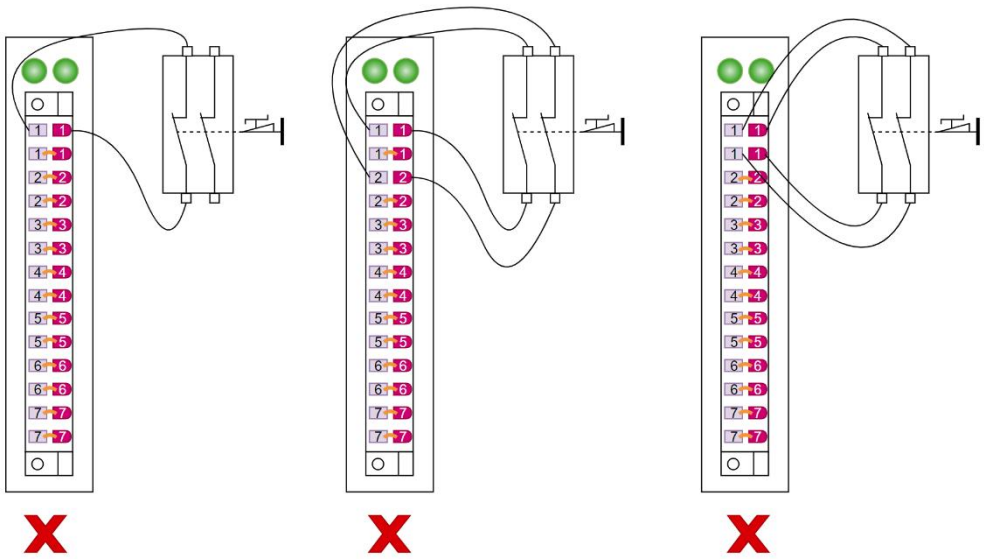


Figure 5-8 Typical examples of incorrect connections to the safety protective stop and safety scram

5.3.3 General I/O Overview

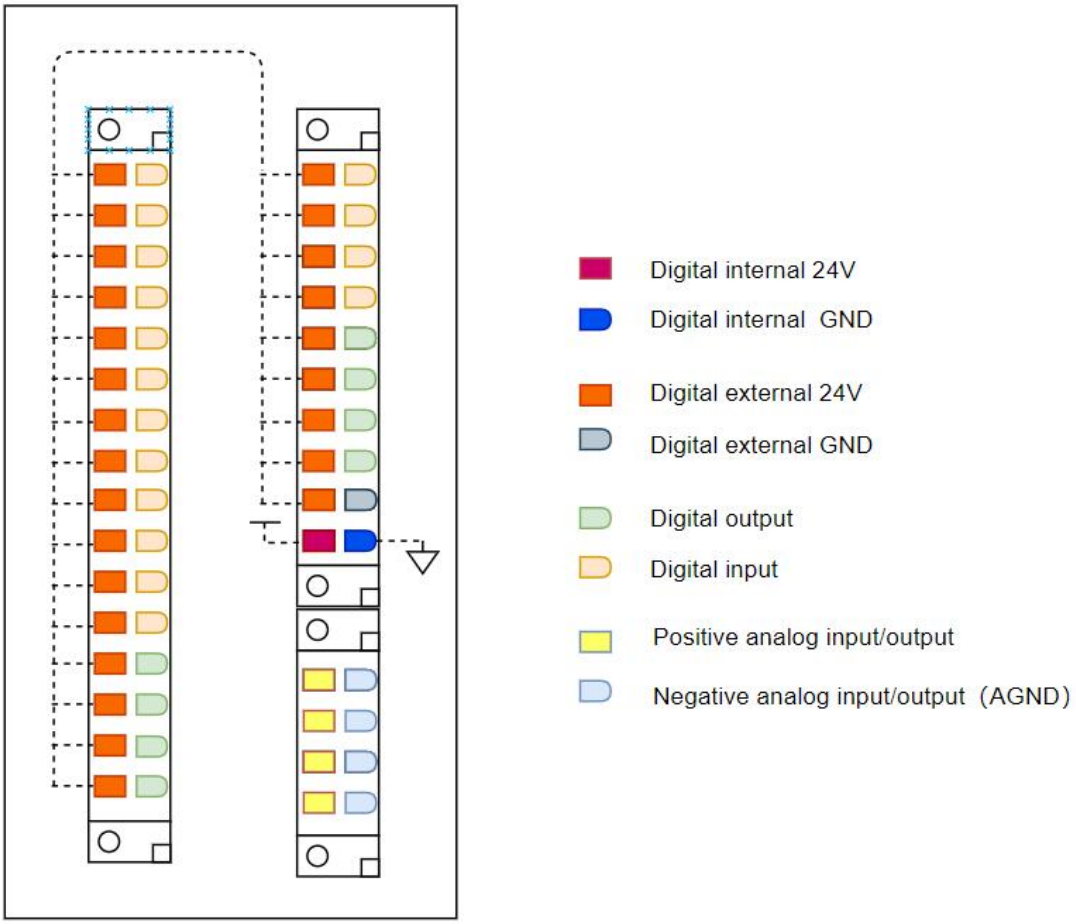


Figure 5-9 Functions of the individual terminals of the input/output module

5.3.4 Digital input external power supply wiring

When digital input, the port should be powered by an external power supply, and the input side should be connected by a relay or PNP type digital loop. The wiring method is referred to as follows.

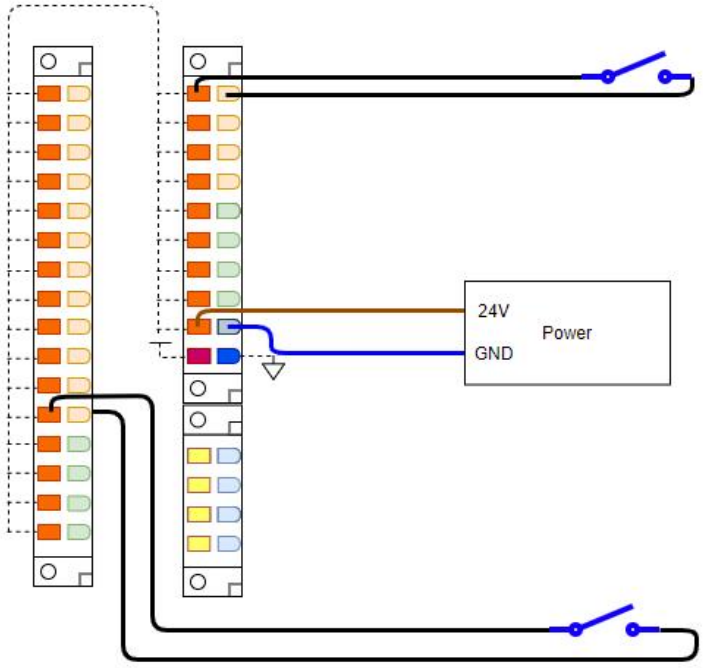


Figure 5-10 Example of connection using relay digital input

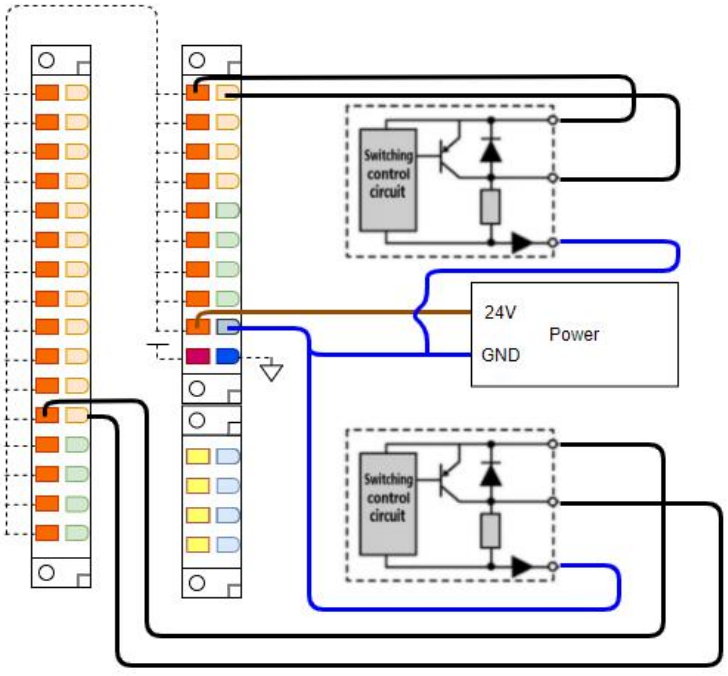


Figure 5-11 Example of digital input wiring using PNP type

5.3.5 Digital input internal power wiring method

When the digital input power supply to the port, you can use the internal power supply of the base board to power the port, and the input end is connected using a relay or PNP type digital loop. The wiring method is referred to as follows:

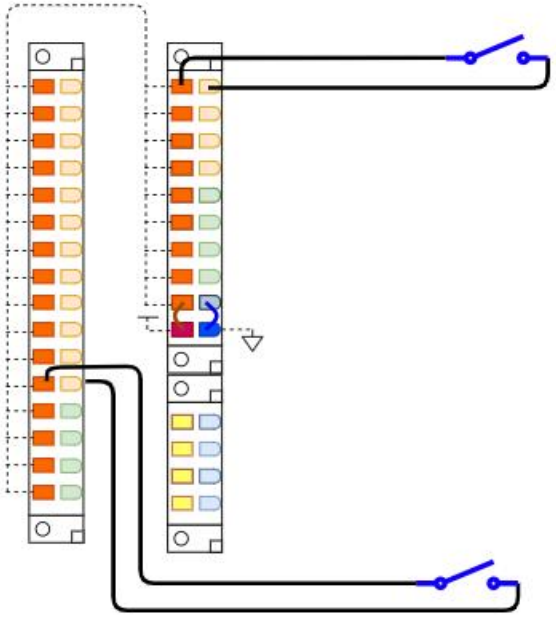


Figure 5-12 example using relay digital input wiring-

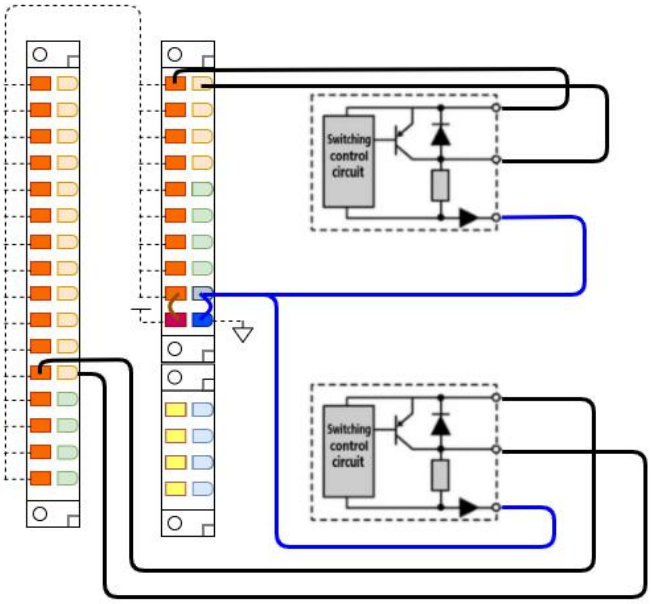


Figure 5-13 Example of typical wiring for external power supply

Digital input for external power supply	argument
External supply voltage	Typical 24V
Maximum output current	Single group maximum output 5A

5.3.6 Digital output external power wiring method

The digital output to the device end is powered by an external power supply, and the output end is connected by a relay or PNP type circuit. The wiring method is as follows:

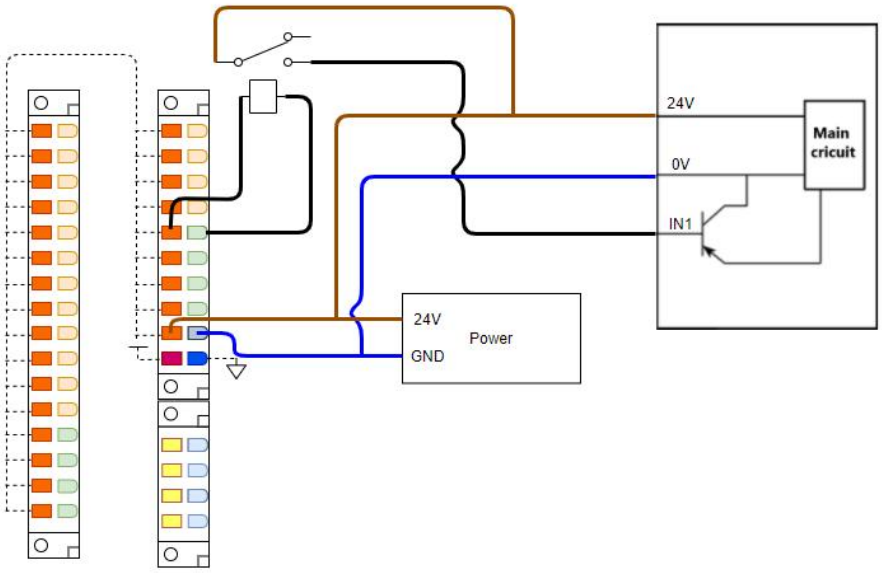


Figure 5-14 Example of relay wiring for digital output

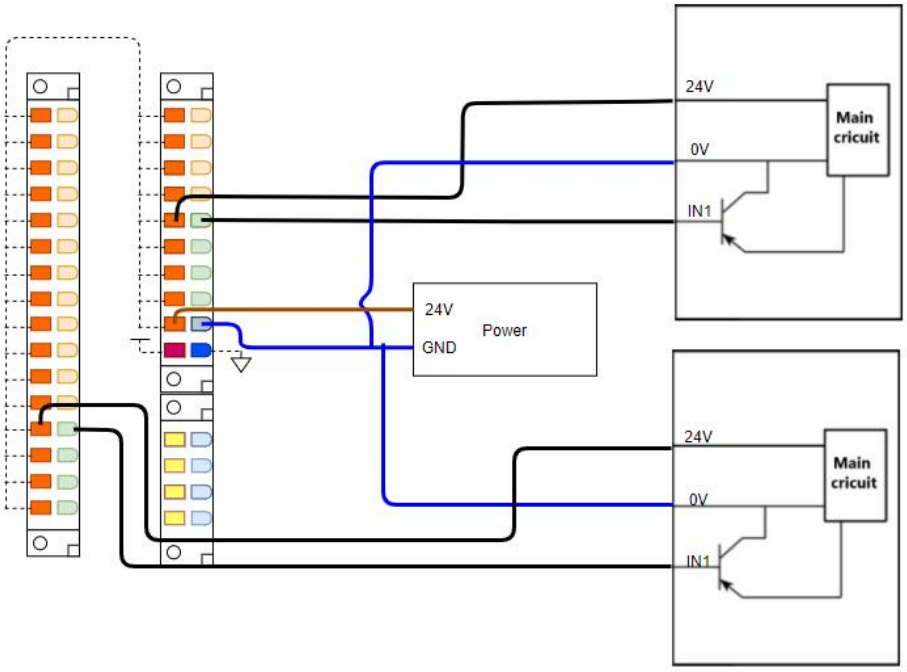


Figure 5-15 Digital output using PNP type circuit wiring example

5.3.7 Digital output internal power wiring

The digital output to the device end is powered by an external power supply, and the output end is connected by a relay or PNP type circuit. The wiring method is as follows:

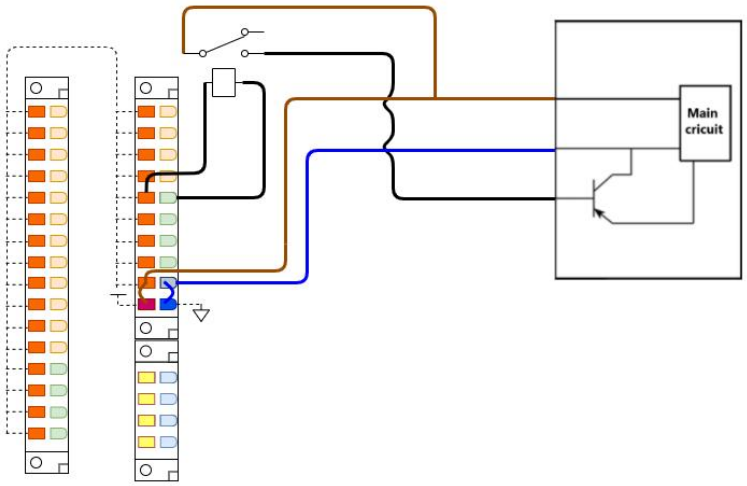


Figure 5-16 Example of relay wiring for digital output

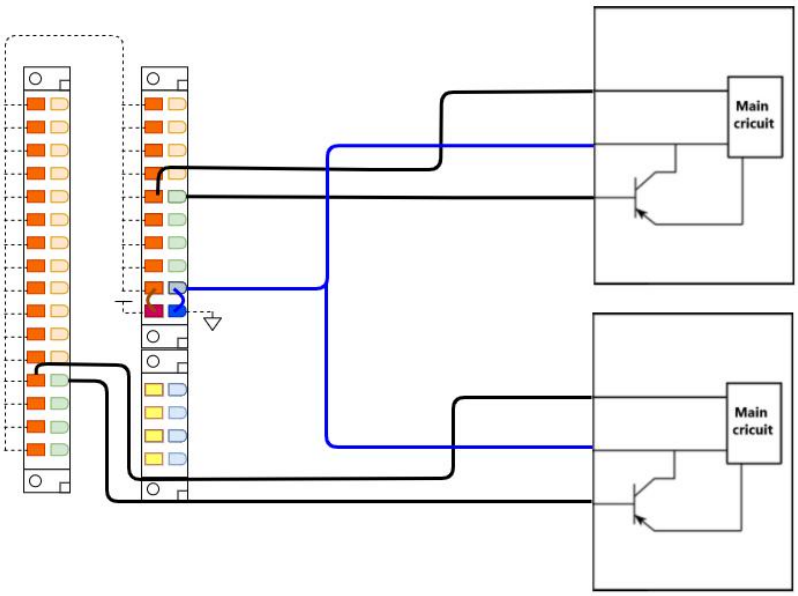


Figure 5-17 Example of PNP type circuit wiring for digital output-

Digital Output item	**Parameters**
Interface type	PNP
Output voltage	Typical 24V; 30V Max
Maximum output current	Maximum 125mA for a single set

5.3.8 Analog input/output interface

Analog input support voltage type and current type sensors, voltage or current input needs to be set in the robot Settings option; Analog outputs only support current type.

Warnings

The analog output port must be loaded, otherwise the robot will report an error. You can close the corresponding analog output port from the robot operation interface must be connected to the load, otherwise the robot will report an error. The corresponding end can be closed from the robot operation interface

The wiring for various situations is shown in the following figure:

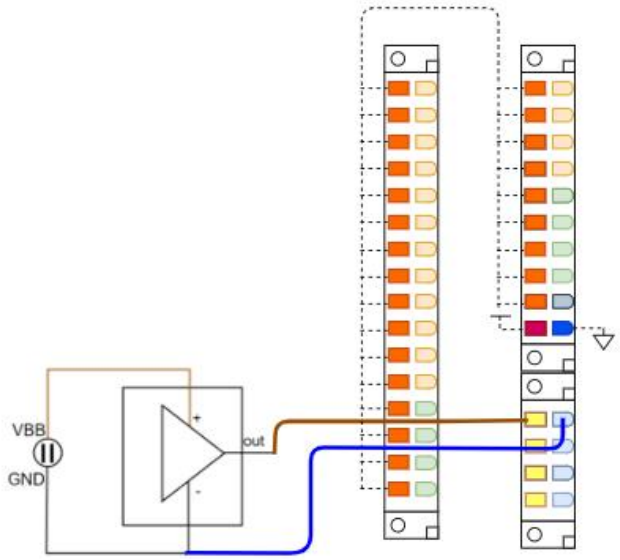


Figure 5-18 Example of correct wiring for analog voltage type input

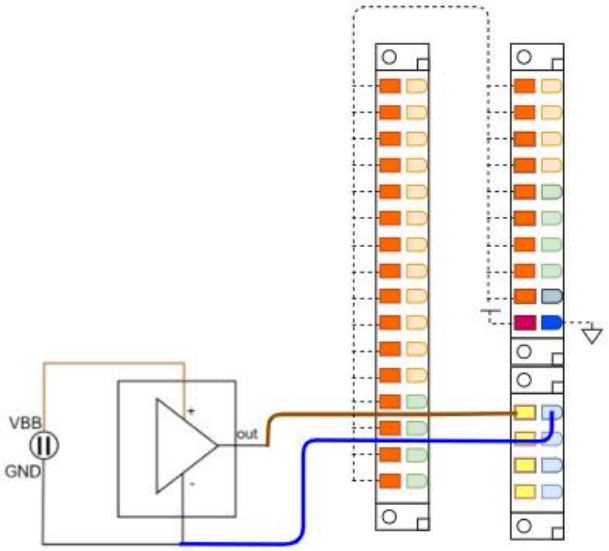


Figure 5-19 Example of correct wiring for analog current type input

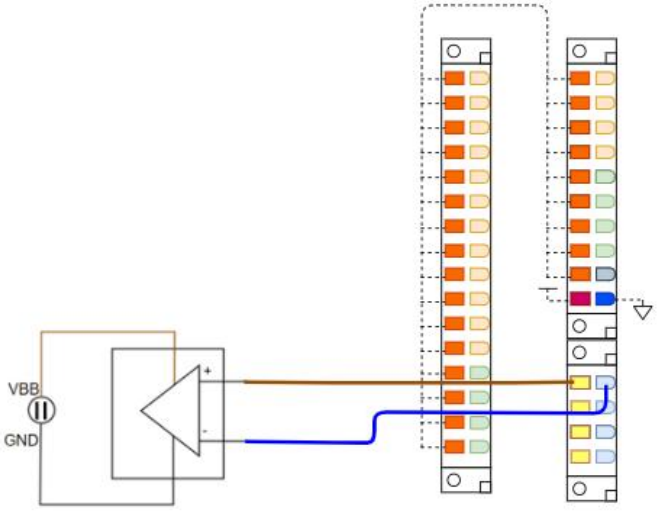


Figure 5-20 Example of correct wiring of analog current type output

Analog input	**Parameters**
Resolution	12bit
Measuring range	In voltage mode: 0-10V; Current mode: 4-20mA;
Input impedance	Current mode: 20Ω;

Analog output item	**Parameters**
Resolution	12bit
Output range	Current mode: 4-20mA;

5.3.9 CAN/485/IO interface

The interface pins on the control cabinet containing CAN, 485 and IO are defined as follows:

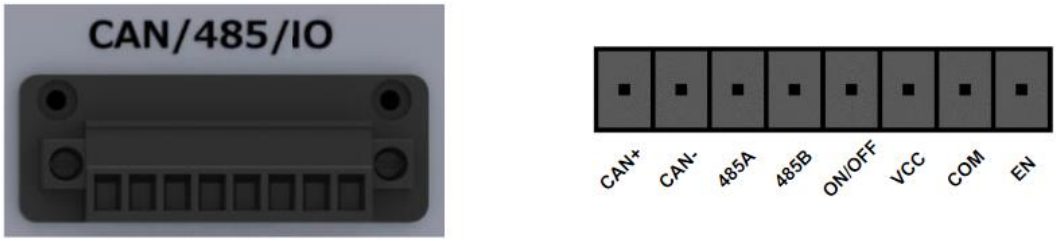


Figure 521 CAN/485/IO interface definition-

Interface	Instructions
CAN+	CAN+
CAN-	CAN-
485A	485A/485+
485B	485B/485-
ON/OFF	External start/stop button interface
VCC	Start stop signaling interface
COM	Start stop signal receiving interface
EN	Internal start-stop button interface

Different wiring methods for power start and stop:

Method ① : Use the power button on the control cabinet and the hand operator to switch the machine on and off

(Use short wiring to short-connect COM and EN interfaces)

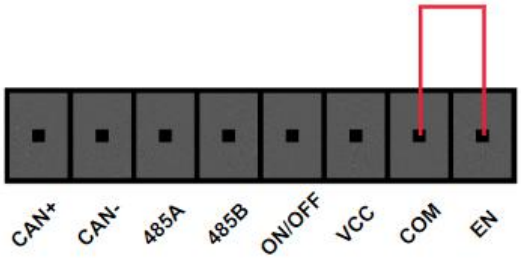


Figure 5-22 Power starting method ① Wiring diagram-

Mode 2: External power supply start and stop button

(Use short wiring to short-circuit COM and EN interfaces, and ON/OFF and VCC interfaces to external self-recovery normal start and stop buttons)

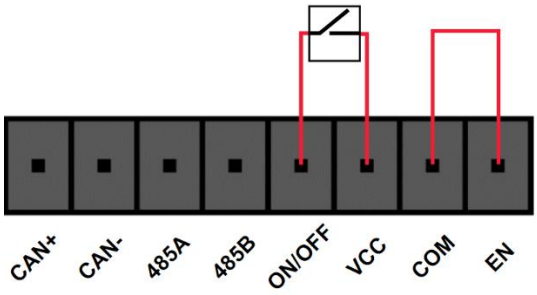


Figure 5-23 Power startup mode ② Wiring diagram

Mode 3: The control cabinet starts after the power is powered on

(Use short wiring to short-circuit VCC and 24VEN two interfaces)

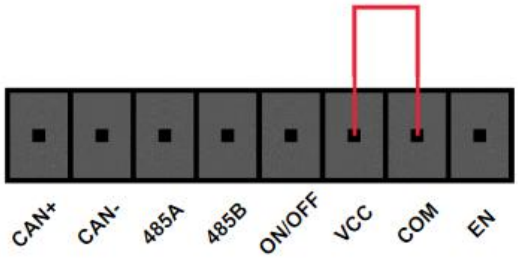


Figure 5-24 Starting mode of power supply ③ Wiring diagram

5.3.10 LAN Network port

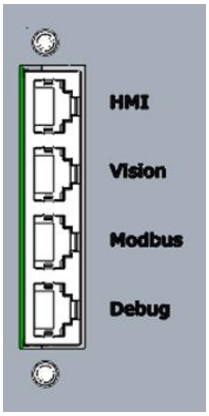


Figure 5-25 Network interface

LAN Network port entry	Instructions
HMI	Connect the demonstrator or tablet. Connect the internal router directly, and the router connects to port ETH0 of the keba.
Vision	Connect the vision. Connect directly to the internal router, which connects to port ETH0 of keba.
Modbus	Bus connection port. Connect directly to the internal router, which connects to port ETH0 of the keba.
Debug	Debug, socket port. Connect the internal keba controller to port ETH1 directly.

5.3.11 Ac input

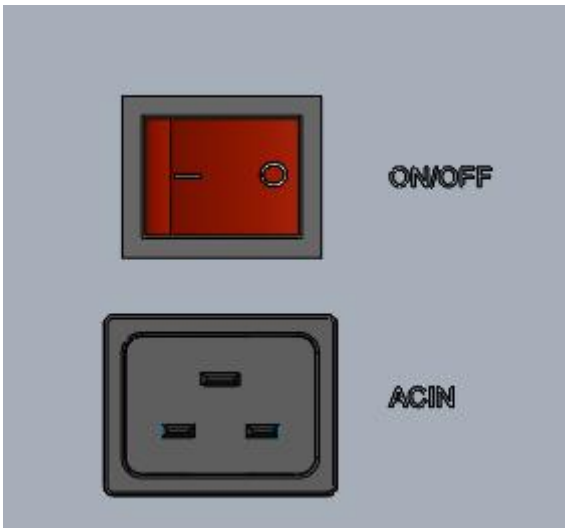


Figure 5-26 AC input interface and switch

Ac input range: AC100~240 V 50/60 Hz. Ac power supply and DC power supply cannot be supplied at the same time. A magnetic ring needs to be inserted into the ACIN power cable to eliminate EMC interference.

6. Maintenance and warranty

6.1 Precautions

- Repair work can only be done by Kuzhuo or an authorized system integrator.
- Maintenance or repair be sure to perform any visual or work environment checks according to all safety instructions in this manual.
- Changes to the control system, robot joints, need to re-calibrate the robot, calibration operation and result judgment methods are in the zero calibration manual. And need to check the parameter Settings, if there is a parameter backup, you can import the backup parameters. If there is no backup, you need to reset the parameters.

The following safety tasks must be followed when operating the robot body or control cabinet:

- Remove the main input cable from the back of the control cabinet to ensure it is completely powered off. Necessary precautions need to be taken to avoid anyone else re-energizing the system during maintenance. Recheck the system even after power outage to make sure it is powered off
- Check the ground connection before turning the system back on.
- Comply with ESD(electrostatic discharge) regulations when disassembling the robot body or control cabinet
- Avoid disassembling the power supply system of the control cabinet. The power supply system of the control cabinet can retain high voltage for several hours after the control cabinet is closed.
- Avoid water or dust entering the robot body or the control cabinet

6.2 Routine inspection items

6.2.1 General cleaning

If dust/dirt/oil is observed on the controller or robotic arm, it can be wiped clean with a rag dampened with cleaner. Cleanser: water, isopropyl alcohol, 10% ethanol, or 10% naphtha.

In very rare cases, a small amount of grease may be seen at the joint. This does not affect the specified function or service life of the joint.

Note: Do not use compressed air to clean the controller or robotic arm, otherwise, the seal and internal components may be damaged.

6.2.2 Control Box

Inspection plan

Inspection items	Inspection method	Once a month	Semiannual	Once a year
Hand operator scram button	Function check	X		
Free Drive mode	Functional check		X	
Safety I/O	Functional test	X		
Demonstrator cables and adapters	Visual inspection		X	
Terminals on the control box	Functional inspection		X	
Control cabinet main power and switch	Functional test			X

Highlight the safety functions of the robot, and recommend monthly testing to ensure normal function.

The following tests must be performed:

6.2.2.1 Test the scram button on the hand operator

- Press the scram button;
- Observe the robot to stop and turn off the joint power;
- Start the robot again.

6.2.2.2 Test the free-drag mode

- Remove attachments or set TCP/ payload according to tool specifications;
- Press and hold the free drag button at the end of the robot to set the robot to free drag mode;
- Move the robot to a position that extends horizontally to the edge of its workspace;
- While holding down the free-drag button, monitor the robot to hold its position without support.

6.2.2.3 Test safety inputs and outputs

- Check which security inputs and security outputs are active and test whether they can be triggered.

6.2.2.4 Visual inspection

- Disconnect the power cord from the controller;
- Check that the terminals are inserted correctly and the wires are not loose;
- Check whether the network cable inside the controller is loose;
- Check the inside of the controller for dirt/dust and, if necessary, clean it with a vacuum cleaner that prevents electrostatic discharge.

6.2.3 Robotic arm

Inspection plan

Inspection items	**Inspection method**	**Once a month**	**Semiannual**	**Once a year**
Check the joint cap	Visual inspection		**X**	
Check the screws on the lid	Functional inspection		**X**	
Check flat rings	Visual inspection		**X**	
Check robot cables and connections	Visual inspection		**X**	
Check the arm mounting bolts	Functional inspection	**X**		
Check tool mounting bolts	Functional inspection	**X**		
Check the screws that connect the joints	Functional test		**X**	

The purpose of functional inspection is to ensure that screws, bolts, tools and robotic arms are not loose. The screws/bolts mentioned in the inspection plan should be checked using a torque wrench.

6.3 System Updates

This section describes how to update the CoDroid robot software. The content of this manual is true and valid at the time it was written. The user will not be notified of future product updates in advance.

Before starting the update, please confirm the following update precautions

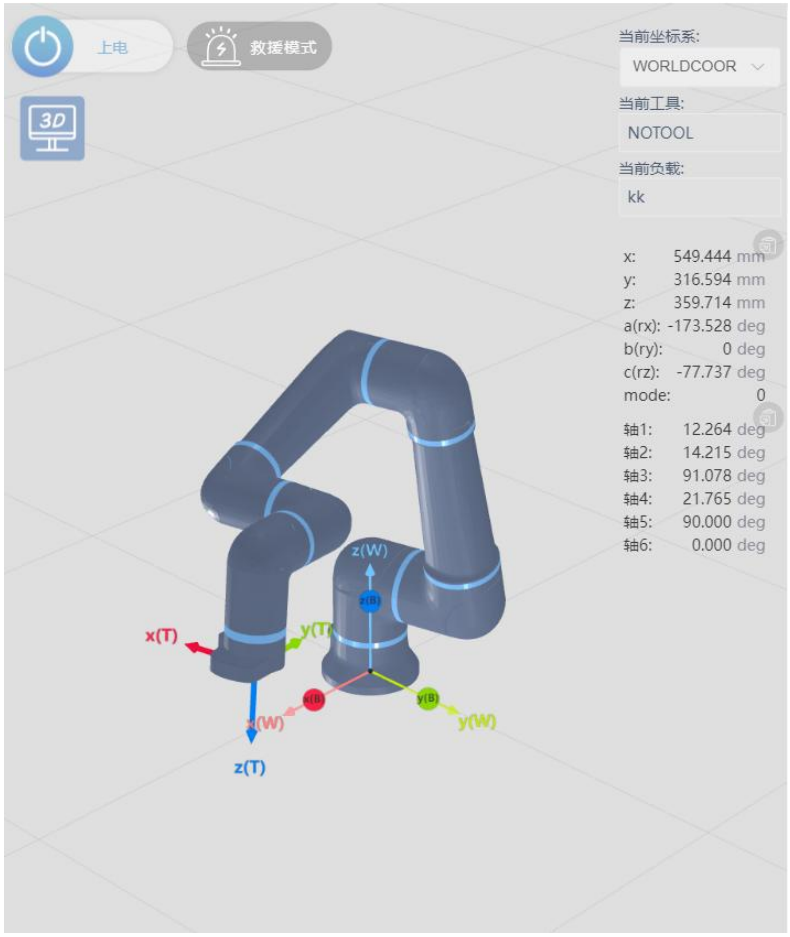
- Be careful to ensure that the power supply is not turned off or disconnected during the update
- Make sure you get the correct version of the compressed update file

- All robot programs have been backed up
- Before updating, check the release notes for the version you want to update to. Contact a CoDroid technician for details

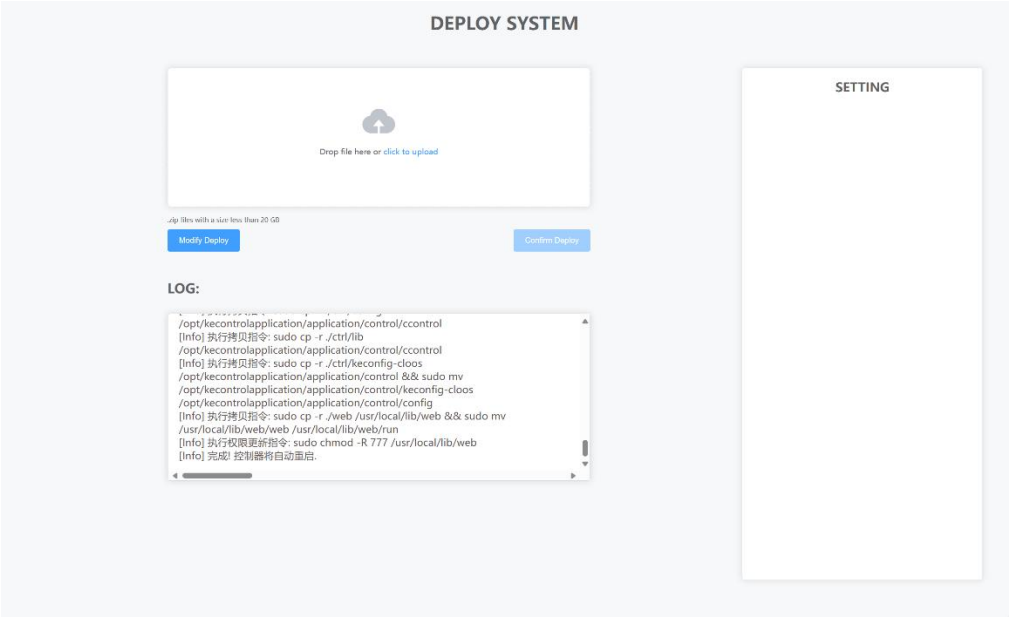
6.3.1 Update Steps

1. Enter Estoncoolo control platform after boot, enter the project TAB, click the project management interface, select the program you want to back up to download for program backup.

