H Series Industrial Cobot Operation and Programming

User Manual

Guangzhou Auctech Automation Technology Ltd

Content

The Introduction

Introduction

Thank you for buying Robot.

This manual describes how to safely and correctly using TV robot operation and programming.

Please read this manual and other related manuals carefully before using the robot, and be sure to understand and be familiar with what is explained in the manual.

Version Relations

The software versions corresponding to this manual are as follows:

Equipment	Software/system version number
Demonstrator	HSPad3-V1.6.11
Controller	HSC3-V1.6.11
Drive and control in one	HSI_V1.6.11

What to Read

This manual is mainly for users of industrial robots. Please ensure that users have a basic knowledge of industrial robots.

Pictorial Description

The following figures may appear in this specification and represent the following meanings:

Graphic	Instructions
	Prompt information, prompt to explain the precautions of operation.
	Warning information, warning of possible losses.
	Example messages that demonstrate specific applications of the concepts described in the examples.

Revision records

Version number	Revision date	Revision Notes
A03	2024/7/3	Supplementary Appendix 13.5, SOCKET Error Codes Correct the IPO mode selection instructions
A04	2024/7/4	Update the dynamics installation pose setting instructions Update the dynamics body friction identification instructions Update the dynamic load configuration description Update the dynamics drag demo setup instructions Update the drag track recording instructions

1 Safety

1.1 General safety rules

Notes

- It is the responsibility of the operator to ensure that the robot production environment complies with and complies with national and local laws, regulations and regulations regarding safety. Robots are different from other equipment, due to its fast speed, wide range of motion, complex posture changes and other characteristics in the production process, it is likely to bring personal injury or equipment accidents, so we must pay special attention to safety matters, so as not to cause unnecessary losses.
- Before installation, operation and maintenance, please be sure to fully grasp the knowledge of robot equipment, safety information and all matters needing attention.

1.2 Instructions description of safety responsibility

Notes

Robot is committed to providing reliable safety information, but cannot ensure the completeness of safety information. Even if the operation is carried out according to the safety precautions, it is not guaranteed that the robot will not cause personal and property losses in complex situations and uncertain factors.

1.3 Emergency stop button

Warnings:

- The emergency stop button is the only means to trigger the emergency stop state of the robot, and it is also the most important means to ensure the safety of the operator and equipment in an emergency. Therefore, when the robot is installed, taught, operated and operated, please make sure that the emergency stop button (including the external emergency stop button connected to the robot system by the integrator) can work normally.
- The normal working condition of the emergency stop button is that the device can enter the emergency stop state after pressing the emergency stop button, and the emergency stop state can be relieved after resetting the emergency stop button. Most reset operations of the emergency stop button are carried out by means of rotation, and the direction of rotation is marked on the surface of the button. There are also some buttons that are reset by pulling up.
- The operator must confirm that the emergency stop button is not abnormal before the robot can be used. All operators must strictly implement the regulations.

Indication of position

The emergency stop button location is shown below.

Demonstrator:

Electric control cabinet:

Tips:

After pressing the emergency stop button, an alarm message will pop up in the information window of the teacher. Operators need to reset the emergency stop button first, and click the [Alarm Confirmation] button to confirm the alarm information before they can continue to operate the robot.

1.4 Precautions for robot and installation

Notes on Matters

- When considering robot safety, the robot body cannot be considered separately, but the whole robot production system environment should be considered comprehensively.
- Robots cannot perform production operations alone. Only after the end effector is installed and the peripheral equipment and equipment system are set up can the production operation be carried out.
- After installing the robot, be sure to set up a safety fence for the robot to prevent personnel from entering the robot working area during the operation of the robot, and to prevent accidents in which the control object is released or flown out due to the power supply or compressed air supply to the end effector is cut off.
- After setting up the safety fence, please be sure to place the robot's teaching device outside the safety fence. Only in this way can the movement of the whole robot be monitored.
- Make sure that no tools, fixtures, discarded objects or other obstacles are left in the robot's working area after the installation operation is completed.
- Before the robot is connected to the power supply, please confirm whether the voltage, frequency and cable specifications of the power supply meet the requirements.
- When installing the attached machine on the end-effector and

manipulator, be sure to follow the size and number of bolts specified in the relevant manual, and use the torque wrench to tighten according to the specified torque. In addition, no rusty or grime bolts should be used. Non-specified fastening and imperfect methods can cause bolts to loosen, which may lead to major accidents.

- The load on the wrist and the manipulator must be controlled within the allowable weight. Failure to comply with the provisions of the allowed handling weight may lead to abnormal movements or premature damage of mechanical components.
- It is strictly prohibited to supply power, compressed air, welding cooling water outside the specification, which will affect the action performance of the robot, and cause abnormal action or failure, damage and other dangerous situations.
- Please ensure that the robot controller and peripheral equipment are properly grounded. The robot controller and peripheral equipment with interference power should be grounded separately. At the same time, if there is a source of electromagnetic interference outside the controller, noise filtering equipment should be installed on the power supply line of the robot control cabinet.
- Although the electromagnetic wave interference is related to its type or intensity, there is no perfect countermeasures with the current technology. In the robot operation, power on medium conditions, should comply with the operation precautions. The recorded data will be lost due to electromagnetic waves, other noise and substrate defects. Therefore, please regularly back up the program files and register files to external storage media such as U disk.
- Situations where robots are not allowed:
  - In combustible environments.
  - In an explosive environment.

- Where there's a lot of radiation.
- Water or high humidity.

1.5 Precautions for robot teaching

Before demonstration	The teaching work should be completed by two people, one of them as a safety personnel, monitoring the whole teaching work, and stopping the robot in time through the safety device when danger is about to occur; The other person as a teaching staff, carefully and carefully complete the teaching task. The security personnel and the demonstration personnel must be the personnel with relevant training. Verify that the robot or peripheral equipment is not in a dangerous state and that nothing unusual has occurred. Confirm the range of motion of the robot, be sure not to get close to the robot or enter under the robot arm, to avoid the workpiece falling and endangering the life and safety of personnel. Confirm the safe passage of the teaching personnel so that they can safely evacuate in case of danger. Confirm the location and status of the emergency stop button (demonstrator, electric control cabinet). An emergency stop switch should be installed outside the safety fence where the entire movement range of the robot can be monitored, so that the safety personnel can easily and quickly press the emergency stop switch to stop the movement of the robot when they find the abnormal or dangerous movement of the robot.
In demonstration	Whether safety personnel or teaching personnel, must always monitor the robot for abnormal movement, the robot and its surroundings for collision, clamping, extrusion risk. It should be confirmed that the procedure or steps are correct before

proceeding with the operation. Editing procedures or steps incorrectly can lead to accidents.

Always pay attention to the movements of the robot and do not work with your back to the robot. Even if the action of the robot is slow to react, it will lead to the possibility of accidents.

After the Demonstration

Before confirming the teaching trajectory is correct, please ensure that there are no people and obstacles within the movement range of the robot. After confirming that there are no people and objects that may collide or interfere with the robot, run the teaching trajectory. At this time, the movement of the robot should be kept at low speed. When it is confirmed that the robot is correct, it can run at high speed.

1.6 Precautions for robot operation

Notes on Matters

Before starting the robot, please make sure that there are no people and items that may interfere with the operation of the robot in the robot operation area.

Before starting the robot, the function and name of each switch, display and signal of the robot should be clear, and ensure that the functions and equipment such as signal lights, external power sources, safety locks, emergency stop buttons are normal.

If there are multiple operators in the robot application system, please make sure that the operators make clear to each other the signal of the robot to start operation, and let other operators determine that it is safe to start the robot.

If the robot has not been running for a long time, please reduce the running rate of the robot and maintain the position where the emergency stop button can be pressed immediately to ensure that the robot can be stopped at any time.

If the robot will produce a lot of waste, metal dust, small debris, etc., in the process of working, please cover the robot and robot controller with a suitable

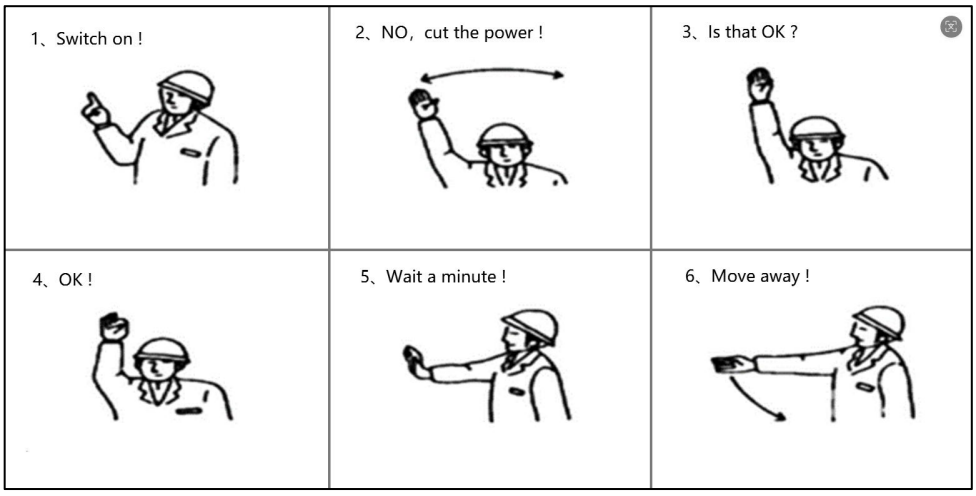
protective shell.

According to the setting place and the work content, the operator should write the starting method, operation method, solution method when abnormal and other relevant operation regulations and check list of the robot, and operate in accordance with the operation regulations. Operating only with the memory and knowledge of the operator can lead to accidents due to forgetting and errors.

Industrial robot gesture method

In large robot application systems, there are usually multiple operators working collaboratively. When communicating at a distance, shouts should be avoided as far as possible, because the noise and other factors in the production environment may not convey the meaning correctly, resulting in accidents. The intention should be correctly communicated through industrial robot gestures and other ways on site.

An example gesture method is shown in the following figure:



1.7 Notes on robot automatic operation

Notes on Matters

- Be sure to confirm that the workpiece has been firmly grasped before the robot runs automatically. If the grip of the robot gripper is not enough, the robot may throw the workpiece out during operation. When the gripper is

electromagnetically driven or pneumatically driven, design a corresponding failure protection system to ensure that the workpiece is not suddenly thrown out once the gripper loses its driving force.

- When the workpiece because of various circumstances, can not avoid being thrown out, to ensure that the safety fence, protective net, etc. can withstand the impact of the workpiece, can effectively protect personnel from injury.
- If the robot shows a fault and stops running, please check the fault shown and troubleshoise it according to the corresponding method to ensure that the robot is in a sound state, and restart the robot in the correct recovery sequence.

Warnings:

Never assume that the robot is not moving because the robot may be waiting for an input signal to continue moving, or it may be temporarily stopped because of the rhythm of the work. The operator can check whether the robot is in the "running" state by looking at the robot motion state on the interface of the teacher.

1.8 Precautions for operator operation

Description of matters

- Operators must wear work clothes, safety helmets and safety shoes before working.
- Before entering the safety fence, please confirm that all safety measures are complete and functioning properly.
- When there are dust, oil and other dirt in the robot production area, please be sure to clean the environment in and around the working area of the robot first to avoid causing people to fall down or polluting the robot during maintenance.
- Only after cutting off the power supply, the operator can enter the action

range of the robot to work.

- After running the robot for a period of time, all parts of the robot may heat up very high, especially the motor and reducer. Therefore, do not touch the above parts immediately after the robot stops.
- The operator should maintain the consciousness of escape at any time when working. It must be ensured that in an emergency, an immediate escape is possible.
- When switching on the power to the robot, make sure that there is no operator within the range of the robot.
- If the overhaul, repair, maintenance and other operations must be carried out in the state of power, at this time, the operation should be carried out in groups of two people. One person should maintain a position where the emergency stop button can be pressed immediately, and the other person should remain alert and quickly operate within the range of the robot's action. In addition, the operation should be performed only after the retreat path is confirmed.
- Never enter the robot range of motion when the robot is in operation, even if the robot does not appear to be moving.
- Make sure that there are other robots around the robot and take care to avoid them.
- Before maintenance work, make sure there is enough space around the robot to avoid potential collision with peripheral equipment. Please keep the peripheral equipment in a fixed state to prevent the sudden movement of these equipment from causing danger.
- If the peripheral equipment has compressed air or water, cut off the supply and remove residual pressure before servicing.
- If it is necessary to replace parts, the replacement parts should only be those approved by Robot.
- Before removing the motor of any joint, please support and secure the

robot arm with a suitable lifting device. After the motor is removed, the braking of the shaft will fail, and if it is not held in place by support, the shaft may fall down due to gravity, causing danger.

- After removing and replacing the motor, be sure to calibrate the zero before you can run the robot again.
- If you need to remove the motor, remove the motor after inserting the zero bolt and fixing the mechanical arm with a wooden block or crane to prevent it from falling off (the zero bolt and block are used to align with the original position, and can not be used to fix the machine). In addition, do not remove the motor while the manipulator is supported by human hands.
- The balance spring device is in a compressed state inside under normal conditions, which is extremely dangerous. It is strictly forbidden to disassemble or decompose (only for models equipped with balance spring devices).
- Pay attention to the electrical components in the electric control cabinet. Even if it is in the state of power off, the energy retained in the device is still dangerous.
- Do not alter or modify the robot. We will not be responsible for any modification of the robot.

1.9 Safety features of robots

Description of Characteristics

robots have the following safety characteristics, please be sure to fully understand and apply them to the design of safety measures.

- All emergency stop lines are hardware logic.
- The emergency stop button is installed on the teaching device and the electric control cabinet. If necessary, the emergency stop button can also be installed externally. Place these buttons where they are easily

visible and easy to press during production operations.

- The motion speed and motion error of the robot will be monitored by the controller in real time. When the motion speed or position is out of tolerance, the robot will immediately stop and alarm the corresponding fault.
- For safety, the maximum speed of the robot will not exceed 250mm/s when the manual mode of the teaching device is used.
- The range of motion of each joint of the robot has been set at the factory. If necessary on site, the value of the soft limit of the joint can be modified, but it cannot exceed the limit of the mechanical limit.
- The teaching device provides user-defined use authority function, and the operator can assign the corresponding authority to the field production personnel according to the field situation, in order to prevent the production personnel from performing some misoperations, resulting in personnel or property losses.

1.10 Common situation on site

Examples of situations

The majority of robot accidents are as follows:

- The robot performs automatic operation without confirming whether there is a person within the movement range of the robot.
- In the automatic operation state, a person enters the action range of the robot.
- The robot starts suddenly during the operation.
- They only pay attention to the robot in front of them and pay no attention to other robots.

The above accidents were caused by "neglecting safe operation steps" and "not expecting the robot to make sudden movements". In other words, they are

all accidents caused by man-made unsafe behaviors such as "moming negligence" and "not following the prescribed steps". Therefore, operators must keep safety precautions in mind during their daily operations and must not be careless.

In addition, "emergencies" will also make the operator too late to implement "emergency stop", "escape" and other behaviors to avoid the accident, which is likely to lead to major accidents.

"Emergencies" generally have the following types:

- The low speed actions of the robot suddenly become high speed actions.
- Other operators performed the operation.
- Different procedures were started due to anomalies in peripheral equipment and program errors.
- Abnormal action due to noise, malfunction, defect, etc.
- Misoperation.
- A high speed action is executed when intended to be performed at a low speed.
- The workpiece carried by the robot drops and scatters.
- The workpiece is in the stop state of clamping and interlocking standby, and suddenly loses control.
- An adjacent or behind robot executes the action.

Action countermeasures

These are just a few examples.There are many forms of "contingencies." In most cases, it is not possible to "stop" or "run away" from a robot acting suddenly, so the following best countermeasures should be implemented to avoid such accidents:

- When the robot is not in use, measures such as "pressing the emergency stop button" and "cutting off the power" should be taken to make the robot

unable to move.

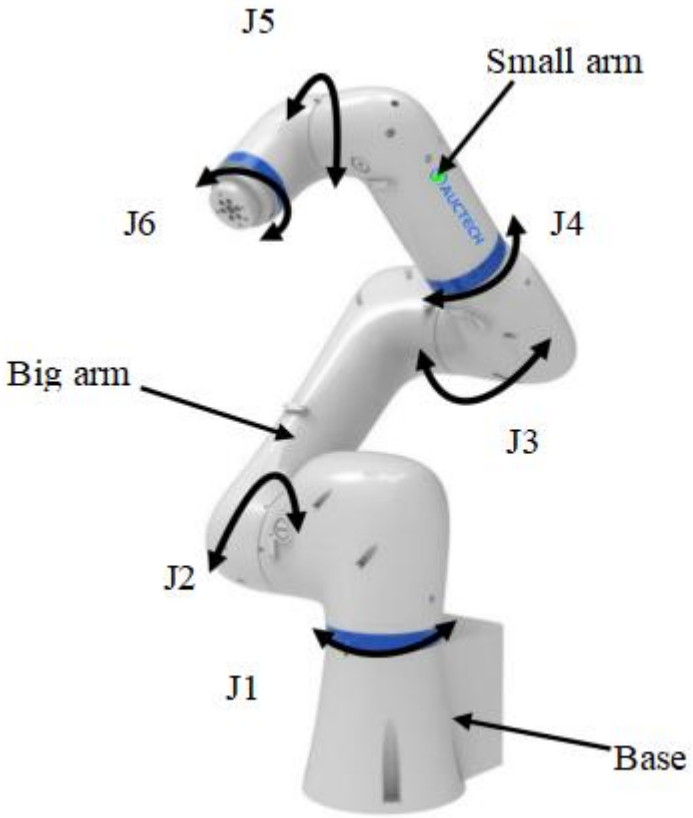
- During robot action, configure safety personnel who can immediately press the emergency stop button to monitor the safety state.
- During the robot operation, the emergency stop button should be pressed immediately.
- During the action of the robot, the action of the robot should be paid attention to at all times, and the operation should not be carried out with the back of the robot. Even if the robot's action response is slow, it will lead to an accident.

The above countermeasures must be fully understood and kept in mind by the operator and adhered to when working in the field.

2 System hardware composition

2.1 Robot body

Introduction The robot body is mainly composed of a base, a large arm, a small arm, and various joints. The robot model body structure and each axis direction are shown in the figure below:



2.2 Electronic control system

Introduction The core components of the robot's electronic control system include: controller, servo driver, IO module, teaching display, power cable and encoding cable.

2.2.1 Control unit

Introduction EPC employs an open, modular architecture based on embedded industrial computers, running real-time Linux systems. It integrates efficient robot motion control algorithms and features advanced fault diagnosis mechanisms. The controller is illustrated below:

2.2.2 Collaborative robotic arm control cabinet

Introduction The appearance of the C5 collaborative electrical control cabinet is shown in the figure below：

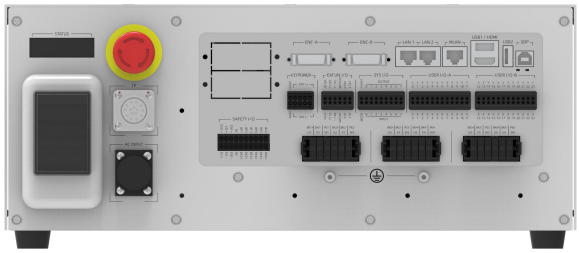
Panel Description

- Emergency Stop Button: Press this button in emergencies to engage the brake and cut the servo signal.
- Power Switch and Indicator Lights: Controls the power supply to the electrical cabinet and displays the circuit status.
- Power Line Connector: Supplies power to the robot's main unit.
- Teaching Pendant Connector: Connects to the teaching pendant for power and network communication.
- Teaching Pendant Switch: Controls the power supply to the teaching pendant.

2.2.3 Drive and control integrated cabinet

Introduction The integrated drive and control unit is a high-performance system combining motion control with servo drive technology. Featuring built-in 4-axis/6-axis motor closed-loop signal sampling inputs and three-phase drive bridge outputs, it provides multiple I/O and communication interfaces for seamless integration with sensors, PLCs, and other devices. Widely used in 1-6-axis

robots, small CNC machines, and specialized automation equipment. The integrated drive controller is shown in the following figure:

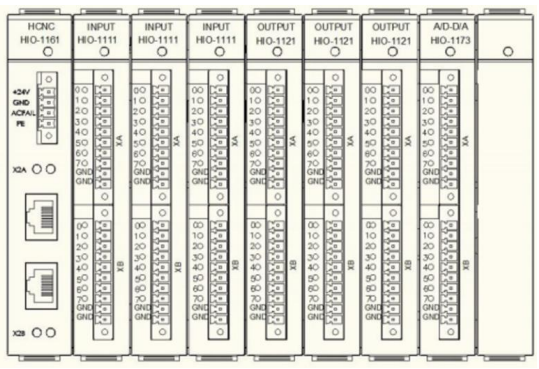


2.2.4 Extend IO unit

Unit Description

TV itself IO unit using EtherCAT bus protocol, has the characteristics of high stability, high reliability. Products through strict anti-corrosion processing, has the input filter and power-fail protection function. IO extension module can be configured for any number of digital quantity input/output boards, supports both analog input and output.

IO equipment unit as shown in the figure below:



2.3 Demonstrator

Introduction The teaching pendant is a handheld device for manual robot movement, program writing, parameter configuration, and device status monitoring. It communicates with the controller via a network port. Robot teaching pendants are commonly abbreviated as 'HSpad'.

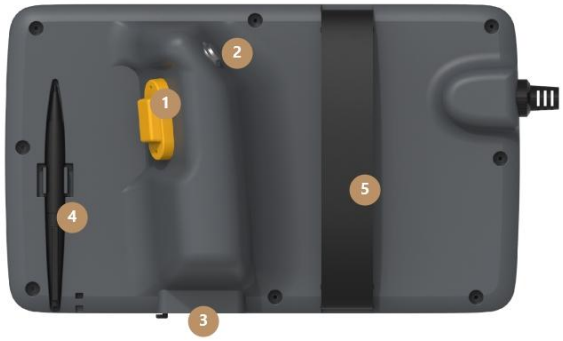
HSpad-05 is shown below:

Front

Label	Introduction
1	Emergency stop button: to quickly stop the robot's movement。 Steps: (1) Press the button vertically to complete the emergency stop operation; (2) Turn the button clockwise to reset the emergency stop button; (3) (3) Confirm the alarm information in the information bar to complete the reset emergency stop operation.
2	Jogging button: Used to jogging the robot.。 The series of buttons has a total of six groups of buttons, each group is divided into positive and negative buttons, corresponding to the positive and negative movement direction of the robot's corresponding axis. Steps: (1) Switch the robot to manual mode; (2) Press the safety switch on the back of the teaching device to enable the robot; (3) Press the axis direction button to complete the robot's jog operation. Configure [Shortcut] to change the button to a backup button. This is a custom button feature. You must first configure the backup button before using it.
3	Speed adjustment button: Adjusts the robot's operating vord(velocity overwrite speed) in automatic and external modes.。 Steps: (1) Select the current robot mode; (2) Click the button to adjust the speed.
4	Home button: Tap to return to the home page, or hold to go to the background interface

5	Back button: Switches between the main menu and the file navigator.
6	Run button: To execute the loaded program.
7	Back button: To go back to the previous line of the program. Steps: (1) Switch the robot to manual mode; (2) Press the safety switch on the back of the teaching pendant to enable the robot; (3) Click the button to complete the program line rollback operation.
8	Stop button: to stop and uninstall the program.
9	Pause button: Pauses the running program.

Back



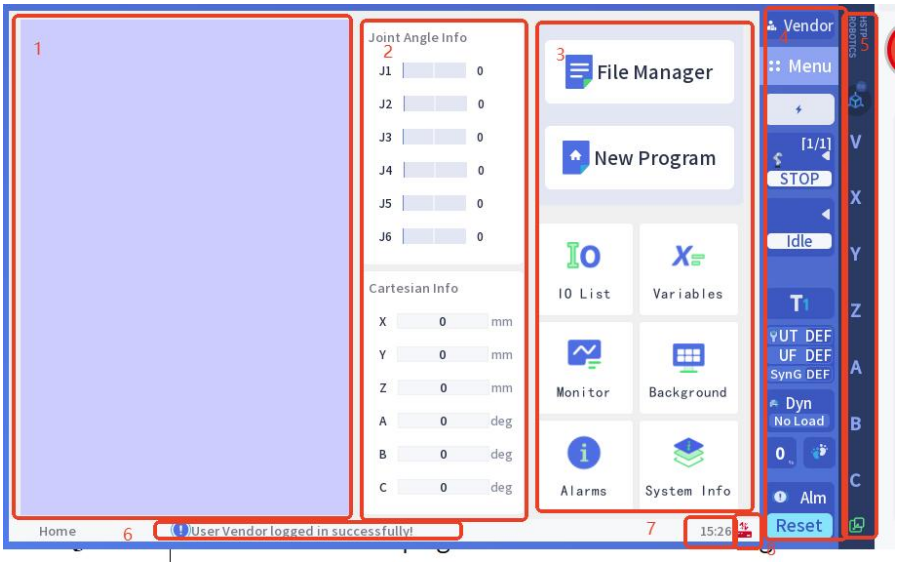
Label	Introduction
1	Three-stage safety switch: Enables the robot in manual mode. The safety switch has three positions: (1) Initial position. (2) Press to the middle position of the first section. (3) Press to the full position of the second section. When the robot is in manual mode, the user must press and hold the safety switch in the middle position to enable the robot to operate. The safety switch is disabled in all other positions except the middle position. The safety switch does not function when the robot is in automatic or external mode.
2	Combo Button: Run the button with a single click based on the combo button configuration.
3	USB/Debug interface: Used for inserting USB drives to copy, retrieve, and perform other file operations. Note: Only FAT32 USB drives are supported.
4	stylus

3 Main interface and operations

3.1 Home page

Main interface

After the trainer is turned on, it defaults to the software home page. The operation interface is shown in the following figure.



Label	Introduction
1	3D visualization area. Displays the robot's current pose, area configuration, and installation posture. Note: Area configuration information is only visible after enabling the relevant features.
2	Current robot pose. Displays the robot's joint angles and TCP Cartesian coordinates.
3	Main function entry. Provides access to common functions or pages for the robot teaching pendant.
4	Status bar. Displays and sets basic statuses of the bot, and provides a user login button.
5	Side bar. Set the robot's current point-and-click coordinate system and feedback teaching device physical button press status.
6	Alarm bar. Displays the robot's alarm information.
7	Clock. Displays the local clock on the teaching device.
8	● Connection status. Displays the connection status between the teaching pendant and the robot controller.

	●  : The red icon indicates no connection to the robot controller. ●  : The yellow status icon indicates a successful network connection, but the robot controller is not ready, and the teaching device cannot perform related operations on the robot. ●  : The green status icon indicates a successful network connection and controller initialization, enabling the teaching pendant to operate the robot normally.

3.2 Status bar

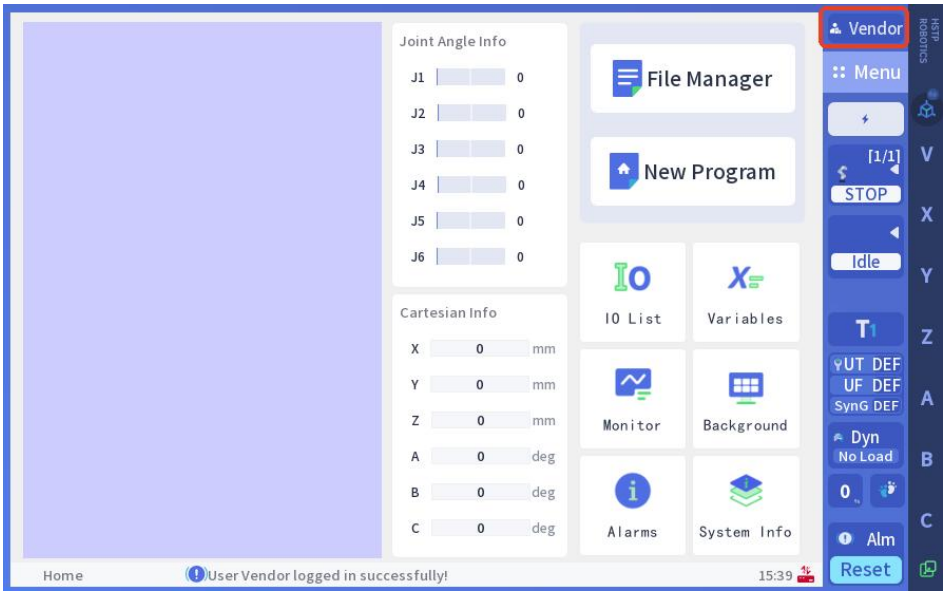
The status bar is a key component of the teaching pendant software, primarily used to display basic statuses and configure related functions of the robot or teaching pendant. As shown in the figure below, the functions or display information from top to bottom are:

Label	Introduction
1	users login
2	Switch to Menu/Home
3	Enable status view
4	Axis Group Information and Control Axis Configuration
5	View and manage program status
6	View and switch motion modes
7	View and switch coordinate systems
8	Check and switch speed levels
9	View and configure running speed
10	View and switch program modes
11	Alarm Reset

- Always fixed to the left side of the screen

3.2.1 User switch

The teaching pendant implements permission division through users, including four types: operator, debugger, administrator, and manufacturer. The operator has the lowest permission level, only able to view most of the robot's status or perform partial function configuration. The manufacturer has the highest permission level, which can access all functions provided by the teaching pendant.As shown in Figure 3-1, the status bar displays the currently logged-in user information. Clicking the control will open a login pop-up window, allowing users to switch between different accounts, each with corresponding operation permissions.



3- 1users login

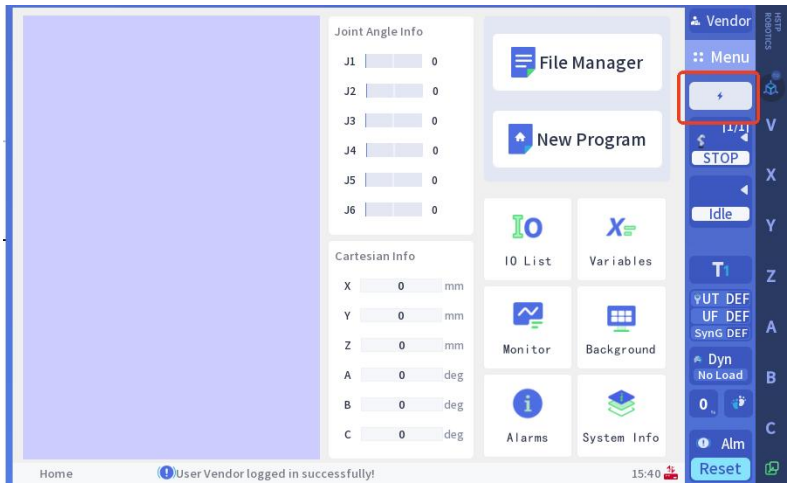
operating steps

Click the control in the red box of Figure 3-1 to open the user login window. Select the desired user in the login window, enter the corresponding password, and click the [Login] button in the user login module window to complete the login operation.

Note：

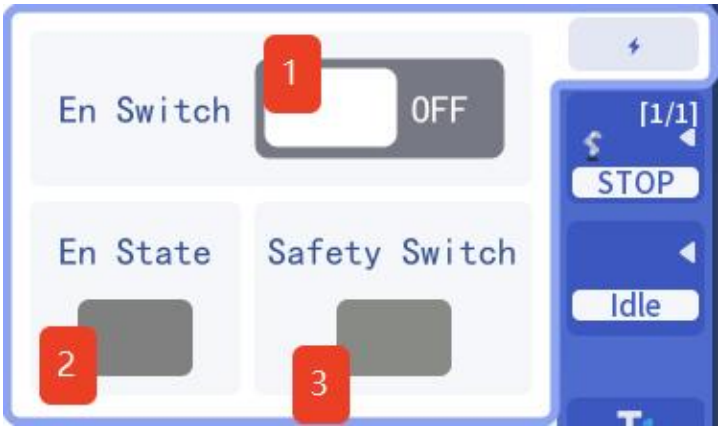
- When the relevant function is not accessible to the current user, the teaching pendant will display a prompt.
- Production user login requires no password. The default password for the debugger user is hstp.
- The default password for the administrator user is HSTP.
- The password for the vendor user is not provided. For assistance, contact the relevant supplier.
- Debuggers and administrators can change their passwords. If you cannot log in with the above password, check whether you have changed the corresponding password. If you forget the password, reset it in the User Permissions section.

3.2.2 Enable status view and toggle



- As shown in the figure above, the status bar displays the enabled status of the current robot, where:
-  ：Indicates that it is currently not enabled.
-  ：Indicates that the above is enabled.

Click the Enable Status Display control in the status bar to open the Enable Status Settings pop-up window, as shown in the figure below.

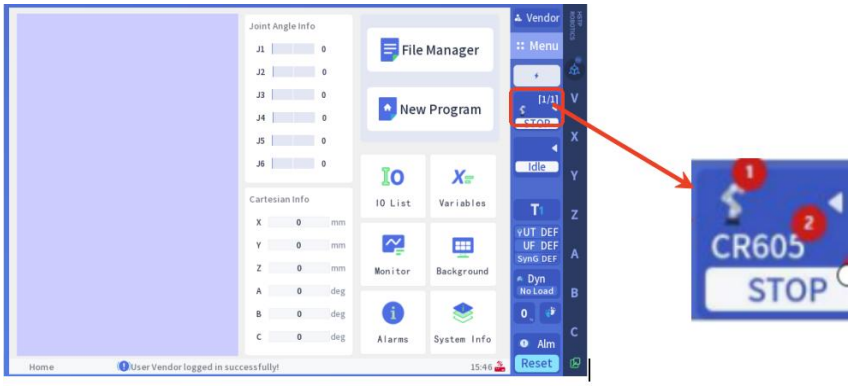


Lable	Introduction
1	Enable toggle. Click to switch the state.
2	The robot is currently enabled. The color block is green when enabled.
3	The three-stage safety switch on the back of the trainer shows a green color block when pressed.

Note：

- In manual mode, you cannot switch the robot's enable status through this enable switch. The corresponding button will be unclickable. Use the three-stage safety switch on the back of the teaching pendant to enable the robot.
- In external mode, you cannot switch the robot's enable state using this enable switch. The corresponding button will be unresponsive. Use the configured IO to control enable on/off.。

This is group status and motion axis switching



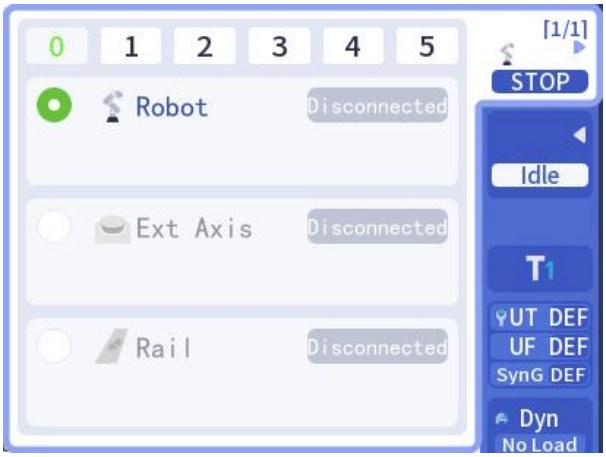
As shown in the figure above, users can view the current model name or robot alias and the robot's current motion status in the status.

Label	Introduction
1	Current control device type
2	Model name or current robot alias
3	Robot movement status

Switch system motion axis

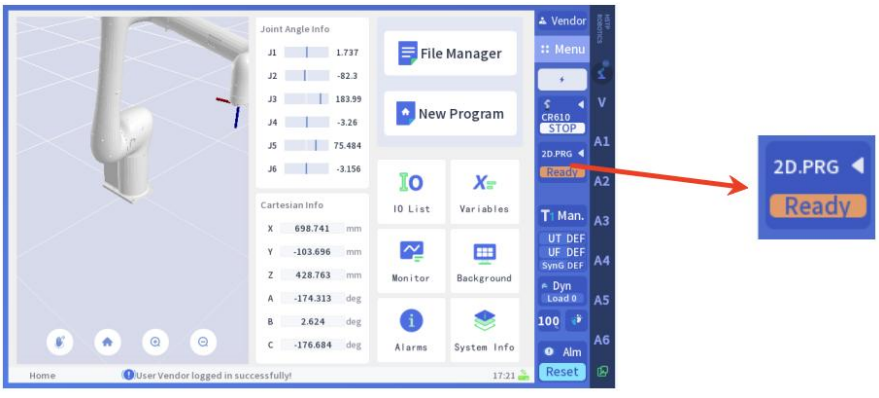
The system may be configured with external shafts or ground rails depending on the situation. To control the external shaft or ground cabinet, you need to switch the current motion shaft of the system to the corresponding device to control it.

Clicking the "Model Name Display" control in the status bar will open the [Control Device Switch] pop-up window as shown in the figure below. Click the device you want to control to switch.



3.2.3 View and control program status

Introduction



As shown in the figure above, users can view the status of the program currently running by the robot in the status bar, where:

Label	Introduction
1	Displays the name of the last opened program in the program editor interface among the currently loaded programs. Offline programs are not displayed.
2	A status indicator displays the status of the relevant program, which has seven states: idle, loaded, ready, running, error, paused, and waiting.

Depending on the combination of different loaders, the display effect of the above controls will be different. See the table below for details.

program status	Control display effect	Introduction

No programs loaded		[Idle] is the status of the main program.
Load the main program		[Ready] is the status of the main program.
Load the main program and a background program		The first [Ready] indicates the main program's status; the second [Ready] indicates the background program's status.
Load the main program and two background programs		The first [Ready] indicates the main program's status; the second [Ready] indicates the status of background program 1; the third [Ready] indicates the status of background program 2.
A background program		[Idle] indicates the main program is not loaded; [Ready] indicates the background program is loaded.
Two background programs		[Idle] indicates no main program is loaded; the first [Ready] indicates the status of background program 1; the second [Ready] indicates the status of background program 2.

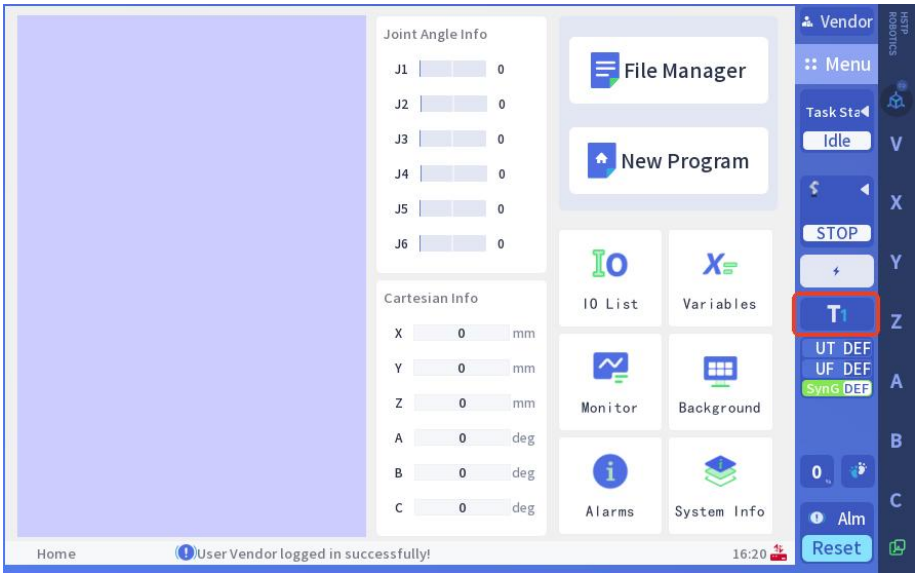
To summarize the above control status: The first status control is always displayed and always displays the status of the main program. The second and third status indicators are dynamically displayed according to the number of background programs actually loaded.

Click the display control above to open the [Program List] pop-up window, as shown below:

Label	Introduction
1	The program name is displayed. Click the corresponding program name area to open the program in the program editor.
2	Program control soft button. Click the [Pause], [Run], or [Uninstall] button of the corresponding program to perform the corresponding operation.
3	For the currently opened program, click the name row area or the [Restore Open] button to open the corresponding program in the program editor.

3.2.4 View and switch modes

Introduction

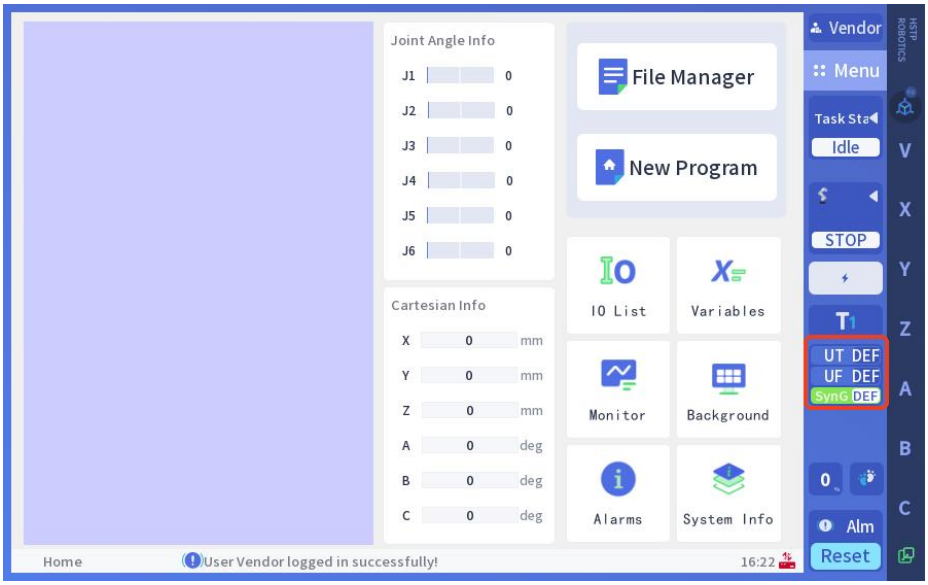


As shown in the figure above, users can view the current operation mode in the status bar. Clicking this control will pop up a mode switching floating window, as shown in the figure below:

Select and click the corresponding mode in the floating window to switch modes.

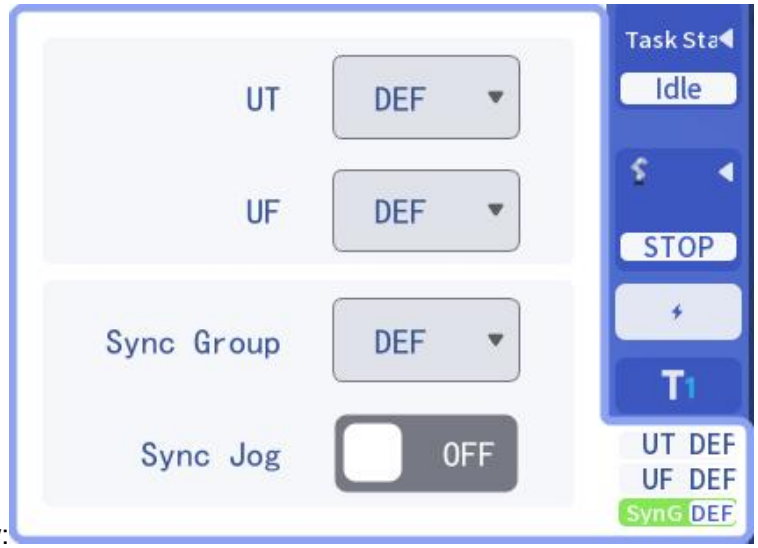
3.2.5 Select coordinate system and collaboration group

Introduction



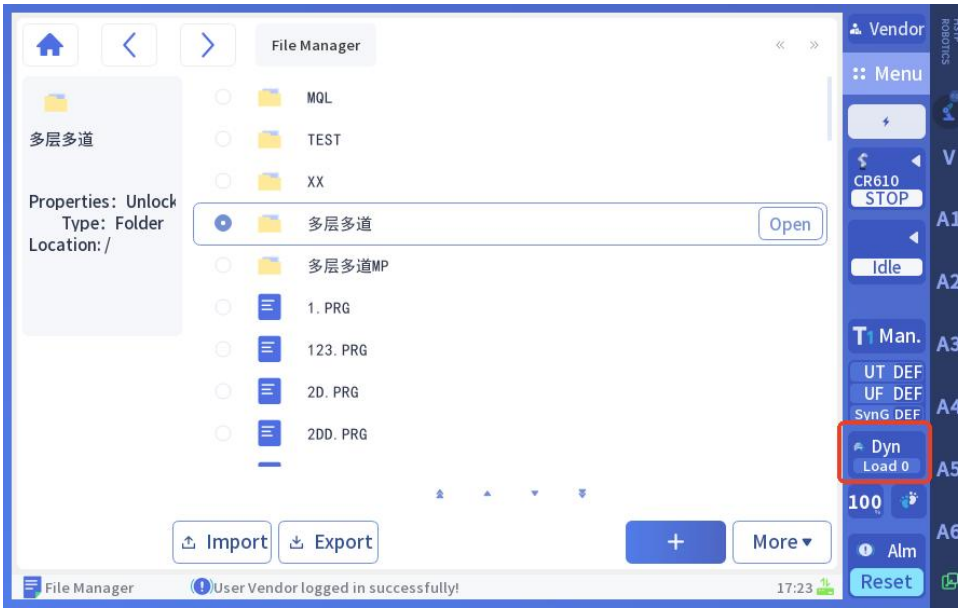
As shown in the figure above, users can view the tool number, workpiece number, and collaborative group number selected by the robot in the status bar.

Clicking the display control will open a pop-up window where you can switch between tools, workpieces, IPO mode, and collaborative groups, as shown below:



3.2.6 Check speed level

Introduction



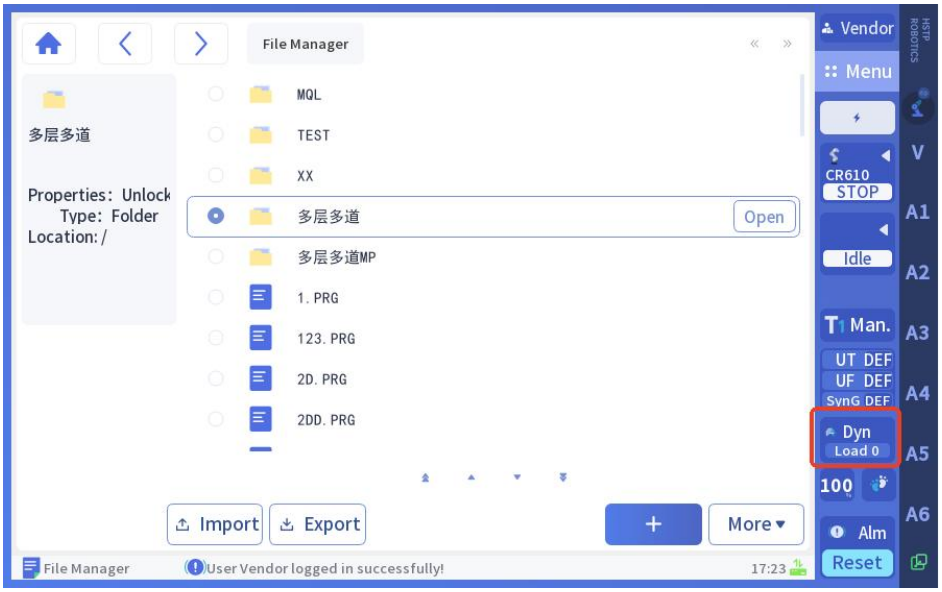
As shown in the figure above, users can view the robot's current speed level in the status bar.

note：

- The speed level can only be viewed in the status bar and cannot be modified. For changes, refer to the shaft group configuration section.
- If the current model supports dynamics and has undergone body identification, this control will be hidden, and the corresponding area will display [Dynamic Load].

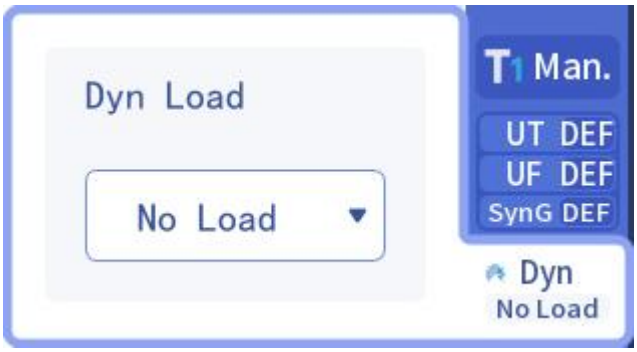
3.2.7 View and select dynamic loads

Introduction



As shown in the figure above, if the current model supports dynamics and ontology identification has been performed, the control that displays speed grade information will display the current dynamic load information, and the current selected load can be viewed.

Click the control to open the [Load Selection] pop-up window. Click the [Dynamic Load Selection] dropdown to switch loads.



3.2.8 View and set playback speed

Introduction

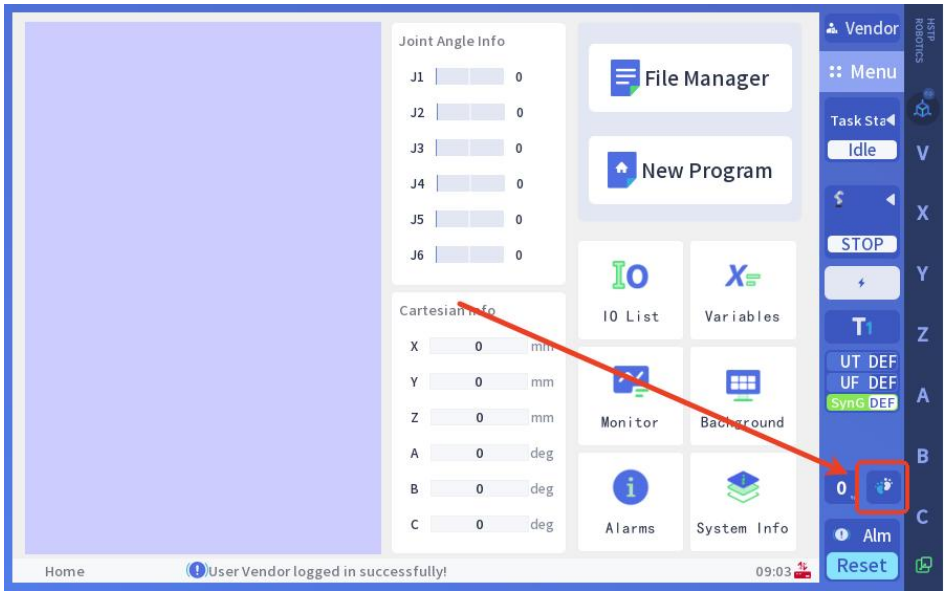
As shown in the figure above, users can view the current zoom level in the status bar. Clicking the control will open a "Zoom Adjustment" pop-up window, where you can adjust the zoom level as shown below:

3.2.9 View and select program modes

Introduction

- The robot has two operating modes, which are:
- Continuous Mode: The program runs non-stop until completion or manual intervention (pause/stop). This mode is ideal for production environments.

- Step-by-Step Mode: The program pauses after each line execution, perfect for debugging and testing.



As shown in the figure above, users can view the current program running mode in the status, where：

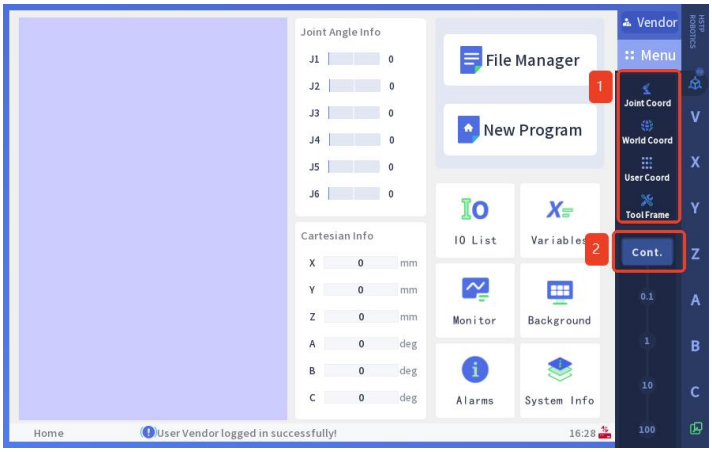
-  ：Indicates that the current mode is single-step execution.
-  ：Indicates continuous operation mode。

Click the control to open the "Run Mode Settings" pop-up window, where you can switch the program's run mode, as shown in the figure below.

3.3 Sidebar

Introduction

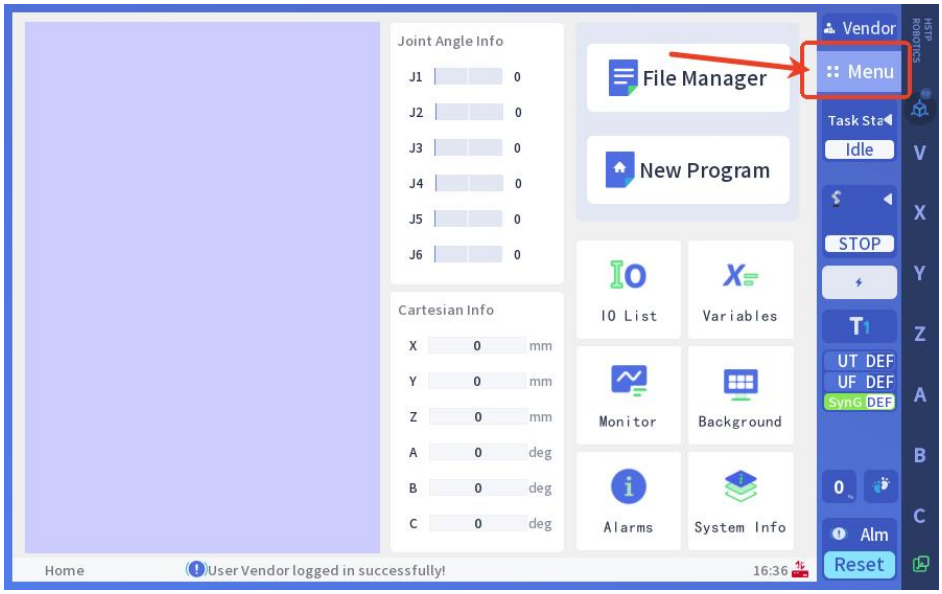
As shown in the figure above, the sidebar is located on the far right of the software. It provides visual feedback when pressing the relevant physical keys and selects the point-to-point coordinate system.：

Label	Introduction
1	LOGO。 Click to open the background view.
2	Select the point-and-drag coordinate system and method.  Label 1: Click the corresponding coordinate system to select a point motion coordinate system. Label 2: A slider allows you to choose between continuous or instantaneous motion.
3	Speed and button press indicator. Indicates whether the corresponding physical button is pressed, with the

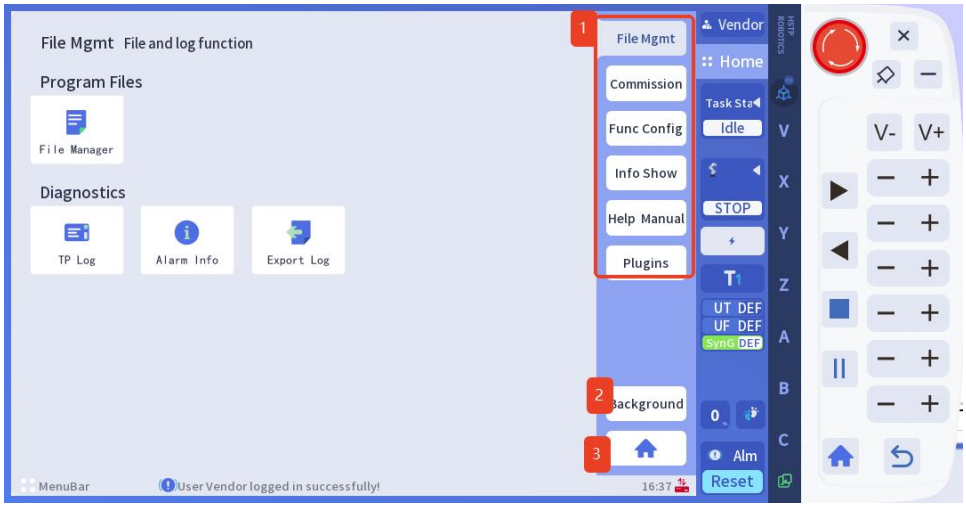
	corresponding press indicator highlighted when pressed. V：Feedback when pressing the physical keys [V-] and [V+]; A1~A6：The feedback for the six groups of [-] and [+] physical point-and-click buttons may vary depending on the selected point-and-click coordinate axis and system axis group configuration, but the corresponding physical buttons remain unchanged.
4	Shortcut key press indicator. Indicates whether the physical combination key is pressed. When pressed, the indicator turns on. You can also tap. If the combination key function is enabled, tapping is equivalent to pressing the physical combination key.

3.4 Menu

Introduction



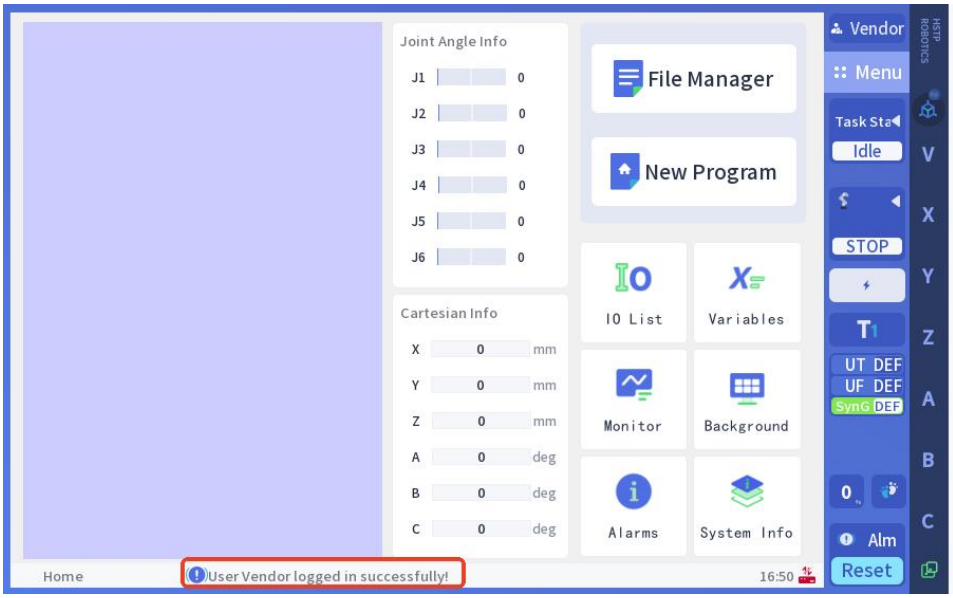
As shown in the image above, click the [Menu] or [Home] button on the status bar to switch to the Home or Menu page. Users can access most functions via the Menu page, as illustrated below:



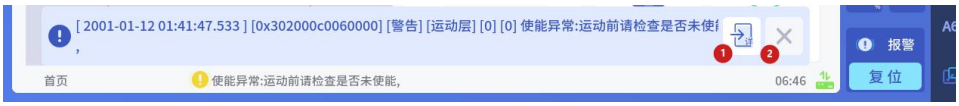
Label	Introduction
1	Submenu Click the corresponding button to switch to different submenus.
2	Backend Button Click to open the backend management interface.
3	Home Button Click this button or the [Home] button on the status bar to return to the software home page.

3.5 View and reset information

Introduction When a robot or teaching pendant generates relevant prompts, warnings, or error messages during operation, they will be displayed in the bottom information bar of the teaching pendant, as shown below.:

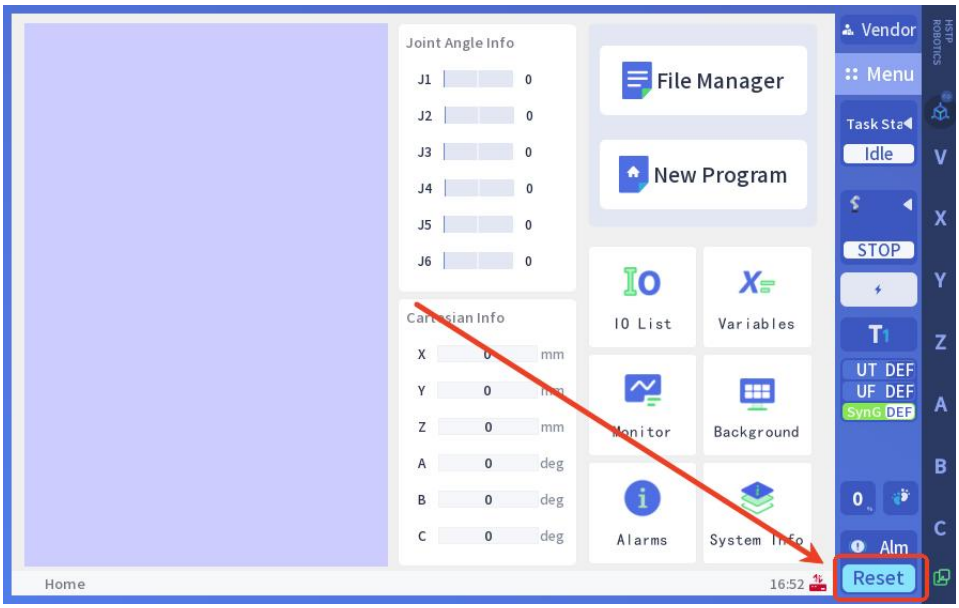


Click the corresponding information to display a pop-up alert window with complete details, as shown below:



Label	Introduction
1	Clicking this button will redirect you to the [Alarm Service Information] page for more details. For instructions on the alarm service information interface, refer to the Alarm Service Information section.
2	Clicking this button will close the floating window.

Click the [Reset] button at the bottom of the status bar to clear information and remove related alarms, as shown below.

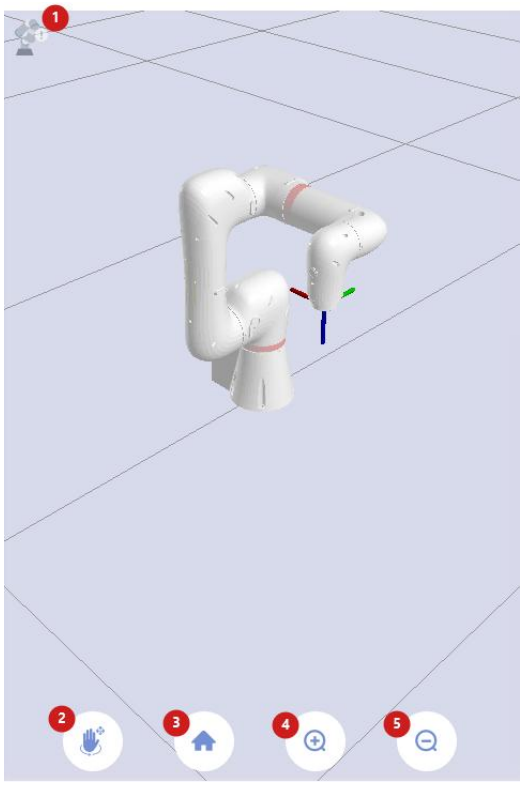


note：

- Press and hold the area outside the button to drag the floating window.
- Clicking on the hint level will not display the floating window.

3.6 3D view

Introduction



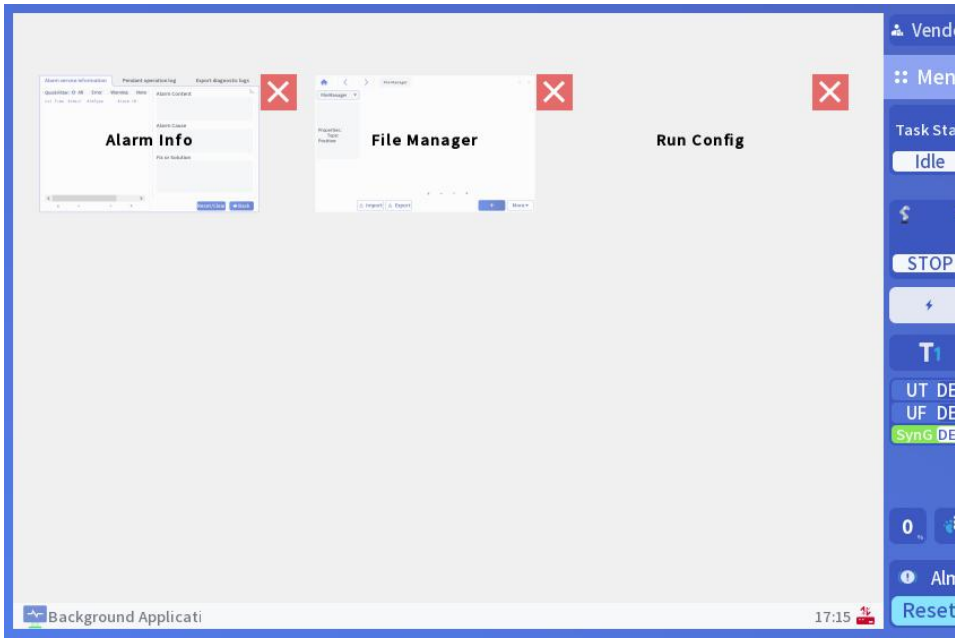
The teaching device provides a 3D view control on the home page or some function pages to visually display the robot's current pose, as shown in the figure above.

Label	Introduction
1	Collision detection status。 After the robot completes the body friction identification, the control will be displayed, showing the current collision detection status. The status icon and its meaning are as follows: ●  ：Collision detection is on； ●  ：Collision detection is not enabled； ●  ：The robotic arm collided。
2	Switch view mode. ● Click this button to switch the view mode. The current icon and view mode meanings are as follows: ●  ：Rotation mode is active. Tap to switch to Pan mode. ●  ：You are in Pan mode. Tap to switch to Rotate mode.
3	Clicking this button will switch the view to default.
4	Click this button to zoom in or out.
5	Click this button to zoom in or out.

3.7 Admin console

Introduction

Long-press the physical [] button to open the backend management page. Here, users can view recently accessed pages and quickly switch to them by clicking the corresponding tab, as shown in the figure below.



Click the [ ] button in the upper right corner of the corresponding page tab to close it.

Note：

- You can keep up to 9 active background tabs.
- If the interface name in a tab shows "Lock", it cannot be closed immediately. Complete the required operations or save data before closing.

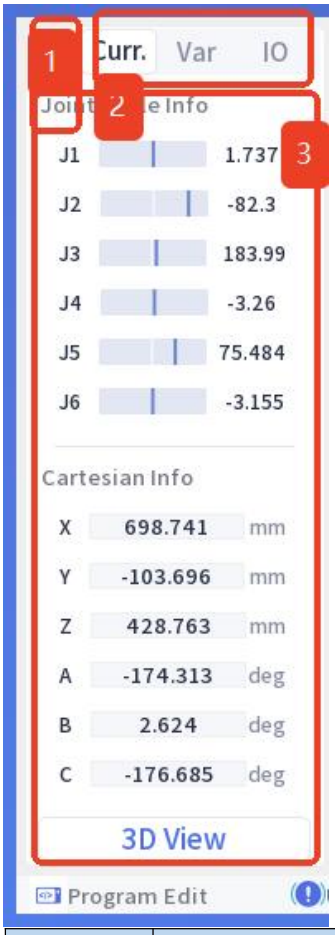
warning：

- Closing the corresponding page may cause unsaved operations on the corresponding feature page to be lost.