2. Switch the robot to the "Power off" state and press the emergency stop.

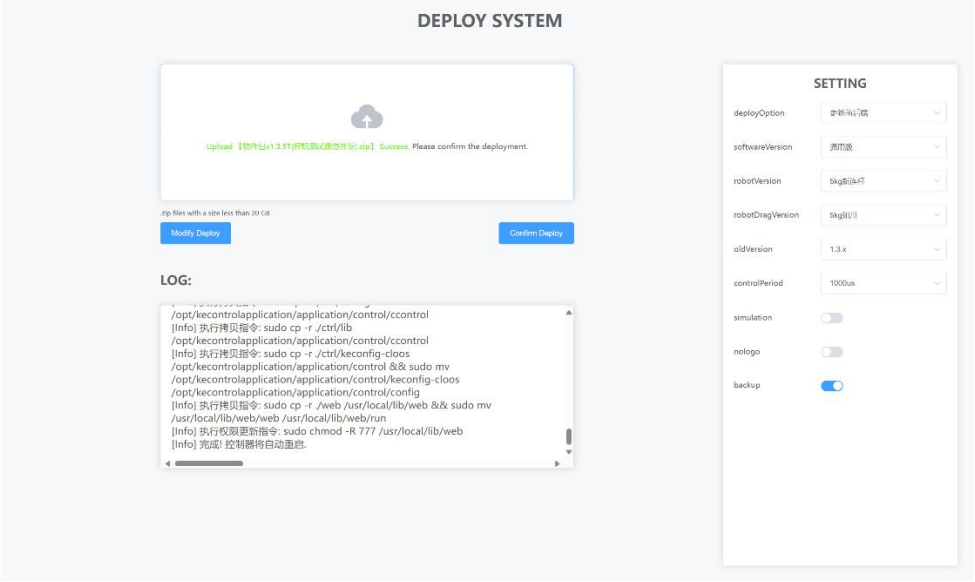


3. Add a new TAB in your browser and enter the robot IP address and update port number: **192.168.101.100:8080** to go to the update screen.

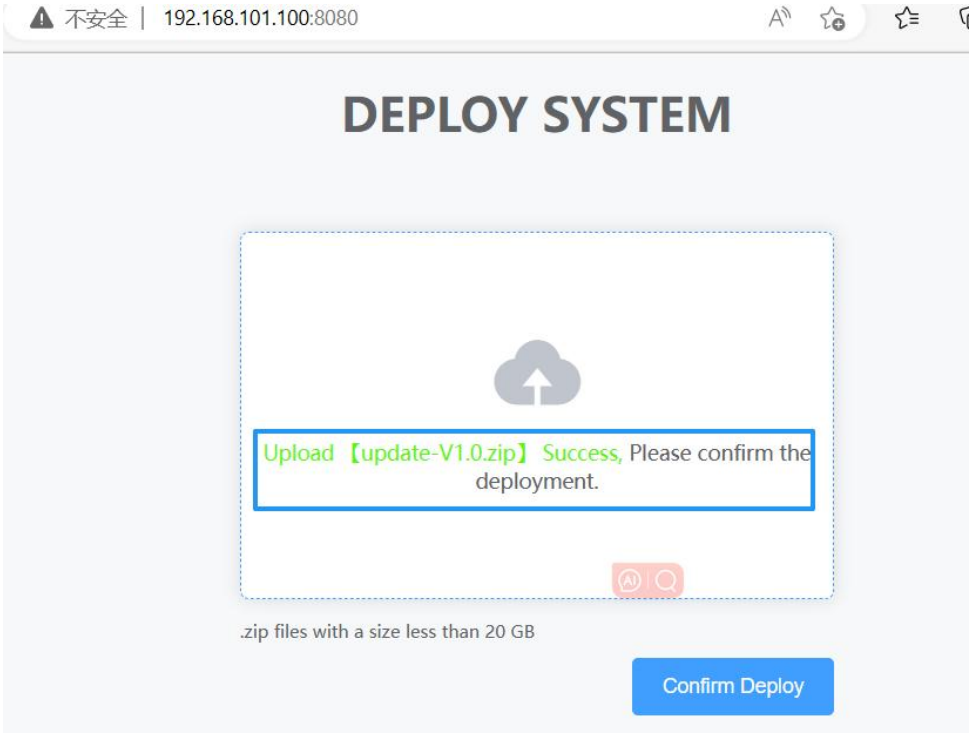


4. Drag the update file into the file selection box, or click the 'click to upload' button to select the file you want to update and wait for the upload to complete.

5. Select the appropriate option based on your model needs.



6. After confirming the update, wait for the robot software to automatically restart, and the update will be completed after the restart is completed.



6.4 Common Errors

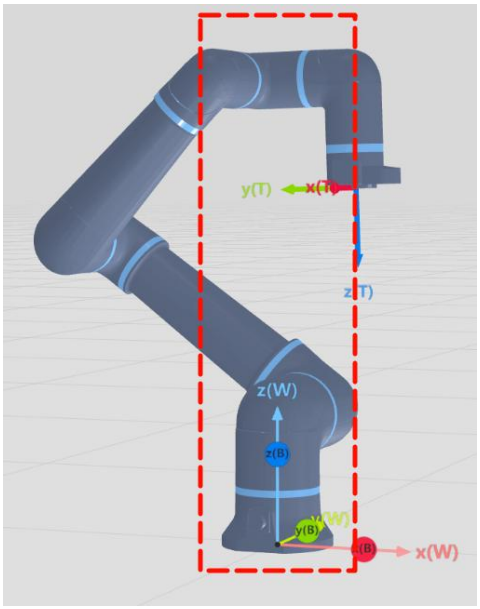
This section lists some common errors that may occur during the use of the robot. If other errors cannot be resolved, you can download the robot log file on the log interface and send the file to the after-sales personnel for analysis and processing.

6.4.1 Singularities/inverse solutions fail

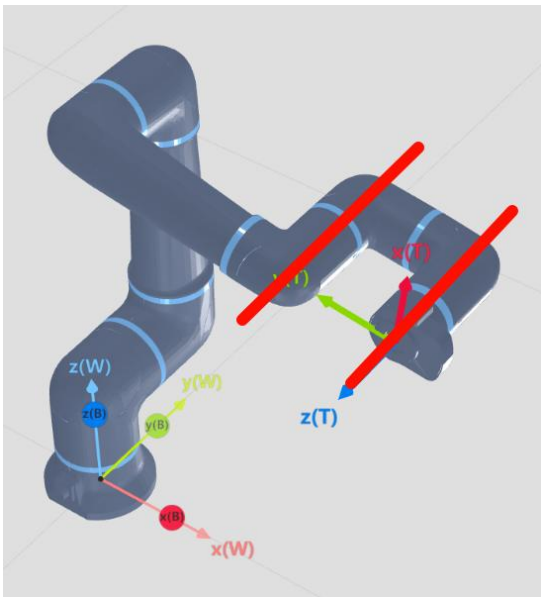
The working range of the robot is a spherical space with the radius of the arm span, but there are some special positions and gestures that are the singularity of the robot, which need to be avoided during use.

Here are three typical kinds of singularities:

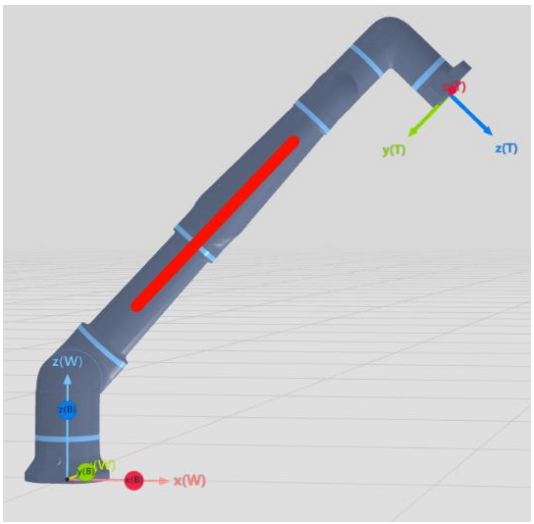
- A cylindrical area with the base of the robot as its base;



- When the a3, a4, a6 joints of the robot are parallel;



- When the Angle between the robot's upper and lower arms is close to 180°



6.4.2 Trigger collision detection

The torque sensor in the joint of the robot will detect the force of the robot in real time. When the force exceeds the expectation, the collision detection will be triggered. At this time, it is necessary to confirm whether the trajectory of the robot is correct and whether there is anything hindering the robot's movement.

If the robot's movement trajectory is correct, the collision detection will still be triggered. At this time, it is necessary to check whether the tool is set correctly, whether the load is set correctly, whether the pipeline of the end tool is normal and so on.

6.4.3 Position/speed out of limit

When the position is out of limit or the speed is out of limit during the robot movement, check whether the program is written correctly. If it is correct, the corresponding parameter limit can be modified in the safety setting in the Settings.

If the position is out of limit, the joint is still out of limit after the error is cleared, and the robot will still alarm when it is powered on again. Therefore, when the joint is out of limit, please follow the following steps.

1. Click,  , clear th 

2. Click t 

escue mode;
3. Click t 

he robot;
4. In the rescue mode, rotate the transfinite joint to the correct position by closing the node;

5. Click t  the robot;
6. Click t  e mode;
7. Repeat the third step to power on the robot.

6.4.4 The joint tracking error is too large

When the joint tracking error is too large during the robot movement, it is necessary to check whether the speed and acceleration of the movement are reasonable, and whether the load of the robot is correct and within the load range of the robot.

6.5 Description of the fault code

At present, the robot has a total of 6 information levels, and the fourth digit of the error code indicates the error level.

Serial Number	Error of the same grade
0	System occupancy
1	Tips
2	Warnings
3	General Errors
4	Serious errors
5	Fatal mistake

- When there is a general error or more, the robot will lose power and stop;
- When there is a warning level error, the robot will slow down and stop;
- If multiple errors occur at the same time, execute according to the highest error level;
- There will be only one error code for the same type of error, but the error content will be specifically displayed on the instructor.

See the appendix for details and error codes.

6.6 Disclaimer

Kuzhuo Technology is committed to creating a better future of human-machine integration. While continuously improving the reliability and performance of our products, we reserve the right to upgrade our products without prior notice. Kuzhuo Technology strives to ensure the accuracy and reliability of the content of this manual, but is not responsible for any errors or omissions in it.

Failure caused by the following circumstances is not covered by this warranty:

- Failure to install, wire, and connect other control devices as required in the user manual;
- Exceed the specifications or standards indicated in the user manual when used;
- Damage to the product due to improper transportation or use;
- Damage caused by accidents or collisions;
- Fire, earthquake, tsunami, lightning, high winds and floods and other natural disasters;
- Changes to system software or internal data;
- Use of this product in radiological equipment, biological testing equipment, or hazardous uses;
- Cannot identify the date of manufacture or warranty start date.
- The fault is not caused by Nanjing Estonkuzhuo Technology Co., LTD.

6.7 Abandoned robot

Abandoned robots must comply with national and local laws and regulations.

7. Overview of the demonstrator interface

7.1 Login interface

The default startup account is admin, password is 123456, and mode is Custom. If the IP of the connected controller has been changed, you can click the red button to set the IP address and port you want to connect to and save it.

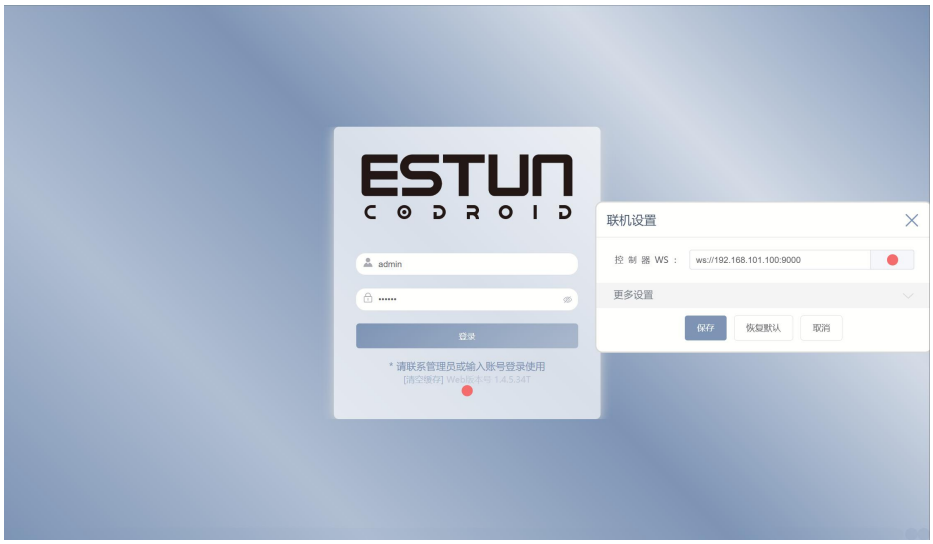


Figure 7-1 Login interface-

7.2 Main interface

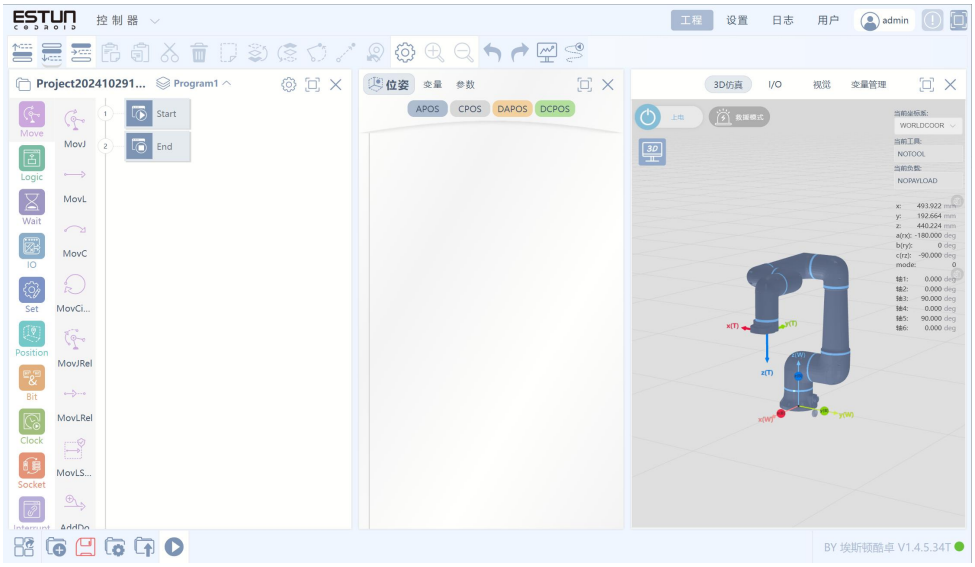


Figure 7-2 Home screen

After successful login, jump to the main interface, which displays TAB content by default and is divided into 4 operable areas: 

7.2.1 Switch the TAB area

Contains "Engineering", "Settings", "Log", "User" four buttons to switch four different display interfaces respectively.

7.2.2 Account setting button

The button displays the current login account. After clicking it, you can "re-log in" to jump to the password interface.

7.2.3 Error message and real-time log window button

The error message and Live log window will pop up.

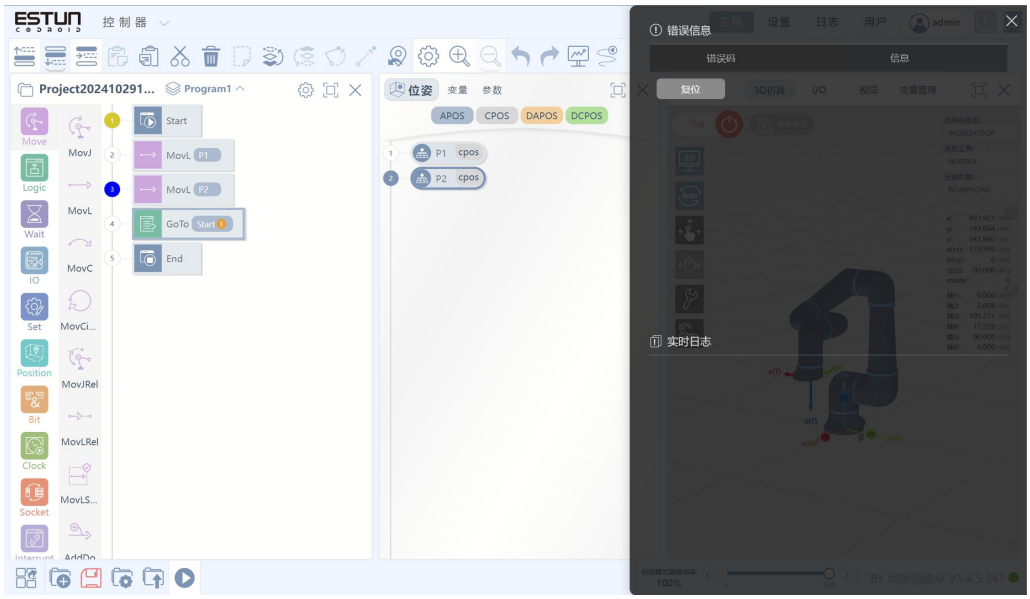


Figure 7-3 Error message and real-time log

When the robot reports an error, an error message will be displayed. The error message includes the error time, error code, and error message description. After confirming that the fault status of the robot is clear, you can press the button to clear the error information. After the error status is cleared, the robot can be powered on again. 

7.2.4 The button is displayed in full screen

Used to switch between full screen and non-full screen (full screen is recommended).

7.3 Engineering TAB

In the engineering TAB, the main menu area, graphics programming area, pose list, variable list, parameter area, 3D display area, IO area, engineering management area, speed multiplier adjustment area.

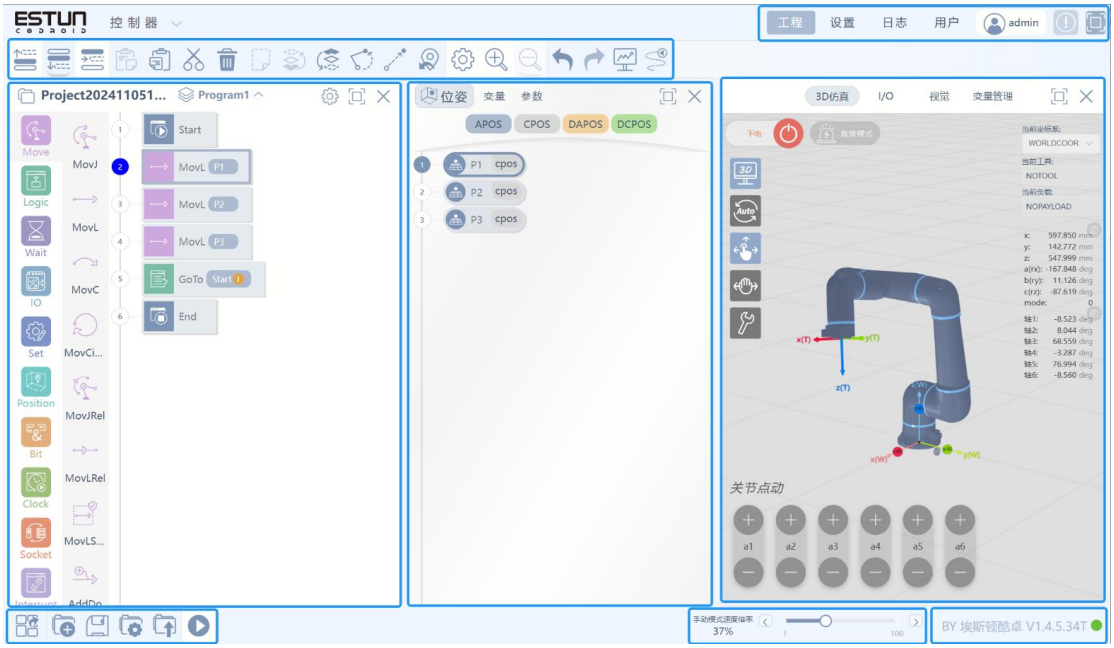


Figure. 7-4 Layout of engineering interface-

7.3.1 Shortcut area

 Insert instructions above;

 Insert instructions below;

 Insert instructions inside;

 Paste instructions;

 Copy the selected instruction;

 Cut the selected instruction;

 Move the instruction up one layer;

 Move instructions down one layer;

 Move the joint to the point position;

 Straight line move to point position;

 Update point position;

 Comment the selected instruction;

 Delete the selected instruction;

 Enlarge the program tree area;

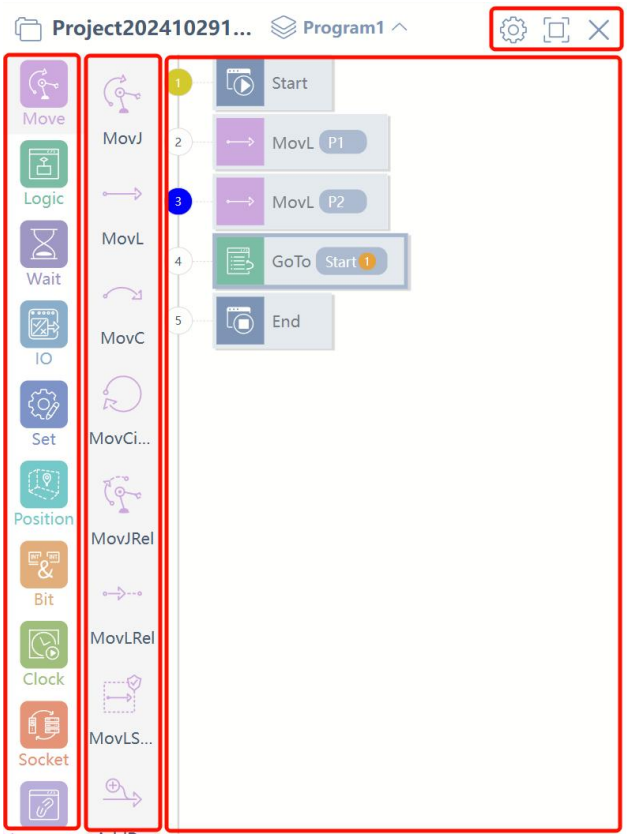
 Shrink the program tree area;

 Undo the current operation;

 Redo the current action;

 Track recording;

7.3.2 Graphical programming area



In the graphics programming area can be divided into four parts, respectively: title area, programming instruction classification area, programming instruction area, program tree area.

7.3.2.1 Title area

The title area has three buttons that are:

Project Property Editing and Task Management:

- Full screen/Restore window, close window
- Name the current project
- Switch multi-program /(single) program
- Multi-program in the program management (subroutine naming, new subroutine, delete subroutine), a project supports a maximum of 30 multi-program.

 Full screen/Recovery window: Full screen/recovery "graphical programming area" display.

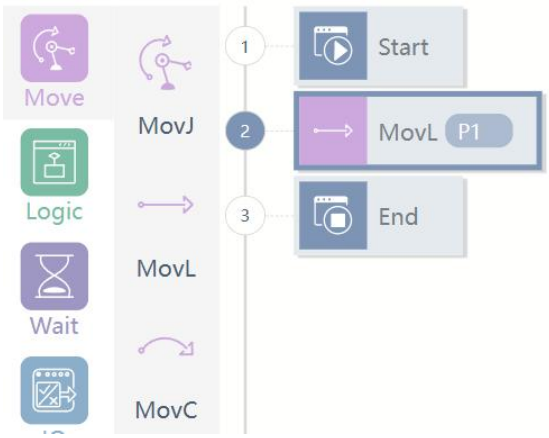
 Close the Graphics Programming Area window.

After you close the Graphics Programming Area, you can click the "Visual Programming" button to restore the display. 

7.3.2.2 Categories of Programming instructions

Currently, they are "Move", "Logic", "Wait", "IO", "Set", "Position", "Bit", "Clock", "Socket", "Interrupt", "Modbus", "Matrix", "String". "RS485" several categories, with the update of the version may be added or deleted.

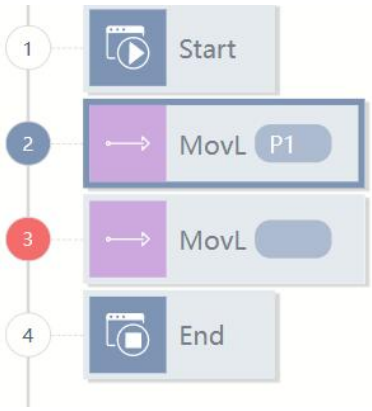
7.3.2.3 Programming instructions



Programming instructions are graphical programming instructions.

After selecting the category, double-click the desired graphical programming instruction to add the instruction to the right program tree, or you can directly drag and drop the instruction to the right program tree.

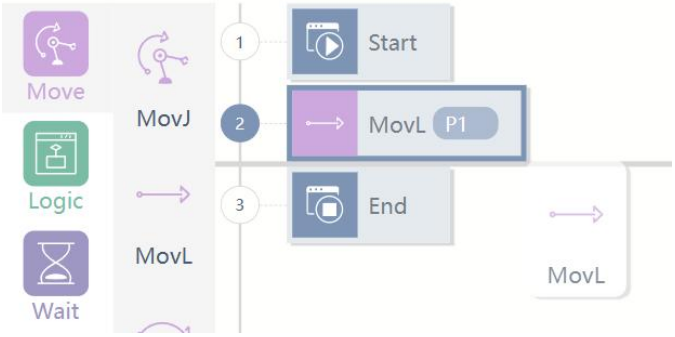
7.3.2.4 Program Tree



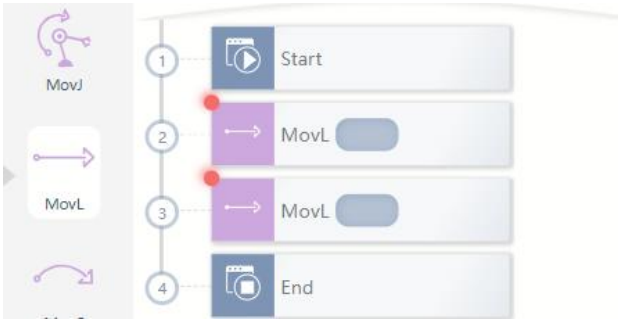
In the program tree, you can add, delete, comment, copy, sort program nodes, you can edit the added program node parameters.

Add instructions

1. You can drag and drop the instructions you want directly into the program tree.



2. Depending on how the shortcut area instruction is inserted, double-click on the instruction to add instructions above, below, or in the sublevel of the currently selected instruction in the program tree.



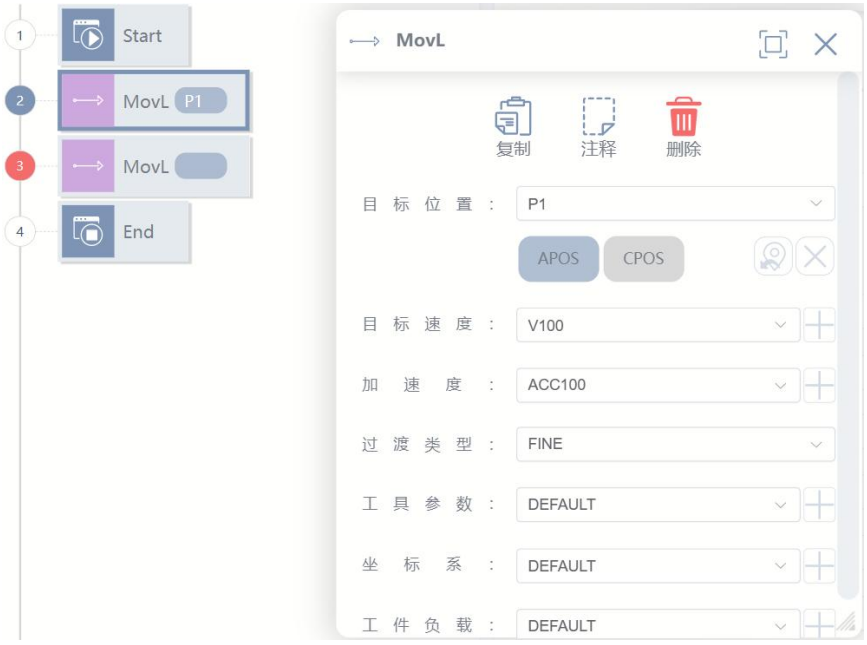
Delete/comment instructions

Double-click the program node you want to delete, or select the list of parameters, corresponding node edit window, and click the Delete button.

Click the program node that needs to be commented, the corresponding node editing

window will pop up, and click the comment button.

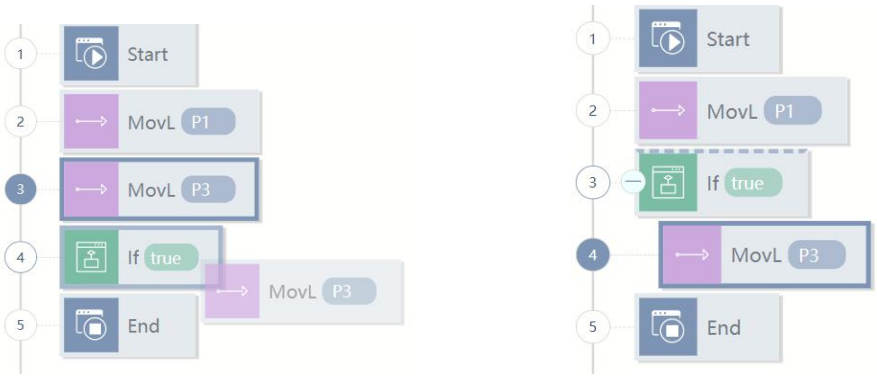
Copy instructions



Click the program node that you want to copy, the corresponding node editing window will pop up, click the copy button.

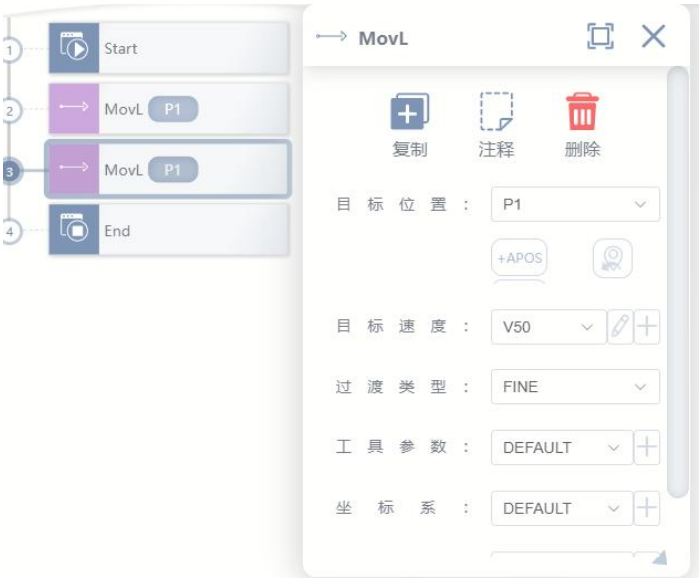
The new node is automatically pasted on the next line of the copied node.

Sort instructions



Select and drag the program node whose sequence needs to be changed and place it in the desired position. Depending on where it is released, you can add instructions above, below, or at the sublevel of an instruction.

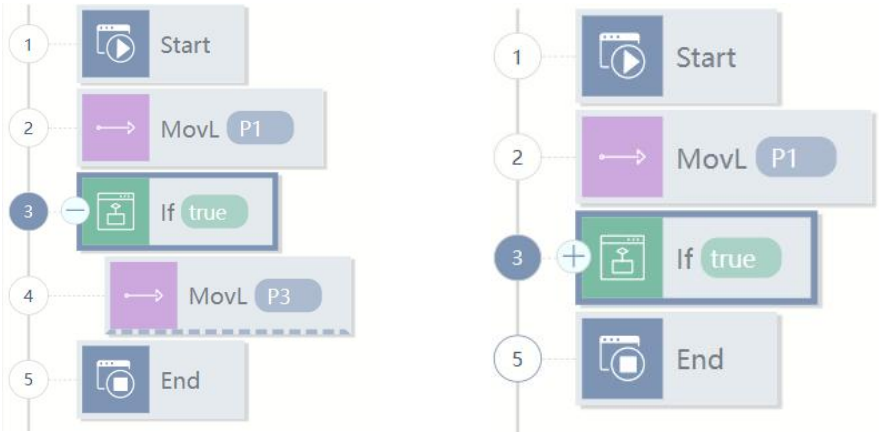
Edit instructions



Double-click the command that needs to be edited or click the parameter list after selecting the command, you can edit the detailed parameters of the command.

Pose list area

Fold instruction



Double-click the instruction that needs to be edited or click the parameter list after selecting the instruction to edit the detailed parameters of the instruction.

Add a new pose

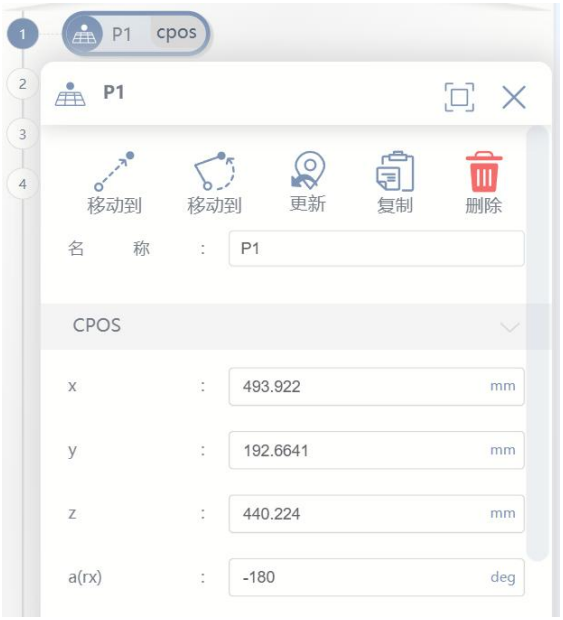
Double-click on the Pose TAB button to add a new pose. Select a different pose type to add the selected pose. Four pose types are:

- CPOS: cartesian position;
- APOS: axis position;
- DCPOS: delta cartesian position increment of cartesian position;
- DAPOS: delta axis position increment of joint position;

CPOS and APOS are the Cartesian pose and joint position of the current robot when added, and all values of DCPOS and DAPOS are 0 when added.

See the Variables section for details on the various point positions.

Edit pose



Click to open the Pose Editor window, in which you can manipulate the position:

- Move to
- Update
- Copy
- Delete
- Edit the point name
- Edit point values CPOS, APOS, DCPOS, DAPOS
- POSCFG configuration

Move into position

In non-automatic mode, there is the "Move to" function with two buttons:

- Move to the current point position in MovL mode.
- Move to the current point position in MovJ mode.

Update pose

Update the current Cartesian pose/joint position to the selected point position by pressing the button.

Copy pose

Copy the selected point through the button and paste it after it, the point name is the last point number added plus 1.

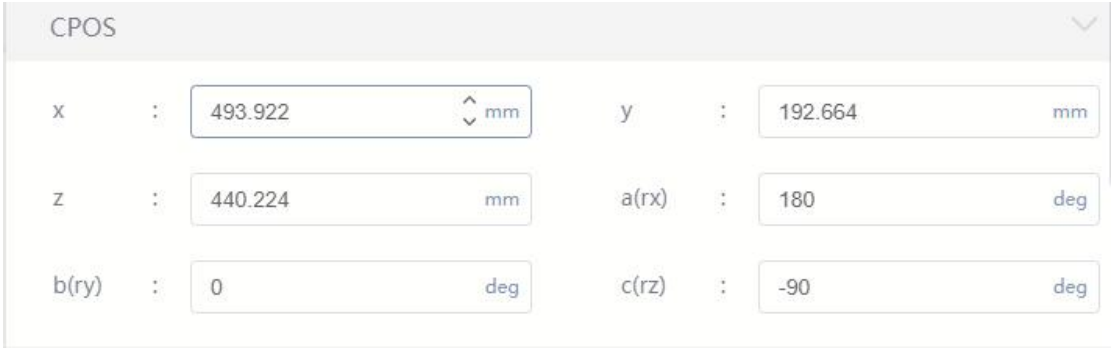
Delete bit pose

Delete the selected bit by pressing the button.

Edit the pose name

Rename the bit through the "Name" text edit box.

Edit the pose values CPOS, APOS, DCPOS, DAPOS



The text boxes under CPOS, APOS, DCPOS, DAPOS can be edited by typing the value to change the value or incremental value of the selected point Cartesian pose/joint position.

POSCG configuration

In the same Cartesian space position, the robot can have multiple joint position combinations (corresponding to multiple solutions of the robot's inverse solution). This property is used to define the morphological configuration data corresponding to the target point in space.

mode = -1 indicates that the current configuration is used. The general six-joint kinematics has eight sets of solutions, and the mode value is defined as 0~7, meaning is as follows:

Mode	腕部中心相对于一轴轴心的关系(flag1) 0：在前；1：在后 $R + L_3 \bullet cos(\theta_2 + \theta_3) + L_2 \bullet sin\,\theta_2 + S \bullet sin(\theta_2 + \theta_3)$	Axis3(flag3) $(\theta_3 + 90 - arc\tan(S/L3))$ 0: [0,180] 1:(-180,0)	Axis5(flag5) (θ_5) 0: [0,180] 1:(-180,0)
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

7.3.3 Variable list area

Variables TAB where defined variables can be stored. Refer to Chapter 8 for specific descriptions of each variable.