3.8 Widget

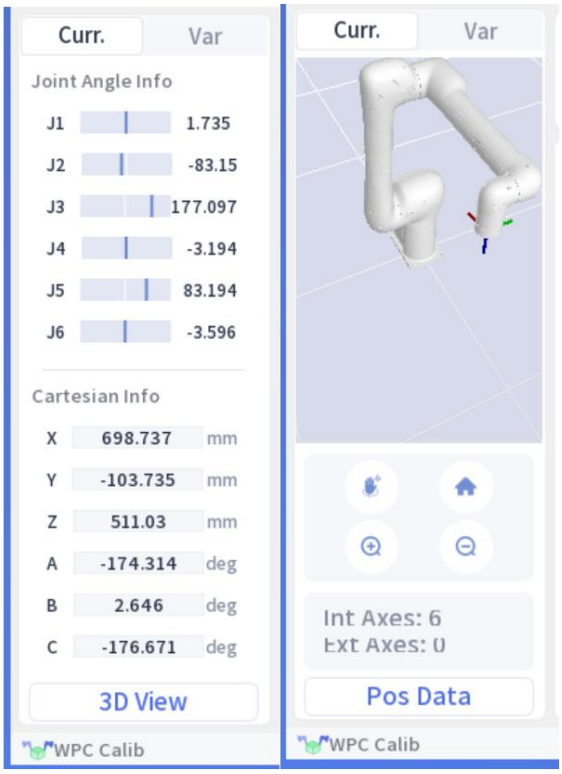
Introduction

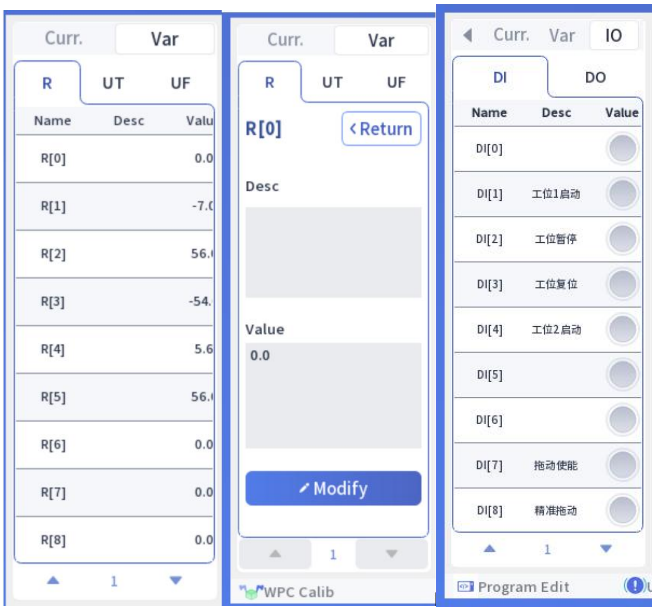
In the teaching device, there is a control on the left side of many function pages for users to view the current position and posture of the robot, related register information and IO status. For convenience of explanation and description, we use "subcomponent" to refer to this control, as shown in the figure below:：



Label	Introduction
1	Tap to minimize the widget
2	Tab Switch.The widget consists of three tabs: [Current], [Variables], and [IO].
3	Tab content display area.。Display different content based on the current selected tab.

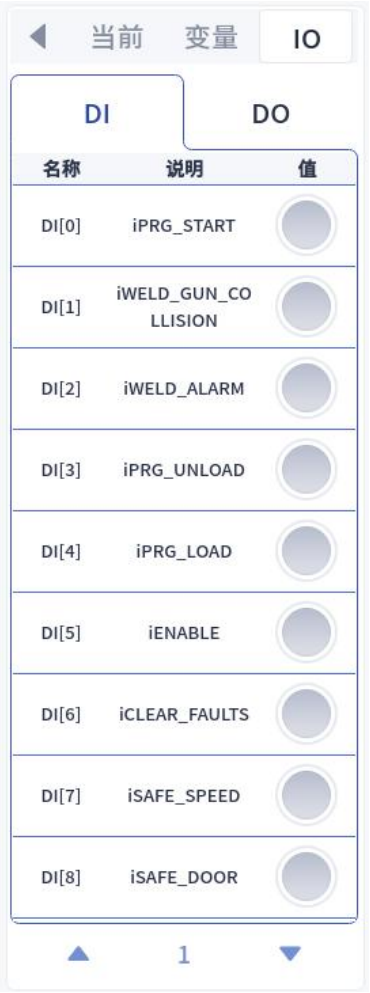
Displays the robot's current pose to the user. Click the [3D View] button to switch to 3D view mode. In 3D view mode, click the [Position Data] button to switch back to normal mode.





As shown in the diagram, the Variable tab displays variable information through multiple tabbed lists. Clicking a variable reveals its register details. On the variable details page, clicking the [Modify] button opens a register modification dialog, while the [Return] button takes you back to the list.

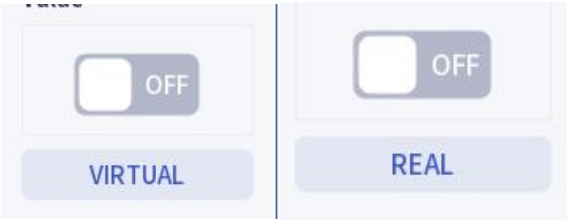
IO TAB



As shown in the figure above, the IO tab displays the IO list to users, consisting of two tab groups: DI and DO.

Tab Title	Introduction
name	The sequence number corresponding to DI or DO.
statement	Instructions or system signal usage notes provided by users corresponding to DI or DO.
value	● ：FALSE/OFF; ● :TRUE/ON。

Click the corresponding IO item to switch to the editing interface for modification, as shown in the figure below.

Label	Introduction
1	IIO value switch indicator. Displays different controls based on the virtual status of the corresponding IO: When the IO is in virtual status, it acts as a value switch, allowing you to toggle the IO value by clicking. 
2	Virtual/real mode toggle button.
3	Click to modify the IO comment.
4	return push-button.

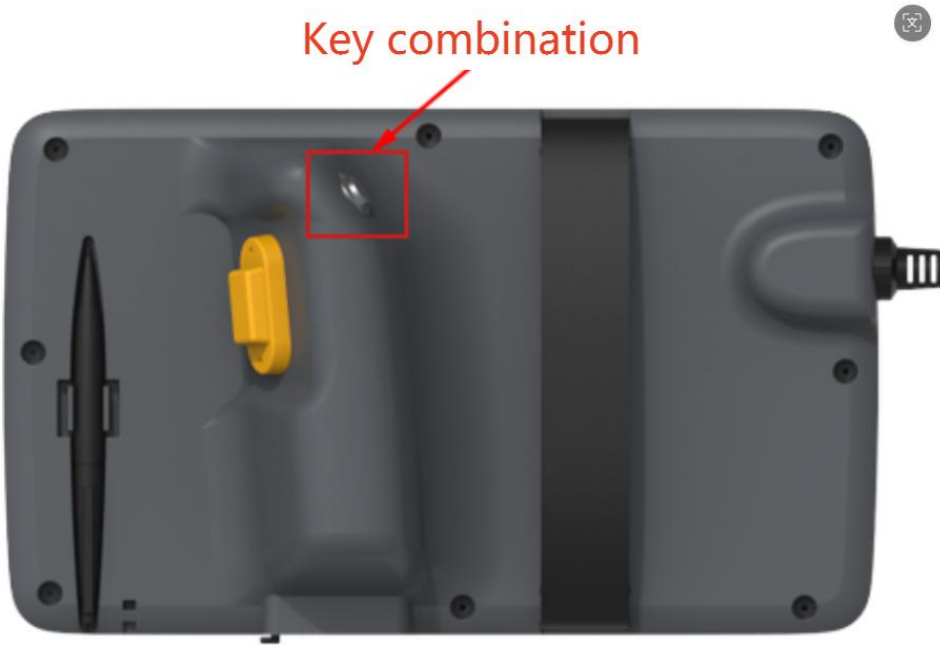
note：

- The widget content varies depending on the actual needs of the function page.

3.9 Combination key,

Introduction Pressing the combination key behind the teaching device while holding other physical keys creates a combination. The supported physical keys and their functions are as follows:

Physical keys	Click method	function
Back button	Press and hold for 1 second	Enter screen calibration mode.
dot-and-run button	Press and hold	Customize



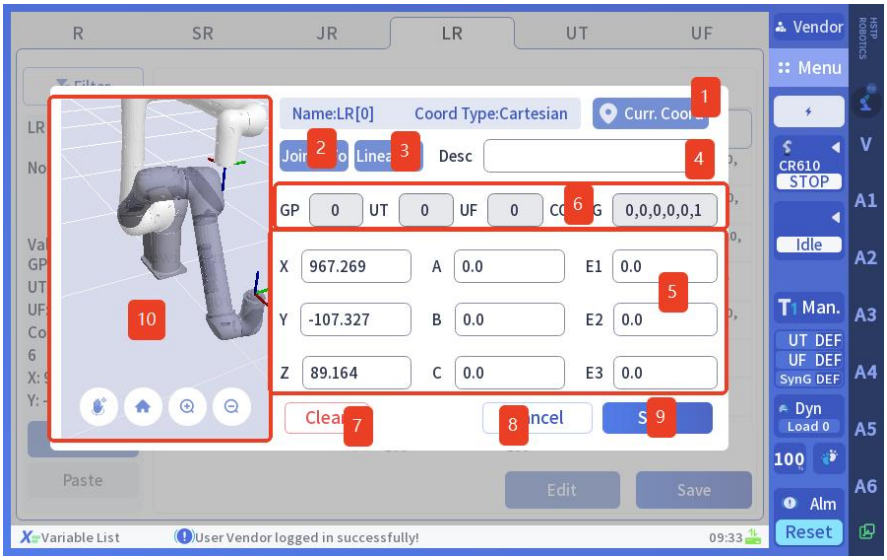
note：

- For configuration of the combination key function for the dot-and-run button, refer to the Combination Key Configuration section.。

3.10 Common pop-up window

3.10.1 Change coordinates pop-up window

Introduction The coordinate modification pop-up is a dedicated window for viewing, modifying, and recording coordinate points. The coordinate modification pop-up is shown in the figure below.



Label	Introduction
1	Get the current coordinates. This button retrieves coordinate data for the current coordinate type. The retrieved data will be displayed in the input box below in the specified format.
2	Joint movement button. Use to move to a recorded point in joint movement mode.
3	Linear motion button. Moves the recorded point to the specified position in linear motion.
4	Enter the description box. This box displays and modifies the description of the point. It is commonly used in modification pop-ups for variable list register variables such as JR, LR, UT, and UF.
5	Point Data Input Box Group. This group displays and modifies coordinate data corresponding to point positions. When the current coordinate system is Cartesian, the input box titles in the point data input box group will be labeled as X, Y, Z, A, B, and C respectively. For joint coordinates, the input box titles will display according to the axis group number and "J+axis number" format, such as J1, J2, J3, J4, J5, J6 for a six-axis robot. For robotic arms with external axes: (1) The LR register coordinate modification pop-up window will display all three external axes regardless of their actual quantity; (2) JR will show the actual configured external axes, with additional point data input boxes displaying the external axis number in the format "E+external axis number".

6	A configuration input box group for displaying Cartesian point coordinates. This group appears only when the coordinate type is Cartesian. It contains three sets of information: UT (tool coordinate system), UF (workpiece coordinate system), and config (pose).
7	Clear button. Clears point data and configuration information. Clicking resets all input fields in the point data input group and the config input field to 0, while resetting the UT and UF input fields to-1.
8	Cancel button. Closes the pop-up window without making any other changes.
9	Save button. Saves changes to point coordinate data. Click to save the modified data and close the pop-up window.
10	3D View Controls. These controls provide users with an intuitive visualization of the robotic arm's current pose. The pop-up interface typically features two models: an opaque solid model representing the arm's current position, and a translucent gray model displaying target point data. If the robotic arm cannot reach the specified target position, a pop-up bubble will display the message "Target inverse kinematics failed. Close 3D display of target position" while hiding the translucent gray model. For more details, please refer to the 3D View section.

Note：

- Some controls in the coordinate modification pop-up window will be hidden based on current functionality requirements.。

4 Functional description

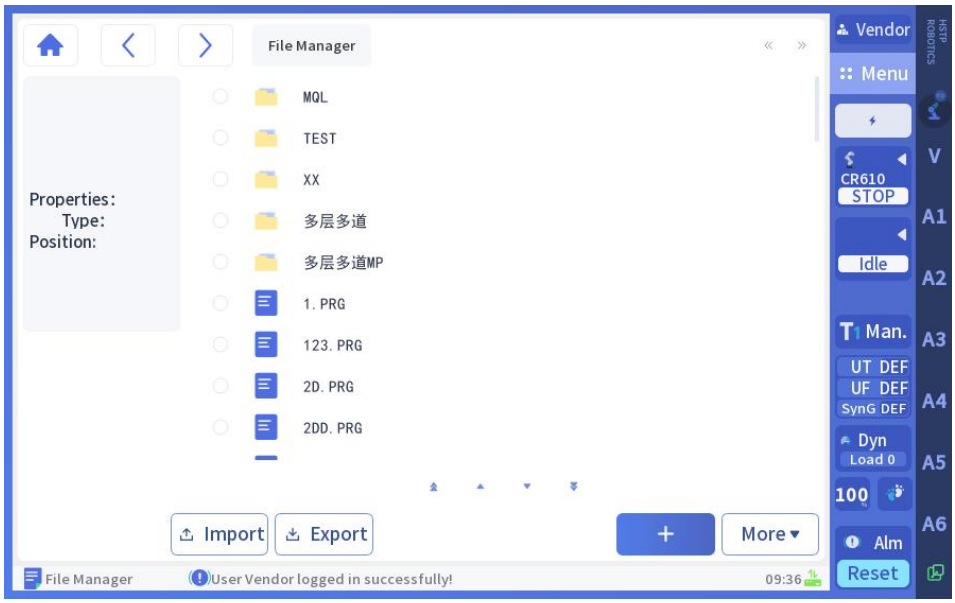
4.1 Program file

4.1.1 Program files management

Introduction

Program file management is typically handled through a program file manager. In this tool, users can view the directory structure of program files and perform various operations such as creating, deleting, opening, loading, copying, pasting, importing, and exporting program files.

The Program Files Manager page is shown in the figure.



Operation path

➢ Access it through the commonly used functions on the home page.

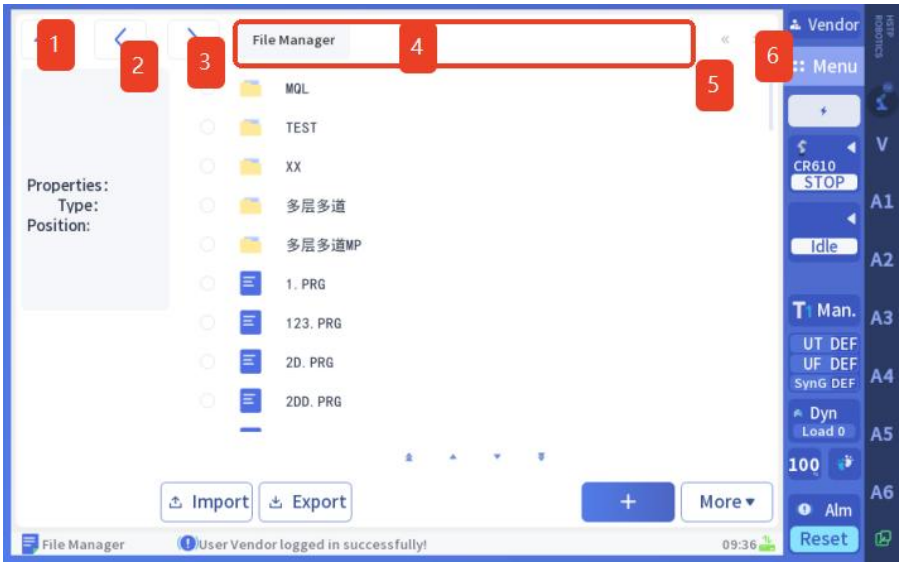
Click the [File Manager] button to enter.

➢ Access the menu interface: [Menu] → [File Information Management] → Program Files: [File Manager]。

4.1.1.1 Browse and navigate

Introduction

The File Explorer's navigation bar is located at the top, featuring functions like Back, Forward, and Jump, as shown in the figure.：



Lable	Introduction
1	Home button . Click this button to return to the trainer home page.
2	Back button. Returns to the user's history directory. If no history directory exists or you are at the top, clicking will have no effect. This feature is similar to the directory back button in Windows File Explorer, but it returns to the history directory instead of the parent directory. The history directory can record up to 10 entries.
3	Forward button. Navigates to the user's previous browsing history. If no history exists or the user is at the end of the history, clicking will have no effect. This feature works similarly to the Windows File Explorer's forward button, but it accesses the history directory instead of navigating to the next folder. The history directory can store up to 10 entries.
4	The directory path navigation bar displays the current directory path and allows quick access to its subdirectories. As shown in the diagram, "File Manager" represents the root directory, "A360" is a subdirectory under the root, and "TETS" is a subdirectory within "A360". These components collectively form the current directory path. All paths in the navigation bar are clickable, redirecting users to their corresponding directories. When paths become too long, the leftmost directory is hidden, with the nearest one displayed at the far right.

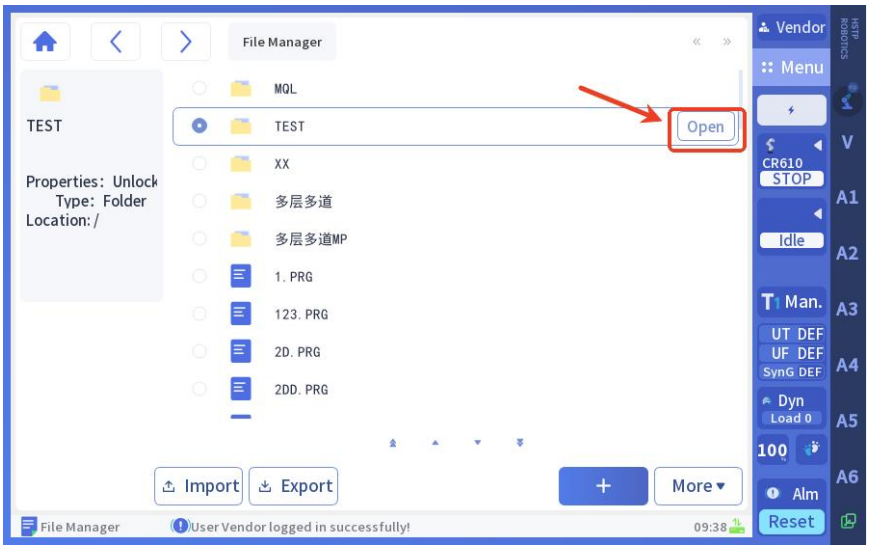
5	Left navigation button. Moves the path left in the directory path navigation bar when the path is too long, helping users view the full path. Clicking it again will not respond when it is at the far left.
6	Right navigation button. Moves the path to the right in the directory path navigation bar when the path is too long, allowing users to view the full path. Clicking it again will not respond when it is at the far right.

open folder

There are two ways to open a folder：

（1）Select the folder you want to access, then click the [Open] button, as shown in the figure below.

（2）Double-click the folder you want to access.



4.1.1.2 Create a new program file

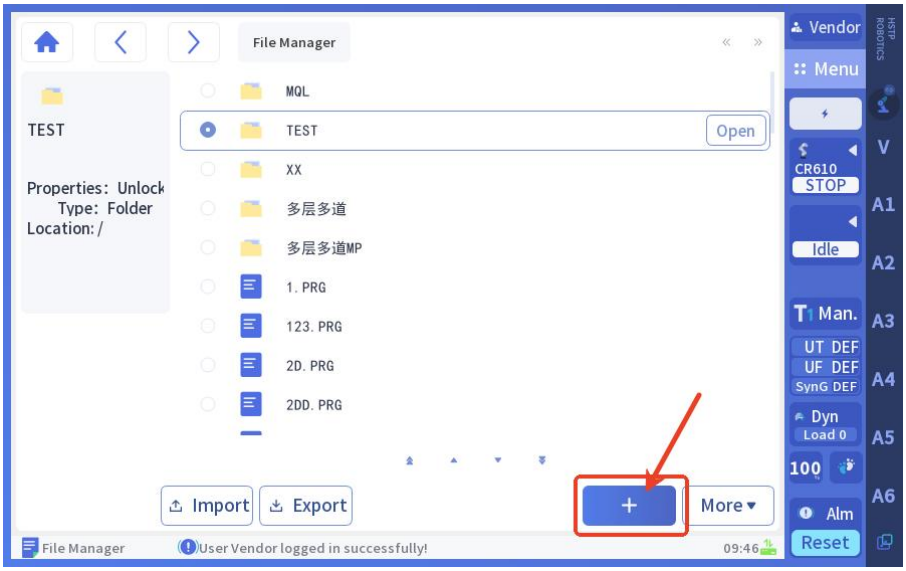
Introduction

- Create a new program file or folder. Robot program files are categorized into two types:
- Main Program Type: The group mask is 0, indicating this program file is a main program that can edit motion-related instructions to control robot movements.

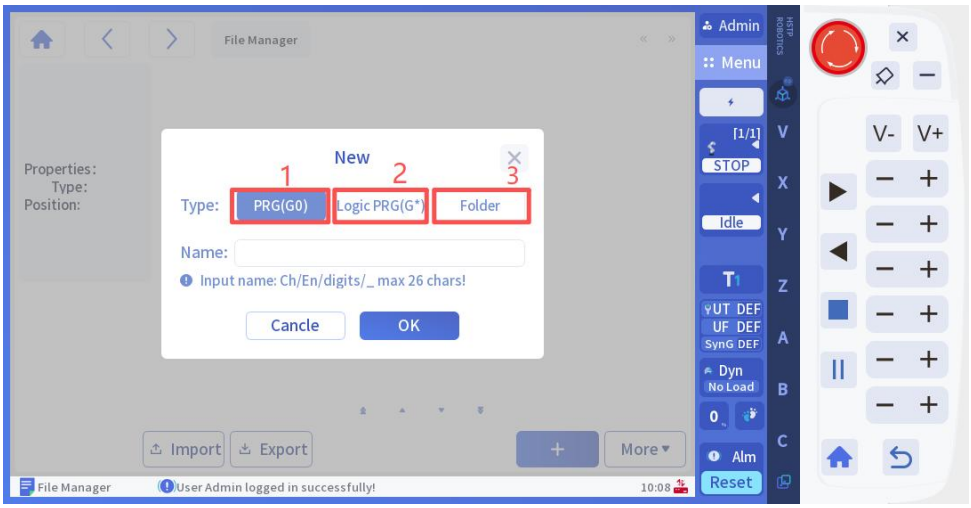
- Background Program Type: The group mask is *, indicating this program file is a background program that cannot edit motion-related instructions.

operating steps

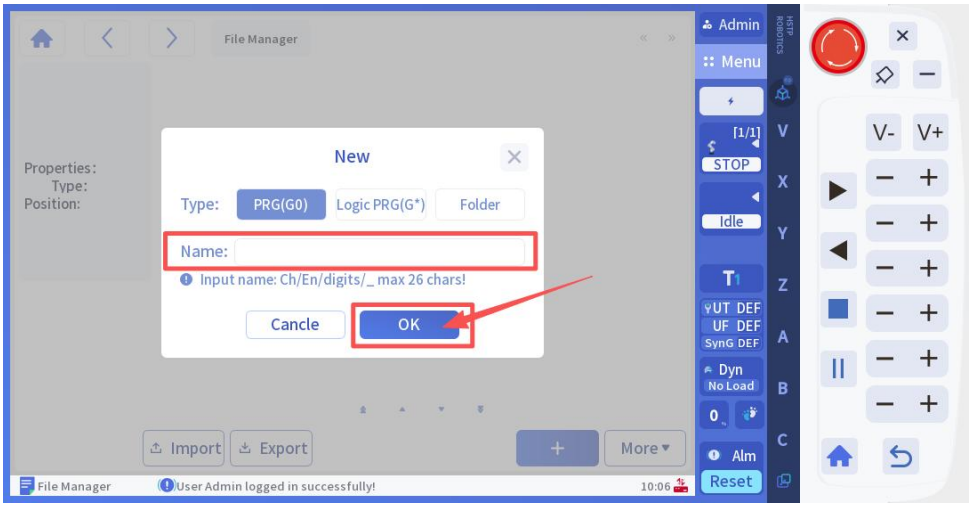
1、Click the [ ] button below.



Select the file type in the pop-up window based on your requirements。



Enter the file or folder name in the [Name] field according to the prompt in the pop-up window.



Click the [OK] button to complete the program or folder creation.

note：

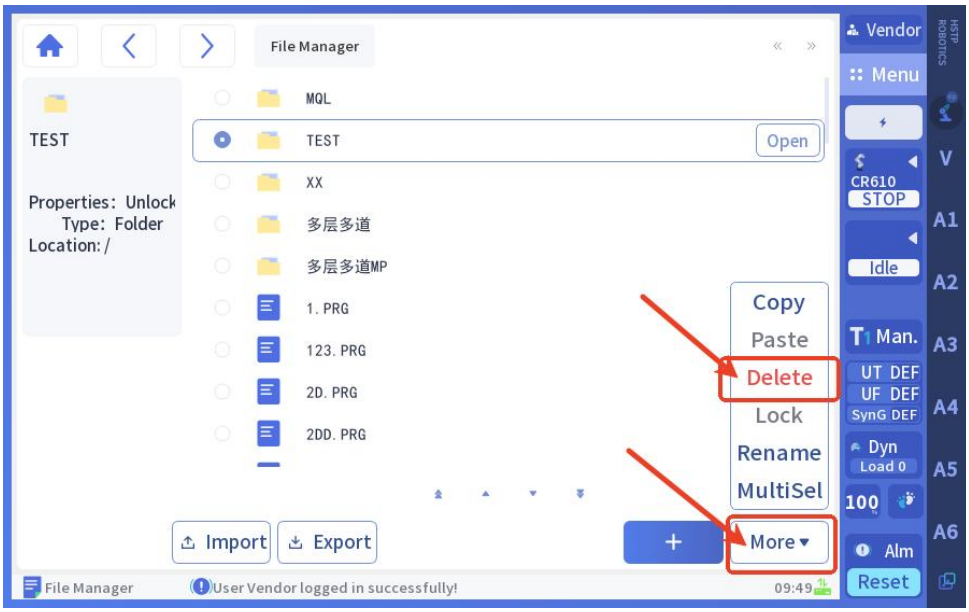
- The file name cannot be the same as any existing files of the same type in the current directory.
- Both program files and folders are stored in the robot controller.

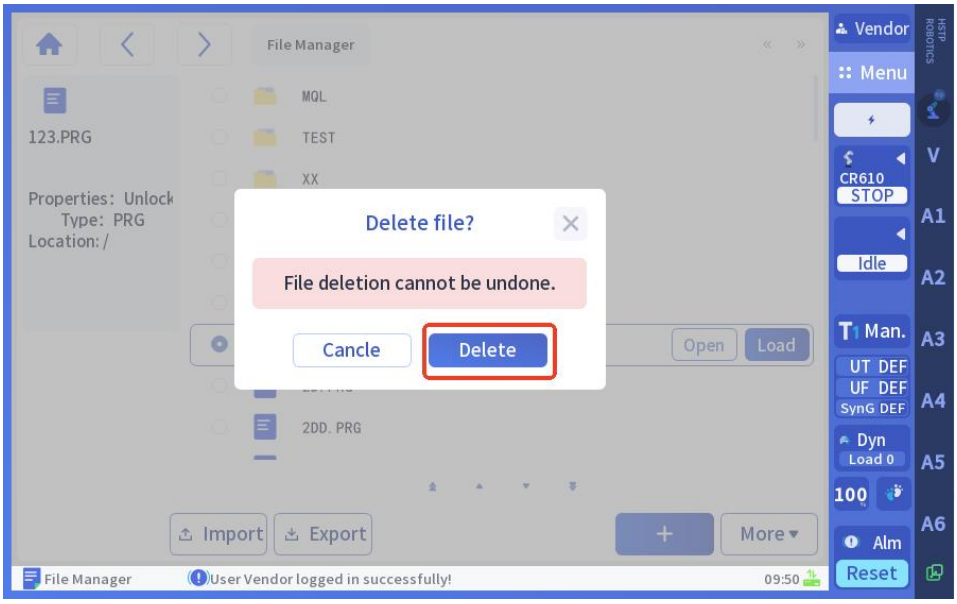
4.1.1.3 Delete the program or folder

Introduction Delete the program file or folder in the controller。

operating steps

1、Select the program file or folder to delete.
2、Click the [More] button to open the additional operation menu.
3、Select the [Delete] option from the additional operation menu.
4、Click the Delete button in the confirmation pop-up





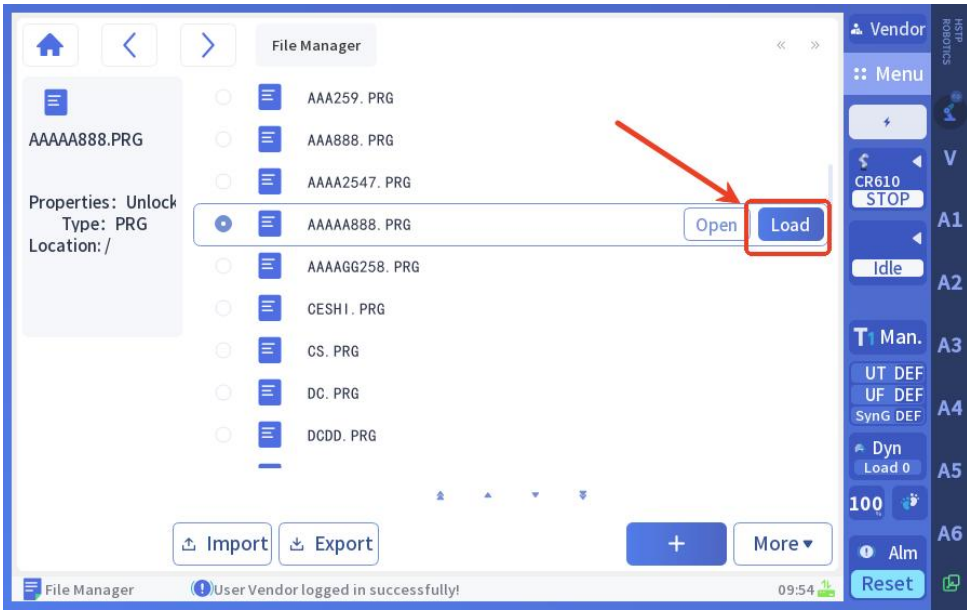
note：

- You cannot delete opened or loaded files. The [Delete] menu item will not be clickable.

4.1.1.4 Loading program

Introduction The loading function loads program files.

operating steps



1、Select the program files to load in File Explorer.

2、Click the [Load] button to complete the loading.

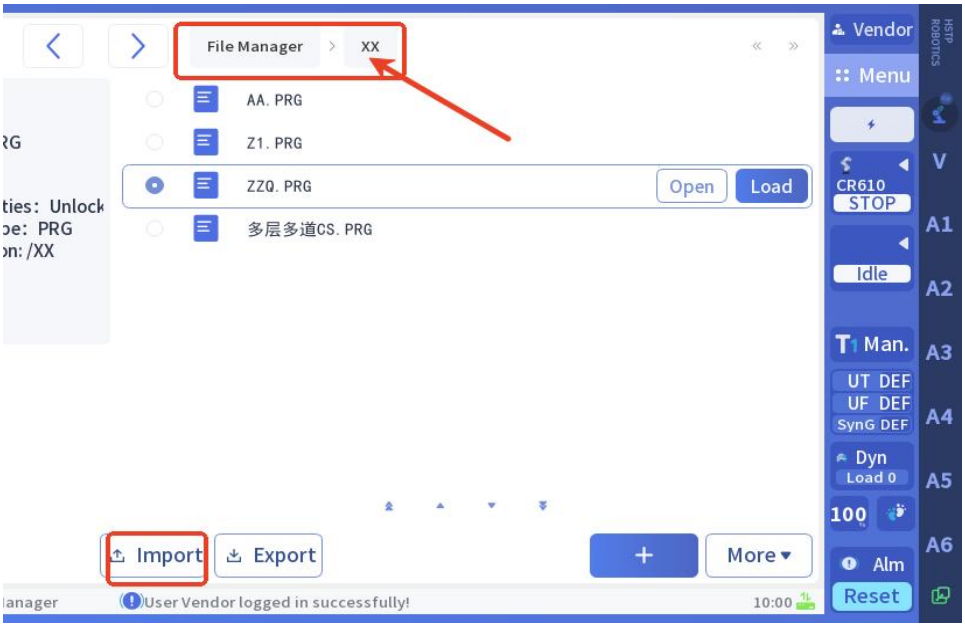
note：

- You can load one main program and two background programs. If you have already loaded one main program and two background programs, the load button will turn gray and become unclickable until you uninstall the loaded programs.
- Opened programs cannot be loaded again. If you click the load button for an open program, a pop-up will appear with the message: "The program is already open. Close it before loading again!"。

4.1.1.5 Import and export

Import Instructions The import function sends files to the controller system or overwrites the corresponding files in the controller system to import or restore them.

operating steps

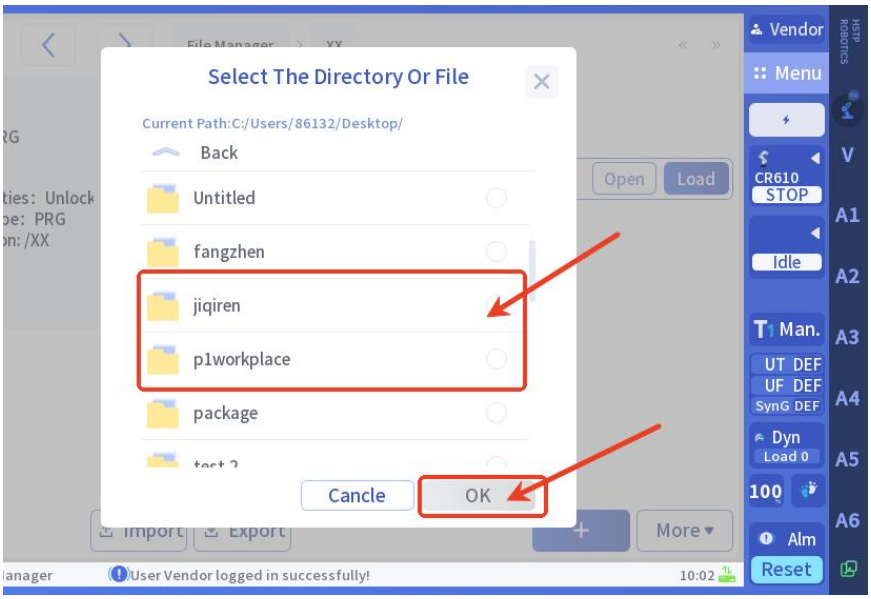


1、Navigate to the directory you want to import into.

2、Click the [Import] button to open the file selection dialog.

3、Select the files or folders you want to import in the dialog (multiple selections across directories are supported).

4、 Click [Confirm] in the file selection dialog.



If a file with the same name already exists in the current directory, a prompt will appear. You can choose to rename or overwrite the file. The renaming option will follow the software's preset rules.

attention ：

- The import function requires a USB drive to be inserted. If the drive is not connected, clicking the button will display a pop-up message: "No USB

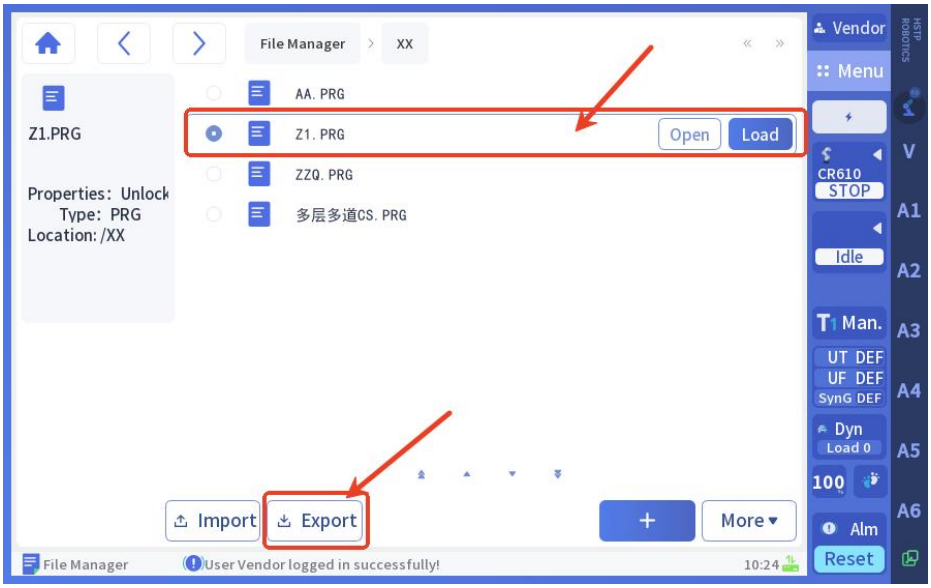
drive detected."

- Selecting a folder for import will only import the folder and its PRG files. Other files will not be imported.
- No single PRG file can exceed 3MB in size. If the file exceeds this limit, it will not be imported, and a pop-up message will appear: "Program file: xxx is too large. Cancel import. Split into subroutines for import."

Export Instructions

The export feature exports the controller system to a target path for backup purposes.

Export steps

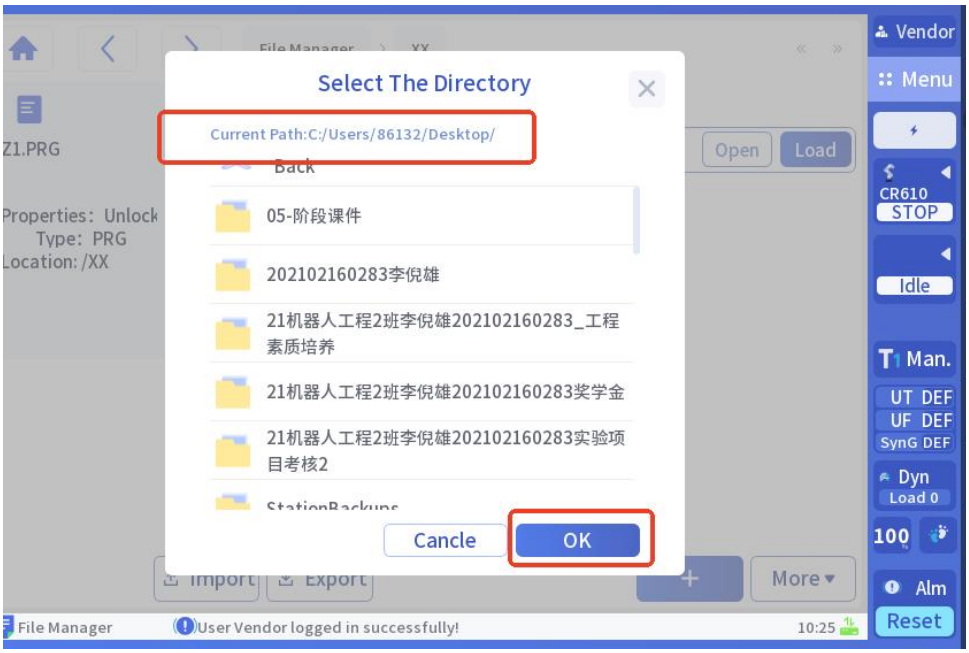


1、Select the PRG file or folder to export.

2、Click the [Export] button to open the directory selection dialog.

3、Choose the destination directory in the dialog.

4、Click [OK] to confirm.



If any exported file shares a name with a file in the export directory list, a pop-up window will appear. Users can choose to rename or overwrite the file. The renaming option will follow the software's preset renaming rules.

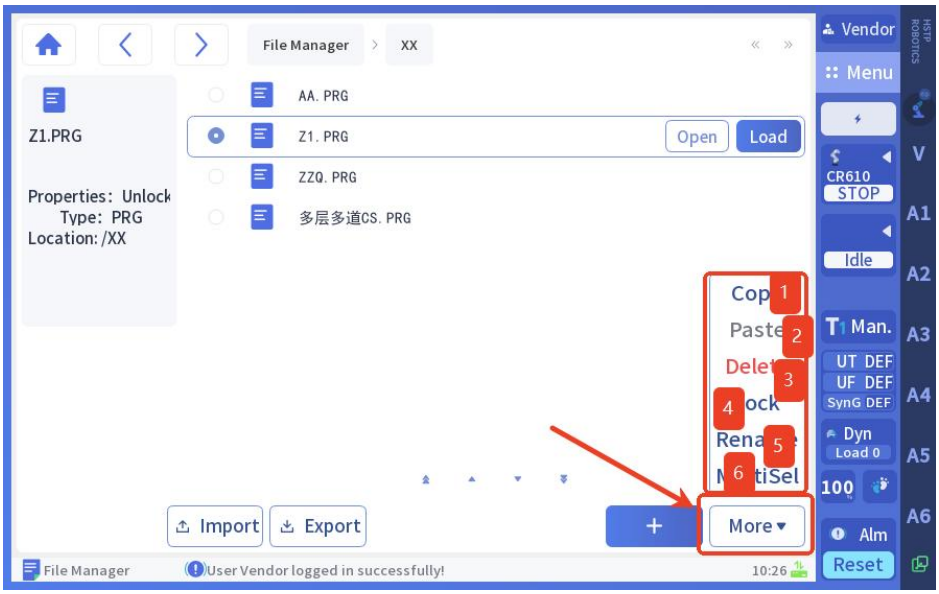
attention ：

- To export, you need to insert a USB drive. If you click the button without a USB drive, a pop-up message will appear: "No USB drive detected."
- To export, select a PRG file or folder first. If no PRG file or folder is selected, a pop-up message will appear: "Export program/folder failed. Select the program/folder to export first."。

- Note:
- The root directory for import and export is configured in the system file management settings. For details, see the program path configuration section.

4.1.1.6 More actions

Introduction Use the [More] button to organize additional rarely used operations. Clicking [More] will display a menu with various rarely used file operations.

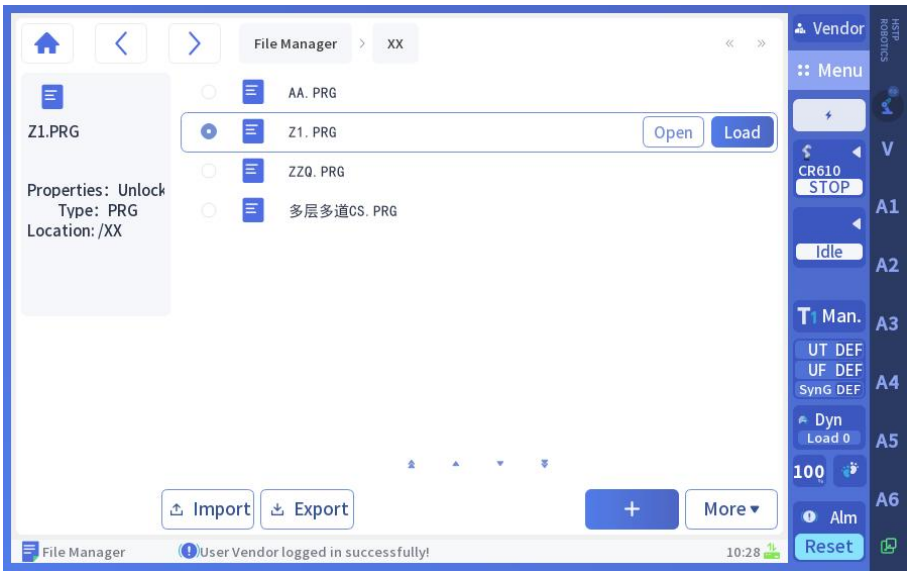


Label	Introduction
1	Copy menu item. This action copies selected files. Clicking updates the copied file object only and should be used with the [Paste] menu item. If you've previously copied files, clicking this button again will change the copied file to the current selection Note: To copy, select a PRG file or folder first. If no PRG file or folder is selected, a pop-up message will appear: "Failed to copy program/folder. Select the program/folder to copy first."
2	Paste menu item. Copies files to the current directory. Click to paste the copied files from the selection. You can paste the same copied file multiple times until it changes. If a directory with the same name already exists in the current directory, the file will be renamed automatically according to the preset renaming rules in the software. If no file object is copied the menu item will appear gray and be unclickable.
3	Delete menu item. For details, see the deletion section in the program file management.
4	[Lock] menu item. This action locks selected PRG files by restricting modifications. Password protection is available for securing PRG files. When clicked, a confirmation pop-up appears. Select [OK] to lock the selected PRG file. If no PRG

	file is selected, the menu item turns gray and becomes unclickable. If the selected PRG file is already locked, the men item changes to [Unlock]. A password entry window will appea Enter the correct password and click [OK] to unlock the file.
5	Rename menu item. Rename selected files and folders. Click t open the rename dialog. After editing, click [OK] to complete th renaming. If the file name already exists in the current director a message will appear.
6	[Multiple Selection] Menu item. Quick access to multipl selection mode.

Multi-selection status description

Long-press the selected folder or file, or click the [Multiple Selection] menu item in the [More] button to enter the multiple selection mode in the program manager. The multiple selection mode in the program manager is shown in the following figure:



Unlike the single selection mode, the multi selection mode features only four buttons: [Delete], [Export], [Copy], and [Cancel]. The [Delete], [Export], and [Copy] buttons perform the same functions as in single selection mode, but now apply to multiple objects. The [Cancel] button exits the multi selection mode.

In multi-selection mode, selected files and folders no longer have the [Open] and [Load] buttons. To access a folder in the current directory, double-click the folder. The folder you double-click will not be selected.

Multi-selection supports cross-directory selection.

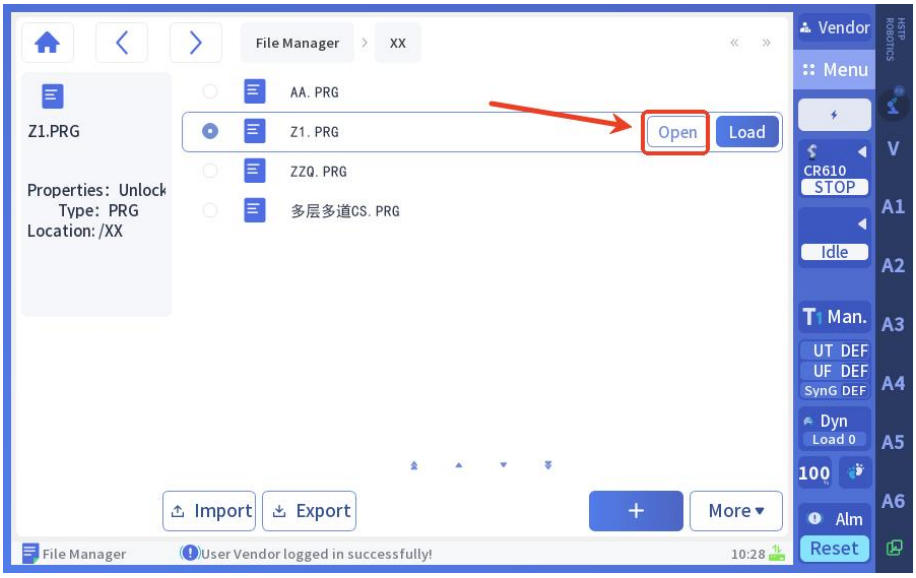
note：

- Open and loaded programs cannot be deleted, locked, renamed, or selected.

4.1.2 Edit program files

4.1.2.1 Open the program file

Operation notice In the program file manager, select the program you want to edit and click the [Open] button to edit the corresponding file. As shown in the figure below:



note：

- You can only open one program file at a time. If a program file is already open, the [Open] button for that file will disappear, but the [Open] button for the folder remains available.

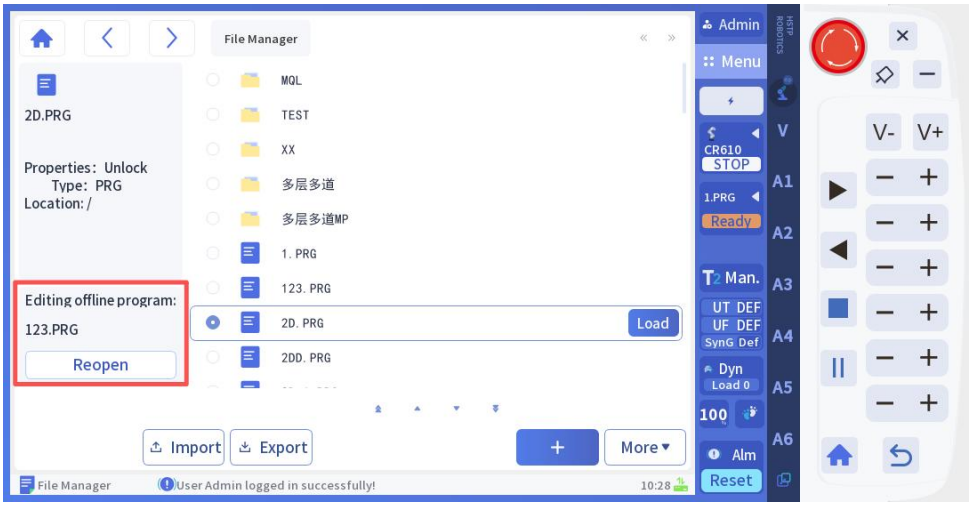
4.1.2.2 Continue editing

Statement When you open a program file and switch to another page in the teaching device, the [Open] button will not appear when you return to the file manager to select a file. In this case, you cannot continue editing the opened file using

the method described in Section 4.1.2.1.

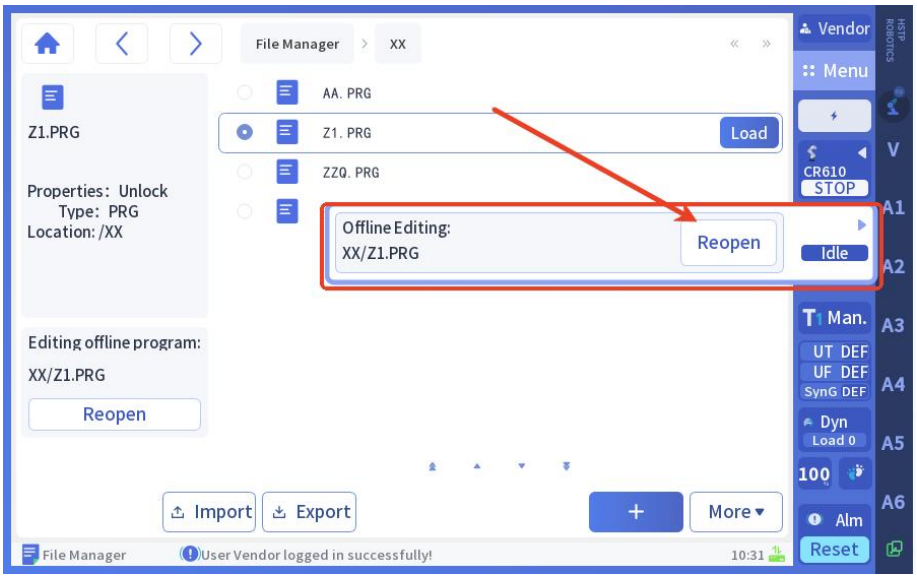
To continue editing an open file, you can do this in three ways。

Method1:File Manager restored.。Open File Explorer and click the [Restore Open] button on the left, as shown in the figure below.



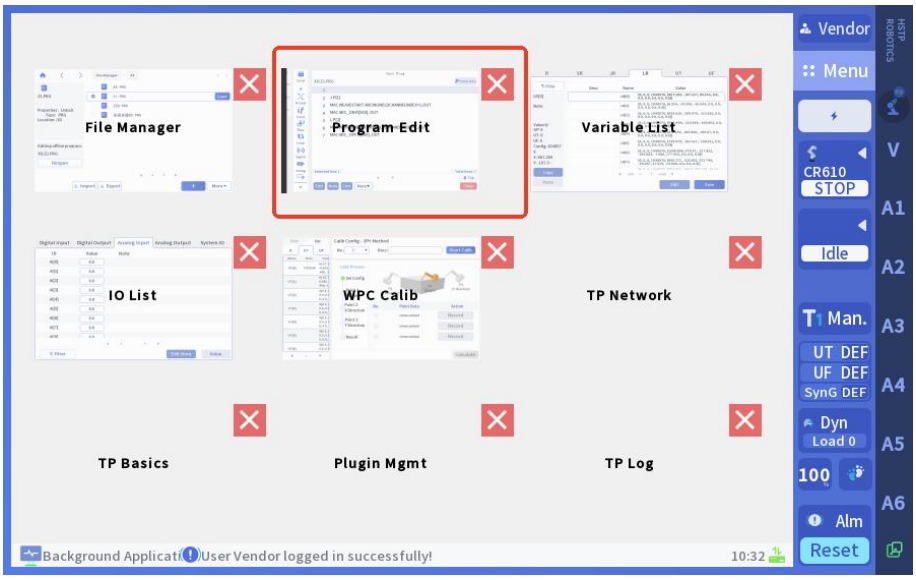
Method2:Restore from [Status Bar]

Click the "Program Status" control in the status bar; then click the [Restore Open] button in the pop-up window to return to the program editor page.



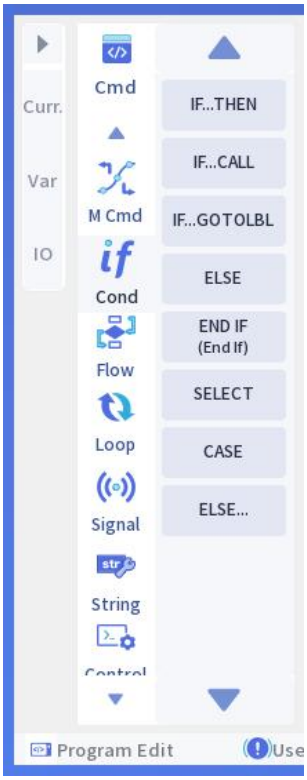
Method3: Return to the admin page

1. Press and hold the physical [ ] button to open the backend management page.
2. On the backend management page, click the Program Editor tab to return to the program editor page.



4.1.2.3 Operation interface

Introduction The program file editor is used to edit the contents of opened or loaded programs. The display of the program file editor differs for opened and loaded programs.The program file editor displays the opened program as shown in the figure below:

Label	Introduction
1	Widget controls. These controls allow users to easily view the robot's current pose, related register information, and IO status. The widgets in the diagram are collapsed. For details, see the widget section.
2	Instruction Information Control. This control allows users to add instructions below selected program lines. It categorizes instructions by type, enabling users to view all categories through buttons or drag-and-drop. Clicking a specific instruction type (e.g., "Conditional Instruction") will display the corresponding instruction control panel on the right side, as shown in the figure.  Click the instruction in the specific instruction control to add the corresponding instruction below the selected program line in the program content control of tab 3. For details, refer to the Insert Instruction section in the program file editing.
3	Program content control. Displays information about opened or loaded programs and allows partial editing of selected program lines. The program name appears in the upper-left corner, shown as "CONTORL.PRG" in the figure. If the current program is a background program, it is marked with "[*]" after the name, such as "XXXX.PRG[*]".

	In manual swipe mode, you can view the full program content by dragging or using the bottom [▲] and [▼] buttons. After selecting an editable program line, corresponding operation buttons will appear behind it, such as the [Record] and [To Point] buttons in the figure. The operation buttons are not fixed and will change according to the current status and instructions. When the current point matches the recorded point in the program line, a marker appears to the left of the line number, indicating the match.
4	The [Location] button opens the P-point information table. When clicked, the table appears above the program content. For details, refer to the P-point information section in the program file editing.
5	The [Record] button records the point position of the selected program line. It appears after motion commands. Clicking it records the current point position as the target point data for the selected program line.
6	The [Go] button enables the robotic arm to reach the target point specified in the selected program line. It appears only after the motion command and executes the movement according to the programmed motion pattern.
7	The [Back to Top] button. Returns quickly to the top of the program content control.
8	[Edit] button. Modify the selected program line. Requires debug privileges or higher. Click to enter edit mode, where you can modify the selected line. For details, see the Edit Instructions section in the program file editing guide.
9	Note button. Use to add or remove comments on the selected line. Requires debug administrator or higher permissions. For details, see the Notes and Comments section in the program file editing guide.
10	Note button. Add a note at the end of the selected line. Requires debug administrator or higher permissions. For details, see the Notes and Comments section in the program file editing guide.
11	The [More] button is a feature for organizing less frequently used operations. When clicked, a menu with less common

	program actions will pop up. For details on the specific content and functions in the menu, please refer to the More Operations section in the Program File Editing guide.
12	[Close] button. Closes the opened program and exits the program file editor page.

The program file editor displays the loaded program as shown in the figure below:

For loaded programs, the program file editor displays additional controls labeled with tags. The following describes these controls:

Label	Introduction
1	Program run identifier. A way to identify the current program run and the instructions to which you can go back. When the program is loading, an arrow icon appears to the right of the line number. Indicates the current line of the program being executed, and Indicates the line to which the program will revert when backing up.
2	The current line number control displays the line number of the running pointer. The selected line number control marks the line number of the currently selected command line.
3	The [Auto Scroll] button toggles between auto and manual

	scrolling modes. When auto scrolling is enabled, the button displays "Auto Scroll" and the content control follows the current line number, keeping it within the visible area. If the line number goes out of view, it automatically scrolls to the current line number to maintain visibility. In manual mode, the button shows "Manual Scroll" and the content no longer follows the line number, allowing users to scroll through the program content as needed.
4	The Online Edit Button activates and deactivates the online editing mode. Requires Administrator privileges. Access is only permitted in Manual mode when the program is in Ready or Paused states. Otherwise, a pop-up will display: "Online editing is not allowed in non-Manual mode!" When clicked, the program file editor enters online editing mode, with the program status changing to Wait mode. The auto-scrolling interface switches to manual mode, and the button changes to "Exit Edit". Clicking again terminates the mode. For detailed instructions, refer to the Online Editing section of the program file editing documentation.

Interface entry path

- Open the selected item in the current directory list of the program file manager by clicking the [Open] or [Load] button.
- Open through the [Restore] button in the Program Files Manager。
- Open the floating window in the status bar, click the loaded program name or the [Restore] button to proceed.

4.1.2.4 Point Information

Introduction

The point table displays all P-point information in the current program. Click the [Point Info] button to expand the P-point table, as shown in the figure.

Label	Statement
1	The drop-down menu for motion-to-point selection offers two options: "Joint" and "Straight Line". This should be used with the [Motion-to-Point] button in Label 4.。
2	The [Delete] button deletes the selected P-point. Requires administrator or higher permissions. A confirmation pop-up will appear. Click [Confirm] to delete the selected P-point.。
3	The [Overlay] button applies overlay to the selected P-point locations.。You need debugging privileges or higher. After clicking, a confirmation pop-up will appear. Click the [Confirm] button to overwrite the selected P point with the current robotic arm position. The overwritten position type will match the original type of the selected P point.
4	The [Move to Point] button moves the robotic arm to the selected P point. Requires administrator or higher privileges. The movement method is selected from the drop-down menu under Label 1 Movement to Point.。

Warning：

- Deleting the P point in use may cause errors.。

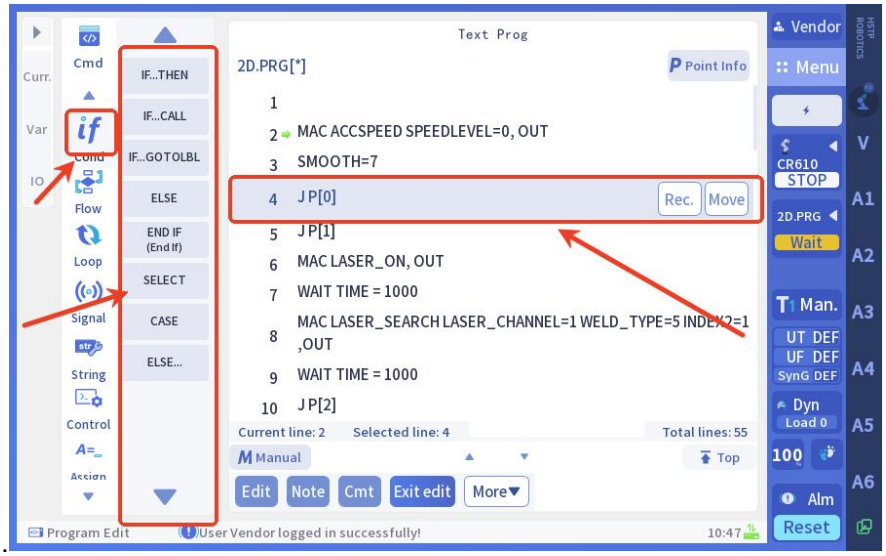
4.1.2.5 Insert instruction

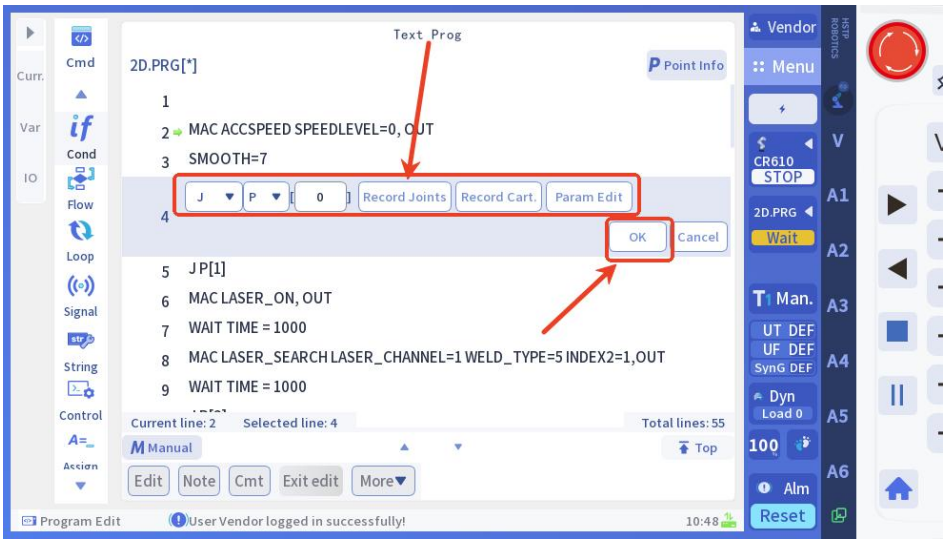
Introduction

Inserting a command means adding a new command. You can select the target location to insert the command.。

Operating steps

1. Select the program line where you want to insert the instruction. The instruction will appear below the selected line.
2. Choose the instruction type from the instruction information control, and the specific instruction control will pop up.
3. Click the desired instruction in the control.
4. Configure the relevant information. After configuration, click [OK].





note：

- Click the [Cmd] button at the top of the command information control to hide the specific command control

4.1.2.6 Modify instruction

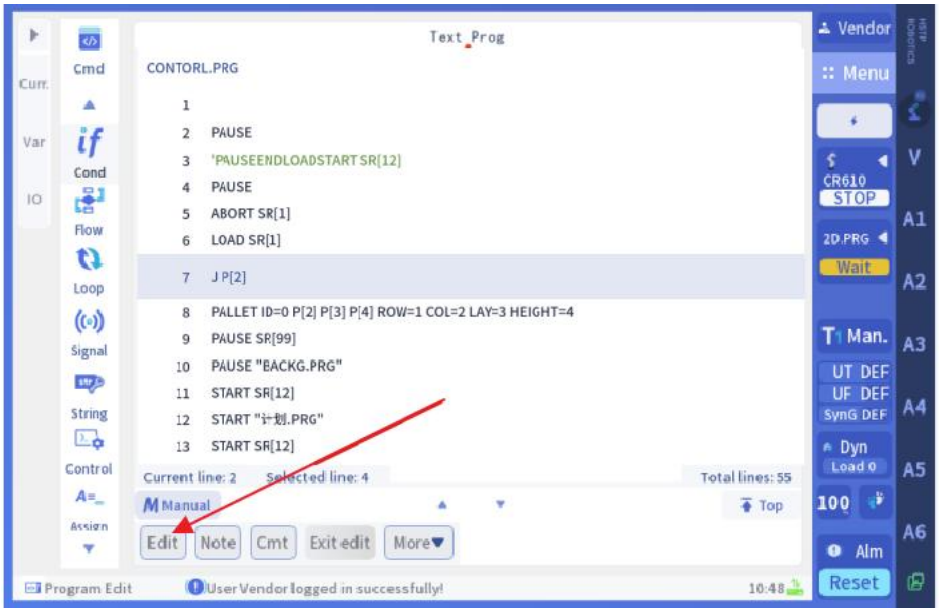
Introduction

Modifying a command means selecting the target command and modifying its content.

Operating steps

This section explains the general modification process for program instructions.

1. Select the instruction you want to modify.
2. Click the "Modify" button, as shown in the image below.



3. Modify the instruction content.

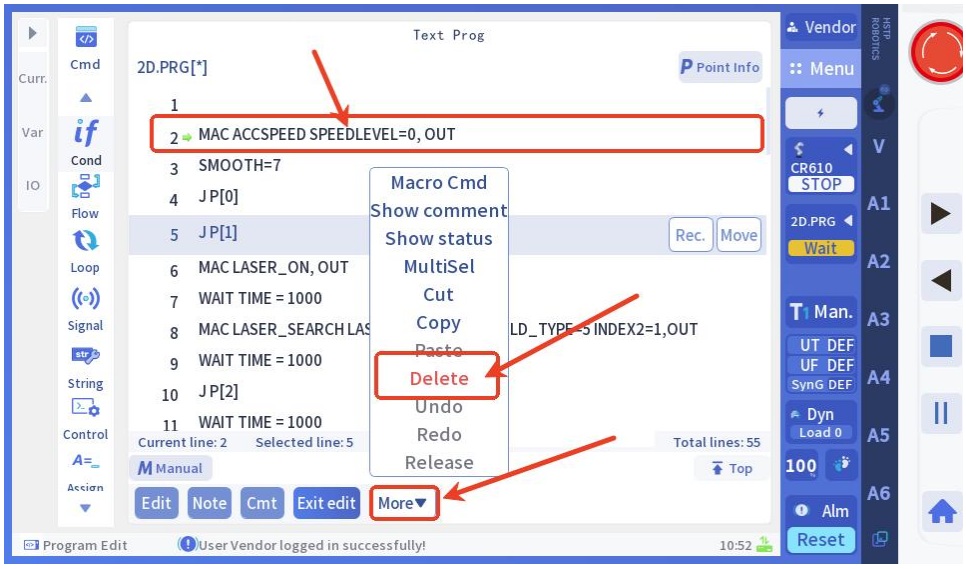
After modification, click the "OK" button.

4.1.2.7 Delete instruction

Introduction

Operating steps

1. Select the command you want to delete.
2. Click [More] to open the additional operation menu.
3. Click the Delete menu item



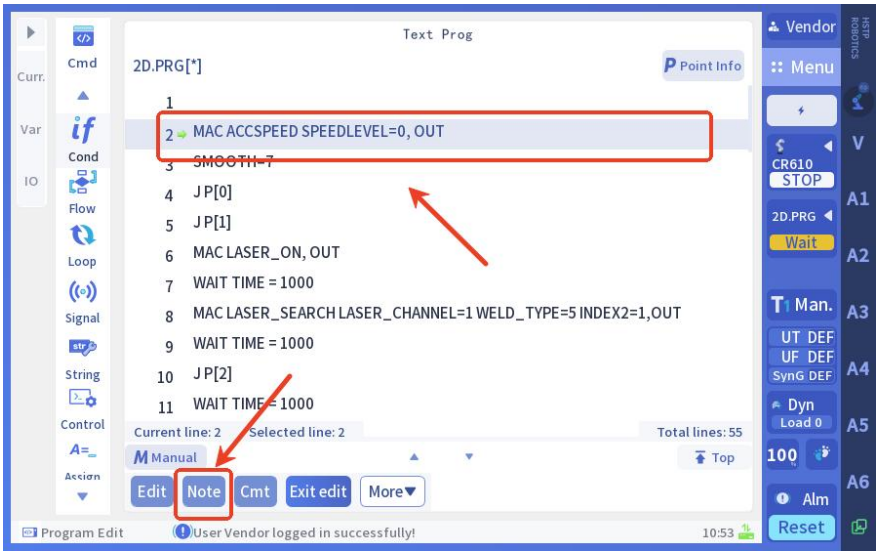
note：

- The deleted command can be restored through the [Undo] menu item in the More Actions menu.

4.1.2.8 Comments

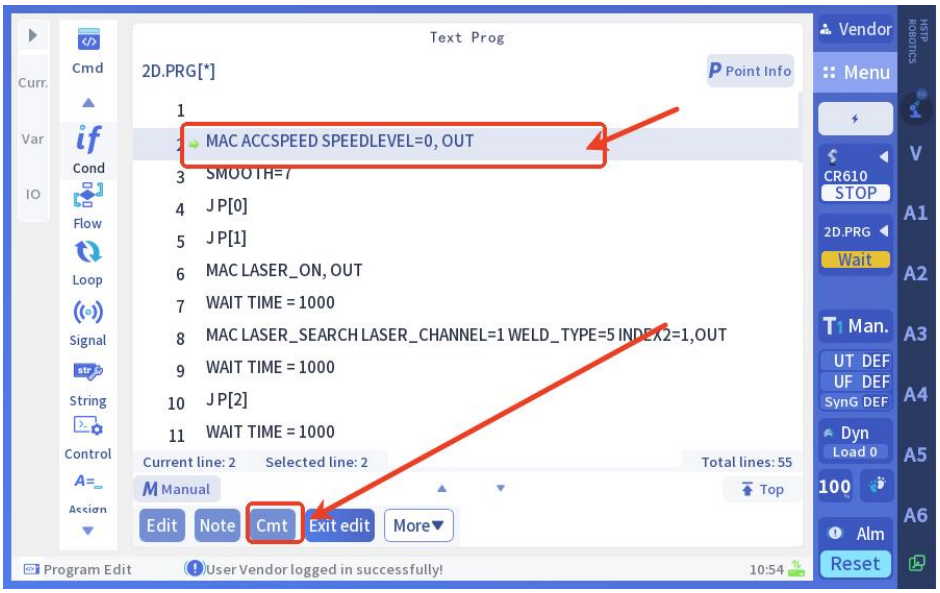
Introduction Select the command you want to comment. Click the [Comment] button. Click [Comment] again to remove the comment.