Classification of Variables

System: storage DI/DO, AI/AO and other variables, do not allow users to create, edit, delete variables

Global: The scope of the variable is "global", allowing users to create, edit, delete variables

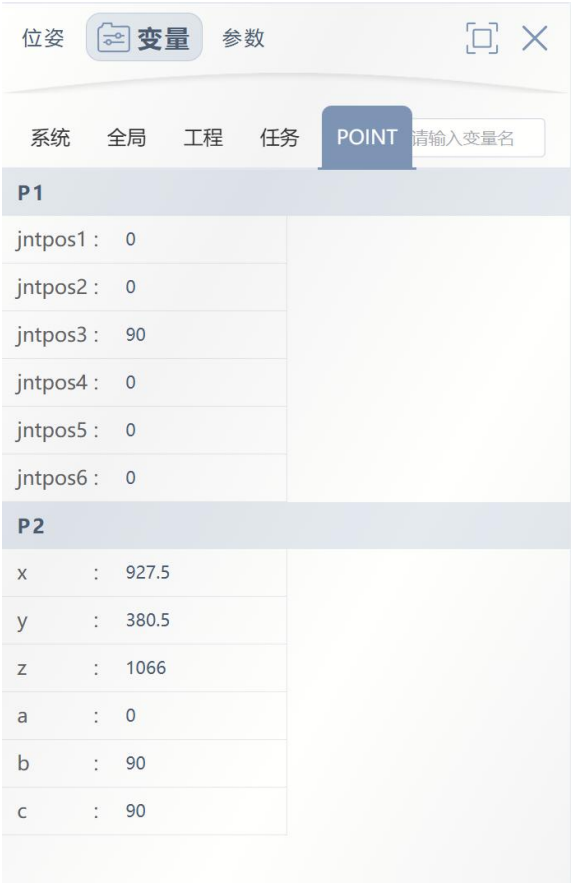
Project: Variables whose scope is "Project", allowing users to create, edit, delete variables

Task: A variable whose scope is "Task", allowing users to create, edit, and delete variables

POINT variable

When the program is running, the POINT page is added where the current point variable is

refreshed in real time.



New variable

In the variable label button, you can click to add a new variable, select a different variable classification, type to add the selected variable classification, type, specific

definition of reference variable introduction.

Edit variables

Click the button to edit the current variable (name, power down save, modify value)

Delete variable

Click the button to delete when variable. 

Hold type variable

The memory of power failure. The program does not initialize the variable to the value below until it starts running, except for the first time.

Save variables

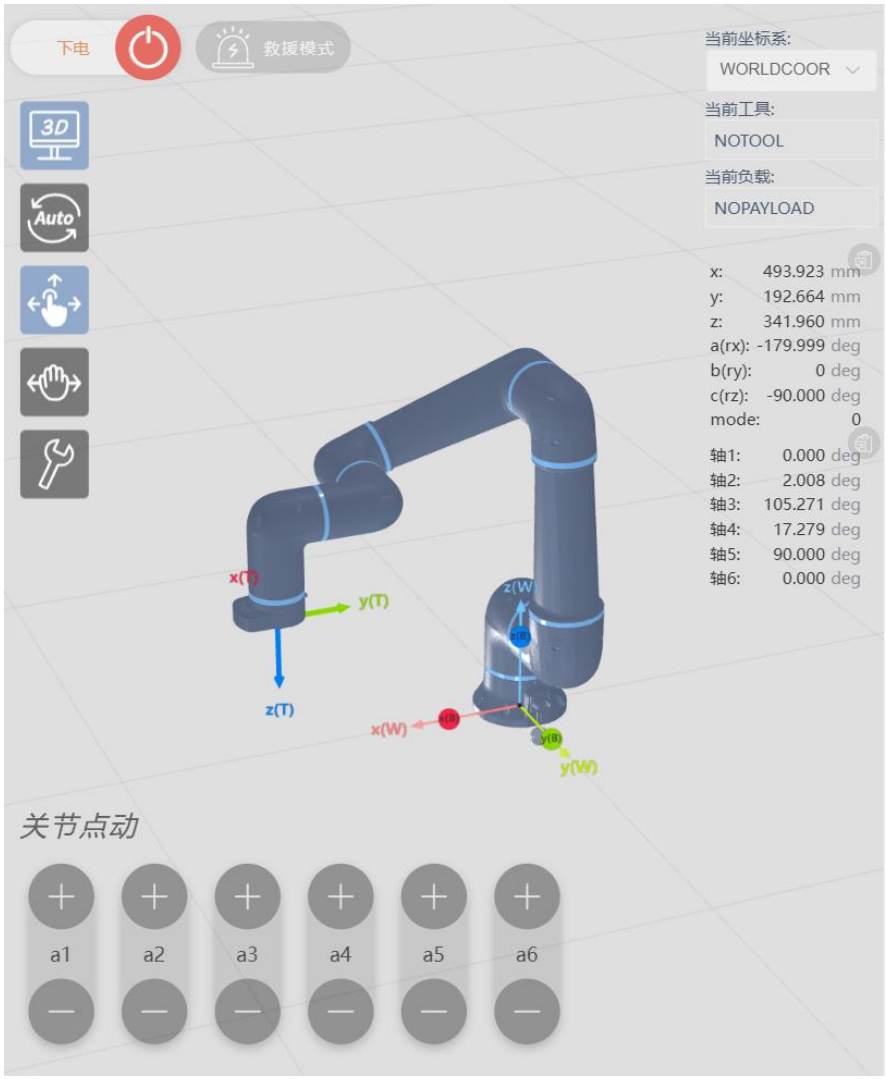
Click the button to save the variable to the controller. 

7.3.4 Parameters Area

View, edit, delete the instruction details of the selected program tree. Each instruction has slightly different parameters. See Chapter 10 for details.

7.3.5 3D Simulation

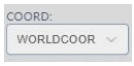
Display real-time robot simulation animation and Cartesian coordinate pose, joint position.



The power-on/power-off button is used to control the switch of robot power-on/power-off;

Rescue mode, in which the joint is clicked without the limit of motion range (enter "rescue mode" when the robot is "powered off", and power on the robot to click the joint after it is turned on);

In automatic mode, if the robot is not on the preset track after the robot program is suspended, the robot must be manually moved to the recovery point before continuing to run the program;



Current coordinate system. Toggle the User coordinate system variable currently in use;

Current tool. Toggle the currently used "Tool Coordinate System" variable in the Settings TAB - Basic - Machinery;

Current load, switch the currently used "Load" variable in the Settings TAB - Foundation - Machinery;

Simulation/real machine mode switch, switch the robot "simulation mode" and "real machine mode" in the power off state, in the simulation mode, the real machine robot will not move;

Automatic mode switch, switch the robot to "automatic mode" to run the program;


Manual mode switch, switch the robot to "manual mode";


Drag sensitivity, you can adjust the drag teaching sensitivity and whether to open the attitude lock;


Toolbox, including switching Angle of view, clear motion trajectory, zero calibration, back to zero, back packing and other tools;

Switch perspective, quickly switch the perspective of 3D simulation;

Clear the track line, clear the end TCP track line in the 3D simulation space;

Zero point calibration, calibration of mechanical zero and torque zero, the use of this function please contact the manufacturer;

Return to zero, click and long press the button in the lower right corner to return to the robot home point;

Back to the packing position, click and long press the button in the lower right corner to return to the robot packing posture;


Target position display, whether to display the target position of the robot's next

instruction in automatic mode;

Point mode switch, switch "joint movement (joint movement)"/" end point movement (Cartesian movement) ", and can adjust different speeds by speed multiplier.

7.3.6 I/0

The I/O interface displays the status of all digital I/ OS and analog I/ OS. In the "unlocked" state, I/ OS can be manually operated on this interface. In the "locked" state, I/ OS cannot be manually operated and can be operated by programs. 

Digital I/O(Sim) and analog I/O(Sim) are virtual I/OS without actual physical ports.

The force option can force the corresponding IO to change to the manually selected state.

7.3.7 Engineering Management Area

You can manage the project in the Project Management menu.

Project Settings, change language, change format, refresh page, lock window, variable management and set online options;

New project, create a robot project;

Save project, when the save project button is "red", there are changes in the current project is not saved, when the save project button is "blue", the current project changes have been saved;

Project management, you can download, copy and delete the saved project;

Import project, import saved in the local project;

Run, run the current project (single step execution, automatic execution), run the project must be in the "automatic mode".

In the project management dialog box, you can download, copy and delete the project. 

7.3.8 Speed multiplier adjustment area

The speed magnification bar can adjust the speed magnification, and the values of manual click mode and automatic operation mode are independent. The value ranges from 1% to 100%.

The actual running speed of robot in automatic mode = motion command speed x speed multiplier.

In manual mode, the maximum moving speed of the closing node is 30° /s × 100%, the maximum Cartesian point moving line speed is 250mm/s × 100%, and the maximum Cartesian point rotating angular speed is 30° /s × 100%. The value can be modified in the relevant options of the Settings TAB.

7.4 Settings TAB

Automatically power off the robot when saving parameters

The robot is automatically powered off when the parameters are saved, and the new parameters are applied when the robot is powered on again.

7.4.1 IO

7.4.1.1 DI function configuration

When the system detects that the corresponding digital input variable meets the trigger condition, it executes the corresponding purpose function. Click to create a new function configuration.

Start dragging: Open Manual drag robot mode in mode;

Stop Drag: Turn off manual drag robot mode in mode;

Power on: Enable the robot to power on;

Power-on in rescue mode: Enable the robot to power on in rescue mode, and the safety detection in this mode is temporarily turned off.

Power-off: enable when the robot is powered off;

Switch to automatic mode: the robot switches to the automatic operation program mode;

Switch to manual mode: the robot switches to manual teaching mode;

Run the last saved program: Run the last saved program in auto run mode;

Run specified program: Run the program specified in the drop down box in auto run mode;

Stop running: the robot stops running the program;

To pause a robot from running a program;

To continue running a paused program;

Error reset: clear the robot error;

7.4.1.2 DO function configuration

When the system detects that the trigger condition is met, the corresponding digital output function is executed. Click to create a new function configuration.

Exception: Output corresponding level when the robot is abnormal;

Program running: robot program running output corresponding level;

Program pause: when the robot program is suspended, output corresponding level;

Located at the safe point: when the robot is located at the safe point, output the

corresponding level;

7.4.1.3 Panel IO Settings

IO Settings enable analog output and set analog input mode on the panel of the control cabinet.

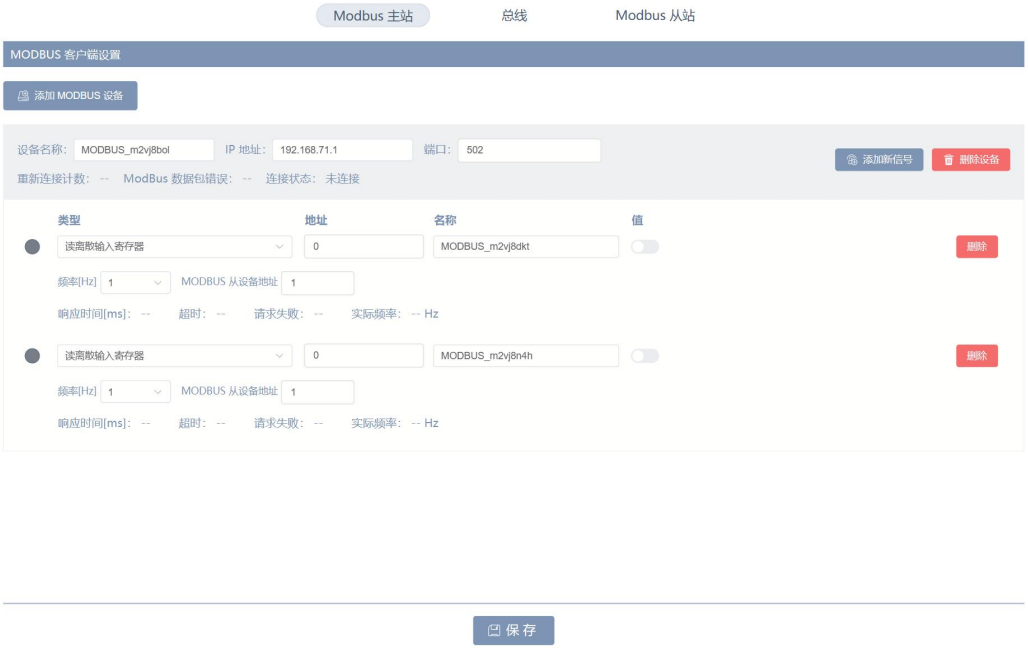
The analog output port must be connected to the load, otherwise the robot will report an error. If not in use, the corresponding port must be closed.

Use analog input port must indicate use mode, current mode or voltage mode.

7.4.2 communication

7.4.2.1 MODBUS master station

The MODBUS Master (client) parameters can be set here. You can create a connection between the local machine and the MODBUS slave station (server) with the target IP address. Each signal has a unique name and is therefore available in the program.



Adding devices

This button adds a new MODBUS slave device.

Remove a device

This button deletes the MODBUS slave device and all of its signals.

Device name

You can change the device name to distinguish one device from another.

IP Address

The IP address of the MODBUS slave device, where you can change the IP address.

Port

The port address of the MODBUS slave device, where you can change the port address.

Reconnect Count

The number of times a TCP connection was closed and reconnected.

Modbus packet error

The number of packets received containing errors (i.e., invalid length, missing data, TCP socket errors).

Connection status

TCP connection status.

Adding signals

This button adds a signal to the appropriate MODBUS slave device.

Remove a signal

This button removes the signal from the corresponding MODBUS slave station device.

Types

Selectable signal type. Available types include:

Read single coil registers (read output coils), read discrete input registers (read input coils), read single hold registers (read output coils), read input registers (read input registers), write single hold registers (write output coils), and write single hold registers (write output registers).

Address

Displays the address on the remote MODBUS slave device, optionally select a different address. The valid address depends on the manufacturer and the configuration of the remote MODBUS slave device.

Name

You can specify a name for the signal. The signal name is used when the signal is used in the program

Frequency

Can be used to change the update frequency of the signal. The update frequency is the

request frequency that points to the remote MODBUS slave station device sending a request to read or write the signal value. When the frequency is set to 0, MODBUS requests will be initiated on demand using instructions in the program type.

Slave device address

This text field can be used to set specific slave device addresses for requests corresponding to specific signals. The value must be in the range 0 to 255, with the default being 255. To change this value, it is recommended to consult the Remote MODBUS device manual first to verify that it functions properly after changing the address from the device.

Response time [ms]

Time between sending a MODBUS request and receiving a response, updated only when communication is active.

Timeout

Number of MODBUS requests that went unanswered.

Requests failed

The number of packets that could not be sent due to invalid socket status.

Actual frequency

Average frequency of signal status updates for the master station equipment (client). This value is recalculated each time the signal receives a response from the slave station device (server).

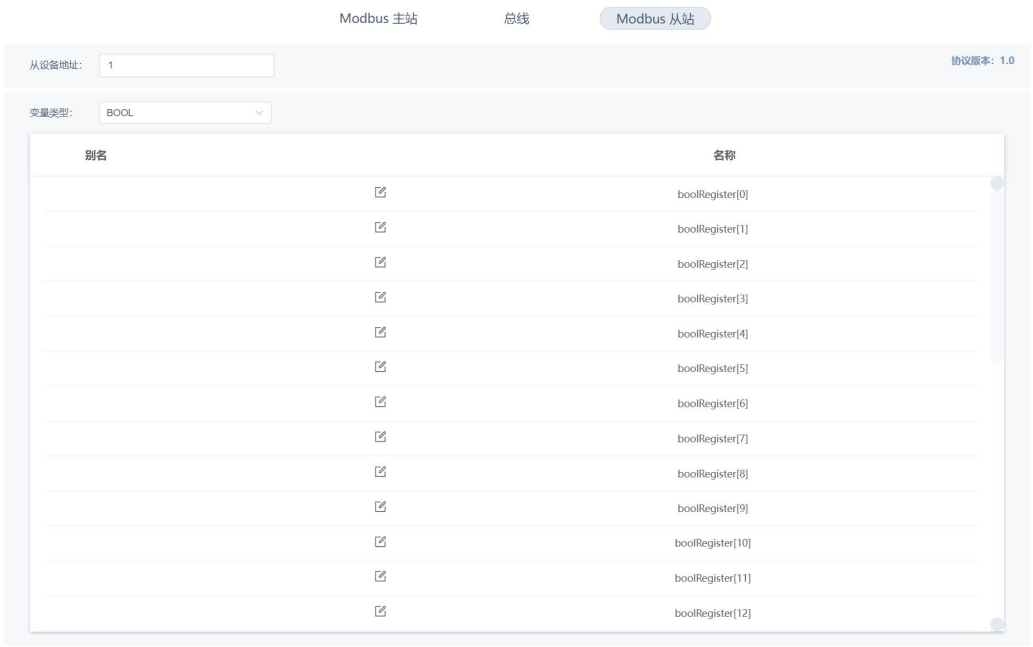
Save

Save your Settings and refresh all MODBUS connections. All MODBUS slave devices will be disconnected and reconnected. All statistics are cleared.

7.4.2.2 MODBUS slave station

The MODBUS slave (client) parameters can be set here. The robot MODBUS slave service will automatically start after the machine is powered on. The IP address is the local IP address and the port is 502. The port cannot be modified.

When the robot serves as the MODBUS slave station, its hold register has pre-defined information, such as robot basic information, joint information, terminal information, IO information, basic control, in addition, it also contains integer type variables, real type variables and Boolean type variables that can be customized.



Slave device address

It is used to distinguish device addresses. It is meaningless for Modbus-TCP, and is useful for conversion between MODBUS protocols.

Variable Types

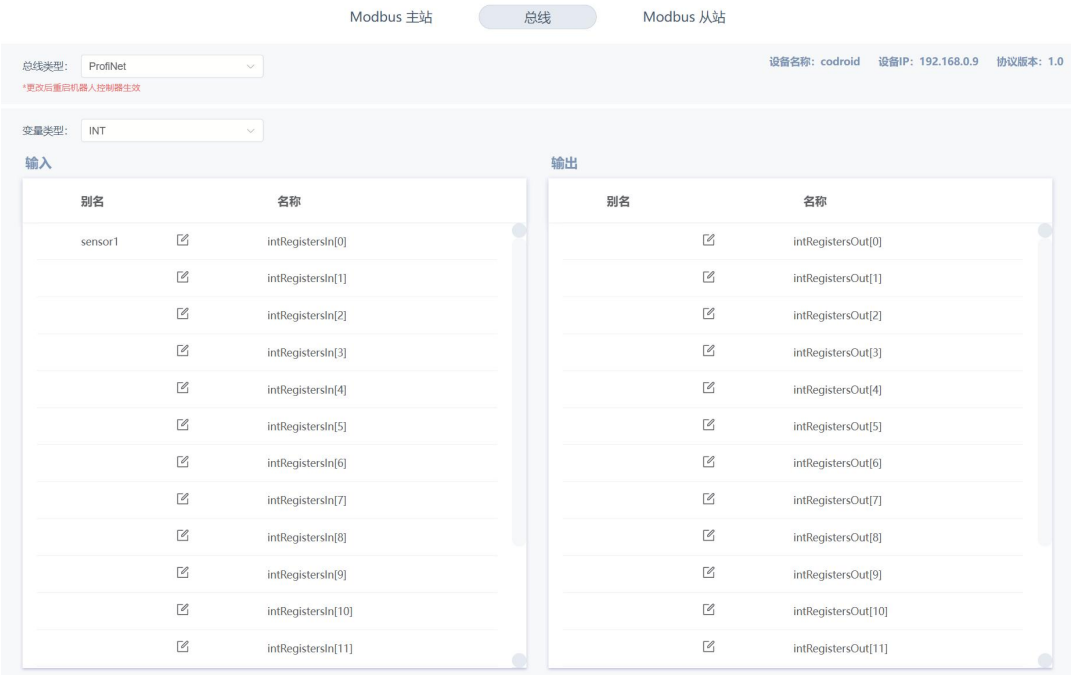
Used to filter the list of variables below. Toggle to display different types of variables.

Aliases

Variables with aliases will be displayed in system variables. Variables without aliases will not be displayed. A variable name cannot be the same as an existing variable name.

7.4.2.3 Bus

The robot can adapt different bus protocols by matching different expansion modules. The supported modules are ProfiNet, EtherNetIP, AnyBus, all of which are only supported as slave devices.



Bus Type

It is used to distinguish different devices. You need to select the corresponding option according to the actual installed expansion module. After the modification, you need to restart the robot controller to make it take effect.

Device name

This robot is the name of the device used as the slave station.

Device IP

The device IP address of the expansion module is used when connecting to other devices. The device IP is not the same as the robot IP.

Protocol version

This bus protocol version.

Variable type

Used to filter the list of variables below. Switch to display different types of variables.

Aliases

Variables with aliases will be displayed in system variables. Variables without aliases will not be displayed. A variable name cannot be the same as an existing variable. After changing the alias, you need to save the alias and save the Settings for it to take effect.

7.4.3 Basics

7.4.3.1 IP address

The IP address of the robot can be changed. The change takes effect after the control cabinet is powered off and restarted.

7.4.3.2 Serial Number

The serial number of the whole machine, the serial number of the control cabinet, the serial number of the manipulator arm and the serial number of each joint are the unique identification of each component of the robot. The serial number of the whole machine will also be indicated on the label of the robot arm and control cabinet.

7.4.3.3 Default tool

By creating a variable of type TOOL in a variable, you can select the tool you created from the drop-down box of the default tool.

The TOOL variable contains the position and rotation of the TCP with respect to the robot's end flange, the mass of the tool, the center of mass of the tool (with respect to the TCP coordinate system), and the inertia tensor of the tool.

The default tool is the tool parameters that are loaded when the robot is turned on. Incorrect selection of the default tool may cause the robot to stop, which may seriously damage the robot joints.

7.4.3.4 Default load

By creating a variable of type PAYLOAD in a variable, you can select the payload to be created in the default payload drop-down box.

The PAYLOAD variable contains the mass, center of mass, and inertia tensor of the load.

The default load is the load parameter that is loaded at boot time. Incorrect selection of the default load may cause the robot to stop, which may seriously damage the robot joints.

7.4.3.5 DH parameters

Users can view the DH parameters of this robot here.

7.4.3.6 Installation

Optional pre-set mounting or customize its mounting offset and mounting rotation with respect to the world coordinate system. The mounting rotation and rotation do not change after the robot is fixed.

7.4.3.7 xyz offset

The installation-offset parameter represents the offset of the robot base with respect to the world coordinate system and has no practical significance in a single robot system. In a multi-robot system, it can represent the relative position relationship between robots.

7.4.3.8 abc rotation

The parameters of the installation-rotation are related to the installation attitude of the robot. When installing at other angles, the parameters need to be set in the installation-rotation. After setting the parameters, the robot model on the right will rotate in real time according to the input parameters. When the simulated robot attitude is consistent with the actual situation, click the Save button and re-power on, the parameters will take effect.

7.4.4 Security

Safety Rules switch can choose whether to enable safety rules, no rules will take effect after the main switch is turned off.

7.4.4.1 Safety point

Set the robot attitude of the safety point position, you can add the output signal when the robot is at the point position in the set IO.

7.4.4.2 Manual mode terminal speed limit

The maximum Cartesian speed of moving to a point in manual mode.

7.4.4.3 Load check sensitivity rating

When the towing robot is enabled, the robot will check whether the current load is correct before the function is opened. If the actual load and the theoretical load deviate too much, the

robot will not open the drag in order to protect itself and the operator's safety. Adjusting the sensitivity level can limit the deviation threshold.

7.4.4.4 Joint collision protection threshold

Kuzo S Series robots have torque sensors in each joint to detect the torque applied to the joint. When the robot is powered on, when the torque value is detected to be greater than the joint output torque limit threshold, the robot will report an error and power off. At this time, it is necessary to check the cause of this situation of the robot, and power on the robot again after the problem is removed.

The reason why the robot may exceed the torque limit is:

1. The actual load at the end does not match the one in the setup;
2. The robot collides;
3. Improper speed and acceleration Settings;
4. Other situations.

You can modify the threshold according to the actual application, but you are not advised to disable the protection, which may cause security risks.

7.4.4.5 End impact protection threshold

The component force and torque of the force on the end of the robot in the x,y, and z directions. When the robot is powered on, when the force and torque are detected to exceed the limit threshold, the robot will report an error and power off. At this time, it is necessary to check the cause of this situation of the robot, and power on the robot again after the problem is removed.

The robot may have torque limit reasons:

1. The actual load at the end does not match the one in the setup;
2. The robot collides;
3. And improper speed and acceleration Settings.

The customer can modify the threshold according to the actual application, but it is not recommended to turn off the protection, which may cause security risks.

7.4.4.6 Joint Limit

Joint limits are used to limit the movement of each robot joint in joint space, defining the range of positions for each joint. The threshold can be modified by the customer according to the practical application. If the threshold is set too small, it will affect the robot's range of motion.

7.4.4.7 End Limit

The end limit is used to limit the motion position of the robot TCP, defining the x,y,z axis direction and rotation position range. Customers can modify the threshold value according to the practical application. If the threshold is set too small, it will affect the robot's range of motion.

7.4.5 Motion

The motion parameters define the maximum speed and acceleration of the robot in automatic mode or manual mode, and the maximum acceleration.

7.4.5.1 Point-motion mode

Joint speed

In manual mode, the joint movement speed is up to 30°/s, where you can limit the maximum speed of manual joint movement.

End line speed

In manual mode, the maximum Cartesian point line speed is 250mm/s, and the maximum line speed of Cartesian manual point can be limited here.

Terminal angular velocity

In manual mode, the maximum end rotation angular velocity of Cartesian spoting is 30°/s, where the maximum end rotation angular velocity of Cartesian manual spoting can be limited.

7.4.5.2 Automatic Mode

Maximum joint speed

The maximum speed the robot can reach while moving in automatic mode. After setting the maximum SPEED, the TCP speed setting will be limited to less than this maximum when creating a variable of type speed.

Maximum joint acceleration

Joint acceleration in auto mode limits the maximum acceleration. The user can adjust the value of the maximum acceleration appropriately according to the application, which can improve the motion beat. However, if the acceleration is set too large, there may be a phenomenon of robot start-stop jitter, and long-term use of unreasonable acceleration may cause damage to the joint reducer.

Maximum joint acceleration

The maximum acceleration the robot can achieve when moving in automatic mode. The smaller the value, the smoother the motion will be but the longer it will take.

Maximum terminal velocity

The maximum linear speed of the robot's end motion in automatic mode. After setting the maximum SPEED, the TCP speed setting will be limited to less than this maximum when creating a variable of type speed.

Maximum terminal acceleration

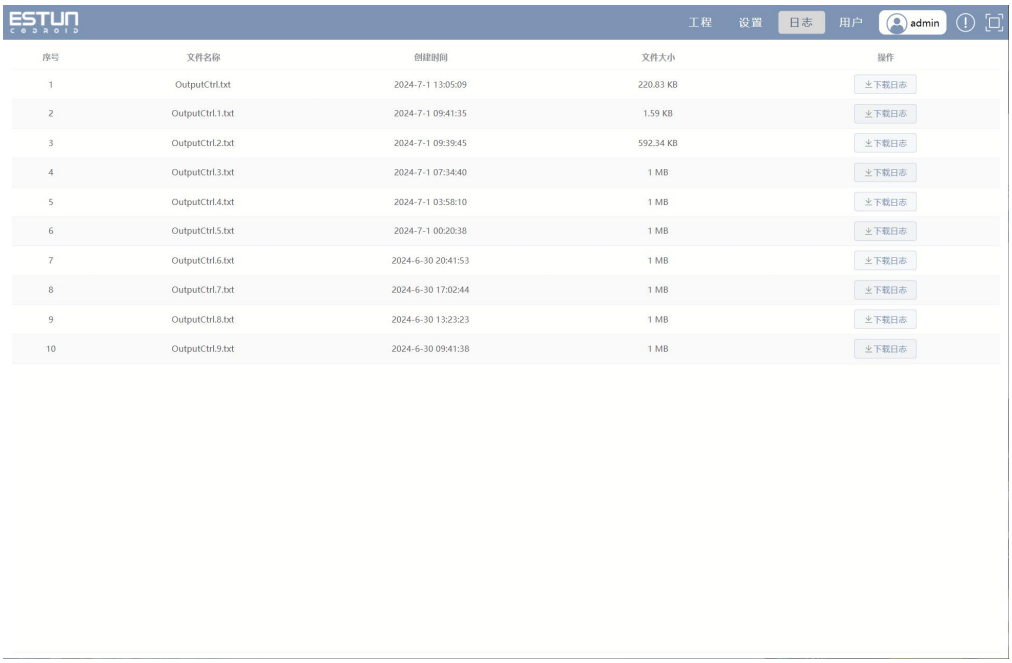
The linear acceleration at the end of the robot in automatic mode limits the maximum acceleration. The user can adjust the value of the maximum acceleration appropriately according to the application, which can improve the motion beat. However, if the acceleration is set too large, there may be a phenomenon of robot start-stop jitter, and long-term use of unreasonable acceleration may cause damage to the joint reducer.

Maximum acceleration at the end

The maximum line plus acceleration that the robot can achieve when moving in automatic mode. The smaller the value, the smoother the motion will be but the longer it will take.

7.5 Log TAB

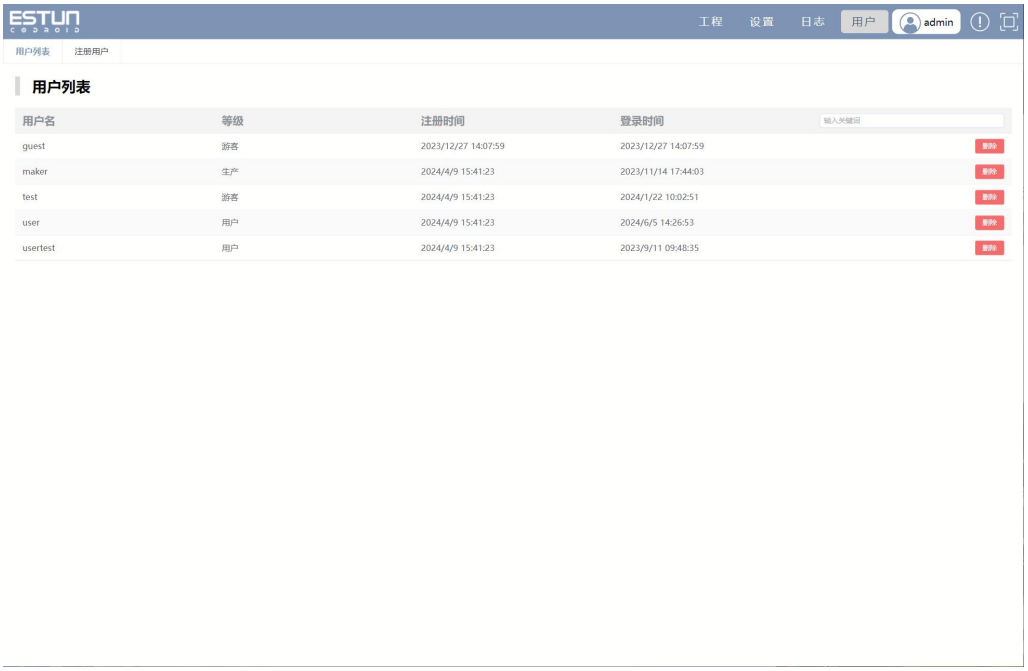
The log module records some operation exceptions of the user and provides related prompts to help the use of the software. At the same time, it can provide the corresponding window prompts when encountering problems, and provide information to professionals to get help and solve problems.



Click the exclamation point button in the upper right corner of the program to view error messages. The blinking button indicates that the program has an error, and the program stops running.

Only the latest 10 system logs are kept. Click the button in the log TAB to download this log information to your local computer. 

7.6 User TAB



The admin user can create and delete users here.

The following lists the initial accounts and passwords that can be used. Different accounts have different permissions. For details, see the appendix.

Account Number	Password	Grade
user	123456	User
admin	123456	Admin

8. Introduction to Variables

8.1 Overview of Variables

Different domains support different variable types, as described below:

- System domain: predefined variables that cannot be edited.
- Global domain: IO data type, PLC data type, socket data type, location data type, region data type, basic data type, clock data type, palletizing data type, system data type.
- Engineering domain: IO data type, socket data type, location data type, basic data type, Palletizing data type, system data type.
- Task Domain: IO data type, socket data type, location data type, basic data type, system data type.

  Retention variables will lose power memory, except for the first run of the program will not initialize the variable to the following variable value before the start of the run.

The following "name" is reserved for the system, the name is not case sensitive, the user cannot create a variable with the same name:

abs, acos, and, asin, assert, atan, break, ceil, collectgarbage, coroutine, cos, debug, deg, do, dofile, else, elseif, end, error, exp, fals e, findEnd, floor, fmod, for, format, function, getAt, getmetatable, goto, huge, if, in, io, ipairs, left, load, loadfile, local, log, math, max min modf, next, nil, not, or, os, package, pairs, pcall, pi, print, rad, random, randomseed, rawequal, rawget, rawset, real, repeat, re quire, return, reverse, right, select, strcmp, setmetatable, sin, sqrt, string, table, tan, then, tonumber, tostring, true, type, until, w hile, xpcall, AI, AO, APOS, APosToCPos, APosToStr, AREA, AreaActivate, AreaDeactivate, ARRAYS, BitAnd, BitNeg, BitOr, BitXOr, BitLSH, Bi tRSH, BOOL, CalcTool, CalcCoord, CALL, CenterPos, CLKRead, CLKReset, CLKStart, CLKStop, CLOCK, CompareAI, CompareSimAI, CPOS, CPosToAP os, CPosToCPos, CPosToStr, DAPOS, DCPOS, DI, DO, ELSE, ELSIF, ENDIF, ENDWHILE, EXTTCP, GetCamPos, GetCurAPos, GetCurCPos, GetCurOverRid e, GetDI8421, GetMatrix, GetSimAIToVar, GetSimDI8421, GetSimDIToVar, GetTrackId, GOTO, Hand, IF, InertiaTensor, INT, IToStr, LABEL, Lo adDyn, MovArch, MovC, MovCW, MovCircle, MovCircleW, MovE, MovH, MovJ, MovJRel, MovJSearch, MovL, MovLRel, MovLSearch, MovLSync, MovJSyn cQuit, MovLSyncQuit, MovLW, OnDistance, OnParameter, PalletFromGet, PalletFromPut, PalletReset, PalletToGet, PalletToPut, PAYLOAD PLCBOOL PLCDINT, PLCINT, PLCREAL, POLYHEDRON, PolyhedronAreaActivate, PolyhedronAreaDeactivate, POSCFG, POSITIONER,

PulseOut, Pul seSimOut, ReadModbusReg, REAL, RefRobotAxis, RET, RETURN, RToStr, RUN, SendMessage, SetAxisVibraBLevel, SetAO, SetCartDyn, SetCoord, SetDIEdge, SetDO, SetDO8421, SetExternalTCP, SetJointDyn, SetMotionMode, SetOverRide, SetPayload, SetPositioner, SetRestorePC, Set RtInfo, SetRtToErr, SetRtWarning, SetMatrix, SetSimAO, SetSimAOByVar, SetSimDIEdge, SetSimDO, SetSimDO8421, SetSimDOByVar, SetTarg etPos, SetTool, SetSyncoord, SimAI, SimAO, SimDI, SimDO, SocketClose, SocketCreate, SocketReadInt, SocketReadReal, SocketReadStr, So cketSendStr, SoftFloatStart, SoftFloatStop, SPEED, Stop, STRING, StrToI, StrToR, SYNCOORD, SynCToUserC, TOOL, Tracking, TranStrToApo s, TranStrToCpos, TranStrToInt, TranStrToReal, TrigCam, trimLeft, trimRight, USERCOOR, Wait, WaitAI, WaitCondition, WaitConvDis, Wai tDI, WaitDI8421, Waitfinish, WaitFinishCAM, WaitSimAI, WaitSimDI, WaitSimDI8421, WaitWObj, WEAVE, WHILE, WriteModbusReg, ZONE, ToolO ffset, UserOffset.

8.2 Variables

8.2.1 POSE

Store the absolute coordinate value and offset value of each axis under the robot joint and Cartesian space.

Parameters	**Parameters**	**Data type**	**Meaning of the parameter**
APOS Store the joint Angle values for each axis under the joint space.	jntpos1	real	Angle of joint 1 axis.
	jntpos2	real	Angle of the joint 2 axis.
	jntpos3	real	Angle of the joint 3 axis.
	jntpos4	real	Angle of the joint 4 axis.
	jntpos5	real	Angle of the joint 5 axis.
	jntpos6	real	Angle of the joint 6 axis.
CPOS Store TCP points in	x	real	The coordinates of the TCP point in the x direction on the reference coordinate system.
	y	real	The coordinate of the TCP point in the y direction on the reference coordinate system.