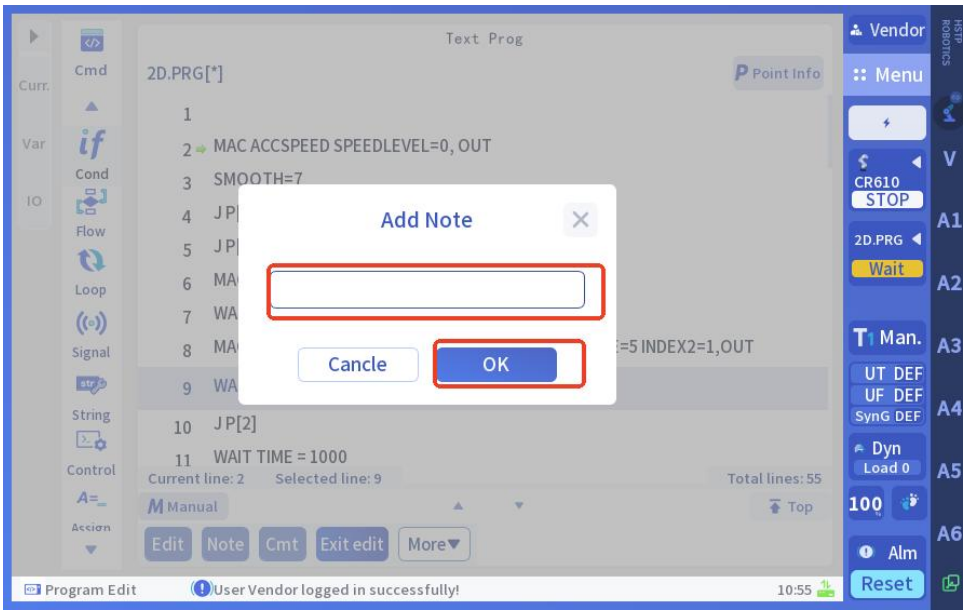
Operating steps



Remark will not affect the program execution

Operating steps

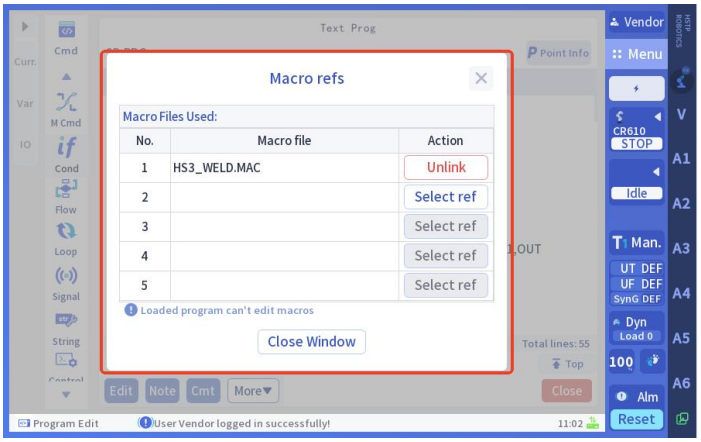




Select the command you want to add a remark to. Click the [Note] button to open the add note dialog. Enter the note content in the input field. Click [OK] to confirm. If the command already has a note at the end, the note content will automatically appear in the input field for modification.

4.1.2.9 More actions

Introduction

Label	Introduction
1	[Macro Reference] menu item. This adds a macro reference statement to the program. Clicking it will open a macro reference configuration pop-up window, as shown in the figure.  Click the [Select Reference] button to display a pop-up window showing only macro files with the ".MAC" extension. Select the desired macro file and confirm. The corresponding row in the macro reference configuration window will display the selected file name, and the operation button on that row will change to [Cancel Reference]. To remove the macro reference, click [Cancel Reference]. Only the previous row can be configured after completing the current one. The maximum number of rows is 5. After configuring the macro file, click [Close Window] to exit.

2	[Show Comments] menu item. Displays comments for register variables and DIO in the program, including corresponding descriptions in the variable list and DIO list. When clicked, a colon and comment will appear after the register number in the program. When not displaying comments： R[1]=-1 Show comments as： R[1:显示]=-1 The annotation for "R[1]" is "Show". After displaying the annotation, the button changes to [Hide Annotation]. Click again to hide it.
3	[Display Status] Button. Shows input/output status in the program. Requires debug level or higher. Click to add a colon and status after input/output numbers. When not displaying status： DO[5]=DI[96] When displaying status： DO[5: OFF]=DI[96: ON] The status of "DO[5]" is "OFF", and the status of "DI[96]" is "ON". After displaying the status, the button changes to the [Hidden Status] button. Clicking it again hides the status.
4	[Multiple Selection] Menu item. Enters the multiple selection program line state. Requires debugger or higher permissions. Cannot add commands or use the [Modify] or [Comment] buttons in multiple selection mode.
5	[Cut] menu item. This action cuts the selected program line. Requires debugger or higher permissions. When clicked, the selected line disappears and the copied instruction object is updated. Use with [Paste]. If you've previously copied or cut an instruction, clicking again will change the copied instruction to the current selection.
6	[Copy] menu item. Copies the selected program line. Requires debug privileges or higher. Must be used with the Paste menu item. Clicking updates only the copied instruction object. If you previously copied or cut an instruction, clicking this item again will change the copied instruction to the current selection.
7	[Paste] menu item. This action copies the selected program line and pastes it below the current line. Requires debug level or

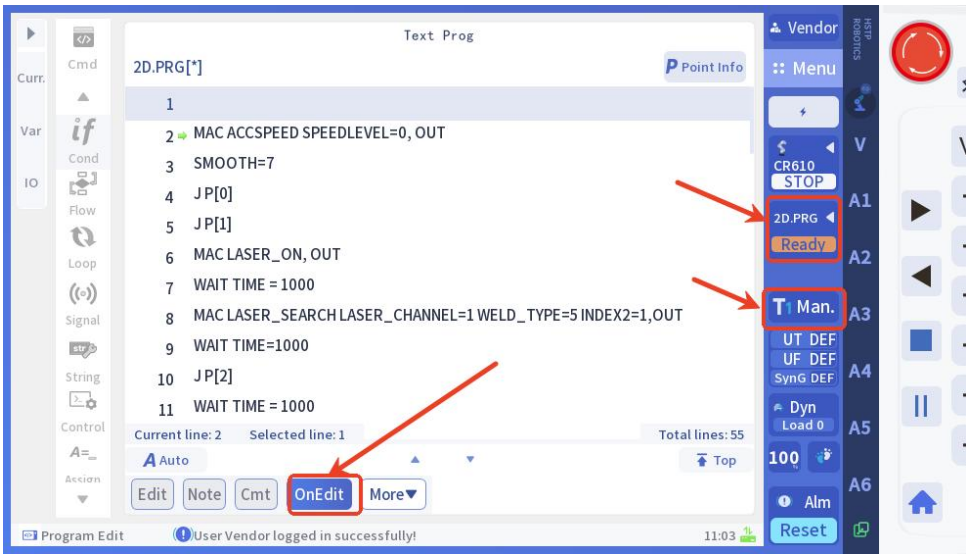
	higher permissions. Click to paste the copied file object from the selection. You can repeat this paste action until the copied object changes.
8	[Delete] menu item. Delete the selected command. For details, see the _Delete Command section in the program file editing guide.
9	[Undo] menu item. Reverts the previous action. If no action can be undone, the menu item appears gray and is not clickable.
10	[Redo] a menu item. Reverts the previous undone action. The menu item turns gray and becomes unclickable if no action can be undone. Note: If you performed another action after undoing, you cannot redo it.
11	[Cancel Wait] menu item. Forces the termination of the delay command "WAIT TIME" and the wait condition command "WAIT..." in signal commands. This menu item is grayed out and unclickable unless the current program is executing these commands.

4.1.2.10 Edit online

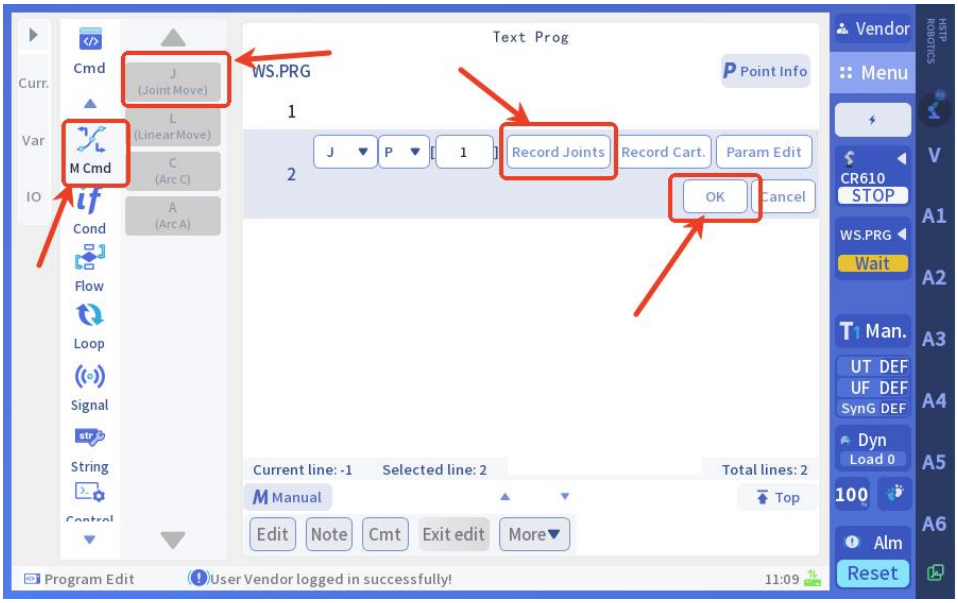
Introduction

In manual mode, when the program is in standby or paused state, open the program file editor for the loaded program and click the [Online Edit] button to enter online editing mode. During this state, the program status changes to "Waiting" while the automatic screen switching mode automatically transitions to manual mode. At this point, users can manually modify or insert program commands without uninstalling the application. For example, users can perform point-to-point movements on the robotic arm, insert new motion commands, record updated positions, and resume program execution. The program will then execute movements according to the updated content.

1. Open the loaded program and confirm that it is in manual mode and in a ready or paused state.
2. Click the [Online Edit] button.



3. Enable the mobile robotic arm position.
4. Select the position for the insertion command.
5. Choose the motion command type and insert the specific command, such as the J motion command.
6. Record the point and click [OK].



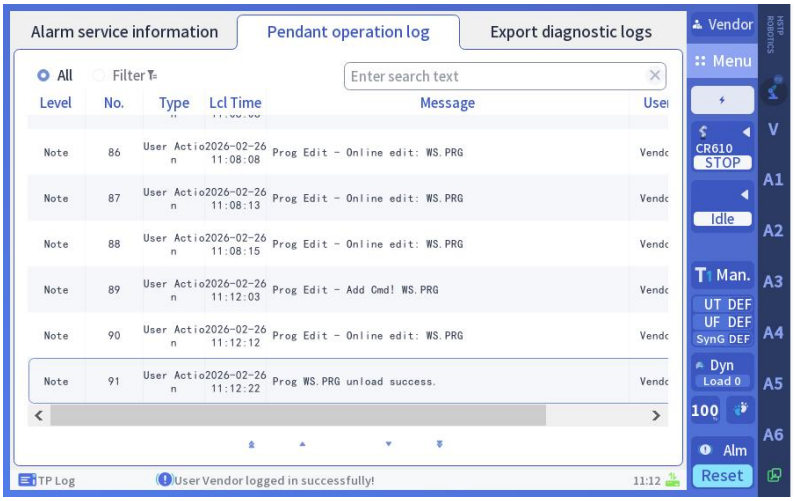
7. Click the [Exit Edit] button. Alternatively, click [Run] in the program status menu of the status bar or press the hardware [▶] key to resume program execution.

4.2 Diagnose

4.2.1 Teach pendant operation Log

Introduction

The teaching device logs user operations and saves them to its operation log for troubleshooting. The operation log can be viewed on the [Teaching Device Operation Log] page, as shown in the figure below.



Path

- There are two ways to enter the [Teachpendant Operation Log] page.
- Access through the [Menu] page: [Menu] → [File Information Management] → Diagnostic: [Teach-Back Operation Log]。
- Step: Go to the [Alarm Service Information] or [Export Diagnostic Log] page, then click the [Teach-Button Operation Log] tab at the top of the page to proceed.。

Log content description

The log must include the following: level, sequence number, type, local time, information, and user.

Levels: prompt, warning, and error.

Sequence number: The serial number assigned to log generation. Type: Includes startup trainer, configuration trainer,

user operation: configuration controller, commissioning, and operational process messages.

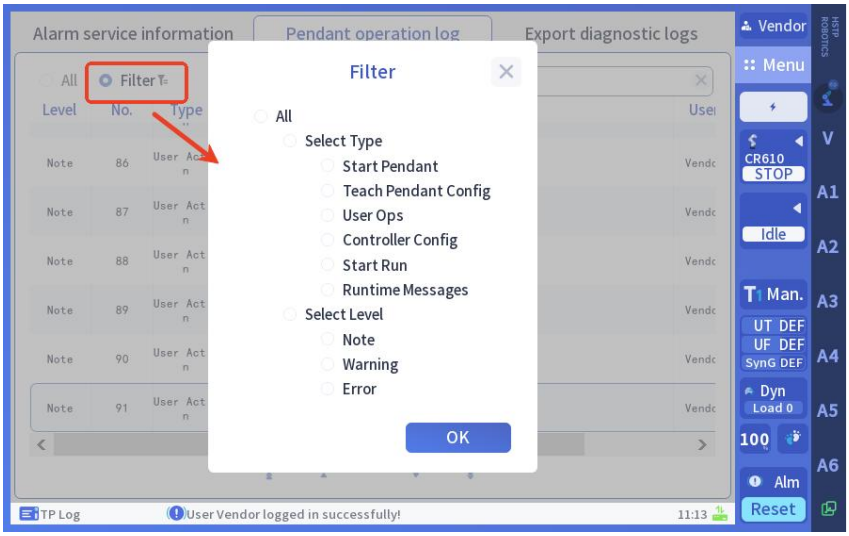
Local time: The system's local time for the trainer hardware.

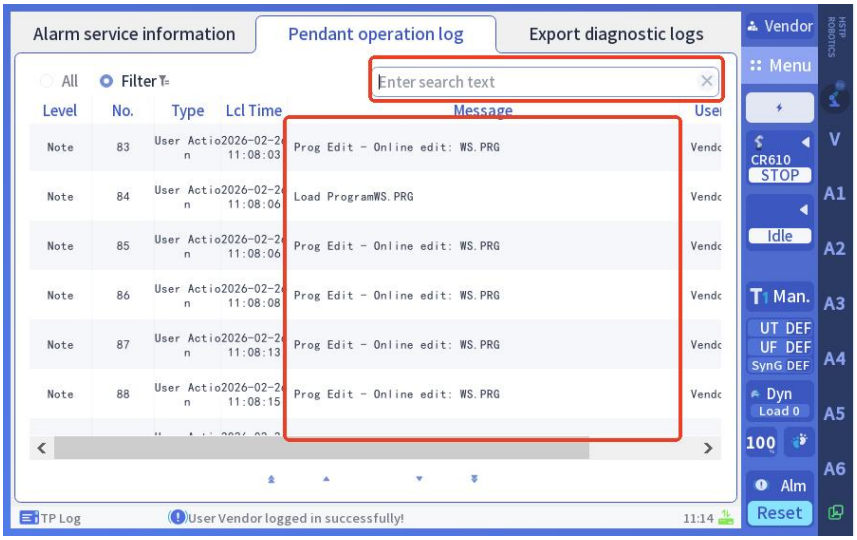
Message: The main content of the log.

User: The trainer's logged-in user during relevant operations.

Filter logs.

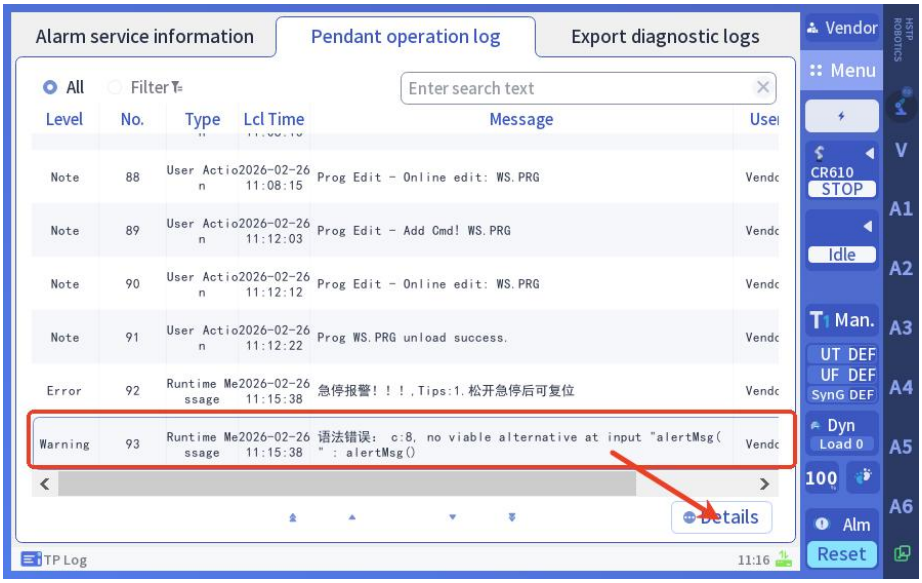
Click the [Filter] radio button on the page to open the [Filter] dialog. You can filter logs by combining log types and levels. After selecting, click [OK] to apply the filter.





detailed information

When the user clicks and selects the log item with the type "Run Process Message", the [Details] button appears in the lower right corner.



After clicking the [Details] button, the page will redirect to the [Details] page, as shown in the following image:

This page displays alarm details including the alarm ID (code), message, cause, and troubleshooting steps. Click [Previous] or [Next] to view the corresponding log entry. Click [Back] to return to the log list.

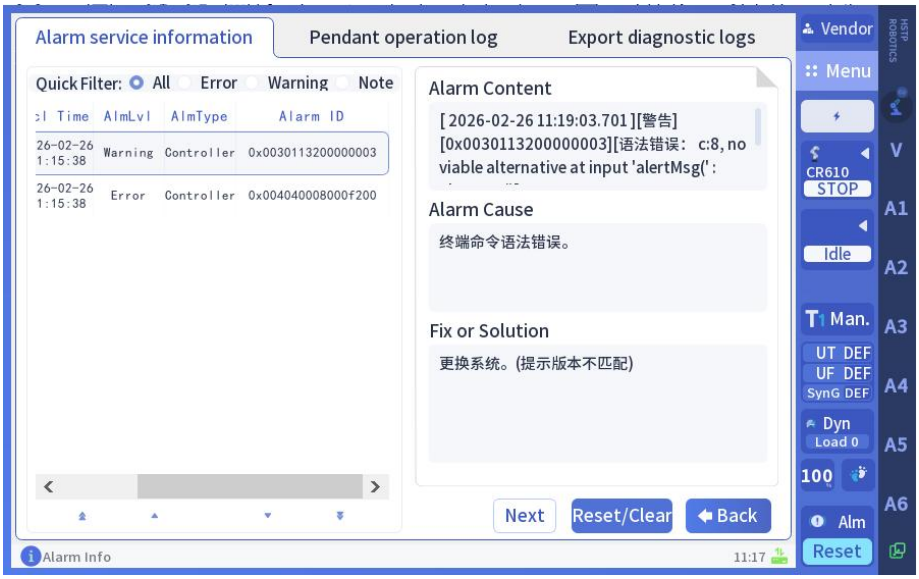
note：

- The information in the teaching device operation log is recorded locally and saved offline.

4.2.2 Alert service information

Introduction

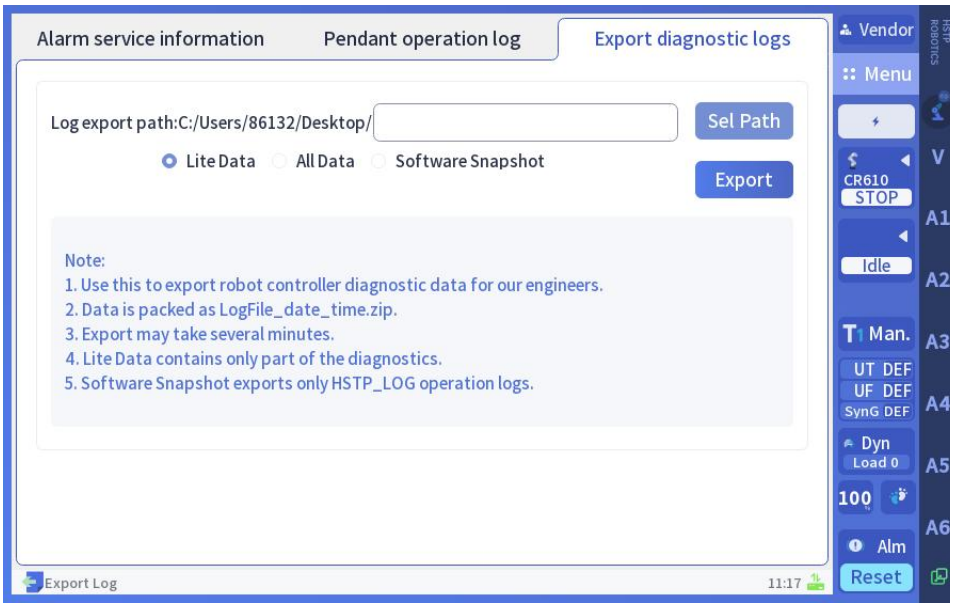
During operation, the teaching pendant may generate alerts from either the robot or the device itself. When new alerts are triggered, they initially appear in the bottom notification bar. However, if multiple alerts are generated simultaneously, viewing all of them through the notification bar becomes impossible. In such cases, users can access the "Alarm Service Information" page to check all pending alerts. The interface layout is illustrated in the following diagram:



4.2.3 Export diagnostic logs

Introduction

When a system fails, providing diagnostic logs to technical support helps quickly identify and resolve the issue. You can export logs from the teaching pendant and controller to a USB drive through the [Export Diagnostic Logs] interface. The operation interface is as follows：



operating steps

1. Insert the USB drive into the teaching programmer. The default export path is the root directory of the USB drive.
2. Click [Select Path] to choose the export path, then click [Export] to export. Wait a few minutes for the log files to be sent to the corresponding path.

Note：

- Export data according to technical specifications. Send logs to relevant personnel for troubleshooting.
- Maintain system environment and retrieve files when issues occur. Due to log volume, data retrieval may take some time,please be patient.
- For non-crash diagnostics, select [Compact Data] export to save time. This option meets most log collection needs. Unlike standard export, it excludes system snapshots and collected data.

4.3 System file management

4.3.1 Data backup

Introduction

The data backup feature enables the preservation of configuration data in controllers. When combined with data recovery capabilities, it proves essential for scenarios like system replacement, multi-machine

deployments, and routine debugging. Users can choose to back up to the instructor's local device, USB drive, or controller's USB drive, with operations tailored to specific backup requirements.。

Path

➢ Access the [Menu] page by following these steps: [Menu] → [File Information Management] → [System File Management] → [Data Backup].

4.3.1.1 Register file backup

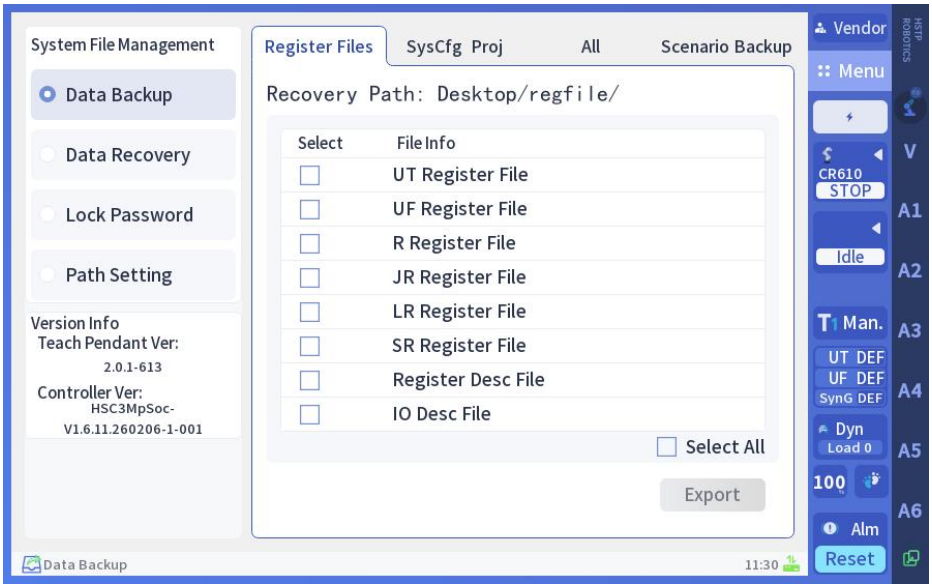
Introduction

Back up the data (including values and descriptions) in the UT, UF, R, JR, LR, and SR registers of the controller, along with the IO specification file.

Path

➢ [Menu] → [File Information Management] → [System File Management] → [Data Backup] tab → [Registry Files] entry。

operating steps



1. Insert the teaching device into the USB drive and ensure successful recognition.
2. Select the register files to export.
3. Click the [Export Backup] button to create a backup.。

note：

- The exported data files are stored in the (/regfile) folder. The button will

only activate after the USB is inserted and at least one backup file is selected.

- No corresponding program data files exist before factory release, so retrieval will fail.

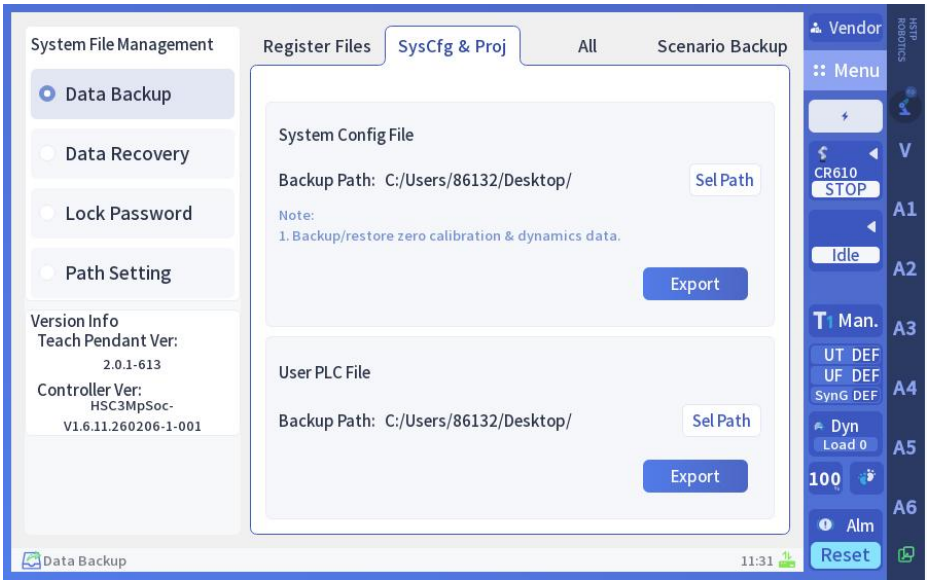
4.3.1.2 System parameters and project backup

4.3.1.2.1 System parameter file backup

Statement Back up the calibration parameters, dynamics, and other identification parameters of the controller system.

Path ➢ [Menu] → [File Information Management] → [System File Management] → [Data Backup] tab → [System Parameters and Projects]。

operating steps



1. The teaching device has been successfully recognized after inserting the USB drive.
2. The default backup path is the root directory of the USB drive. To change the path, click [Select Path] and choose another backup location.
3. Click [Export Backup]. Wait a moment, and you will see a message indicating that the backup file has been exported.

note：

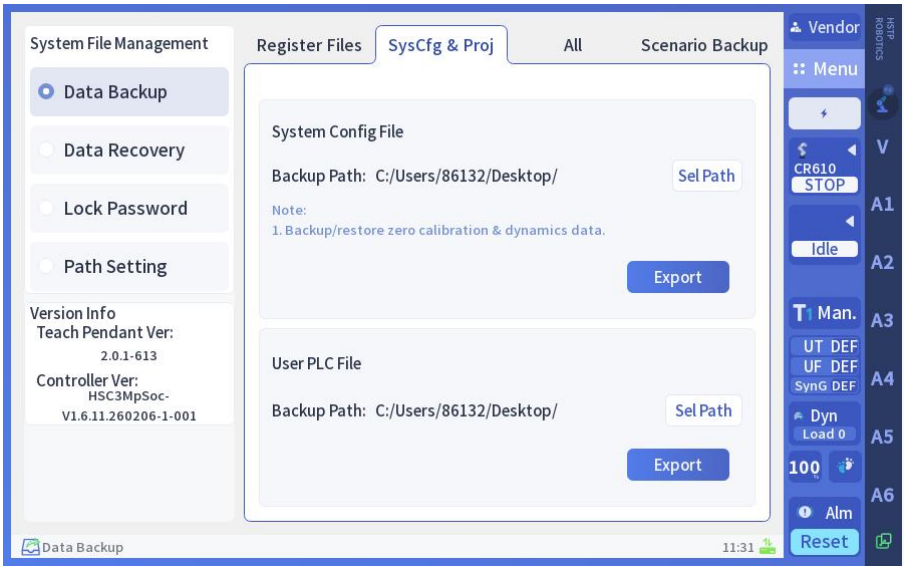
- The button requires the USB drive to be recognized before clicking.
- Backup completed. The generated/data/Calbration.zip files in the selected path

4.3.1.2.2 User PLC file backup

Statement Backup the user PLC program files in the controller system

path ➢ [Menu] → [File Information Management] → [System File Management] → [Data Backup] tab → [System Parameters and Projects]

Operating step



1、The teaching device has been successfully recognized after inserting the USB drive.

2、 The default backup path is the root directory of the USB drive. To change it, click [Select Path] and choose another backup path.

3、Click [Export Backup], and wait for the system to complete the export of the corresponding backup files.

note：

- The button requires USB recognition before clicking. After backing up the user's PLC file, the backup path is generated in the Application directory,

containing.app and.crc files.

4.3.1.3 Full backup

Statement

Full backup involves compressing all controller-related files, including register files, register descriptions, user PLC files, system parameter files, process package data, and engineering power-off retention variables. The backup can be stored locally on the teaching pendant or transferred to a USB drive.

Backup path description

You can select the backup path from the [Backup Path] dropdown menu, which includes the following options:

Path options	File Path	statement
Teaching device USB	/BackupAll	Insert the USB flash drive into the USB port of the teaching device. The [Export Backup] button can only be clicked after the USB flash drive is inserted.
Local	/opt/hstp/cache /sysbak/BackupAll	The backup file is saved locally on the trainer. Copy it to a USB drive separately.
USB in Controller	/BackupAll	Insert the USB flash drive into the controller's USB port

Path

➢ [Menu] → [File Information Management] → [System File Management] → [Data Backup] tab → [All] tab

operating steps

1、Select a suitable backup path.

2、Click [Export Backup] and wait for the completion prompt.

note：

- The backup file name is machine model-version number-backup-controller system date time.tar.gz

4.3.2 Restore of data

Introduction The data recovery feature restores backup files (including configuration and data files) to the controller system. Combined with data backup functionality, it proves essential in scenarios like equipment replacement or multi-machine deployments. Recovery paths include the teaching pendant's local drive, its USB drive, or the controller's USB drive.

Path

➢ [Menu] → [File Information Management] → [System File Management] → [Data Recovery]。

note：

- After successful import restoration, power off and restart to apply the restored data.。

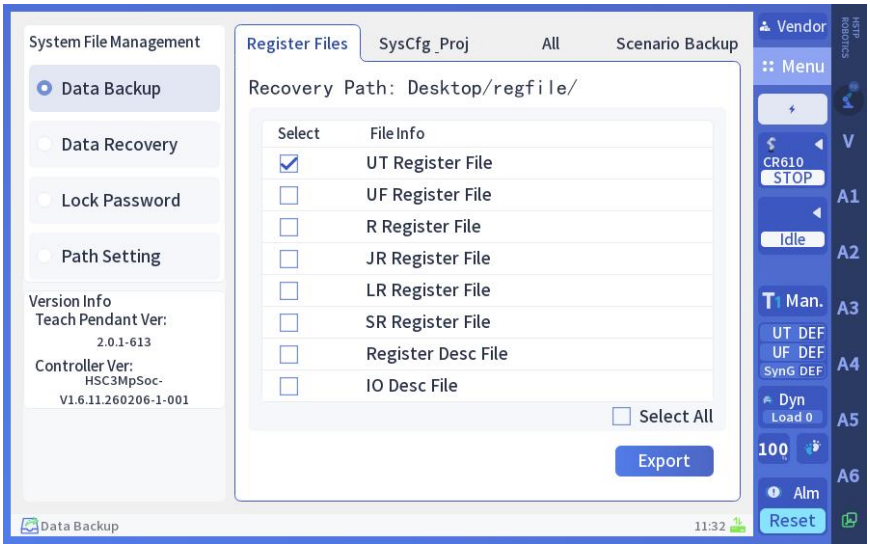
4.3.2.1 Restore registry file

Introduction The restore operation imports the controller's register files (variable list register file and IO specification file) via a USB drive.

Path

➢ Menu → File Information Management → System File Management: Go to the "Data Recovery" tab and select the "Registry Files" option.。

step



1、The teach device has been successfully recognized after inserting the USB drive.

2、Select the register files to import.

3、 Click [Import and Restore]. Wait a moment, then you will see the message "Imported successfully. Power off and restart the controller." The restored content will take effect after restarting.

note：

- The button requires the USB drive to be inserted and at least one recovery file selected before clicking.
- The recovery files will be searched in the root directory of the USB drive and the regfile directory. After successful recovery import, power off and restart the system to apply the recovery data.。

4.3.2.2 System parameters and project restore

4.3.2.2.1 Restore system parameters file

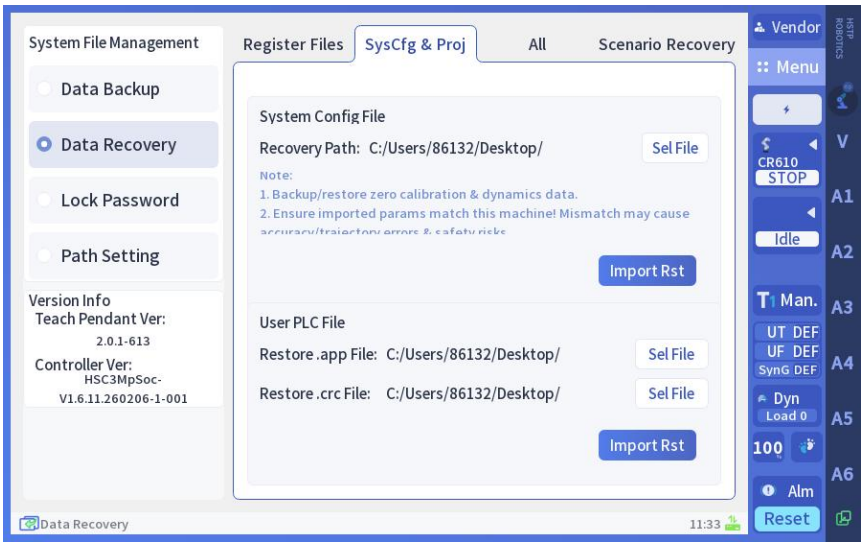
Introduction

Import the backup package of system parameter files via USB to restore calibration parameters, dynamic parameters, and other identification parameters.

Path

➢ [Menu] → [File Information Management] → [System File Management] → [Data Recovery] tab → [System Parameters and Projects]

operating steps



The teaching device has been successfully recognized after inserting the USB drive. Click [Select File] (Calbration.zip) to confirm. Then click [Import and Restore]. Wait a moment. The system will display "Imported successfully. Power off and restart the controller." After restarting, the imported content will take effect.

Note：

- The button requires a USB drive to be recognized before clicking. After successfully importing the file, power off and restart the controller.

4.3.2.2.2 Restore project files

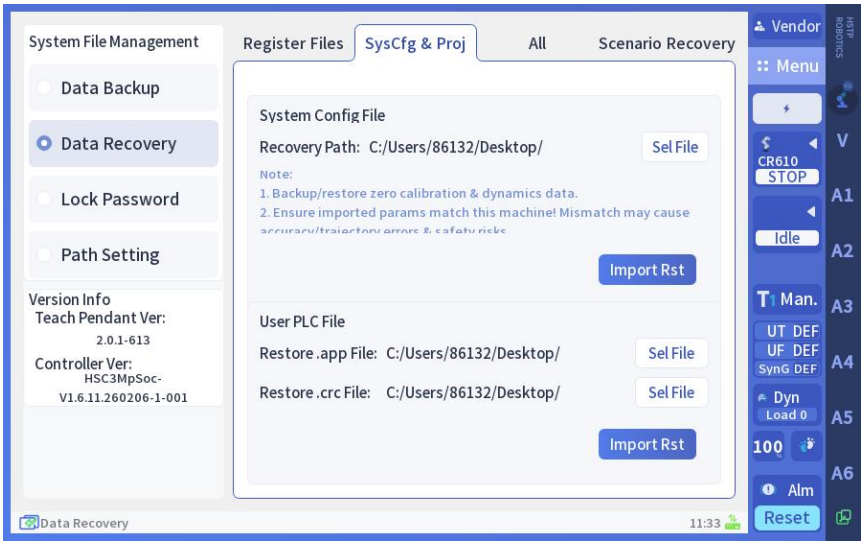
Introduction

You can import new user PLC program files via USB to update or restore the system.

Path

➢ [Menu] → [File Information Management] → [System File Management] → [Data Recovery] tab → [System Parameters and Projects]。

Restore UI



1、The teaching device is successfully recognized after inserting the USB drive. Click the [Select File] button in the Restore app to choose the target file. Similarly, select the [Select File] button in the Restore CRC app to confirm the target file. Finally, click the [Import Restore] button. The system

will display "File import successful. Power off and restart the controller!" After rebooting, the imported user PLC will take effect.

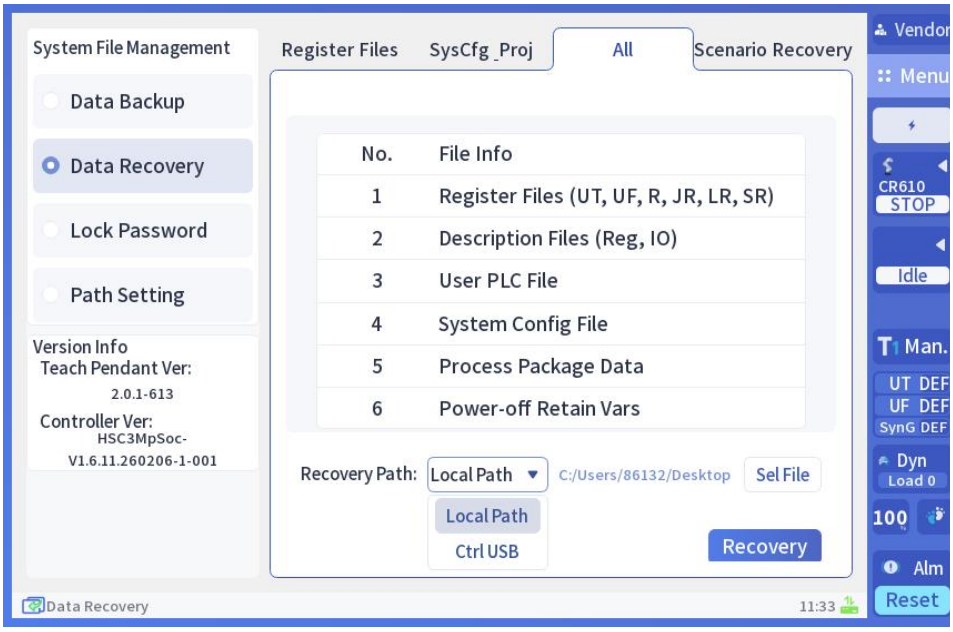
note：

- The button requires a USB drive to be recognized before clicking. After successfully importing the file, power off and restart the controller.

4.3.2.3 Restore all

Introduction

The full restore process compresses backup files containing register files, register descriptions, user PLC files, system parameter files, process package data, and engineering power-off retained variables. These files are then sent to the controller via the teaching pendant or USB drive, where they are decompressed and restored.



Path

➢ Access the [Menu] page through the following steps: [Menu] → [File Information Management] → [System File Management] → [Data Recovery] tab → [Restore All] option。

Restore path settings

To restore the path, select from the drop-down menu [Restore Path] as shown below.

Path options	File path	statement
Teaching device USB drive	/BackupAll	Insert the USB flash drive into the teaching device's USB port. The [Restore] button can only be clicked after the USB flash drive is inserted.
Local	/opt/hstp/cache/sysbak/BackupAll	Save to the teaching device
USB Controller	/BackupAll	Insert the USB flash drive into the controller's USB port

operation

1、Click [Select File] to confirm the selected file for restoration. Click [Restore with One Click] and you will see the message "File import successful. Please power off and restart the controller." After restarting, the restored content will take effect.

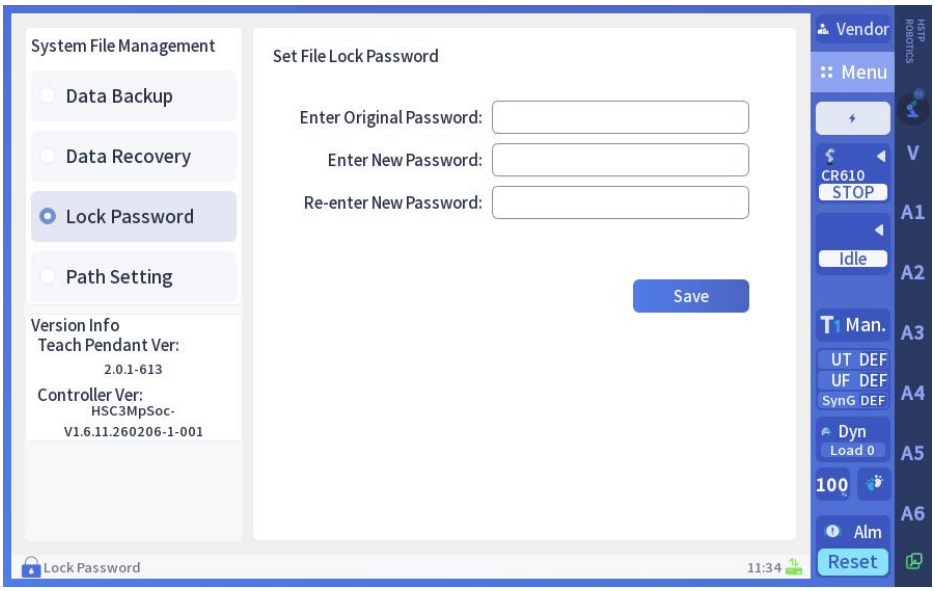
Note ：

- The restored file name is typically in the format: machine model-version number-backup-controller system date time.tar.gz (You can also modify the file name).
- After Import the file successfully. Power off and restart the controller!

4.3.3 Lock password settings

Introduction

In the file manager, users can lock specific program files to prevent accidental deletion or modification. The teaching device supports setting unlock passwords for program files, enabling more granular permission control. The operation interface is shown in the figure below:



Path

➢ [Menu] → [File Information Management] → [System File Management] → [File Lock Password].

operating steps

1、Enter the original password and new password. Click [Save] to save the new password.

note：

- The initial password is unlocked as "hstp".

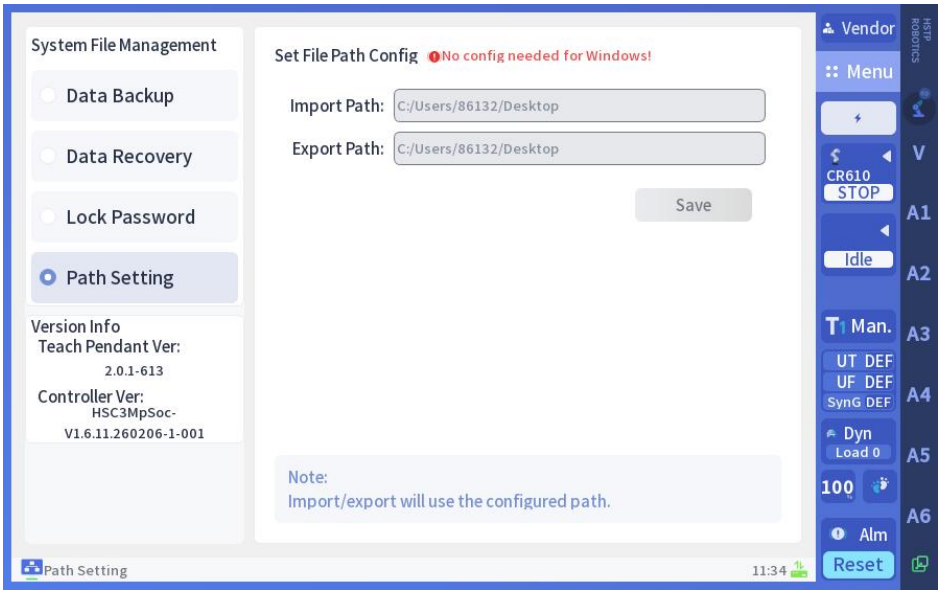
4.3.4 File path settings

Introduction

Path

➢ [Menu] → [File Information Management] → [System File Management]: [File Lock Password]

operating steps



1、The default path is the root directory of the U disk. With debug permissions, you can manually enter import and export paths based on the U disk root directory, as shown in the figure above. Click [Save]. When the system prompts that the settings are successful, the configuration is complete.

note：

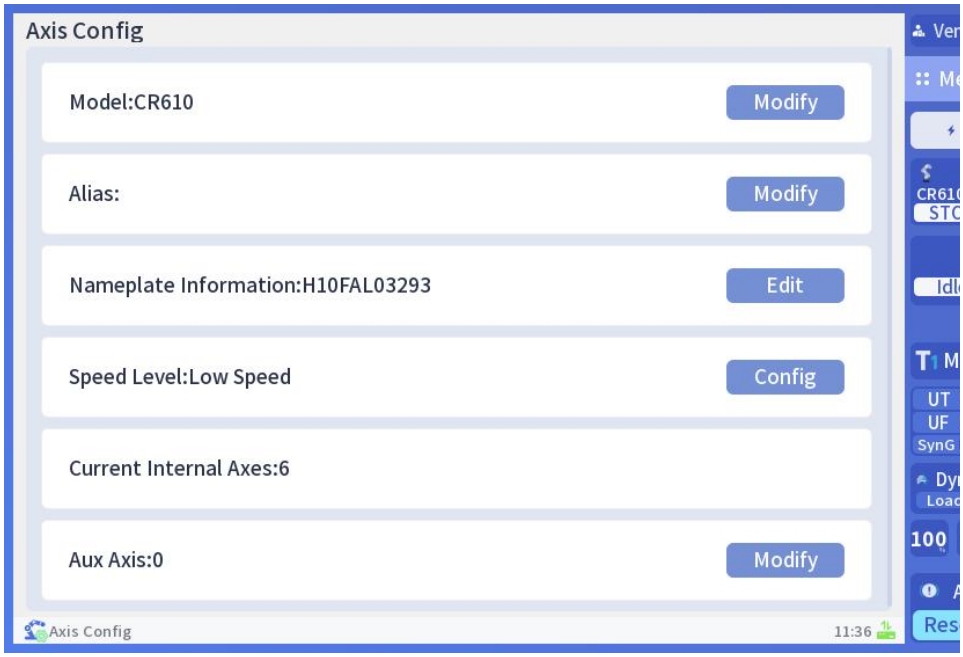
- For instructions on importing and exporting programs, see the file manager (Import and Export section).

4.4 Axis group configuration

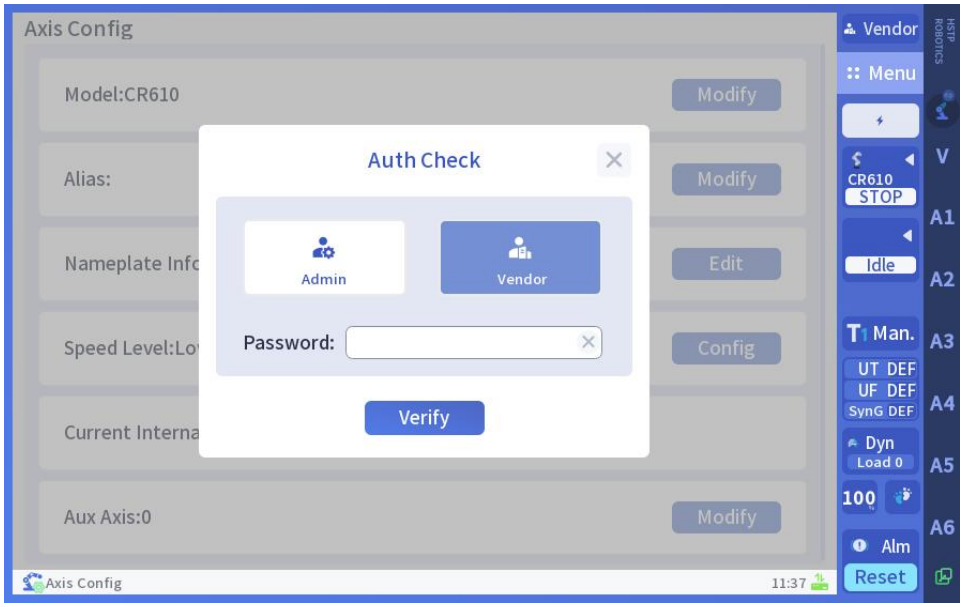
4.4.1 Axis group configuration

Introduction

The axis group configuration function allows you to set the current robotic arm's model, alias, speed level, number of axes, and additional axes. The page is shown in the following figure:



When modifying the configuration of various axis groups, a permission verification pop-up window will appear as shown in the red box in the following figure. You need to enter the administrator or manufacturer password. The relevant modifications can be made only after the verification is passed.



Path

➢ [Menu] → [Start] → [Axis Group Configuration]: [Axis Group Configuration]

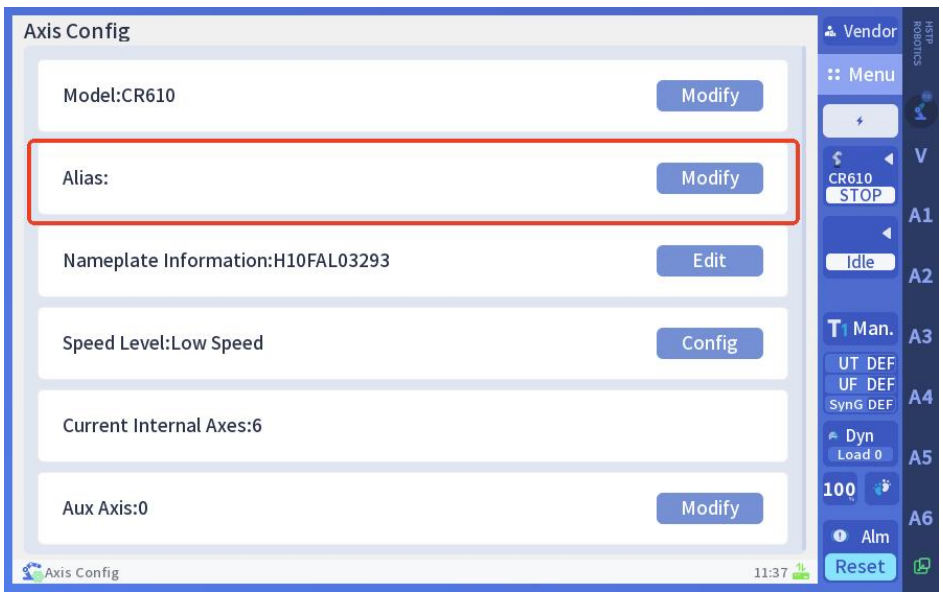
Switch Robot Type operation

1、Click the [Modify] button for the model; verify permissions (if already verified, this step will be skipped automatically); select the target model from the dropdown menu; click [Save]; restart the robot control system to complete the setup.

caution：

- After the robotype switch and system restart, recalibrate the zero point promptly.

Alias modification



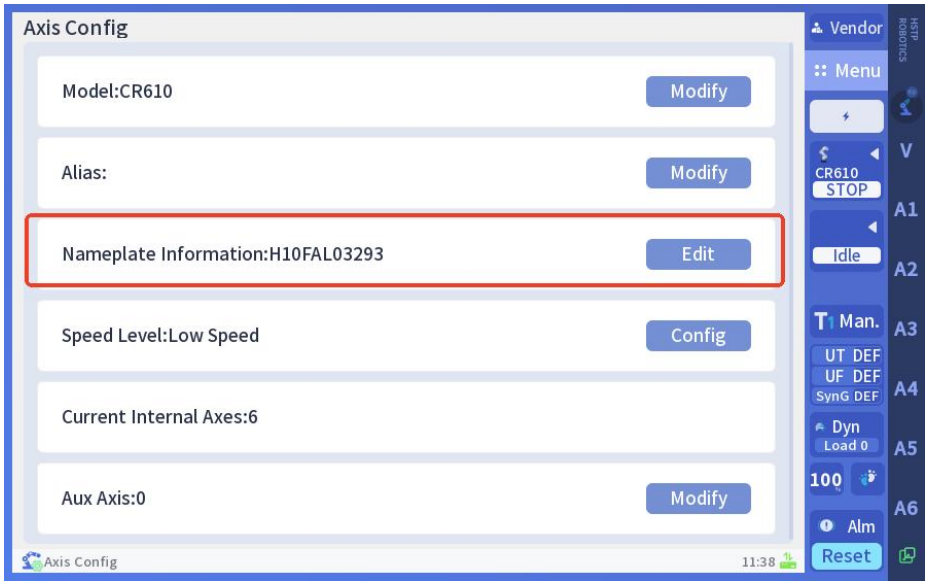
1、Users can assign aliases to robotic arms, such as "Robot 1", to help them quickly identify the devices. Follow these steps: 1. Click the [Modify] button for the alias; 2. Verify permissions (if already verified, this step will be skipped); 3. Enter the alias in the input field; 4. Click [Save] to complete the

modification.

note：

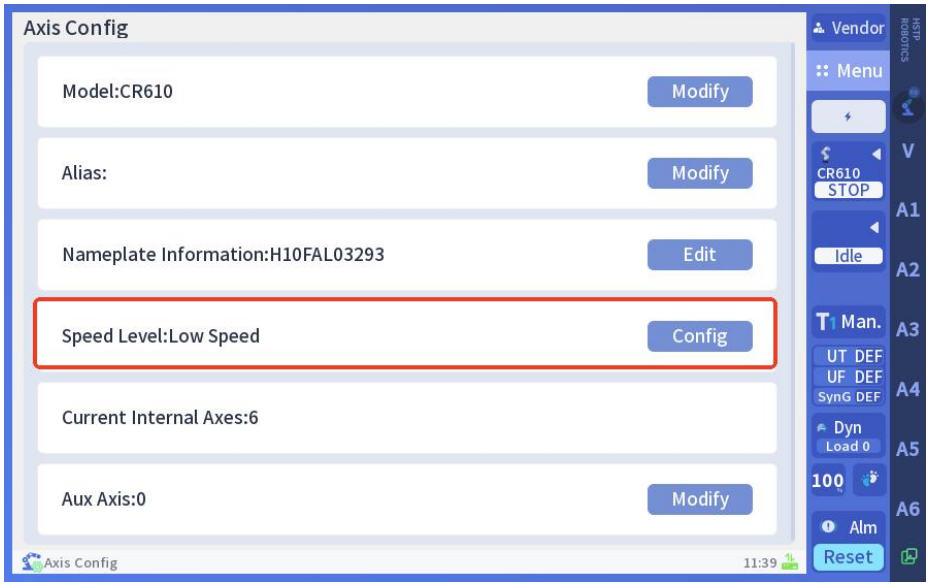
- Clear and save the alias input box to cancel the alias setting. After modifying the alias, the corresponding model name in the status bar will display the set alias.。

Edit nameplate information



2、Click the [Modify License Plate Number] button in the license plate information. Verify your permissions (if already verified, this step will be skipped). Enter the license plate information. Click [Save] to complete the modification.

Change speed level



Speed level is actually a set of preset values related to motion control and planning in the system. Users can adjust the values of relevant parameters according to the actual load of the robotic arm to achieve better control effect.：

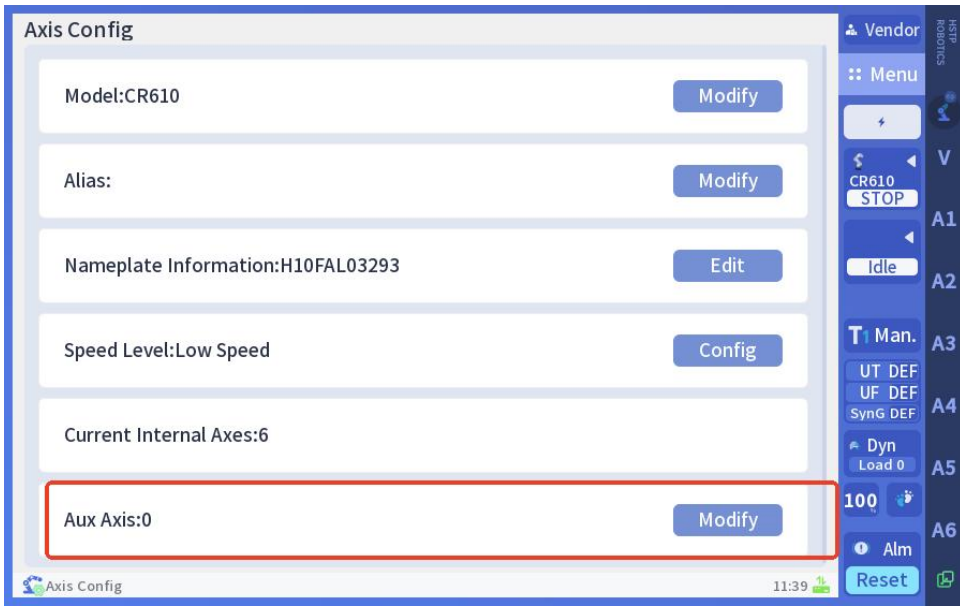
1、Low speed: Recommended for heavy loads.

2、Medium speed: Recommended for moderate loads.

3、 High speed: Recommended for light loads.

Configuration steps: 1. Click the [Configuration] button for the desired speed level. 2. Verify permissions. 3. Select the target speed level. 4. Click [Save] to complete the modification.

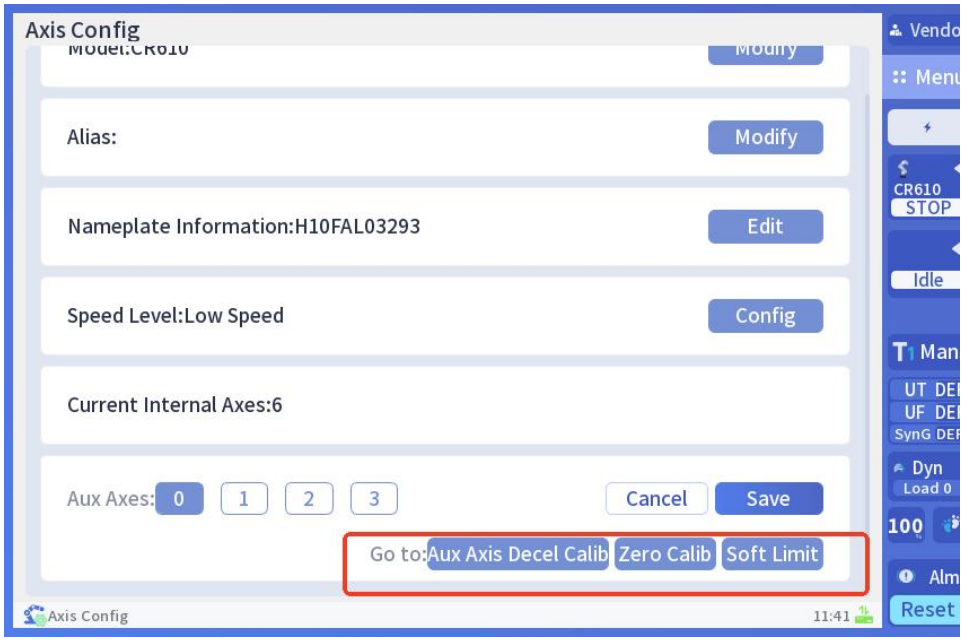
External Axis



1、Click the Modify button for the additional axis; verify permissions; select the number of additional axes; click Save to complete the modification.

note：

- Click the button in the red box below to quickly switch to the corresponding function page。



4.4.2 Collaboration group configuration

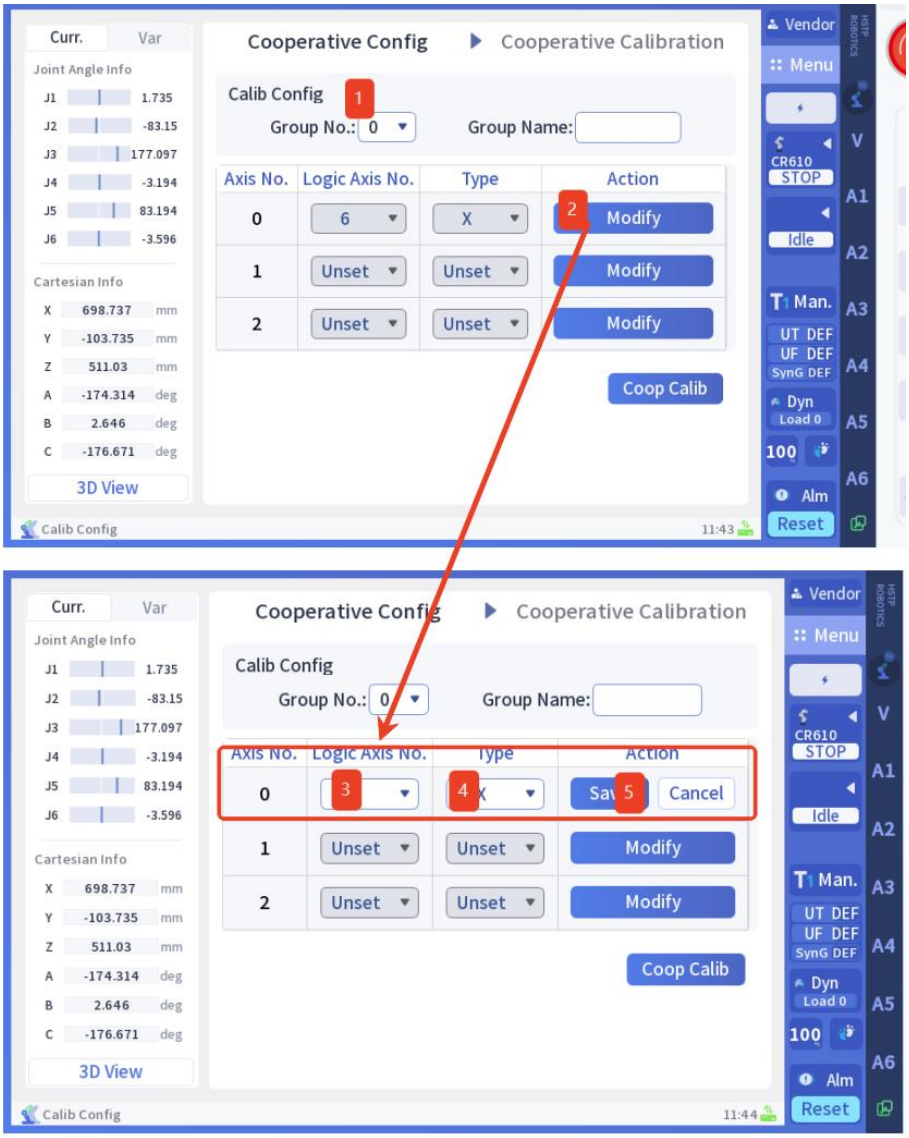
Introduction

A collaborative group refers to the integrated assembly formed by combining a robot's axis group with an external axis positioner. The configuration process involves assigning a collaborative group number and setting up the external axis for synchronized motion with the robot. In the PRG, the system initiates coordinated movement by invoking the COORD_NUM command.

Path

➢ [Menu] → [Start] → Axis Group Configuration: [Collaborative Group Configuration]

operating steps



1、Click the [Collaboration Group Number] dropdown to select the group to configure.

2、Click the [Modify] button in the target collaboration axis. Select the external axis index corresponding to the collaboration axis from the [Logical Axis Number] dropdown.

3、Choose the appropriate type for the collaboration axis from the [Type] dropdown. Finally, click [Save] to complete the configuration.

note：

- After configuring each axis, click the [Save] button. After completing the collaborative group configuration, calibrate the collaborative group promptly.

4.5 Calibration

4.5.1 Zero-point calibration and zero-point recovery

Introduction

A robotic arm can only execute proper multi-axis coordinated movements after zero-point calibration. Without this calibration, the robot will fail to follow the predetermined trajectory designed by the operator. Prior to calibration, all joint axes must be precisely aligned to their mechanical zero points (i.e., the zero-point reference lines of each joint).

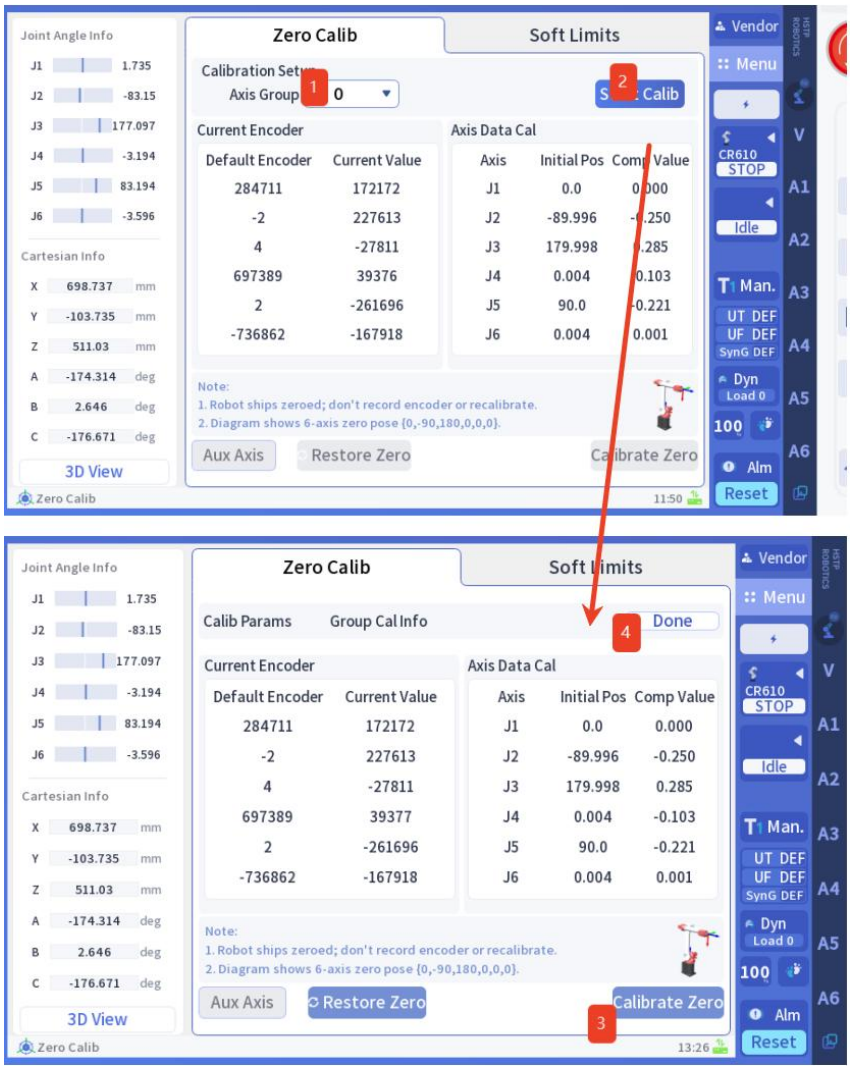
Label	Introduction
1	Tab switch. Click to switch to the system soft limit configuratio page.
2	Calibrate axis group dropdown. Click to select the axis group number to calibrate.
3	Start calibration. Tap to begin.
4	Default encoder value: The encoder value corresponding to th initial position saved during the last calibration for each axis.。
5	Current encoder value: The actual encoder value obtained by the system for the corresponding axis.
6	Initial position: the corresponding angle when the axis is calibrated. The teaching software will automatically fill in the appropriate value according to the model, and generally does

	not need to be modified by the user.
7	Compensation values. These are the zero-point attitude compensation values for each axis during factory calibration. Set at the factory and generally do not require user modification.
8	Additional axis button. Click to switch to additional axis calibration; the button text will change to "Internal axis" after clicking, and click again to switch back to internal axis calibration.
9	Restore the zero point button. Click to restore the zero point. Click after calibration starts.
10	Calibrate the zero point button. Click to calibrate the zero poin according to the set parameters.