Cartesian coordinates.	z	real	The coordinates of the TCP point in the z direction on the reference coordinate system.
	a(rx)	real	Euler Angle of rotation of the TCP point with respect to the X-axis of the fixed reference coordinate system.
	b(ry)	real	Euler Angle of rotation of the TCP point with respect to the y axis of the fixed reference coordinate system.
	c(rz)	real	Euler Angle of rotation of the TCP point with respect to the z-axis of the fixed reference coordinate system.
DAPOS Store the relative joint Angle offset of each axis under the joint space.	djntpos1	real	Angular offset of the joint 1 axis.
	djntpos2	real	The angular offset of the joint 2 axis.
	djntpos3	real	Angular offset of the joint 3 axis.
	djntpos4	real	Angular offset of the joint 4 axis.
	djntpos5	real	The angular offset of the joint 5 axis.
	djntpos6	real	Angular offset of joint 6 axis.
DCPOS Store the position of the TCP point in Cartesian coordinates.	dx	real	The coordinate offset of the TCP point in the x direction on the reference coordinate system.
	dy	real	The coordinate offset of the TCP point in the y direction on the reference coordinate system.
	dz	real	The offset of the TCP point's coordinates in the z direction on the reference coordinate system.
	da	real	Euler Angle offset of the rotation of the TCP point with respect to the X-axis of the reference coordinate system.
	db	real	Euler Angle offset of the rotation of the TCP point with respect to the y axis of the reference coordinate system.
	dc	real	Euler Angle offset of the rotation of the TCP point with respect to the z-axis of the reference coordinate system.

8.2.2 Basic Data Types

Parameters	Data type	Scope	Remarks
STRING	String	Global, Project, Task	String
BOOL	Boole	The whole picture, the project, the task	Value range: true, false
INT	Plastic	Global, engineering, task	Value range: -99999999999 to 99999999999
REAL	Real numbers	Global, engineering, task	Value range: -99999999999 to 99999999999
BoolOneArray	Boolean array	Global, project, task	Data length: 1~255
IntOneArray	Shaping Array	The whole picture, the project, the task	Data length: 1~255
RealOneArray	Arrays of real numbers	Global, project, task	Data length: 1~255

8.2.3 SPEED

Used to define the speed of movement of the robot and the external axis. In order to facilitate the use of users, the system presets commonly used speed variables (system variables, do not allow users to modify), at the same time can support users in the scope of global, engineering, program three variables to create, delete, modify and other operations.

Parameters	Data type	Meaning of the parameter
per	real	Percentage joint speed. Used to specify the speed of movement when the joint movement command is used, for commands such as MovJ, with values ranging from 1% to 100%.

tcp	real	TCP line speed. Define the line speed of the end point of the robot for MovL, MovC and other straight line arc motion instructions.
ori	real	Space spin speed. Define the rotation speed of the robot's end point attitude for MovL, MovC and other straight arc motion instructions.
exj_l	real	External axis speed. Define the speed of motion of the external straight axis.
exj_r	real	External axial angular velocity. Define the speed of motion of the external axis of rotation.

8.2.4 ACC

Used to define the acceleration of the motion of the robot and the external axis. In order to have a fast enough motion speed, this parameter is usually adjusted, but it is not recommended that the value is too large, otherwise it will cause vibration and even damage the joint after long-term operation.

Parameters	**Data type**	**Meaning of the parameter**
joint	real	Joint acceleration hundred. Used to specify the acceleration of motion when the joint movement command is specified, suitable for commands such as MovJ.
tcp	real	TCP line acceleration. Define the line acceleration of the end point of the robot for MovL, MovC and other straight line arc motion instructions.
ori	real	Space rotation acceleration. Define the rotational acceleration of the robot's end point attitude for MovL, MovC and other straight arc motion instructions.

8.2.5 ZONE

Used to define how a certain motion ends or to define the size of the turning zone between two motion tracks. In order to facilitate the use of users, the system presets commonly used transition variables (system variables, do not allow users to modify), at the same time can support users in the scope of global, engineering, program three variables to create the variable, delete, modify and other operations.

Parameters	**Data type**	**Meaning of the parameter**

per	real	Percentage of turns. Apply to MovJ,MovL,MovC and other motion commands that indicate how far away you are from the target point to start turning.
dis	real	Cartesian space turning area size. For MovL,MovC and other linear arc motion instructions, define the size of the turning area of the Cartesian space trajectory, that is, when the robot moves to the distance from the target point and dis mm, start to turn to the next target point, the unit is mm.

8.2.6 TOOL

The TOOL type variable is used to record the tool parameters and define the displacement and rotation of the tool end relative to the robot flange.

Parameters	**Parameters**	**Data type**	**Meaning of the parameter**
TOOL Used to record tool parameters and define tool end displacement and rotation relative to robot flange.	x	real	TCP displacement offset in the x direction with respect to the flange coordinate system, in mm.
	y	real	TCP displacement offset with respect to the flange coordinate system in the y direction, in mm.
	z	real	TCP displacement offset in the z direction with respect to the flange coordinate system, in mm.
	a	real	TCP Euler Angle of rotation with respect to the z axis of the flange coordinate system, in deg.
	b	real	TCP Euler Angle of rotation with respect to the y' axis of the flange coordinate system, in deg.
	c	real	TCP Euler Angle of rotation with respect to the x "" axis of the flange coordinate system, in deg.
dyn(LoadDyn) Used to store robot end tools and load quality information parameters.	M	real	Quality information of the tool in kg.
Pos The position of	Mx	real	The offset, in mm, of the center of gravity C of the installed tool or the clamped load in

the installed tool or load on the coordinate system OTool-XYZ.			the X direction of the coordinate system OTool-XYZ.
	My	real	The offset, in mm, of the center of gravity C of the installed tool or the clamped load in the Y direction of the coordinate system OTool-XYZ.
	Mz	real	The offset, in mm, of the center of gravity C of the installed tool or the clamped load in the Z direction of the coordinate system OTool-XYZ.
Tensor The inertia tensor of the installed tool or load as determined by the output coordinate system Otool-XYZ.	Ixx	real	The moment of inertia of the installed tool or clamped load turning in the X direction at the center of gravity, in kg·mm².
	Iyy	real	The moment of inertia of the installed tool or clamped load turning in the Y direction at the center of gravity, in kg·mm².
	Izz	real	The moment of inertia of the installed tool or clamped load turning in the Z direction at the center of gravity, in kg·mm².
	Ixy	real	The product of inertia of the installed tool or clamped load in the XY cross direction at the center of gravity, in kg·mm².
	Ixz	real	The product of inertia of the installed tool or clamped load in the direction XZ crosses at the center of gravity, in kg·mm².
	Iyz	real	The product of inertia of the installed tool or clamped load in the YZ cross direction at the center of gravity, in kg·mm².

8.2.7 USERCOOR

The USERCOOR type variable is used to record the parameters of the user coordinate system and define the displacement and rotation of the user coordinate system with respect to the world coordinate system.

Parameters	**Data type**	**Meaning of the parameter**
x	real	The offset of the origin of the user coordinate system with respect to the world coordinate system in the x direction, in mm.
y	real	The offset of the origin of the user coordinate system with

		respect to the world coordinate system in the x direction, in mm.
z	real	The offset of the origin of the user coordinate system with respect to the world coordinate system in the x direction, in mm.
a	real	Euler Angle of rotation of the user coordinate system with respect to the z-axis of the world coordinate system, in deg.
b	real	Euler Angle of rotation of the user coordinate system with respect to the y' axis of the world coordinate system, in deg.
c	real	Euler Angle of rotation of the user coordinate system with respect to the x "" axis of the world coordinate system, in deg.

8.2.8 PLAYLOAD

The variable PAYLOAD is used to record the load parameters of the workpiece, define the load parameters of the end of the robot and the quality information. The detailed quality information is helpful for the full model calculation of the robot dynamics, so as to improve the motion beat.

Parameters	**Parameters**	**Data type**	**Meaning of the parameter**
dyn(LoadDyn) Used to store robot end tools and load quality information parameters.	M	real	Quality information for the tool.
CenterPos The position of the installed tool or load on the coordinate system OTool-XYZ.	Mx	real	The offset, in mm, of the center of gravity C of the installed tool or the clamped load in the X direction of the coordinate system OTool-XYZ.
	My	real	The offset, in mm, of the center of gravity C of the installed tool or the clamped load in the Y direction of the coordinate system OTool-XYZ.
	Mz	real	The offset, in mm, of the center of gravity C of the installed tool or the clamped load in the Z direction of the coordinate system

			OTool-XYZ.
InertiaTensor Inertiatensor The inertia tensor of the installed tool or load determined by the output coordinate system Otool-XYZ.	Ixx	real	The moment of inertia of the installed tool or clamped load turning in the X direction at the center of gravity, in kg·mm².
	Iyy	real	The moment of inertia of the installed tool or clamped load turning in the Y direction at the center of gravity, in kg·mm².
	Izz	real	The moment of inertia of the installed tool or clamped load turning in the Z direction at the center of gravity, in kg·mm².
	Ixy	real	The product of inertia of the installed tool or clamped load in the XY cross direction at the center of gravity, in kg·mm².
	Ixz	real	The product of inertia of the installed tool or clamped load in the direction XZ crosses at the center of gravity, in kg·mm².
	Iyz	real	The product of inertia of the installed tool or clamped load in the YZ cross direction at the center of gravity, in kg·mm².

8.2.9 CLOCK

CLOCK stores the value of the clock information.

Arguments	Data type	Meaning of the parameter
state	bool	Enable status of clock variable.
value	int	The count value of the clock variable.

8.2.10 LsScale

The LsScale variable is used to record the gain ratio threshold parameter of each joint axis, and is used to improve the low-speed jitter phenomenon of the robot in a certain speed interval, and is used together with the speed interval threshold parameter. Setting range: [100, 1000], unit: %. This variable can only be created and modified in the global area.

Parameters	Data type	Meaning of the parameter
J1	int	Proportional gain threshold corresponding to the J1 axis, in %

J2	int	Proportional gain threshold corresponding to the J2 axis, in %
J3	int	Proportional gain threshold corresponding to the J3 axis, in %
J4	int	Proportional gain threshold corresponding to the J4 axis, in %
J5	int	Proportional gain threshold corresponding to the J5 axis, in %
J6	int	Proportional gain threshold corresponding to the J6 axis, in %

8.2.11 LsThresh

The LsThresh type variable is used to record the threshold parameters of the speed interval of each joint axis, to improve the low-speed jitter phenomenon of the robot in a certain speed interval, and is used together with the gain proportion threshold parameter. Set range: [10, 1000], unit: r/min. This variable can be created and modified only in the global domain.

argument	**Data type**	**Meaning of the parameter**
J1	int	Speed interval threshold corresponding to the J1 axis, in r/min
J2	int	Speed interval threshold corresponding to the J2 axis, in r/min
J3	int	Speed interval threshold corresponding to the J3 axis, in r/min
J4	int	Speed interval threshold corresponding to the J4 axis, in r/min
J5	int	Speed interval threshold corresponding to the J5 axis, in r/min
J6	int	Speed interval threshold corresponding to the J6 axis, in r/min

8.2.12 VibrationSuppression

VibrationSuppression Indicates vibration suppression parameters.

Parameters	**Data type**	**Meaning of the parameter**

Frequency X	real	Natural frequency of vibration in the X direction
Frequency Y	real	Natural frequency of vibration in the Y direction
Frequency Z	real	Natural frequency of vibration in the Z direction
Damping Ratio X	real	Damping ratio in the X direction
Damping Ratio Y	real	Damping ratio in the Y direction
Damping Ratio Z	real	Damping ratio in the Z direction

8.2.13 Matrix2

Matrix2-type variables are used to record two-point arrays.

Parameters	Data type	Meaning of the parameter
Matrix2 Name	string	Array name

8.2.14 Matrix3

Matrix2-type variables are used to record two-point arrays.

argument	Data type	Meaning of the parameter
Matrix3 Name	string	Array name

8.2.15 Matrix4

Matrix2-type variables are used to record two-point arrays.

Parameters	Data type	Meaning of the parameter
Matrix4 name	string	Array name

8.2.16 Matrix9

Matrix2-type variables are used to record two-point arrays.

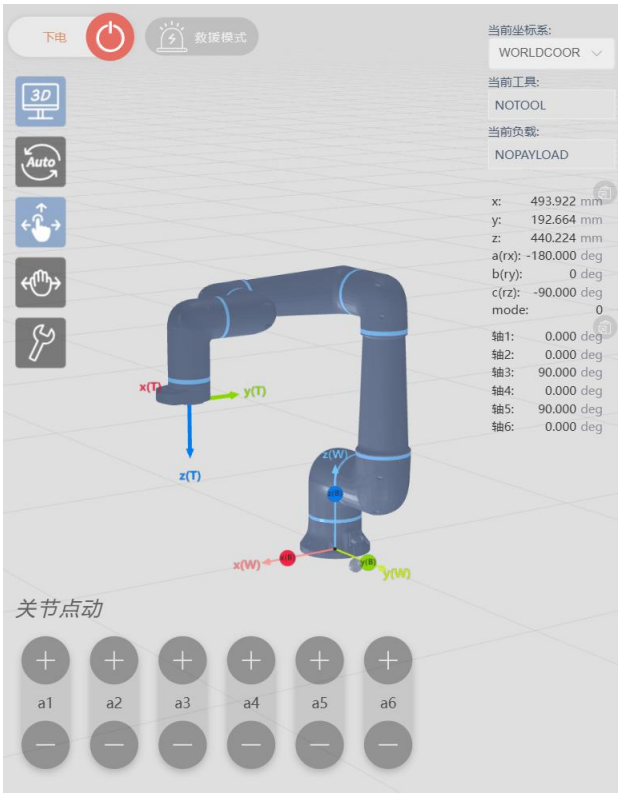
argument	Data type	Meaning of the parameter
Matrix9 name	string	Array name

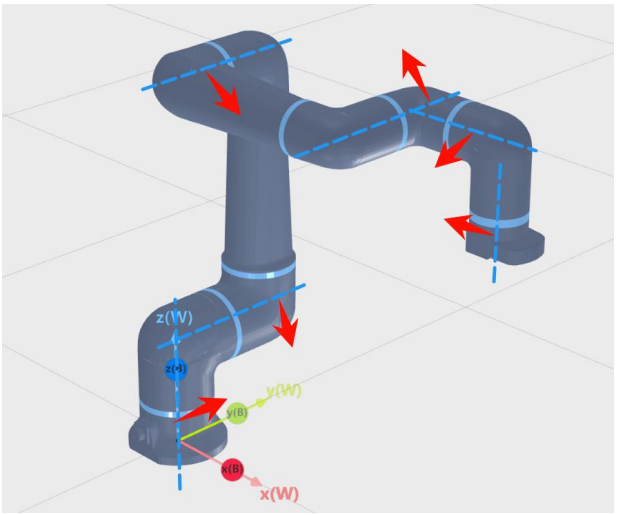
9. Coordinate system and variable calibration

This chapter will describe joint coordinate system, world coordinate system, user coordinate system, tool coordinate system and their use;

9.1 Joint coordinate system

Joint coordinate system or joint space, the independent motion of robot joints, is joint motion.

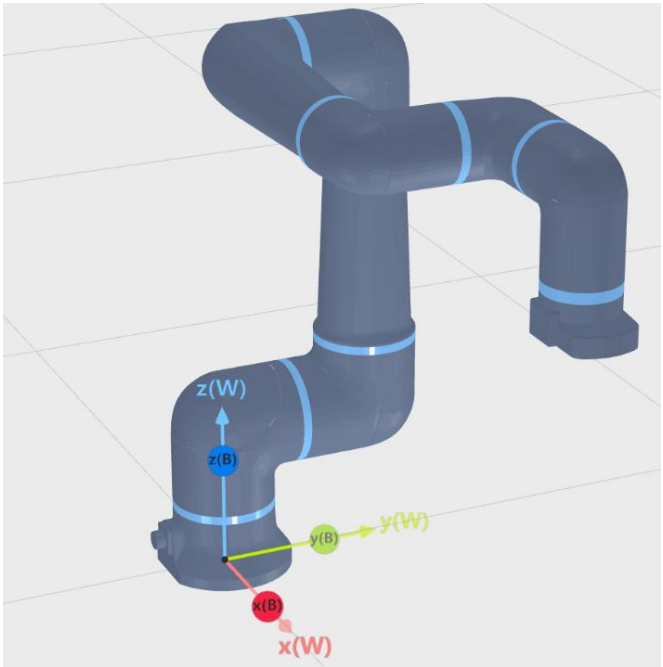




9.2 World coordinate system

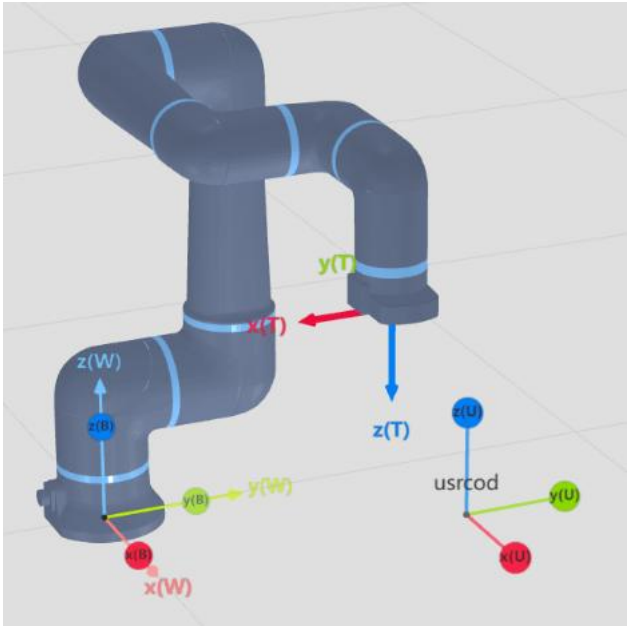
The Cartesian coordinate system of Kuzo cooperative robot is the right hand coordinate system, and its Euler Angle format is X-Y-Z fixed Angle. For example, the pose [900mm, 200mm, 1200mm, 20°, 30°, 45°] is interpreted as moving to x=900mm, y=200mm, z=1200mm of the reference coordinate system, taking the end TCP point as the rotation center, and first rotating the end 20° along the X-axis of the reference coordinate system. Then rotate the end 30° along the Y axis of the world coordinate system, and then rotate the end 45° along the Z axis of the world coordinate system.

When leaving the factory, the robot is located at the position [0, 0, 0, 0, 0, 0] of the world coordinate system by default, that is, the robot base coordinate system coincides with the pose of the world coordinate system. The navigation insert of the robot base points to the negative direction of the Y-axis of the robot base coordinate system, and the positive direction of the Z-axis points to the inside of the base.



The installation of the robot can be selected from the preset installation mode or customize its installation offset and installation rotation with respect to the world coordinate system.

9.3 User coordinate system and Calibration (USERCOOR)



The user can create a new user coordinate system variable. The user coordinate system is offset based on the world coordinate system. The offset value can be directly input by the user or assisted calibration.

When you need to calibrate the user coordinate system, you can use the **"three-point**

calibration method" to help you create.

You can start the calibration of the user coordinate system by:

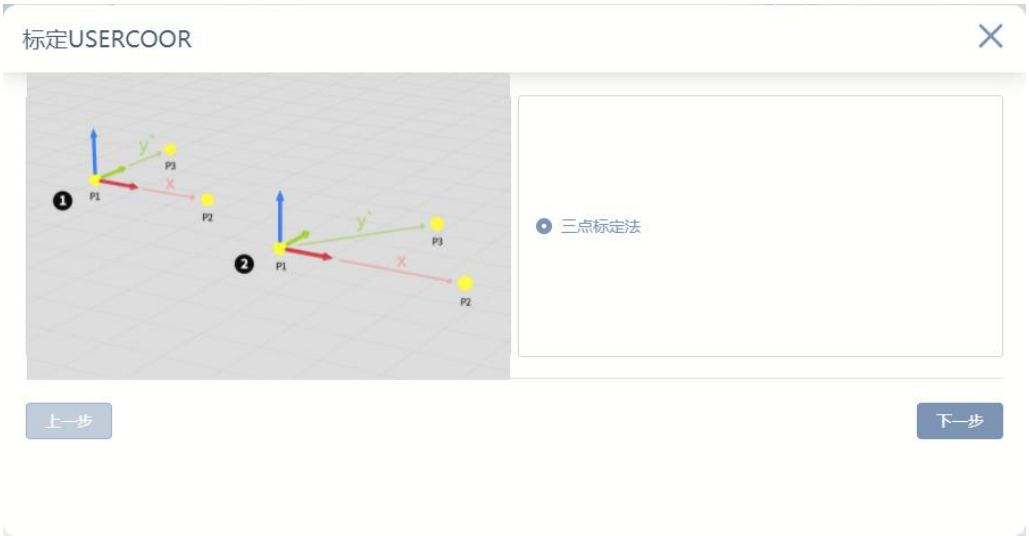
1. Click to enter the variable management interface 

2. Click to pop up the Variable management window 

3. Define the variable type as USERCOOR through the drop down menu
4. Name the defined variable of type USERCOOR
5. Click the button to enter the calibration guide and complete the calibration of the user coordinate system according to the guidance window 

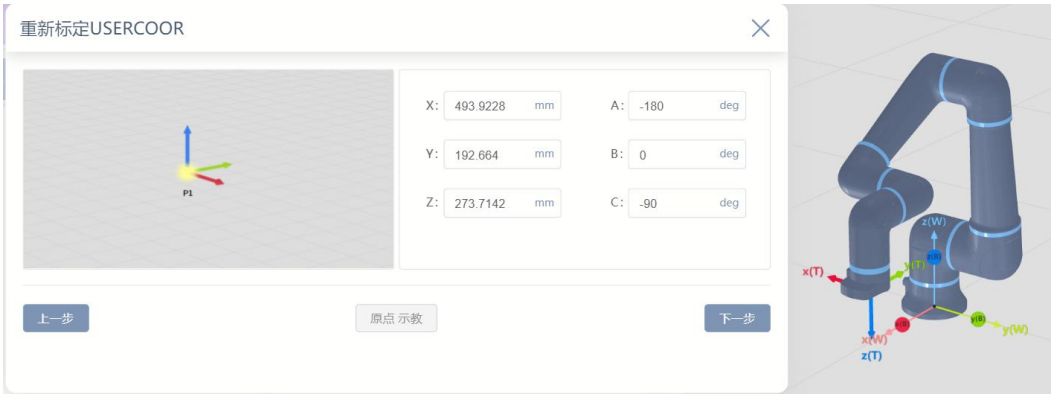
6. Click the button after completing the calibration of user coordinate system. 

9.3.1 Three-point calibration method



Define the origin, x+ axis direction, and y+ axis direction. The plane is defined using the right hand rule, so the z+ axis is the cross product of the x+ and y+ axes.

9.3.1.1 Start the calibration

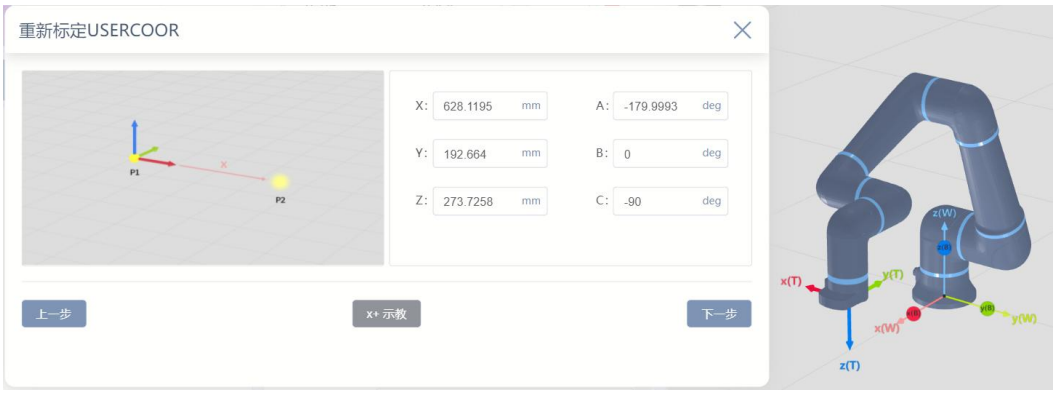


1. Define the user coordinate system origin

   1) Point the robot to define the origin pose

   2) Click the button to show that the current pose is P1 (origin) 方向 1 示教

   3) Click the button to enter the next guide interface 下一步

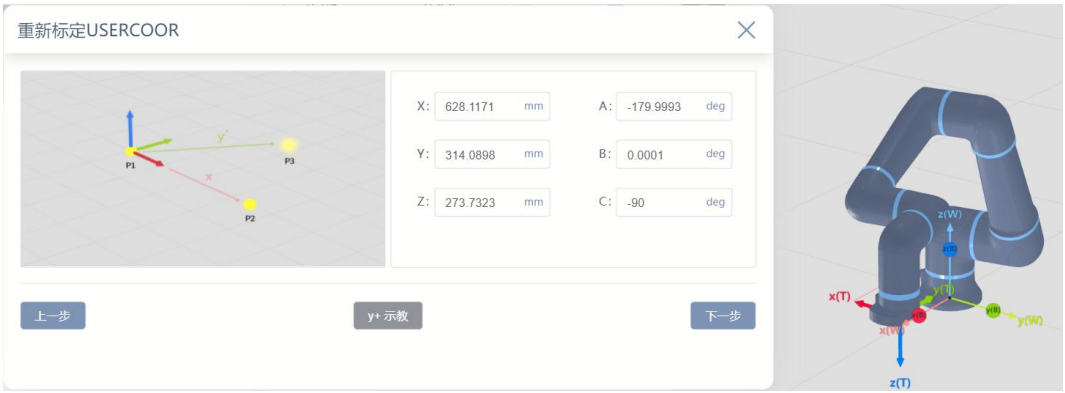


2. Define the user coordinate system x+ orientation

   4) Click the robot to define the user coordinate x+ direction

   5) Click the button to show that the current point position is x+ direction 方向 2 示教

   6) Click the button to enter the next guide interface 下一步



3. Define the user coordinate system y+ orientation

7) Click the robot to define the user coordinate y+ direction

8) Click the button to show that the current point is y+ direction 

9) Click the button to complete the calibration of the user coordinate system 

9.3.1.2 Successful calibration

The values of x,y and z of the user coordinate system will be automatically filled into the "Calibration USERCOOR" window, click the button to complete the calibration guide, and then choose whether to save the variable. 

9.3.1.3 Calibration failure

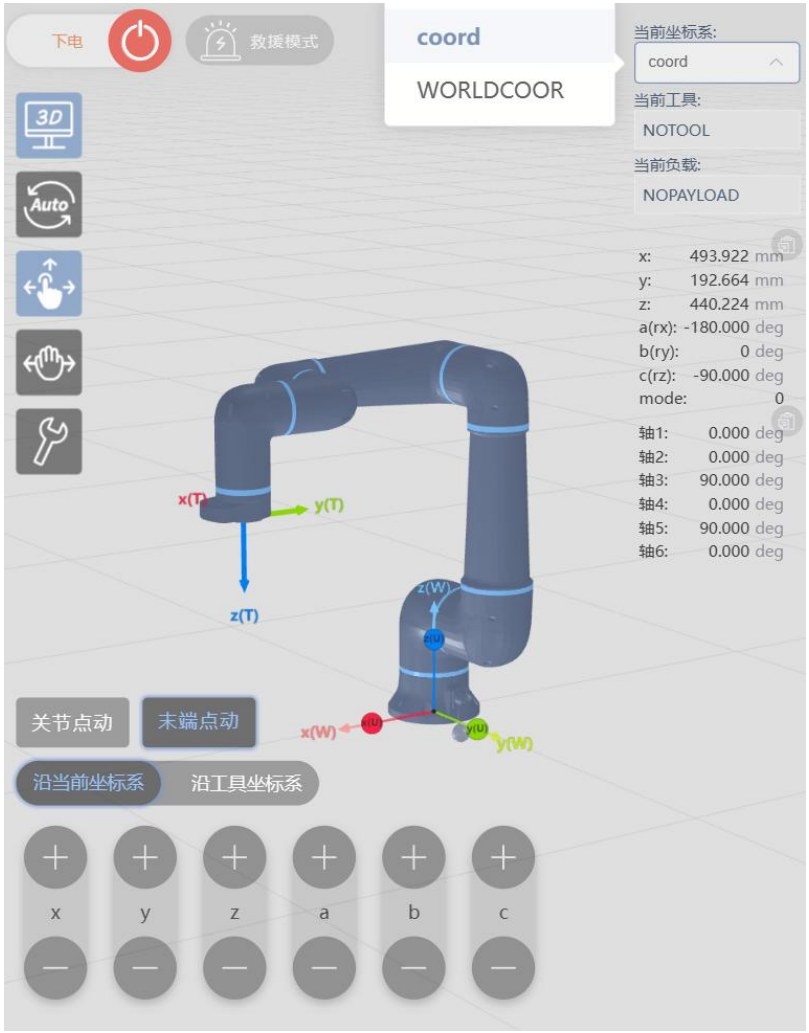
If the "Calibrate USERCOOR" window does not automatically fill in any values and is accompanied by a "user coordinate calibration failed" message, the calibration failed.

Please restart the calibration and note the following principles:

- Calibrate the defined origin, and avoid 2 or more duplicate points in the x+ and y+ three points.

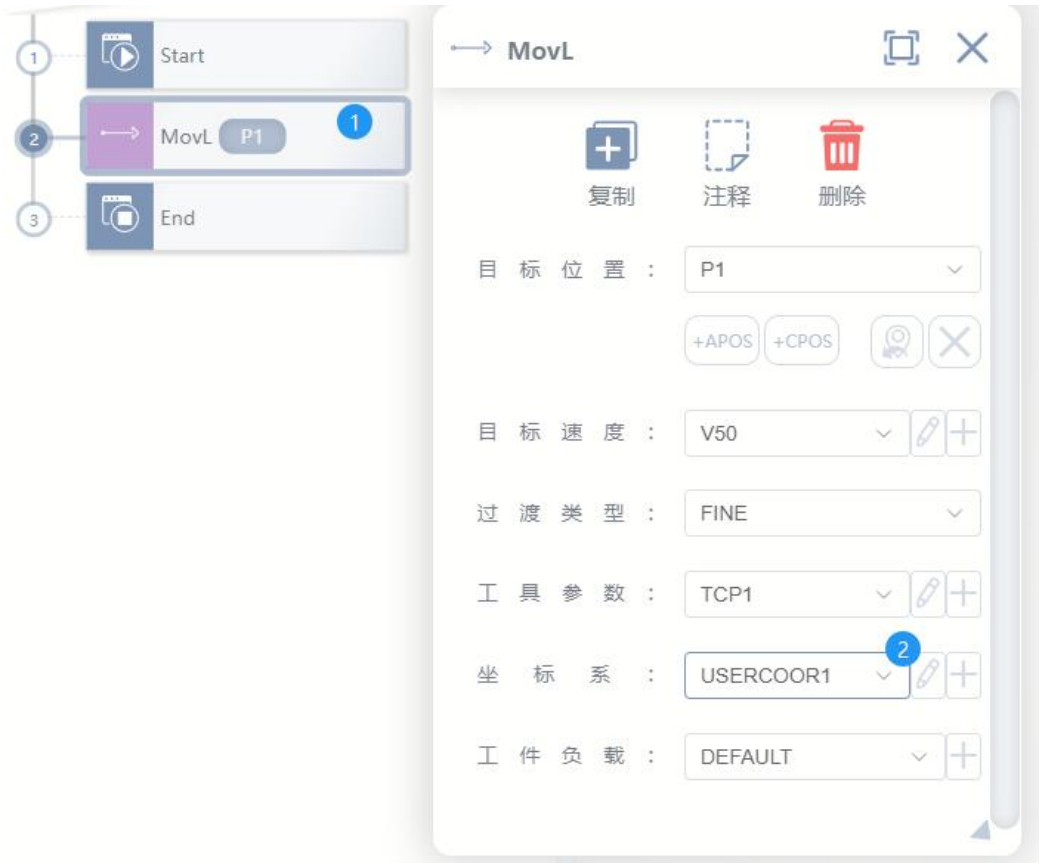
9.3.2 Use the user coordinate system

9.3.2.1 Use a user coordinate system when clicking



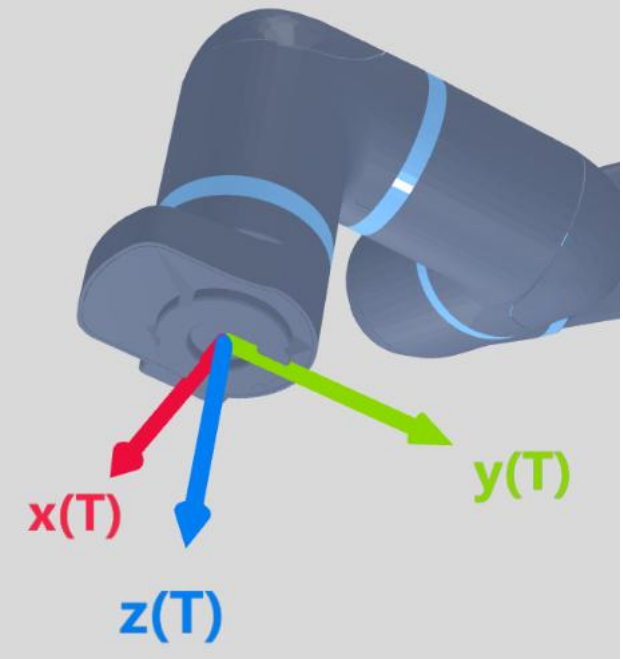
When the end point robot can choose to move along the current coordinate system, the current coordinate system can choose to move along the user coordinate system.

9.3.2.2 The user coordinate system is used in the program



1. Click on the Mov instruction node in the program tree
2. Select Use defined user coordinate system from the drop-down menu of "Coordinate System"

9.4 Tools and Calibration (TOOL)



The user can create a new tool variable, and the tool coordinate system is offset based on the default tool coordinate system (NOTOOL) at the end of the flange. The offset value can be directly input by the user or assisted calibration. The origin of the default tool coordinate system is located at the center of the end of the flange, the Z axis points out of the flange, and the Y axis points to the installation positioning pin hole.

When the tool coordinate system needs **to be calibrated, the "four-way calibration method**" and "**one-point calibration method (attitude)**" can be used to obtain the translation and rotation of TCP (tool center point) relative to the center of the tool output flange.

You can start the calibration of the tool coordinate system by:

1. Click to enter the variable management interface 

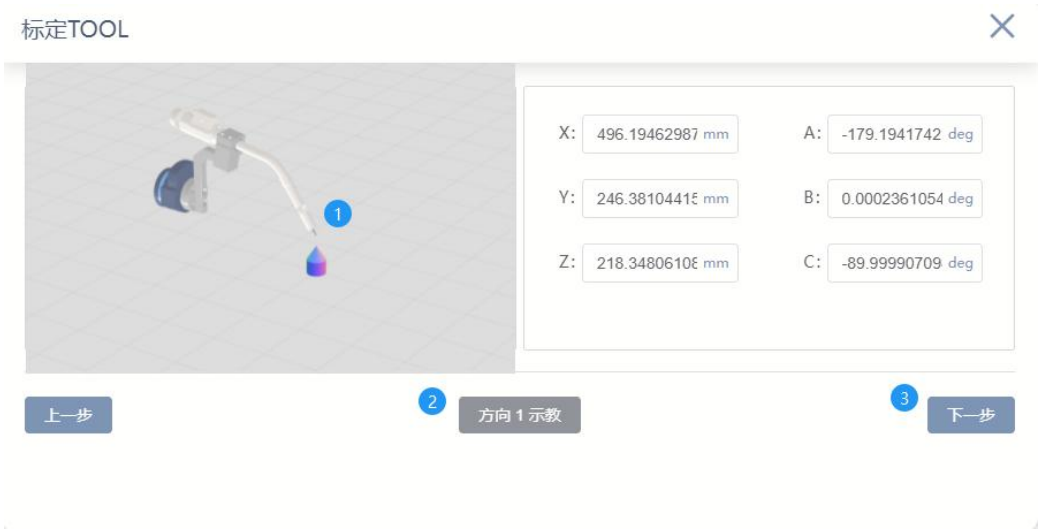
2. Click to pop up the Variable management window 

3. Define the variable type as TOOL through the drop down menu
4. Name the defined TOOL type variable
5. Click the button to enter the calibration guide, complete the calibration of TCP pose according to the boot window 

6. Click the button after completing the calibration of the tool coordinate system. 

9.4.1 Four way calibration method

Move the robot (click or drag) to four different positions, each time touching the tool point with the same needle tip placed in the space and clicking the "Direction Show" button. The four positions are performed to obtain the TCP translation relationship with respect to the center of the tool output flange.



9.4.1.1 Start calibration

1. The mobile robot makes the TCP (tool center point) contact with the tip of the needle placed in space

2. Click the Teach button to record the actual pose of the current robot. 

3. Click the button to repeat step 1 and Step 2 until the fourth point position, then click the button to complete the direction teaching. 

9.4.1.2 Successful calibration

The values of x,y and z of the successfully calibrated TOOL coordinate system will be automatically filled into the "Calibration Tool" window. Click the button to complete the calibration guide. At this time, you can choose to continue the calibration attitude 

(one-point calibration method) or save the variable.

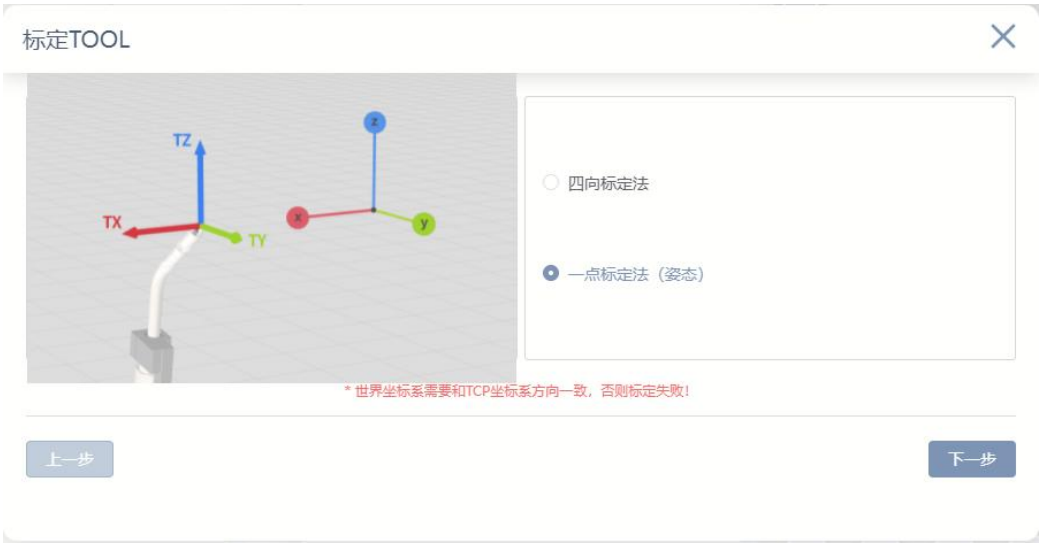
9.4.2 Calibration failure

If the Calibration TOOL window does not automatically fill in any values and is accompanied by a "4-point calibration failure" prompt, the calibration failed.

Start calibrating again, paying attention to the following principles:

- Four pose variations are sufficient.
- Tip to tip (the center point of the tool is enough to touch the tip in the space).

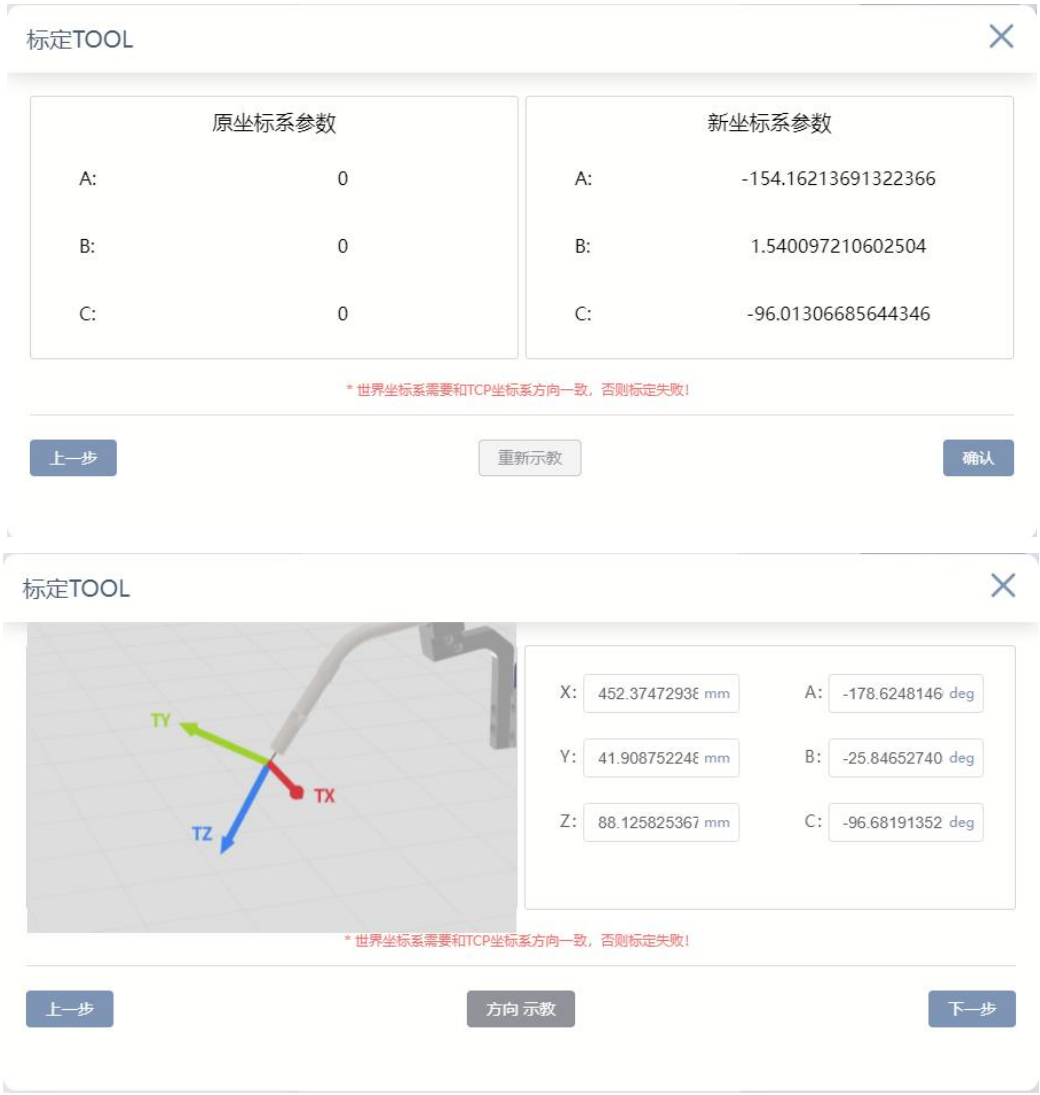
9.4.3 One point calibration method (attitude)



After completing the four-way calibration method (to get the translation relationship of TCP with respect to the center of the tool output flange), you can start some calibration method (attitude) to get the rotation relationship of TCP with respect to the center of the tool output flange.

Adjust the target TCP attitude to align with the world coordinate system and then click and confirm to calibrate the TCP attitude. 

If it is re-calibrated, the variable needs to be saved before it can take effect.



9.4.3.1 Start calibrating

The mobile robot makes the desired tool coordinate system direction coincide with the robot's world coordinate system direction, and clicks the button to complete the direction teaching. 

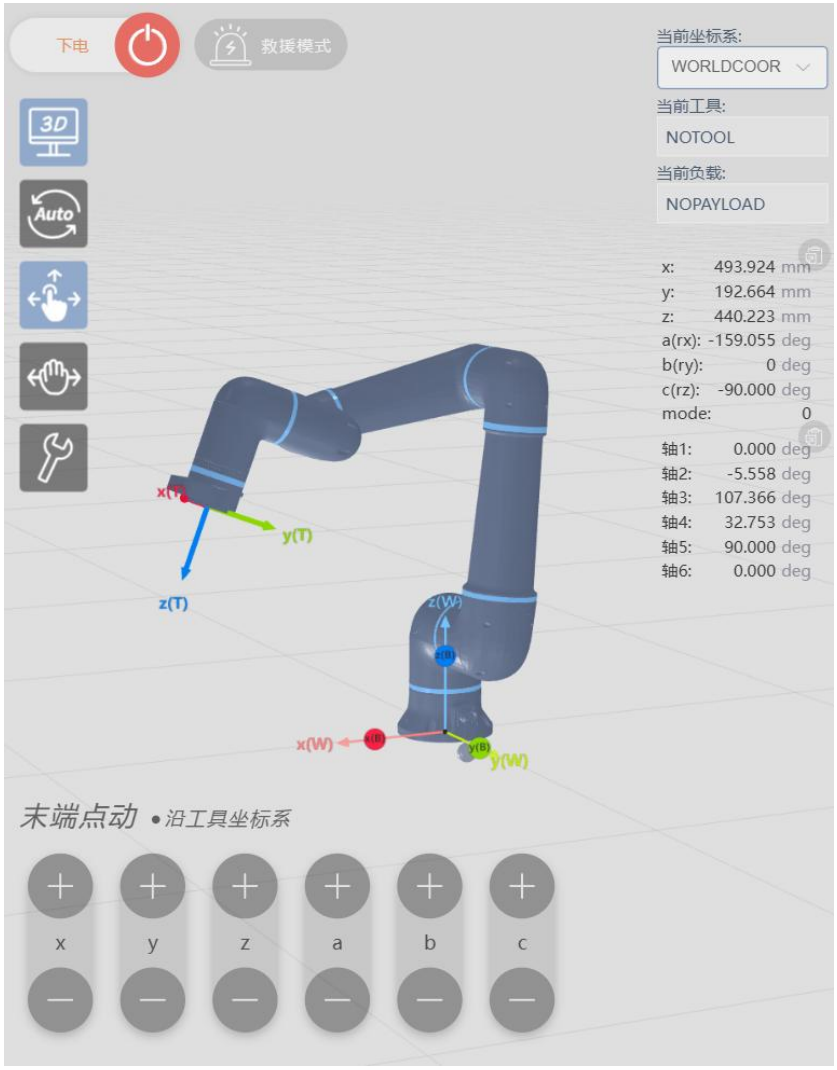
9.4.3.2 Calibration result

When calibrating the attitude, the robot can not verify whether it is accurate, which can be judged by the user by moving the tool coordinate system visually.

After completing the "four-way calibration method" and "one-point calibration method (attitude)", a complete tool coordinate system is obtained after the translation and rotation of TCP (tool center point) relative to the center of the tool output flange.

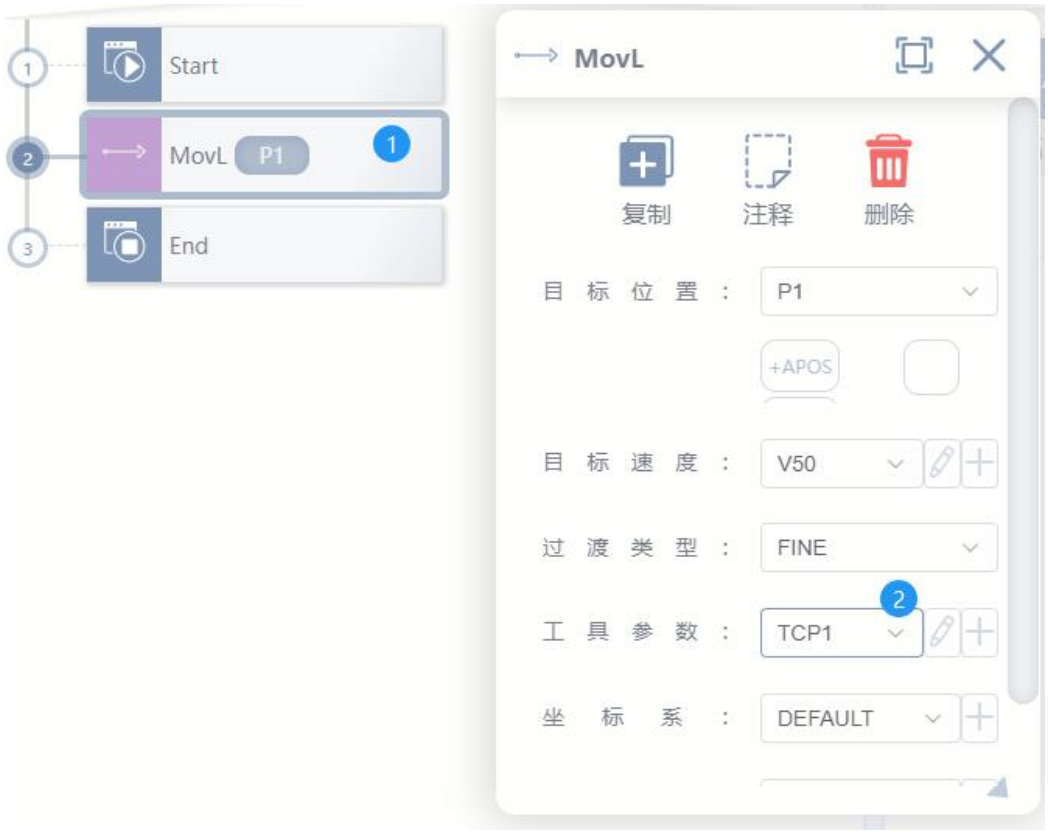
9.4.4 Using the tool coordinate system

9.4.4.1 Use the tool coordinate system when clicking



When the end point robot can be selected along the tool coordinate system, the current tool can be selected as the target tool coordinate system to move along the tool coordinate system. The current tool can be switched in the Settings TAB - Machinery.

9.4.4.2 Use tools in program



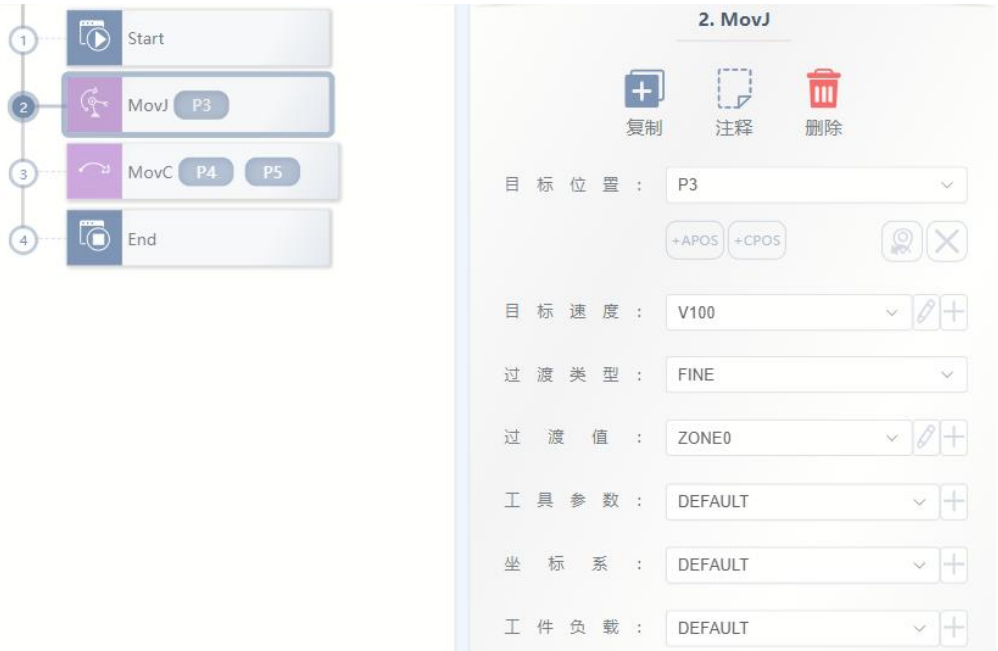
1. Click on the Mov instruction node in the program tree
2. Select the coordinate system to use the tool in the drop-down menu of "Tool Parameters"

10. Instruction Introduction

10.1 Move motion instruction

10.1.1 MovJ

This command indicates that each joint of the robot moves from point to point, and the end trajectory of the robot is an irregular curve. Double-click the added MoveJ command or select parameters in the programming command details area and click MoveJ to configure the command parameters.



argument	Instructions
Target location	You can select the point that has been shown in the target location option, only APOS and CPOS can be added.
Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is a percentage. See the Variables section for details on how to create and set the SPEED type variable.
Transition	The way the robot transitions as it approaches the end point. FINE: No transition.

Types	RELATIVE: Relative transition.
Transition value	The transition value as the robot approaches the end point. This parameter can be set when RELATIVE vs. ABSOLUTE type variables are selected. ZONE variables can be predefined or created by the system. Use ZONE type variables, see the section on Variables.
Tool parameters	The parameters of the tool used by the robot to execute the trajectory. Set to "DEFAULT", if the instruction was not preceded by a SetTool instruction, to use the options in the Setup - Machine - Default tool, or if the instruction was changed before using a SetTool instruction, the value changed by the most recent SetTool instruction above the current instruction. If it is set to the TOOL variable, you can select the variable predefined by the system or create it by yourself. When the coordinate system is different from the current tool parameter, the system will switch the tool parameter to the set tool parameter. If the tool parameters change between this segment and the preceding and following segment tracks, transitions are not supported. TOOL type variable Settings, see the Variables section.
Coordinate System	The coordinate system parameters used by the robot to execute this trajectory. Set to "DEFAULT", if no SetCoord instruction is used before the instruction, the value of the Default coordinate system is the same as the value of the world coordinate system (WORLDCOOR), if the instruction is changed by the SetCoord instruction before the instruction, it is the value changed by the last SetCoord instruction above the current instruction. When it is set to USERCOOR type variable, it can be predefined by the system or created by itself; When a coordinate system is different from the current coordinate system, the system will switch the coordinate system to the set coordinate system. USERCOOR type variable Settings, see the Variables section.
Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction.

	When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload parameter. If the workpiece load changes between this segment and the front and back segment trajectories, transitions are not supported. For details about the PAYLOAD variable Settings, see the Variables section.

10.1.2 MovL

The MovL command is a linear motion command, through which the TCP point of the robot can move in a straight line to the target position at a set speed. If the start and stop posture of the movement are different, the posture will rotate synchronously with the position to the end position during the operation. Compared with the joint motion, the linear movement may pass through the singularity. Double-click the added MoveL instruction or select the parameter in the Programming instruction details area and click MoveL to configure the instruction parameters.

argument	Instructions
Target location	You can select the taught point in the target location option. Only APOS and CPOS can be added. DAPOS and DCPOS are not selectable.

Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is absolute value, unit mm/s. For details about how to create and set the SPEED type variable, see the Variables section.
Transition Types	The way the robot transitions as it approaches the end point. FINE: No transition. RELATIVE: Relative transition. ABSOLUTE: Absolute transition.
Transitional value	The transition value as the robot approaches the end point. This parameter can be set when RELATIVE vs. ABSOLUTE type variables are selected. ZONE variables can be predefined or created by the system. For details about how to use ZONE variables, see the section on Variables.
Tool parameters	The tool parameters used by the robot to execute this trajectory. Set to "DEFAULT", if the instruction was not preceded by a SetTool instruction, to use the options in the Setup - Machine - Default tool, or if the instruction was changed before using a SetTool instruction, the value changed by the most recent SetTool instruction above the current instruction. If it is set to the TOOL variable, you can select the variable predefined by the system or create it by yourself. When the coordinate system is different from the current tool parameter, the system will switch the tool parameter to the set tool parameter. If the tool parameters change between this segment and the preceding and following segment tracks, transitions are not supported. TOOL type variable Settings, see the Variables section.
Coordinate System	The coordinate system parameters used by the robot to execute this trajectory. Set to "DEFAULT", if no SetCoord instruction is used before the instruction, the value of the Default coordinate system is the same as the value of the world coordinate system (WORLDCOOR), if the instruction is changed by the SetCoord instruction before the instruction, it is the value changed by the last SetCoord instruction above the current instruction. When it is set to USERCOOR type variable, it can be predefined by the system or created by itself; When a coordinate system is different from the current coordinate system, the system will switch the coordinate system to the set coordinate system.

	USERCOOR type variable Settings, see the Variables section.
Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction. When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload parameter. If the workpiece load changes between this segment and the front and back segment trajectories, transitions are not supported. For details about the PAYLOAD variable Settings, see the Variables section.

10.1.3 MovC

The arc instruction must contain three poses, which must not be on a uniform line in the space of the three poses. When using this command, the TCP point of the robot moves from the starting position to the target position through the middle position, and the starting position is the end point of the previous moving command. When the MoveC command is used, if the starting and ending posture of the movement is different, the posture will rotate synchronously with the position to the end position during the operation, but it may not necessarily pass through the middle position. In contrast to joint motion, arc movement may pass through singularities. Double-click the added MoveC instruction or select parameters in the programming instruction details area and click MoveC to configure the instruction parameters.

Parameters	**Instructions**
Middle position	Arc intermediate auxiliary point position, type can only be APOS or CPOS.
Target position	Arc end position, type can only be APOS or CPOS.
Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is absolute value, unit mm/s. For details about how to create and set the SPEED type variable, see the Variables section.
Transition Types	The way the robot transitions as it approaches the end point. FINE: No transition. RELATIVE: Relative transition. ABSOLUTE: Absolute transition.
Transient value	The transition value as the robot approaches the end point. This parameter can be set when RELATIVE vs. ABSOLUTE type variables are selected. ZONE variables can be predefined or created by the system. The

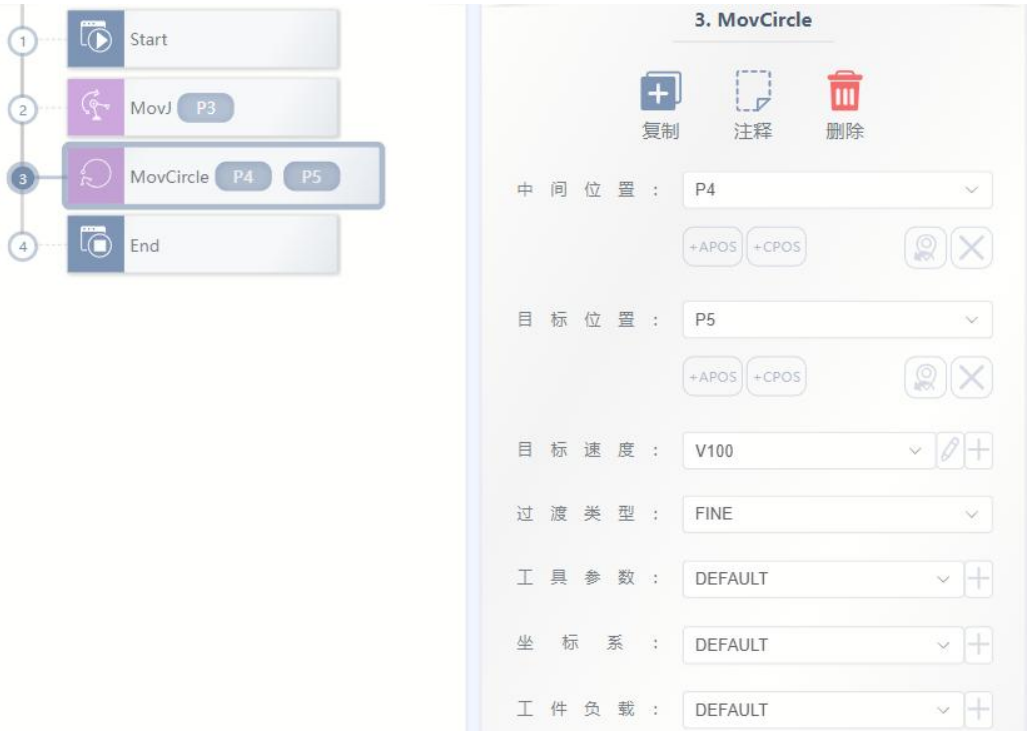
	larger the transition value, the larger the transition radius. ZONE type variables are used, see the Variables section for details.
Tool parameters	The tool parameters used by the robot to execute this trajectory. Set to "DEFAULT", if the instruction was not preceded by a SetTool instruction, to use the options in the Setup - Machine - Default tool, or if the instruction was changed before using a SetTool instruction, the value changed by the most recent SetTool instruction above the current instruction. If the variable is set to TOOL, it can be predefined or created by the system. When the coordinate system is different from the current tool parameter, the system will switch the tool parameter to the set tool parameter. If the tool parameters change between this segment and the preceding and following segment tracks, transitions are not supported. TOOL type variable Settings, see the Variables section.
Coordinate System	The coordinate system parameters used by the robot to execute this trajectory. Set to "DEFAULT", if no SetCoord instruction is used before the instruction, the value of the Default coordinate system is the same as the value of the world coordinate system (WORLDCOOR), if the instruction is changed by the SetCoord instruction before the instruction, it is the value changed by the last SetCoord instruction above the current instruction. When it is set to USERCOOR type variable, it can be predefined by the system or created by itself; When a coordinate system is different from the current coordinate system, the system will switch the coordinate system to the set coordinate system. USERCOOR type variable Settings, see the Variables section.
Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction. When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload parameter. If the workpiece load changes between this segment and the front and back segment trajectories, transitions are not supported. For details about the PAYLOAD variable Settings, see the Variables

	section.

10.1.4 MovCircle

The full circle command refers to the robot TCP point from the starting position, and the positions in these three pose Spaces must not be on a uniform line. When using this command, the robot TCP point moves from the starting position, through the middle position to the target position in a full circle, and the posture remains unchanged during the whole circle movement. Compared with the joint motion, the circular motion may pass through the singularity.

Double-click the added MovCircle instruction or select the parameter in the programming instruction details area and click MoveCircle to configure the instruction parameters.



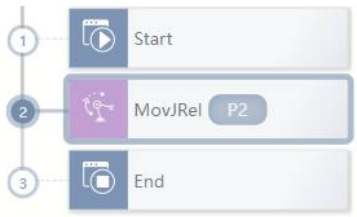
Parameters	**Parameters**
Intermediate position	Arc intermediate auxiliary point position, type can be APOS or CPOS.
Target position	Arc end position, type can be APOS or CPOS.
Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is absolute

	value, unit mm/s. For details about how to create and set the SPEED type variable, see the Variables section.
Transition Types	The way the robot transitions as it approaches the end point. FINE: No transition. RELATIVE: Relative transition. ABSOLUTE: Absolute transition.
Transitional value	The transition value as the robot approaches the end point. This parameter can be set when RELATIVE vs. ABSOLUTE type variables are selected. ZONE variables can be predefined or created by the system. The larger the transition value, the larger the transition radius. ZONE type variables are used, see the Variables section for details.
Tool parameter	The tool parameters used by the robot to execute this trajectory. Set to "DEFAULT", if the instruction was not preceded by a SetTool instruction, to use the options in the Setup - Machine - Default tool, or if the instruction was changed before using a SetTool instruction, the value changed by the most recent SetTool instruction above the current instruction. If it is set to the TOOL variable, you can select the variable predefined by the system or create it by yourself. When the coordinate system is different from the current tool parameter, the system will switch the tool parameter to the set tool parameter. If the tool parameters change between this segment and the preceding and following segment tracks, transitions are not supported. TOOL type variable Settings, see the Variables section.
Coordinate System	The coordinate system parameters used by the robot to execute this trajectory. Set to "DEFAULT", if no SetCoord instruction is used before the instruction, the value of the Default coordinate system is the same as the value of the world coordinate system (WORLDCOOR), if the instruction is changed by the SetCoord instruction before the instruction, it is the value changed by the last SetCoord instruction above the current instruction. When it is set to USERCOOR type variable, it can be predefined by the system or created by itself; When a coordinate system is different from the current coordinate system, the system will switch the coordinate system to the set coordinate system. USERCOOR type variable Settings, see the Variables section.

Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction. When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload parameter. If the workpiece load changes between this segment and the front and back segment trajectories, transitions are not supported. For details about the PAYLOAD variable Settings, see the Variables section.

10.1.5 MovJRel

MovJRel is the interpolation relative offset instruction. This command always takes the current robot position or the target position of the previous motion instruction as the starting position, and then the relative movement displacement of the robot is offset.

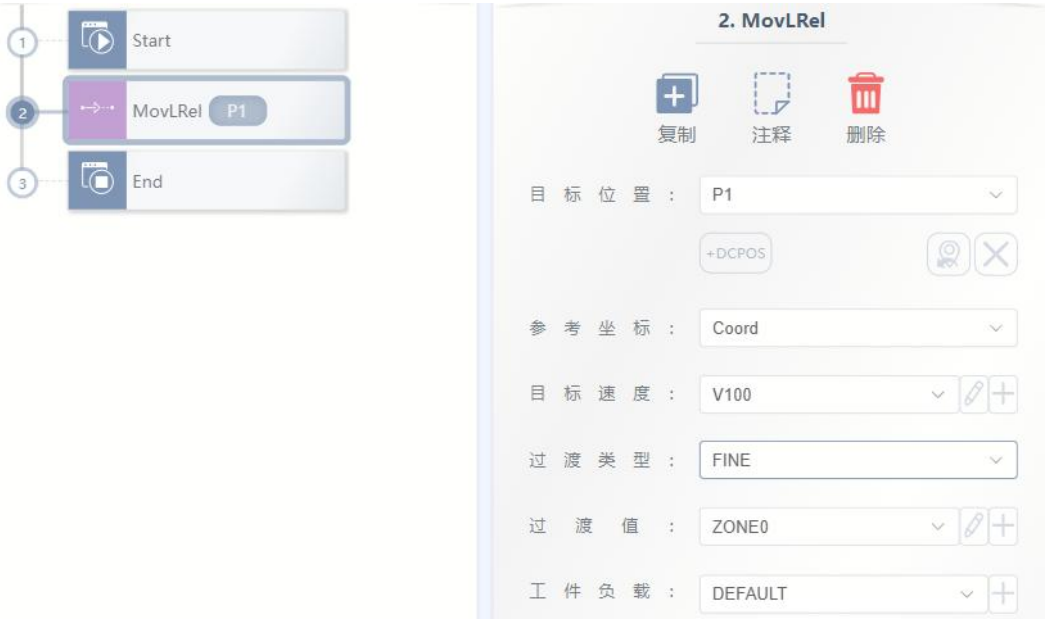


Parameters	Instructions
Target relative position	The increment of the position the robot is to move when executing this instruction.

Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is a percentage. See the Variables section for details on how to create and set the SPEED type variable.
Transition Types	The way the robot transitions as it approaches the end point. FINE: No transition. RELATIVE: Relative transition.
Transition value	The transition value as the robot approaches the end point. ZONE type variable, which can be predefined by the system or created by itself; The larger the transition value, the larger the transition radius. For details about how to use ZONE variables, see the section on Variables.
Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction. When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload For details about the PAYLOAD variable Settings, see the Variables section.

10.1.6 MovLRel

MovLRel interpolates relative offset instruction. This command always takes the current robot position or the target position of the previous motion instruction as the starting position, and then the robot moves relative to the coordinate system or tool.

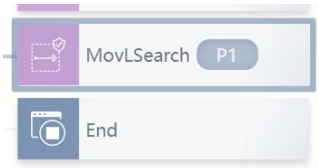


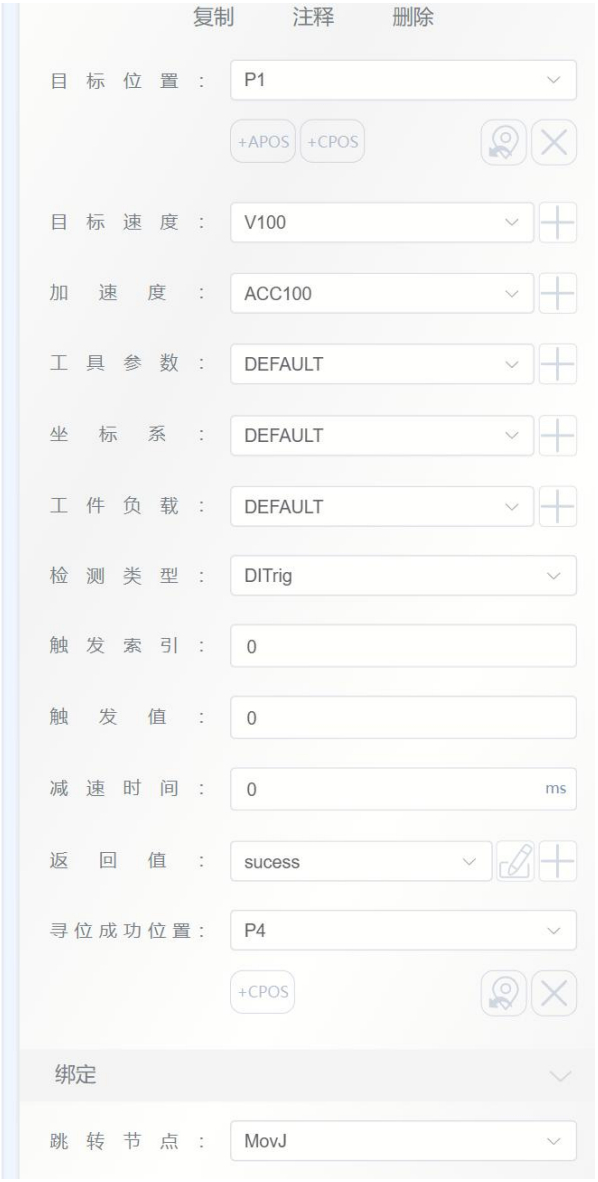
Parameters	Instructions
Target location	The increment of the position the robot is to move when executing this command.
Reference coordinates	Coordinate offset or tool offset selection; – Coord: offset from the current user's coordinate system; – Tool: offset from the tool coordinate system, that is, Tx, Ty, and Tz translate or rotate.
Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is absolute value, unit mm/s. For details about how to create and set the SPEED type variable, see the Variables section.
Transition Types	The way the robot transitions as it approaches the end point. FINE: No transition. RELATIVE: Relative transition.
Transition value	The transition value as the robot approaches the end point. ZONE type variable, which can be predefined by the system or created by itself; The larger the transition value, the larger the transition radius. ZONE type variables are used, see the Variables section for details.
Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before

	this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction. When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload. For details about the PAYLOAD variable, see the Variables section.

10.1.7 MovLSearch

The movlsearch instruction refers to the IO detection or moment detection when executing this MovL instruction.





Parameters	Instructions
Target location	You can select the taught point in the target location option. Only APOS and CPOS can be added. DAPOS and DCPOS are not selectable.
Target speed	Set to SPEED type variable, you can choose the system predefined value, or you can create your own; Where, the target speed is absolute value, unit mm/s. For details about how to create and set the SPEED type variable, see the Variables section.
Acceleration	Set to ACC type variable.
Tool	The tool parameters used by the robot to execute this trajectory. Set to "DEFAULT", if the instruction was not preceded by a SetTool

parameters	instruction, to use the options in the Setup - Machine - Default tool, or if the instruction was changed before using a SetTool instruction, the value changed by the most recent SetTool instruction above the current instruction. If it is set to the TOOL variable, you can select the variable predefined by the system or create it by yourself. When the coordinate system is different from the current tool parameter, the system will switch the tool parameter to the set tool parameter. If the tool parameters change between this segment and the preceding and following segment tracks, transitions are not supported. TOOL type variable Settings, see the Variables section.
Coordinate System	The coordinate system parameters used by the robot to execute this trajectory. Set to "DEFAULT", if no SetCoord instruction is used before the instruction, the value of the Default coordinate system is the same as the value of the world coordinate system (WORLDCOOR), if the instruction is changed by the SetCoord instruction before the instruction, it is the value changed by the last SetCoord instruction above the current instruction. When it is set to USERCOOR type variable, it can be predefined by the system or created by itself; When a coordinate system is different from the current coordinate system, the system will switch the coordinate system to the set coordinate system. USERCOOR type variable Settings, see the Variables section.
Job Load	The workpiece load parameters used when the robot performs this trajectory. Set to "DEFAULT", if the SetPaylod instruction was not used before this instruction, it means the option in Setup - Mechanical - Default load is used, if the instruction was changed before the SetPaylod instruction, it is the value changed by the most recent SetPaylod instruction above the current instruction. When it is set to the PAYLOAD, you can choose the variable predefined by the system or create it by yourself. When the payload parameter is different from the current payload parameter, the system will switch the payload parameter to the set payload parameter. If the workpiece load changes between this segment and the front and back segment trajectories, transitions are not supported. For details about the PAYLOAD variable Settings, see the Variables section.
Detection	DITrig: Physical digital quantity IO input detection. AITrig: Physical analog quantity IO input detection.

type	SIMDITrig: Virtual digital quantity IO input detection. SIMAITrig: virtual analog quantity IO input detection. TorqTrig: Torque detection.
Trigger Index	For InputTrig, this parameter represents the IO port number that needs to be detected. For TorqTirg, this parameter represents the axis number that needs to be detected.
Trigger value	For InputTrig, the threshold for IO detection. For TorqTirg, the threshold for moment detection, in kilogrammes of rated torque.
Deceleration time	The time used to adjust the deceleration acceleration. The smaller the value, the faster the deceleration, and the slower the vice versa.
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure.
Search for a successful position	The result position of the successful search.
Jump node	When no signal is detected within the waiting period, the label name of the program to jump to the row.