Path

➢ [Menu] → [Start] → Calibrate: [Zero Point Calibration]

Zero-point calibration steps



1、Click the [Axis Group Name] dropdown to select the axis group for calibration. Click [Start Calibration]. Click [Calibrate Zero] to perform calibration and save data. Click [Complete] to finish and exit calibration mode.。

note：

- The robot has been calibrated for zero point and rod length compensation before delivery, so users generally do not need to calibrate it..
- Before calibrating, move each joint axis of the robot to the mechanical zero point (i.e., the zero mark position of each joint), then disable the motor;
- Click the [Compensation Amount] corresponding to the axis to modify

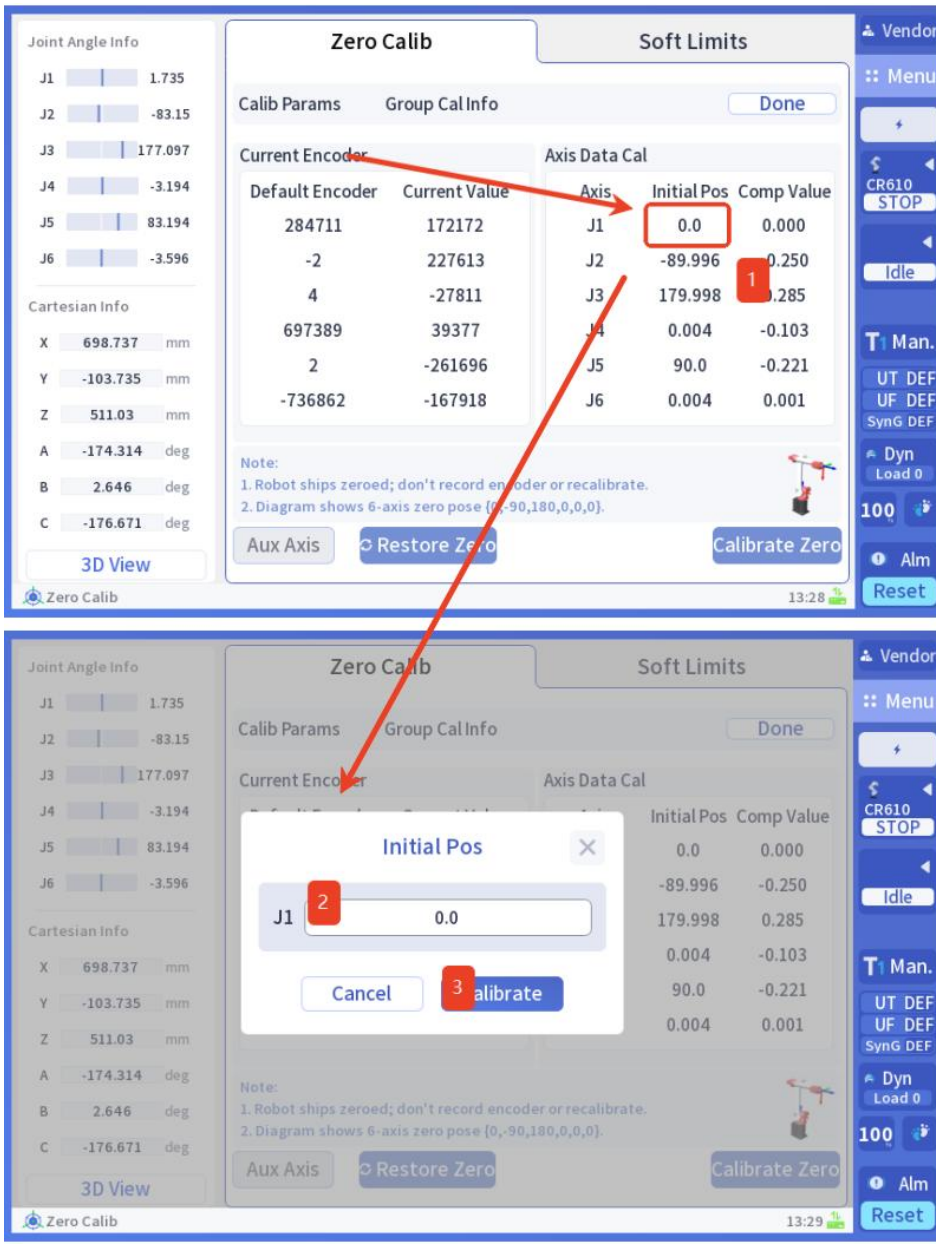
it. Users generally do not need to modify the compensation amount.。

warning：

- Incorrect calibration may cause the robot to move in space incorrectly, which may cause safety issues. Calibrate carefully.

Single-axis zero-point calibration

In some cases, due to space limitations, the joints of the robotic arm cannot be moved to the zero position, so the zero position calibration can only be performed on a single joint axis.



1、Click the "Initial Position" property value of the target calibration axis. Enter the joint's zero point position data in the pop-up input box. Click the [Calibrate] button to complete the single-axis zero position calibration.。

tips：

- Before performing single-axis calibration, click the [Start Calibration] button. Note that the [Calibrate Zero] button calibrates all axes of the robotic arm.。

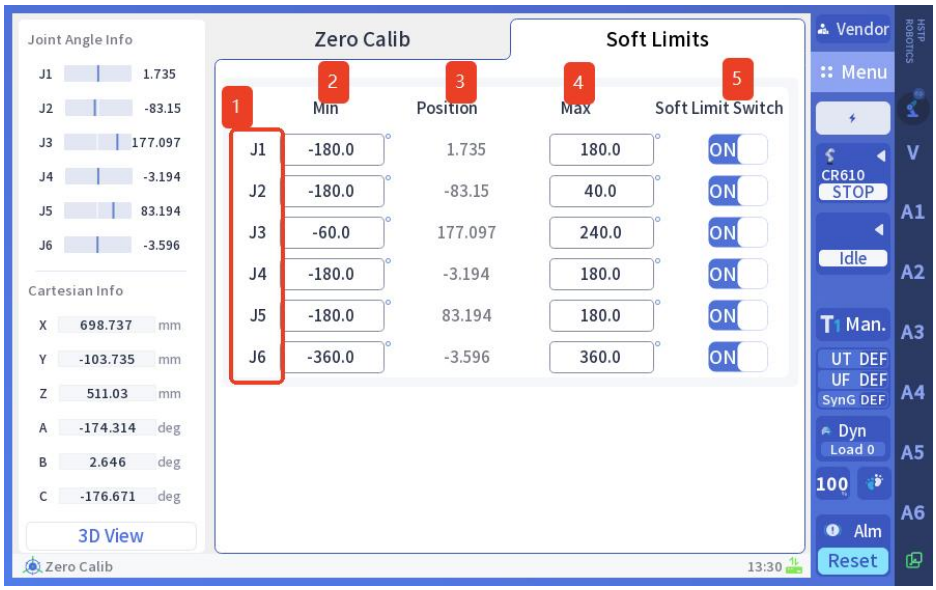
How to quickly restore zero

- If there is an error or mis-calibration: In fact, the system still saves the last calibration data. Click [Restore Zero] on the [Zero Point Calibration] page.
- If the system upgrade causes unexpected loss of zero-point data due to switching devices, obtain the factory-recorded zero-point data from the manufacturer and re-import it.
- Encoder power loss: Reset each joint to the reference zero point in the slot, then clear the encoder's multi-turn value.
- For collision detection and tooth skipping scenarios: First, calibrate the return slots of each joint to the zero point; then clear the encoder's multi-turn count, and finally import the factory-recorded zero-point data obtained from the manufacturer.

4.5.2 Joint limit

Introduction

By setting the system (joint) soft limit, the motion range of each axis of the robotic arm can be limited. The joint angle soft limit value is usually set before the factory, and the user does not need to set it again.



The configuration page for the system soft limit is presented in a table format. The table properties and corresponding label descriptions are as follows:

Label	Introduction
1	Axis number explanation. J represents the internal axis, E represents the external axis.
2	Minimum joint angle for the corresponding axis movement range.
3	The current joint angle value of the corresponding axis.
4	The maximum joint angle of the corresponding axis movement range.
5	Switch button. Click to enable or disable the soft limit of the corresponding axis. When the soft limit is disabled, the axis movement range will not be restricted by the system software.

Path

➢ [Menu] → [Start] → Calibration: [System Soft Limit]

operating steps

1、Modify the movement range [Minimum] and [Maximum] of the corresponding axis as needed. Adjust the [Soft Limit Switch] of the corresponding axis as needed. Click [Save] to complete the modification.

note：

- Click the [Cancel] button to discard unsaved changes.

Warning：

- Before setting the soft limit, contact the technical service personnel for professional guidance. Turning off the joint soft limit or setting the wrong soft limit may cause problems such as collision with the machine or cable entanglement in the mechanical arm.

4.6 Coordinate system calibration

4.6.1 User tool frame calibration

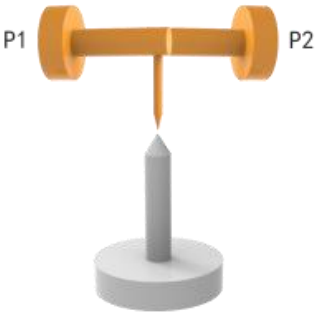
4.6.1.1 Calibration method

Introduction

- Tool coordinate system calibration is divided into four-axis robot calibration and six-axis robot calibration, where
- The four-axis robot tool coordinate system calibration supports 2-point and 4-point methods；
- The six-axis robot tool coordinate system calibration supports 4-point and 6-point methods。

SCARA 2-point method

The system moves the center point of the tool gripped by the SCARA robot's end effector from two different directions to a fixed reference point and records the current position of the robotic arm. It then calculates the TCP of the tool based on the recorded two-point information.

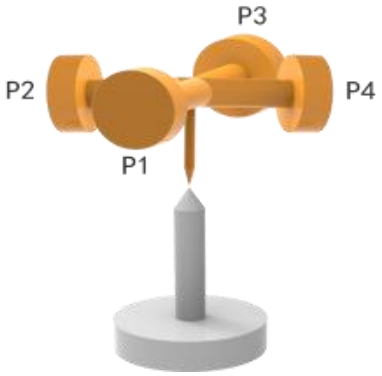


Scarab

The system moves the center point of the tool held by the SCARA robot's

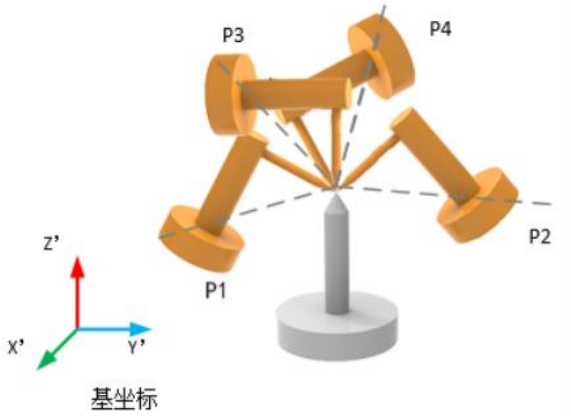
4-point method

end effector from four different directions to a fixed reference point and records the current position of the robotic arm. Based on the recorded four points, the system calculates the TCP (Tool Center Point) of the tool. The 4-point method calibrates the position, indicating the offset distance between the fixture coordinate system and the flange center point. Therefore, in the tool coordinate system UT[x], only the XYZ coordinates are typically numerical, while ABC coordinates are zero.



Six-axis robot 4-point method

Move the center point of the tool gripped by the robot end from four different directions to a fixed reference point and record the current position of the robotic arm. The system will calculate the TCP of the tool based on the recorded four point positions.
The 4-point method calibrates position by indicating the offset distance between the fixture coordinate system and the flange center point. Therefore, in the tool coordinate system UT[x], only XYZ values are typically specified, while ABC are set to 0.



Six-axis robot

The 6-point method for tool coordinate system calibration is similar to the

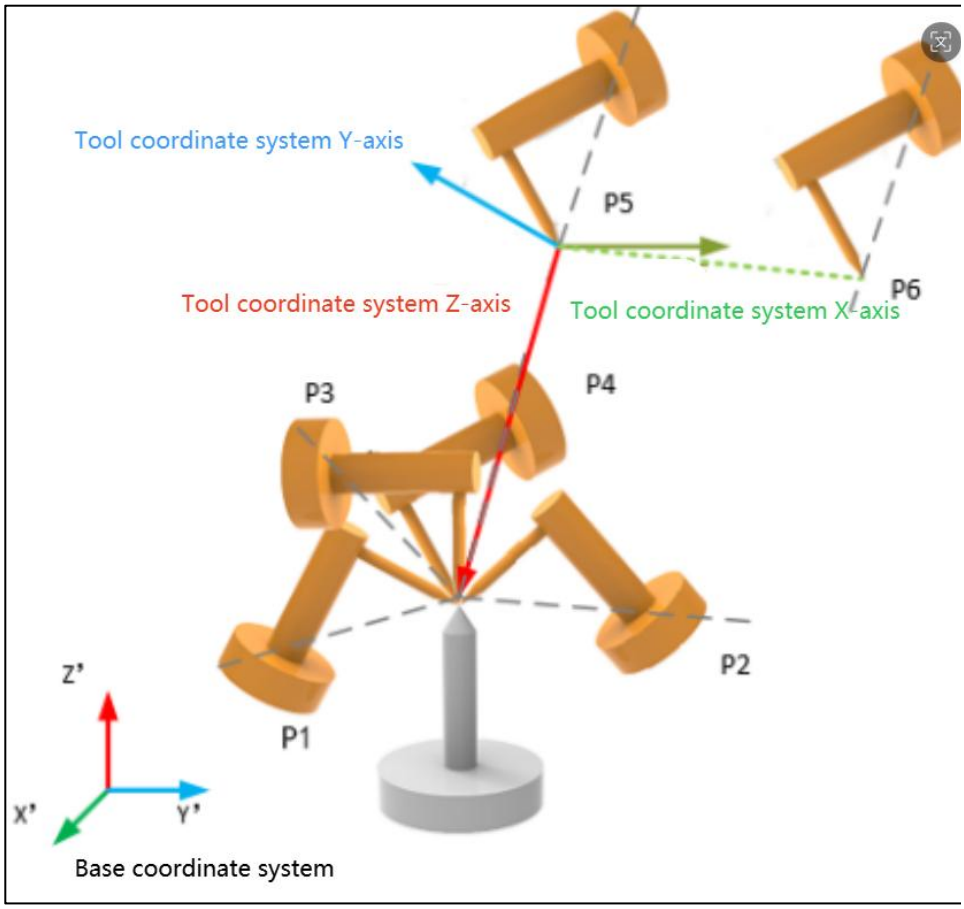
6-point method

4-point method, as it can calibrate the tool's orientation. The first four reference points are used for position calibration, while the fifth and sixth points record the tool's z-axis position and the point on the zx plane, respectively.

Pose calibration operation essentials：

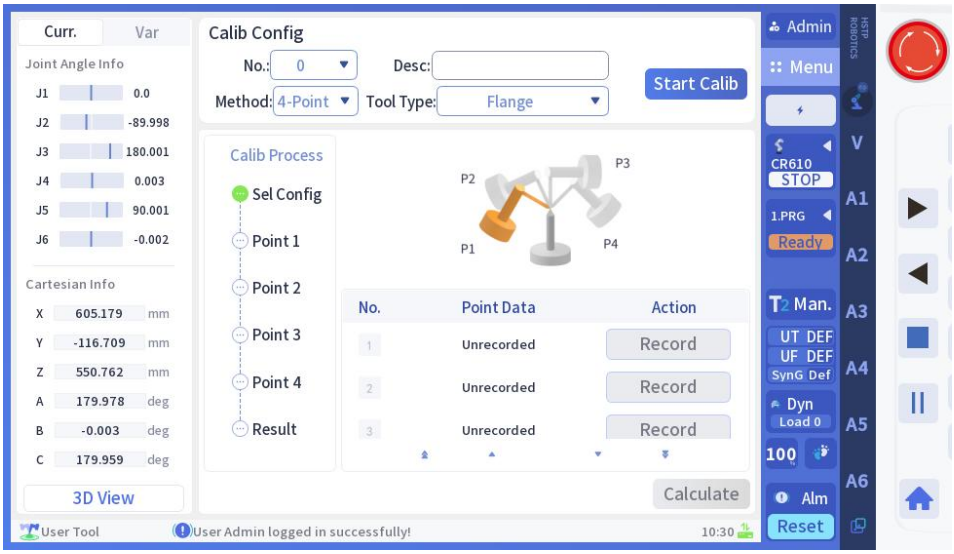
- Typically, the Z-axis direction in the tool coordinate system should align with the line connecting the center of the robotic arm's flange to the fixture's center (hereafter termed the fixture axis). This ensures Z-axis movement follows the fixture axis when using the tool coordinate system. Therefore, when calibrating the fifth point, the invisible fixture axis (indicated by the gray dashed line in the figure below) must be collinear with vector P5P4.
- In practical calibration, reference objects can be added to determine the position of the invisible fixture axis. Alternatively, the axis direction can be determined by properly selecting the orientation of the fourth point. For example, when calibrating the fourth point, align the fixture axis with the Z 'axis of the base coordinate system. When calibrating the fifth point, simply select a point on the Z' axis of the base coordinate system (i.e., press the Z-axis button during teaching point positioning)。
- To calibrate the sixth point, select a point on the ZX plane. The system calculates the Y-axis of the tool coordinate system using the cross product P5P4 × P5P6, and subsequently determines the X-axis.

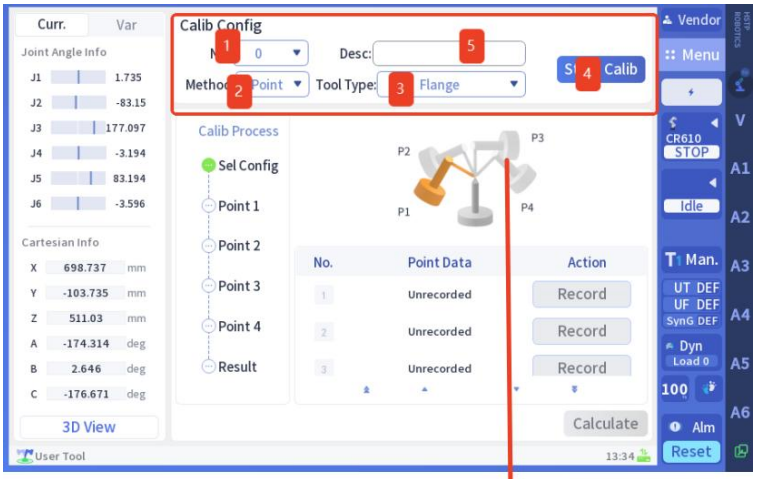
Note: When calibrating the 5th and 6th points, the posture of the robotic arm must be the same as that of the 4th point, that is, the posture remains unchanged.

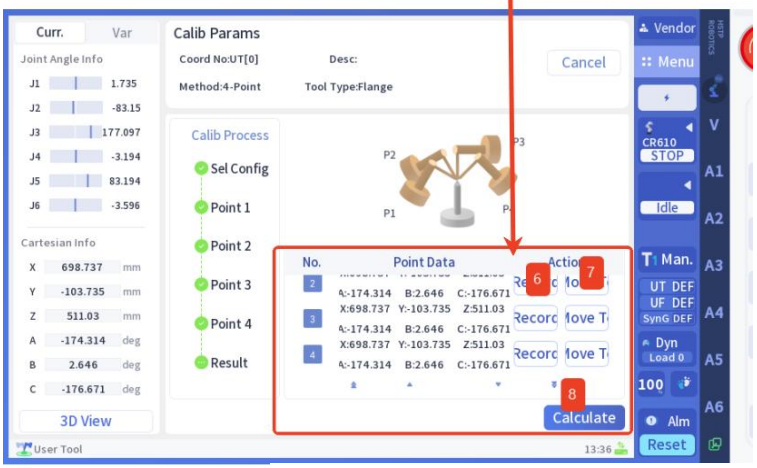


4.6.1.2 Operation UI

Introduction After understanding the differences in the calibration methods, users can perform specific calibration operations through the [User Tool Calibration] page, as shown in the figure below.







Label	Introduction
1	The tool number (0-15) corresponds to the UT number in the status bar and the UT register number in the variable list.

2	Calibration method selection: 4-axis robots offer [2-point] and [4-point] calibration, while 6-axis robots provide [4-point] and [6-point] calibration.
3	Change the tool type. The default is [Flange Tool]. You can also select [External Tool]。
4	Start calibration. After confirming the tool number, calibration method, tool type, and tool description, click [Start Calibration] to begin the calibration process.
5	Calibration number description. To distinguish the calibrated tool coordinate system when referencing, a description is usually added to the current calibrated tool during calibration.
6	Record the point and move the robot to the correct position. Click [Record].
7	Move to point. Move to specified record point.
8	[Calculate] means to execute the calibration button. This button can only be clicked after recording all points. After clicking [Calculate], wait a moment and you will see a message indicating successful calibration.
9	The calibration result is not saved. Clicking will not save the controller system file.
10	The calibration result save button saves the calibration data to the controller system file.

Path

- Menu → Run → Calibration: User Tool Calibration

operating steps

1、Select the calibration result storage number from the dropdown menu.

2、Choose the appropriate calibration method from the calibration method dropdown.

3、Click the [Start Calibration] button to initiate the calibration process.

4、According to the selected calibration method, sequentially move the robot to reference points as required. At each reference point, click the [Record] button corresponding to the selected sequence number on the interface to log the data.

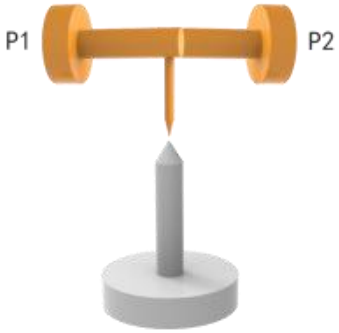
5、Click the [Calculate] button to compute the calibration results.

6、Save the calibration results by clicking the [Save] button in the "Calibration Results" pop-up window.

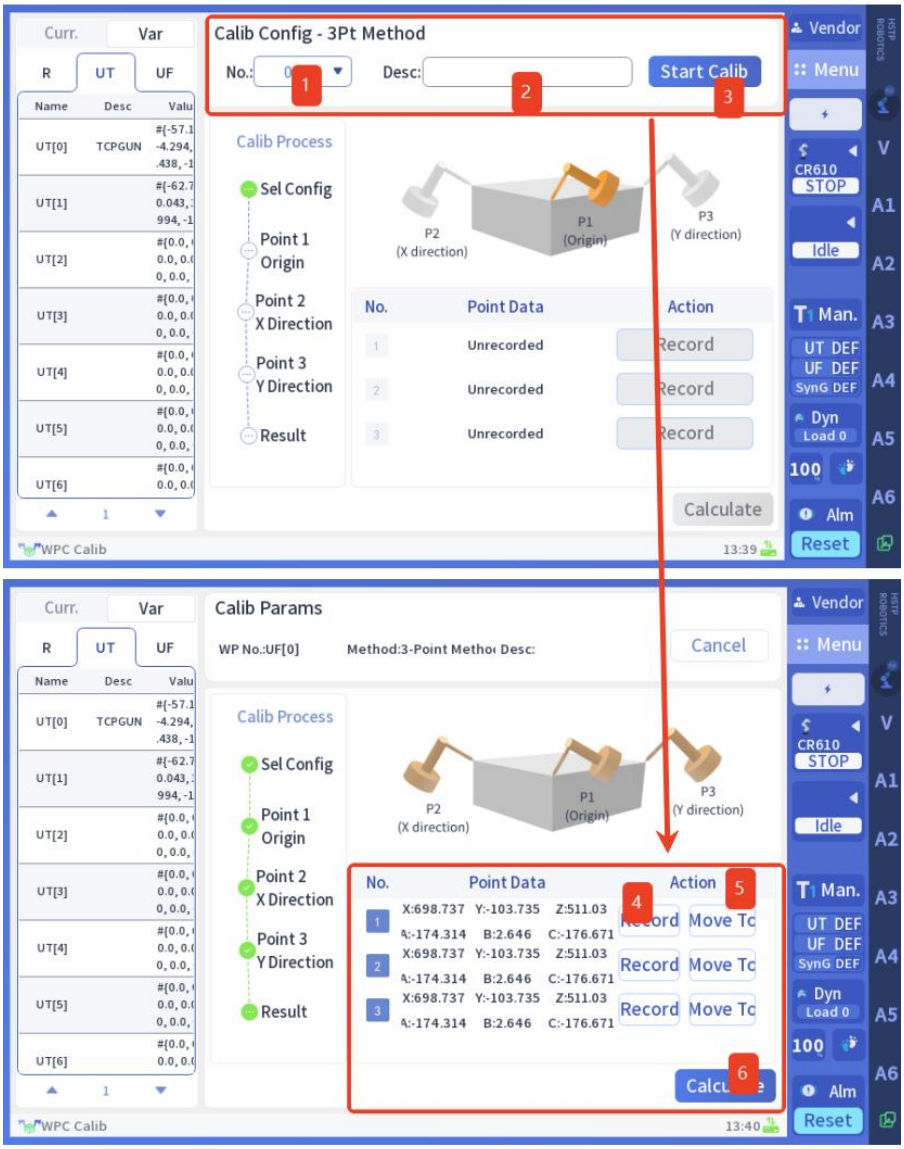
4.6.2 User workpiece frame calibration

Introduction

The workpiece coordinate system is calibrated using the three-point method. The robot is moved to the origin of the workpiece coordinate system, a point in the X-direction, and a point in the Y-direction, with recorded data. The system calculates the coordinate system based on the recorded point positions, achieving the calibration. The schematic diagram of the point positions is shown below:



The user workpiece calibration page in the teaching pendant allows workpiece calibration. The interface is shown in the following figure.



Label	Introduction
1	Workpiece number (0-15), corresponding to the status bar UF number and variable list UF register number.。
2	Calibration number description. To distinguish the calibrated workpiece coordinate system when referencing, you can edit the current calibration workpiece description.
3	To begin the calibration process, first confirm the workpiece number and write the tool specifications, then click [Start Calibration] to proceed.
4	Move the robot to the correct position and click [Record].
5	Move to the specified recorded point
6	Calculate, but Calibration results are not saved to system files.

7	The calibration result save button allows you to save the calibration data to the system file.
8	[Calculate] means to execute the calibration button. This buttor can only be clicked after recording all points. After clicking [Calculate], wait a moment and you will see a message indicating successful calibration.

Path

➢ Menu → Run → Calibrate: User Workpiece Calibration

operating steps

1、Select the user part number to calibrate and configure its description (i.e., the UF register description). Click the [Start Calibration] button. Move to the base coordinate origin and click [Record] to obtain the origin coordinates. Move to a point in the X direction of the calibration base coordinates and click [Record] to obtain the X coordinate. Move to a point in the Y direction of the calibration base coordinates and click [Record] to obtain the Y coordinate. Click the [Calculate] button to confirm the program's calculated calibration values. In the calibration result pop-up window, click [Save] to store the calibration results in the controller.。

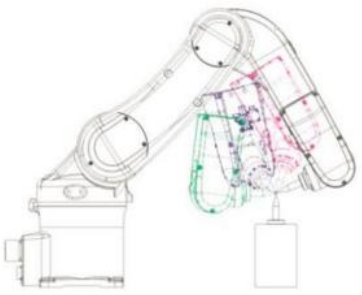
note：

- Before calibration, switch the current working coordinate system to the default value DEF, then move the robotic arm for calibration.

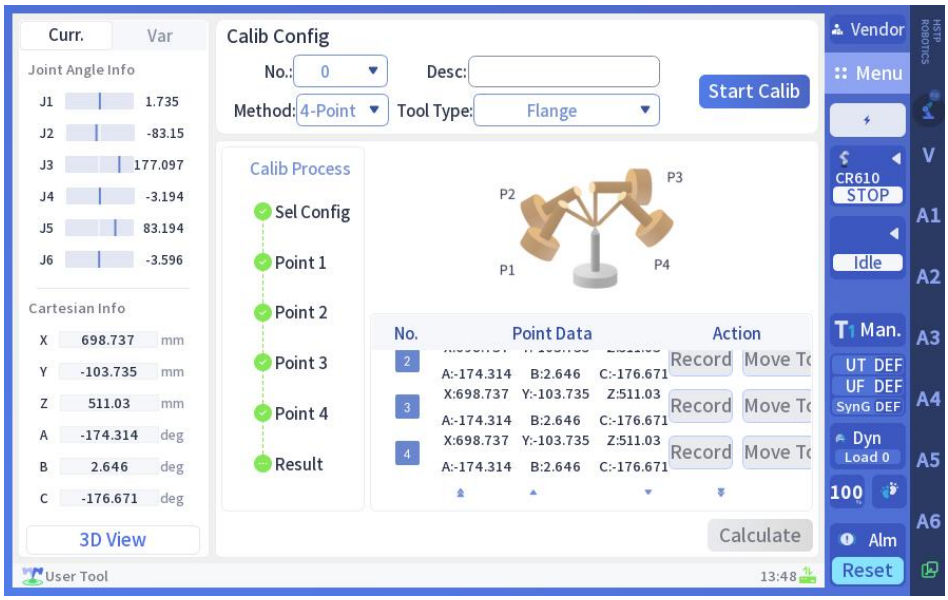
4.6.3 20-Point calibration

Introduction

In applications requiring high TCP precision for tools, 20-point calibration can be employed. This method establishes a high-precision tool coordinate system, though it necessitates simultaneous adjustment of existing joint zero points to maintain optimal TCP accuracy. Particularly effective in scenarios where collisions cause zero point displacement that compromises TCP precision, 20-point calibration enables simultaneous calibration of high-precision TCP coordinates while restoring original joint zero values.

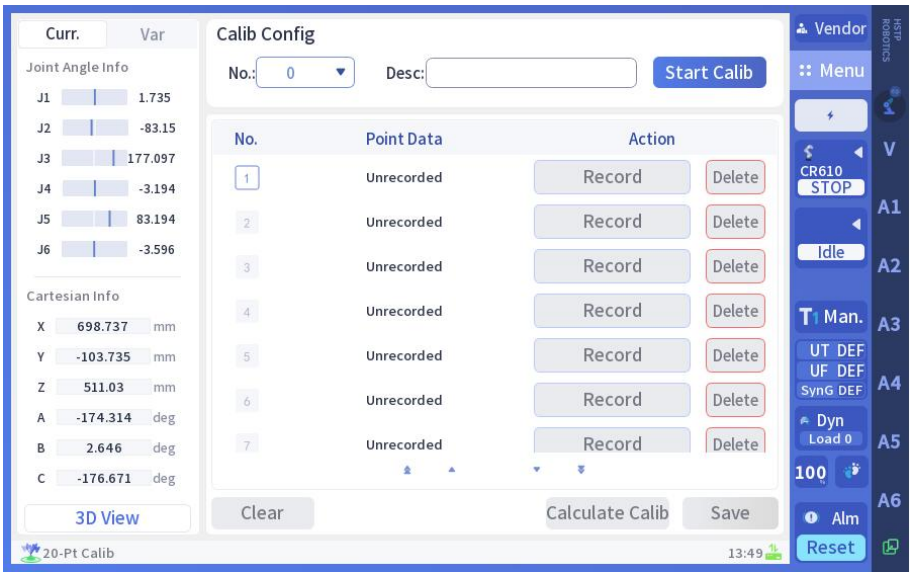


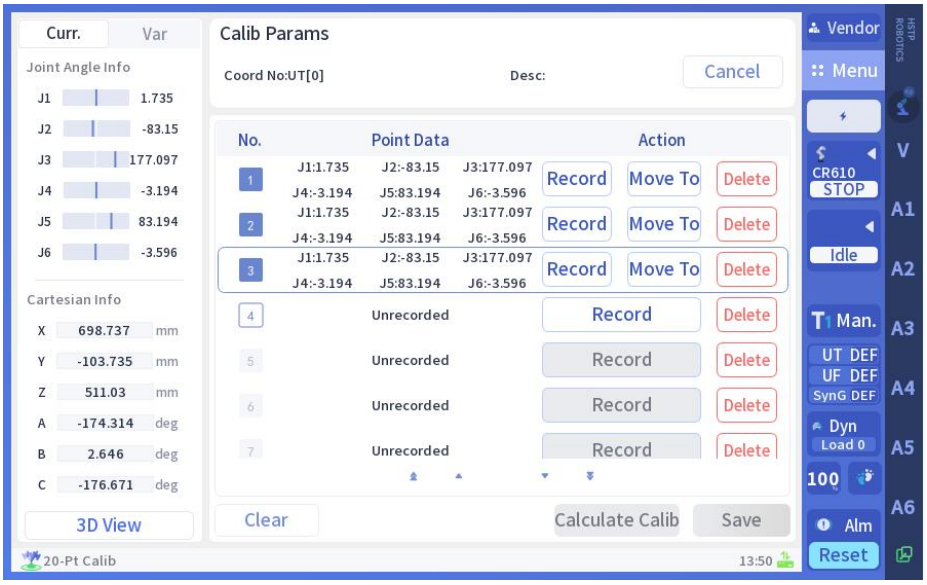
The 20-point calibration can be performed in the teaching device through the [20-point Calibration] page. The operation interface is shown in the following figure.



Path ➢ Menu → Run → Calibration: 20-point calibration

operating steps





1、Select the tool number to calibrate. Move the robot to approach the reference point in various poses (with maximum variation between each pose) to obtain point data. Once more than 10 points are collected, the [Calibrate] button becomes clickable. Users can decide whether to proceed with calibration based on the current point count. Click [Calibrate] to calculate the zero point offset and tool coordinate system values. Click [Save] to store the relevant information.

Tips：

- The calibration is based on 20 points, which is only an approximate number. Usually, only 10 points are needed to complete the calibration, but the more points are obtained, the more accurate the

calibration result is.

- The 20-point calibration will cause the zero point of each joint to shift, which may cause the actual position of the point in the program to change.
- The 20-point calibration can mark a tool coordinate system with relatively high accuracy. In order to ensure high accuracy, it will modify the zero point value of each axis of the system.

Warning：

- After calibration, the zero point will cause irreversible changes in the robot's zero point position. Unless you know the impact of 20 calibration, do not perform 20 calibration.
- Before calibration, it is recommended to back up data using the "Data Backup-System Parameters Backup" function.

4.6.4 Collaborative group calibration

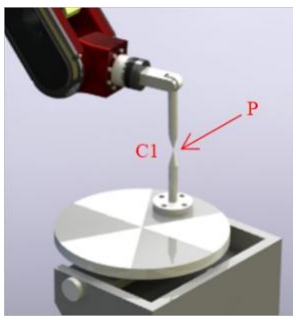
4.6.4.1 Calibration method

Introduction The system provides two methods for collaborative group calibration: the 3-point method and the 5-point method. The 3-point method is used for scenarios with 1 external axis, and the 5-point method is used for scenarios with 2 external axes.

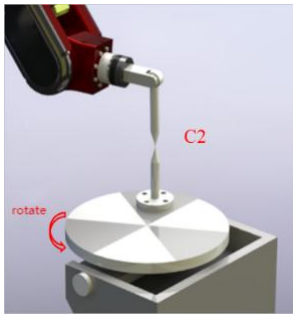
3-point method

1、Locate any point (P) on the turret of the positioner (as far away as possible from the turret's rotation)

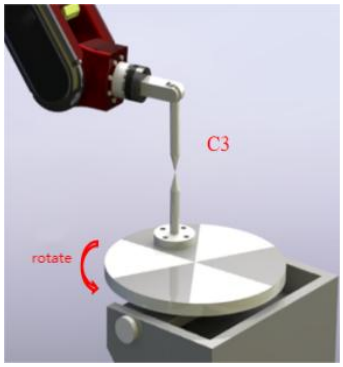
2、Merge the robot's control point with point P and record the first point C1.



3、Rotate the positioner shaft by any angle. The rotation is unlimited, but select an angle greater than 30 degrees. Select the positive direction for calibration. Merge the robot's controller point with the selected P point and record the second point C2.



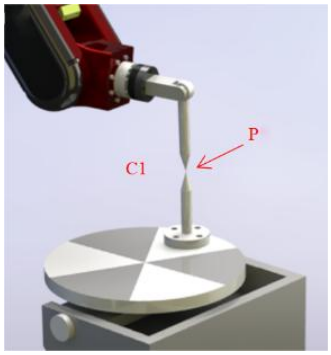
4、Rotate the positioner shaft in the direction specified in 2, merge the robot's control point with the rotated P point, and record the third point C3.



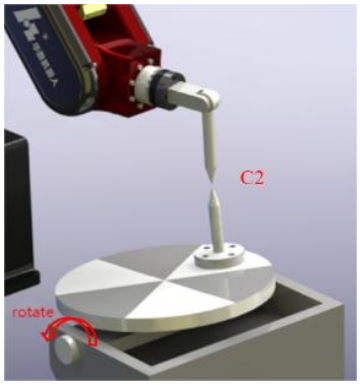
5-point method

1、Select any point (P) on the turntable (as far away as possible from the rotation center).

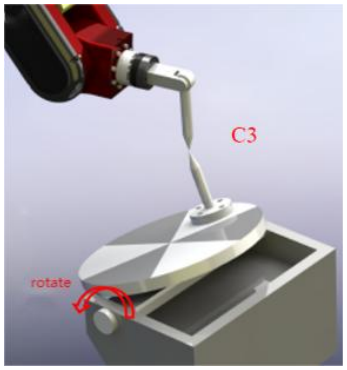
2、First align the first axis of the positioner with the horizontal ground plane, then merge the robot's control point with point P, and record point C1.。



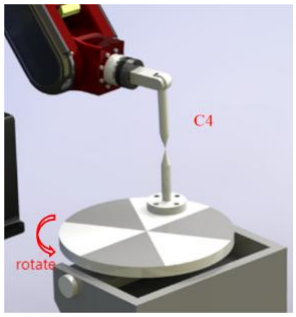
3、Rotate the first shaft (converted to a machine shaft) by approximately 30 degrees, merge the robot's control point with point P, and record point C2.



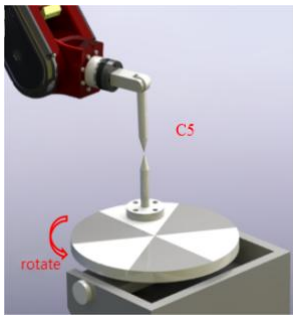
4、Rotate the first axis of the positioner shaft by approximately 30 degrees to merge the robot's control point with point P, then record point C3.。



5、Return to position C1 and rotate the second axis of the positioner approximately 30 degrees. Merge the robot's control point with point P, then record point C4.。



6、Rotate the second axis of the positioner shaft by approximately 30 degrees. Merge the robot's control point with point P, then record point C5.。



4.6.4.2 Operation UI

Introduction

When the robot axis group and external axis displacement unit perform collaborative actions, their mutual positional relationship must be pre-determined. This positional determination process is referred to as robot-axis group-external-axis group calibration.

The collaborative calibration operation can be performed through the [Collaborative Calibration] function in the teaching pendant, as shown in the following
interface:

Label	Introduction
1	Collaboration group number (0-3) corresponds to the group number in the [Collaboration Group] and [Collaboration Group Configuration] sections of the status bar.
2	Calibration method: Select [3-point method] or [5-point method].
3	Create a group name and write a memorable comment for the calibrated group. This is optional.
4	Start calibration. Click [Start Calibration] to begin the calibration process.
5	Cancel calibration. Click to exit the current calibration.
6	[Record Location] records the robot's current position

7	[Move to Point] Move the robotic arm to the recorded point.
8	[Calculate] means to execute the calibration button. This button can only be clicked after recording all points. After clicking [Calculate], wait a moment and you will see a message indicating successful calibration.
9	Display calibration results.
10	Do not save calibration results。
11	The calibration result save button allows you to save the calibration data to the system file.

operating steps

1、Click the [Collaborative Group Number] dropdown to select the collaborative group number (this cannot be set arbitrarily; you must choose the newly configured collaborative group number). Click the [Collaborative Group Name] input box and enter a descriptive note for the group number. Select the [Calibration Method] dropdown option and choose the 5-point method (use 5-point calibration for two external axes in synchronization and 3-point calibration for one external axis). Click the [Start Calibration] button to sequentially obtain coordinate data after moving the robotic arm and external axis (calibration method details below). After recording, click the [Calculate] button to display the calibration results. Click the [Save] button to store the results in the system file, completing the collaborative group calibration.

- To perform collaborative calibration, the following conditions must be met: The external axis has been zero-point calibrated. The external axis limit has been set. The tool calibration has been performed. The calibration must be performed in the robot's forward direction.

4.7 Basic move

4.7.1 Adjust posture

Introduction

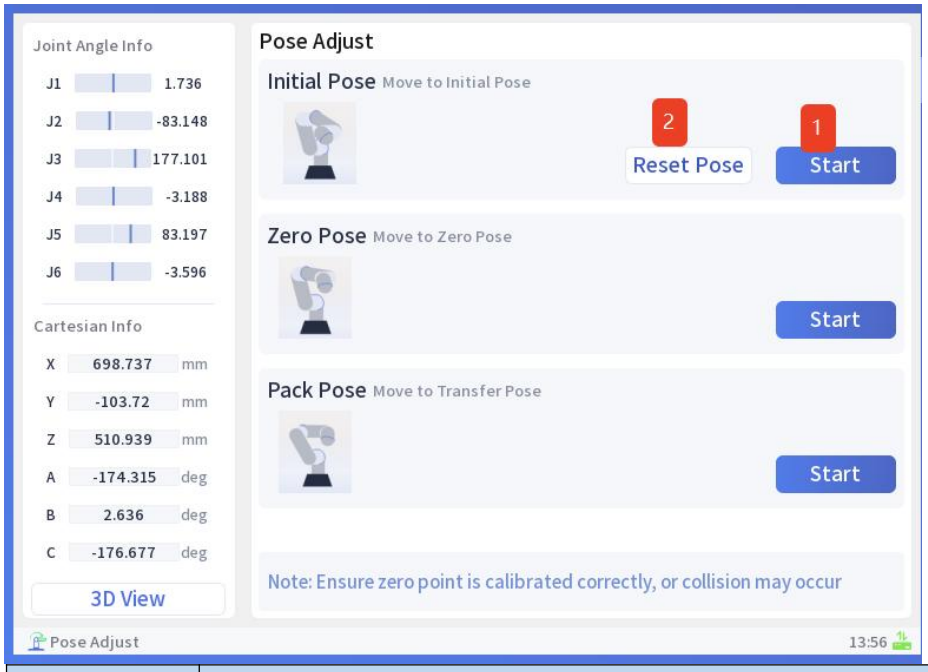
The system has preset several commonly used postures for the robotic arm. Users can quickly adjust the arm's posture through the "Posture Adjustment" function when needed. The system includes three preset postures:

Initial Posture: Customizable by users, defaulting to zero-point posture.

Zero-Point Posture: The robotic arm's zero-point posture, which varies

across different robotic arms.

Packaging Posture: A posture designed for convenient packaging and transportation, with different robotic arms having distinct packaging postures. The posture adjustment interface is as follows:



Label	Introduction
1	After clicking the [Start Exercise] button, the robotic arm will move to the corresponding posture.
2	Click the【Reset Initial Pose】 button to reset the pose.

Path

➢ [Menu] → [Start] → Basic Movement: [Pose Adjustment]

operating steps

1、Enable the robot;

2、Click the [Start Movement] button for the corresponding posture to display the movement progress window;

3、The movement is complete and the robot has reached the preset position.

Warning：

- Keep an eye on the environment around the robotic arm during movement to avoid collisions

Reset Initial Pose

Click the [Reset Initial Pose] button in the Initial Pose section to display the reset interface, as shown below:

Follow these steps: A

1、djust the angles of each axis as needed;

2、Click [Save] to complete the changes.

Note:

- The system uses the JR[999] register to store the initial posture. Avoid using this register.
- Click the [Restore Default Position] button to reset the initial posture to the default zero position.
- Click the [Get Current Position] button to retrieve the robot's current posture and automatically fill in the data.
- After clicking both [Restore Default Position] and [Get Current Position] buttons, you must click [Save] to confirm the changes in the system.

4.8 System function

4.8.1 Run configuration

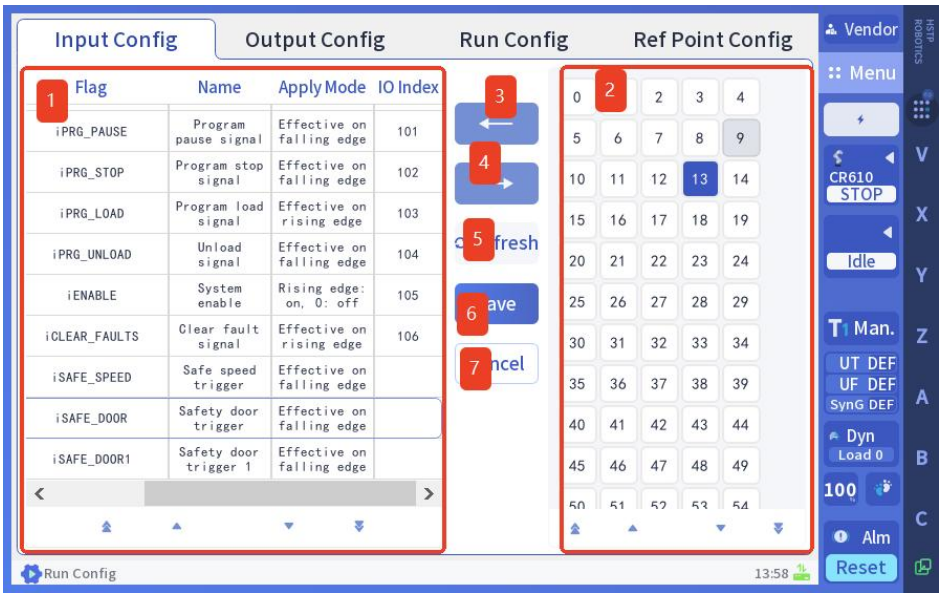
4.8.1.1 Input signal configuration

Introduction

In external mode, the robotic arm can only be controlled by external signals. To use external control, you must first perform the Input Configuration operation before sending the relevant signals to control the robotic arm to perform the corresponding actions.

"Input signal configuration" maps and binds system signals with IO inputs. Once the mapping is established, the robotic arm can be controlled through IO signals.

The configuration input page is shown below:



Label	Statement
1	The system input signal list displays all input signals, including their flag bits, names, activation methods, and the current mapped input IO indices。
2	The system displays an IO index table showing all input IO indices in the current system. Clicking an index number selects the corresponding IO. Indices with dark blue shading indicate the currently selected IO, while gray-shaded indices represent unavailable IOs, indicating that mapping has been established between the corresponding IO and related signals, and no duplicate mapping is allowed.
3	[Create mapping] button. Click to map the selected signal and IO index.
4	[Unmap] button. Click to unmap the selected signal.
5	[Refresh] Button. Click to refresh all signal mapping information.
6	[Save] Button. This appears only after modifying the mapping information. Click to save the changes.
7	[Undo] button. Appears only after modifying mapping information. Click to undo all unsaved changes.

Path

➢ [Menu] → [Function Configuration] → System Functions: [Run Configuration] → [Input Configuration] tab。

operating steps

1、Select the signal to configure its IO mapping in the left signal list.

2、In the right IO index table, select the target IO index for mapping.

Grayed-out options indicate that the corresponding mapping has already been established.

3、Click the [←] button to create a mapping;

4、 click the [Save] button to complete the mapping.

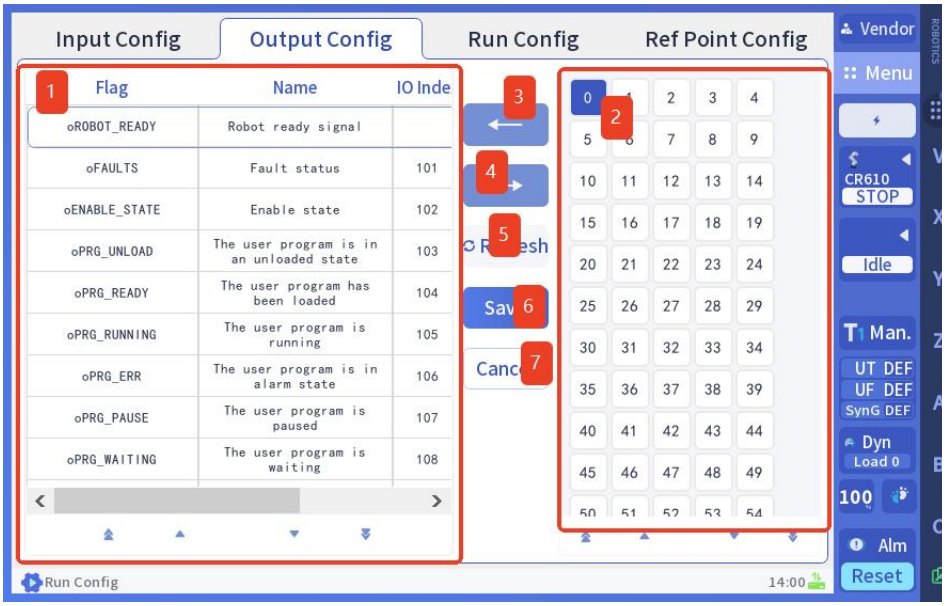
Note：

- Repeat steps 1, 2, and 3 as needed to map multiple signals, then click [Save].
- Click [→] to remove the selected signal's established IO index.

4.8.1.2 Output signal configuration

Introduction

The system can send partial status of the robotic arm to the external system via output IO. By establishing a mapping between the robot's status flags and corresponding output IO, only after this mapping is set up can the robot's status flags be transmitted through their designated output channels. The output configuration interface is shown in the figure below：



Label	Statement
1	The system output signal list displays all output signals, including their flag bits, names, and the current mapped output IO index.

2	The system output IO index table displays all output IO index numbers in the current system. Click the corresponding index number to select it. The index number with dark blue shading is currently selected; the index number with gray shading is not selectable, indicating that the corresponding IO and related signals have been mappe and cannot be mapped again.
3	[Create Mapping Button] Click to create a mapping for the selected signal and IO index.
4	[Unmap] Click to remove the established IO mapping for the selected signal.
5	[Refresh button]. Click to refresh all signal mapping information.
6	[Save button]. Appears only after modifying the mapping information. Click to save the changes.
7	[Undo button]. Appears only after modifying mapping information Click to undo all unsaved changes.

Path

➢ [Menu] → [Function Configuration] → System Functions: [Run Configuration] → [Output Configuration] tab。

operating steps

1、Select the signal to configure its IO mapping in the left signal list.

2、In the right IO index table, select the target IO index for mapping. Grayed-out options indicate that the corresponding mapping has already been established.

3、Click the [] button to create a mapping;

4、 click the [Save] button to complete the mapping.

Note：

- Repeat steps 1, 2, and 3 as needed to map multiple signals, then click [Save].
- Click [] to remove the selected signal's established IO index.

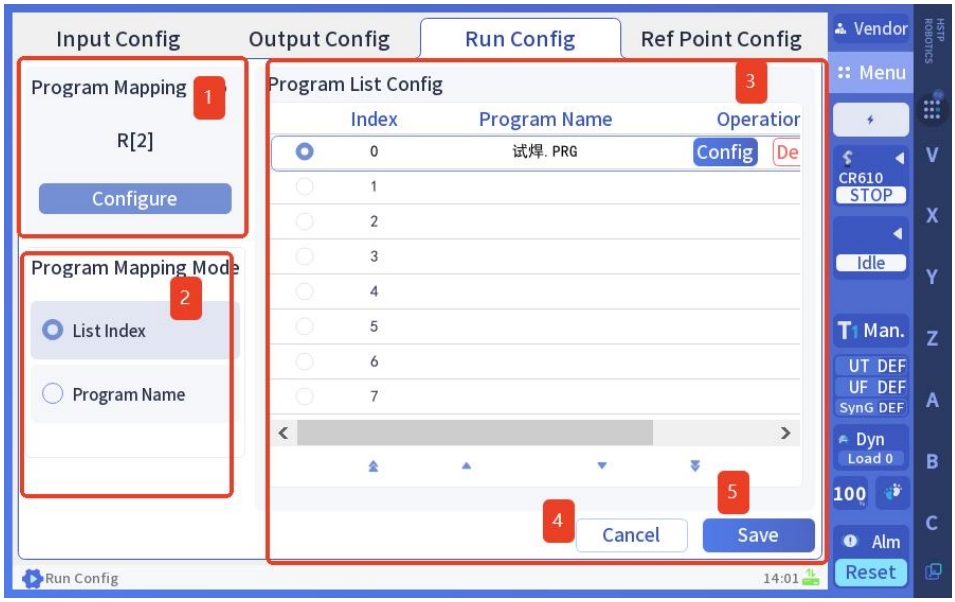
4.8.1.3 Program configuration

Introduction

In external mode, the robotic arm can only be controlled by external signals. When receiving an input signal from the loading program, the system indexes and loads the program using the specified R register values. The R register values and specific programs are mapped through two indexing methods.:

- List number mapping: The system contains a user-configurable program list. The R register value serves as the list index to locate the corresponding program.
- Program name mapping: The R register value is converted into a program name, which is then used to locate the corresponding program.

The program configuration interface is shown below:



Label	Introduction
1	Program mapping information configuration area. Displays and configures the mapping information source register.
2	Select the program mapping method.

3	Mapping method configuration area. Displays different views based on the selected program mapping method. The diagram shows the configuration view for the "List Number" mapping method.
4	[Cancel button]. Cancel current unsaved changes
5	[Save Button]. Save the current changes.

Path

➢ [Menu] → [Function Configuration] → System Functions: [Run Configuration] → [Program Run Configuration] tab。

Configure index register

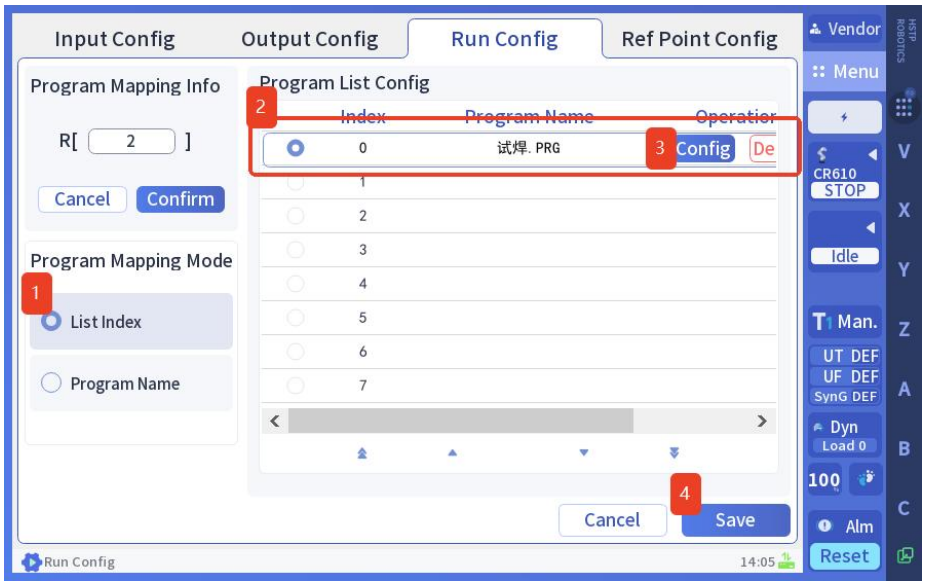
1、Click the [Configure] button in the program mapping configuration area.

2、Enter the target index number in the input box.

3、Click the [Confirm] button.

4、 Click the [Save] button to complete the configuration.

List numbering mapping method



1、Select [List Number] in the program mapping method.

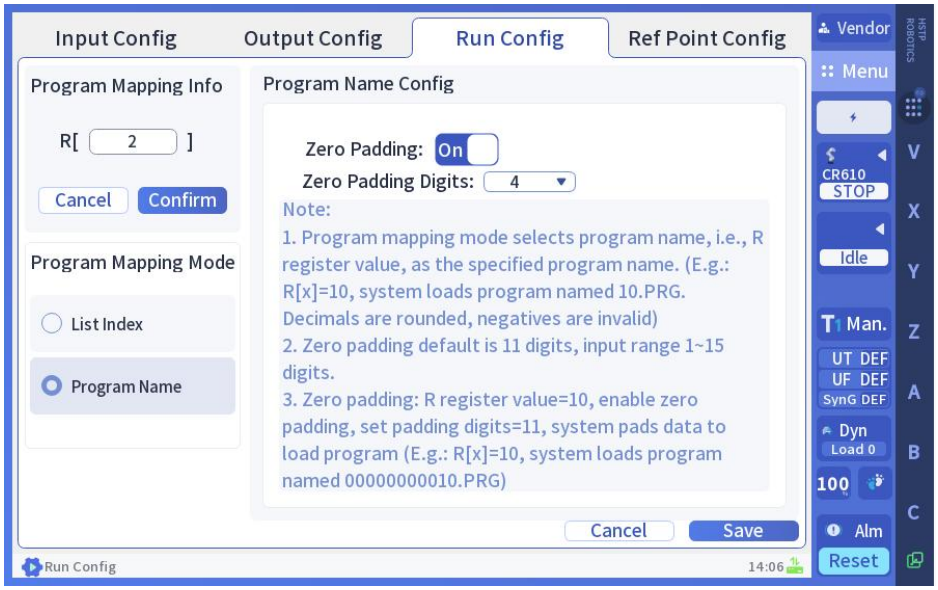
2、In the left list, select the number to configure.

3、Click [Configure] in the list item. In the pop-up program file selection window, select the target program.

4、Click [Save] to complete the configuration.

Program Name Mapping Method

The program name mapping method and operation view are shown in the following figure:



Label	Introduction
High-level zero-switching	A switch to pad the value in the source R register with zeros when converting it to the program name.

Top zero padding	The drop-down menu allows you to select the zeroing digit after enabling the high-level zeroing switch.

note：

- To make any changes in the program configuration interface, click the [Save] button.
- You can configure multiple items at once and save them by clicking [Save].

example：

Assuming that the high zero filling switch is enabled and the zero filling number is configured to 8, when R[x]=10, the system loads the program name 00000010.PRG, that is, 6 zeros are added in front of 10, so that the length of the converted file name reaches 8.

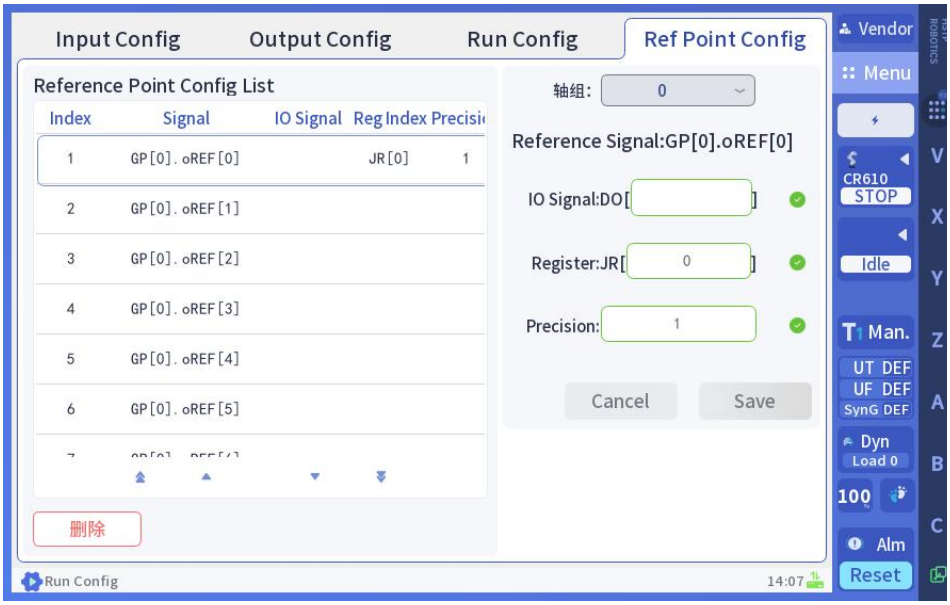
4.8.1.4 Reference point configuration

Introduction

The reference point signal output function enables the system to generate an oREF signal when the robotic arm reaches a specified joint position. To activate this feature, users must first configure a reference point and link it to the corresponding output I/O port.

When configuring the operation, specify the JR register and joint precision range. If the robot is at the specified JR register position (e.g., JR[0]={0, -90,180,0, 90,0}) with the error between the current position and the set value within the precision range, the system outputs the position signal oREF.

The reference point configuration page is shown below:



Path

➢ [Menu] → [Function Configuration] → System Functions: [Run Configuration] → [Reference Point Configuration] tab。

operating steps

In the left [Reference Point Configuration List], select the reference point to configure.

In the right configuration area, enter the corresponding DO index, JR register index, and precision.

Click [Save] to complete the configuration.

Note：

- DO[100~129] and DO[200~229] are reserved for internal system use; avoid using them.
- Click the [Clear] button to reset all reference point configurations.
- The reference points will only take effect when the IO index, register index, and precision are configured.

4.8.2 Region settings

Introduction

The spatial restriction function is a safety mechanism that limits the range of motion of a robotic arm. Through specific configuration settings, it prevents the arm from entering or exceeding designated spatial boundaries. This

function requires defining the parameters and types of these areas. Once configured and the area monitoring feature activated, the system will automatically inspect and respond to the specified parameters.

area type

The region has the following four types:

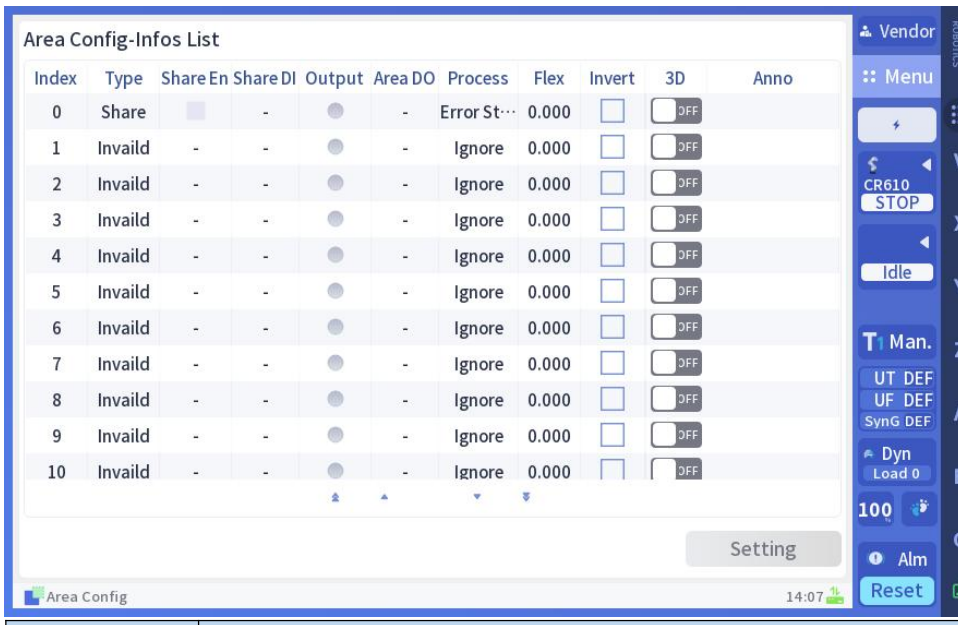
- Invalid area: The current setting is invalid. Only signal detection is performed in this area, and no error reporting or shutdown is applied。
- Interference zone: the area where the robot cannot enter or stay. If the robot is in the interference zone, switch the corresponding area type to invalid zone before the robot can move.
- Safe zone: refers to the area where the robot is not allowed to leave. If the robot is outside the safe zone, switch the corresponding area type to invalid zone before the robot can move.
- Shared Zone: A zone that can control whether to exhibit interference zone characteristics, with selectable processing modes for entry. Each shared zone is equipped with an external input signal called Shared Zone Enable DI, which controls access. When the DI value is FALSE/OFF, the shared zone remains disabled, behaving like an invalid zone. When TRUE/ON, the shared zone is enabled, and the robotic arm responds according to the configured processing mode upon entry.

Path

➢ [Menu] → [Function Configuration] → System Functions: [Area Configuration]。

Operation notice

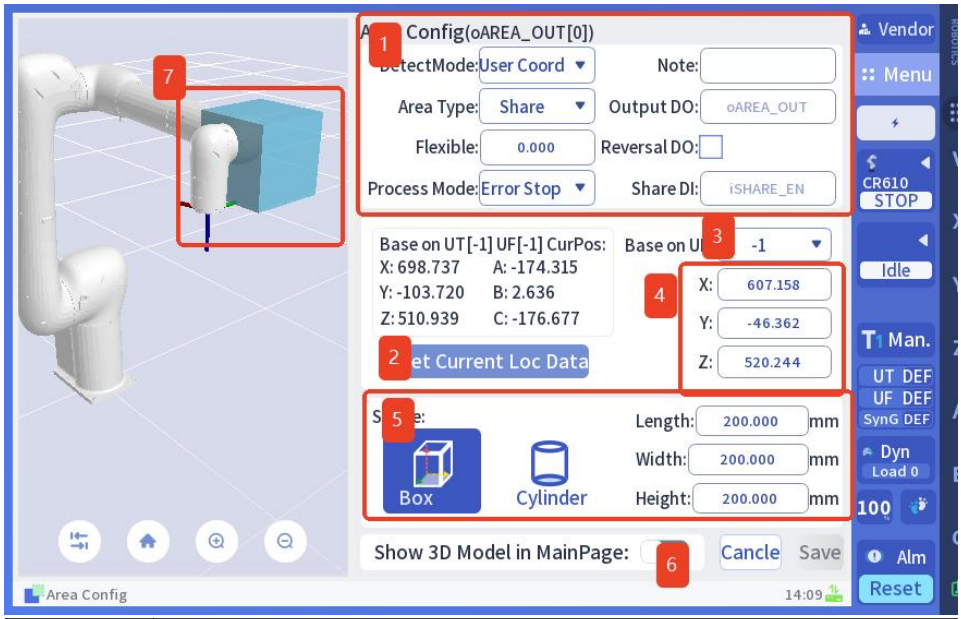
The regional configuration consists of a regional information list page and a detailed regional configuration page. The regional information list page is shown below:



Tab header	Statement
index	Region index number
type	The type currently configured in the corresponding region.
Enable sharing	When the corresponding region is a shared area, the current value of the corresponding shared DI。 ：Indicates that the shared DI's current value is FALSE/OFF, meaning the shared area is disabled. ：Indicates that the current value of the shared DI is TRUE/ON, enabling the shared area.。
DI index number	Number of DI index for the corresponding region (valid only for shared area types)。
value	The current value of the corresponding DO。
region DO	The DO configured for the corresponding area will set its value to TRUE/FALSE when the robotic arm is in that area.。
processing mode	The processing mode of the system when the robotic arm is in the corresponding area. There are four processing modes: Ignore: no processing is performed; Error stop: the system throws an error message and stops; Warning stop: the system throws a warning message and stops.
Offset value	The offset value attribute defines the permissible error for determining whether a mechanism is within or outside a

	specified area. This offset value can also be interpreted as the wall thickness of the area: the safety zone corresponds to increasing the wall thickness inward, while the interference/sharing zone corresponds to increasing it outward. In practical applications, the expansion value should be greater than the theoretical value to enhance safety.。
Signal inversion	Invert the DO value of the corresponding area, setting it to TRUE/ON when the robotic arm is not in the area. Click the signal inversion checkbox to adjust directly.
3D display	Indicates whether the corresponding area is visualized in the 3D view on the home page. Click the 3D display switch to set it directly.
remark	Notes for the corresponding region

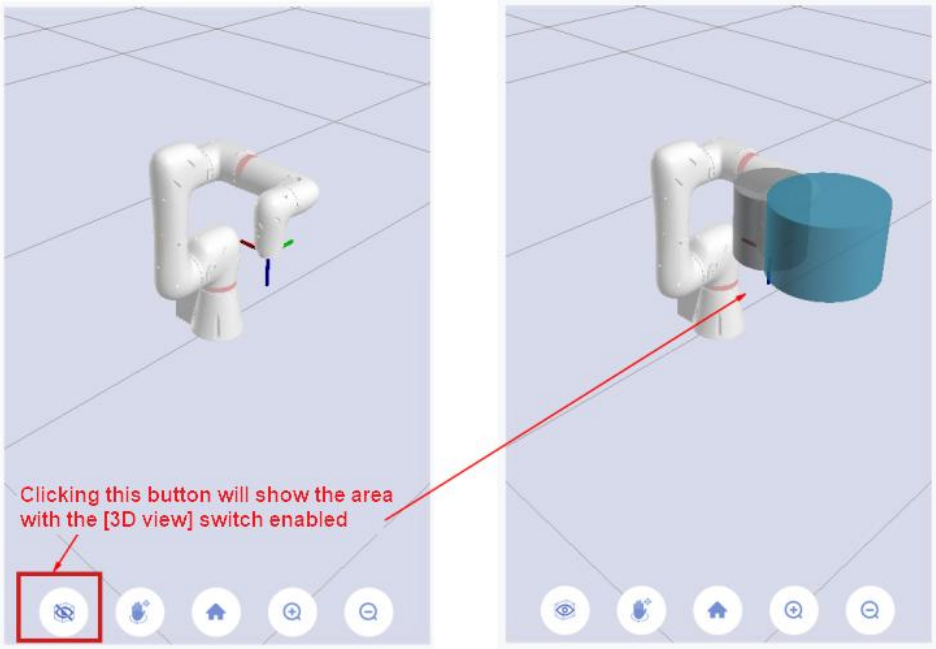
After selecting the required area item in the area information list, users can click the [Configure] button in the lower-right corner to access the detailed configuration page for the specified area properties. The detailed configuration page is shown in the figure below:



Label	Statement
1	Configure the region type, output signal (region DO), offset value, signal inversion, processing mode, and notes in LABEL1
2	Click to get the current Cartesian coordinates and enter them into the input box of Label 4。
3	Select the reference coordinate system for the origin of the

	corresponding area in the drop-down list. This sets the reference coordinate system for the origin coordinates specified in Label 4. If the workpiece number is-1 or the corresponding workpiece number is not calibrated, the reference coordinate system defaults to the base coordinate system.。
4	Set the region's origin。
5	Select the shape of the area and configure its size。
6	Home 3D View Visualization Switch。
7	3D visualization of the current area's location, shape, and size。

When the 3D view switch for the relevant area is enabled, the 3D view on the home page will display the [] button. Clicking this button will show the area with the [3D view] switch enabled, as shown in the figure below.