10.1.8 AddDo

The AddDO instruction must be added after the motion instruction, including MovJ, MovL, MovC, MovCircle, MovJRel, MovLRel, MovLW, MovCW, MovCircleW. The main purpose of this instruction is not to interrupt the transition between two motion instructions. If AddDO is added between two motion instructions, the IO operation in the child control will not interrupt the transition, otherwise the previous move instruction transition value of this instruction will not take effect.

After executing this command, the robot can perform IO operations. AddDO instructions must add child controls, can only for child controls: SetDO, SetAO, SetSimDO, SetSimAO, SetDO8421, SetSimDO8421.

10.1.9 OnDistance

Distance trigger command, without interrupting the transition, the controller can be moved some distance from the starting point or some distance from the end point to trigger the IO operation. The OnDistance instruction must be sandwiched between two motion instructions, and child controls must be added. Child controls can only be SetDO, SetAO, SetSimDO, SetSimAO, SetDO8421, SetSimDO8421.

The legend indicates that when the robot TCP moves from P3 to P4, DO1 is set to 1 2000ms before moving to the position of 100mm. If the trigger time is changed from -2000ms to 2000ms, it means that DO1 is set to 1 2s after the robot moves to the 100mm position.

Parameters	**Instructions**
Trigger type	Specify whether the current trigger is triggered some distance from the start or some distance from the end. When the trigger type is set to FromBegin, after the robot moves to the set distance, the controller will enter the trigger wait. After the trigger time is delayed, the relevant trigger operation will be executed.

	When the trigger type is set to FromEnd, after the robot moves to the set distance, the controller will enter the trigger wait, and after the trigger time is delayed, the relevant trigger operation will be executed.
Trigger distance	Trigger distance parameter.
Trigger time	Trigger Delay Time when the trigger condition is met. When it is negative, it means that the trigger signal is triggered in advance

10.1.10 OnParameter

Percentage of distance trigger instruction, which is used to trigger the set io operation at a certain point in the straight trajectory without interrupting the transition.

When the robot moves to the set trigger percentage path, the controller enters the trigger wait, and after delaying the trigger time, the relevant trigger operation begins to be executed.

The legend indicates that when the robot TCP moves from P1 to P3, DO0 is set to 1 20ms in advance when it moves to 10% position. If the trigger time is changed from 20ms to -20ms, it means that the DO0 is set to 1 when the robot moves to 10% position 20ms in advance.

Parameters	**Instructions**
Path percentage	Percentage of paths that need to be triggered.
Trigger time	Trigger Delay Time when the trigger condition is met. When it is negative, it means that the trigger signal is triggered in advance.

Triggered action	After the trigger conditions are met, the robot needs to perform the trigger operation.

10.2 Logic Indicates the logic control command

10.2.1 GoTo

GOTO instructions are used to jump to different parts of a program.

10.2.2 if

The IF instruction is used for the conditional judgment expression jump control, and the result of the judgment expression must be Bool. When the result of the conditional judgment expression is true, the program executes the contents of the program block under IF.

In the figure, 1 is the overall expression, you can select the number in the box 3 to add the expression, 2 is the expression that is currently being edited. The parameters of the expression are edited as follows:

Expression	**Instructions**

type	
value	Contains constant values and variable values, constant values currently only support digital quantities and true, false, variable can be selected to provide the basic variable, currently contains all the IO signals.
Operator	Operators include logical and or not operators and various mathematical operators such as addition, subtraction, multiplication and division.
Functions	Functions that provide common mathematical operations include sine, cosine, integer, complement, etc.

10.2.3 ElseIf

The ELSIF instruction relies on the IF instruction, followed by the IF instruction control, when the IF logic does not hold, the ELSIF logic is determined. The ELSIF expression is set in the same way as the IF expression.

10.2.4 else

The ELSE instruction depends on the IF or ELSIF instruction, followed by the IF or ELSIF

instruction control, when the IF or ELSIF condition is not determined, the statement will be executed. The ELSE instruction has no parameter configuration.

10.2.5 While

The WHILE instruction loops through the substatement when the condition is met. The loop control expression must be of type BOOL.

Edit the parameters of the expression as follows:

Expression type	**Instructions**
value	Contains constant values and variable values, constant values currently only support digital quantities and true, false, variable can be selected to provide the basic variable, currently contains all the IO signals.

Operator	Operators include logical and or not operators and various mathematical operators such as addition, subtraction, multiplication and division.
Functions	Functions that provide common mathematical operations include sine, cosine, integer, complement, etc.

10.2.6 …=…

Create an expression that can be used to assign a value to a variable. At present, the assignment command supports all I/O and INT, BOOL, REAL type variables for assignment, its configuration interface is as follows:

10.2.7 RETURN

Return the instruction. Normally, after executing this instruction, the program will jump to the end of the program, or if a RETURN instruction was used in a subroutine called in the CALL instruction, return to the one level above the CALL instruction. For example, if Return is used in a

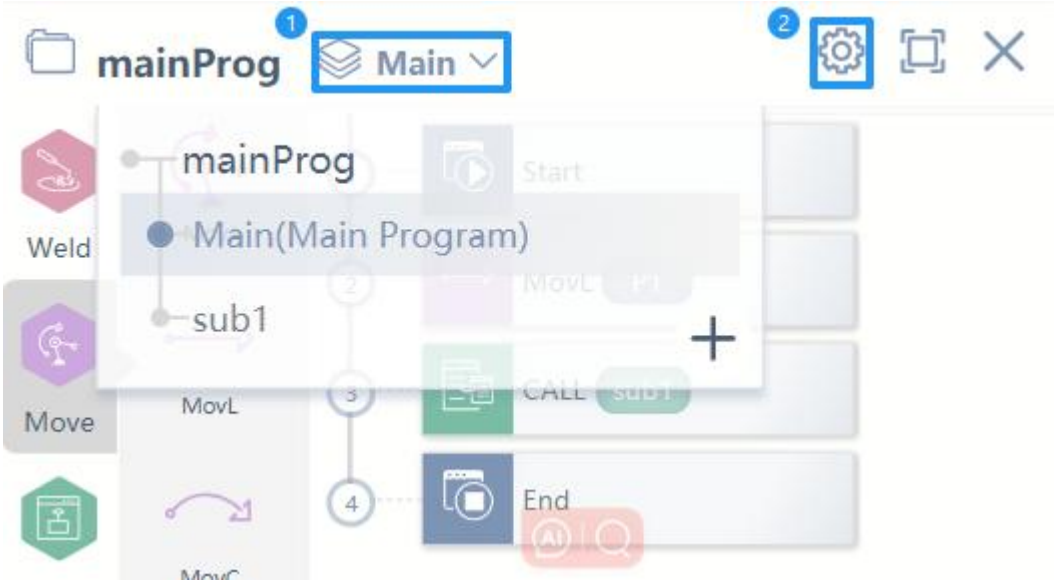
called subroutine of the main program, it will return to the main program.

10.2.8 CALL

Call instruction, the current program jumps to another subroutine under the same project, after the execution of the subroutine jumps back to the current program.

If you need to write a subroutine, you need to click project Settings in the lower left corner of the engineering interface to change the default program editing to multi-program editing.

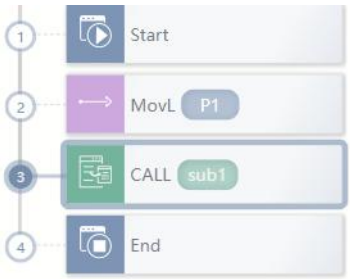
After switching to multi-program, you can click 1 current program name or 2 project properties edit in the engineering editing interface as shown below to create a new subroutine. After clicking the current program name, you can switch the current edited program in the pop-up drop-down menu.

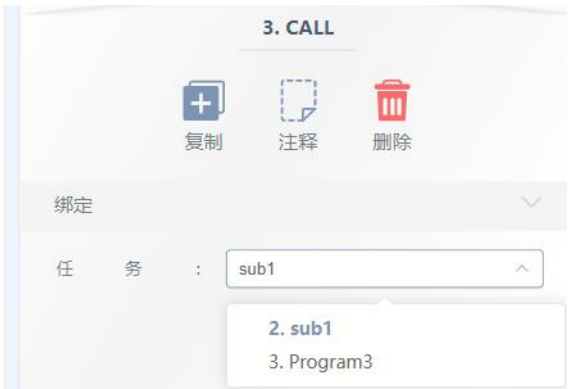


Click the project property edit button in the pop-up project property edit bar can also name the current project and subroutine. In the following figure, the current project name can be

changed in 1, and the program name can be changed in 2 and 3. Each project must contain a main program, except for the main program, other subprograms can be deleted.

The CALL instruction setting interface is as follows, you can switch the called program in the task selection box





10.2.9 RUN

Programs (tasks) run instructions in parallel, so that the robot can run other programs (tasks) while running the current program (task) (the current program continues to run without jumping at the same time to start another program), and the running tasks must be under the same

project. If the program (task) that needs to run is program2, in the program is:

10.2.10 Label

The Label directive is used to define the GOTO jump target.

10.3 Wait wait instruction

10.3.1 Wait

Used to set the robot wait time, the time unit is ms, can be int constant type.

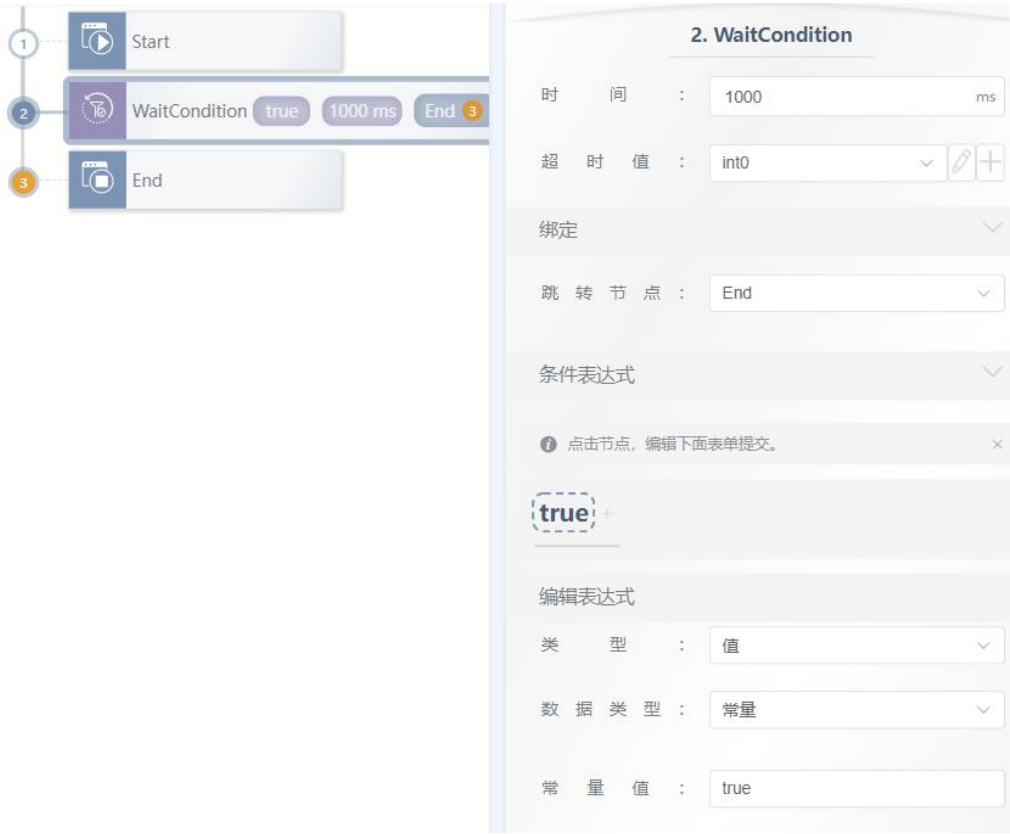
10.3.2 WaitFinish

Used to synchronize robot movements as well as program execution. The robot will directly transition to the next command when the previous command is executed to trigger the progress. You must add child controls when adding WaitFinish instructions, which can add SetDO or SetAO and their ilk. As the example program shows, the SetDO instruction will trigger when the first MovL instruction is 20% executed. The second MovL instruction will be transitioned directly after the first MovL instruction has been executed. If WaitFinish is deleted, SetDO will be executed after the first MovL instruction is executed, and then the second MovL instruction will be executed. In this case, there is no transition between the two MovL instructions, and there is a pause.

argument	Instructions
Trigger progress	Percentage of run time that the last move instruction triggered the WaitFinish neutron control.

10.3.3 WaitCondition

Sets the condition for the robot to perform the wait. If the condition is not met within the set time, the timeout state is returned. When the "criterion condition" is true, the next instruction is executed, otherwise the program will continue to wait until the expression is true.

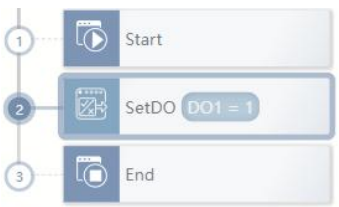


Arguments	Instructions
Time	The time required to perform the wait, in ms. If the value of this argument is 0, it will force you to wait for the discriminant condition to be true before proceeding to the next instruction. If the value of this argument is non-0, even if the discriminant condition is still not true, the system will skip the instruction after waiting for a given amount of time and continue to execute the next instruction.
Time-out Value	Select a variable and assign it a value in either of the following two cases. If the input condition is true and the instruction completes execution, assign a value of 0 to the variable; When the instruction completes due to a timeout, the variable is assigned a value of 1.
Skip node	When executing a timeout, select the row where the tag name of the jump is located.
Conditional expression	Condition for instruction execution to wait. If the condition is not met within the set time, the timeout status is returned. Expression editing is the same as IF instruction.

10.4 IO input/output instruction

10.4.1 SetDO

Set the numeric output port to TRUE (1) or FALSE (0). Where DO0-DO15 indicates the 16 digital output ports of the control cabinet, DI0-DI15 indicates the 16 input ports of the control cabinet, switch0-switch3 indicates the status of the end button of the robot.



Parameters	Instructions
Ports	Set the port number of the digital output DO.
Set value	Set port values, 0 for high and 1 for low.

10.4.2 SetAO

Set the analog output ports (AO0-AO3) to a certain value in 4mA-20mA.

Parameters	Instructions
Ports	Set the port number of the digital output AO.
Set value	Set port value, current only supported, range 4mA-20mA

10.4.3 SetSimDO

Set the virtual quantity output port to TRUE (1) or FALSE (0). The virtual numeric output does not work on the actual physical port, and is used only as a logical verification. The parameters are basically the same as SetDO.

10.4.4 SetSimAO

Set the virtual analog output port to a certain value. The virtual analog output does not work on the actual physical port, it can only be used as a logical verification, and the parameters are basically the same as SetSimAO.

10.4.5 WaitDI

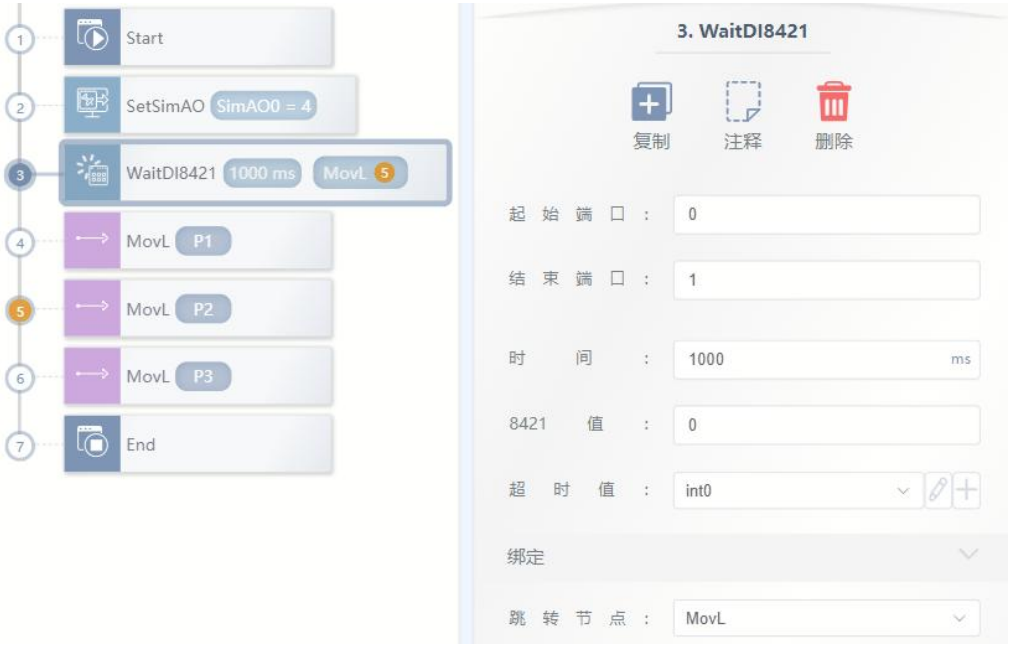
This instruction is used to wait for a numeric input DI port status within a specified period of time. If the waiting condition is met within the specified period, the program continues to execute downwards; If the condition is not met within the set period, the timeout judgment value is set to 1, and the program jumps to the jump node.

Parameters	Instructions
Port variables	Input port number to wait for.
Port value	Wait for the digital quantity to enter port level. 0: Low level

	1: High
Duration (ms)	The unit of time spent waiting for the signal to change is ms
Timeout value	Returns the result of the command execution to the variable set in the timeout value, which can only be an INT variable. When the signal is successfully waited within the waiting period, the running value of the timeout value will be set to 0; When no signal is detected within the waiting period, the running value of the timeout value is set to 1.
Skip Node	When no signal is detected within the waiting period, the label name of the program to jump to the row.

10.4.6 WaitDI8421

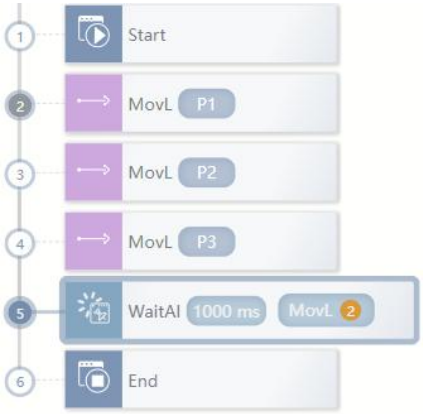
This instruction is used to wait for a set of consecutive numeric quantities to enter the DI port's state combination within a specified period of time. If the waiting condition is met within the set period, the program continues to execute downwards; If the condition is not met within the set period, the program will jump to the jump node after the timeout judgment value is 1.



Parameters	Instructions
Starting port	The starting port number of the continuous DI port in this segment represents the lower end of the 8421 conversion value
End port	End port number of the continuous DI port in this segment. Represents the high end of 8421 conversion value
Time (ms)	The duration of waiting for the input DI signal of the set of digits, which can be an int constant type, in ms.
8421 Value	Convert the VALUE of the continuous DI port to a decimal number according to rule 8421. If the value is equal to the value, the condition is met. For example, if the start port is 0, the end port is 2, and the value of 8421 is 4, then the conditions are met when DI0 is 0, DI1 is 0, and DI2 is 1.
Time-out value	Returns the result of the command execution to the variable set in the timeout value, which can only be an INT variable. When the signal is successfully waited within the waiting period, the running value of the timeout value will be set to 0; When no signal is detected within the waiting period, the running value of the timeout value is set to 1.
Skip Node	When no signal is detected within the waiting period, the label name of the program to jump to the row.

10.4.7 WaitAI

This instruction is used to wait for an analog to enter the AI port state within a specified amount of time. If the waiting condition is met within the specified period, the program continues to execute downwards; If the condition is not met within the set period, the timeout judgment value is set to 1, and the program jumps to the jump node.



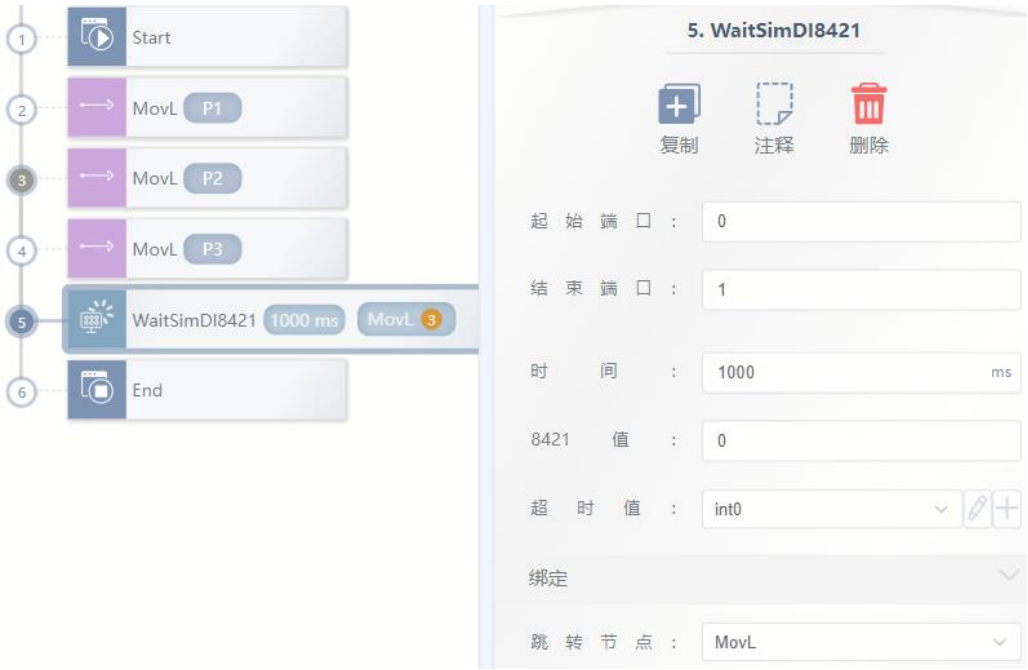
argument	Instructions
Port variables	Amount of analog to wait for Enter the port number.
Port value	Waiting analog input port current value (4mA-20mA)
Duration (ms)	The unit of time spent waiting for the signal to change is ms
Timeout value	Returns the result of the command execution to the variable set in the timeout value, which can only be an INT variable. When the signal is successfully waited within the waiting period, the running value of the timeout value will be set to 0; When no signal is detected within the waiting period, the running value of the timeout value is set to 1.
Skip Node	When no signal is detected within the waiting period, the label name of the program to jump to the row.

10.4.8 WaitSimDI

Wait until the virtual digit input port is set or reset, the parameter description is basically the same as for WaitDI.

10.4.9 WaitSimDI8421

The state combination used to wait for a set of consecutive virtual digits to enter SimDI ports within a specified period of time. The parameter description is basically the same as WaitDI8421.



10.4.10 WaitSimAI

Wait until the virtual analog input port value is equal to the given value, the parameter description is basically the same as WaitAI.

10.4.11 GetDI8421

This instruction is used to take a contiguous piece of DI port status (treat it as binary data) and return it in decimal.

Arguments	**Instructions**
Starting port	The starting DI port number you want to get, the lowest value of 8421.
End port	The end DI port number you want to get, the highest bit of the 8421 value.
Returned value	A variable of type INT. When the program is running, the obtained port status is converted from binary to decimal and passed into the int variable. For example, when the start port is set to 0 and the end port is set to 2:

	When DI0 is 0, DI1 is 0, and DI2 is 1, the return value is 4; When DI0 is 0, DI1 is 1, and DI2 is 1, the return value is 6.

10.4.12 GetSimDI8421

This instruction is used to obtain a continuous number of virtual input SimDI port status (as a piece of binary data) and return in decimal number, the parameter description is the same as GetDI8421 instruction.

10.4.13 SetDO8421

Set a continuous DO port state (think of it as a piece of binary data), convert the incoming decimal number to binary number set to the specified DO port.

Parameters	Instructions
Starting port	The low point at which the value will need to be set to binary outgoing.
End port	The high point at which the value will need to be set in binary outgoing.
Returned value	The decimal setting value that you want the port to output. For example, when the starting port is 0, the ending port is 2, and the setpoint is 6: DO0 is 0, DO1 is 1, and DO2 is 1.

10.4.14 SetSimDO8421

Sets the state of a consecutive SimDO port (as a piece of binary data), converts the incoming decimal number to a binary number and sets it to the specified SimDO port. The parameter description is basically the same as SetDO8421.

10.4.15 GetSimDIToVar

Map the virtual digital quantity input signal to some variable. Its configuration interface is as follows:

Parameters	Instructions
Ports	Virtual digital quantity output type port.
Variables	The obtained virtual numeric quantity mapping variable of type BOOL.

10.4.16 SetSimDOByVar

Map the value of a Boolean variable to a numeric output variable.

Parameters	Instructions
Ports	Virtual numeric output type port, output after receiving variable value mapping.
Variables	Boolean variable that outputs its value to the port variable.

10.4.17 GetSimAIToVar

Map the virtual analog input signal to some variable. Its configuration interface is as follows:

Parameters	Instructions
Ports	Virtual analog output type port.
Variables	Obtained virtual analog mapping variable of type REAL type.

10.4.18 SetSimAOByVar

Map the value of some REAL type variable to a virtual analog output variable.

Parameters	Instructions
Ports	Virtual analog outputs type variables, receives the value of the variable and outputs it.
Variables	A variable of type REAL outputs its value to the port variable.

10.5 Set setup instruction

10.5.1 SetTool

Set tool parameter instructions. Switch to this tool parameter.

Parameters	Instructions
Tool parameters	Select the tool variable that has been created.

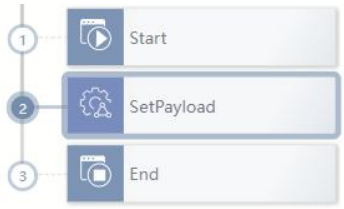
10.5.2 SetCoord

Set user coordinate system instruction. Switch to this user coordinate system.

Parameters	Instructions
Coordinate system	Select the coordinate system variable that has been created.

10.5.3 SetPayload

Select the job load parameter directive. Switch to this workpiece load parameter.



Parameters	Instructions
Work load	Select the load variable that has been created.

10.5.4 Stop

This command is used to stop the execution of all activation programs.

10.5.5 AutoGainEnable

This command is used to enable the self-adjustment of servo parameters. After this command is executed, the system will optimize the subsequent motion commands at low speed according to the incoming LsScale and LsThresh parameters.

10.5.6 AutoGainDisable

This command is used to disable the autogaindisable command.

10.5.7 EnableVibrationSuppression

The enablevibrationsuppression command is used to enable the vibration suppression function.

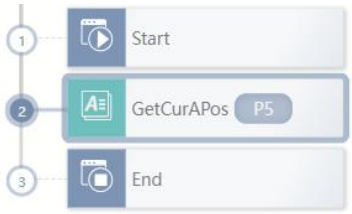
10.5.8 DisableVibrationSuppression

This command is used to disable the vibrationsuppression function.

10.6 Position Indicates the position operation

10.6.1 GetCurAPos

This instruction is used to get the position in the current joint coordinate system, assigned to the Apos type variable. The APOS type variable can be added by clicking +APOS.



Parameters	Instructions
Storage location	The current Apos numerical variables

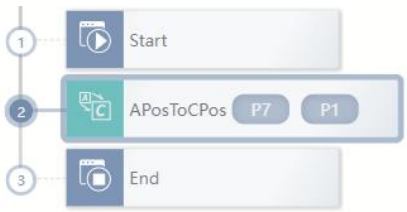
10.6.2 GetCurCPos

This instruction is used to get the Cartesian space position in the current reference coordinate system, assigned to the Cpos type variable. The APos type variable can be added by clicking +CPOS.

Parameters	Instructions
Storage location	Current Cpos numerical variable

10.6.3 APosToCPos

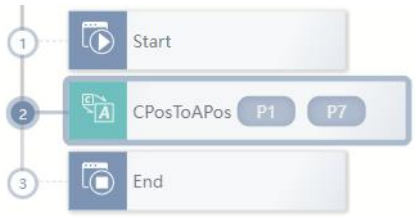
Robot position point conversion instruction, given the APos point under the base system and the reference coordinate system and tool parameters of the target CPos point to be converted, the value of the CPos point with tool parameters under the target coordinate system can be obtained.



Parameters	**Instructions**
Pre-conversion point	Apos variable before conversion
Post-conversion point	The converted Cpos variable
Tool parameters	Tool TCP parameters involved when converting
Coordinate system	Coordinate system parameters involved in the conversion

10.6.4 CPosToAPos

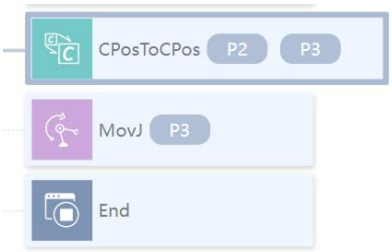
Robot position point conversion instruction, given the CPos point and its reference coordinate system and tool parameters, the value of the target APos point can be obtained.



Parameters	**Instructions**
Pre-conversion point	Cpos variable before conversion
Tool parameters	Tool TCP parameters involved when converting
Coordinate system	Coordinate system parameters involved in the conversion
Point after conversion	The converted Apos variable

10.6.5 CPosToCPos

Robot position point conversion instruction, given the CPos point and its reference coordinate system and tool parameters, as well as the reference coordinate system and tool parameters of the target CPos point to be converted, the value of the target CPos point can be obtained.



Parameters	**Instructions**
Pre-conversion point	Cpos variable before conversion
Target tool parameters	Tool TCP parameters involved in Cpos before conversion
Target user coordinate system	Coordinate system parameters involved in Cpos before conversion
Post-conversion point	The converted Apos variable
Benchmark tool parameters	Tool TCP parameters involved in the converted Cpos
The user coordinate system	The converted Cpos coordinate system parameters

10.6.6 ToolOffset

Robot tool coordinate system offset command, this command can rotate or offset the reference tool coordinate system to generate a new tool coordinate system. Given the reference TOOL coordinate system tool and the offset that needs to be offset or rotated, the value of the target tool coordinate system can be obtained.

Parameters	Instructions
Tool parameters	TCP parameters of the tool before migration
Offset	Offset DCpos parameter
Tool parameters	Tool TCP parameters after the offset

10.6.7 UserOffset

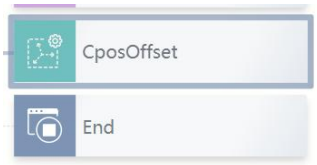
This command can be used to rotate or offset the base user coordinate system to generate a new user coordinate system. Given the reference user coordinate system USERCOORD and the offset that needs to be offset or rotated, the value of the target user coordinate system can be obtained.

Parameters	Instructions
Coordinate	Coordinate system parameters before offset

system	
Offset	Offset DCpos parameter
Coordinate system	The offset coordinate system parameters

10.6.8 CposOffset

Robot Cpos offset instruction, through which the original Cpos can be rotated or offset to generate a target Cpos.



Parameters	**Instructions**
Coordinate system	Cpos parameters before offset
Offset	Offset DCpos parameter
Coordinate system	Offset Cpos parameters

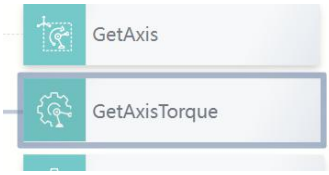
10.6.9 GetAxis

This command the user gets the Angle of the specified axis.

Arguments	Instructions
Origin	Selected Apos
shaft	The selected axis number
Angle value	Angle value

10.6.10 GetAxisTorque

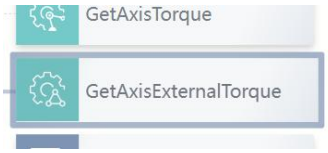
This command is used to get the actual output torque of the robot joint.



Parameters	Instructions
shaft	Selected axis number
Moment value	The total torque value of the robot joint output

10.6.11 GetAxisExternalTorque

This command is used to obtain the torque output by the robot joints against external loads.

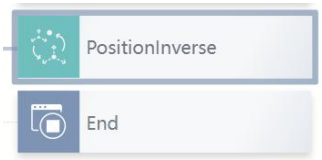


Parameters	Instructions

shaft	The selected axis number
Moment value	The value of external forces applied to the robot, such as the force applied by tools, collisions

10.6.12 PositionInverse

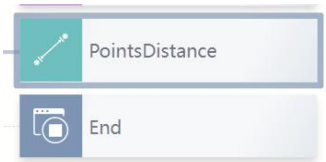
This instruction is used to calculate the inverse of the pose transform.



parameter	**Instructions**
Original Point	Position inversion for original position
Memory Points	Invert the pose for the result pose

10.6.13 PointsDistance

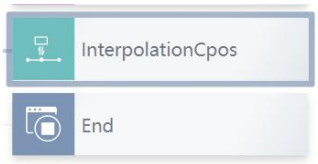
This command is used to calculate the distance between two pointsdistance points.



parameter	Instructions
Original Point	Starting pose
Memory Point	Finish pose
Distance	The distance between the two pose points

10.6.14 InterpolationCpos

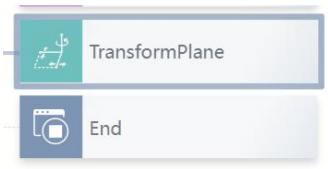
This instruction is used to calculate pose interpolation between the start and end points.



Parameters	Instructions
Starting points	Starting pose
At the end of	Finish position
Coefficient	Interpolation coefficient, interval 0 to 1,0 is the starting point, 1 is the end point
Memory Points	Pose interpolation results pose

10.6.15 TransformPlane

This instruction makes a transformation in the XY(YZZX) plane, rotating first around the Z(XY) axis at the base point, then translating along the X(YIZ) axis and Y(ZIX) axis. The base point can be dragged into the position variable, the array variable or the teaching point, the memory point is the transformed pose.



Parameters	**Instructions**
Original Point	Original point
Flat surface	The plane in the point coordinate system
Rotation Angle	The rotation Angle along the selected plane
Translation in the __ direction	Translation distance in both directions of the selected plane
Memory point	Result point variable

10.7 Bit operation instruction

10.7.1 BitAnd

Implement the bit-and-sum operation. This instruction performs a bit-and-sum operation on two operands and assigns the result to the first operand.

Arguments	Instructions
Operand 1	Variable of type INT; The result is also assigned to the operand
Operand 2	Variable of type INT

10.7.2 BitNeg

Implement the operation of taking and reversing bits. This instruction performs a bitwise inverse operation on the operand and assigns the result to the operand.

Arguments	Instructions
Operand 1	Variable of type INT; The result is also assigned to the operand

10.7.3 BitOr

Implement bitor operations. This instruction performs a bitwise or operation on two

operands and assigns the result to the first operand.

argument	Instructions
Operand 1	Variable of type INT; The result is also assigned to the operand
Operand 2	An INT type variable

10.7.4 BitXOr

Implement xOR operations by bit. This instruction performs an Xor-by-bit operation on two operands and assigns the result to the first operand.

Arguments	Instructions
Operand 1	Variable of type INT; The result is also assigned to the operand
Operand 2	An INT type variable

10.7.5 BitLSH

Implement the operation of moving left by bit. This instruction performs a left shift by bit operation on the first operand by the number of left shifts specified by the second operand and assigns the result to the first operand.

argument	Instructions
Operand 1	Variable of type INT; The result is also assigned to the operand
Operand 2	Variable of type INT

10.7.6 BitRSH

Implement the operation of shifting to the right bit by bit. This instruction performs a right shift by bit operation on the first operand by the number of right shifts specified by the second operand and assigns the result to the first operand.

Arguments	Instructions
Operand 1	INT variables; The result is also assigned to the operand
The operand 2	An INT type variable

10.8 Clock clock instruction

Create a CLOCK type variable when using a clock instruction.

10.8.1 CLKStart

Start the specified clock (after starting, you can see from the list of variables that the state of the specified clock variable is true and the value is the time recorded).

10.8.2 CLKStop

Stop the specified clock (whose state is false, but does not reset).

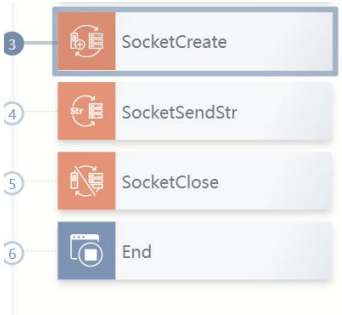
10.8.3 CLKReset

Reset the state of the clock value specified.

10.9 Socket instruction

10.9.1 SocketCreate

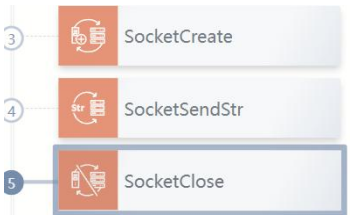
Create socket clients to facilitate data interaction with the server. Create the client locally and establish a connection with the server side according to the server side parameters passed in.



Parameters	Instructions
socket name	The name of the socket to be created. This value is passed in as a socket variable
IP address	The ip address of the server you want to connect to
Port number	Port number of the server you want to connect to
Returned value	Operation is successful, the return value of the variable returns a value of 0 indicates success, the return value of 1 is for failure. If the socket has been created, return 1, this operation does not indicate whether communication is established

10.9.2 SocketClose

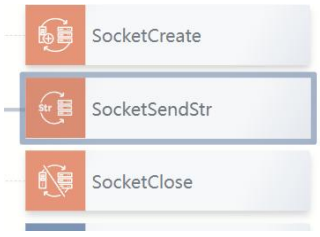
Close the socket client you have created previously. Based on the socket name passed in, close the client you created and return the success of the operation.



Arguments	Instructions
socket name	The name of the socket to be closed. This value is passed in as a socket variable
Returned value	Operation is successful, the return value of the variable returns a value of 0 indicates success, the return value of 1 is for failure

10.9.3 SocketSendStr

Send a string to the server side to interact with the command. Send a string to the server that has established a connection, and return success or failure.

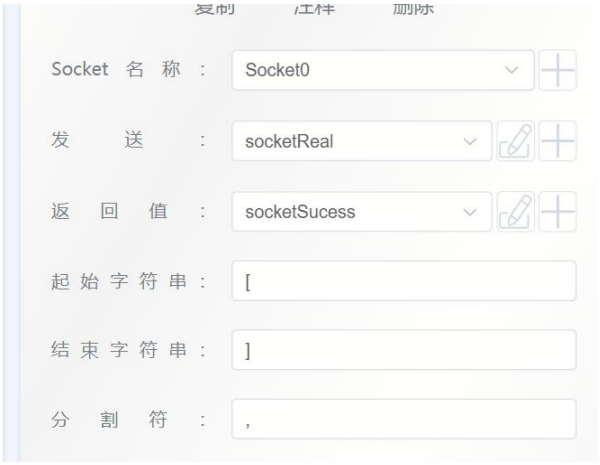


Arguments	Instructions
socket name	The name of the socket to which the send operation is to be performed. This value is passed in as a socket variable
Send	The string data needs to be sent to the server side
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure

End of a newline	Whether to add a "\n" line break

10.9.4 SocketSendReal

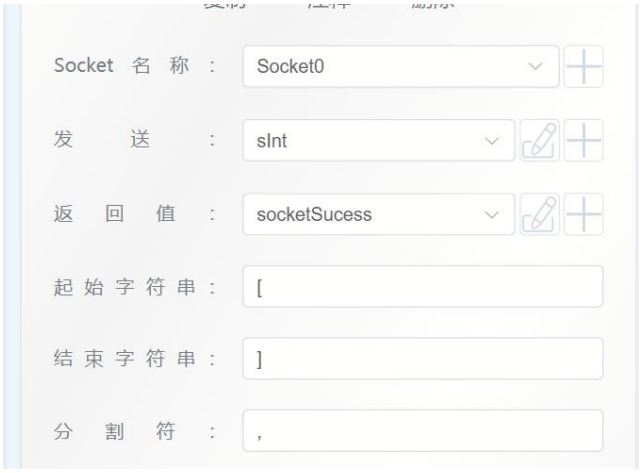
Send the real array to the server side for the interaction of the command. Send the real array to the server that has established a connection, and return success or failure. The start and end characters of the sent string can be customized, and the data is separated by delimiters.



Arguments	**Instructions**
socket name	The name of the socket to which the send operation is to be performed. This value is passed in as a socket variable
Send	real array data needs to be sent to the server side.
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure
Starting string	Starting string
End of the string	Ending string
Separator	Separator between data
A newline at the end	Whether to add a "\n" line break

10.9.5 SocketSendInt

Send an int array to the server side to interact with the command. Send an int array to the server that has established a connection, and return success or failure.



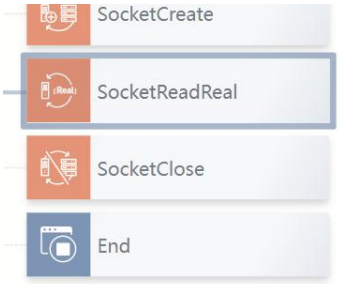
Arguments	**Instructions**
socket name	The name of the socket to which the send operation is to be performed. This value is passed in as a socket variable
Send	int array data needs to be sent to the server.
The return value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Starting string
Ending string	Ending string
Separator	Separator between data
End of line break	Whether to add a "\n" line break

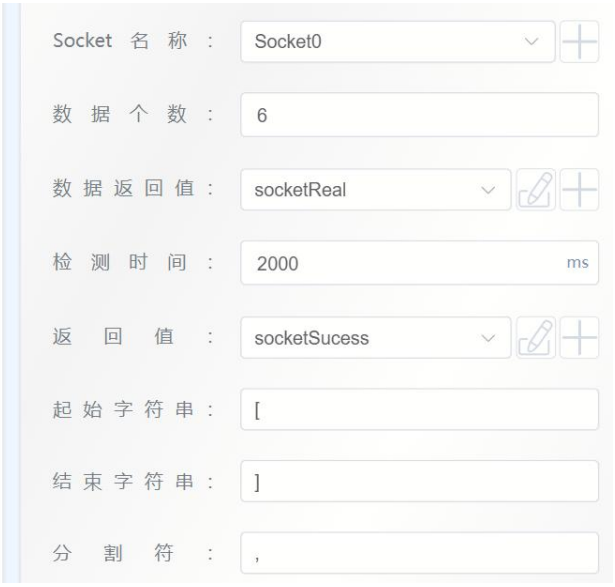
10.9.6 SocketReadReal

Read the string sent from the server and store it as a real array. Wait for and receive the string sent by the server in the format of a starting and ending string, with data separated by

delimiters. When the string is received, the robot system will split and parse it and store it in an array in

order.

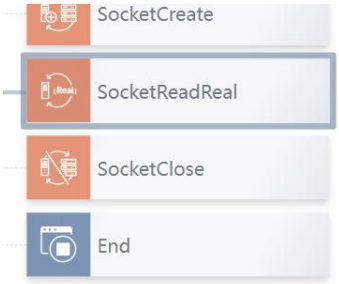


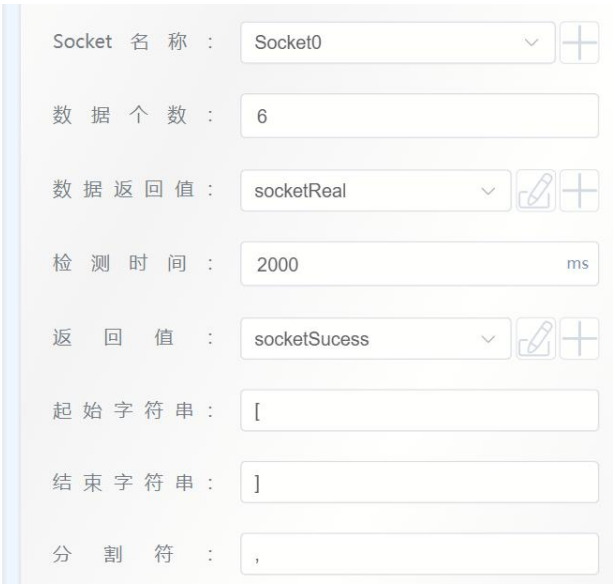


Parameters	**Instructions**
The name of the socket	The name of the socket where the read operation is to be performed. This value is passed in as a socket variable
Number of data	The number of data to be read and stored into the array.
Data return value	To read and the value of the converted into an array variable, and return the array variable.
Detect time	Wait time for the server to send data. Timeout alarm.
The return value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Starting string
Ending string	Ending string
Separator	Separator between data

10.9.7 SocketReadInt

Read the string sent by the server and store it as an int array. Wait for and receive the string sent by the server in the format of a starting and ending string, with data separated by delimiters. When the string is received, the robot system will split and parse it and store it in an array in order.





Parameters	**Instructions**
socket name	The name of the socket in which the read operation is to be performed. This value is passed in as a socket variable
Number of data	The number of data to be read and stored into the array.
Data return value	Store the read and converted value into an array variable and return that array variable.
Detect time	Waiting for the server to send data waiting time. Timeout alarm.
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Starting string
Ending string	End string
Separator	Separator between data

10.9.8 SocketReadStr

Read the string sent from the server and store it as a string. Waiting to send and receive the server string, the format is up at the end of the string and string data. When the string is received, the robot system will split and parse it into character variables.

Parameters	**Instructions**
The name of the socket	The name of the socket in which the read operation is to be performed. This value is passed in as a socket variable
Data return value	string data sent by the server. The value is returned as a string variable.
Detection time	Wait time for the server to send data. Timeout alarm.
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Starting a string
Ending string	End of the string

10.10 ModbusTcp directive

10.10.1 GetModConState

This command is used to get the connection status of the robot communicating with the outside world using ModbusTCP.

Parameters	Instructions
Device name	The name of the Modbus device that you want to operate
Connection status	Returns the current connection status of type BOOL

10.10.2 ReadSingleCoilReg

This instruction is used to read the individual coil register of the specified Modbus.

Parameters	Instructions
Device name	The name of the Modbus device that you want to operate
Address	The address of the register to be read
Destination register value	The variable that holds the read register value is of type BOOL

Slave device address	Slave device address
Timeout time	Waiting for reading time, timeout alarm
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure

10.10.3 ReadDiscreteInputReg

This instruction is used to read discrete input registers of the specified Modbus.

Parameters	**Instructions**
Device name	The name of the Modbus device that you want to operate
Address	The address of the register to be read
Destination register value	The variable that holds the read register value, of type Int
Slave device address	Slave device address
Timeout time	Time to wait for a read, timeout alarm
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure

10.10.4 ReadSingleHoldReg

This instruction is used to read a single hold register for the specified Modbus.

Parameters	**Instructions**
Device name	The name of the Modbus device that you want to operate
Address	The address of the register to be read
Destination register value	The variable that holds the read register value, of type Int
Slave device address	Slave device address
Timeout time	Time to wait for a read, timeout alarm
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure

10.10.5 ReadInputReg

This instruction is used to read the input register of the specified Modbus.

Parameters	Instructions
Device name	The name of the Modbus device that you want to operate
Address	The address of the register to be read
Destination register value	The variable that holds the read register value, of type Int
Slave device address	Slave device address
Timeout time	Time to wait for a read, timeout alarm
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure

10.10.6 WriteSingleCoilReg

This instruction is used to write to a single coil register of the specified Modbus.

Parameters	**Instructions**
Device name	The name of the Modbus device that you want to operate
Address	Register address to write to
Destination register value	The variable that holds the value written to the register is of type BOOL
Slave device address	Slave device address
Timeout time	Time to wait for a read, timeout alarm
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure

10.10.7 WriteSingleHoldReg

This instruction is used to write to a single hold register of the specified Modbus.

Arguments	**Instructions**
Device name	The name of the Modbus device that you want to operate
Address	Register address to write to
Destination register value	The variable that holds the value written to the register is of type Int
Slave device address	Slave device address
Timeout time	Time to wait for a read, timeout alarm
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure

10.11 Matrix array instruction

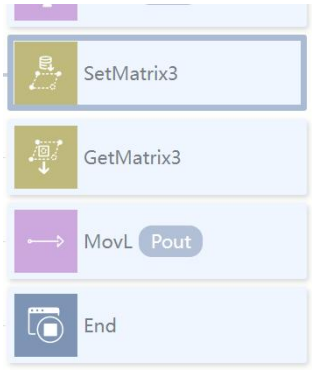
10.11.1 SetMatrix2

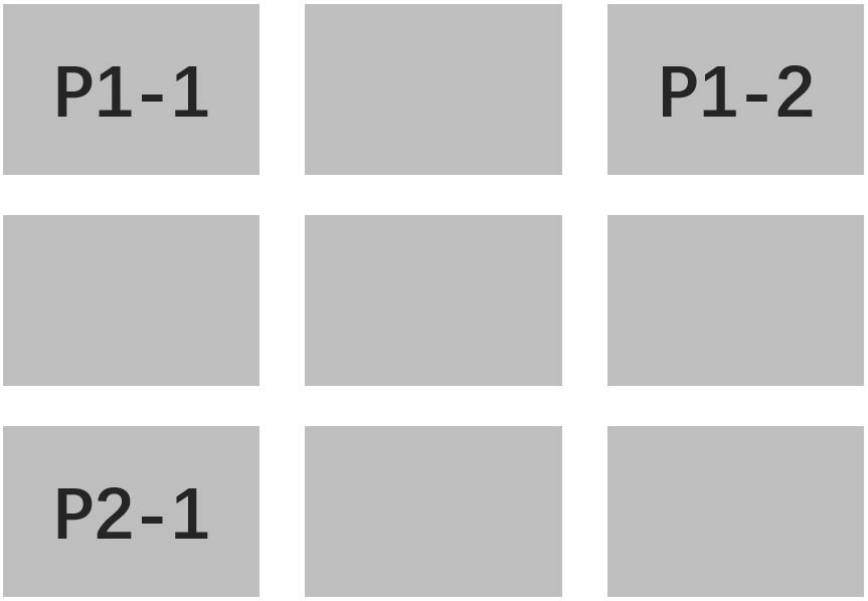
To specify two points to form a line array in space, and this line array is divided equally according to the number of rows set to get a matrix point group.

Parameters	Instructions
Matrix name	The Matrix name you want to operate
p1	Specify the first point of the line array, of type CPOS
p2	Specify the last point of the line array with type CPOS
Number	Number of rows in the generated array, type INT

10.11.2 SetMatrix3

To specify three points to form a parallelogram array in space, and this parallelogram is divided equally according to the number of rows and columns set to get a matrix point group.



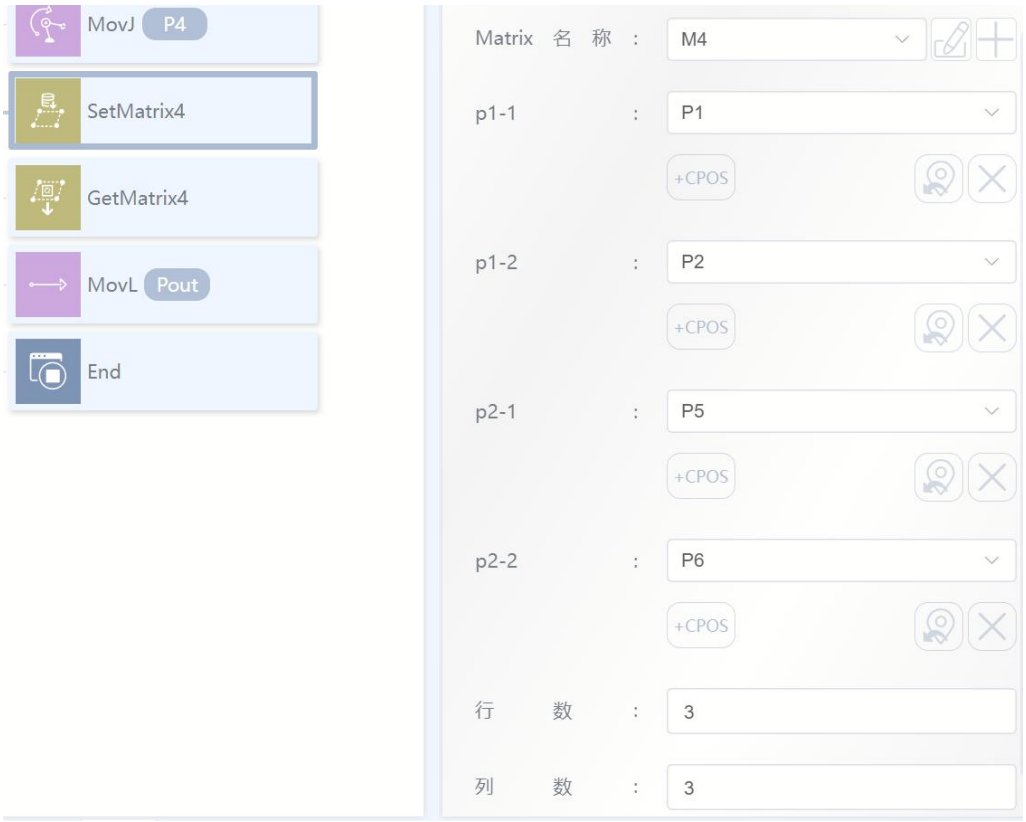


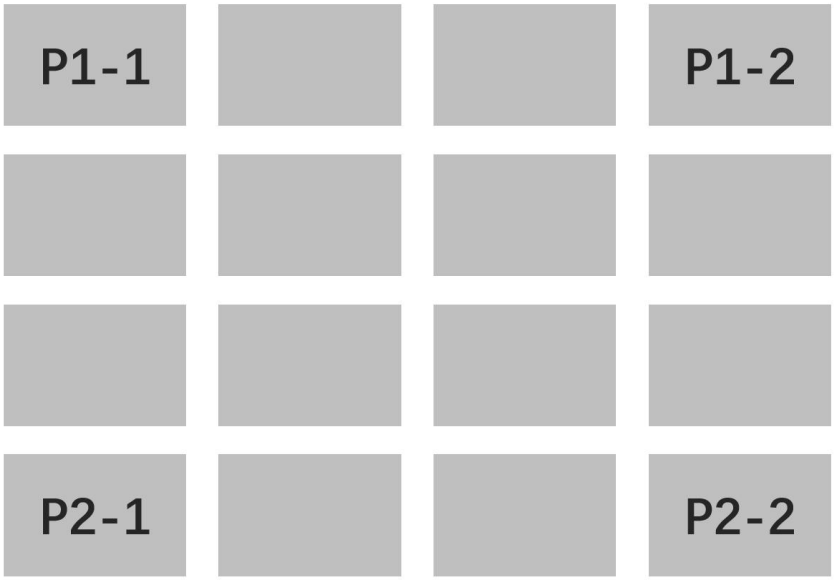
argument	**Instructions**
Matrix name	The Matrix name you want to operate
p1-1	Specify the first point of the first row of the parallelogram, also known as the origin, with type CPOS.
p1-2	Specify the last point in the first row of the parallelogram, of type CPOS
p2-1	Specify the first point of the last row of the parallelogram, of type CPOS
Number of rows	Number of rows in the generated array, type INT

Number of columns	The number of columns in the generated array, type INT

10.11.3 SetMatrix4

To specify four points to form a parallelogram array in space, and this parallelogram according to the number of rows and columns equal to get a matrix point group. Compared with the Matrix3 instruction, this function can obtain more accurate point position. When calculating the target point position, the array will be divided into 4 areas, and then automatically select the three points closest to the target point in their respective areas to do Matrix3 operation.



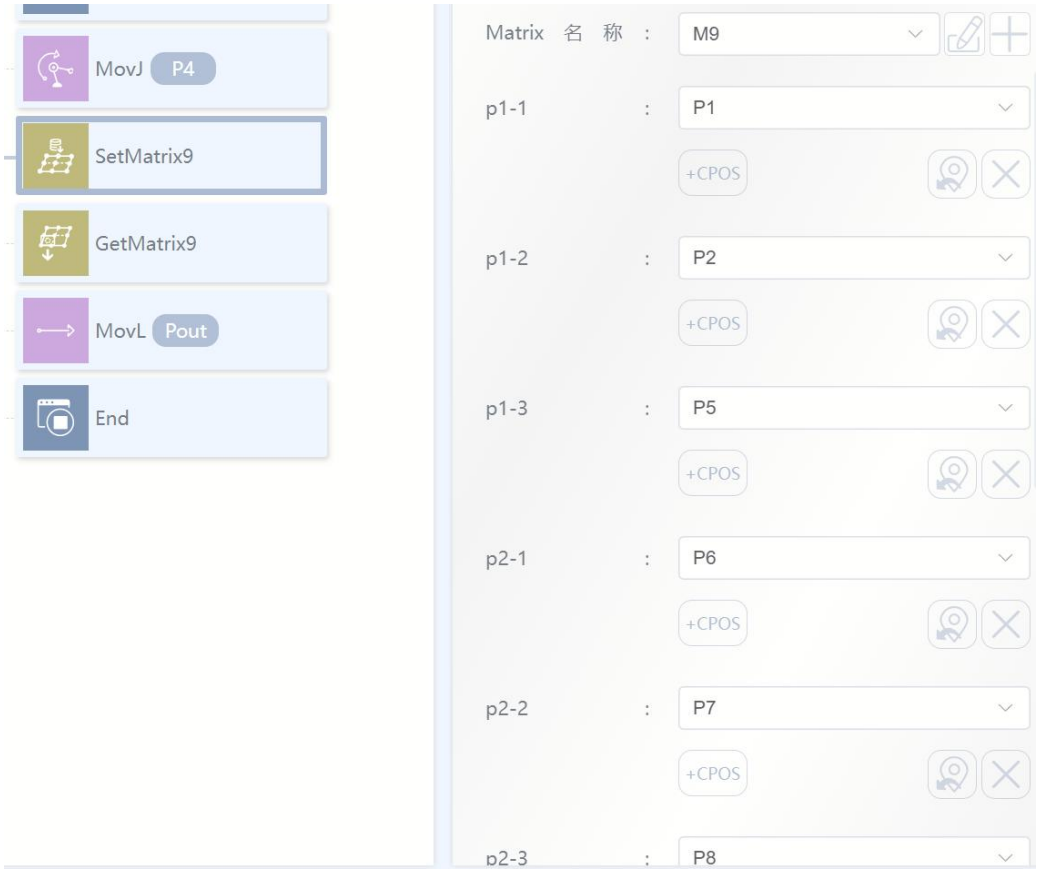


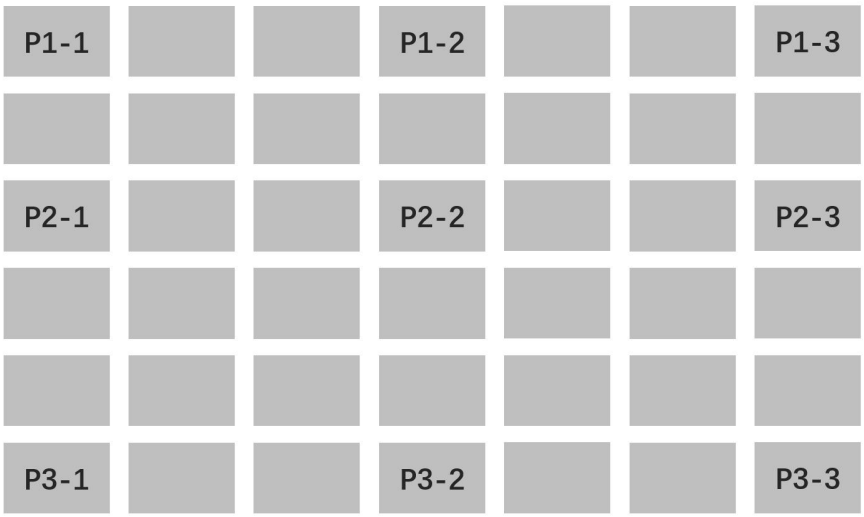
argument	Instructions
Matrix name	The Matrix name you want to operate
p1-1	Specify the first point of the first row of the parallelogram, also known as the origin, with type CPOS.
p1-2	Specify the last point in the first row of the parallelogram, of type CPOS
p2-1	Specify the first point of the last row of the parallelogram, of type CPOS
p2-2	Specify the last point of the last row of the parallelogram, of type CPOS
Number of rows	Number of rows in the generated array, type INT
Number of columns	The number of columns in the generated array, type INT

10.11.4 SetMatrix9

To specify four points to form a parallelogram array in space, and this parallelogram according to the number of rows and columns equal to get a matrix point group. Compared with the Matrix3 instruction, this function can obtain more accurate point positions. When calculating the target point position, the array will be divided into 9 areas, and then automatically select the three points closest to the target point in their respective areas to do Matrix3 calculation. When rows or columns are even, the middle point should be selected near the first point of the row or

column in the middle position.



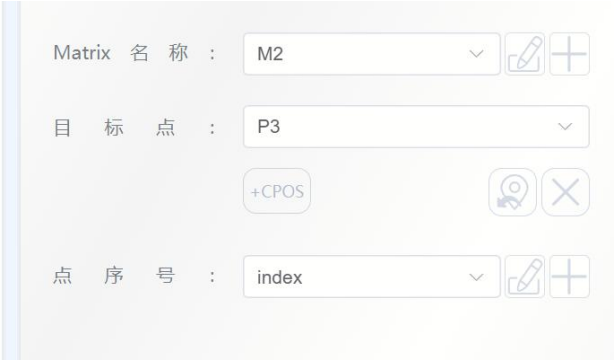


Parameters	Instructions
Matrix name	The Matrix name you want to operate
p1-1	Specify the first point of the first row of the parallelogram, also known as the origin, with type CPOS.
p1-2	Specify the point near the first point at the middle of the first row of the

	parallelogram, of type CPOS
p1-2	Specify the last point in the first row of the parallelogram, of type CPOS
p2-1	Specify the first point of the last row of the parallelogram, of type CPOS
p2-2	Specify the point near the first point at the middle position of the parallelogram, of type CPOS
p2-3	Specify the last point of the last row of the parallelogram, of type CPOS
p3-1	Specify the first point of the last row of the parallelogram, of type CPOS
p3-2	Specify the point near the first point at the middle of the last row of the parallelogram, of type CPOS
p3-2	Specify the last point of the last row of the parallelogram, of type CPOS
Number of rows	Number of rows in the generated array, type INT
Number of columns	The number of columns in the generated array, type INT

10.11.5 GetMatrix2

Take the value of that point in the corresponding column after the execution of the SetMatrix instruction and assign it to the target point. The pose of the target point as well as the additional axis Angle value is consistent with the p1 point of the SetMatrix instruction.



Parameters	**Instructions**
Matrix name	The Matrix name you want to operate
Target point	Used to save the fetched point value of type CPOS

Dot sequence number	Have to take the serial number in the Matrix type INT, serial Numbers starting from 0

10.11.6 GetMatrix3

Take the value of that point in the corresponding column after the execution of the SetMatrix instruction and assign it to the target point. The attitude of the target point as well as the additional axis Angle value is consistent with the point p1-1 of the SetMatrix instruction.

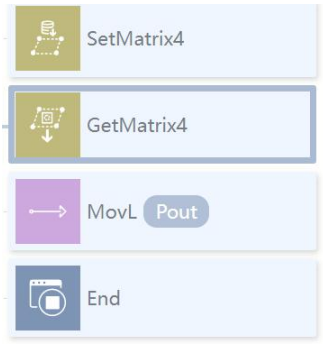
Parameters	Instructions
Matrix name	The Matrix name you want to operate
Number of rows	Have to take the line number in the matrix type INT, serial Numbers starting from 0
Number of columns	The row number of the point to be taken in the matrix, of type INT, with the serial number counting from 0
Target point	Used to save the fetched point value of type CPOS

10.11.7 GetMatrix4

Take SetMatrix instruction execution after the value of the corresponding point of ranks, assigned to the target. The attitude of the target point as well as the additional axis Angle value is consistent with the point p1-1 of the SetMatrix instruction. This function than Matrix3 instructions can obtain more accurate point, calculating target point array can be divided into

four regions, and then automatically in the respective area for the three point closest to the target point selected Matrix3

operations.



Parameters	Instructions
Matrix name	The Matrix name you want to operate
Number of rows	The row number of the point to be taken in the matrix, type INT, the serial number counts from 0
Number of columns	The row number of the point to be taken in the matrix, of type INT, with the serial number counting from 0
Target point	Used to save the fetched point value of type CPOS

10.11.8 GetMatrix9

Take the value of that point in the corresponding column after the execution of the SetMatrix instruction and assign it to the target point. The attitude of the target point as well as the additional axis Angle value is consistent with the point p1-1 of the SetMatrix instruction. Compared with Matrix3 instruction, this function can obtain a more accurate point position. When calculating the target point position, the array will be divided into 9 regions, and then the three points closest to the target point in their respective regions will be automatically selected for Matrix3 calculation.

argument	Instructions
Matrix name	The Matrix name you want to operate
Number of rows	The row number of the point to be taken in the matrix, type INT, the serial number counts from 0
Number of columns	Have to take the line number in the matrix type INT, serial Numbers starting from 0
Target point	Used to save the fetched point value of type CPOS

10.12 String String instruction

10.12.1 APosToStr

This instruction is used to convert the Apos variable to a string variable.

Arguments	Instructions

APOS to be converted	The value of the APOS to be converted
Store string	The converted string variable
Starting string	Add starting string
Ending string	Add Ending string
Splitter	Space symbol between numeric values
Angle of the unit	The form of angular numerical units in APOS

10.12.2 CPosToStr

This instruction is used to convert the Cpos variable to a string variable.

Arguments	**Instructions**
CPOS to be converted	CPOS value to be converted
Store string	The converted string variable
Starting string	Add starting string
Ending string	Add Ending string

Splitter	Space symbol between numeric values
Unit of Angle	Numeric unit form of Angle in CPOS
Unit of length	Numeric unit of length in CPOS

10.12.3 DAPosToStr

The instruction is used for converting DApos variable string variable.

Arguments	**Instructions**
To convert DAPOS	The DAPOS value to be converted
Store string	The converted string variable
Starting string	Add starting string
Ending string	Add Ending string
Splitter	Space symbol between numeric values
Angle of the unit	Numeric unit form of Angle in DAPOS

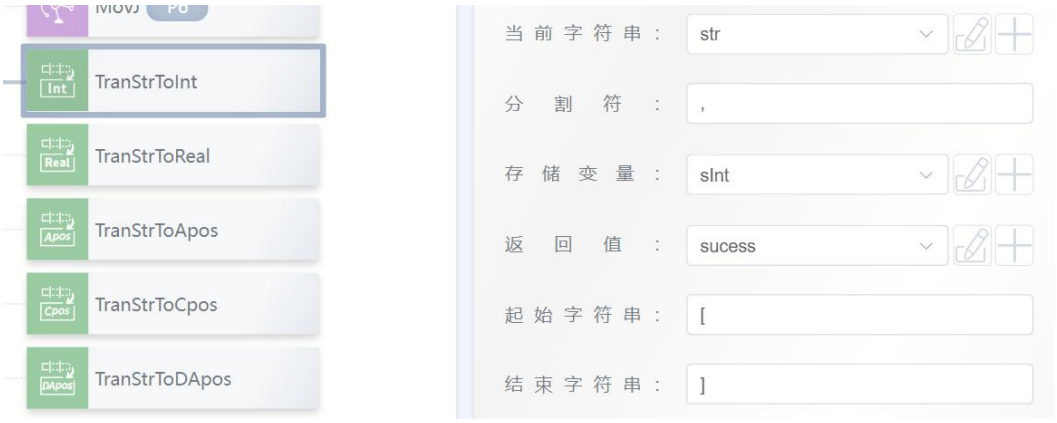
10.12.4 DCPosToStr

This instruction is used to convert DCpos variables to string variables.

Arguments	**Instructions**
DCPOS to be converted	DCPOS value to be converted
Store string	The converted string variable
Starting string	Add starting string
Ending string	Add Ending string
Splitter	Space symbol between numeric values
Unit of Angle	Numeric unit form of Angle in DCPOS
Unit of length	Numeric unit form of length in DCPOS

10.12.5 TranStrToInt

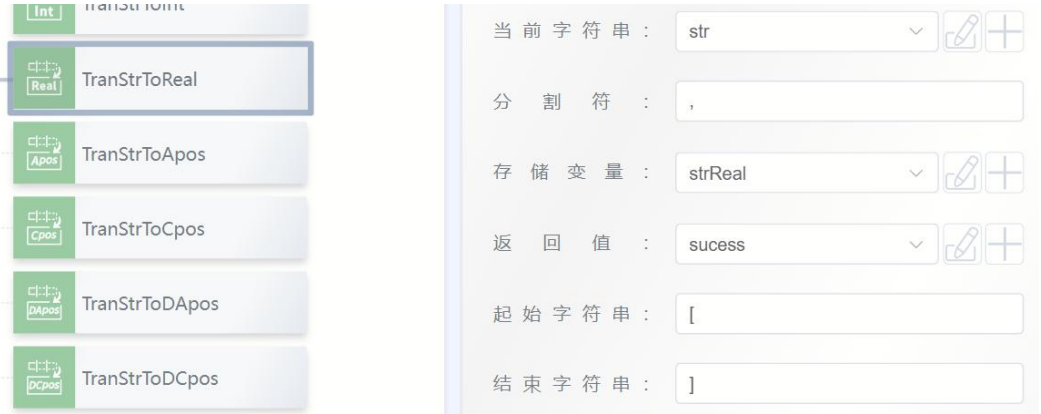
This instruction converts a string variable to an int variable.



Arguments	Instructions
Current string	The string to be converted
separator	Space symbol between numeric values
Storing variables	The converted int variable
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Add starting string
Ending string	Add Ending string

10.12.6 TranStrToReal

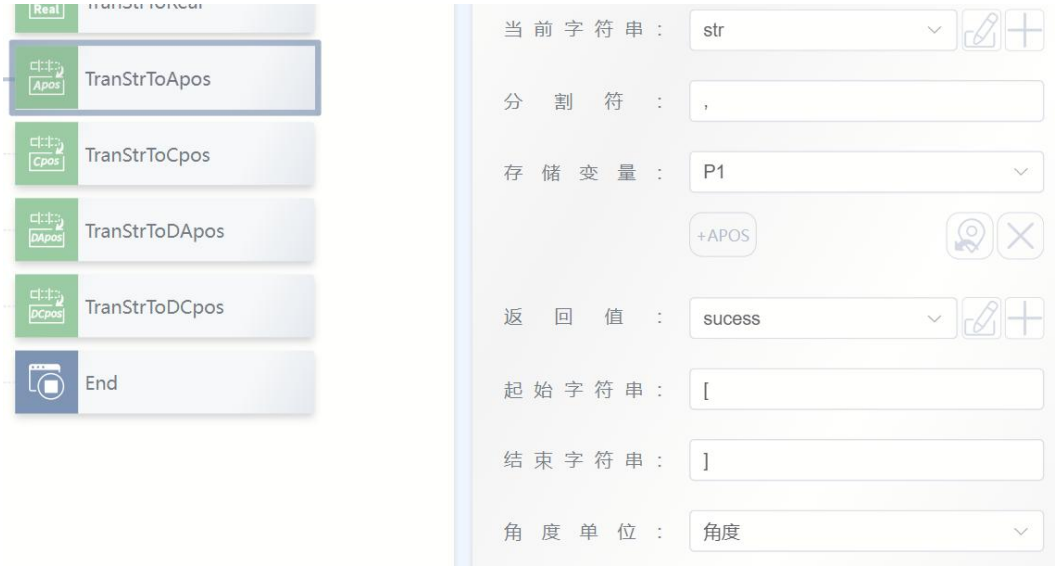
This instruction is used to convert a string variable to a real variable.



Arguments	Instructions
Current string	The string to be converted
Splitter	Space symbol between numeric values
Storing variables	The converted real variable
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Add starting string
Ending string	Add Ending string

10.12.7 TranStrToApos

This instruction is used to convert a string variable to an Apos variable.

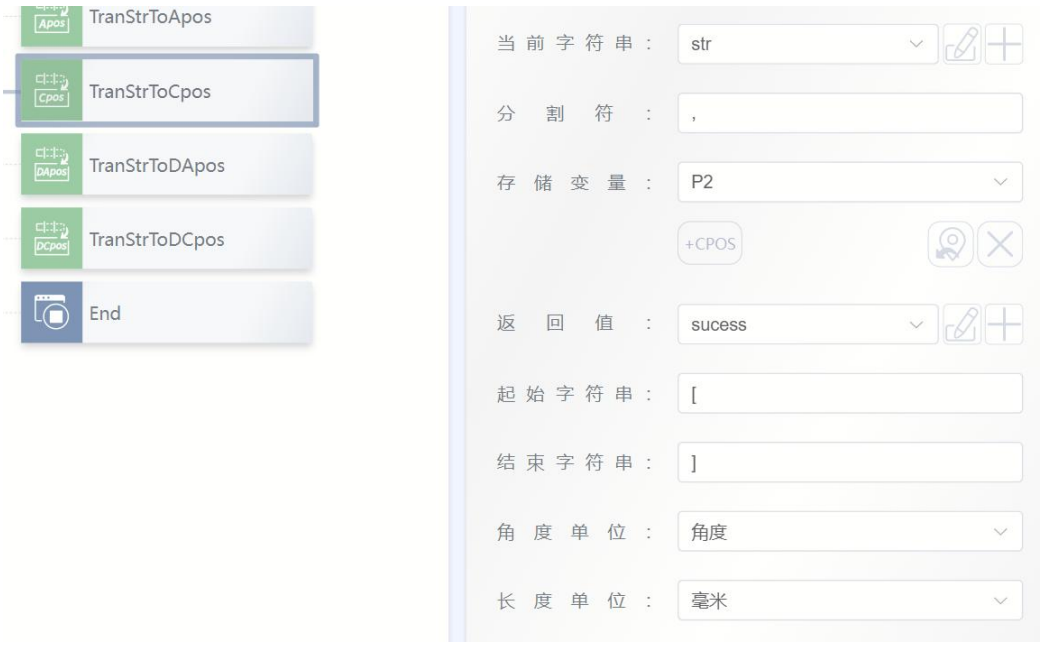


Arguments	Instructions
Current string	The string to be converted
Splitter	Space symbol between numeric values
Storing	The converted Apos variable

variables	
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Add starting string
Ending string	Add Ending string
Angle units	APOS angular numerical unit form

10.12.8 TranStrToCpos

This instruction is used to convert a string variable to a Cpos variable.