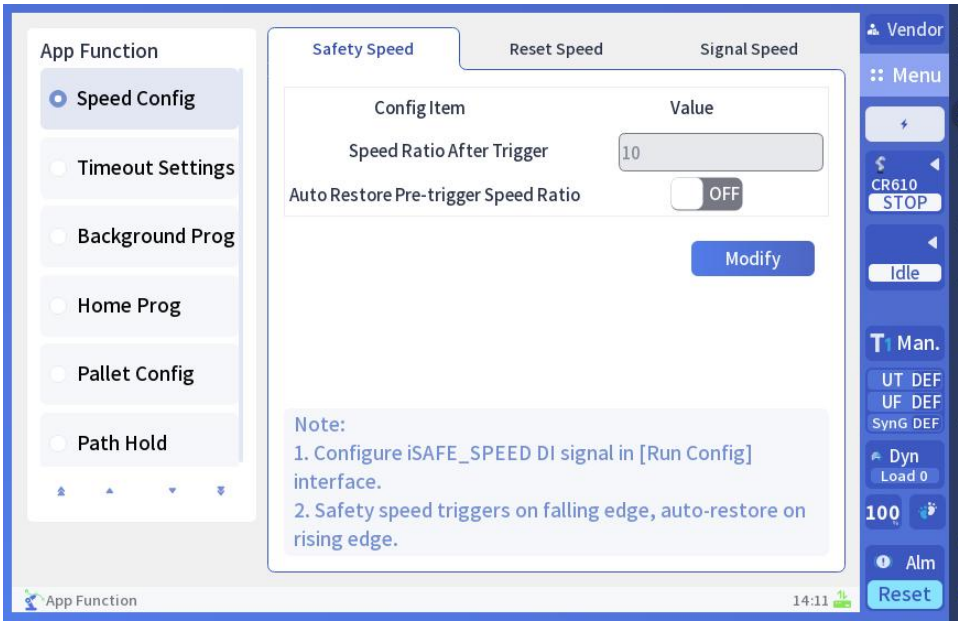


Note：

- After completing the configuration on the regional details page, click the [Save] button in the lower right corner.。

4.8.3 Application function

Introduction

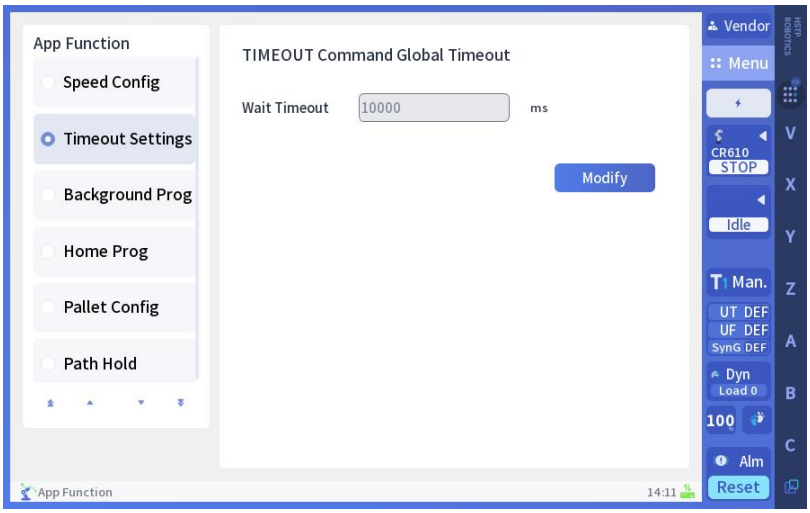


Path

➢ [Menu] → [Function Configuration] → System Functions: [Application Functions]。

4.8.3.1 Timeout settings

The timeout setting function is the global time setting for the TIMEOUT timeout command.



Path

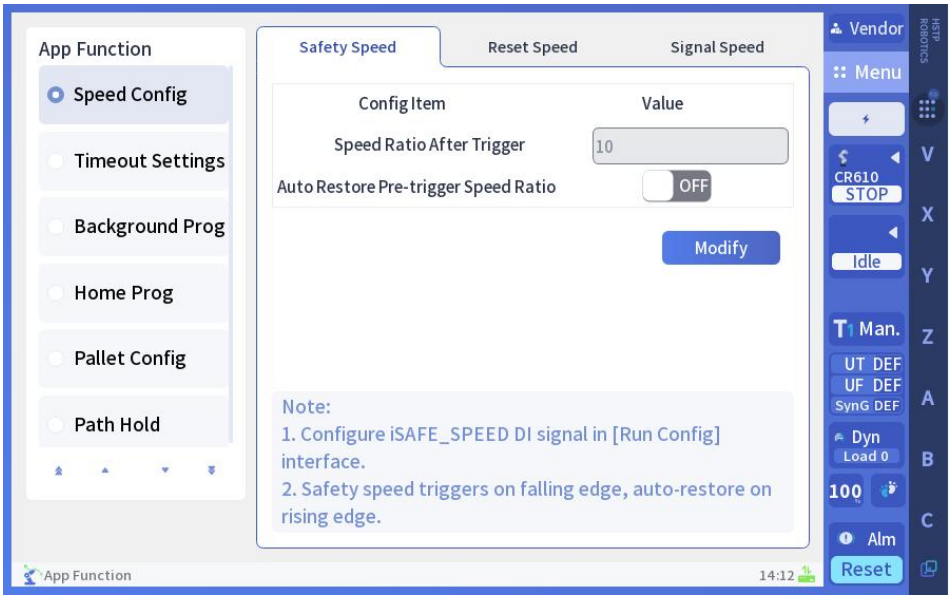
➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Timeout Settings] tab.

4.8.3.2 Speed control

4.8.3.2.1 Safety speed

Introduction

In practical applications, robots operate at high speed within safety barriers. When opening the barrier door to enter the enclosed area, the robotic arm's speed must be reduced to prevent accidents. Upon closing the door, the robot can either automatically resume its original speed for production or manually adjust the speed ratio via the teaching pendant's multiplier button instead of automatic recovery.



- The [Triggered Speed Multiplier] column shows the speed value set for the robotic arm after the corresponding IO signal takes effect. This value ranges from [1,100]. If the robot's movement rate exceeds the set safety rate, it automatically switches to the safety speed.。
- [Auto Restore Original Speed Ratio Switch] When enabled, the robot resumes its original speed ratio after resetting the corresponding IO signal. When disabled, the robot maintains the speed ratio after receiving the trigger signal.。

Path

➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Speed Configuration] tab → [Safe Speed] tab

Case study of restoring the original speed multiplier function:

Set the speed multiplier to 10% after triggering, enable automatic recovery, and configure the safety signal as DI[200] with V as the current speed.

Case 1：

```
DI[200]=ON,V2=100%
DI[200]=OFF     ‘ Safety speed triggers take effect   V=V1=10%
DI[200]=ON      ‘ V=V2=100% (Speed recovery)
```

Case 2：

```
DI[200]=ON,     ‘ V2=100%
DI[200]=OFF     ‘ V=V1=10%
‘ change robot vord by manual , V3=50%
DI[200]=ON      ‘ Speed recovery V=V2=100%
```

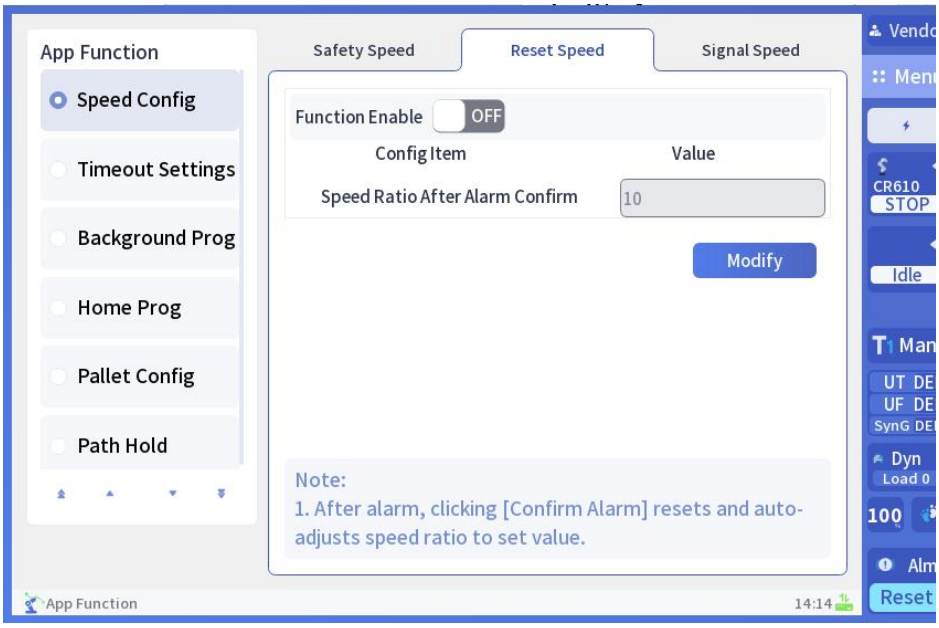
note：

- The IO trigger signal can be a virtual signal.
- The default safety speed multiplier is 10.
- The safety speed trigger and recovery function is only available in automatic and external modes.
- When iSAFE_speed is not configured, the safety speed trigger and recovery functions are disabled, and iSAFE_speed activates on the falling edge.
- The input IO index for the iSAFE_speed flag is configured in the runtime configuration.

- The IO trigger for speed multiplier changes is not mutually exclusive with other VORD operations. Users can still adjust the speed multiplier via the teaching pendant or VORD commands.
- For multi-axis groups, the system currently only supports speed multiplier changes for robot groups, with external axis groups not yet supported.
- If the set speed exceeds the running speed, the system will not respond.

4.8.3.2.2 Reset speed

Introduction The reset speed function automatically adjusts the current magnification setting after clicking [Confirm Alarm] to reset the robotic arm following a system alarm.



- To activate this feature, the system must first trigger an alarm before clicking the "Alarm Confirmation" button. If the system does not trigger an alarm and you click the button directly, the feature will not work.
- This feature requires configuring the alarm confirmation speed multiplier, with the configuration information saved in the system as a file.

Path ➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Speed Configuration] tab → [Reset Speed] label

Note：

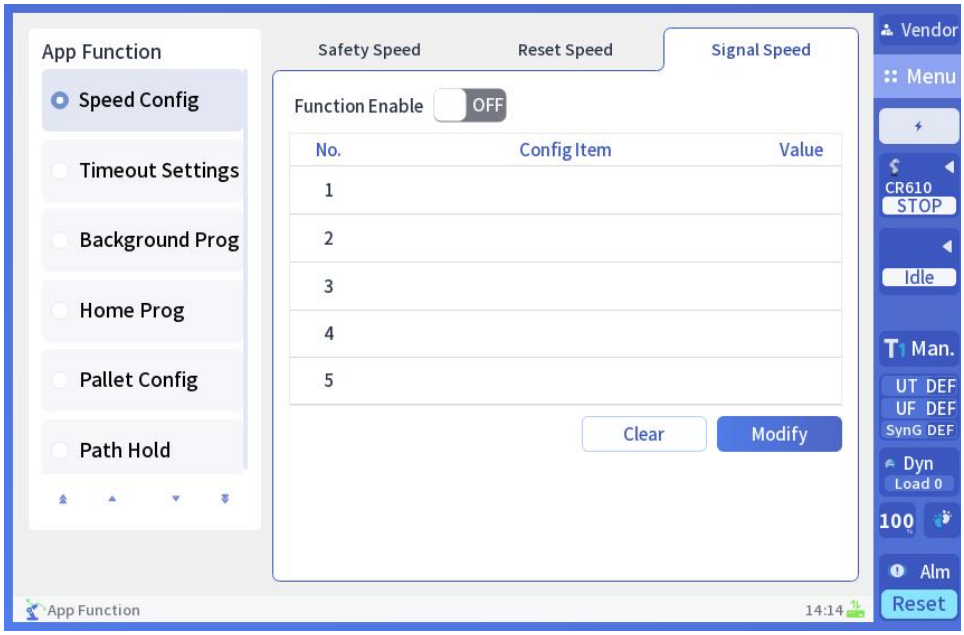
- The default reset speed multiplier is 10.
- The reset speed trigger function is active only in automatic and external modes.
- The reset speed trigger does not conflict with other speed multiplier changes via VORD. Users can still adjust the multiplier through the teaching pendant or VORD commands.
- When the reset speed is triggered, the teaching pendant will display a

notification: "Reset speed triggered."

- Before activating the reset speed, the system must first verify the robot's current speed multiplier.
- If the current speed multiplier exceeds the configured reset speed multiplier, the function will activate upon configuration, allowing the reset speed to be triggered. Otherwise, the function will remain inactive, preventing the reset speed from being activated without any error notification.
- For multi-axis groups, the system currently only supports speed multiplier adjustments for robot groups, with external axis groups not yet supported.

4.8.3.2.3 Signal speed

Introduction In practical applications, the robotic arm will enable different operating speeds according to different application scenarios, and the function supports users to customize the configuration of signal speed.



- After completing the function configuration, you must enable the feature to activate the signal speed trigger function.
- The following options require configuration: the triggering signal and its activation mode (ON/OFF), and the system's operational speed multiplier after signal triggering.
- The function triggers using signal edges, where the configured IO signal edges adjust the robotic arm's speed multiplier to the preset value.

Path

➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Speed Configuration] tab → [Signal Speed] label

operating steps

1、Navigate to the menu → Configuration → Controller Configuration → Speed Configuration. Under the "Signal Speed" tab,
2、select any row from the "Signal Speed Configuration Items" list, then click the [Change] button to open the signal configuration pop-up.
3、The signal configuration register accepts values [0,999], the second input box accepts [0,7], and the multiplier speed value ranges from [1,100].
4、Enable this feature.

Configuration example: "R[N].X=OFF/ON, value V"

- N is the R register number;
- X denotes the number of bits, specifically the binary bit count. The user can configure the R[N] register value, which converts decimal data into binary control signal outputs at specified speeds.
- OFF/ON indicates the signal switch.
- V represents the multiplier value.

Case 1: Modify signal configuration according to instructions

```
'Assume the current multiplier is 50
R[150].1=OFF        '  Current speed multiplier remains unchanged
R[150].1=ON         ' The speed multiplier is set to 20
```

Case 2: Modify the R register value to adjust the response signal speed

```
'Assume the current multiplier is 50
R[150]=60     ' The current speed multiplier remains unchanged (register data is 60, which in binary is 0011 1100, with the first bit being 0, indicating OFF).
R[150]=70     ' The current speed multiplier is set to 20 (register data is 70, which in binary is 0100 0110, with the first bit being 1, indicating ON).
```

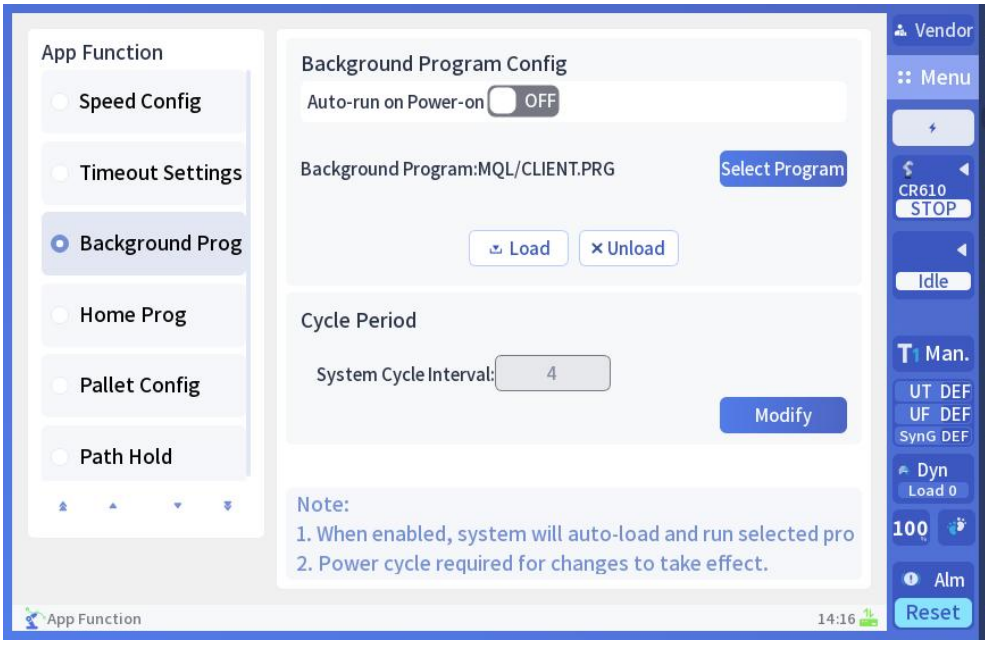
note：

- The IO trigger signal may be a virtual signal.
- The signal speed trigger function is active only in automatic and external modes.
- The IO-triggered speed multiplier change does not conflict with other VORD operations. Users can still modify the speed multiplier via the teaching pendant or through VORD commands.
- When triggered, the teaching pendant will display the message "Signal speed triggered".
- For multi-axis groups, the system currently only supports speed multiplier adjustments for robot groups, with external axis groups not yet supported.

4.8.3.3 Background program configuration

Introduction

The background program configuration feature allows users to set up automatic background program execution upon power-on. The background program is the logical PRG program that runs concurrently with the main program in the controller's background. The main program group mask is 0, while the multi-threaded logical program group mask is *. Additionally, the RUN background program command can be executed within the main program to invoke the background program.



Path

➢ Access the [Menu] page by following these steps: [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Background Program Configuration] tab.

operating steps

1、Click to select a program. Choose the program to run automatically upon power-on.

2、Click to enable the function and activate the configuration.

3、Set the number of 4ms program cycles, default is 1.

4、The power-on restart function takes effect after configuration. If the selected program group mask is 0, the setting does not take effect, and the program will not load or run upon power-on.

constraint condition

The background program runs under the following conditions:：

- The background RUN program is unaffected by the current robotic arm's enabled status.
- Servo alarm. When a servo alarm occurs, the main program terminates. After the alarm is cleared, the main program resumes operation. The background RUN program remains unaffected by servo alarms.
- The main program's error status does not impact the background program's operation.
- External signal outputs follow the main program's status.
- External signal inputs: Load the main program; PAUSE is valid only for the main program; Start, unload, and stop apply to both programs.
- When switching to the background program interface, the background program is not controlled by the trainer buttons (Pause, Unload, Back, Start).
- Both the main program and background program can be individually paused, and can be individually resumed.

Note：

- When editors modify backend logic programs, the system automatically updates the latest version without requiring reinstallation. The ABORT, END, and PAUSE instructions in multi-threading only affect the current program, not the main program.
- At most, two sets of logic programs (backend) and one set of robot motion control (main) can run simultaneously.
- After configuring and powering on the self-running program, the main program must contain only one unique RUN background program instruction during loading. Failure to comply will exceed the two-group loading limit and result in loading failure.
- When executing the RUN background program, if a program with a group

mask set to "*" contains motion commands, the robot will halt operation and trigger an error.

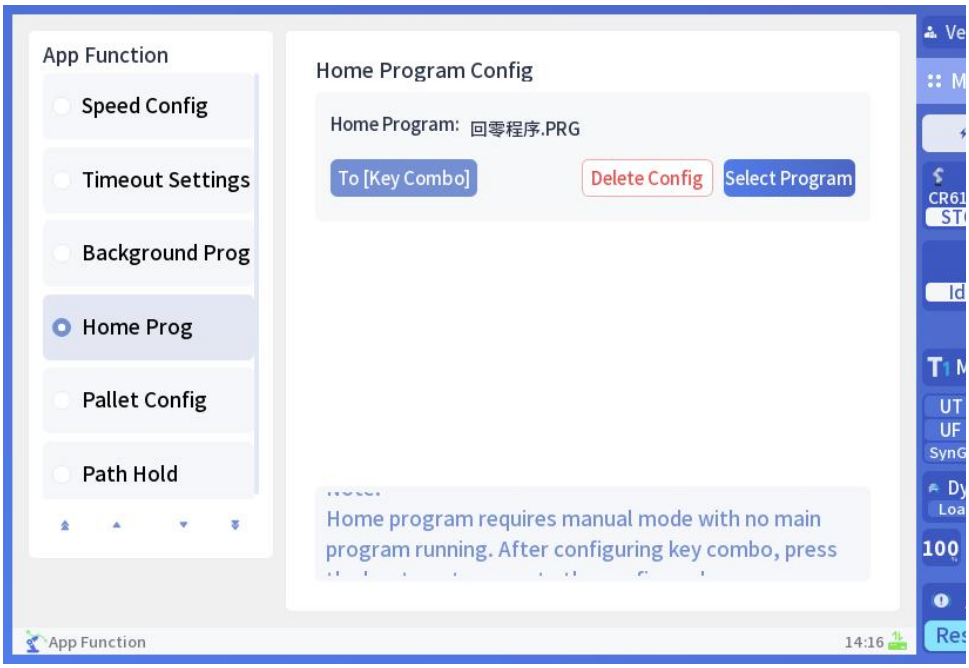
- Programs executed after RUN commands must remain in non-running status. Attempting to run a program already in operation via RUN commands will generate an alarm.
- RUN program alarm description: During program execution, alarm messages from either the main program or RUN program will appear on the teaching pendant. Regarding alarm priority, there is no distinction between main and RUN programs. The system displays the first alarm message of equal priority.

4.8.3.4 Return home point program configuration

Introduction

- The one-click Homepoint return function consists of two components: Homepoint return program configuration and one-click Homepoint return operation.

When the return to Homepoint signal is triggered, the system will automatically execute the Homepoint return program, thus simplifying the operation steps of the robotic arm to return to the Homepoint.

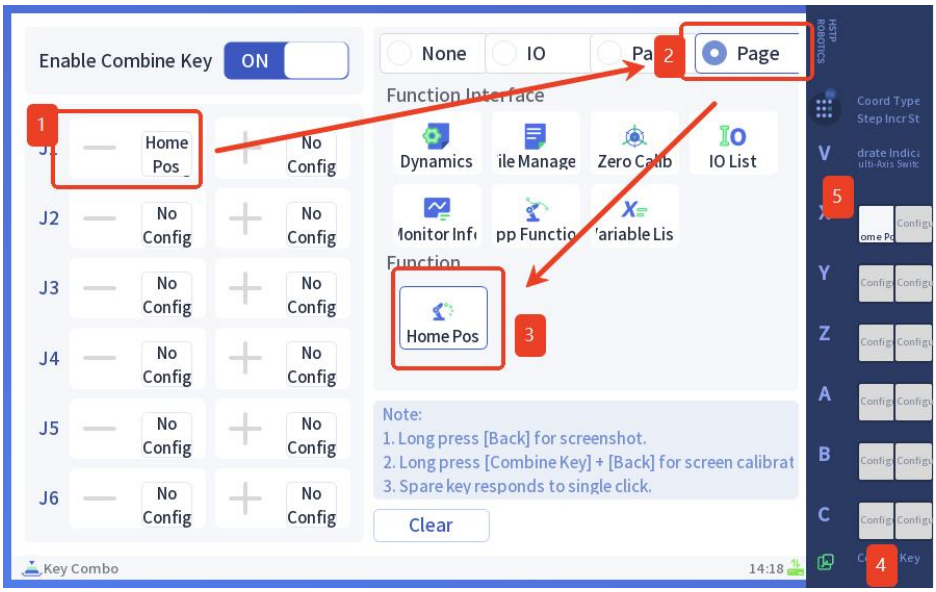


Path

➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Return to Homepoint Program Configuration] tab。

operating steps

1、Click [Menu] → [Function Configuration] → Teaching Pendant Configuration: [Combination Key Configuration].



2、Enable the group button function and select the button position to configure. As shown in the figure, the button is on the negative direction of the J1 axis.

3、Click the configured type: function page.

4、The configuration function to select is: 【return homepoint】 . Save the configuration function.

5、Press the combination button on the back of the teaching device.

6、Simultaneously press the negative direction button of the J1 axis.

7、Follow the above steps to enable the one-touch homepoint return function.

note：

- The homepoint return procedure requires enabling robot (servo poweron)
- The homepoint return program must be executed when the main program is not loaded (idle).
- The homepoint return program can only be executed in manual or automatic mode.
- After completing the homepoint return program, load it once to update the latest program to the controller.

4.8.3.5 Pallet parameter configuration

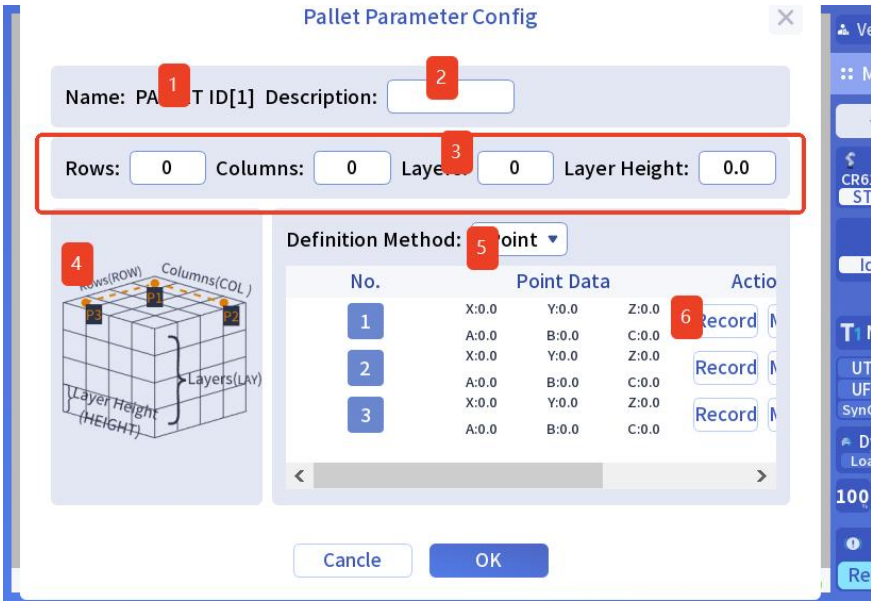
Introduction The pallet functionality is implemented using built-in variables. The software provides 16 predefined system variables, and each pallet can be configured with specific information through the application's parameter settings list.

Path

➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Tray Parameter Configuration] tab

Pallet Parameters Configuration Interface

Select the corresponding pallet parameter row, click the [Modify] button, and enter the pallet configuration interface:



Label	Statement
1	Name ID of the Pallet variable
2	Description of the current Pallet variable

3	Configuration information for the current Pallet variable
4	A diagram showing the Pallet rows and layers
5	Calibration method for Pallet variables
6	Record the current position of the robotic arm in the pallet variable for calibration.

Note：

- If the floor height exceeds the robotic arm's travel range, an' Unreachable Point 'alarm will be triggered when attempting to reach the pallet position. The pallet's reachable status is not verified during configuration.
- The pallet's offset angle calculation is based on point P1, meaning the posture remains constant during the offset process, with only positional changes occurring.

4.8.3.6 Path keep feature

Introduction

The path retention feature enables robotic arm movement after program suspension. When the program resumes, the arm returns to its paused position before executing new commands. Activated, this function allows the current execution pointer to be moved during pauses. Upon restart, the robotic arm travels from its current position to the target point specified by the new pointer line. To prevent collisions, the system prompts users to confirm program execution when resuming operations, especially if there's significant positional change between the pause and restart points.

Path ➢ [Menu] → [Function Configuration] → System Functions: [Application Functions] → [Path Maintenance] tab。

Note：

- During the robotic arm's return to the pause point, users should monitor the surrounding environment to prevent collisions or other safety hazards.
- When the input position deviates from the allowable range, click Save to transmit the data to the controller. The default position deviation is 50mm, and the default posture deviation angle is 5 degrees.

4.8.3.6.1 Execute pointer change notification

Introduction After the program pauses, the user can manually move or jump to the current execution pointer. When the program starts, the robotic arm will move from the current position to the target point specified by the new pointer line. There is a possibility of large trajectory changes. To prevent collision, the user is prompted to confirm whether to jump to the corresponding pointer to run the program.

operating steps

1、When the main program pauses, select another instruction and click [Jump] to move the pointer.

2、When the user clicks the main program run button, a pop-up window appears with a message: "The current pointer exists."

3、Click [OK] to start program execution from the current pointer line. Click [Cancel] to skip execution.

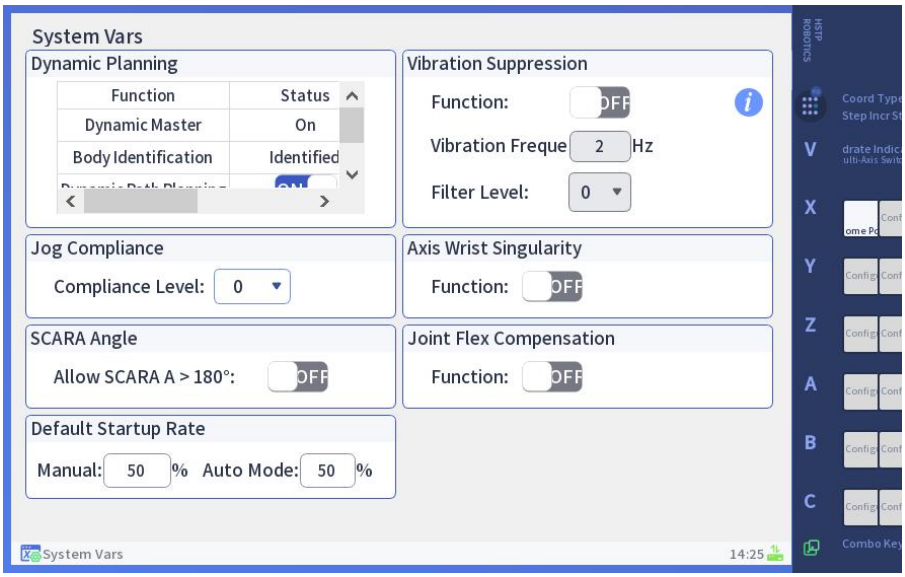
Note：

- After confirming the pointer jump, verify the current position of the robotic arm and the point data of the motion command pointed by the pointer. Pay attention to the environment around the robotic arm to avoid safety risks such as collision.

4.8.4 System variables

Introduction

The [System Variables] interface allows modifying and setting system internal function variables.



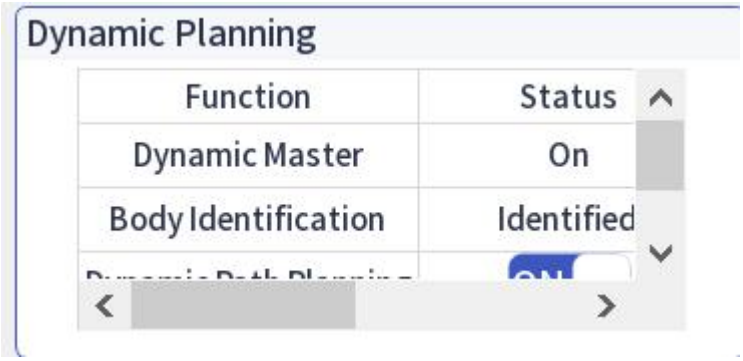
Path

➢ Menu Bar → Function Configuration → System Functions: System Variables

4.8.4.1 Dynamics planning feature

Introduction

The dynamic planning information view and dynamic planning switch function allow users to set the dynamic adaptive planning switch parameters.



Dynamic master switch: If the dynamic model is established and the dynamic model is calibrated, the dynamic master switch will be automatically turned on.

Body identification information: shows whether the robot arm has completed the friction identification of the body during factory production.

Dynamic path planning switch: The default switch is off. When enabled, dynamic speed and trajectory planning is used。

Note：

- The error of dynamic identification and load identification of each robotic arm may be greatly affected by the influence of ontology difference.。

4.8.4.2 Vibration suppression

Introduction

The vibration suppression function can filter the output commands (position and speed) after trajectory planning is completed to meet the requirements of speed stability. The command filtering may change the trajectory characteristics and reduce the trajectory accuracy. Therefore, there are parameters to set whether to filter, the filtering level, and the maximum position deviation.

- Vibration Frequency: The natural frequency of the robotic arm, ranging from 0.00~20.00Hz. This non-negative value can be input with two decimal places.
- Filter Level Setting: Different levels indicate varying filtering effects. Higher values enhance filtering while reducing vibration (maximum 7), though this increases trajectory deviation. Level 0 means no filtering.

4.8.4.3 Jog Smoothness settings

Introduction Adjust the smoothness of the user's interface movements. If the smoothness is too soft, the start and stop may be slow (long time to move to the point). If the value is too small, the user may experience jitter. Users can set the interface according to their actual requirements.

Note：

- Value range: 1 to 9; default value is 5. A higher value results in smoother acceleration and slower changes.
- SMOOTH=1 enables the fastest acceleration during start-stop phases, which may cause jitter; SMOOTH=9 enables the slowest acceleration during start-stop phases, providing the smoothest performance.

4.8.4.4 Singularity protection

Introduction

After enabling the wrist singularity function, when the robotic arm performs linear motion by stepping, the speed of the robotic arm remains unchanged when approaching the wrist singularity, and the 4th and 6th axes can pass through the position where the 5th axis Angle is equal to 0 without flipping.

note：

- The configuration is used in jogging operation. When controlled by a program, the WRISTJNT command can be used to activate the wrist's singular motion zone.

4.8.4.5 SCARA angle A

Introduction

This function variable allows changing the representation of the A angle for SCARA models. By default, the A angle for SCARA models is set to allow values beyond ±180 degrees. This variable enables you to modify and check whether the function is enabled.。

Note：

- When displaying the current A-angle value on the teaching pendant, the A-angle value may exceed ±180.
- When recording spatial points, the A-angle value may exceed ±180.
- When executing the J LR[x] and L LR[x] instructions, the system directly moves to the current A angle value.

4.8.4.6 Joint flexibility compensation function

Introduction

Flexible compensation is a system function based on the rigid-flexible coupling dynamics of the robot to overcome the joint deformation of the robot under different loads. Turning on the flexible compensation function can improve the absolute positioning accuracy and trajectory accuracy of the robot.

note：

- The supported robot models are JR607 and JR6210.

4.8.4.7 Set the default vord after controller startup

Introduction

After setting the default vord in manual mode, the startup vord for T1 and T2 in manual mode will be the set value when power is cut and the system restarts.

In non-manual mode, the startup vord for automatic mode and external mode will be the set value after power is cut and the system restarts.

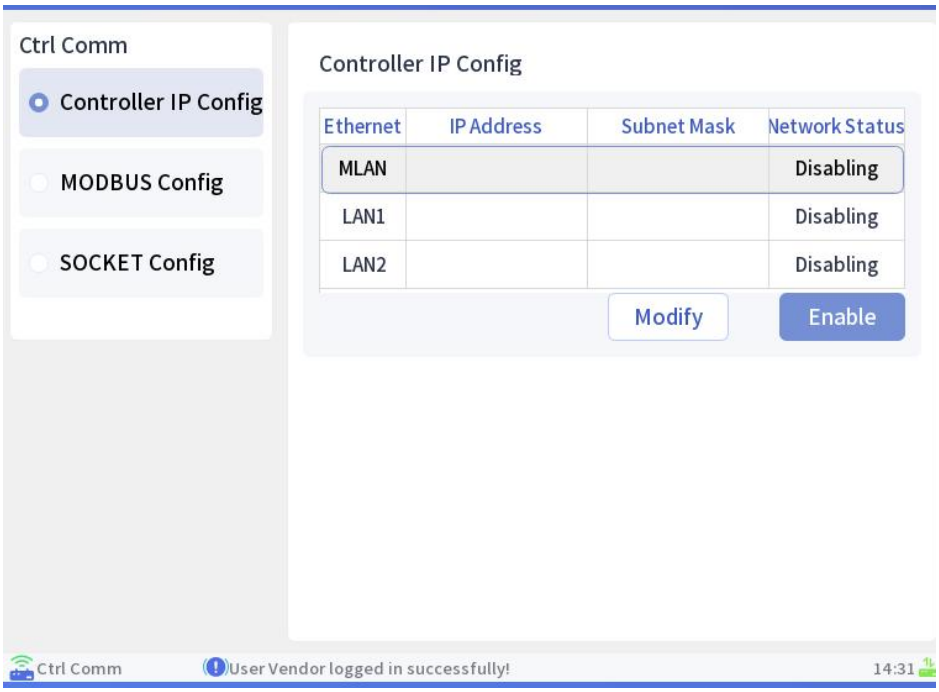
4.8.5 Communication settings

4.8.5.1 Controller IP configuration

Introduction

The robot controller has two or more network interfaces. In some scenarios, users can configure the IP address of the robot controller's network interface through the [Controller Communication Configuration] page in the teaching device.

The operation interface is shown in the following figure



Path

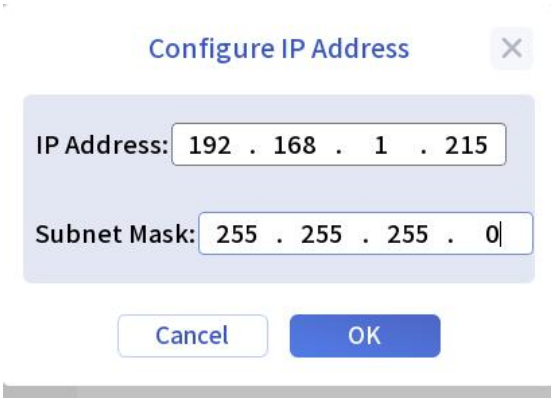
➢ Menu Bar → Function Configuration → System Functions: Communication Configuration

operating steps

1、Log in as a debugger or higher-level user. In the menu bar,

2、select → Function Configuration → System Functions: Communication Configuration, then choose Controller IP Configuration.

3、Select the corresponding network port configuration line and click [Modify]. The pop-up window allows you to configure the IP address and subnet mask.



4、After configuration, click OK. Follow the on-screen prompts to enable the feature and restart the system for configuration to take effect.

5、Adjust the network status by clicking Enable or Disable and restarting the system.

note：

- The IP address cannot match the controller's IP address. For example, configuring 10.10.56.213 as 10.10.56.214 or conflicting with an IP address added to the old-style n file will cause system module errors.
- Note: The system has already used the segment number with IP address 192.168.2.X. Avoid using it.
- The system does not allow setting two IP addresses in the same subnet (10.10.56.x), as this may cause network conflicts and communication chaos, affecting network stability and security.

Standard Cabinet Controller Default IP

internet access	Default IP
LAN0	IP：10.10.56.214 Port：23234 IP：90.0.0.1 Port：23234
LAN1/ECAT	IP：192.168.2.214 Port：23234

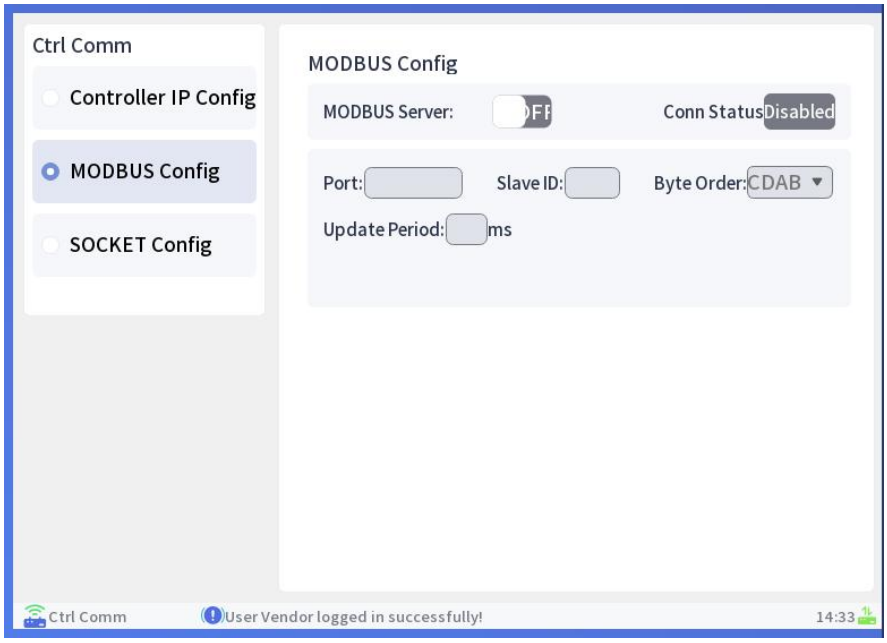
Integrated Drive and Control Cabinet Controller IP Specifications

internet access	Default IP
Teach Device Network Port	IP：10.10.56.214 Port：23234
MLAN	IP：10.10.57.213 Port：23234
LAN2	IP：10.10.58.212 Port：23234
LAN1	IP：10.10.59.211 Port：23234

4.8.5.2 MODBUS configure

Introduction

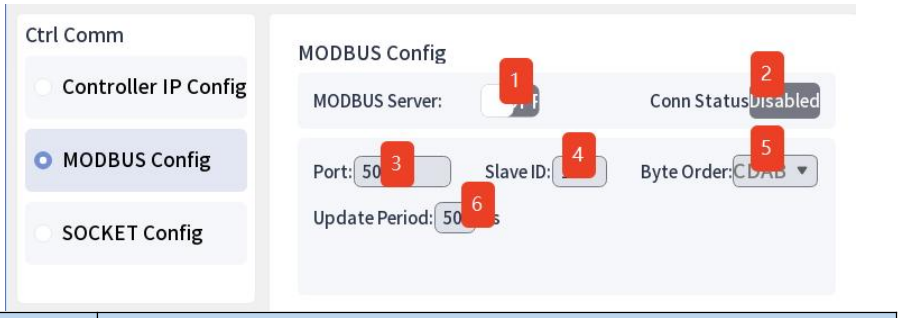
When a robot acts as a Modbus slave (server), the function must first be enabled in the teaching pendant. For details of the communication protocol, please refer to the "V1.6.11_Modbus Slave Control Protocol User Manual" 。 The operation interface is shown in the figure below.



Path

➢ Menu Bar → Function Configuration → System Functions: Communication Configuration → "MODBUS Configuration" tab

operating steps



Label	Introduction
1	Modbus server function switch. Enables or disables Modbus communication.。
2	Display the current communication connection status
3	Port settings. Configure the communication port. The default value is 507. When setting the port, avoid conflicts with other TCP/IP ports.。
4	Slave ID. The slave ID for configuring the robotic arm. The default value is 1. For example, in Siemens PLC communication, if the PLC programmer fails to configure the slave address for a function block, the PLC may use broadcast mode. In this case, the

	SlaveID must be set to 255. (If communication fails and the slave address is incorrect, you can connect the PLC to a computer's debugging tool to verify the actual message sent by the PLC, identify the correct slave ID, and then input it accordingly.)
5	High and low bit order. Configures the sequence of high and low bits for a 32-bit single-precision floating-point data composed of two 16-bit Modbus registers.
6	Data update period. Controls the speed of system data synchronization. The default value is 50, measured in milliseconds. A smaller update period means faster data updates, but it also increases the CPU load on the controller..
7	[Cancel button]. restore the current modification to the last saved data.
8	[Save button] saves the modified data to the system. The changes take effect only after restarting the system.

note：

- After saving, restart the system for changes to take effect.
- Modifications are only allowed when the MODBUS function switch is on.

4.8.5.3 SOCKET configure

Introduction

The SocketCmd network terminal command is a standardized string-based control protocol agreed upon by the robotic arm controller. After establishing Socket communication between the host computer and the robotic arm, sending the specified command string allows the robotic arm to perform relevant actions or retrieve system status information.

When using the SocketCmd network terminal command, the robotic arm acts as the server with the default communication port 23333. The IP address is the IP address of the robotic arm controller.

To use the SocketCmd terminal command, connect the host computer and the robotic arm controller via a network cable (select the appropriate network port), then enable the SocketCmd function on the teaching pendant's interface. The host computer acts as the client, connecting to the robotic arm controller's IP address and port number. By sending a protocol string, you can control the robotic arm.

Path

➢ Menu Bar → Function Configuration → System Functions: Communication Configuration.

note：

- The IP address of the LAN0 network port on the controller in the IPC cabinet：10.10.56.214.
- The integrated drive and control electrical cabinet has multiple network ports, with the following IP addresses: MLAN IP: 10.10.57.213; LAN2 IP: 10.10.58.212; LAN1 IP: 10.10.59.211. For details, refer to the controller IP address section in Chapter 4.8.5.1.1.
- To use SocketCmd, connect the host computer and robotic arm controller with a network cable (select the appropriate network port) and

enable SocketCmd on the teaching pendant. The host computer acts as the client, connecting to the robotic arm controller's IP address and port number. Send the protocol string to control the robotic arm.

- For details, please refer to the<<V1.6.11_SocketCmd Secondary Development Communication Protocol Manual_A01_CN>>

4.9 Dynamics

4.9.1 Installation posture settings

Introduction

Generally, robotic arms are installed on stable workbenches or floors, requiring no additional configuration. However, when using reverse installation, wall-mounted side installation, or dynamic-related functions, the current installation posture must be set. This can be done through the [Installation Posture Settings] page in the teaching pendant, as shown in the figure below.

Path

➢ Menu Bar → Function Configuration → Dynamics: Installation Pose Settings

operating steps

6、Select or set the installation posture based on the actual robot posture. First, check the "Common Posture Installation" interface for available options. If no suitable posture is available, click to enter the "Custom

Posture Installation" interface to configure your own installation posture.

7、Set local gravity

8、Click the [Save] button to save the data to the system and restart the controller.

9、After setting, the ontology friction identification function needs to be executed again.

point out ：

- The installation posture is set to improve the accuracy of the dynamic model and ensure its precision. If inverted use is required, the installation posture should be set before identification. The safe posture should also be correctly set before using the dynamic function.
- After changing the installation posture, the robot arm body must perform friction identification again. Otherwise, it will affect the related functions of the robot arm system and lead to safety risks such as collision with the machine.
- After setting the installation posture, the software's 3D view will change the default perspective to the installation posture.

4.9.2 Body friction recognition

Introduction

The dynamic parameters of the robot are very important for the accuracy and reliability of the robot. Each parameter needs to be configured in the control system before the factory, and the identification process is not provided externally.

The body friction identification operation can be performed through the [Body Friction Identification] page in the teaching device. The operation interface is shown in the following figure:

Label	explain
1	Recognition status display (changes to green after recognition)
2	Return to the start of the incentive trajectory
3	Trial Run Incentive Trajectory
4	Run the trajectory
5	Identify Calculation

Path

➢ [Menu] → [Function Configuration] → [Dynamics] → [Body Friction Identification]

operating steps

Run three incentive paths in sequence, one by one.

10、Enable the upper one.

11、Click the [Back to Start] button to return to the start point.

12、Click the [Trial Run] button and wait for the trial run to complete. The [Start Incentive] button will then become clickable.

13、Click the [Start Incentive] button and wait for the incentive to complete.

14、Activate the next trajectory in sequence.

15、After completing all three paths, you can tap the [Identify] button.

16、Click the [Identify] button. Wait a moment. When the identification is successful, you can choose to save or not save the data.

point out ：

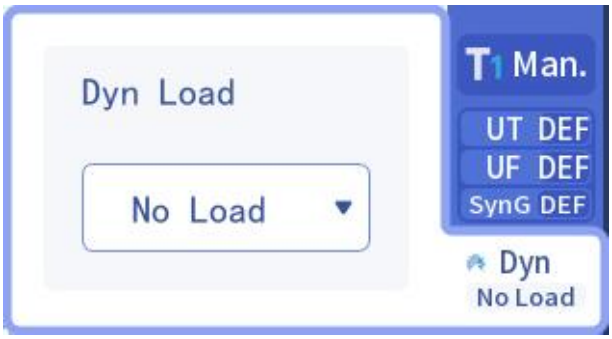
- To go back to the start of the track, you need to do it in manual mode.
- The required workspace for the actuator trajectory is large. Check whether the actuator trajectory may collide with the machine.
- If you do not save the identification data, the system will restore the last saved identification data after restarting.
- The starting point of each excitation trajectory matches the identification trajectory. To execute the identification trajectory, click the [Return to Starting Point] button in the corresponding row to return to the starting point and then run the trial and trajectory excitation functions.

4.9.3 Load configuration

Introduction	Load configuration is a system function that adjusts dynamic parameters to determine the current load conditions of a robotic arm. These parameters can be manually input or acquired through load identification methods. Proper configuration of load parameters enhances the robotic arm's operational performance and optimizes its dynamic functions, including collision detection and drag teaching capabilities. To prevent errors or unintended effects in these functions, the robotic arm must undergo load identification before use, with the corresponding load code selected.

4.9.3.1 Dynamic load switching

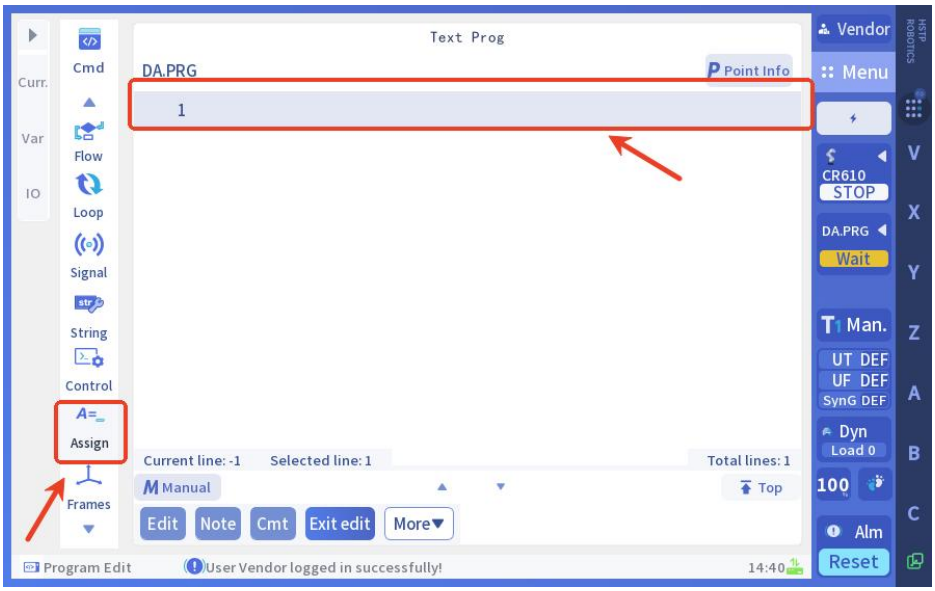
Introduction	Dynamic load switching is a function that allows the robotic arm to reacquire relevant load parameters under the current load number by switching the load ID, thereby determining the current load status. The system contains 16 sets of load ID data. Load switching can be performed in two ways: either directly via the teaching pendant button or through command-based switching.
Method 1	Click the [Dynamic Load] control in the trainer status bar to select the corresponding load number from the drop-down list.



Method 2 The PAYLOAD assignment instruction is set in the program.

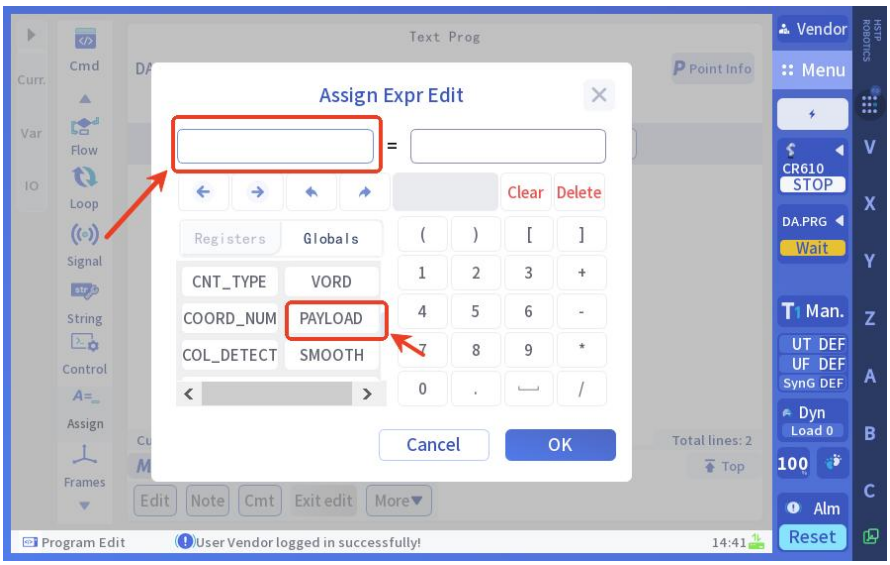
17、Select the line of code to insert.

18、Select the assignment instruction type in the instruction information to open the assignment expression editing pop-up window.



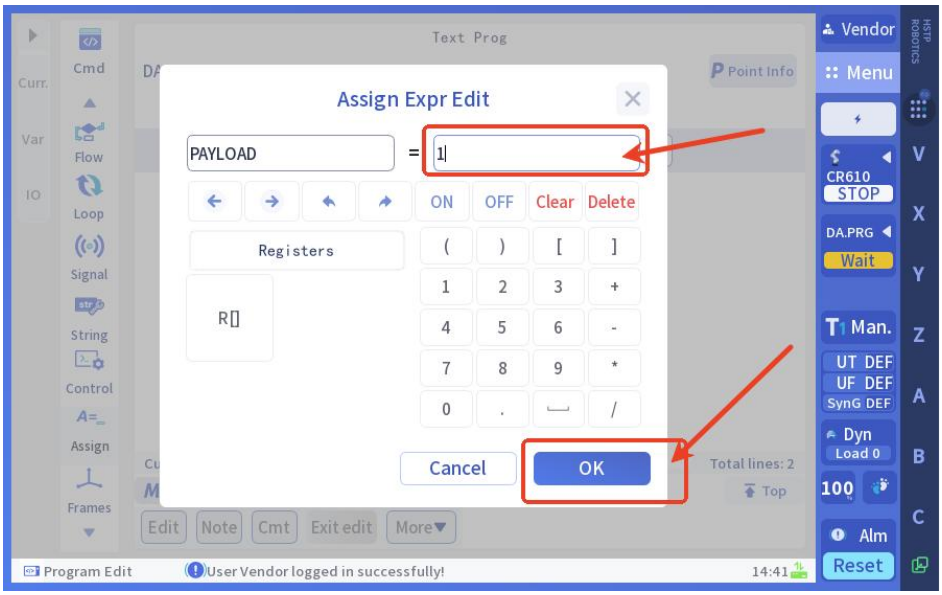
19、Select the left input box and choose the Global Variables tab.

20、Find and click the [PAYLOAD] button in the Global Variables tab.



21、Click the input box on the right and enter the value.

22、Click the [Confirm] button in the pop-up window to add the command.



Example:

```
PAYLOAD=1 '指定当前使用编号为1的负载，可设置编号范围为-1到15，默认负载为-1空载。
```

point out ：

- When the load of the robotic arm changes, it should be switched to the corresponding load number in time, otherwise it may lead to the decrease of motion accuracy, false alarm of collision detection or adverse effects on the drag force.
- The load number switching policy for loading and unloading programs is the same as that for tooling workpiece switching. After unloading, the current load remains the same as the load used before unloading.

4.9.3.2 Load parameter configuration

Introduction

Load parameter configuration sets parameters such as load quality, center of mass position, and inertia tensor. There are two methods to configure load parameters:

- Configure load parameters manually: If you know the load parameters,

you can modify the values directly and save them to complete the configuration.

- Automatically generate load parameter information based on load identification. This method is suitable for load parameter configuration when load information is unknown.

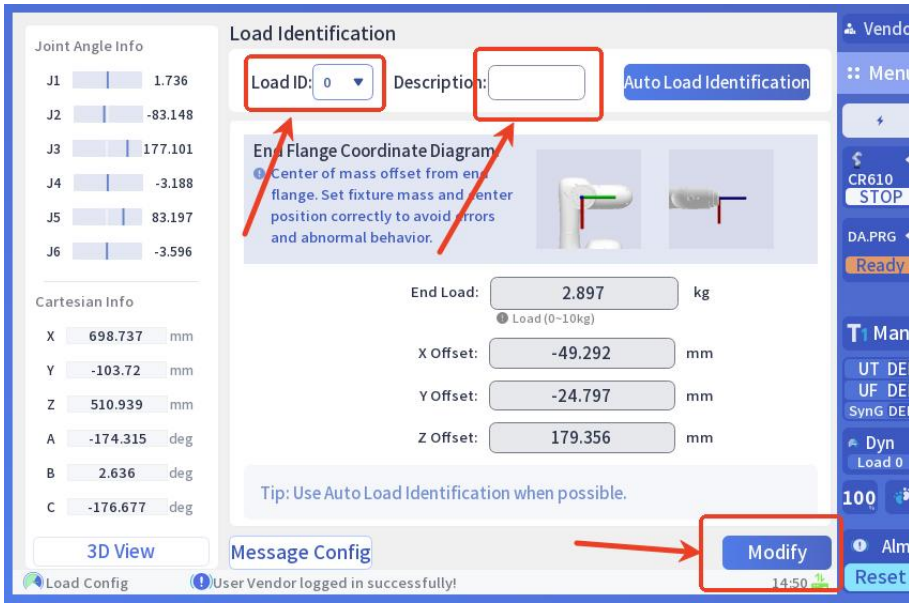
Method 1 Configure load parameters manually.

23、Go to the load configuration page.

24、Click the Load Number Selection dropdown to select a load number and enter the parameter page. The page displays load parameters for that number, which is not the current load number used by the robot.

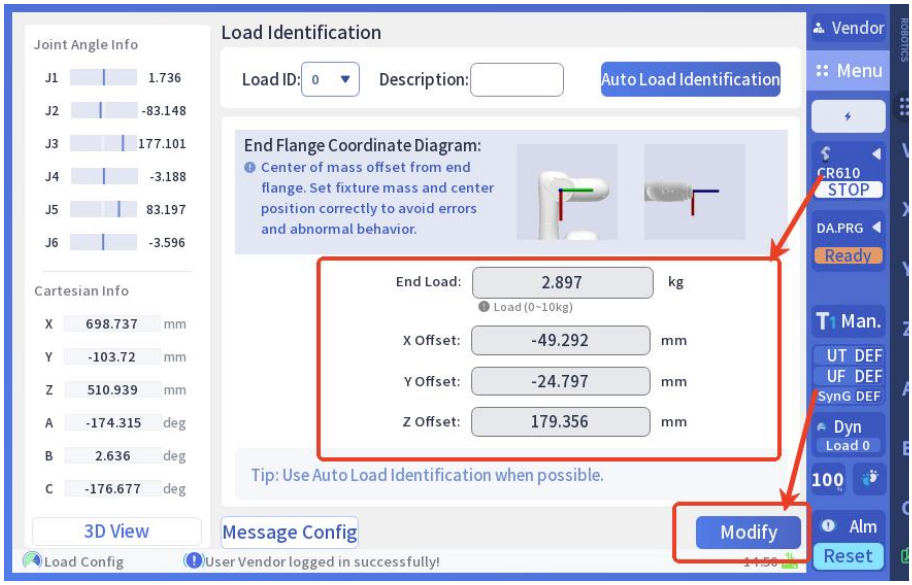
25、Click the "Description" input box to add notes to this load number.

26、Click the Edit button.



27、Select the corresponding load parameter input box and manually enter the load parameter value based on the actual load data.

28、Click the Save button to save the changes and complete the load parameter configuration.



warn ：

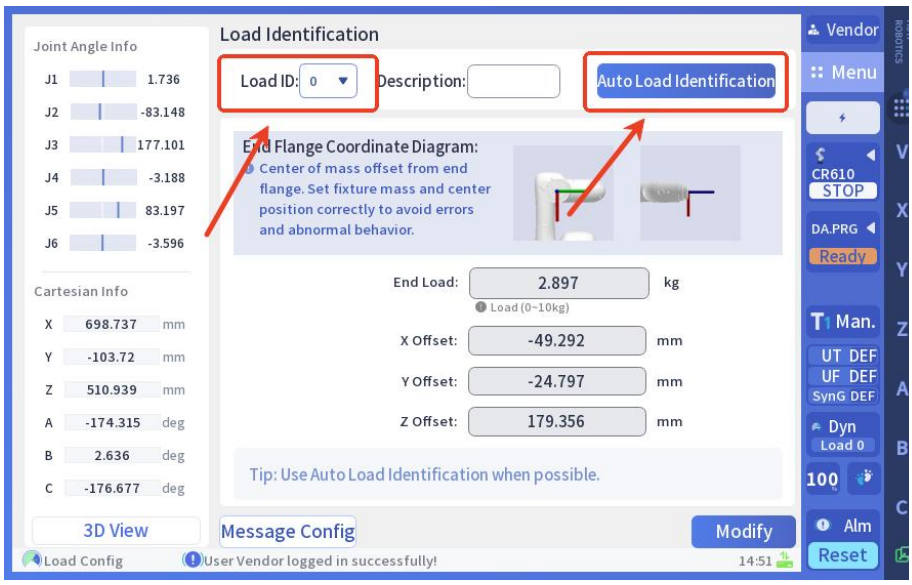
- As the BR series robotic arm adopts an internally rotating structure, users should pay attention to the following two scenarios when manually adjusting load identification parameters:
- Avoid the position where the third axis joint of the robotic arm approaches 270°
- The fourth joint axis of the robotic arm should be within the range of 0 to 180 degrees
- If the model is under load, we recommend using the first solution. Also, avoid the three axes approaching 270° to prevent interference between the load and the main body.

Method 2 Automatically generate load parameters based on load identification.

29、Go to the load configuration page.

30、Click the Load Number dropdown and select the corresponding load number (to identify the load based on this number).

31、Click the [Automatic Load Identification] button to access the load identification settings page.



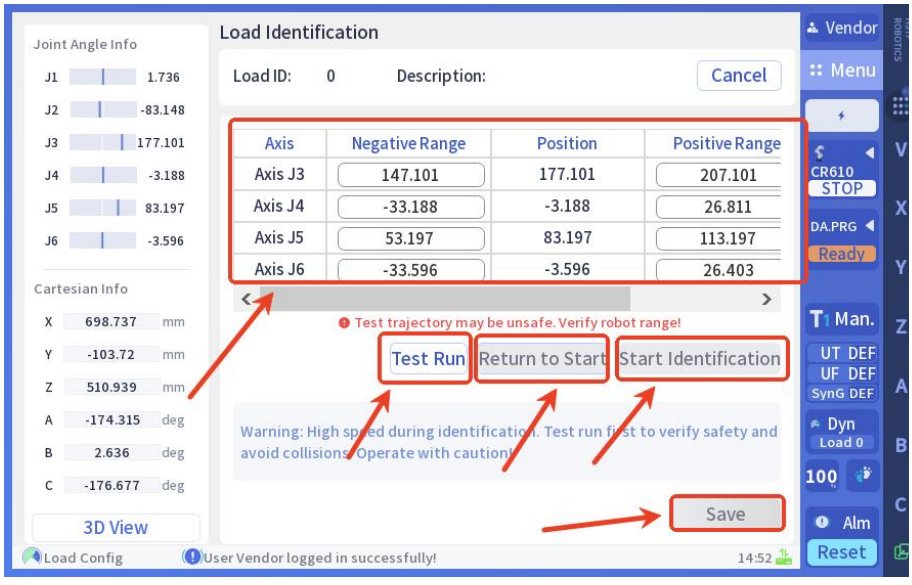
32、Click the "Negative Range" or "Positive Range" input box to reset the movement range of axes A3 to A6 relative to the current position. The default positive/negative range is automatically set based on the soft limit range.

33、Click the [Trial Run] button. A confirmation pop-up will appear. Click [Confirm] to start a slow trial run and check the safety of the environment. (After a successful trial run, you can click [Return to Origin], [Start Excitation], and [Save].)

34、Click the [Return to Origin] button to confirm the robotic arm's return to the trajectory's origin.

35、Click the [Start Incentive] button to initiate high-speed trajectory recognition (this is part of the load identification data collection process, with the same trajectory repeated ten times).

36、After identification, click [Save] to complete the load parameter configuration.



Path

➢ After completing the object load identification, access the menu → Function Configuration → Dynamic Functions: Load Identification.

➢ point out ：

- The trajectory trial run can only be performed in manual mode. High-speed trajectory identification can be performed in either manual or automatic mode.
- Before load identification, the corresponding load should be hung on the end of the robotic arm, and the robotic arm system should be idle.
- The collision detection function is automatically disabled during high-speed tracking.

4.9.4 Collaboration security settings

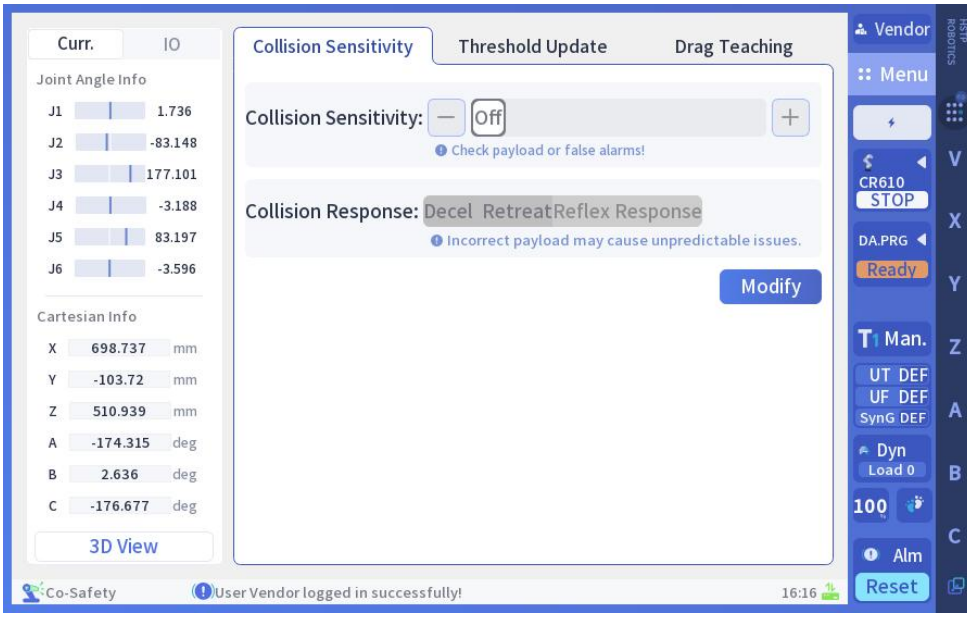
Introduction

The collaborative safety settings primarily allow configuration of dynamic parameters, including collision detection sensitivity and the flexibility of drag-and-teach.

Path

➢ Menu → Function Configuration → Dynamic Functions: Collaboration Security Settings

Collision detection sensitivity settings



Graph

operating steps ：

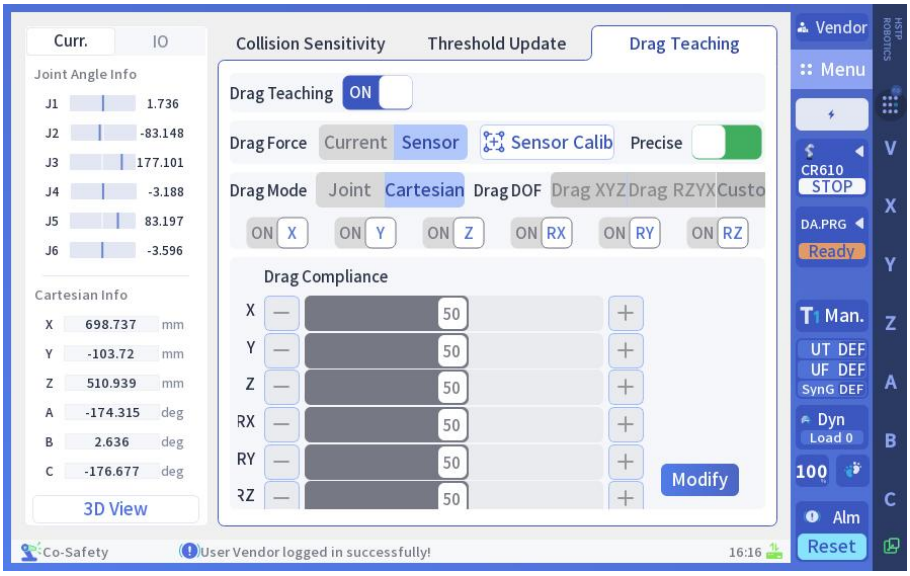
37、Click the [Edit] button to enter editing mode and drag.

38、Drag the [Collision Detection Sensitivity] slider to set the appropriate sensitivity value.

39、Tap Slow Back or Reflex Response to switch collision response mode.

40、Click the [Confirm] button. When the message "Settings completed" appears, the modification is finished.

Drag smoothness settings



41、Click the [Drag Teaching Settings] tab at the top of the page to switch to the [Drag Teaching Settings] page.

42、Drag the teaching switch. The robotic arm can only perform drag teaching when the switch is turned on.

43、You can select the source of the pull force, which is mainly [current] or [sensor].

44、Drag mode can be selected as [Joint] drag or [Cartesian] drag.

45、In joint drag mode, click the [Modify] button to adjust the drag smoothness from axis 1 to axis 6.

46、In Cartesian drag mode, click the [Modify] button to adjust the drag degrees of freedom and sensitivity. The drag degrees of freedom include [Drag XYZ] and [Drag lRZYX] as shortcut options, with [Custom] providing six settings for X, Y, Z, RX, RY, and RZ. After configuring these settings, the drag sensitivity and degrees of freedom will be automatically applied.

47、Click the [Save] button. The message "Configuration successful!" will appear, indicating the configuration modification is complete.

point out ：

- Debugging requires higher permissions.
- The entity must be identified.

4.9.5 Drag to record

Introduction

Users use the drag function of the collaborative robot to drag the robotic arm for trajectory teaching. The software will record the motion trajectory of the robotic arm shown by the user, and then generate the corresponding robot program, which saves the point of the trajectory shown by the user. The program can be executed to complete the reproduction of the trajectory shown by the user.

Path

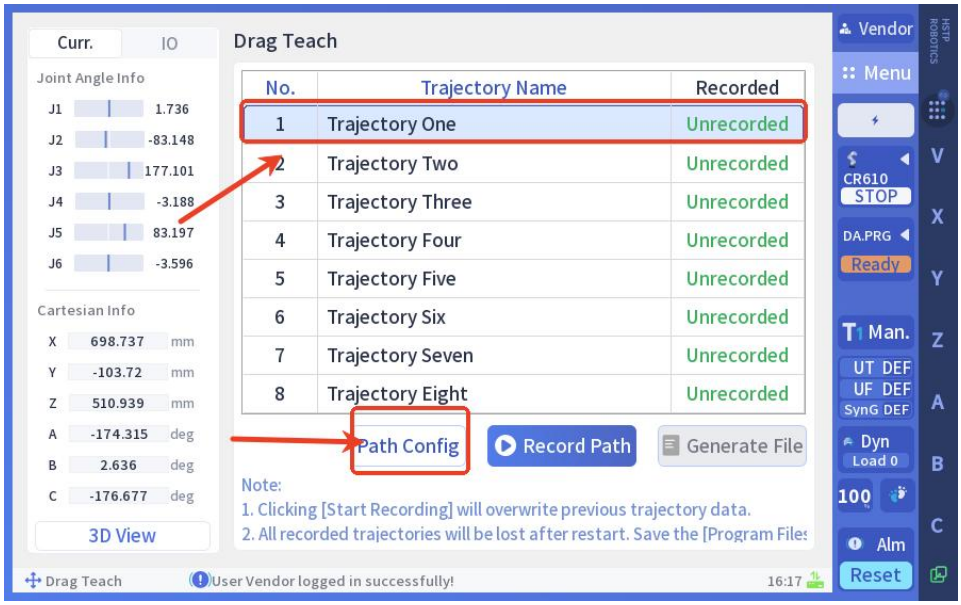
➢ Access the menu → Function Configuration → Dynamic Functions: Drag to record trajectory.

operating steps

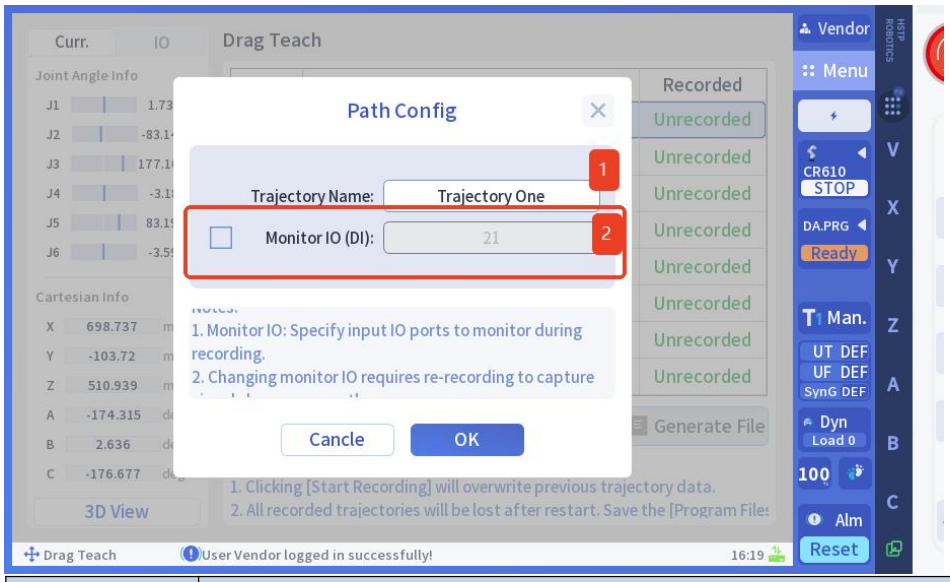
48、Select path:

49、Go to the drag track recording page.

50、Select the track to record. The right content box displays the recording status of the current track. Up to 8 tracks can be recorded simultaneously. After selecting a track, click [Track Parameters Configuration].



After clicking the [Trajectory Parameters Configuration] button, the following pop-up window will appear:

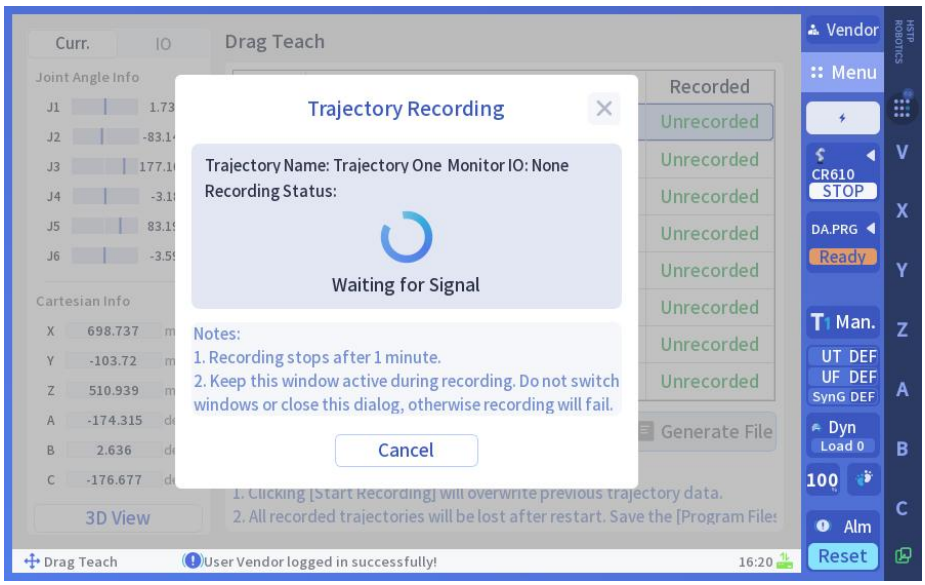


Label	explain
1	Name the track. Customize the name of the recorded path.
2	Monitor the IO (DI) input box. Number for enabling and monitoring input IO. Monitoring IO refers to the input IO recorded simultaneously during trajectory recording. It captures changes in this IO while dragging the recording trajectory. The current version only supports recording one IO. You can choose to record or not: Check the left box to record (the input box becomes editable), or uncheck it to skip recording (the input box becomes uneditable). Default is unchecked.

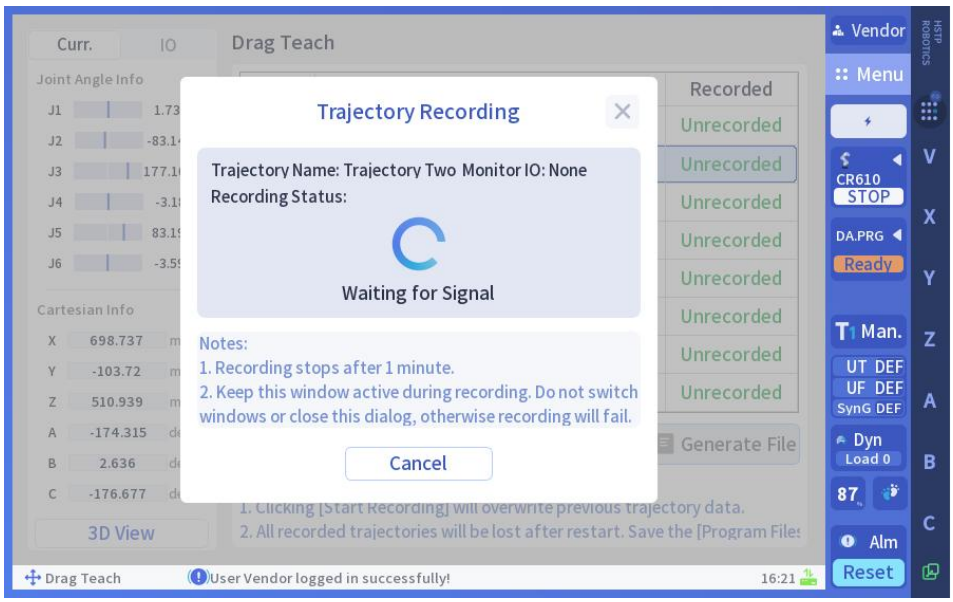
51、Click the [OK] button in the pop-up to configure the selected trajectory. If the trajectory status in the table shows "Not Recorded", it means the trajectory has not been recorded. You cannot click the [Generate Program File] button to generate the trajectory. In this case, click the [Start Trajectory Recording] button to begin the recording.

52、Drag the teaching trajectory:

Click the [Start Track Recording] button to open the track recording pop-up window.

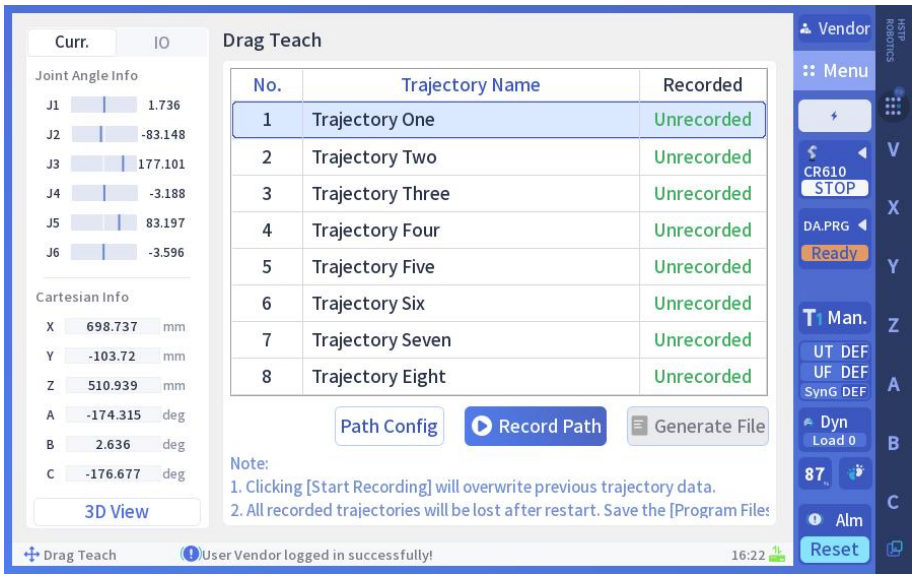


The recording status is displayed in the trajectory recording pop-up window. The "Waiting for Teaching Signal" shown in the figure above indicates the software is awaiting the user's trigger signal. When the user presses the drag button at the flange end, the pop-up window changes to "Waiting for Teaching Completion", as shown in the figure below.



This page indicates that the software has entered the trajectory teaching recording mode. The software will record the user's teaching motion trajectory of the robotic arm in real time. The recording duration is limited to one minute. Any trajectory exceeding this time will not be recorded. If the user wants to cancel the recording midway, they can click the [Cancel] button. The software will return to the main page, and the user can click the [Start Trajectory Recording] button again to pop up the trajectory recording window. During the recording process, releasing the drag button will automatically end the recording and return to the main page.

53、Generate trajectory:



After successful recording, the trajectory status will change to "Recorded". Click the [Generate Program File] button to proceed with the next step of trajectory program generation. A pop-up window will appear as follows:

Label	explain
1	Generate a PRG program name input box. Customize the final PRG program name for users.
2	Recreate the IO (DO) input box. Number for enabling and input reproduction IO. During trajectory replay, the IO (Output Type IO) is also replayed. The current version only supports replaying one IO. This option can only be checked

	after selecting the monitoring IO in Step 1 and successfully recording. You can choose to replay or not. Checking the left box enables replay and changes the input box to editable. Unchecking it disables replay, with the input box remaining uneditable by default.
3	Point sampling period input box. For custom point sampling periods. The system generates a trajectory by sampling points at intervals of 10. The default setting is 10, meaning it samples points every 10th point along the original trajectory. A higher value reduces the precision of the recorded trajectory. The range is [5,50].
4	Restore the estimated operating multiplier dropdown. Used to estimate the playback speed during reproduction. The recorded trajectory will be retraced at the set speed ratio. However, in some cases, accurate retracing may not be possible due to limitations such as the maximum speed of the aircraft model or command specifications.
5	Smoothness input box. Smoothness for custom trajectory runs. This configuration item ranges from 1 to 100. A higher value increases trajectory smoothing (corresponding to the CNT instruction in the generation program).

point out ：

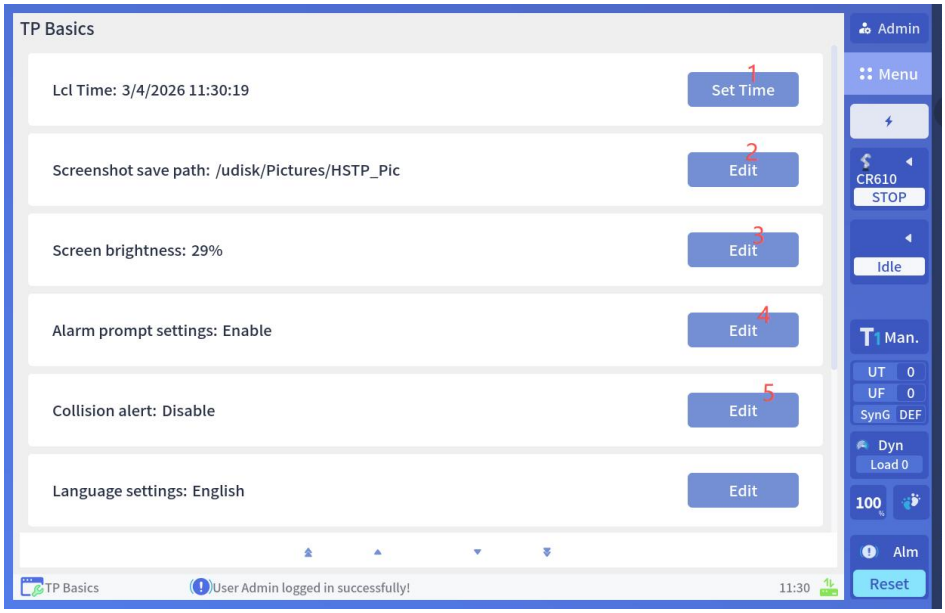
- The drag signal for the collaborative model is the drag button at the end of the flange.

- Recording the same track will overwrite the original data. A pop-up will appear with the message: "This track is already recorded. Re-recording will overwrite the original track data." Please proceed with caution.
- After recording a trajectory, you can regenerate it. Click [Generate Program File] to change the configuration, then click [OK]. The system will regenerate the trajectory program based on the configured parameters.

4.10 Teaching assistant configuration

4.10.1 Teaching assistant basic configuration

Introduction Users can configure basic behaviors, parameters, or functions of the teaching device, such as local time, screen capture file save location, screen brightness, alarm prompt configuration switch, and collision detection prompt switch. The configuration operation interface is shown in the following figure.



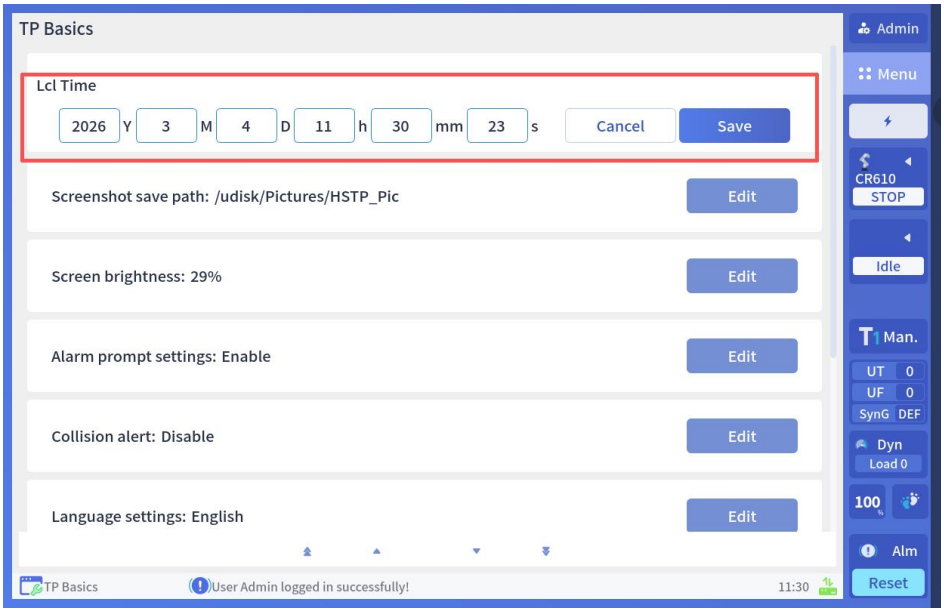
Label	explain
1	Edit local time
2	Change the location of the screenshot file
3	Adjust screen brightness

4	Change alarm notification settings
5	Change collision detection prompt

Path

➢ [Menu] → [Function Configuration] → [Teach-Button Configuration]: [Teach-Button Basic Configuration]

demonstrator Time Settings

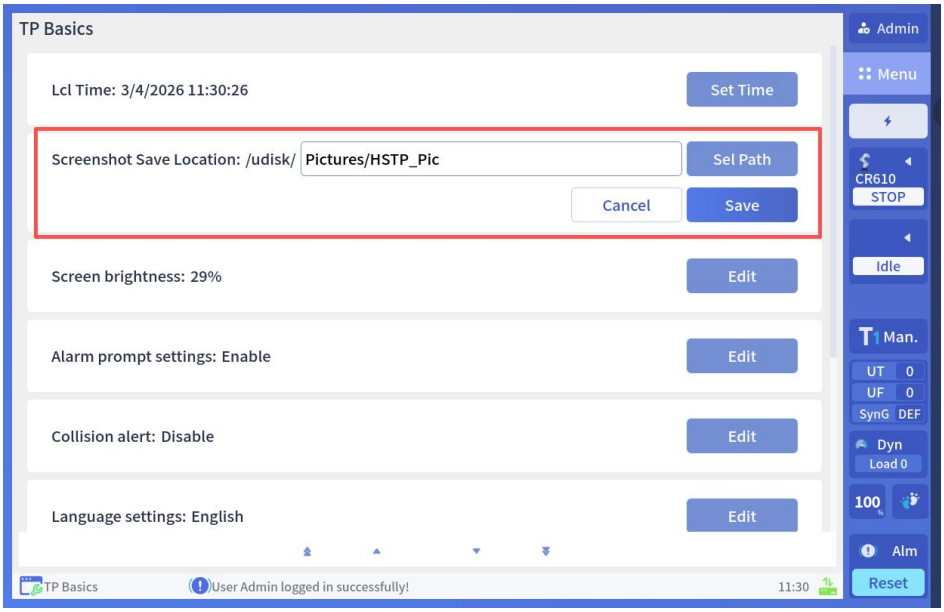


54、Click the [Modify Current Time] button to expand the local time modification page, as shown in the figure above.

55、Enter the date and time in the format: year, month, day, hour, minute, second.

56、Click Save to complete the modification.

Change the screenshot file location



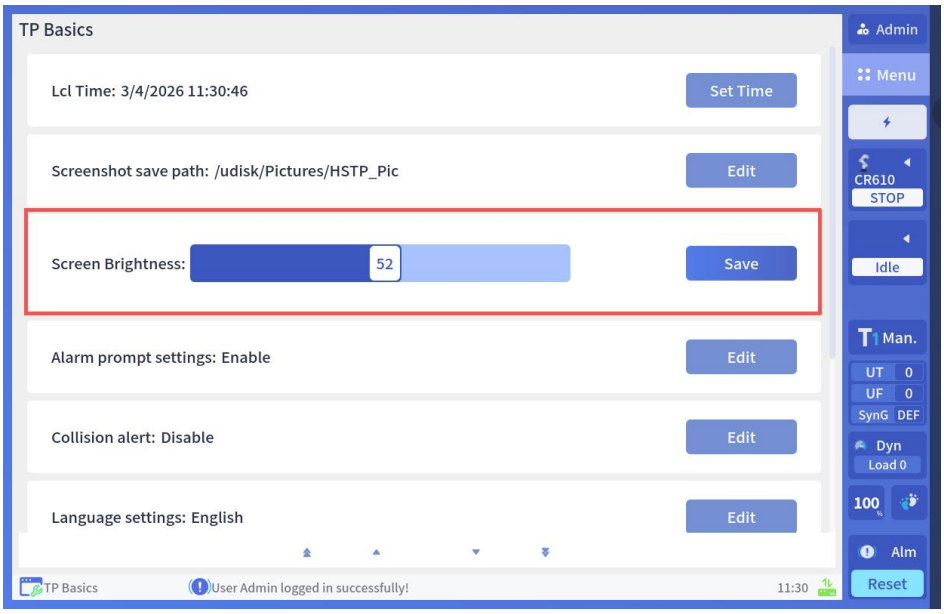
57、Click the [Edit] button in the screenshot file save location to expand the configuration interface, as shown in the figure above.

58、Insert the USB drive and click the [Select Path] button.

59、Select the folder in the pop-up window where you want to save the screenshot.

60、Click the Save button to complete the modification.

Screen brightness



Click the [Modify] button in Screen Brightness to expand the configuration interface, as shown in the figure above.

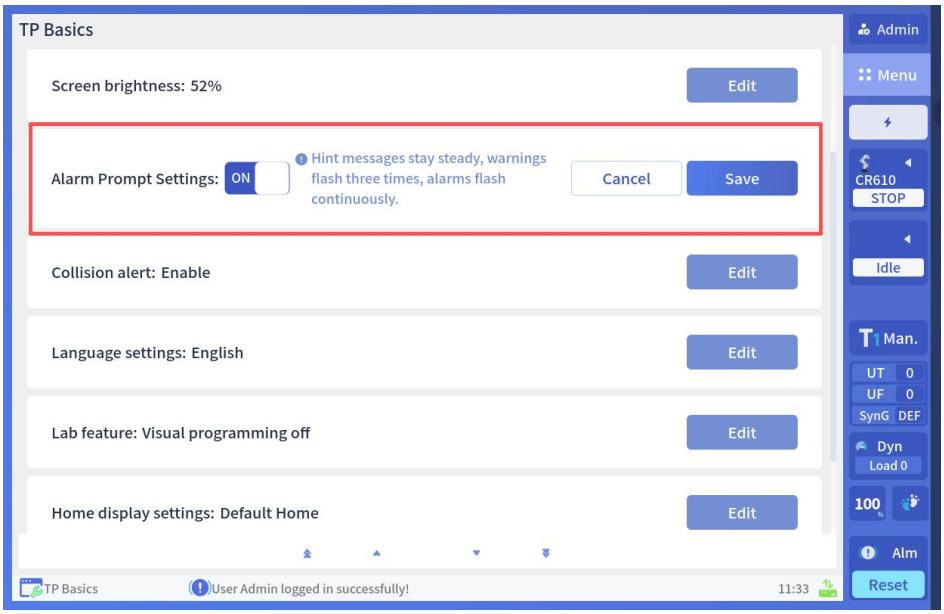
61、Drag the [Screen Brightness] slider to adjust the brightness. Click [Save] to apply the changes and close the page.

Alert configuration switch

The alarm indicator flashes the core operation interface border (the red border pointed by the arrow in the figure above) when the control system

generates a warning or alarm to enhance the user's awareness of the warning or alarm. When the switch is turned on:

- When the system triggers an alarm, the core operation interface border will continuously flash red.
- When the system issues a warning, the core operation interface border flashes three times in yellow.

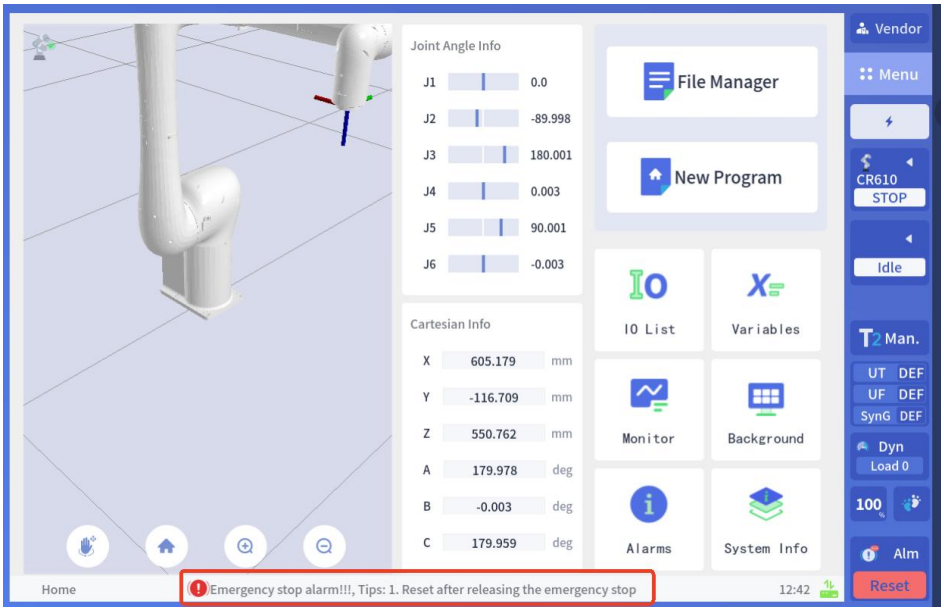


operating steps ：

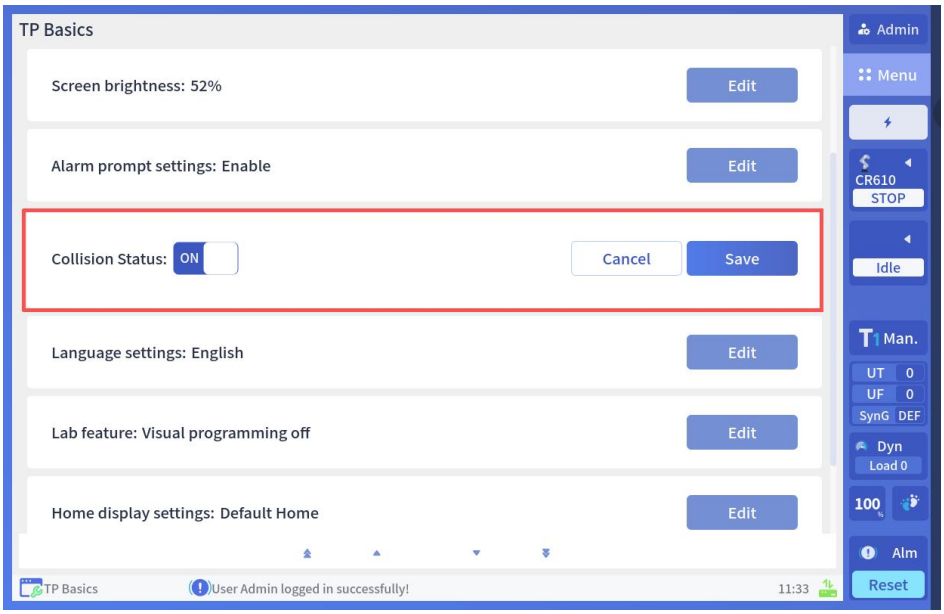
62、Click the [Modify] button in the alarm prompt configuration to expand the configuration interface shown in the figure above.

63、Click [Alarm Notification Configuration] to turn the switch on or off. Click [Save] to apply the changes, or click [Cancel] to discard them and hide the configuration page.

Collision detection prompt switch



After enabling this feature, if the system detects a collision detection alarm, an icon and text prompt will appear in the upper left corner of the 3D visualization window to enhance the user's awareness of the robot's collision status, as shown in the figure above.



operating steps ：

64、Click the [Modify] button in the collision detection status to expand the configuration interface shown above.

65、Click the Collision Detection Status toggle button to turn collision detection on or off.

66、Click Save to complete the changes.

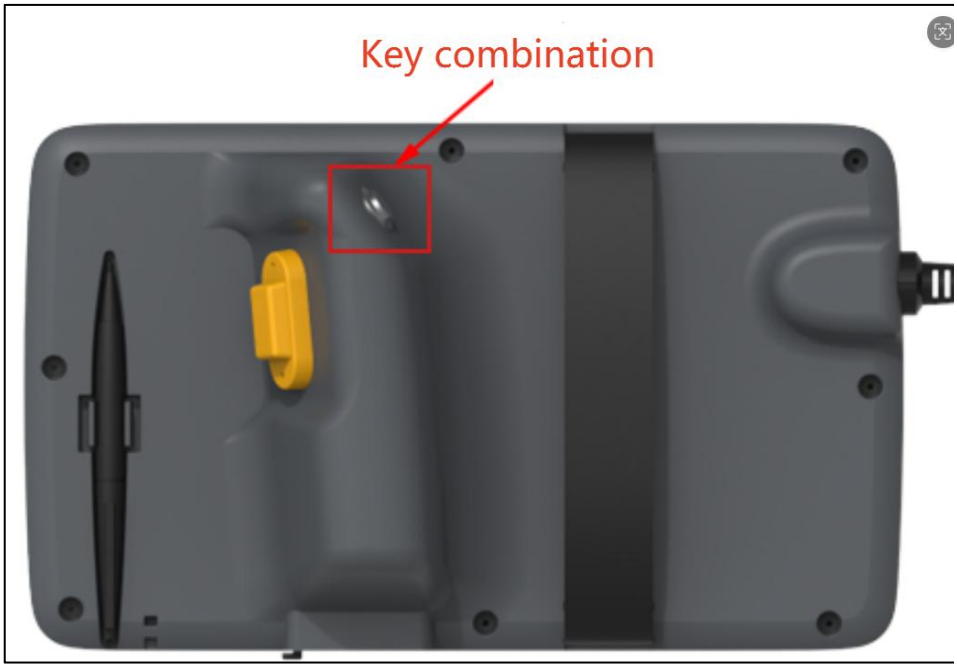
Note that this function requires the following prerequisites. Otherwise, it is invalid:

- The current device supports 3D model display.

- The current model supports and has enabled Dynamics.
- Object friction recognition is complete.

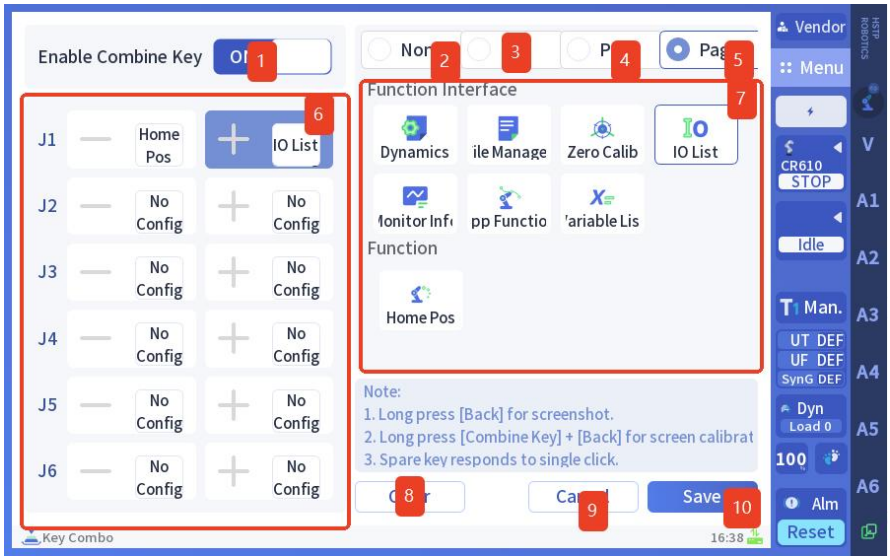
4.10.2 Combo key settings

Introduction



The combination button function can be activated by first pressing the physical [Combination] button and then the [Combination Function] button to trigger pre-configured functions. This feature expands the physical buttons of the display device, significantly enhancing operational convenience.

Before using the combination key function, configure the specific behavior of the combination key through the [Combination Key Configuration] page of the trainer. The operation interface is shown in the following figure.



Label	explain
1	Combo function button switch Open-Enable combo key function Disable-Do not enable the combination key function (regardless of configuration) Note: The combined function of the home button and back button remains unaffected.
2	Clear the selected [Function] buttons, resetting them to default configuration.
3	Set the IO mapping for the selected [Function] button.
4	The selected [Function] button sets the process package page transition.
5	The selected [Function] button redirects to the function settings page.
6	Click to select the [Function Combination] button to configure.
7	Function configuration page.
8	Clear all configuration settings for the [Pulse Motion] function keys.
9	Undo changes.
10	Save your changes.

The shortcut key action can be configured to: switch the value of a specified DO, open a specified process package, open a specified function page, or execute a specific function.

Path

➢ [Menu] → [Function Configuration] → [Teach-Button Configuration]: [Combination Button Configuration]

IO signal Configure steps

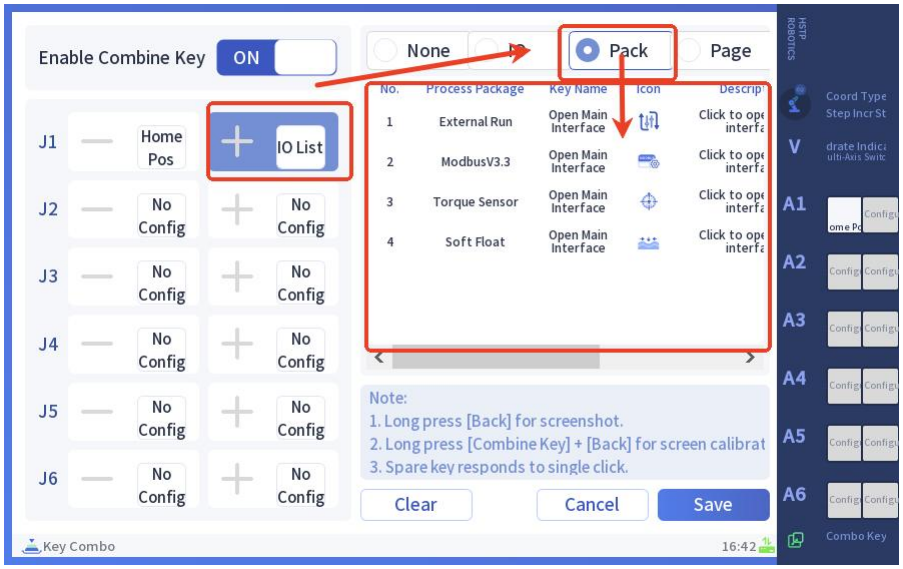
67、In the left list, select the button to configure, then click the [IO Signal] radio button to access the detailed settings page, as shown in the figure above.

68、Enter the IO index (as a DO index);

69、Click [Save] to complete the configuration, or click [Cancel] to cancel.

Process Package Configure steps

Select the [Function Combination] button, then click the [Process Package] checkbox to access the process package configuration page.

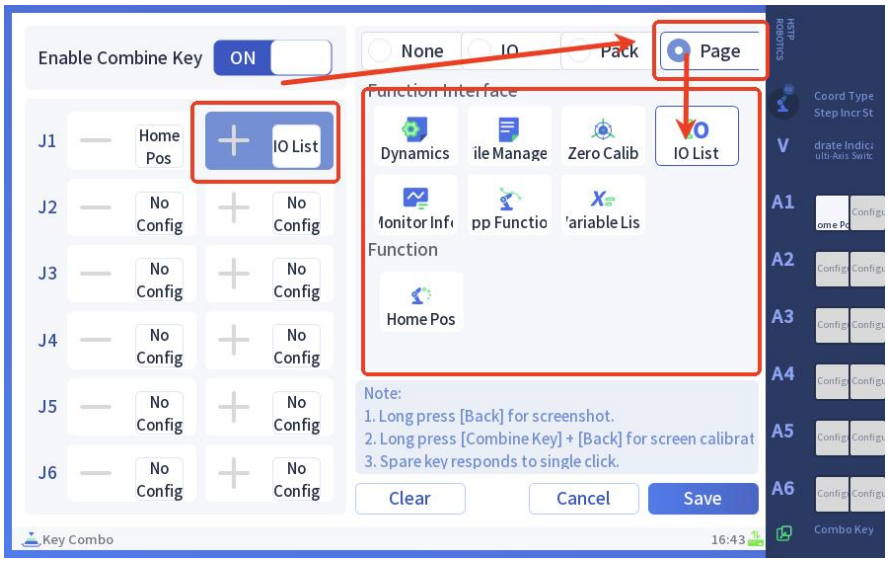


70、In the left list, select the button to configure, then click the [Process Package] radio button to access the detailed settings page, as shown in the figure above.

71、Select the process package you want to open from the list using the corresponding shortcut key.

72、Click [Save] to complete the configuration, or click [Cancel] to cancel.

Function Page Configure steps



73、In the left list, select the button to configure, then click the [Function Page] radio button to access the detailed settings page, as shown in the figure above.

74、In the right-hand interface, select the function page or action you want to open or perform using the corresponding shortcut key.

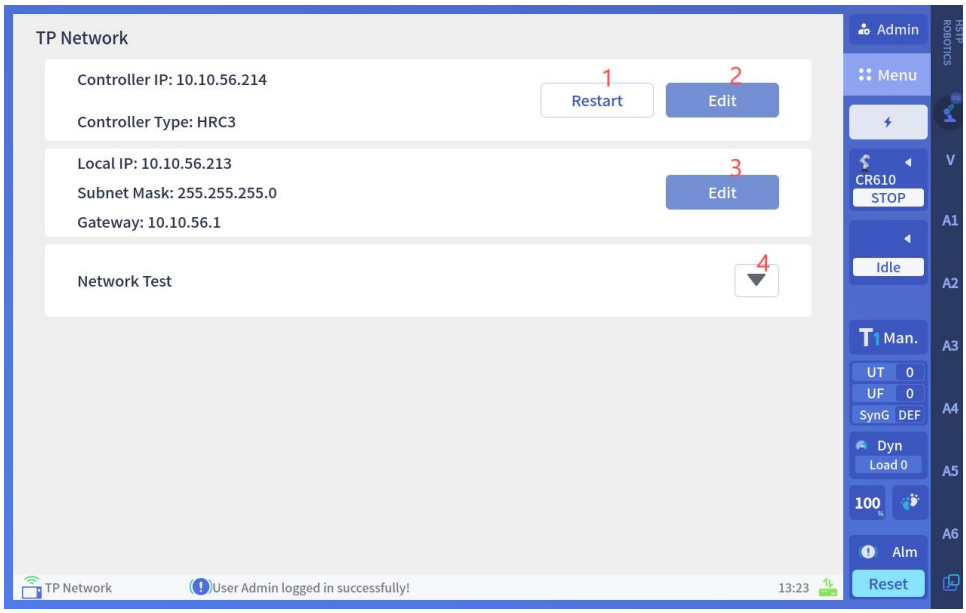
75、Click [Save] to complete the configuration, or click [Cancel] to cancel.

4.10.3 Teaching assistant communication configuration

Introduction

The teaching device and controller communicate via a Socket connection, with the controller acting as the server and the teaching device as the client. To establish a connection, the teaching device must be statically configured with its IP address, while the controller (server) requires configuration of its IP address and port number.

The teaching device configuration provides the following features: modifying the local IP address, adjusting the controller IP and port settings, and performing network tests by pinging other devices to facilitate network connection debugging.

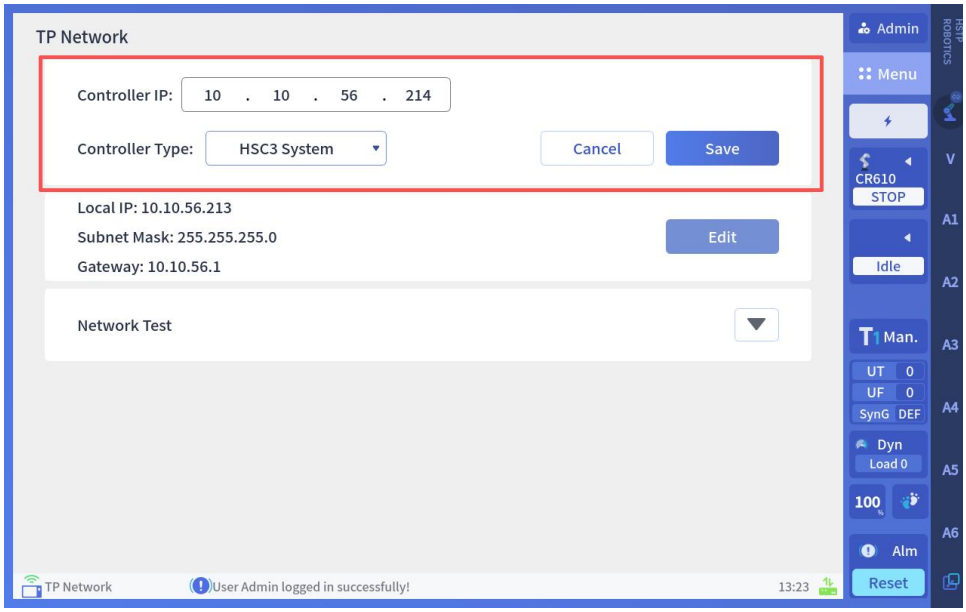


Label	explain
1	Reinforcement Teaching Tool Software
2	Change the controller IP address and port number
3	Modify the local IP address of the teaching device
4	Expand and collapse the network test page

Path

➢ [Menu] → [Function Configuration] → [Teach-Button Configuration]: [Teach-Button Communication Configuration]

Change the controller IP address and port



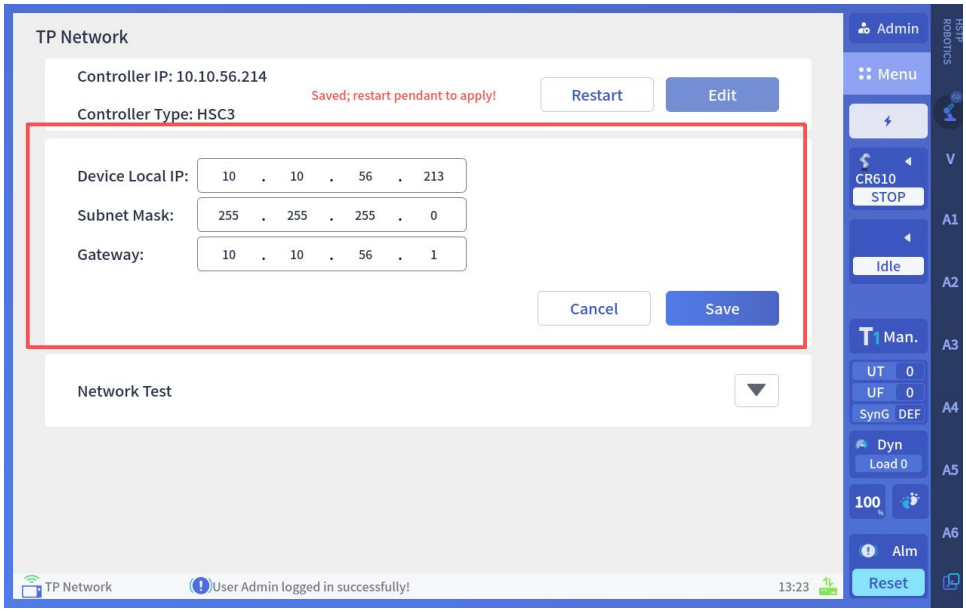
76、Click the [Edit] button to expand the configuration page.

77、Enter the IP address and port number in sequence. The IP address should match the network port of the controller connected to the teaching device. If a router is used, use the router's configuration for the corresponding port. The default port number is [23234].

78、Click the [Save] button. The system will confirm the save and restart the teaching device software.

79、Click [Re-Enable Teaching Tool] to quickly re-enable the software.

Modify the local IP address of the teaching device



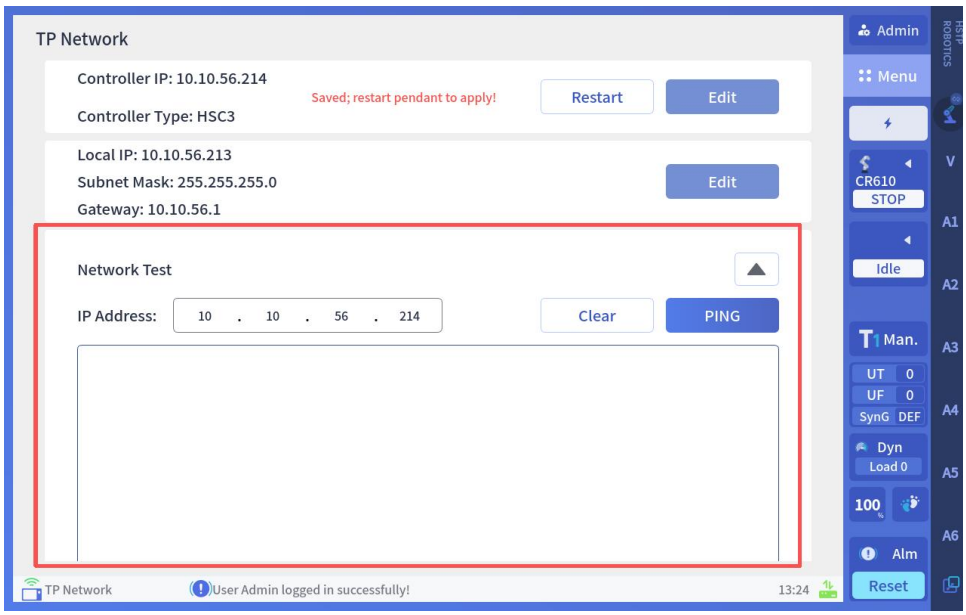
80、Click the [Edit] button to expand the edit page.

81、Enter the IP address, subnet mask, and default gateway in order.

82、Click Save to hide the configuration page.

Network test

The network test function uses the Ping command to check network connectivity. Click the [] button to open the network test page.



operating steps ：

83、Enter the IP address to test the connection in the input box.

84、Click the [PING] button to display the ping information in the [Information Box] below.

85、Click the [PING] button once to execute the Ping command.

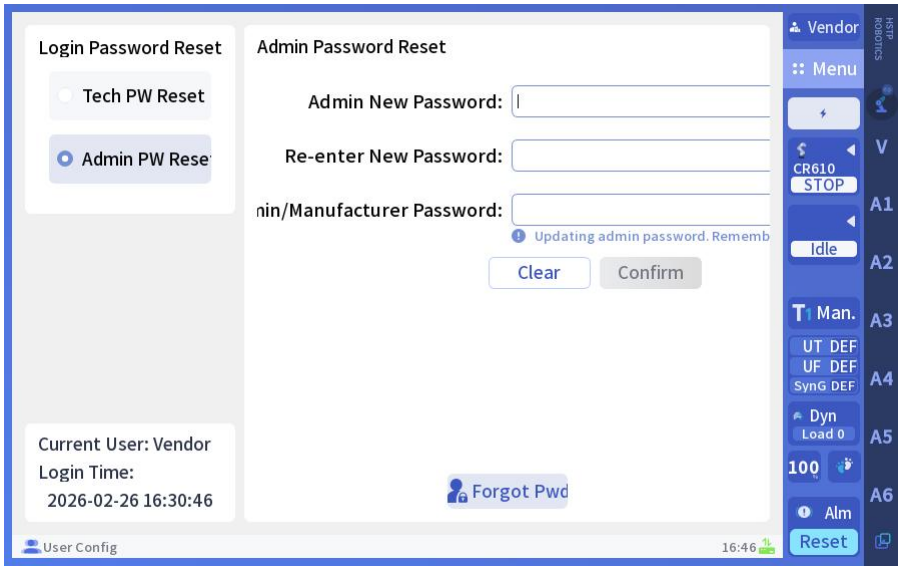
86、Click the Clear button to clear the Ping output.

4.10.4 User permission settings

Introduction

The user configuration provides the function of password modification. Currently, the software has 4 types of permissions, which are divided from small to low, including production staff, debugging staff, administrator, and manufacturer.

When the device is factory-fabricated, all users except the production staff have login passwords. If a user does not want to use the default password for security reasons and needs to change it, they can change the login password of the debugger or administrator.



Path

➢ [Menu] → [Function Configuration] → [Teach-Button Configuration]: [User Configuration]

Debug Change password

87、Enter the new password and confirm it.

88、Enter the current administrator or vendor password to verify.

89、Click the [Confirm] button to complete the change.

90、Click the Clear button to clear the input field.

conservator Change password

91、Enter the new password and confirm it.

92、Enter the current administrator or vendor password to verify.

93、Click the [Confirm] button to complete the change.

94、Click the Clear button to clear the input field.

conservator Reset password

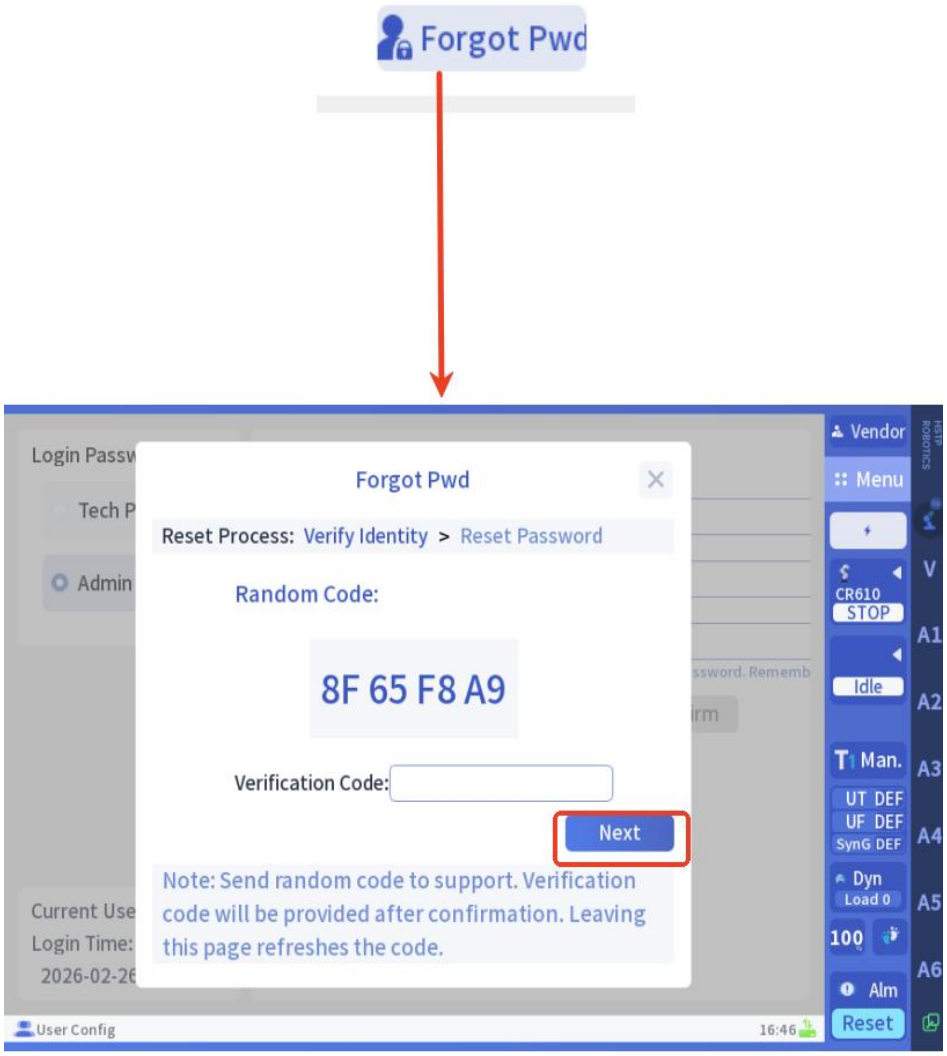
If you forget the administrator password, you can reset it by following these steps:

95、Click the [Forgot Password] button in the administrator password reset interface to display the password reset pop-up window.

96、Follow the instructions: Send the random code from the password reminder pop-up to our technical support or service center staff for verification. After receiving the code, enter it and click [Next].

97、After successful verification, you will be redirected to the password reset page. Enter the new password and confirm it.

98、Click the [Save] button to reset your password.



point out ：

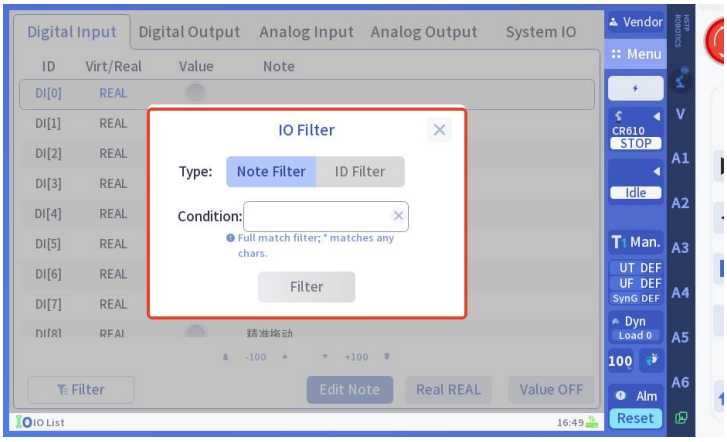
- The password recovery code will be updated with a random code after you exit. Please contact our technical support or service center staff in advance.

4.11 Display information

4.11.1 IO tabulation

Introduction The IO list displays the current input/output status. The system IO is categorized into digital and analog types, with reserved digital IO allocated for system needs.

The [Filter] and [Edit Description] buttons are common features on all tabs.

Label	explain
1	Filter button. Filter the IO list of the current tab for numbering or description. After clicking, a filter pop-up will appear, as shown ir the image.  It includes type options and filter condition input boxes. Select the type to filter and enter the filter condition. Then click the [Filter] button in the pop-up window. Click the [Filter] button to filter the IO list in the current tab. After the filter operation is completed, if no items

	meet the filter criteria, a pop-up window will display "No data under the set filter conditions!" and no further actions will be taken. If there are list items that meet the filter criteria, the list will hide items that do not meet the criteria and retain only those that do. The [Filter] button will then display as [Clear Filter].
2	[Edit] button. Adds a description to selected items in the IO list of the current tab. You need the administrator or higher permissions. After clicking, a pop-up window appears for entering descriptions. Enter the description and click [Confirm] to add it to the selected items in the IO list of the current tab. If the selected list item already has a description, the input in the pop-up window will display it for modification.

Path

- Click the [IO List] button on the home page to enter.
- Access via Menu → Display Info → Register: IO List.

point out ：

- The IO occupied by external runtime configuration is fixed based on the external runtime configuration flag. If you select the IO occupied by external runtime configuration and click the [Modify Description] button, a pop-up window will display the message: "The IO description corresponding to external runtime cannot be changed."

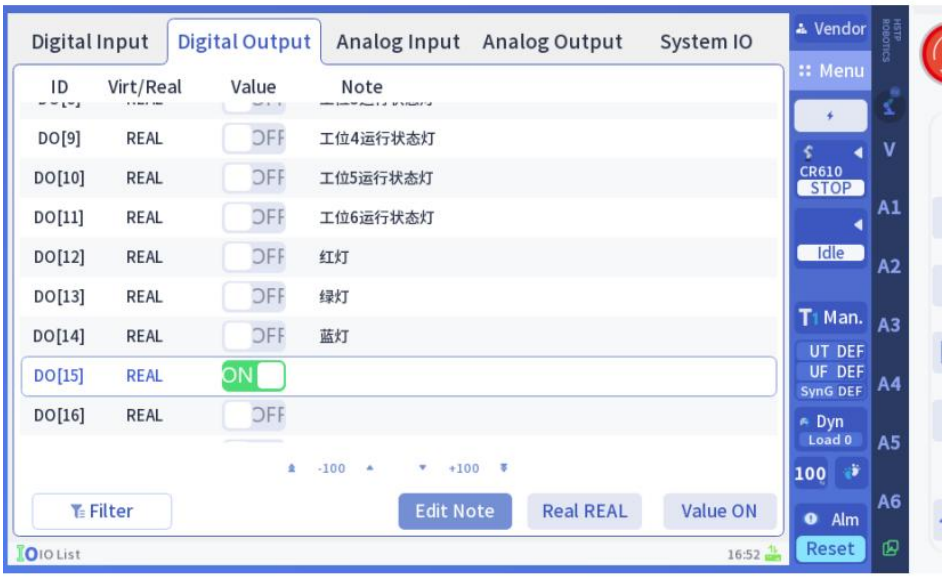
4.11.1.1 Figure IO

Introduction

The digital input/output (DIO) has 512 ports, but some are reserved for the system and assigned to the system IO tab.

Digital IO has both real and virtual states, which can be switched via the [State] button.

You can control the value of the IO variable with the [Value] button.



Select a list item in Digital IO. The Status and Value buttons change based on the item's status and value, as shown in the figure below.

Clicking the [Status] and [Value] buttons on a selected list item switches its IO status and value. For example, selecting an item in the diagram changes its status from "VIRTUAL" to "REAL" and its value from "OFF" to "ON".

The "Value" column in the list uses intuitive patterns. The patterns used for numeric input and numeric output are different and represent the following meanings:

-  Indicates that the digital input is currently OFF.
-  Indicates that the digital input is currently ON.
- 

Indicates that the digital output is currently OFF.
- 

Indicates that the digital output is currently ON.

Click the pattern listed above to achieve the same function as the [Value] button, which switches the value of the list item to which the pattern belongs.

---

point out ：

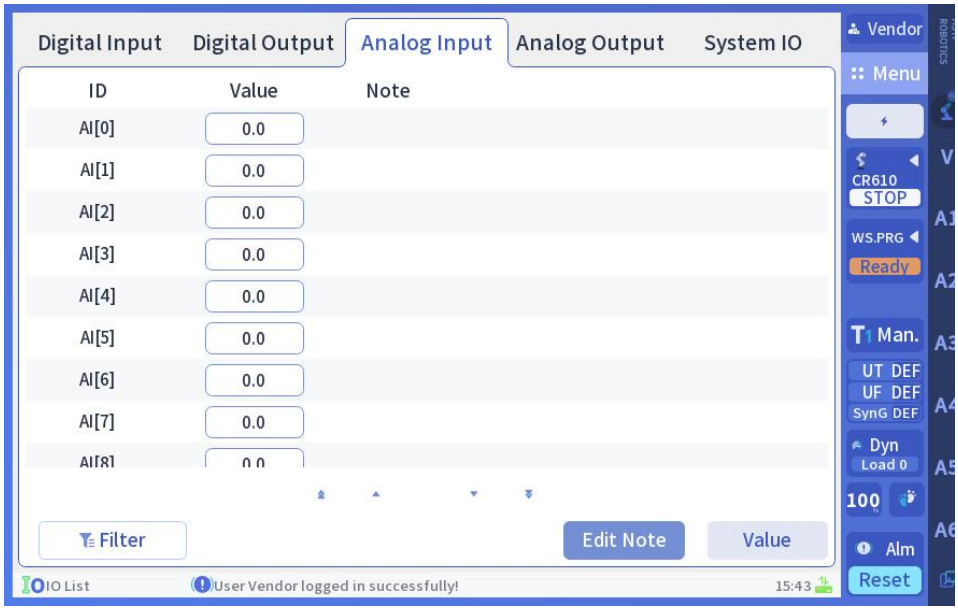
- Real input cannot be changed by teaching software.

4.11.1.2 Analog quantity IO

Introduction

The analog input and output (AIO) each have 16 channels.

Analog I/O does not distinguish between real and virtual states, so there is no [Status] button. The analog I/O values are set by the system's voltage levels. Check the analog I/O values to obtain the current status of the system's analog inputs and outputs.



Only analog output values can be modified by the teaching software.

Select an item in the list and click the [Value] button to display the AIO value input dialog. Enter the desired value within the range [-24.0, 24.0]. Click [OK] to update the analog output value.

You can also enable the editing state by double-clicking the input box in the "Value" column of the analog output list, and modify the value to update the corresponding list item.

4.11.1.3 System IO

Introduction

System IO is reserved for the system. Although it can be modified, users are not advised to do so.

The system IO list contains input and output entries with numbers, requiring a "IO type" column to specify the input/output type for each entry. The rest of the list is similar to a digital IO list.

The system uses digital inputs from DI[100] to DI[129].

The system occupies digital outputs from DO[100] to DO[129] and DO[200] to DO[229].

4.11.2 Variable list

Introduction

The variable list contains different register variables, as follows:

R: numeric register.

SR: String register.

JR: Joint coordinate register.

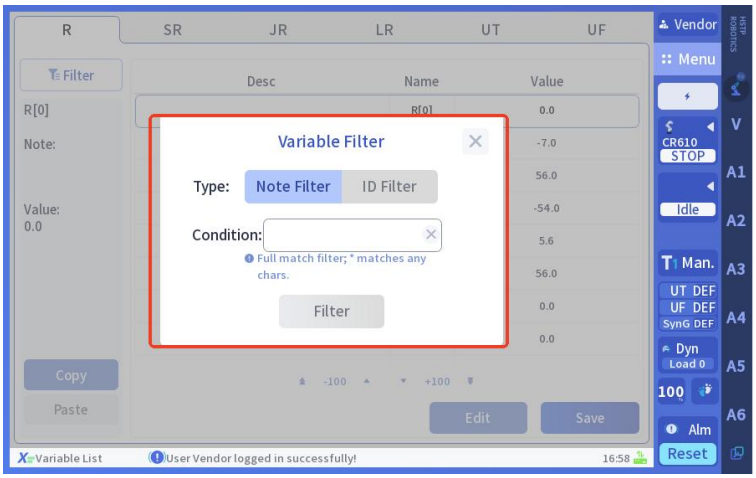
LR: Cartesian coordinate register.

UT: Tool coordinate system register.

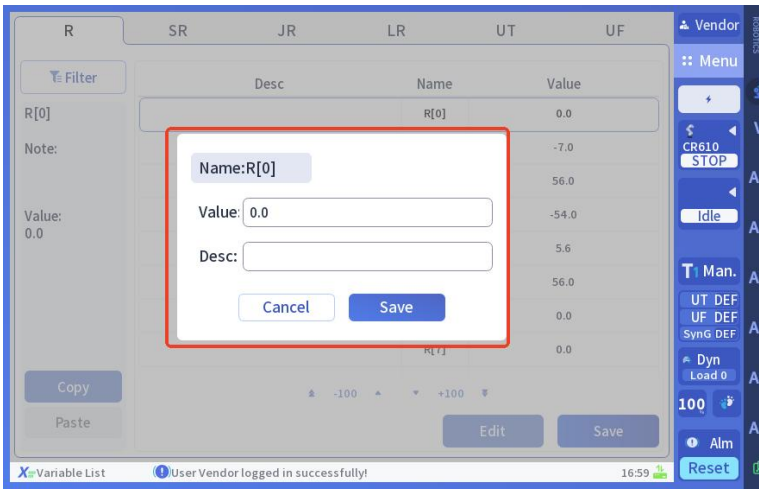
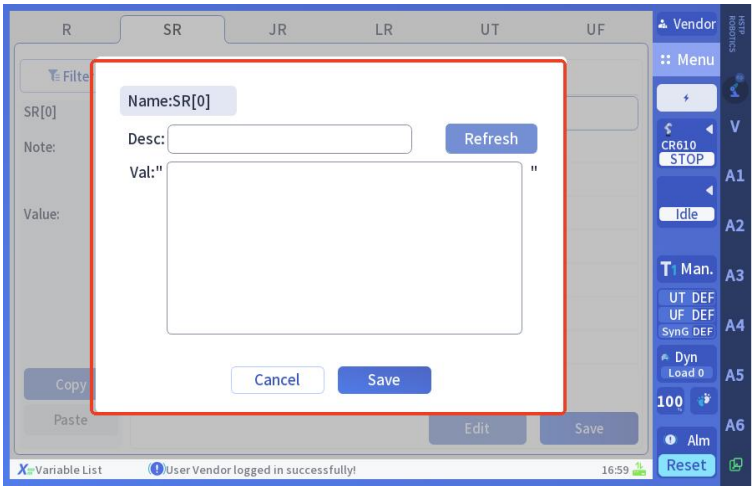
UF: Base coordinate system register.

These different types of register variables are global variables set by the system. They have their own characteristics, but most of

their functions are the same, except for the different functions of the modification pop-up window. The specific functions are shown in the following figure:

Label	explain
1	Filter button. Filters the variable list for the current tab to number or label it. After clicking, a filter pop-up will appear, as shown in the image.  It includes type options and filter condition input boxes. Select the type to filter and enter the filter condition. Then click the [Filter] button in the pop-up window. Click the [Filter] button to filter the variable list in the

	current tab. After the filter operation is completed, if no items meet the filter criteria, a pop-up window will display "No data under the set filter conditions!" and no further actions will be taken. If there are list items that meet the filter criteria, the list will hide items that do not meet the criteria and retain only those that do. The [Filter] button will then display as [Clear Filter].
2	Copy button. Copies selected items from the variable list in the current tab. You need the administrator or higher permissions. Clicking updates only the copied variable value object and displays a bubble message "Copied successfully!" until you click the [Copy] button again. Use with the [Paste] button. If the list item is not selected, the button appears gray and is not clickable.
3	Paste button. Copies the variable value object for the current tab list. You need the administrator or higher permissions. Select the list item to paste. Click the [Paste] button, and a confirmation pop-up will appear. Click [Confirm] to paste the copied variable value into the selected list item. Supports double-click paste for the same copy variable value object. If no variable value object is copied, the button appears gray and is not clickable. Note: Variable descriptions cannot be copied and pasted.