Arguments	**Instructions**
Current string	The string to be converted
Splitter	Space symbol between numeric values
Storing variables	Converted Cpos variables
Returned	The return value variable of whether the operation was successful, with

value	a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Add starting string
Ending string	Add end string
Angle units	Numeric unit form of Angle in Cpos
Unit of length	Numeric unit of length in Cpos

10.12.9 TranStrToDApos

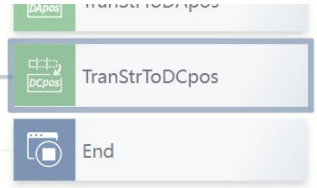
This instruction is used to convert a string variable to a DApos variable.

Arguments	**Instructions**
Current string	The string to be converted
Splitter	Space symbol between numeric values
Storing variables	The converted DApos variable
Returned value	The return value variable of whether the operation was successful. A return value of 0 indicates success, and a return value of 1 indicates failure
Starting	Add starting string

string	
Ending string	Add Ending string
Angle units	Numeric unit form of Angle in DApos

10.12.10 TranStrToDCpos

This instruction is used to convert a string variable to a DCpos variable.



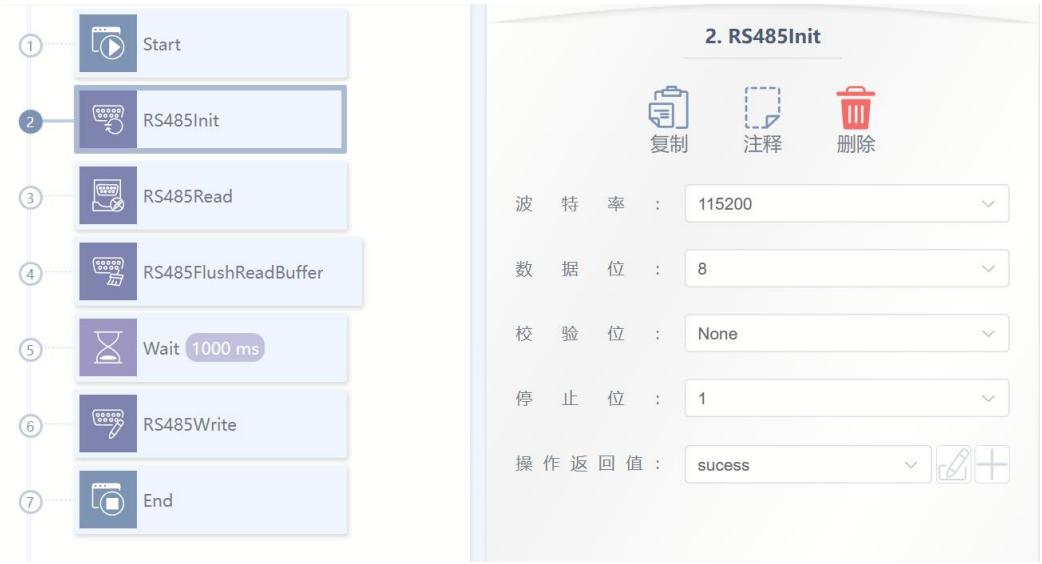
Arguments	**Instructions**
Current string	The string to be converted
Splitter	Space symbol between numeric values
Storing variables	Converted DCpos variable
Returned value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure
Starting string	Add starting string
Ending string	Add Ending string
Angle units	Numeric unit form of Angle in DCpos

Unit of length	Numeric unit form of length in DCpos

10.13 RS485

10.13.1 RS485Init

This instruction is used to initialize the RS485 port on the control cabinet.



Parameters	Instructions
Baud Rate	RS485 communication baud rate
Data Bits	RS485 communication data bit
Check bit	RS485 communication check bit
Stop bit	RS485 communication stop bit
Operation Return value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure

10.13.2 RS485Read

This instruction is used to read the RS485 data on the control cabinet.

Parameters	Instructions
Int array	The variable in which the data to be read is stored
Timeout duration	Read timeout period. If no read period is reached, an error is reported
Number of Bytes	Length of data read in bytes
The return value operation	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure

10.13.3 RS485FlushReadBuffer

This command is used to clear the cache data read by the RS485 port on the control cabinet. Generally, it can be cleared after reading to ensure that the next read is normal. Or it can be unclear and wait for several times to process the data together after receiving.

10.13.4 RS485Write

This instruction is used to send data on the RS485 on the control cabinet.

Parameters	Instructions
Int array	The variable in which the data to be sent is stored
Number of bytes	Length of bytes of data sent
Operation return value	The return value variable of whether the operation was successful, with a return value of 0 indicating success and a return value of 1 indicating failure

10.14 Mathematical operation function

In the "IF" instruction and "... =..." In the instruction, a mathematical or string operator function may be used. This section describes the "mathematical functions" that can be used.

10.14.1 sin

Sine trigonometric function.

- Parameter 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.2 cos

Cosine trigonometry.

- Parameter 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.3 tan

Tangent trigonometry.

- Argument 1: int or real type variable or constant.
- Function return: constant of type real.

10.14.4 asin

Arcsine trigonometry.

- Parameter 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.5 acos

Inverse cosine trigonometry.

- Argument 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.6 atan

Atan trigonometric function.

- Argument 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.7 atan2

X/Y inverse tangent value function that returns the radian value of the X axis to the point (x, y).

- Argument 1: int or real type variable or constant.
- Parameter 2: Variable or constant of type int or real.
- Function return value: constant of type real.

10.14.8 sinh

Hyperbolic sine function.

- Parameter 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.9 cosh

Hyperbolic cosine function.

- Argument 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.10 tanh

Hyperbolic tangent function.

- Argument 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.11 log

Natural logarithm function.

- Parameter 1: Variable or constant of type int or real.
- Function return value: constant of type real.

10.14.12 log10

Log function with base 10.

- Argument 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.13 sqrt

Square root function.

- Argument 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.14 exp

Exponential function with base e.

- Parameter 1: Variable or constant of type int or real.
- Function return value: constant of type real.

10.14.15 pow

Exponential function.

- Parameter 1: int or real type variable or constant, base number.
- Parameter 2: int or real variable or constant, exponent.
- Function return value: constant of type real.

10.14.16 deg

Radian to angular function.

- Parameter 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.17 rad

Angular to radian function.

- Parameter 1: int or real type variable or constant.
- Function return value: constant of type real.

10.14.18 fmod

Complementary function.

- Argument 1: int or real type variable or constant, dividend.
- Parameter 2: int or real variable or constant, divisor.
- Function return: constant of type real.

10.14.19 floor

Take the whole function down.

- Argument 1: int or real type variable or constant.
- Function return: constant of type int.

10.14.20 random

Take random integers between 2 arguments.

- Parameter 1: Variable or constant of type int.
- Parameter 2: Variable or constant of type int.
- Function return: constant of type int.

10.15 String function

10.15.1 byte

Take the ASCCII code of the character in the NTH bit of the string.

- Parameter 1: A string variable or constant.
- Parameter 2: Variable or constant of type int.
- Function return: constant of type int.

10.15.2 char

Returns the character corresponding to the ASCCII code.

- Parameter 1: Variable or constant of type int.
- Function return: constant of type string.

10.15.3 find2

Return string neutron string set.

- Parameter 1: string type variable or constant.
- Parameter 2: string variable or constant
- Function return: constant of type int. (-1 if no corresponding character or string can be found)

10.15.4 findEnd

String reverse lookup command to find the last occurrence of the specified string in a string

and return the index number.

- Parameter 1: The searched source string, a variable or constant of type string.
- Parameter 2: The specified string to be looked for, a variable or constant of type string.
- Function return value: the index number after searching, variable of type int.

10.15.5 format

Format string instruction, by passing a reasonable format control character in argument 1, followed by any number of arguments to fill this format control character, and return the filled data.

- Parameter 1: string format, string type variable or constant.
- Parameter 2: The parameter of the format control character to be filled, a string/real/int variable or constant.
- Parameter 3: Parameter of formatting control character to be filled, variable or constant of type string/real/int.
-

... (The argument is unlimited, the total length as long as it does not exceed the length of a single instruction string)

- Function return value: The number of successfully divided and saved to the array, variable of type int

The format string starts with %. The following uses are supported: %c - takes a number and converts it to the corresponding character %d in the ASCII table, %i - Accepts a number and converts it to a signed integer format %o - Accepts a number and converts it to an octal number format %u - Accepts a number and converts it to an unsigned integer format %x - Accepts a number and converts it to a hexadecimal number format, using the lowercase letter x %X - Accept a number and convert it to hexadecimal number format, using the uppercase letter X %f - Accept a number and convert it to floating-point format %s - Accept a string and format the string with the parameters given Sample: format("%%c: %c", 83) Output S format("%+d", 17.0) Output +17 format("%05d", 17) Output 00017 format("%o", 17) Output 21 format("%u", 3.14) Output 3 format("%x", 13) Output d format("%X", 13) Output D format("%6.3f", 13) Output 13.000 format("%s", "monkey") Output monkey format("%10s", "monkey") Output monkey

10.15.6 getAt

A single string fetch command that takes the string data of a certain bit and returns the retrieved data.

- Parameter 1: The string to be intercepted, a variable or constant of type string.
- Parameter 2: The bit to be obtained, a variable or constant of type int.
- Function return value: The obtained string, a variable of type string.

10.15.7 gsub

Search for a substring within string s and replace a with string b.

- Parameter 1: A variable or constant of type string.
- Parameter 2: string variable or constant.
- Parameter 3: string variable or constant.
- Return value: A string constant.

10.15.8 len

Calculate the string length.

- Parameter 1: string type variable or constant.
- Function return: constant of type int.

10.15.9 left

String fetch Left instruction to intercept the specified number of strings starting from the left side of the string and return the intercepted data.

- Parameter 1: The string to be intercepted, a variable or constant of type string.
- Parameter 2: The number to be intercepted, a variable or constant of type int.
- Function return value: the intercepted string, a variable of type string.

10.15.10 lower

Returns the string in lower case format.

- Parameter 1: A variable or constant of type string.

- Return value: A string constant.

10.15.11 right

String fetch Right instruction to intercept a specified number of strings starting from the right side of the string and return the intercepted data.

- Parameter 1: The string to be intercepted, a variable or constant of type string.
- Parameter 2: The number to be intercepted, a variable or constant of type int.
- Function return value: the intercepted string, a variable of type string.

10.15.12 reverse

String reversal instruction, which returns after reversing a string.

- Parameter 1: The string to be reversed, a variable or constant of type string.
- Function return value: inverted string, type string variable.

10.15.13 strcmp

String comparison instruction that returns the ASCII difference between the first different character.

- Parameter 1: string data for comparison, a variable or constant of type string.
- Parameter 2: string data to be compared, string variable or constant.
- Function return value: Returned ASCII value, variable of type int.

10.15.14 trimLeft

Trimleft removes the whitespace left of the string and returns the modified string data.

- Parameter 1: The string to be pruned, a string variable or constant.
- Function return value: pruned string, variable of type string.

10.15.15 trimRight

String right trim command, remove the space on the right of the string, and return the modified string data.

- Parameter 1: The string to be pruned, a string variable or constant.

- Function return value: pruned string, variable of type string.

10.15.16 upper

Returns the uppercase format of the string.

- Parameter 1: A variable or constant of type string.
- Return value: A string constant.

10.15.17 IToStr

Integer to string instruction, converts integer data to string type data, and returns the converted string.

- Parameter 1: Integer data to be converted, variable or constant of type int.
- Function return value: Converted string data, variable of type string.

10.15.18 RToStr

Real type parameter to string instruction, converts real data to string type data, and returns the converted string.

- Parameter 1: To be converted to REAL data, REAL type variable or constant.
- Function return value: Converted string data, type string variable.

10.15.19 StrToI

String to integer instruction, converts string data to integer type data, and returns the converted integer data.

- Parameter 1: The string data to be converted, a string variable or constant.
- Function return value: Converted integer data, variable of type int.

10.15.20 StrToR

String to Real data instruction, converts string data to real data, and returns the converted real number.

- Parameter 1: string data to be converted, a string variable or constant.
- Function return value: Converted REAL data, REAL type variable.

10.15.21 Append

The Append directive is used to append strings.

- Argument 1: String to append 1.
- Parameter 2: String to append 1.
- The function returns: string 1+ string 2 of type string.

11. Appendix

11.1 Error Codes

There are currently six message levels for the robot, and the fourth digit of the error code indicates the error level.

Serial Number	Error level
0	System occupancy
1	Tips
2	Warnings
3	General Errors
4	Serious errors
5	Fatal mistake

- When there is a general error or more, the robot will lose power and stop;
- When there is a warning level error, the robot will slow down and stop;
- If multiple errors occur at the same time, execute according to the highest error level;
- There will be only one error code for the same type of error, but the error content will be specifically displayed on the instructor.

Error code	Error description
40000000-4FFFFFFF Memory Pool	
4F040000	Failed to add memory region, region already exists /MemPoolZoneIsExist
4F040001	Failed to release the memory block because the corresponding memory block /MemPoolFreeMemFailed is not found
50000000-5FFFFFFF Indicates that the robot body is related	
50010000	Robot power-on tips/Robot_Power_On
50010001	Robot power-off prompt/Robot_Power_Off
50010002	Robot encoder Calibration tip/Robot_Encoder_Reference
50030003	Robot state Switch timeout/Robot_Change_State_Timeout
50040004	Axis state exception/Robot_Axis_State_Error

50030005	Position singular when clicked/Robot_Jog_Position_Singular
50010006	Reset/Robot_Reset
50030007	Reset timeout/Robot_Reset_Timeout
50030008	Joint position overrun/Joint_Position_Limit
50030009	End position overrun/Cart_Position_Limit
5003000A	Joint expected position jump/Joint_Position_Jump
5003000B	Joint output torque jump/Joint_Torque_Jump
5003000C	The joint tracking error is too large/Tracking_Error_Limit
5003000D	Joint speed over limit/Joint_Over_Speed
5003000E	Joint collision detection trigger/Joint_Collision
5003000F	Cannot perform valid calculation for joint collision detection/Cant_Compute_Joint_Collision
50030010	Cannot efficiently calculate for end collision detection/Cant_Compute_Cart_Collision
50030011	End collision detection triggers/Cart_Collision
50030012	End speed overrun/Cart_Over_Speed
50030013	Error in drag/Drag_Error
50030014	Error while dragging Stop/Drag_Stop_Error
50030015	Cannot do end point/Cant_Cart_Jog
50020016	Error/Motion_Controller_Reset_Error occurred while resetting motion planner
50020017	Motion planner sets initial position error/Motion_Controller_Set_Init_Position_Error
50020018	Motion Planner adds instruction error/Motion_Controller_Add_Instruction_Error
50030019	Emergency stop/Estop
5002001A	Configuring parameters, this operation is prohibited/Parameter_Configuring
5003001B	Error occurred during parameter configuration/Parameter_Configure_Error
5002001C	Scram on power-on press/Switch_On_Estop
60000000-6FFFFFFF Robot Model related error	
60020000	Motion Planner path calculation error/MCI_Path_Compute_Failed
60020001	Motion Planner run error/MCI_Path_Run_Failed
60020003	Node data conversion to json/Software_Data_2_json_Failed
60020004	Failed to obtain shared memory node/Software_Get_Node_Failed
60030003	Unable to get the inverse of the robot's velocity Jacobian

	matrix/Can_Not_Get_Inv_Vel_Jacobian
60030004	Unable to obtain the inverse of the robot's force Jacobian/Can_Not_Get_Inv_Force_Jacobian
60030005	Unable to obtain the forward kinematic position of the robot/Can_Not_Get_Fwd_Kine_Pos
60030006	Unable to obtain the forward kinematic velocity of the robot/Can_Not_Get_Fwd_Kine_Vel
60030007	The inverse kinematic position of the robot cannot be obtained/Can_Not_Get_Inv_Kine_Pos
60030008	Unable to obtain the robot's inverse kinematic velocity/Can_Not_Get_Inv_Kine_Vel
60030009	The wrong pos mode/Wrong_Pos_Mode_Is_Set for the robot is set
6003000A	Joint overrun/Jnt_Pos_Out_Of_Range
6003000B	Unable to obtain robot inverse dynamics/Can_Not_Get_Inv_Dyn
6003000C	Unable to obtain robot joint equivalent moment of inertia/Can_Not_Get_Inertia_Trq
6003000D	Unable to obtain robot joint equivalent heavy torque/Can_Not_Get_Gravity_Trq
6003000E	Unable to obtain robot joint equivalent CoriolIS_TRQ/Can_Not_Get_Coriolis_Trq
6003000F	Inertia matrix/Can_Not_Get_Inertia_Matrix for robot dynamics model cannot be obtained
60030010	The gravity matrix/Can_Not_Get_Gravity_Matrix of the robot dynamics model could not be obtained
60030011	The Coriolis_matrix/Can_Not_Get_Coriolis_Matrix of the robot dynamics model cannot be obtained
61010000	Unknown file/File_Unknown_Error
61010001	File parsing error/File_Parse_Failed
61010002	File loading error/File_Load_Failed
61010003	File conversion error for specific format/Yaml_To_Json_Failed
61010004	Format specific file write error/Yaml_Write_From_Json_Failed
70000000-71000000 Calibration error	
70020000	Fit matrix Regression_Matrix_Not_Full_Rank
70020001	Three points of calibration collinearity/Three_Points_Collinearity
71000000-75FFFFFF Generic error	
71020000	Robot initial position unknown/Robot_Init_Position_Unkown

71020001	Initial condition is insufficient, waiting to be replenished, no error/Init_Condition_Not_Enough
71020002	For relative motion, the type of reference coordinate system entered does not exist/Ref_Coor_Not_Exist
71020003	Transition type unknown/Zone_Type_Unkown
71020004	The type of the point is unknown/Point_Type_Unkown
71020005	Arc type unknown/Circle_Type_Unkown
71020006	The Move instruction queue is full/Size_Out_Of_Range
71020007	Speed is not positive/Speed_Negative
71020008	Unable to create path/Can_Not_Create_Path
71020009	Index out of range/Index_Out_Of_Range
7102000A	Solution failed/Solve_Failed
7102000B	Trajectory planning failure/Can_Not_Planning
7102000C	Move type does not exist/Mov_Ins_Type_Not_Exist
7102000D	Move type mismatch/Mov_Ins_Type_Not_Match
7102000E	Trigger type mismatch/Trigger_Type_Not_Match
7102000F	Trigger corresponding Move instruction Id does not exist/Trigger_Mov_Id_Not_Exist
76000000-77FFFFFF Welding error	
76020000	Swing type does not exist/Welding_Type_Not_Exist
76020001	The swing is negative/Welding_Amplitude_Negative
76020002	The swing frequency is negative/Welding_Frequency_Negative
76020003	The swing Angle is negative/Welding_Azimuth_Negative
76020004	The operating Angle is negative/Welding_Elevation_Negative
76020005	The left residence time is negative/Welding_Left_Stop_Time_Negative
76020006	The right dwell time is negative/Welding_Right_Stop_Time_Negative
76020007	Low frequency/Welding_Frequency_Too_Low
76020008	High frequency/Welding_Frequency_Too_High
76020009	The stay time is too long/Welding_Stop_Time_Too_Long
7602000A	Azimuth too large/Welding_Azimuth_Too_Large
7602000B	Path type does not exist/Path_Type_Not_Exist
7602000C	The weld direction is the same as the Z direction of the current tcp, and the swing direction/Can_Not_Determine_Swing_Direction cannot be determined

7602000D	Compensation mode does not exist/Compensation_Type_Not_Exist
7602000E	Compensation value update failure/Updata_Compensation_Failed
78000000-7FFFFFFF dynamic algorithm error	
78030000	Input parameter dimension does not match the robot/Size_Mismatch
78030001	External force estimator initialization failed/TrqEstimator_Init_Failed
78030002	The force estimator is not set to an initial state/TrqEstimator_Not_Inited
78030003	The Kalman filter built into the force estimator cannot update the output/Can_Not_Update_Kalman_Filter
78030104	Cannot get the joint force estimator estimated/Can_Not_Get_Estimated_Jnt_Trq
78030105	Cannot get the joint acceleration estimated by the force estimator/Can_Not_Get_Estimated_Jnt_Acc
78030106	Collision detector not initialized successfully/Collision_Detector_Not_Inited
78030107	Unable to get the status of the collision detection/Can_Not_Get_Collision_State
78030108	Admittance controller was not successfully initialized/Admittance_Controller_Not_Inited
78030109	Admittance controller parameters are incorrectly set/Wrong_Admittance_Paras
7803010A	Unable to update output of joint admittance instruction program/Can_Not_Update_Teach_AD_Jnt
7803010B	End space axle lock unsuccessful initialization/Cart_Lock_Not_Inited
7803010C	Unable to set end lock axis orientation/Can_Not_Set_Cart_Lock_Dir
7803010D	Unable to obtain end impedance force of end shaft lock/Can_Not_Get_Cart_Lock_Resis_Wrench
7803010E	Unable to obtain impedance force at end lock shaft conversion to joint end/Can_Not_Get_Cart_Lock_Resis_JntTrq
7803010F	Unable to update the output of the instructor/Can_Not_Update_Teach
78030110	Out of joint limit/Teach_Over_JP_Limit in drag and drop mode
80000000-8FFFFFFF triggers security rule	
80030000	Joint tracking error overrun trigger/Axis_Tracking_Error_Detection
80030001	Joint collision detection trigger/Axis_Collision_Detection
80030002	Joint position restriction trigger/Axis_Position_Limit_Detection
80030003	Joint speed limit trigger/Axis_Velocity_Limit_Detection
91000000-91FFFFFF Engineering Management	

91010000	The WHILE control expression is empty/Ins_Sys_Proj_Manager_While_Expr_Is_Empty
91010001	IF the control expression is empty/Ins_Sys_Proj_Manager_If_Expr_Is_Empty
91010002	The ELSEIF control expression is empty/Ins_Sys_Proj_Manager_Else_If_Expr_Is_Empty
91010003	The ELSE control is followed by ELSEIF/Ins_Sys_Proj_Manager_Else_Is_Not_last
91010004	Unknown operator/Ins_Sys_Proj_Manager_Unkown_Operator_Type
91010005	The variable name of the data is not a string type/Ins_Sys_Proj_Manager_Data_Var_Is_Not_String
91010006	The wait time argument is not an integer/Ins_Sys_Proj_Manager_Wait_Time_Is_Not_Integer
91010007	Control parameter is invalid/Ins_Sys_Proj_Manager_Param_Invalid
91010008	Invalid control type/Ins_Sys_Proj_Manager_Widget_Type_Invalid
91010009	Project starts to run/Ins_Sys_Proj_Exec_Proj_To_Run
9101000A	Project Stops running/Ins_Sys_Proj_Exec_Proj_To_Idle
9101000B	Task Status error/Ins_Sys_Proj_Exec_Task_State_Error
91021007	Failed to open configuration file/Ins_Sys_Proj_Manager_Open_Config_File_Failed
91011008	Failed to save global variables/Ins_Sys_Proj_Manager_Save_Global_User_Data_Failed
91011009	Failed to get global variables/Ins_Sys_Proj_Manager_Get_Global_User_Data_Failed
9101100A	Failed to save project variables/Ins_Sys_Proj_Manager_Save_Proj_User_Data_Failed
9101100B	Failed to get project variables/Ins_Sys_Proj_Manager_Get_Proj_User_Data_Failed
9101100C	Failed to save project/Ins_Sys_Proj_Manager_Save_Proj_Failed
9102100D	Failed to read project file/Ins_Sys_Proj_Manager_Get_Proj_File_Failed
9102100E	Failed to read lua file/Ins_Sys_Proj_Manager_Get_Lua_File_Failed
92000000-92FFFFFF declaration	
92020000	Array variable index out of range/Ins_Sys_Ins_Def_Index_Out_Of_Range
92020001	Failed to find variable by variable name/Ins_Sys_Ins_Def_Get_Var_By_Name_Failed
92020002	Unknown variable type/Ins_Sys_Ins_Def_Unknown_VTType
92020003	Failed to find IO port/Ins_Sys_Ins_Def_Fined_IO_Failed
92020004	Request parameter error/Ins_Sys_Ins_Invalid_Request_Param

93000000-93FFFFFF Request processing	
93010000	Failed to set shared memory node/Ins_Sys_Req_Processor_Set_Node_Value_Failed
93010001	Failed to convert CPOS to APOS/Ins_Sys_Req_Processor_CPOS2APOS_Failed
93010002	Failed to convert APOS to CPOS/Ins_Sys_Req_Processor_APOS2CPOS_Failed
93010003	Point data calculation failed/Ins_Sys_Req_Processor_Pos_Transform_Failed
93010004	Motion kernel state error/Ins_Sys_Req_Processor_MCKernel_State_Error
93010005	Calibration failure/Ins_Sys_Req_Processor_Calibrate_Failed
94000000-94FFFFFF Message push	
94010002	The subscribed topic does not exist/Ins_Sys_Publish_Topic_Is_Not_Exist
94010003	Failed to open the topic configuration file/Ins_Sys_Publish_Open_Topic_Config_File_Failed
94010004	Parsing the topic profile failed/Ins_Sys_Publish_Parse_Topic_Config_File_Failed
94010005	topic name duplicate/Ins_Sys_Publish_Topic_Name_Repeat
94010006	The memory node/Ins_Sys_Publish_Find_Node_Failed for the topic was not found
95000000 - 95FFFFFF websocket	
96000000-96FFFFFF engineering run	
96010000	Resolve to unknown instruction/Ins_Sys_Proj_Exec_Ins_Parse_Unknown
96020001	Failed to load instruction/Ins_Sys_Proj_Exec_Ins_Load_Failed
96020003	The motion kernel state does not support this directive/Ins_Sys_Proj_Exec_Mc_Kernel_State_Not_Macth
96020004	The project state does not support this directive/Ins_Sys_Proj_Exec_State_Not_Macth
96020005	Invalid project control directive/Ins_Sys_Proj_Exec_Invalid_Proj_Cmd
96020006	Project data loading failure/Ins_Sys_Proj_Exec_Load_Proj_Data_Failed
96020007	Project loading failure/Ins_Sys_Proj_Exec_Load_Proj_Failed
96020008	Invalid control ID/Ins_Sys_Proj_Exec_Invalid_Start_Widget at the start of the project
97000000-97FFFFFF task run	

97020000	Too many addDo instructions/Ins_Sys_Task_Exec_Too_Manay_Add_Do_Ins
97020001	The jump control does not exist/Ins_Sys_Task_Exec_Jump_Widget_Is_Not_Exist
97020002	IO port number parameter is invalid/Ins_Sys_Task_Exec_IO_Port_Num_Illegal
97020003	lua Expression execution failed/Ins_Sys_Task_Exec_Lua_Run_Expression_Failed
97020004	Invalid task control directive/Ins_Sys_Task_Exec_Invalid_Task_Cmd
97020005	Failed to execute AddDo command/Ins_Sys_Task_Exec_Run_Add_Do_Ins_Faied
97020006	Instructions waiting to be executed on the command queue failed/Ins_Sys_Task_Exec_Run_Ins_Pending_Queue_Faied
97020007	Unknown instruction/Ins_Sys_Task_Exec_Run_Unknown_Ins was executed
97020008	lua loading instruction failed/Ins_Sys_Task_Exec_Lua_Load_Ins_Failed
97020009	lua command execution failed/Ins_Sys_Task_Exec_Lua_Run_Ins_Failed
9702000A	Failed to write instruction to motion kernel/Ins_Sys_Task_Exec_Write_Ins_2_MC_Failed
9702000B	Failed to update the AddDo command status/Ins_Sys_Task_Exec_Update_Add_Do_Ins_Failed
9702000C	Failed to register variables to lua/Ins_Sys_Task_Exec_Rigister_Var_2_Lua_Failed
9702000D	lua initialization failed/Ins_Sys_Task_Exec_Lua_Init_Failed
9702000E	lua configuration script load initialization failed/Ins_Sys_Task_Exec_Lua_Config_Script_Load_Failed
9702000F	Unknown user variable type/Ins_Sys_Task_Exec_Unknown_VTType
97020010	Failed to create Path/Ins_Sys_Task_Exec_Create_Path_Failed
97020011	Path calculation failed/Ins_Sys_Task_Exec_Compute_Path_Failed
97020012	Running Path failed/Ins_Sys_Task_Exec_Run_Path_Failed
97020013	OnDistance cannot associate the MovJ directive/Ins_Sys_Task_Exec_On_Distance_Invalid
97020014	The argument/Ins_Sys_Task_Exec_Var_Is_Invalid is invalid

11.2 User level and permissions

Categories	Feature	user	admin
Engineering	New Construction	✔	✔
	Toggle	✔	✔
	Save	✔	✔

	Copy	✔	✔
	Download	✔	✔
	Delete	✔	✔
	Import	✔	✔
	Autorun	✔	✔
	Stop	✔	✔
	Single step run	✔	✔
	Run pointer	✔	✔
	Single task vs. multitask switch	✔	✔
Visual programming	Control Viewing	✔	✔
	Drag and drop instructions	✔	✔
	Add instructions	✔	✔
	Instruction checked	✔	✔
	Instruction Properties Edit	✔	✔
	Instruction copy	✔	✔
	Instruction delete	✔	✔
	Tree instruction expansion and contraction	✔	✔
	Instruction attribute edit check	✔	✔
	Conditional expression check	✔	✔
	goto class instruction target value check	✔	✔
	Check result prompt information	✔	✔
Pose	Add pose	✔	✔
	Remove pose	✔	✔
	Copy pose	✔	✔
	Add pose from mov class control	✔	✔
	Update pose from mov class control	✔	✔

Variables	Add variables	✔	✔
	Remove variables	✔	✔
	Edit variables	✔	✔
	Variable Display	✔	✔
	Real-time running variables	✔	✔
	Adds a variable of the specified type from the control properties	✔	✔
Settings	Basics	✘ (without permission)	✘ (not authorized)
	Mechanical - Mount	✘ (without permission)	✔
	Mechanical - relative to the world coordinate system	✘ (without permission)	✔
	Mechanical -DH	✘ (no entry)	✘ (no entry)
	Safety - Joint/end limit	✘ (no permission)	✔
	Security - Other	✘ (without permission)	✔
	Sport - Auto mode	✘ (Without permissions)	✔
	Sport - Manual mode	✘ (without permission)	✔

	Motion-Servo	✘ (no entry)	✘
	Debug	✘	✘
3D simulation	Simulation display	✔	✔
	Switching perspective	✔	✔
	Clear track lines	✔	✔
	Return to zero position	✔	✔
	Return to pack position	✔	✔
	Switch coordinate system	✔	✔
	Teach mode configuration	✔	✔
	Automatic mode	✔	✔
	Manual mode	✔	✔
	Close node action	✔	✔
	End dotting	✔	✔
	I/O configuration	✔	✔
	Peripheral	✔	✔
Journal	View	✔	✔
	Download	✔	✔
Plugins	List of welding process templates	✔	✔
	Adding Templates	✔	✔
	Edit Template	✔	✔
	JOB Number Selection	✔	✔
Monitor	Surveillance system	✘	✘
	Specify monitoring data	✘	✘
Debugging	path data sending	✘	✘
	Debug data cache	✘	✘
Configuration	Change configuration values	✘	✘
	Change the configuration structure	✘	✘

User	Register a new user	✘	✔
	Delete a user	✘	✔
Bus	Register editing	✘	✔
Error message	Clear errors	✔	✔
	Reset	✔	✔
	Live log	✔	✔
Additional features	Undo & Redo	✔	✔
	Reload configuration	✔	✔
	Refresh the page	✔	✔
	Module window maximization	✔	✔
	Close module window	✔	✔
	On-line setup	✔	✔
	On-line state	✔	✔
	Lock window	✔	✔
	Switch between English and Chinese	✔	✔
	Trace ID related functions	✘	✘

11.3 Claims

DECALARATION OF INCORPORATION

According to the following EU Directive(s)

Machinery Directive: 2006/42/EC

Electromagnetic Compatibility Directive: 2014/30/EU

Manufacturer: NANJING ESTUN CODROID TECHNOLOGY CO., LTD.

Address: 5/F, Building 1, Jiangning Double Innovation Base, National University Science Park, Southeast University, No.33, Southeast University Road, Jiangning District. Nanjing, 211102 Jiangsu, P. R. China

Declares that the machine described here after

Product name: Collaborative Robots

Model(s): S3-60 Eco,S5-90 Eco,S10-140 Eco,S20-180 Eco
S3-60 Pro,S5-90 Pro,S10-140 Pro,S20-180 Pro

Serial No.:

Provided that it is used and maintained in accordance with the general accepted codes of good practice and the recommendations of the instruction manual, meet the essential safety and health requirements of the Machinery Directive

Person authorized to compile the technical file:

Name: ESTUN Robotics Europe AG

Address: Graben Strasse 25,6340 Baar,Switzerland

We confirm that:

a) The specific technical documents which we provided pursuant to Appendix VII Part B
b) The assembly instructions which are provided pursuant Appendix VI
c) The declaration of incorporation which we provided pursuant to Appendix II Part 1 Section B of Directive 2006/42/EC

Upon justified request, we provide specific documents regarding the products listed above within an adequate period. The document will be made available via e-mail

The following essential health and safety requirements are executed and observed according to annex I of the directive specific above:

1.1.1, 1.1.2, 1.1.3, 1.1.5, 1.1.6, 1.3.1, 1.3.2, 1.3.3, 1.3.4, 1.3.7, 1.3.8, 1.3.9, 1.4.1, 1.4.2.1, 1.4.3, 1.5.2, 1.5.4, 1.5.5, 1.5.6, 1.5.8, 1.5.9, 1.5.11, 1.5.14, 1.5.15, 1.6.1, 1.6.2, 1.6.4, 1.6.5, 1.7.1, 1.7.2, 1.7.3, 1.7.4

For the most specific risks of this machine, safety and compliance with the essential requirements of the Directive has been based on elements of:

EN ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1: 2018	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
EN IEC 61000-6-2:2019	Electromagnetic compatibility (EMC) - Part 6-2: Generic standards - Immunity for industrial environments

EN IEC 61000-6-4:2019	Electromagnetic compatibility (EMC) - Part 6-4: Generic standards - Emission standard for industrial environment

The relevant technical documentation is compiled in accordance with part B of Annex VII 2006/42/EC, and are available by reasoned requested by national authorities via proper methods.
It must not be put into service until the final machinery into which it is to be incorporated has been declared in conformity with the provisions of the above Directives.

Print Name:	Ge QingQing	Position:	Chief Engineer of Robot Development
Signature:	葛庆卿	Date:	2024.9.6
Place:	5/F, Building 1, Jiangning Double Innovation Base, National University Science Park, Southeast University, No.33, Southeast University Road, Jiangning District. Nanjing, 211102 Jiangsu, P. R. China		

12. Spare parts list

Spare parts and consumable parts of the system, the items described in the table and their service life are for reference only, the actual state depends on the frequency of use and maintenance.

No.	Name	Model number	Manufacturer	Durability
1	Key Switch	NP6-22Y2	CHINT	100000 times
2	Enable Button (black)	HE6B-M200BPN10	IDEC	100000 times
3	Button	HBGO12SH-10W/J/S(N)	HBAN	50000 times
4	Button	MP16S/F11-EDY -24V/B	CMP	100000 times
5	Button	NP6-22ZS	CHINT	100000 times
6	Switch Power Supply	LMFXXXX-20B48	MORNSUN	25000h
7	Fan	JC6025B24UC2	JENCE	70000h
8	Thermal Protect	BW-BCM-95°C	SAFTTY	10000cycles

13. Contact information

- **Company address:**

Nanjing Estun Kuzhuo Technology Co., LTD

1888 Jiyin Dadao, Jiangning Economic Development Zone, Nanjing

5F, Building 1, Jiangning Double Innovation Base, National University Science Park, Southeast University, 33 Dongnan University Road, Jiangning District, Nanjing City, Jiangsu Province

Nanjing Estun CoDroid Technology Co., Ltd

5/F, Building 1, Jiangning Double Innovation Base, National University Science Park, Southeast University,

No.33, Southeast University Road, Jiangning District. Nanjing, 211102 Jiangsu, P. R. China

National service hotline: 400-025-3336

- **European agency address:**

ESTUN Robotics Europe AGGraben

Strasse 25,6340 Baar,Switzerland

Specifications, ratings and dimensions of this product are subject to change without prior notice in order to improve the product.

For enquiries about the contents of this material, please contact the sales Department of the Company.