4	Modify button. Modifies selected items in the current tab list. You need the administrator or higher permissions. Clicking will open a modification pop-up window. The [Modify] button displays different pop-up windows depending on the register variable type, but all include variable descriptions and value configuration. The pop-up window for the R register variable is shown below:  You only need to configure the variable values and descriptions. The pop-up window for the SR register variable is shown below:  In contrast, an additional [Refresh] button has been added, which updates the current variable values

	and descriptions to match those in the system. The variable value input field has also been upgraded to support multi-line character input. The register variables JR, LR, UT, and UF store coordinates. The coordinate modification pop-up window is the feature modification interface. For detailed configuration, refer to the coordinate modification section in the common pop-up window documentation. After completing the data configuration for the modified pop-up, click the [Save] button to save the updated data to the system's corresponding registry and files.
5	Save button. Saves unsaved registry variables from the current tab list to system files.

Path

- Click the [Variable List] button on the home page to enter.
- Access the menu → Display Info → Register: Variable List.

point out ：

- The program may generate unsaved changes to register variables. If you need to save them, click the [Save] button manually.

4.11.3 Monitoring information

Introduction

The monitoring page integrates the required information during the operation of the robotic arm and contains a lot of display information, such as IO, variables, 3D view, etc. Users can easily and quickly monitor the important information during the operation of the robotic arm and confirm the normal operation of the robotic arm.

The monitoring information page is shown in the following figure:

Label	explain
1	Alert bar. Displays alarm information.
2	reset button 。 To perform a reset operation. After clicking, the alarm information will be cleared and the error status of the robotic arm will be removed.
3	Connection status control. Displays the connection status between the teaching pendant and the robotic arm controller. ● The red status icon indicates that the robotic arm controller is not connected. ● The yellow status icon indicates a successful network connection, but the robotic arm controller is not ready, and the teaching device cannot perform related operations on the robotic arm. ● The green status icon indicates successful network

	connection and controller initialization, enabling the teaching pendant to perform operations on the robotic arm.
4	Enable the status control. Displays the current enabled status of the robotic arm. ●  : indicates that the arm is not enabled. The arm cannot move. ●  : indicates that the arm is enabled and can move.
5	Operation mode display control. Displays the current robotic arm operation mode.
6	Motion speed display control. Displays the current speed of the robotic arm movement.
7	3D view. For intuitive display of the current position and posture of the robotic arm. For details, see the 3D view section.
8	Register variable display control. Displays the value of the register variable for user monitoring. The fixed display and monitoring R[0]~R[15] and LR[0]~LR[15] are currently active.
9	Process status display control. Displays the name and status of the current loader. The program  name marked by an icon indicates a loaded  main program, while the program name marked by an icon indicates a loaded background program. The label after the program name indicates the status of the program. For example, all programs in the figure are in the "ready"

	status.
10	Program runtime display control. Displays and records the total runtime of the current program. Note: This time starts recording when any program loads and resets when no program is loaded.
11	Display the runtime control on startup. Displays and records the teaching device's running time. The teaching device starts recording when it is turned on and all modules are loaded, and resets when it is turned off.
12	Digital output display control. Displays the value of the digital output (DO) for user monitoring. The included icons mean the following: ●  Indicates that the digital output is currently OFF. ●  Indicates that the digital output is currently ON. The icon meaning in the numeric input display control is consistent with this. DO[0] to DO[16] are currently fixed for display and monitoring.
13	Digital input display control. Displays the value of a digital input (DI) for user monitoring. The meanings of the included icons are shown in the explanation of the numeric output display control above. The fixed display and monitoring are DI[0] to DI[16].
14	Program content display control. Displays loaded program content without editing the program.

	The program name appears in the top-left corner, as shown in the "A112A.PRG" in the figure. If the current program is a background program, a [*] is added after the program name to indicate it, for example, "XXXX.PRG[*]". In manual swipe mode, you can view the full program content by dragging or using the bottom [][] button.
15	Program run identifier. A way to identify the current program's running instructions and the instructions to which you can go back. When the program is in the loading state, an arrow icon appears to the right of the line number. Indicates the line of the program to run, and the line to which to revert when backing up.
16	[Auto Slide] Button Switch between auto-swipe and manual swipe. Tap to switch between auto-swipe and manual swipe. When in auto-scroll mode, the button displays "Auto-Scroll" and the program content control follows the current line number, keeping it within the visible area. If the current line number goes out of view, it automatically scrolls to the line number to maintain visibility. When in manual scrolling mode, the button displays "Manual Scroll". The program content no longer follows the current line number of the running command. Users can scroll through the program content as needed.
17	[Back to top] button. Returns quickly to the top of the program content.

Path

- Go to the Monitoring Information button on the home page.
- Access through the [Monitor] button on the Program Files Manager page.
- From Menu → Display Information → Monitor Information: Monitor Information Entry.

4.12 Help

4.12.1 Display information

4.12.1.1 System info

Introduction View the system version information. The interface is shown in the following figure.

Label	explain
1	Version information options page. Users can upgrade and authorize the system and software in the version information.
2	Current system time of the connected controller
3	Current device nameplate number
4	Host computer software version information
5	System version information

Path

- Menu bar → Help and Manual → Help: System Information
- Home> Common Functions> System Information

4.12.1.2 System authorization

Introduction

The control system requires registration and authorization before use. Follow the software interface prompts to complete the authorization process, as shown in the figure below.

Label	explain
1	System authorization tab.
2	Static description.
3	SN code information text display box.
4	Click the [Get SN Code] button to save the device's SN code to a sn.txt file. Then send the file to the manufacturer for authorization.
5	Click the [Select File] button to choose the genAuthCode registration authorization file provided by the manufacturer.
6	Click the [Authorize] button to power off and restart the device to complete authorization.

Path

➢ Menu bar → Help and Manual → Help: System Information → System Authorization tab

operating steps

99、Insert the USB drive.

100、Click the [Get SN Code] button to retrieve the controller's SN code and save it to the sn.txt file in the root directory of the USB drive.

101、Send the obtained sn.txt file to the technical service personnel to obtain the genAutoCode file.

102、Copy the genAutoCode file to the USB drive.

103、Reinsert the USB drive.

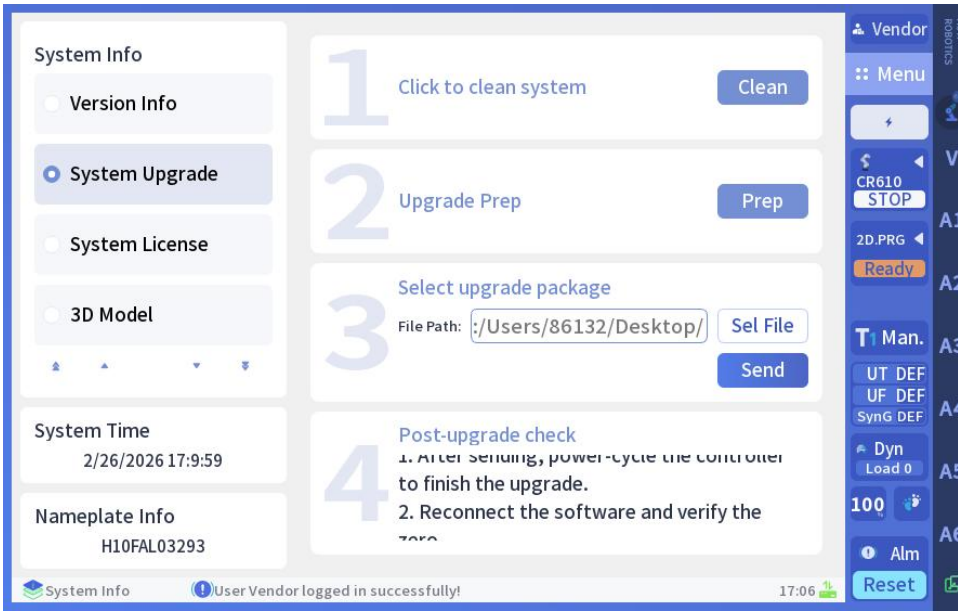
104、Click the [Select File] button and choose the genAutoCode file in the file selection dialog.

105、Click the [Authorize] button to complete authorization.

4.12.1.3 System upgrade

Introduction

The teaching pendant can be used to upgrade the control system software. The upgrade software package has the.tar.gz extension. Users can authorize the upgrade by following the steps prompted by the software interface, as shown in the figure below.



Path

➢ Menu bar → Help and Manual → Help: System Information

➢ Left-side options in system information: System version upgrade

operating steps

106、Click the Clear System button.

107、Click the Prepare Upgrade button.

108、Insert the USB drive with the upgrade package into the trainer or controller

109、Click the [Select File] button to choose the upgraded file.

110、Click the [Send update package] button.

111、Power off and restart the control system and software.

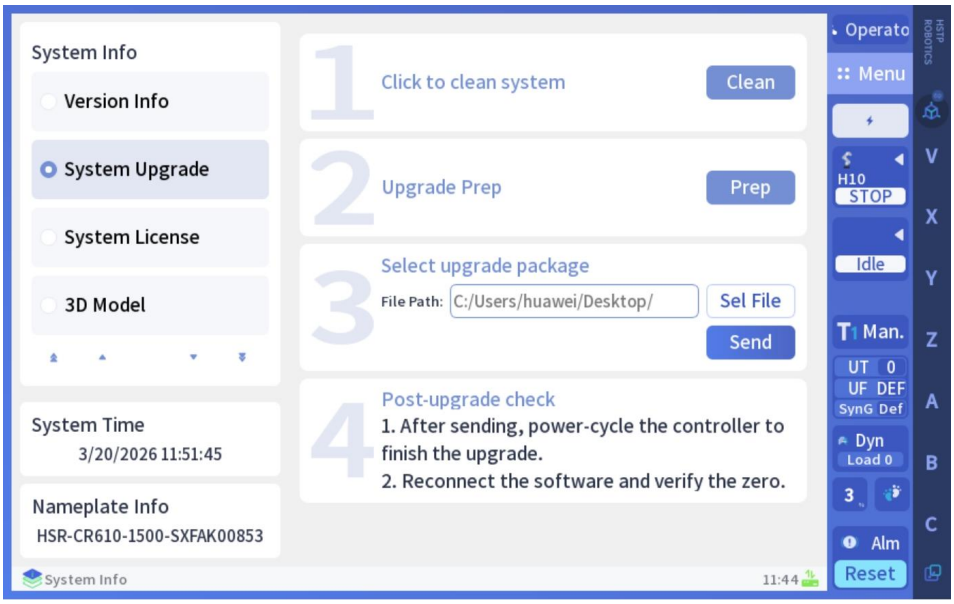
point out ：

- Before upgrading the system, return the robot to the zero position, power off and restart it once, or power it on for the first time. Do not perform any operations before executing the upgrade. First, [Clear System] to release disk space.
- If the USB drive fails to be recognized, you can uninstall the SD card from the USB storage in the teaching programmer system settings and reinstall it.

4.12.1.4 Software upgrading

Introduction

The software version of the teaching device is upgraded to provide the corresponding function interface. The software package suffix is.tar.gz. Users can authorize the operation step by step according to the prompts on the software interface.



Path

- Menu bar → Help and Manual → Help: System Information
- Left-side options in system information: software version upgrade

operating steps

112、Insert the USB drive with the upgrade package into the trainer or controller

113、Click the [Select File] button and choose the upgrade package in the pop-up file selection window.

114、Click Upgrade.

115、The interface guide software is upgrading. Do not power off.

116、The software prompts that the upgrade is successful. Restart and click the [Restart] button.

warn ：

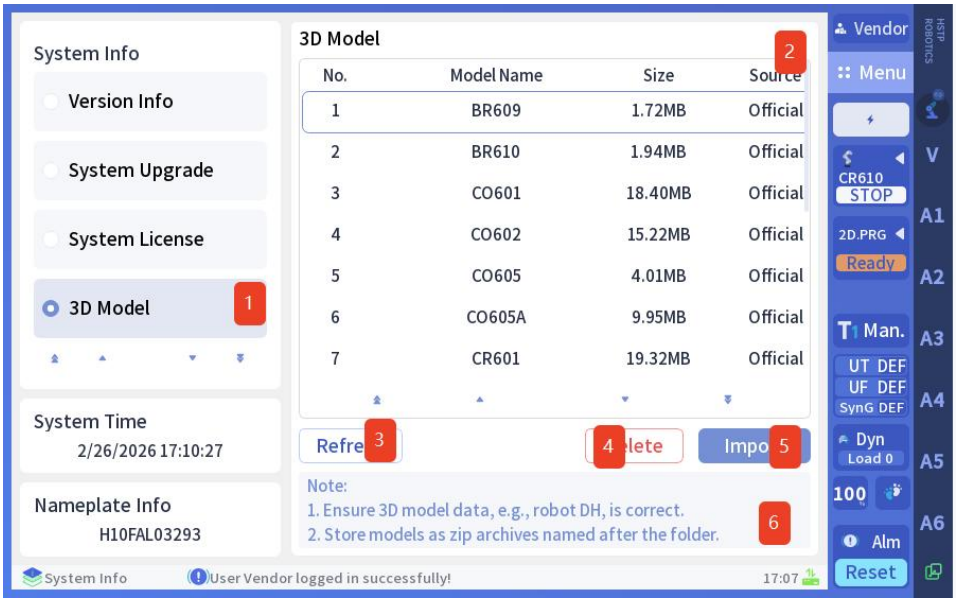
- Do not power off during the upgrade process, as it may damage the teaching device software system and cause it to malfunction.

4.12.1.5 Three-dimensional model

Introduction

To display a robot's current pose in 3D visualization controls, the teaching pendant must contain a corresponding robot model. The system provides default 3D models for common robot types. For models not included by default, users can import or manage 3D models through the [3D Model] page

in the teaching pendant. This enables real-time pose visualization. The interface is shown in the figure below:



Label	explain
1	3D Model Options Tab.
2	The list of model names built into the software.
3	Refresh the current list.
4	Delete the selected list file. Official model files cannot be deleted.
5	Import 3D model file button to open the file import pop-up.
6	Static description.

Path

- Menu bar → Help and Manual → Help: System Information
- Left options in system information: 3D model

Import 3D model

117、Click the Import button

118、Click the [Select File] button to choose the 3D model to import

119、Click the [Import Model Package] button and wait for the file import prompt.

120、Click the [Re-Enable] button to select the matching 3D model file for the device to take effect.

model building For specific production methods, please consult the technical service personnel.

point out ：

- Supports debuggers and higher-level permissions to access this interface.
- The [Delete] and [Import] functions in the interface are available only to administrators and vendors.
- The import of the model file exceeds 50 MB and will fail.
- The model package must be a ZIP-compressed file.
- The software model file of the Enlightenment Teaching Tool is only effective after it is displayed.
- If the user-imported model file name matches the model folder name in the upgrade package, the user-imported model file will be displayed first.

4.12.2 Robot parameters

Introduction Robot parameters can view the detailed system information of the robotic arm.

Path

➢ Menu Bar → Help and Manual → Help: Robot Parameters

4.13 Operate

Introduction

The menu function allows you to directly click command buttons to operate the system/software.

operation declaration

121、Revelation Teaching Device: Software functions of the Revelation Teaching Device.

122、Clear System: Clear system function buttons. Click to execute the system cleanup function.

123、Close the system: Disable the system function button. Click to disable the system function. The system can only be started after power-off and reboot.

124、Restart the system: Clear the system function button and click to restart the system.

point out ：

- All users can restart the device. Other functions [Close System], [Clean System], and [Restart System] require administrator or higher permissions.

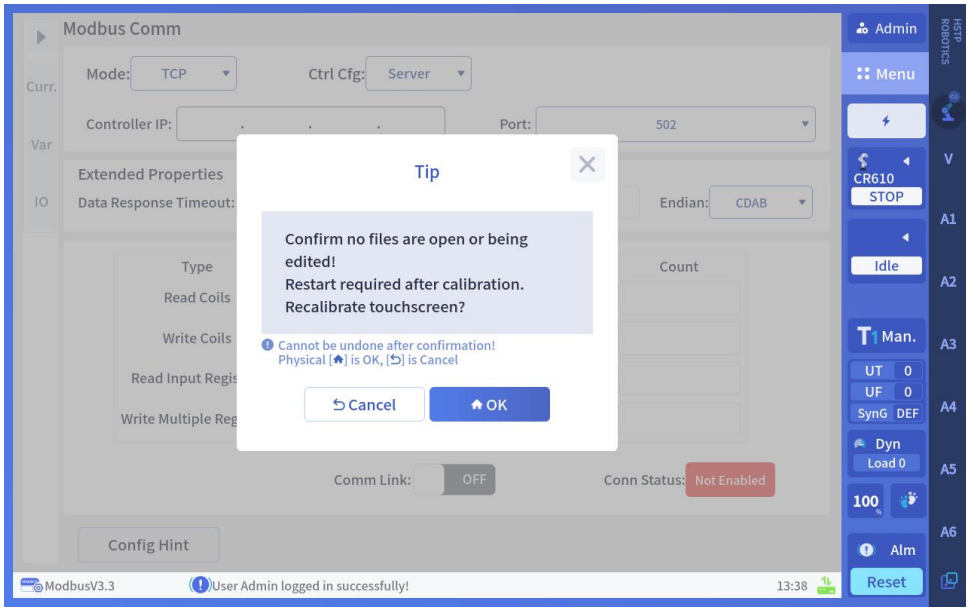
- Manual mode does not support the [Re-Enable Device] function.
- When enabled, the system does not support [Turn off system] and [Restart system].

4.14 Other functions of the teaching device

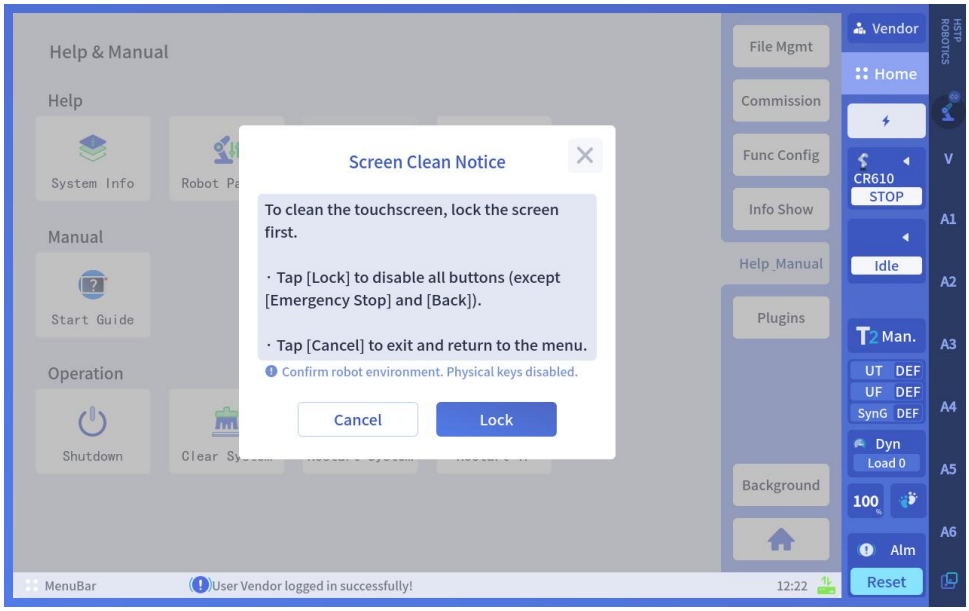
4.14.1 Screen cleaning

Introduction

To prevent users from accidentally touching the touch screen while cleaning the screen, you can use the Screen Clean function. When you start to lock the function, the screen will enter the protection state to prevent accidental operation.



When cleaning is complete, if you want to exit this mode, simply hold the designated  button for 2 seconds (or hold the [] button) to stop the screen cleaning mode and ensure safety during the cleaning process.



Path

➢ Menu bar → Help and Manual → Help: Screen Cleaning

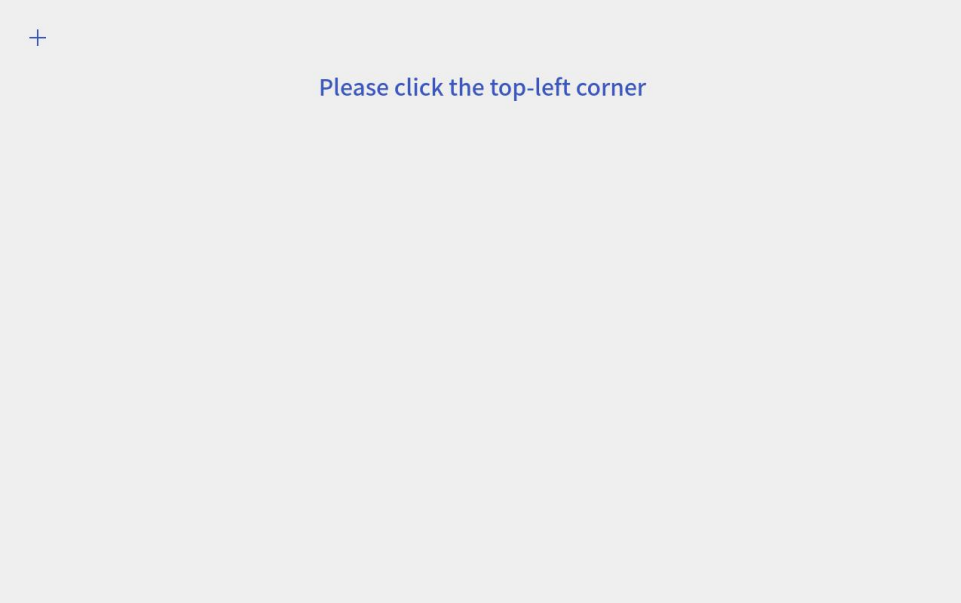
point out ：

- Entering screen cleaning mode will disable all buttons except the [Emergency Stop] and [Return] buttons.

4.14.2 Screen calibration

Introduction

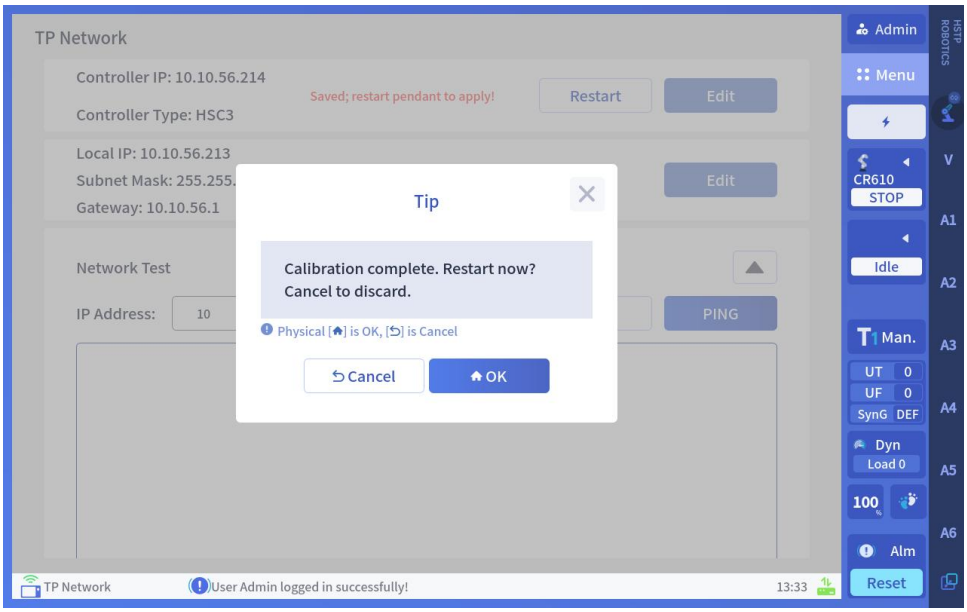
To ensure that the input position of the touch screen corresponds accurately to the display position of the screen, and to improve the user experience and accuracy, the trainer can calibrate the screen.



Path

- Menu bar → Help and Manual → Help: Screen Calibration
- Physical button operation: Press and hold the [Combo] button and the [] button on the trainer for 1s to enter the interface.

operation declaration



125、Open the calibration confirmation pop-up window via the menu or physical shortcut keys, as shown in the figure above.

126、Click the [Confirm] button in the pop-up window to enter calibration mode (if the screen is not clear, press the physical [] button to confirm).

127、Follow the on-screen instructions to calibrate the steps in sequence.

128、After calibration, a restart confirmation window will pop up. The

calibration data will take effect after restarting.

point out ：

- Environmental lighting: Calibrate in a uniform environment and avoid strong direct light.
- Clean screen: Ensure the touch screen is clean and dust-free before calibration.
- Stable operation: During touch screen calibration, keep your hands stable and avoid accidental touch.
- Save Settings: After completing all calibration steps, the system will save the new calibration settings.
- Test calibration results: Restart the app or device to test the touch screen response for greater accuracy.

4.14.3 Screenshot

Introduction	The software supports taking screenshots of the current screen interface via hardware buttons and saving them to the folder on the USB drive (default path: \Udisk\Pictures\HSTP_Pic). Users can set the save path in the trainer's basic configuration. Screenshots should be named with "HSTP_DateTimeSecond".
operating steps	129、Insert the USB drive. The bubble displays "USB drive inserted" 130、Long press the [] button to receive a message prompt

5 Plugin pack

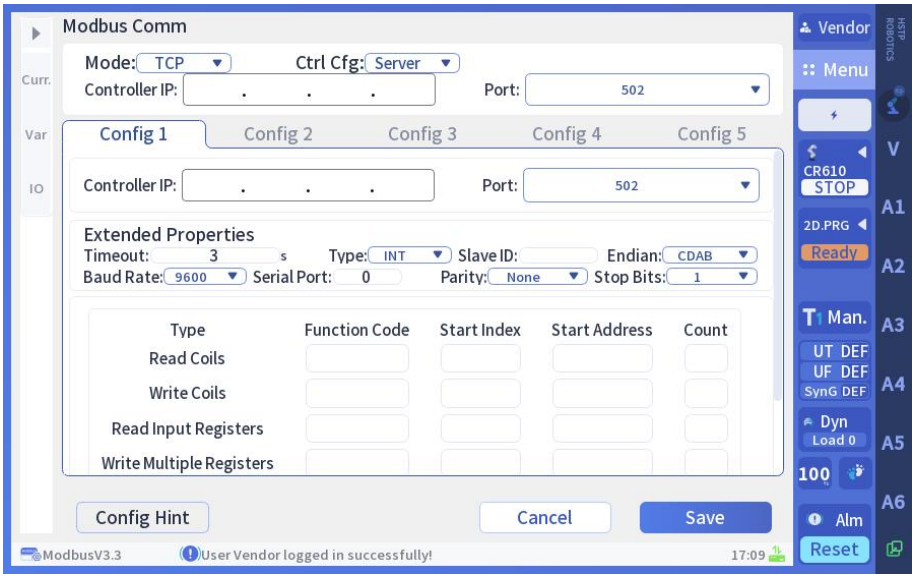
5.1 Introduction to the plugin pack

The software team currently provides the plugin package

5.1.1 Modbus communication V3.3

Introduction

The MODBUS protocol enables seamless interaction between the system's internal engineering applications and the teaching programmer interface, ensuring user-friendly operation.

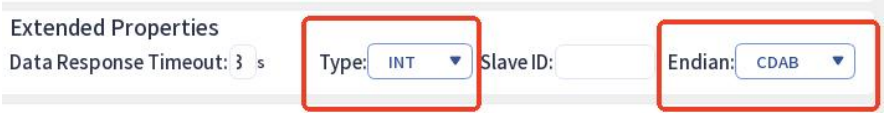


Path

1. Function location: Menu bar → Plugin package → MODBUSV3.3
2. Configure communication mode, controller type, extended properties, and register index associations. Click to enable the communication connection, save the configuration information to the controller, and start the device to initiate communication.

data type

The data types include INT (16-bit), DINT (32-bit), and REAL (32-bit).



If the data size is within the range of "-32768 to 32767", select INT. For larger sizes, use DINT. For data with decimal points, choose REAL.

Height setting

The data byte order can be configured in two formats: 'CDAB' or 'ABCD'. By default, we use 'CDAB'. Most PLC manufacturers adopt 'CDAB' as the standard byte order, while only a few German-made PLCs use 'ABCD' —for example, Siemens PLCs.

point out ：

- This plugin package is only compatible with 1.6.6_20221026 and all versions of 1.6.9.
- The latest system versions 1.6.6 and 1.6.9 support the transmission of floating-point data types and 32-bit long integers.

5.1.1.1 MODBUS TCP server

Configuration interface

Label Description

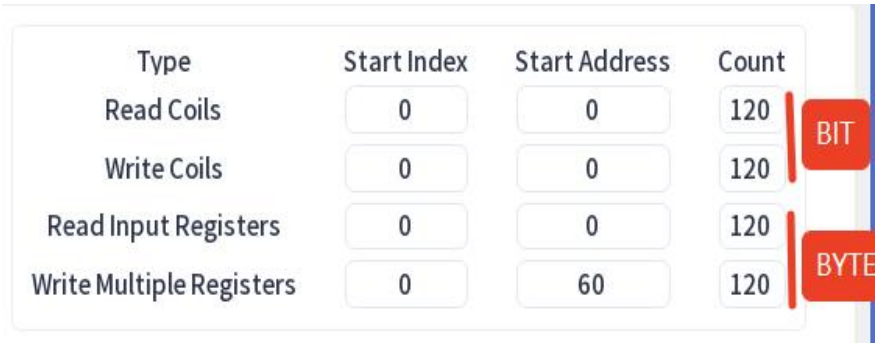
Label	explain
1	Set the communication mode to TCP
2	Controller settings can be configured as "client" or "server".
3	IP settings. If the robot acts as a server, enter the teaching device IP address. If the robot acts as a client, enter the other party's IP address.

4	Port settings. The client-server port is used for connecting the client and server. The data input rang is 502-506.
5	Request timeout. Enter a value between [1-1000]. The default is "3". If the host computer fails to send a request message within the set time, the connection will be automatically disconnected.
6	Select the register data type for read/write operation Options include: INT, DINT, and REAL.
7	Enter a server ID in the range [1,147]. The default value is "1".
8	Set the 32-bit data high and low bits. Options: CDAB and ABCD. Default value is "CDAB".
9	Read the coil status starting index. Set the index number for the first DI (digital input).
10	Set the number of read coil states. Range: 0-120.
11	Read the coil status starting address. Set the startin address for reading coil status. If you are the server, set it to "0". If you are the client, set it according to your actual needs. If no starting address is specified the default is "0".
12	Write the coil status start index. Set the index numb for the first DO (digital output).
13	Set the number of coil states. Range: 0-120.
14	Write the coil status start address. Set the start address for writing coil status. If you are the server, set it to "0". If you are the client, set it according to your actual needs. If no start address is specified, th default is "0".
15	Read the starting index of the register. Set the index number of the first R register.

16	Set the read hold register count. Range: 0-120.
17	Set the starting address of the read register. If you a a server, set it to "0". If you are a client, set it according to your actual needs. If no starting addres is specified, the default is "0".
18	Write the starting register index. Set the index numb of the first R register.
19	Set the number of write registers. Range: 0-120.
20	Write the starting address of the register. Set the starting address for the write register. If you are a server, set it to "0". If you are a client, set it accordin to your actual needs. If no starting address is specified, the default is "0".
21	For communication connection, when ON is selecte click "Confirm" to take effect.
22	Display connection status.
23	The [Cancel] button re-reads the controller's configured parameter file data.
24	The [Save] button saves the interface configuration parameters and generates a file to be saved in the system.
25	Static description. Describes the occupied IO and R registers.
26	[Function Code Configuration]. Set the following function codes: 0x02,0x04,0x0F,0x10. (Robots do not need to set function codes when serving as a server)

parameter Configuration

Modbus communication data formats are primarily categorized into two types: bits and bytes. As shown in the figure 6.1.2.1-1 below:



When the Modbus protocol packet specifies register indices and quantities as shown in the diagram above, the data transmission methods for the two data types are as follows:

- The BOOL-type data receiver captures coil status updates from 100 inputs (DI30-DI129), with internal data mapping starting at address 10.
- The BOOL-type data transmission sends coil status through 90 output lines (DO40-DO139), with internal data mapping starting from address 0.
- The INT-type data reception process involves collecting data from the 30 R registers (R50-R79) of the receiving device, with the internal data mapping starting from address 0.
- INT-type data is transmitted to the recipient via 40 R registers (R90-R129), with the internal data mapping starting at address 60.

User example

When the robot server responds to the function code data request sent by the client, the relationship between the data parameter configuration in the process package interface is shown in Figure 6.1.2.1-2 below:

- When the client transmits the function codes 01 and 0F, it performs read/write operations on the DI coil within the same segment,

meaning the DI channel becomes both readable and writable.

- When the client sends only the 01 function code request data, it reads the data from our DI; when the client sends the 02 function code, it reads the data from our DO.
- If the customer sends the 03 and 10 function codes, it means they are reading and writing to our R register for the same segment, meaning our R register is readable and writable at this time.
- When the client sends the 04 function code, it reads the data for "writing multiple registers". If the client only sends the 03 function code, it reads the data for "reading input registers".

How to use

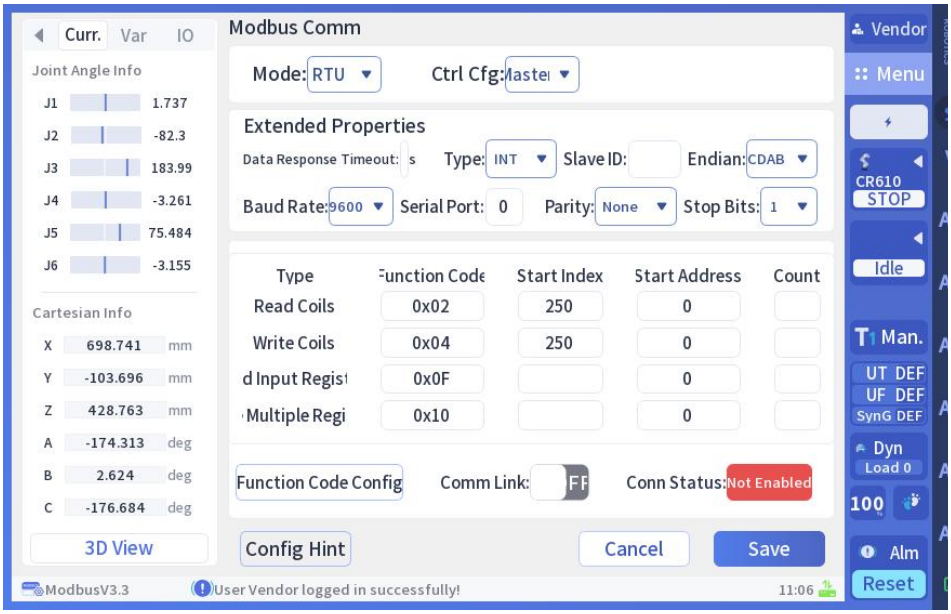
131、Communication mode: select "TCP";
132、Controller settings: select "Server";
133、Port settings: select any one of 502,503,504,505,506,507,508, or 509.
134、Data type. It can be set as INT by default or selected based on the actual application requirements.
135、Set the register index number and the number of registers. Note that if the server is the client, the number of registers must be greater than or equal to the number of registers on the client.
136、Communication connection: OFF.
137、After completing the above settings, click Save.
138、(Note: No function code is required on the server)

pay attention to ：

- When selecting DINT or REAL data types, note that these convert two 16-bit registers into a 32-bit format. As the client-side robot uses 10 read/write registers, the server must provide at least 20 registers to support this operation.

5.1.1.2 MODBUS TCP client

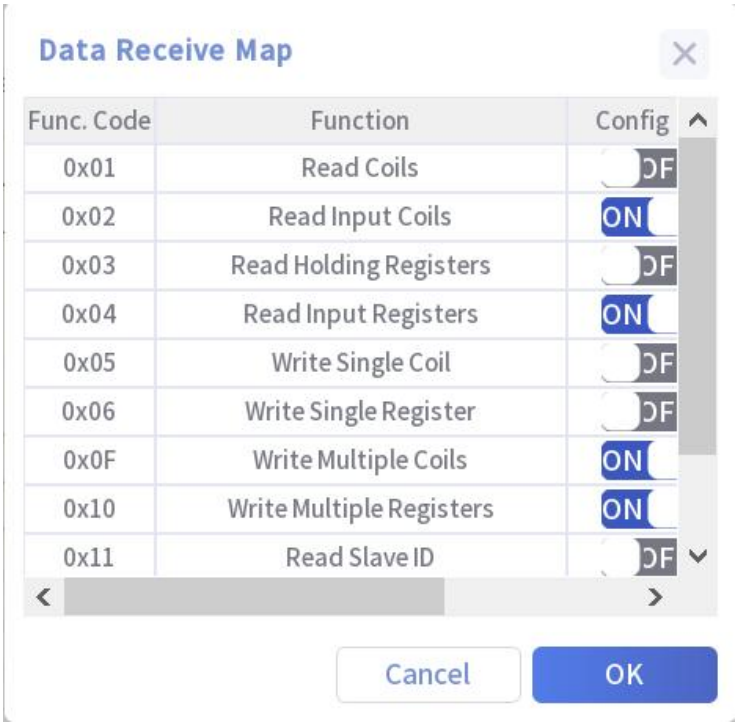
Introduction



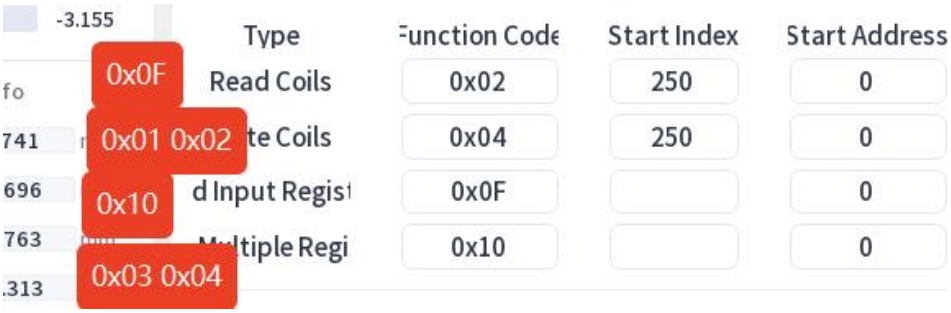
The MODBUS TCP client supports five sub-interfaces (Configuration 1, 2, 3, 4, and 5), each configured to communicate with a different server.

Function code configuration

Click the [Function Code Configuration] button to display the following interface:



When the robot acts as a client, the relationship between the function code request data sent and the parameter configuration in the process package interface is shown in Figure 6.1.2.2-1 below:



graph 6.1.2.2-1

How to use

139、Communication mode: select "TCP";

140、Controller settings: select "Client";

141、Port settings: select any one of 502,503,504,505,506,507,508, or 509.

142、Data type. It can be set as INT by default or selected based on the actual application requirements.

143、Set the register index number and the number of registers. Note that if the server is the client, the number of registers must be greater than or equal to the number of registers on the client.

144、Function code configuration. The available function codes are 01,02,03,04,0F, and 10.

145、Communication connection: OFF.

146、After completing the above settings, click Save.

point out ：

- Version 3.3 enables independent configuration of client ID, data type, and bit-level settings across five client interfaces.
- Client IDs for Configuration 1 to Configuration 5 cannot be identical. Setting the same ID will cause communication errors.
- Port numbers for Configuration 1 to Configuration 5 cannot be the same. Setting the same port number will cause communication errors.

5.1.1.3 MODBUS RTU

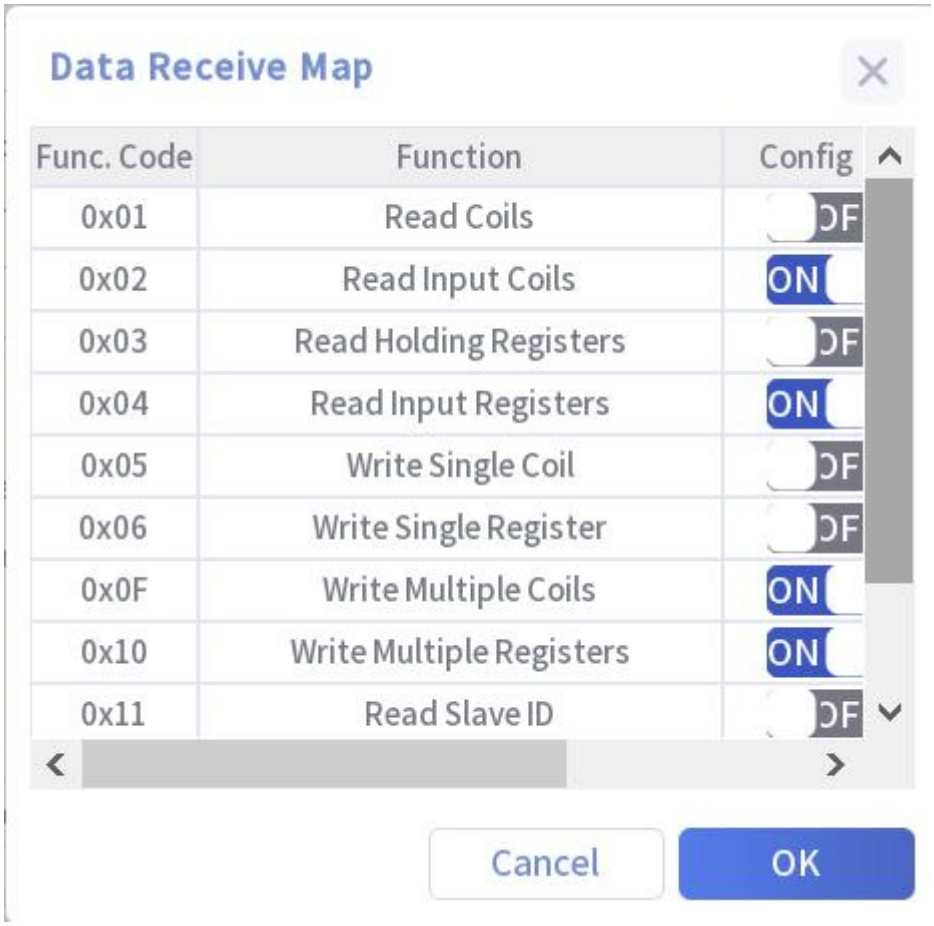
explain

Label	explain
1	Set the communication mode to RTU
2	The controller can be set as a "slave station" or "master station".
3	Request timeout. Enter a value between [1-1000]. The default is "3". If the host computer fails to send a request message within the set time, the connection will be automatically disconnected.
4	Select the register data type for read/write operations. Options include: INT, DINT, and REAL.
5	Set the 32-bit data high and low bits. Options: CDAB and ABCD. Default value is "CDAB".
6	Enter a station ID. The range is [1,147]. The default value is 1.
7	The higher the baud rate, the faster the data transmission speed. Generally, 9600 is recommended. If the field communication's baud rate requirements cannot be met, 115200 can be selected.

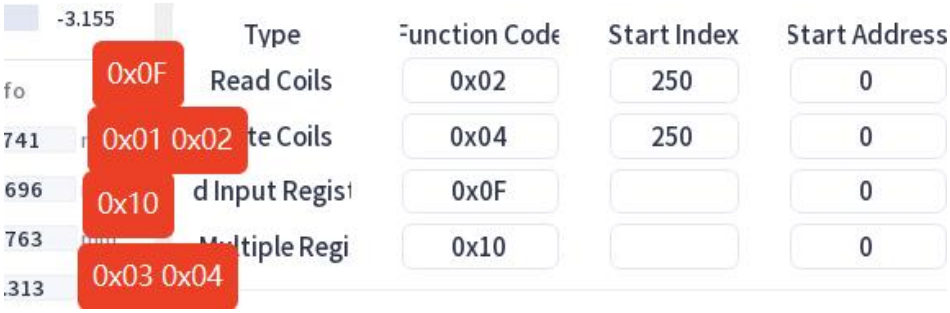
8	Serial port number. Enter the serial port number of the local device. The data input range is 0-30.
9	Check parity. Default is "none".
10	Stop position. Default is 1.
11	[Function Code Configuration]. Default function codes: 0x01,0x02,0x03,0x04,0x0F,0x10. (Do not set function codes when the robot acts as a slave station.)
12	For communication connection, when ON is selected, click "Confirm" to take effect.
13	Display connection status.
14	Static description. Describes the occupied IO and R registers.
15	The [Cancel] button re-reads the controller's configured parameter file data.
16	The [Save] button saves the interface configuration parameters and generates a file to be saved in the system.

Function code configuration

Click the [Function Code Configuration] button to display the following interface:



When the robot acts as a client, the relationship between the function code request data sent and the parameter configuration in the process package interface is shown in Figure 6.1.2.2-1 below:



How to use

147、Communication mode: select "TRU";

148、Controller settings: select "master station" or "slave station";

149、Data type. It can be set as INT by default or selected based on the actual application requirements.

150、For serial port configuration, use port 10 for HPC102 and port 0 for HPC200.

151、Set the baud rate to 9600 or 115200, no parity check, stop bit is 1;

152、Set the register index number and the number of registers. Note that if the slave station is the case, the number of registers must be greater than

or equal to the number of registers of the master station.

153、Function code configuration. The available function codes are 01,02,03,04,0F, and 10.

154、Connection: point is.

155、After setting all the above, click OK.

156、(Note: Step 7 can be skipped when creating a "from station")

point out ：

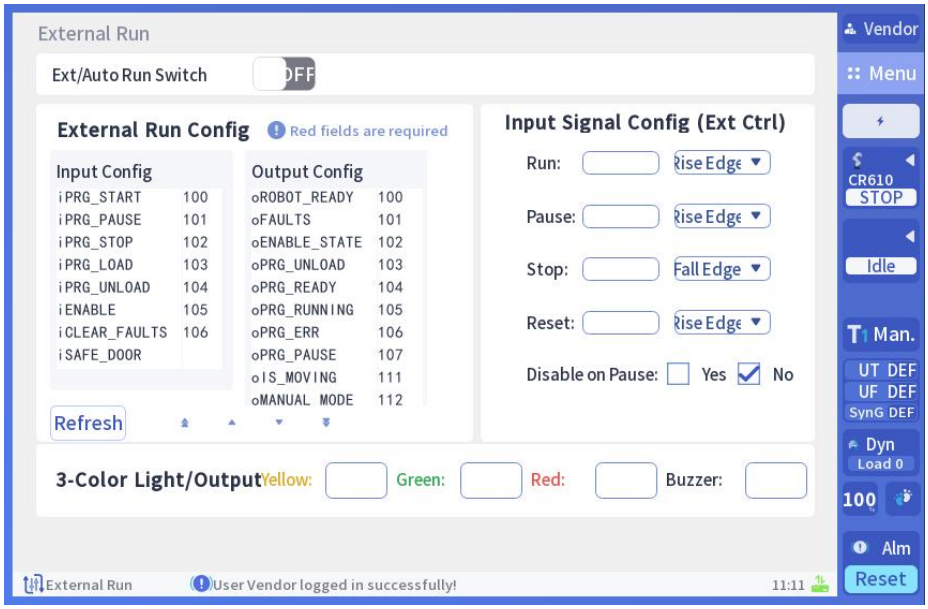
- The serial port number is typically associated with the IPC model we use. Currently, we have two IPC models: the smaller ICP (model HPC200) with serial port 0, and the larger IPC (model HPC102) with serial port 10. The integrated drive and control serial port is designated as 21.

5.1.2 External operation process package

Introduction

The existing Type III external mode configuration requires cumbersome manual setup through the teaching pendant's external operation interface, involving multiple input/output signal bindings. When each external input signal demands physical control buttons, this process becomes labor-intensive and may not align with field conditions. Consequently, field technicians often need to develop user-defined PLC external control logic or utilize external modules in system applications to simplify operations and achieve one-touch start-stop functionality. However, the unfamiliarity of field personnel with system programming environments significantly increases debugging complexity. To address this, there is an urgent need to integrate all operational workflows into the teaching pendant for streamlined implementation.

The external mode function, in simple terms, involves binding input signals through a teaching pendant to trigger standard operations for external programs, such as enabling, loading, running, pausing, stopping, and clearing alarms. By connecting output signals, it also displays the robot's status indicators including program status, enable status, alarm status, current mode, and zone outputs.



Path

➢ Menu Bar → Plugin Packages → External Run Process Packages

External operation configuration

Click the Refresh button to retrieve the DI and DO configuration inputs from the running configuration. Manual input is not allowed. If any input or output signal is not configured, a prompt will appear when you click Confirm: "Running configuration input/output signals not fully configured"

Three-color light configuration

The three-color light signal system consists of green, yellow, and red indicators. When the robot is in operation mode (i.e., during cyclic motion), the green light illuminates. When the robot is in preparation mode (i.e., after the PRG program is loaded and the motor is enabled), the yellow light lights up. When the robot is in alarm mode (i.e., when the teaching pendant interface displays any error), the red light activates.

Input signal configuration

External control buttons. The four digital input signals connected to the IO module are controlled by four buttons: Start (rising edge), Pause (rising edge), Stop (falling edge), and Reset (rising edge).

point out ：

- The external trigger signal is triggered by the selection button of rising edge trigger or falling edge trigger, so that the user can choose whether to use rising edge trigger or falling edge trigger according to the actual use situation.

- The three-color light signal cannot share the same DO signal. This requires adding a restriction in the teaching interface. For example, if the yellow light is configured to output DO[0], the green and yellow lights cannot share this DO[0] signal.
- External control button signals cannot be configured to share the same DI signal. For example, if the run signal is set as DI[1], other button signals cannot be configured to use DI[1].

5.1.3 Welding process package

explain

The welding procedure package is a welding parameter setting module of the welding system, which can set the welding machine brand, welding machine working mode, welding current and voltage and other parameters.

operating steps

Click Plug-in Package-Welding Process Package in the primary menu bar to access the welding process package interface.

System configuration: This module allows users to set welding methods, select welding machine brands, configure operating modes, enable/reduce status restoration, activate arc detection with specified intervals, and adjust forward/reverse wire feeding time.

The configuration of welding machine curve is to set the mapping curve of arc welding current, voltage, and laser power and wire feeding speed.

Process parameters: Configure the arc welding/laser welding process parameters (30 configurable sets), including welding machine operating mode, arc initiation/termination voltage/current, welding voltage/current, welding speed, laser power, wire feed speed, etc.

Welding status: The functional parameters display the values of communication readiness, successful arc initiation, welding fault, successful positioning, and anti-collision contact signal. The process parameters display welding voltage, welding current, and wire feeder speed.

Laser Positioning: Configuring Sensor Communication and Calibrating Laser Position

Production statistics: Count the number of times a specified program runs, i.e., the production quantity of a specified product.

Contact-based positioning: Parameters are configured for the contact-based positioning channel, and the channels contents are accessed during program execution.

Arc tracking: track and correct the welding path to improve the welding quality and precision.

Multi-layer and multi-channel: Parameters for multi-layer and multi-channel channels are set for program invocation during welding offset trajectory.

External program: Configure the external running MAIN.PRG.

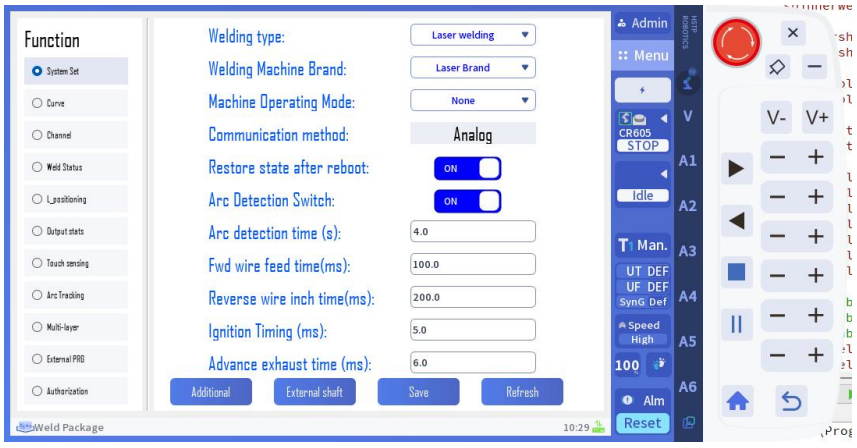
Authorization: The process package authorization interface. The welding process package requires authorization before use.

5.1.3.1 Basic setup of the welding process package

1) Welding System Configuration

	This feature is accessible only to debuggers and above, with production users limited to viewing.

Open the teaching device selection menu: Plugin Package → Welding Process Package → System Configuration, as shown in Figure 14-2.



As shown in the interface, the welding system includes 7 configuration items (Figure 14-2).

(1) Welding method: Arc welding or laser welding is available.

(2) Welding machine brands: Currently equipped with Magmet, Autar, Panasonic, and Fornis models.

(3) Welding machine operating modes: Different brands of welding machines have corresponding operating modes. Analog welding machine operating mode: None

(4) Welding communication mode: displays the current communication mode of the welding machine.

(5) Restore program state after restart: Enable to restore the program state after restarting. Restore the program state. The previously loaded program will automatically load and resume execution from the previous line after reboot.

After disabling this feature, restarting the program will not restore its state, and programs loaded before the restart will not load automatically. This feature is disabled by default. Closed.

(6) Arc detection switch: Enables or disables arc detection. The function is enabled by default.

(7) Arc detection time: Set the arc detection time, ranging from 0 to 10 seconds, with a default of 4 seconds.

(8) Forward point feed time: Set the forward point feed time, ranging from 70 to 800ms.

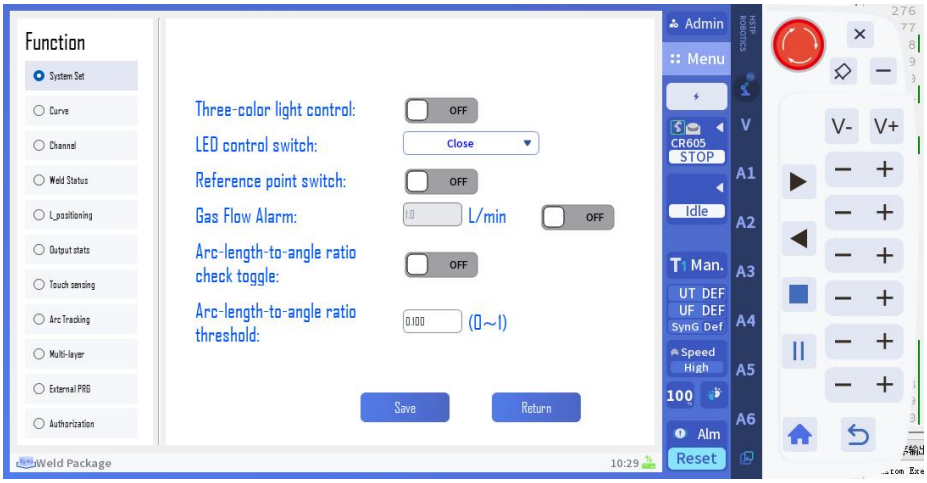
(9) Reverse point feed time: Set the reverse point feed time, ranging from 100 to 1000ms.

(10) Pre-welding gas time (displayed when laser welding mode is selected): Set the pre-welding gas time, ranging from 0 to 1000ms.

(11) Advance gas collection time (displayed when laser welding is selected): Set the advance gas collection time, ranging from 0 to 1000ms.

Note: Changes to welding method, welding machine brand, welding mode, and arc detection switch in system configuration take effect immediately. Arc detection time and spot wire feeding time require clicking "Save Configuration" to apply. Without clicking "Save Configuration" after modification, the system will revert to previous settings. Click "Save Configuration" after restarting the system to apply the latest configuration.

Click the Additional Function Settings button to access the configuration interface, where you can adjust the three-color light switch, LED control switch, reference point switch, and set the gas flow detection threshold parameters.



(1) Three-color light control switch: Enables or disables the three-color light, with default off.

(2) LED control switch: Enables or disables the LED display, with the display being off by default.

(3) Reference Point Switch: Enables or disables the reference point detection function during external

mode startup. After modification, click Save in Configuration> Controller> Run> Program to apply changes. This function is enabled by default.

(4) Gas flow detection threshold: Set the gas flow rate, with a switch adjacent to it. After activation, it is used during the actual welding process.

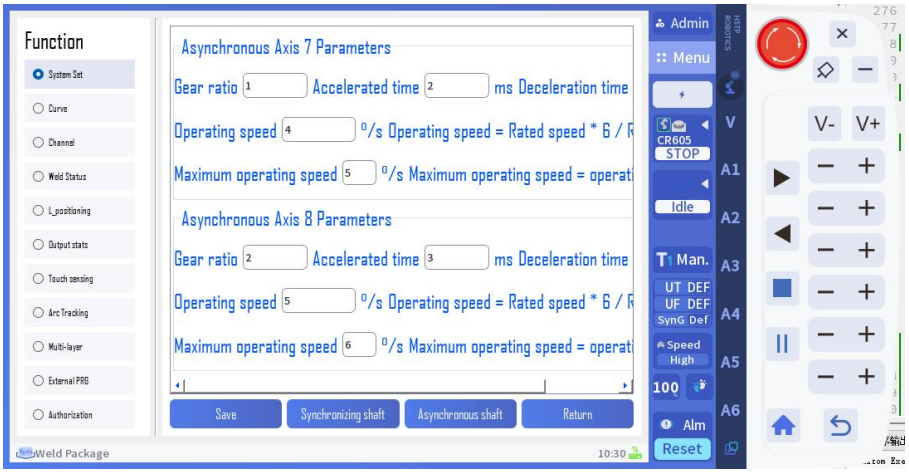
When starting the arc, it checks if there is still gas. If no gas is present, the arc will fail and welding will stop. The setting range is 0.1~30L/min。

(5) Proportional detection switch for arc length and attitude angle: Enables or disables proportional detection of arc length and attitude angle.Yes, when the arc length between two adjacent points in the displayed arc is shorter than other segments after opening, the attitude changes of these two points also

The degree remains unaffected, but severe cases may cause jitter. The program will trigger an alarm when executing this arc segment. Disabling this setting prevents the issue. It measures the distance between two points on an arc. This feature helps reduce arc jitter.

(6) Proportion detection threshold for arc length and attitude angle: Set the proportion detection threshold for arc length and attitude angle, which can Adjust this threshold based on the arc requirements of the taught welding segment. A higher threshold ensures smoother arc operation for successful teaching. Set the range 0 to 1.

Click the External Shaft Configuration button to access the interface where you can configure the shaft type, pitch, reduction ratio, acceleration/deceleration time, operating speed, and maximum operating speed of the external shaft.



1 Curve Configuration

1) Configuration of Arc Welding Voltage Curve

Different welding machine brands have distinct voltage curves. On the configuration page, select

Voltage Curve at the top to adjust the input and output voltage settings.

The steps are:

(1) If the minimum and maximum voltage values of the welding machine are known, enter them directly into the input box. Otherwise, perform trial measurements as outlined in steps 2 to 5 to determine the output voltage curve.

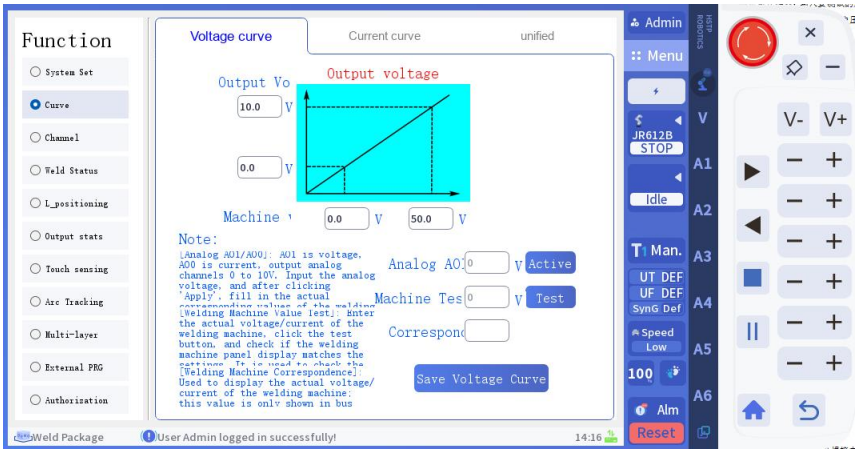
(2) Enter the voltage value (range: 0~10) in the input field to the right of AO1 (start from 0 and gradually increase for testing), then click "Effective".

(3) Monitor the welders voltage display to check if it reaches the minimum value. If so, adjust the AO1 setting and repeat step 2 until the panels minimum voltage changes. Record the AO1 value at this point and enter it into the output voltage input field.

(4) Enter a voltage value (range: 0~10) in the input field to the right of AO1 (start with 10 and gradually decrease for testing), then click "Activate".

(5) Repeat steps 3 and 4 until the welders maximum voltage value changes. Record the A01 value at this point and enter it into the output voltage field.

(6) Verify the accuracy of the output voltage curve. In the welder voltage test input field, enter the target voltage value. Compare the displayed voltage on the panel with the input value (theoretical value) to ensure the deviation remains within the allowable range (maximum deviation: 1% of full scale). If the deviation is excessive, reconfigure the settings according to steps 1-5.



2) Welding Current Curve Configuration

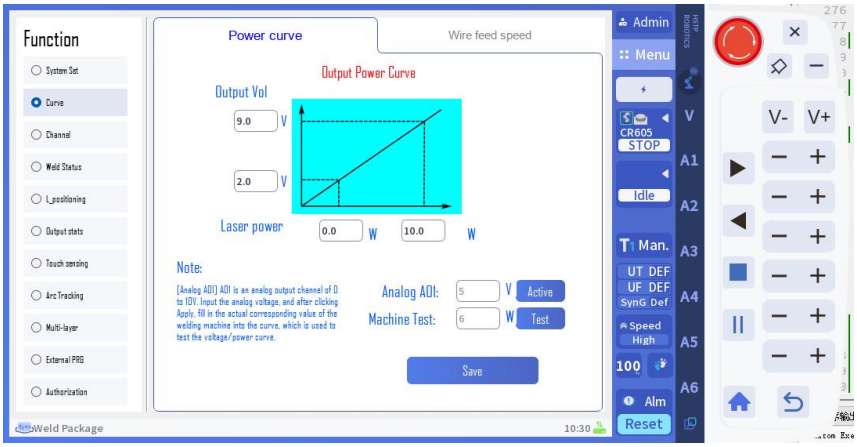
Refer to Section 2.2 for the welding voltage curve configuration procedure.

3) onfiguration of Arc Welding Unified Curve

For analog communication welding machines, first switch to the unified mode on the panel, then

configure according to the welding voltage curve setup procedure in Section 2.2.

4) Configuration of Laser Welding Power Curve

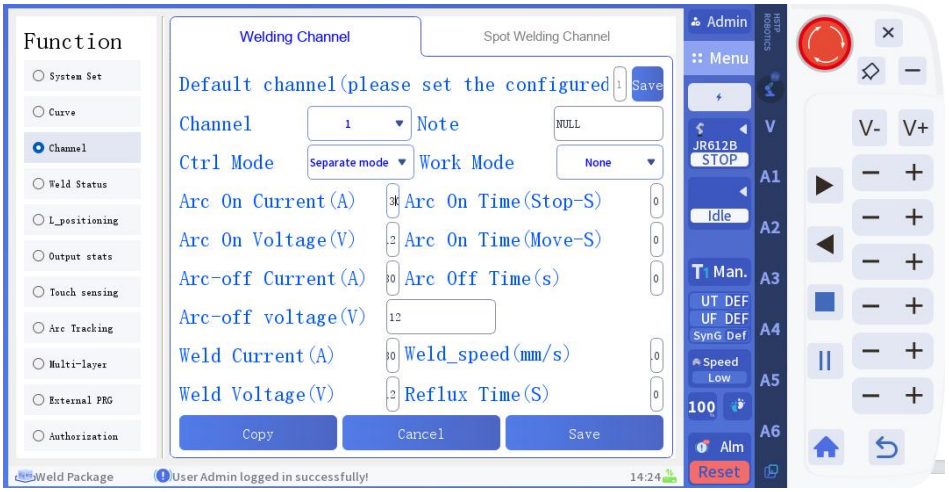


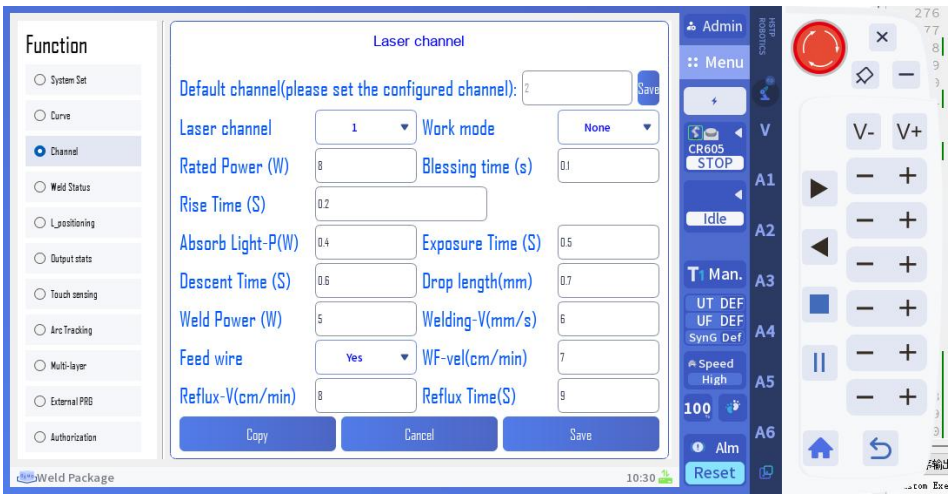
Refer to the configuration process of welding voltage curve in 2.2 for configuration.

5) Configuration of Laser Welding Wire Feeding Speed Curve

Refer to the configuration process of welding voltage curve in 2.2 for configuration.

2 Process Parameters





(1) When using arc welding, both the welding channel and swing welding channel are displayed. For laser welding, only the laser channel is shown. The welding channel includes default channel number,

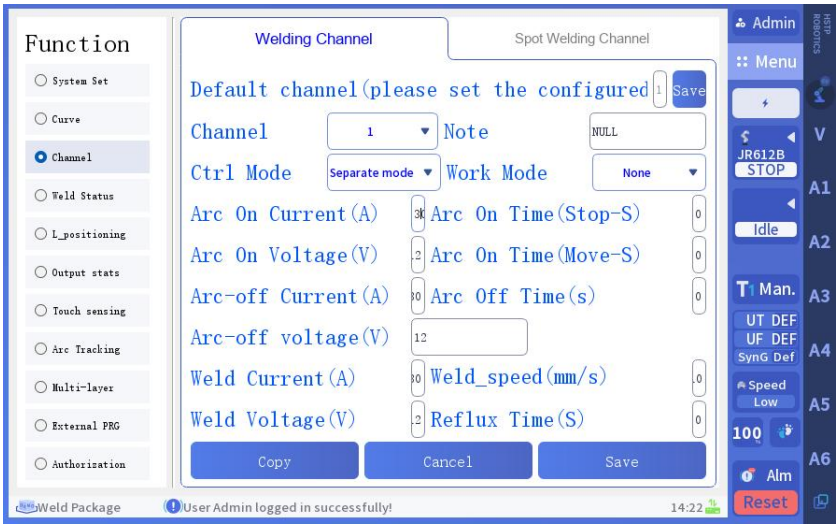
channel number, channel description, control mode, welding machine operating mode, arc initiation voltage, arc termination voltage, arc termination current, post-arc wait time, arc termination time, welding voltage, welding current, retraction time, and welding speed. The swing welding channel includes default channel number, channel number, channel description, swing shape, swing frequency, swing plane, swing amplitude, whether to pause, left/right pause time, elevation angle, azimuth angle, reference point 1, and reference point 2. The laser channel includes default channel number, channel number, operating mode, light-on power/time, slow-up time, light-off power/time, slow-down time/distance, welding power/speed, whether to feed wire, wire feeding speed, retraction speed, and retraction time.

(2) The interface of welding channel and swing welding channel is provided with the copy, reset and save button.

6) Welding Channel

A welding channel is a set of data saved after welding parameter settings. The number of channels can be configured, with a maximum of 30. Once set, the program can call the welding channels.

Open the manipulator selection menu: Plug-in Package → Welding Process Package → Process Parameters → Welding Channel. The welding channel supports 1 6 configurable parameters, including: default channel number, channel number, channel description, control mode, welding machine operating mode, arc initiation voltage, arc termination voltage, arc termination current, post-arc wait time, arc termination time, welding voltage, welding current, retraction time, and welding speed, as shown in the figure below.



Default channel number: (1~30)

Channel number: the sequence number of a set of process parameters (1~30)

Channel Description: Text description of current welding parameters

Control modes: There are three control modes: separate mode, unified mode

Welding machine operating mode: Select the corresponding operating mode based on the brand of the welding machine. The analog welding machine operates in silent mode.

Arc initiation voltage: voltage of welding machine in arc initiation stage

Arc-starting current: current of welding machine in arc-starting stage

Arc-closing voltage: voltage of arc-welding machine in arc-closing stage

Arc-closing current: current of arc-welding machine in arc-closing stage

Arc start time (pause): The duration the robot stays at the arc start point after successful arc initiation

Arc initiation time (dynamic): The time from arc initiation current/voltage to welding current/voltage (during robot movement)

Arc closure time: The time to close the arc

Welding voltage: The voltage of the welding machine during the welding phase

Welding current: current of welding machine in welding stage

Welding speed: the welding running speed of robot in welding stage

Retraction time: the time for wire retraction

7) Copy

Channel replication steps:

(1) Click the "Copy" button below the channel list to open the channel copy dialog box shown in the figure below.

(2) The channel copy dialog includes the source channel number and the target channel number. The source channel number displays the row number to be copied in the edit box, and the target channel number specifies the channel to which the copy is applied.

(3) Click OK to copy the channel.

8) Cancel

Canceling the operation will undo changes made before the last save. The effect takes effect immediately.

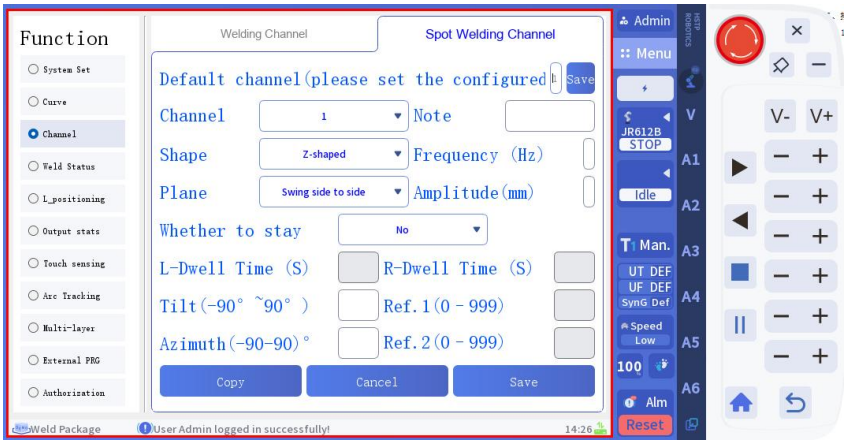
9) Storage

Modify the displayed parameter values. Welding channel number, control mode, and welding machine mode are optional, while other values are input fields. Enter the value you want to change, then click "Save" to complete the modification.

10) Swaging Welding Channel

The swing welding channel is a set of data saved after welding parameter settings. The number of channels can be configured, with a maximum of 10. Once set, the program can call the swing welding channels.

To access the manipulator selection menu: Plug-in Package → Welding Process Package → Process Parameters → Swing Welding Channel. The adjustable parameters for the swing welding channel include: default channel number, channel number, channel description, swing shape, swing frequency, swing plane, swing amplitude, stay mode, left/right stay time, elevation angle, azimuth angle, reference point 1, reference point 2, and 12 other parameters, totaling 14 items, as shown in the figure below.



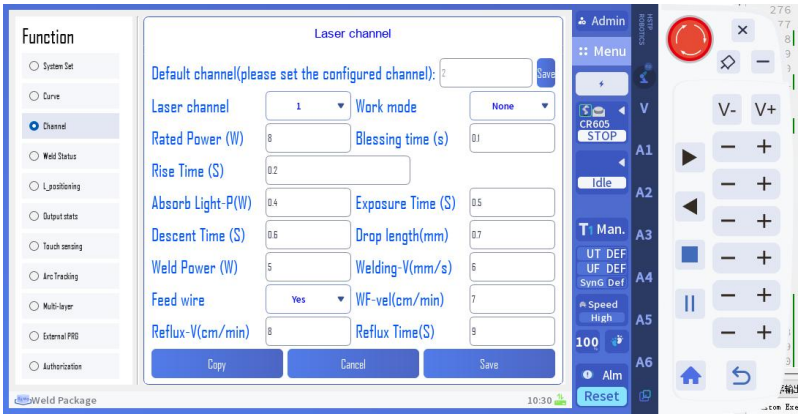
The operation of the flip-chip channel is similar to that of the welding channel.

11) Laser Channel

A laser channel is a set of saved parameters for laser welding. The number of channels can be configured, with a maximum of 15. Once set, the program can call the channels.

Open the manipulator selection menu: Plug-in Package → Welding Process Package → Process Parameters → Laser Channel. The weld channel supports 1 6 configurable parameters, including: default channel number, channel number, operating mode, light-on power/time, ramp-up time, light-off power/time, ramp-down time/distance, welding power/speed, wire feeding/feeding speed, retraction speed, and

retraction time, as shown in the figure below.

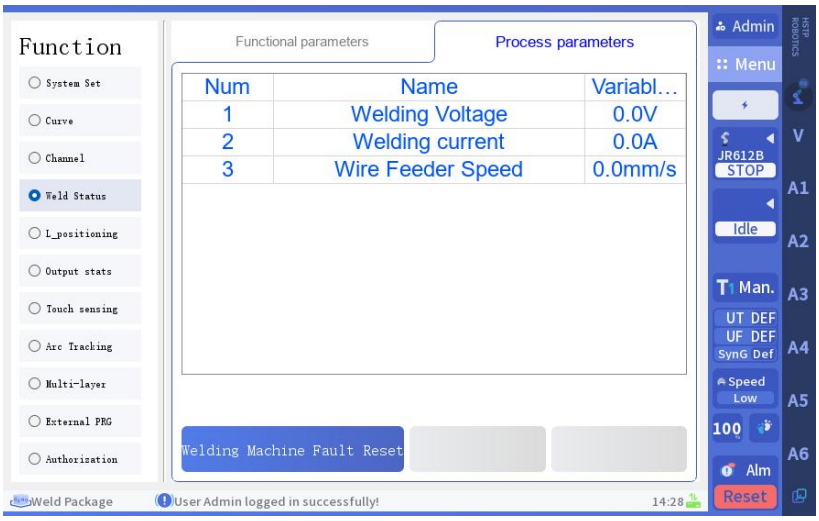


The operation of laser channel such as copy, cancel and save is similar to that of welding channel.

3 Welding Condition

The function parameters and process parameters of the current welding are displayed under the bus communication.

Function parameter status

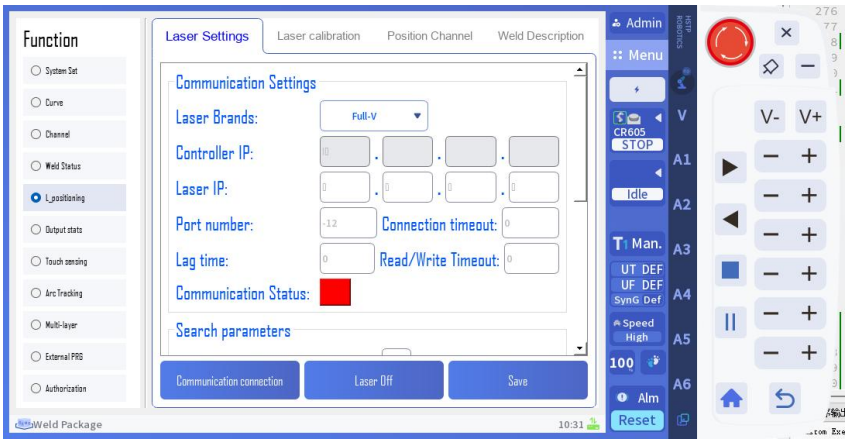


Process parameter data

4 Laser Positioning

1) Communication and Tracking Parameter Settings

To configure the laser tracking path, navigate through the following menu sequence: Main Menu → Plugin Package → Welding Process Package → Laser Positioning. This will open the Laser Positioning Parameter Setup interface (as shown below), which includes three sections: Communication Settings, Search Parameter Settings (containing parameters for laser positioning), and Tracking Parameter Settings.



communications setting：

Laser tracking brand: Select the brand of the laser.

Controller IP address: The controllers IP address is automatically read and does not require configuration.

Laser IP address: Set the IP address of the laser.

Port number: Set the port number for laser communication.

Connection timeout: Set the connection timeout time between the robot and the laser.

Delay time: Set the delay time.

Read/Write timeout: Set the read/write timeout time.

Communication status: Displays the lasers connection status. Green indicates a successful connection, while red indicates no connection.

Search parameters:

Search count: Set the number of bit searches.

Search speed: Set the search speed.

Search distance: Set the search range.

Search starting point precision: Set the precision for the starting point of the search.

Tracking parameters:

Weld end point accuracy: Set the weld end point accuracy. X-axis tracking accuracy: Set the X-axis tracking accuracy.

Y-axis tracking accuracy: Set the Y-axis tracking accuracy.

Z-axis tracking accuracy: Set the Z-axis tracking accuracy.

Communication Connection: After completing the input communication settings, click the Communication Connection button to establish communication with the laser.

To check if the connection is successful, monitor the communication status.

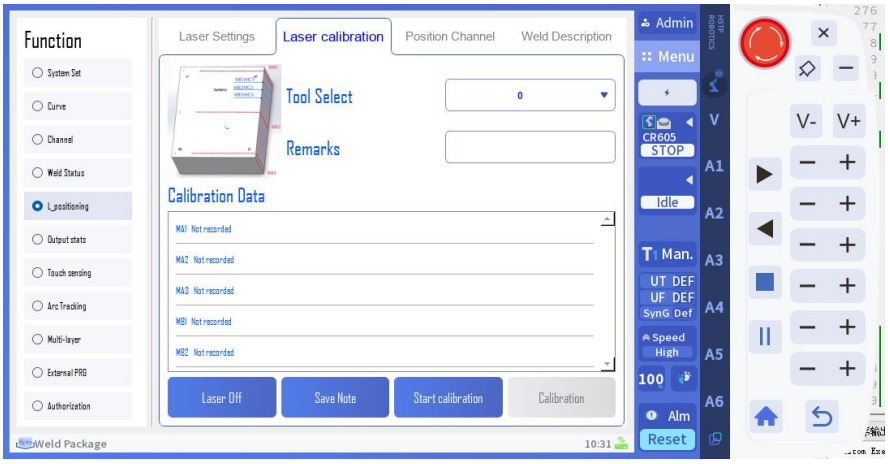
Laser on: After the communication connection is established, click to turn the laser on or off, or use the auxiliary button to configure it.

Turn off the laser.

Save parameters: After entering search and tracking parameters, click "Save parameters" to save the input parameters.

2) Laser Calibration

The laser calibration interface is designed to calibrate the relative positions between robots and sensors. The tool coordinate system must match the current tool coordinate system, and tool coordinate data must be selected during calibration. The calibration process involves recording 9 points (the detailed procedure will be explained later). Calibration begins when the first point is recorded and ends after the last point is recorded. Click the "Calibration Complete" button below to finish the laser calibration (as shown in the figure).

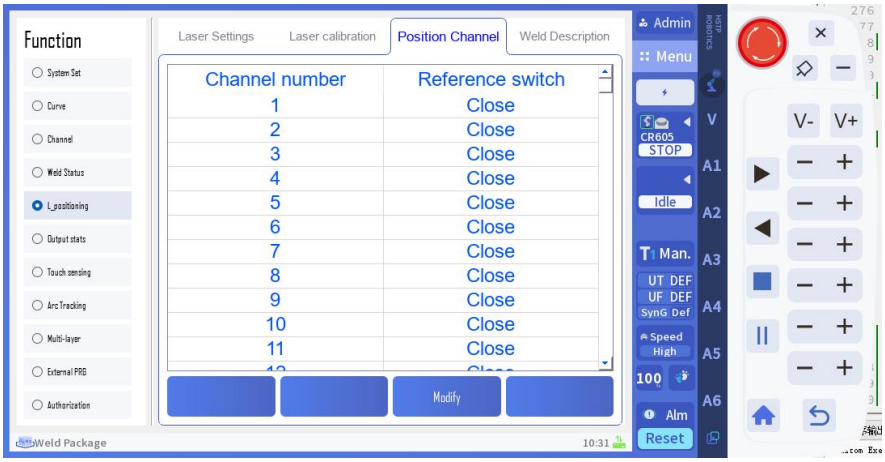


The positioning channel and laser positioning instruction are used together.

3) Positioning Channel

As shown in the figure below, select the corresponding channel and click the "Modify" button. The

system will open the reference switch for that channel in the pop-up window. During the laser positioning command, the current position is recorded as the reference position. Subsequently, closing the reference switch and running the program will enable trajectory correction.



4) Welding Joint Description

Schematic diagram of scanned weld types:

5 Production Statistics

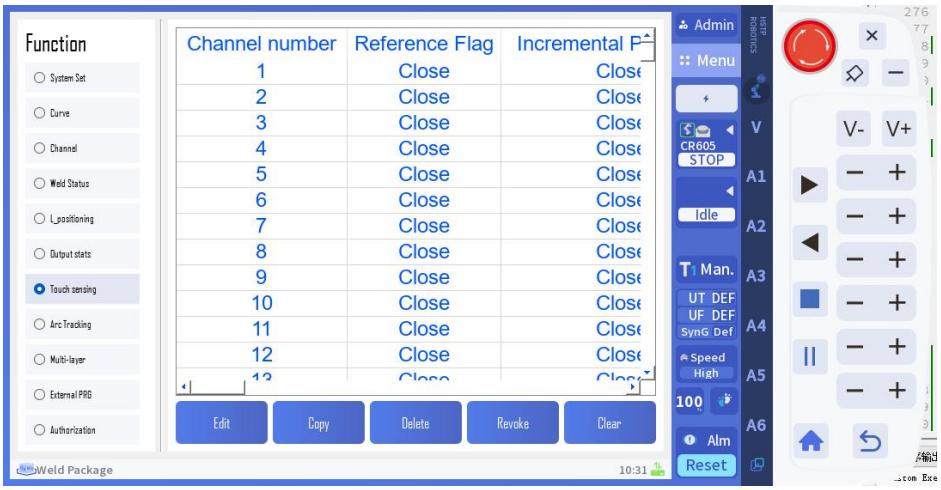
Open the teaching device selection menu: Plugin Package → Welding Process Package → Output Statistics; as shown in the figure below:

1. After adding the output statistics command to the program, turn the switch on (on for open, off for closed) for the current workstation. This will record the operational count of the workstation, i.e., the generated quantity.

2. Clicking the reset button will display a confirmation dialog asking if you want to reset the current workstation output. Click OK to reset the output.

3. Welding time statistics: Displays the total welding time for all welds (resetting R[930] resets the counter).

6 Contact Positioning



Process parameter description:

Channel number: The channel number ranges from 1 to 1 0. Set the process channel number for initiating contact-based positioning command calls.

Reference Flag: On/Off. This switch controls the writing of reference position data for channel numbers in the program. When enabled, the detected position is stored as reference data in the variable. When disabled, the reference data is protected from writing. (If the reference flag is disabled but the workpiece moves relative to the reference position, the data will be rewritten after the next positioning cycle, overwriting the previous reference data. This would invalidate the subsequent program and require rewriting the trajectory section.)

Incremental positioning: On/Off. When set to On, the system will carry over the offset from the previous position when starting the next positioning point, reducing excessive deviation to prevent missing the target position and improving positioning efficiency. When set to Off, the system will strictly follow the initial positioning point before starting the positioning.

Positioning distance: The distance from the starting point to the tool. Exceeding this distance will

trigger an alarm.

Positioning speed: The rate at which the tool moves from the starting point to the workpiece. A lower positioning speed results in higher precision.

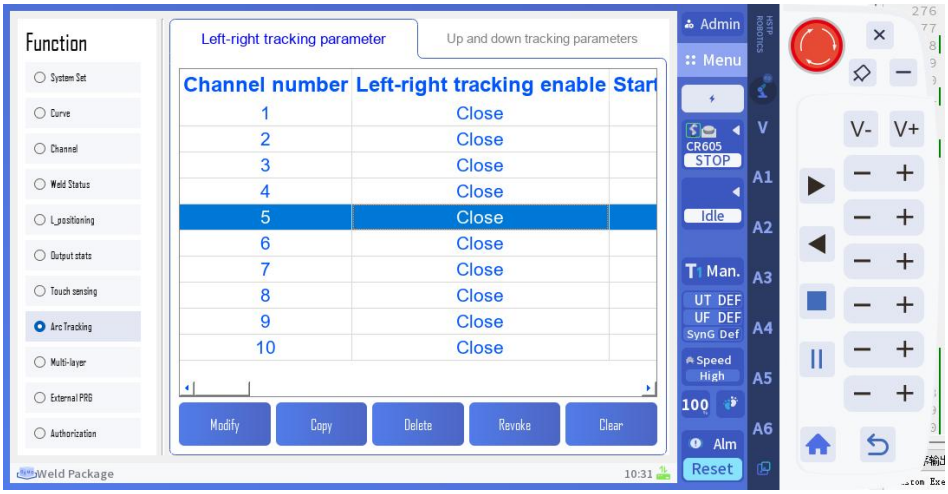
Auto-return: On/Off. Set to On, the robot will return along the previous path after the probe contacts the workpiece, based on the auto-return distance and speed. Set to Off, the robot will not return.

Auto-return distance: Set the distance for automatic return. When this distance exceeds the starting point of the position search, the robot stops moving and stops running.

Automatic return speed: Sets the speed at which the welding wire or nozzle returns after contacting the workpiece during positioning.

7 Arc Tracking

To configure process parameters: navigate to Main Menu → Plugin Package → Welding Process Package → Arc Tracking. The Arc Tracking parameter interface (divided into left/right and top/bottom sections) supports configuration for 10 channels, with tracking channels accessible through Arc Tracking commands.



1) Left and right parameter descriptions:

Left and right tracking enable: A switch to activate left and right arc tracking.

Tracking count for left and right: Arc tracking starts from the 2nd cycle for 120A and 4th cycle for 180A.

Current data delay period: varies by brand and robot communication compensation (Aotai 36, Megmeet 18).

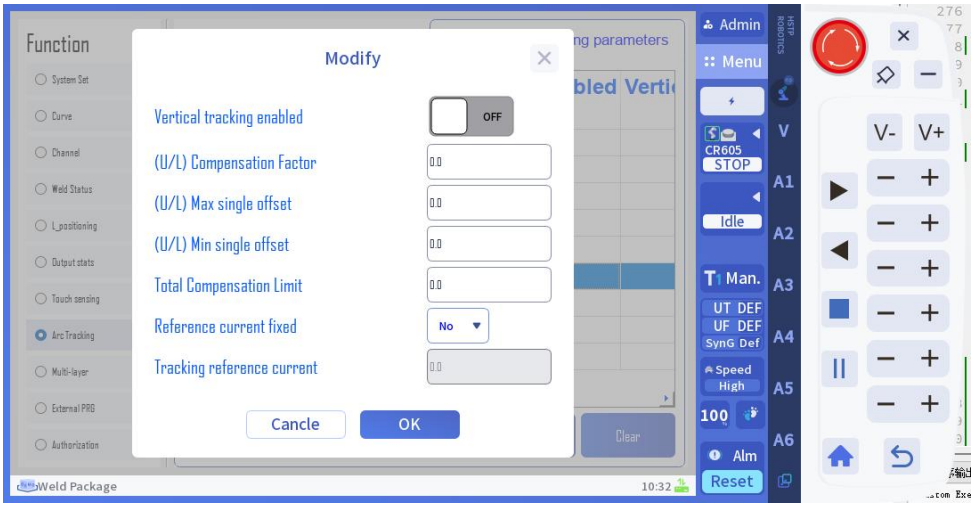
Left and right compensation coefficients: This parameter defines the coefficient value for amplitude compensation during deviation compensation (recommended value: 0.08).

Maximum single compensation distance per swing cycle (left/right): The maximum compensation distance for each swing cycle (recommended: 1mm).

Minimum single compensation distance: The minimum left-right compensation distance per swing cycle (recommended: 0.01mm).

Total compensation limit for left and right directions: During welding, the cumulative compensation value after each phase addition (recommended: 500mm).

2) Explanation of Up and Down Parameters:



Up and Down Tracking Enable: A switch to enable arc tracking for both top and bottom arcs.

Upper and lower compensation coefficients: This parameter defines the coefficient value for amplitude compensation during deviation compensation (recommended value: 0.1).

Maximum single compensation range: Maximum left-right compensation distance per swing cycle (recommended: 0.8mm).

Minimum compensation range for vertical movement: The minimum left-right compensation distance per swing cycle (recommended: 0.01mm).

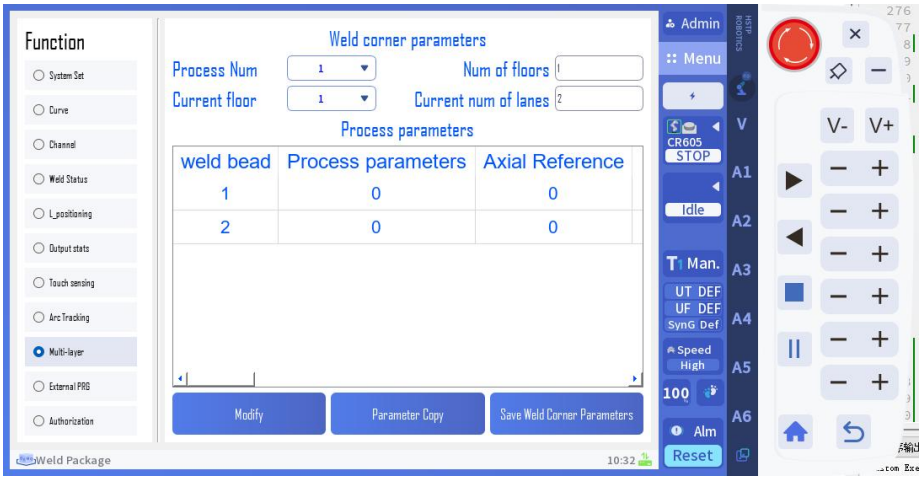
Total compensation limit for vertical movement: During welding, the cumulative compensation value after each phase addition (recommended: 500mm).

Fixed upper and lower reference currents: During welding, the cumulative compensation value is calculated after each compensation addition (recommended: no).

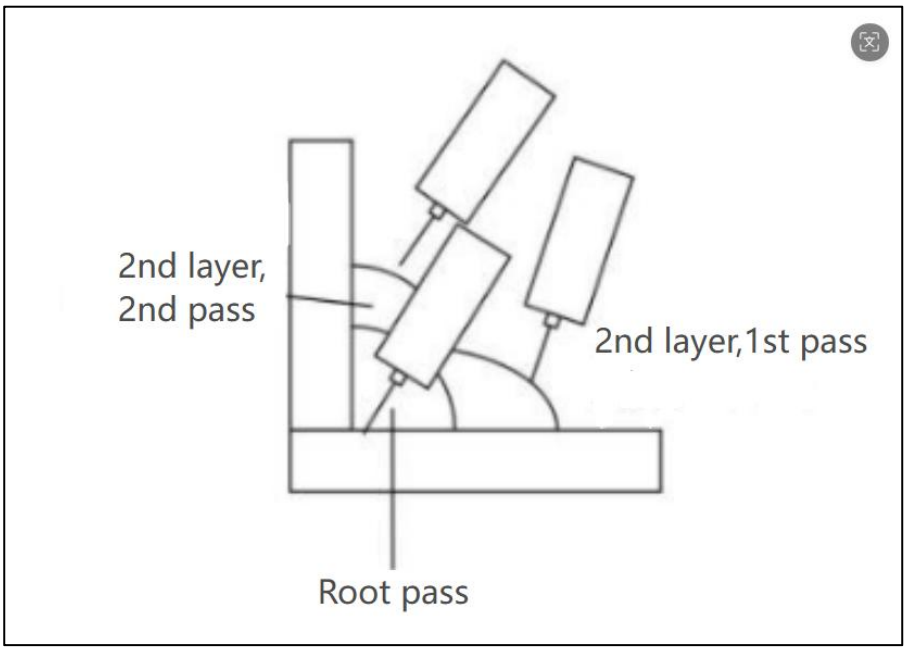
Reference current for upper and lower tracking: The reference current value for upper and lower deviation correction.

8 Multi-layer and multi-channel

To access the multi-layer and multi-pass welding parameter settings, navigate through: Main Menu → Plugin Package → Welding Process Package → Multi-layer and Multi-pass. The interface (as shown below) consists of two sections: welding angle parameters and process parameters.



Welding Angle Parameters: Planning of Welding Quantity of Corresponding Weld



Process ID channel: Process IDs 1-30 are configured to call multi-layer, multi-channel process IDs.

Layers: Set the number of layers for multi-layer, multi-pass welding. The value matches the dropdown options for the current layer count (e.g., 2 layers in the image above).

Current layer: Select a specific layer from the dropdown to edit (generated automatically based on the total number of layers).

Number of current weld passes: The number of weld passes for the current layer (as shown in the figure above. If the current layer number is set to 1, the number of passes is set to 1; if the current layer number is set to 2, the number of passes is set to 2).

Process parameters: specific values for each welding procedure.

Weld track: The sequence number of the previous layer.

Process parameter: Sets the welding process channel number to be used during welding.

Axis reference: The fixed number is currently set to 8

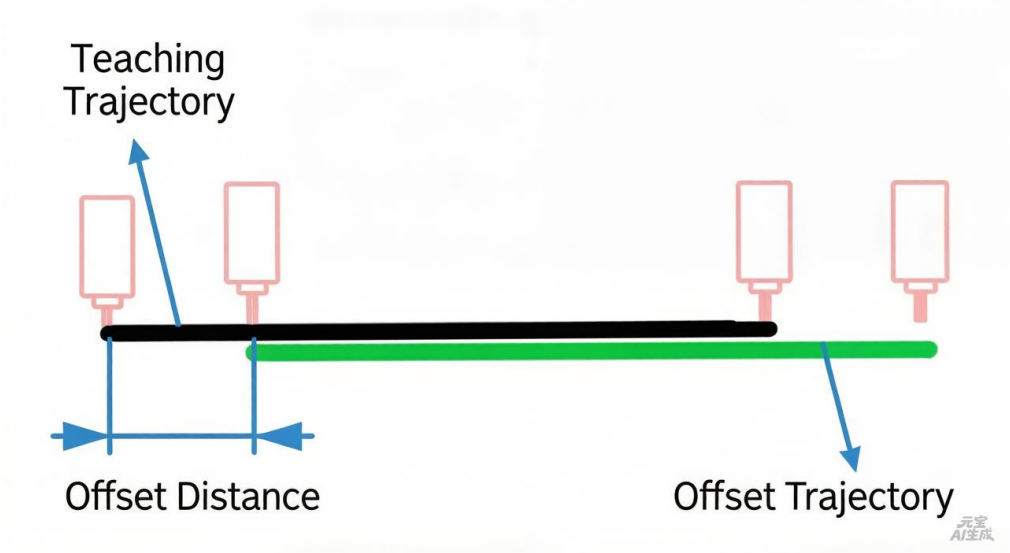
Swing: Set whether to swing during welding this seam.

Swing welding channel: Specifies the channel number for this weld seam during welding.

Multilayer Multichannel Switch: Set to enable the current parameters, otherwise they are disabled.

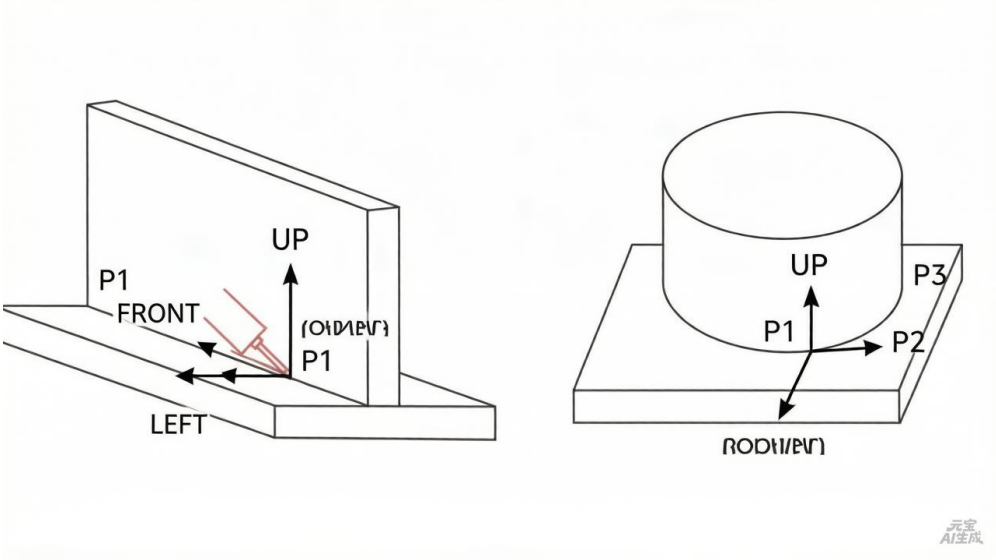
Direction reversal: Whether to weld from the arc closing point back to the arc starting point or from the arc closing point to the arc starting point after the current arc closes.

Start/End Point Offset: Based on the teaching point trajectory, the overall trajectory is offset in the forward or backward direction, with welding direction as positive and the opposite as negative.

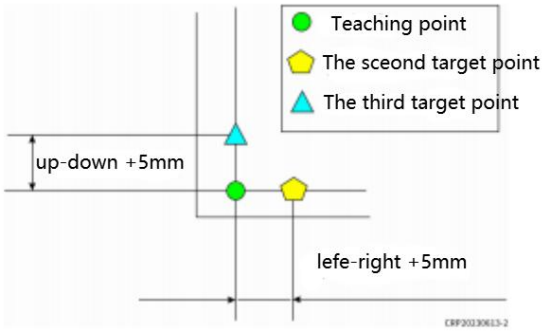


Left/Right offset: Adjusts the trajectory left or right based on the teaching point path, as shown in the figure below. (Different weld types require different algorithm coordinate systems).

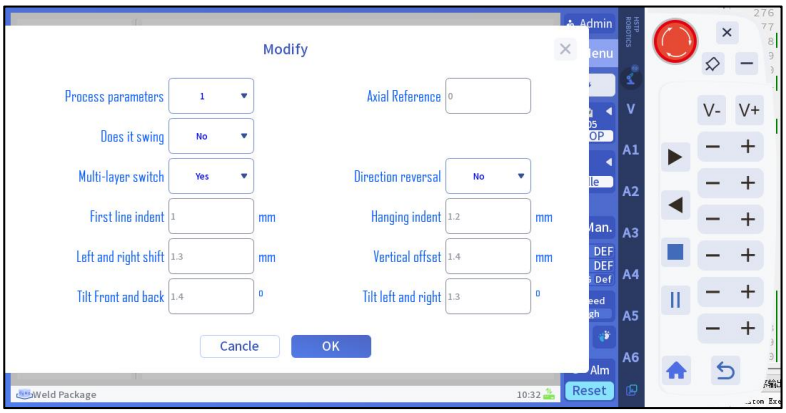
Vertical offset: Adjusts the trajectory vertically relative to the teaching point path. Refer to the figure below (different weld types require corresponding algorithm coordinate systems).



For example, in the fillet weld shown below, the translational offsets for the target positions of the three weld passes are set relative to the first taught point. Example: left-right offset of the weld seam = +5 mm, up-down offset of the weld seam = +5 mm. (Offsets along the front-back, up-down, and left-right directions of the weld seam can be combined to shift a single point simultaneously.)



Front/Back/Lateral Tilt Angle: The welding torch angle is a critical factor in welding conditions, which is a function that adjusts the torch angle through numerical settings. Click the "Parameter Modification" button to set the process parameters for each layer and pass (as shown in Figure 14-30). After setting, the welding process for that pass will use the specified parameters.



Click the "Copy Parameters" button to copy the selected track parameters to the target track (as shown in the figure).

Click the Save button to save the modified weld angle parameters.

9 External Procedures

operating steps

1. In the main menu, select Plugin Package → Welding Process Package → External Program.

2. Click the cursor to select any workstation number, then click the "Configure" button. (There are 10 workstation numbers in total, and you can configure 10 external programs simultaneously)

3. The program selection interface will pop up. Click to select the external program to run.

3. Click OK to complete the external program configuration.

5. Click Save on the toolbar to complete the external program configuration.

	After configuring the input/output and program, the MAIN.PRG main program will automatically execute when switching to external mode. The system will display the program for Station 1 if no program is running. Upon startup, the current program will be shown. If an error message appears stating External operation failed. The robot is not at the reference point. Switch to teaching mode and move the robot to the reference point!, you must configure the reference point and move the robot to the reference point in the menu → Function Configuration → Run Configuration → Program Run Configuration. Note: When configuring workstation programs, you cannot set up a program file named MAIN.PRG.

10 Authorization

explain

The welding process system requires separate authorization

Figure 14-32 Authorization Function

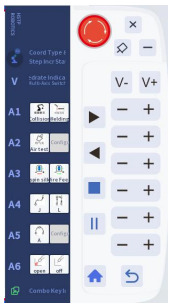
operating steps ：

(1) In the main menu, select Plugin Package → Welding Process Package → Authorization.

(2) Insert the USB drive and click to retrieve the serial number (SN). A file named XX-XX-XX-XX-XX-XXsn_threesolder will be generated on the drive. Send this file to our company for registration, where XX represents the system SN.

(3) Transfer the registration file XX-XX-XX-XX-XX-XXgenThreeArcAuthcode to a USB drive, insert it, and click Register to complete the process.

5.1.3.2 Introduction to welding button function



1. Collision Resolution: When a robot collides, press the corresponding A1 "-" key to activate the collision resolution switch. The collision signal is then cleared, and the robot has 15 seconds to move away from the collision point. If the robot fails to move within this time, the switch will automatically close. To continue moving the robot, press the switch again. If the robot completes its movement within the time limit, press the A1 "-" key to deactivate the collision resolution.

2. Welding Switch: This is the main control switch for welding. It only activates in automatic mode and is disabled in teaching mode. Turning it off disables welding-related functions. To activate, press the A1 "+" key; to deactivate, press the A1 "+" key again.

3. Gas inspection: Press the A2 "-" key to open the gas valve. Press the "-" key again to close the gas valve.

4. Filament Feeding Control: Press the A3 "+" button under the physical button to initiate forward filament feeding. Press and hold for a set duration to enable forward intermittent feeding, while pressing and releasing for a set duration activates forward continuous feeding. Both modes will stop feeding upon release. Press the A3 "-" button for reverse filament feeding. Press and hold for a set duration to activate reverse intermittent feeding, while pressing and releasing for a set duration enables reverse continuous feeding. Both modes will stop feeding upon release. The duration can be configured in the process package system.

5. The A4" button is the joint insertion command key. Pressing it activates the joint insertion command, which takes effect when the program starts editing.

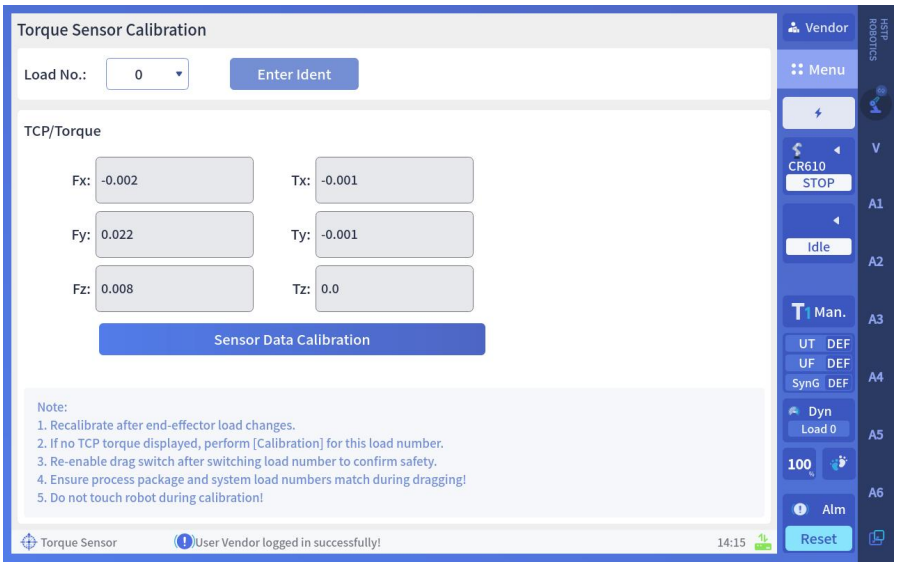
6. The A4 "+" button is for inserting a straight-line command. Pressing it activates the command, which takes effect when the program starts editing.

7. The A5" button is used to insert an arc command. Pressing it activates the command, which takes effect when the program starts editing.

8. During arc welding, press the A6 "button to insert the arc initiation command. The system will activate the default channel number configured in the process packages welding channel. For laser welding, press the A6" button to insert the laser beam activation command. The system will activate the default channel number set in the process packages laser channel. The program becomes effective when the combination key is pressed during editing.

9. The A6 "+" button inserts a closing arc command. When pressed, it activates the closing arc channel (the default channel number in the welding process package). For laser welding, the A6 "+" button inserts a closing light command, activating the closing light channel (the default channel number in the laser process package). Pressing the combination key button during program editing activates these commands.

5.1.4 Torque sensor process package

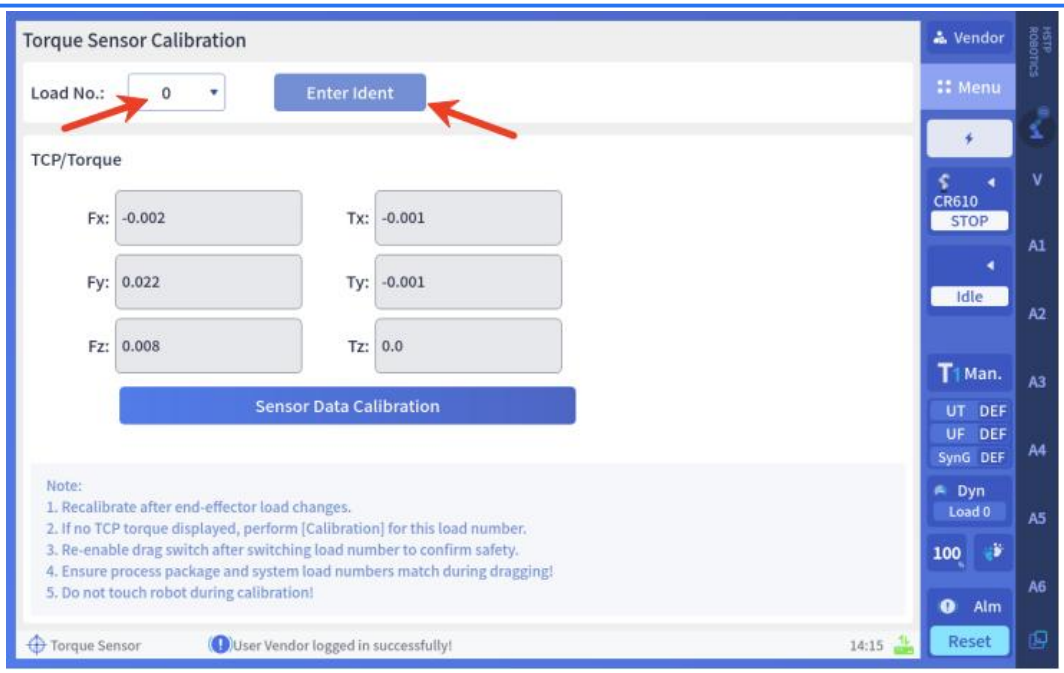


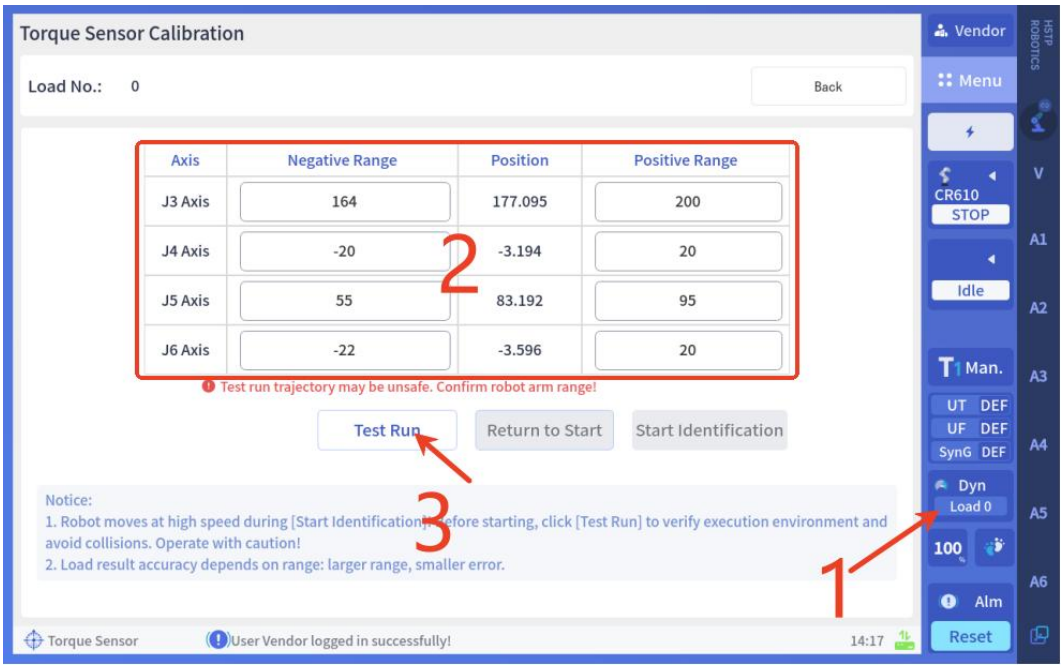
This process package is only applicable to version V1.6.11 and the robot configuration is dedicated to robots, and dynamic identification also needs to be done. The sensor needs to be connected correctly (the determination method is CA input?) The group[0].forceoriginal directive returns a value. For the newly adapted sensor, the sensor switch in the parameter file needs to be turned on. Enter CA as group[0].dynsensor.sensoren =1 and save var.saveGp(0,"dynsensor"). Newly adapted sensors, changes in sensor installation methods, and alterations in terminal loads all require calibration. After calibration, calibration needs to be performed each time the power is turned on.

5.1.4.1 Sensor calibration

Sensor calibration and calibration process

1. Switch the tool artifact number on the right side of the teaching device to the default.
2. Switch the load number on the right side of the teaching device to no-load, and select the load number that needs to be calibrated for the process package.
3. Ensure that the load number of the calibration interface is consistent with the expected selected load number;
4. Set the motion range of each axis;
5. Click "Trial Run";
6. Click "Start Recognition". At this point, the robot is running a program. You need to wait for the program to finish.
7. Click "Back";
8. Click "Calibration";
9. Sensor calibration and verification have been completed.





The first page of the process package displays the current load number, the external force value of the sensor based on the current calibration parameters, calibration and calibration buttons. The calibration button is used to calibrate the sensor that has already been set to zero drift (Note: The sensor needs to be calibrated each time the power is turned on and the machine is restarted). The calibration button is used when the sensor has not been calibrated, when the sensor installation method or the terminal load changes. In these cases, the sensor needs to be recalibrated. The calibration parameters are bound to the load number and automatically saved. During the calibration of the sensor, the load parameters will be identified simultaneously, and users do not need to re-identify the load separately.

5.1.4.2 Pick up calibration

In the menu, select the plugin package → Torque Sensor Calibration → Calibration Instructions:

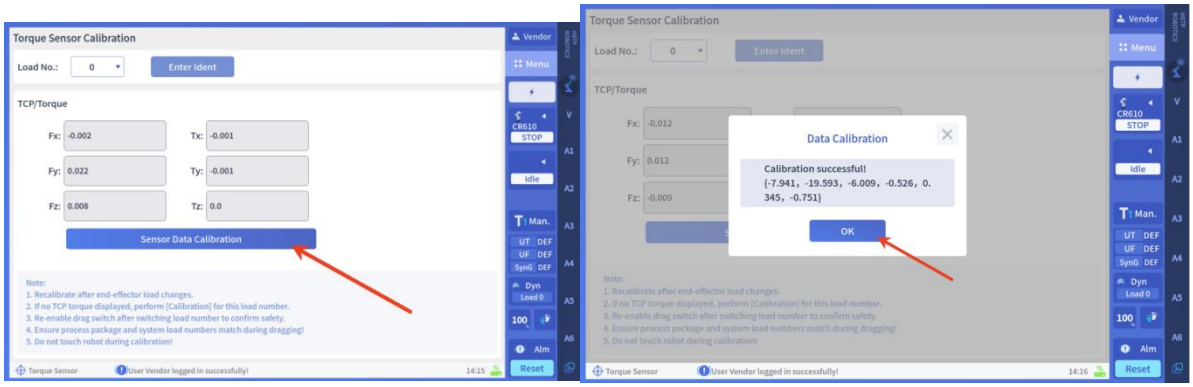
1. Recalibration is required after the terminal load changes.

2. If there is no display of the TCP torque at the end, it indicates that the "calibration" has not been p erformed under this number. Please perform the "calibration" operation.

3. After switching the load number, the drag switch must be turned on again to confirm the safety of th e drag.

4. During the dragging process, please ensure that the load numbers of the process package and the s ystem are consistent!

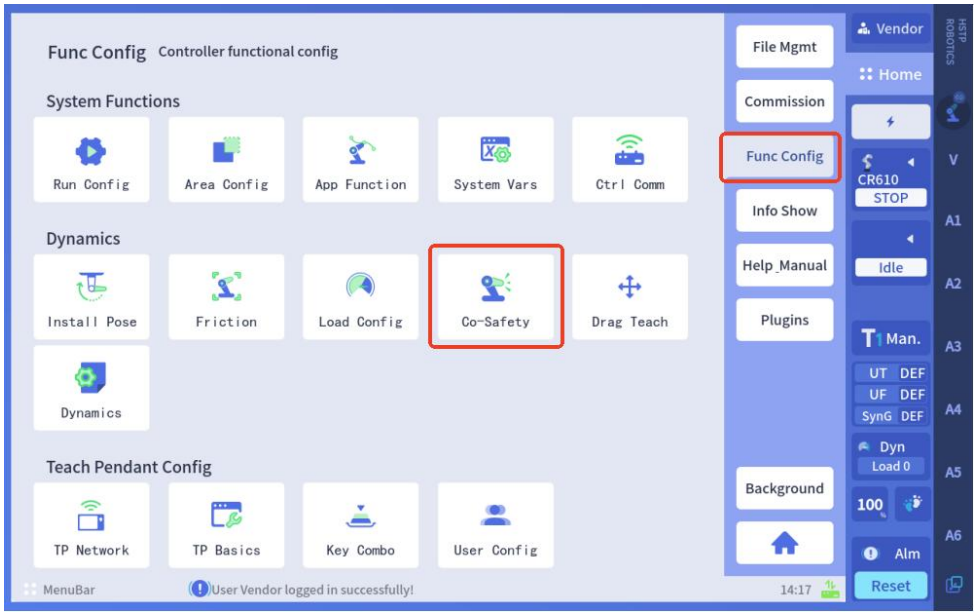
5. Please do not touch the robot during the calibration process!

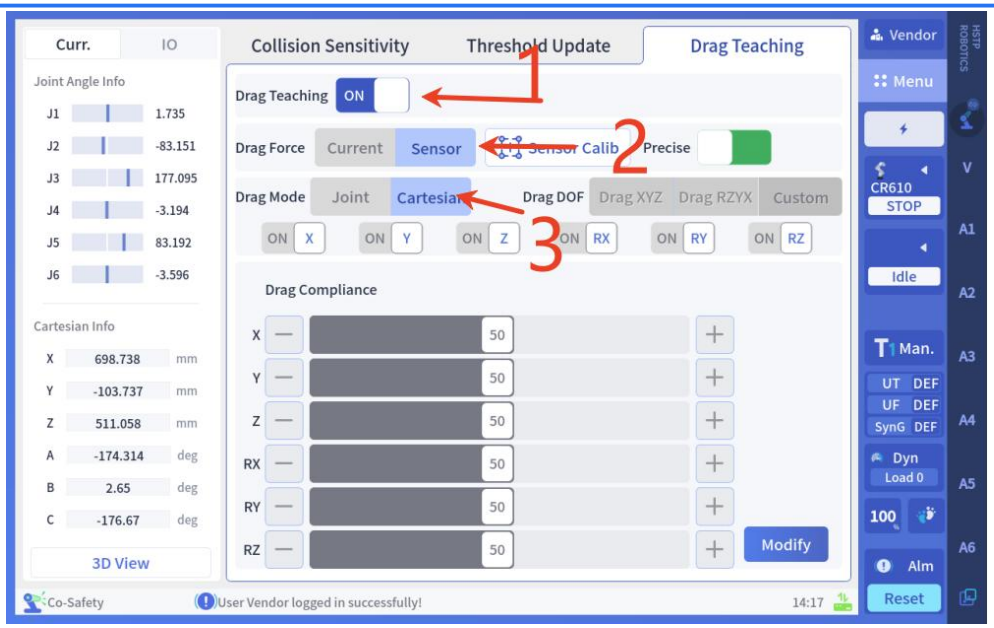


5.1.4.3 Drag the teaching settings

In the main menu, select Function Configuration → Collaboration Security Settings

Operation steps: As shown in the illustration





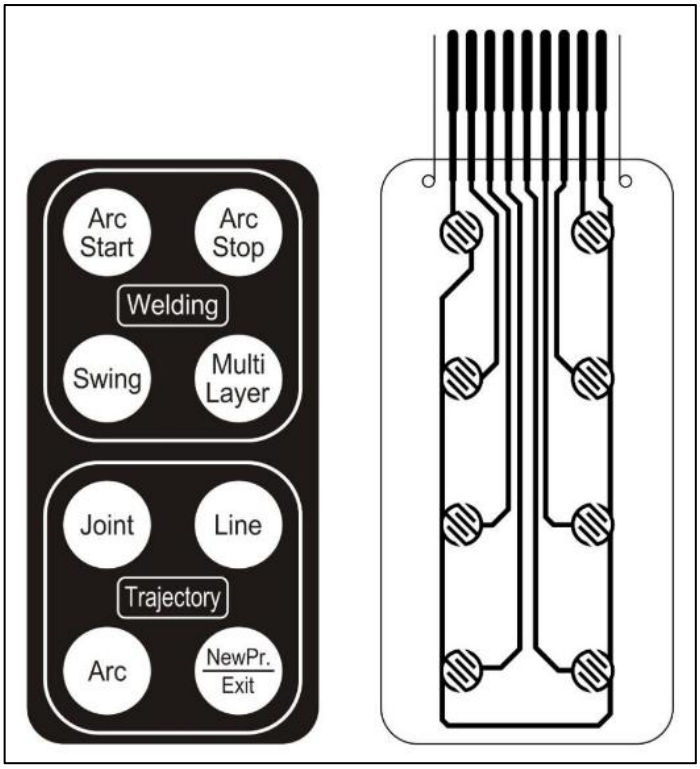
5.1.5 Drag the button

Two modes of dragging:

Flexible drag: Press and hold the flexible drag button for a long time without releasing it to drag the ro bot

Precise drag: Long press the flexible drag button without releasing it. Press the precise drag button onc e to switch to precise drag

5.1.6 Quick programming key



There are eight buttons in total. The mapping to DI is as follows:

Function declaration	Corresponding DI signals
Arc start: Start welding	D12
Arc stop: Stop welding	D13
Swing: Swing welding	D14
Multi-layer welding	D15
Joint: Joint trajectory	D16
Line: Line trajectory	D17
Arc/Arc trajectory	D18
New Pr./Exit: New program/Exit program	D19

When the teaching device is switched to manual mode, that is, T1 or T2 mode, the program editing interface is opened.

1. Press the "Arc Start" button. The DI12 signal will be connected. If the rising edge of DI12 is detected, insert the "Arc Start" command.

2. Press the "Arc Stop" button. The DI13 signal will be connected. If the rising edge of DI13 is detected, insert the "Arc End" command.

3. Press the "Swing" button, and the DI14 signal will be connected. At this point, when the "Arc Start" button is pressed again, the "Swing Welding Start" command and the "Arc Start" command will be inserted simultaneously.

4. "Multi-layer" button: When the " Multi-Layer" button is pressed and the rising edge of DI15 is detected, then press the start "Arc Start" button. When the rising edge of DI12 is detected, "Multi-layer Multi-channel Start" will be inserted into the teaching device. When the "Multilayer" button is pressed and the rising edge of DI15 is detected, then press the start "JS/ End" button. When the rising edge of DI13 is detected, "Multilayer Multi-channel End" is inserted into the teaching device. When the "Multilayer" button is pressed and the rising edge of DI15 is detected, then press the "J/ Joint" button again and the rising edge of DI16 is detected. Insert the "Joint" command with the "MP" attribute into the teaching device, which is the point position with multiple layers and multiple offsets: "Joint P[0] MP", the rest of the straight line and arc commands are used in the same way when combined with this key.

5. Press the "Joint" button. The DI16 signal will be connected. If the rising edge of DI16 is detected, insert the "Joint" command and record the current robot position.

6. Press the "Line" button to connect the DI17 signal. If the rising edge of DI17 is detected, insert the "straight Line" command and record the current robot position.

7. Press the "Arc" button to connect the DI18 signal. If the rising edge of DI18 is detected, insert the "Arc" command and record the current robot position.

8. As long as the robot is not in the "running" state, press and hold the "New Pr./Exit" button for more than 3 seconds. The teaching device will create a new program and enter the editing state.

6 Communications

6.1 MODBUS communication

Modbus is a general-purpose serial communication protocol that works in a master-slave mode. There is only one master device in the whole network, and the other devices are slaves. The master is responsible for initializing the system communication settings and sending information to the slaves, who receive the information and respond to the master's query or make response actions according to the master's message. When the master does not send a request, the slave will not send out data by itself, and there is no direct communication between the slave and the slave.

Modbus supports two kinds of data transmission rules for serial port (RS485) and Ethernet port, i.e. Modbus RTU and Modbus TCP/IP.

In Modbus TCP mode, the master, also known as Client, takes the initiative to connect to the slave; the slave, also known as Server, usually opens a port and waits for the client (master) to connect. Standard Modbus TCP uses port 502 by default.

When using Modbus RTU, you have to set the serial communication parameters (serial port number, baud rate, parity, etc.), and all devices on a Modbus network must select the same transmission mode and serial port parameters.

In versions after V1.6.11. For the Modbus communication function, the robot arm can be both master and slave at the same time, and they do not affect each other. This is because the memory opened by the master and slave functions are independent of each other and can run simultaneously.

When the robot acts as a master, it uses the protocol-less generic Modbus function. The user calls the relevant Modbus function commands inside the program to perform communication and read/write operations on the slave data (sensors).

When the robot arm acts as a slave (server), the system provides Modbus slave control protocol for communication. Modbus slave control protocol is a kind of pre-defined control protocol within the robotic arm controller system, and each register has a clear meaning. The user writes the relevant values into the registers to control the robot arm to perform the corresponding operations. To use the Modbus slave control protocol, you need to enable the Modbus server function switch on the demonstrator/workstation software interface, and then wait for the host computer to connect to establish communication.

ModbusCmd slave control protocol function only supports ModbusTCP. If the end-user's device is a serial port, it must use the gateway to transfer.

The configuration operation and register data protocol of ModbusCmd Slave Control Protocol function are described in the following sections.

6.2 MODBUS master

When the robotic arm is a modbus master, the communication can be realized in the background program. The program example is as follows

```
<program>
LBL[0]
R[0]= OPEN_MODBUS_TCP(1,"10.10.56.112",502,0)       'establish
IF R[0]=-1 THEN
    THROW "MODBUSClient 1 was successfully created, but the connection to slave 1 failed! Reconnect slave 1 when reading or writing registers“
LEVEL=WARNING
END IF

LBL[1]
'******Programming operation that writes the parameters R,DO,DI of the robot arm to the slave station*****************************************************************************
'Write the DO value of the robot arm to the coil of slave 1.
R[0] = WRITE_MDB_COILSTATUS(1,0,3,DO[0])     'Robotic arm DO[0~2] --> Coil COIL[0~2] of the slave station
'Write the R value of the robot arm to the coil of slave 1. Write the R register value to the Boolean coil and the result is “1 if not 0”.
R[1]=3.2
R[2]=0
R[3]=-1
R[0]= WRITE_MDB_COILSTATUS(1,0,3,R[1]) 'Robotic arm R[1~3]--> Coil COIL[0~2] of slave, COIL[0]=1,COIL[1]=0,COIL[2]=1 of slave
```

'Write the DI value of the robot arm to the coil of slave 1.

'Because in general there is no such need scenario. If you really need it, you can assign DI to R hosting first, and then write it.

R[1]=DI[0]

R[2]=DI[1]

R[3]=DI[2]

R[0]= WRITE_MDB_COILSTATUS(1,0,3,R[1]) 'Robotic arm DI[0~2] --> Robotic arm R[1~3]--> Coil COIL[0~2] of slave station

'Write the value of the R register of the robot arm to the save register of slave 1,DATATYPE=0 means that the input register of the slave is 16-bit INT type.

R[5]=5

R[6]=6

R[7]=7

R[0]= WRITE_MDB_HOLDREG(1,0,3,R[5],0) 'Robotic arm R[5~7]--> Coil HOLD_REG[0~2] of slave station

'******Operation of reading register data from a slave station into the robotic arm system**********

'The value of 10 coils starting from 0 will be read from slave 1 and saved in the R,DO,VDI of the robot arm; the user can choose any one of the saving methods

R[0]= READ_MDB_COILSTATUS(1,0,10,R[10]) 'Coil COIL[0~9] of slave --> R[10~19] of robot arm

R[0]= READ_MDB_COILSTATUS(1,0,10,DO[10]) 'Coil COIL[0~9] of slave --> DO[10~19] of robot arm

R[0]= READ_MDB_COILSTATUS(1,0,10,VDI[33]) 'Read PLC's COIL[0~9]--> Save in VDI[33~42]

'Read the value of slave 1, 10 discrete input coils from 0, and save it in the R,DO,VDI of the robot arm.

R[0]= READ_MDB_INPUTSTATUS(1,0,10,R[20]) 'Discrete input InputStaus[0~9] from slave 1 --> R[20~19] from robotic arm

R[0]= READ_MDB_INPUTSTATUS(1,0,10,DO[20]) 'Discrete input InputStaus[0~9] from Slave 1 --> DO[20~29] of robotic arm

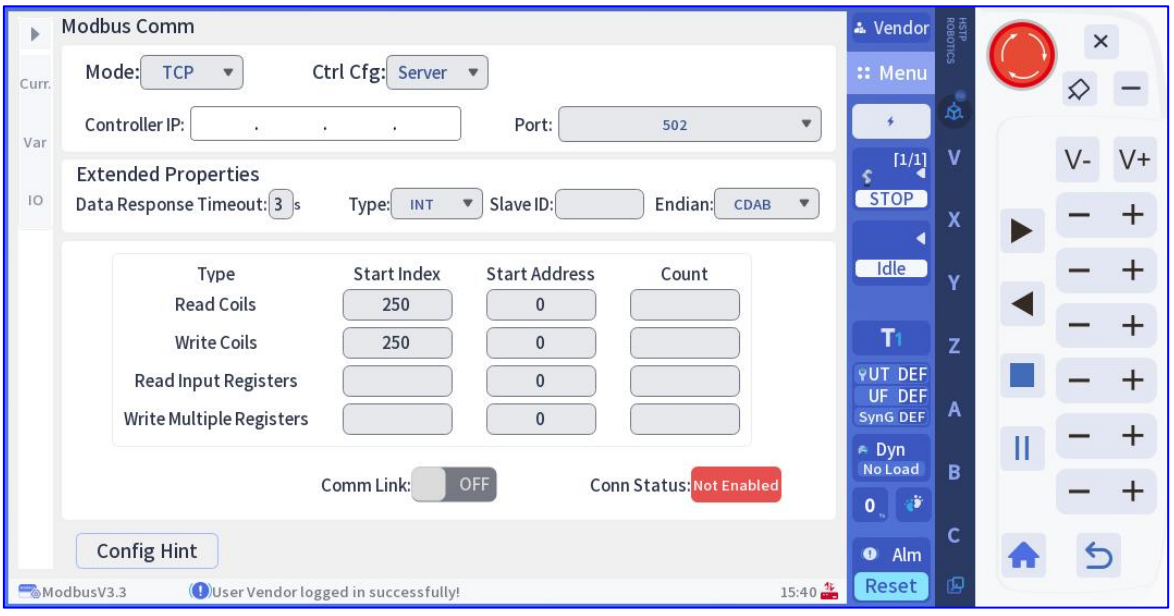
```
'Reads the values of 3 input registers from slave 1, starting from 10, and saves them in the robot arm's R.
DATATYPE=0 means that the slave's input registers are 16-bit INT types.
R[0]= READ_MDB_INPUTREG(1,10,3,R[30],0)      'Slave 1 input register INPUT_REG[10~12] --> R[30~32] of robot arm
'Read the value of slave 1, starting from 10,5 save registers, saved in R of the robot arm.
R[0]= READ_MDB_HOLDREG(1,10,5,R[40],0)      'Save register of slave 1 HOLD_REG[10~14] --> R[40~44] of robotic arm
WAIT TIME=10          'Delay 10ms
GOTO LBL[1]
R[0]= CLOSE_MODBUS_TCP(1)    'Close the connection between the robot arm and slave 1
<end>
```

6.3 MODBUS slave

6.3.1 Configuration interface

When the robot acts as a slave (server) and needs to communicate using the Modbus slave control protocol, you need to enable this function on the demonstrator/workstation software interface first.

Path of Oscillator Software Interface: Main Menu - **[**Communication**]** - **[**MODBUS Configuration**]**.



There are five configuration items on the configuration screen:

(1) Modbus server function switch. Used to turn on and off the Modbus communication function.

(2) Port Settings. Used to configure the communication port. The default value is 507.Setting the port does not allow conflicts with other ports that use the TCP/IP function.

(3) Slave ID number. Used to configure the slave ID (SlaveID) of the robot arm. The default value is 1. Take communication with Siemens PLC as an example, if the PLC writer does not set the slave address of the function block, Siemens PLC may use the broadcast form, so the value of SlaveID must be set to 255. (If communication fails and the alarm slave address is incorrect, you can use the debugging assistant of your computer to connect with the PLC, check the actual message sent by the PLC to determine the actual slave id number, and then fill in the place.)

(4) High-Low Bit. Used to configure the high and low bit order of a 32-bit single precision floating point data consisting of two 16-bit Modbus registers.

(5) Data update cycle. Used to regulate how fast or slow the system data is synchronized. The default value is 50 in ms. the smaller the update period value, the faster the data update, but the greater the CPU performance burden on the controller.

6.3.2 Protocol instructions

6.3.2.1 Coil Status register coilStatus

The CoilStatus register is readable and writable. Function codes: 0x01, 0x05, 0x0f.

Its data protocol is shown in Table 3-1.

Table 2-1 Coil status register data protocol

CoilStatus Address (10 scale)	Data Type	element	note
0	Bit	DO[0]	Output IO
1	Bit	DO[1]	
2	Bit	DO[2]	
3	Bit	DO[3]	
...	Bit	...	
63	Bit	DO[63]	
64	Bit	VDI[0]	The value of the VDI variable, i.e. the value of the virtual input IO. If users need to load or run the

			program in external mode, they must first configure the IOs in the Run Configuration interface on the software interface of the Teachers/Workstations, and bind the IOs to the externally-controlled signals (here, it should be noted that the bound signals need to be virtual signals, i.e., the bound signals are those that are displayed as VIRTUAL on the Digital Input/Output interface). After the signal binding is completed, the state of the program can be controlled by writing the value of VDI[x] through ModbusCmd.
65	Bit	VDI[2]	
...	Bit	...	
96	Bit	VDI[32]	
...	Bit	...	
127	Bit	VDI[63]	

6.3.2.2 Discrete Input inputStatus

Discrete Input (InputStatus) read-only. Function code: 0x02.

Its data protocol is shown in Table 3-2.

Table 2-2 Discrete Input Data Protocol

InputStatus Address (decimal system)	data type	content	notes
0	Bit	DI[0]	Real input IO status
1	Bit	DI[1]	
2	Bit	DI[2]	
...	Bit	...	
63	Bit	DI[63]	

6.3.2.3 Input register inputReg

The InputReg is read-only. Function code: 0x04.

Its data protocol is shown in Table 3-3.

Table 2-3 Input Register Data Protocols

InputStatus Address (decimal system)	data type	content	notes
1	byte	Alarm status	0: No alarm 1: Emergency stop alarm 2: Servo alarm 3：Forward over acceleration 4：Axis reaches limit 5：Forward looking over limit 6：Position unreachable 7：Other alarms
2	byte	Enable state	0: Not enabled 1: Enabled

3	byte	The current operating mode of the robot	0: None 1: Manual T1 mode 2: Manual T2 mode 3: Automatic mode 4: External mode
4	byte	Tool number currently called	Default: -1 to 15
5	byte	Tool number currently called	Default: -1 to 15
6	byte	The coordinate system of the current robot's point motion	0: None 1: Axis coordinate system 2: World coordinate system 3: Base coordinate system 4: Workpiece coordinate system (user coordinate system) 5: Tool coordinate system
7	(Double Word) Floating Point Data	J1 Low position coordinates	Current joint position value
8		J1 Coordinate high position	
9	(Double Word) Floating Point Data	J2 Low position coordinates	
10		J2 Coordinate high position	
11	(Double Word) Floating Point Data	J3 Low position coordinates	
12		J3 Coordinate high position	

13	(Double Word) Floating Point Data	J4 Low position coordinates	
14		J4 Coordinate high position	
15	(Double Word) Floating Point Data	J5 Low position coordinates	
16		J5 Coordinate high position	
17	(Double Word) Floating Point Data	J6 Low position coordinates	
18		J6 Coordinate high position	
19	(Double Word) Floating Point Data	E1 Low position coordinates	
20		E1 Coordinate high position	
21	(Double Word) Floating Point Data	E2 Low position coordinates	
22		E2 Coordinate high position	
23	(Double Word) Floating Point Data	E3 Low position coordinates	
24		E3 Coordinate high position	
25	(Double Word) Floating Point Data	X Low position coordinates	The Cartesian coordinate position value of the robot in the current tool/workpiece coordinate system

26		X Coordinate high position	
27	(Double Word) Floating Point Data	Y Low position coordinates	
28		Y Coordinate high position	
29	(Double Word) Floating Point Data	Z Low position coordinates	
30		Z Coordinate high position	
31	(Double Word) Floating Point Data	A Low position coordinates	
32		A Coordinate high position	
33	(Double Word) Floating Point Data	B Low position coordinates	
34		B Coordinate high position	
35	(Double Word) Floating Point Data	CLow position coordinates	
36		C Coordinate high position	
37	(Double Word) Floating Point Data	E1 Low position coordinates	
38		E1 Coordinate high position	
39	(Double Word)	E2 Low position coordinates	

40	Floating Point Data	E2 Coordinate high position	
41	(Double Word) Floating Point Data	E3 Low position coordinates	
42		E3 Coordinate high position	
43	(Double Word) Floating Point Data	J1 Low joint velocity	Current actual joint velocity value
44		J1 High joint velocity	
45	(Double Word) Floating Point Data	J2 Low joint velocity	
46		J2 High joint velocity	
47	(Double Word) Floating Point Data	J3 Low joint velocity	
48		J3 High joint velocity	
49	(Double Word) Floating Point Data	J4 Low joint velocity	
50		J4 High joint velocity	
51	(Double Word) Floating Point Data	J5 Low joint velocity	
52		J5 High joint velocity	
53	(Double Word)	J6 Low joint velocity	
54		J6 High joint velocity	

	Floating Point Data		
55	(Double Word) Floating Point Data	E1 Low joint velocity	
56		E1 High joint velocity	
57	(Double Word) Floating Point Data	E2 Low joint velocity	
58		E2 High joint velocity	
59	(Double Word) Floating Point Data	E3 Low joint velocity	
60		E3 High joint velocity	
61	(Double Word) Floating Point Data	J1 The actual current value of the joint is low	Actual value of current joint current
62		J1High actual joint current value	
63	(Double Word) Floating Point Data	J2 The actual current value of the joint is low	
64		J2 High actual joint current value	
65	(Double Word) Floating Point Data	J3 The actual current value of the joint is low	
66		J3 High actual joint current value	
67	(Double Word)	J4The actual current value of the joint is low	

68	Floating Point Data	J4 High actual joint current value	
69	(Double Word) Floating Point Data	J5 The actual current value of the joint is low	
70		J5 High actual joint current value	
71	(Double Word) Floating Point Data	J6 The actual current value of the joint is low	
72		J6 High actual joint current value	
73	(Double Word) Floating Point Data	E1 The actual current value of the joint is low	
74		E1 High actual joint current value	
75	(Double Word) Floating Point Data	E2 The actual current value of the joint is low	
76		E2 High actual joint current value	
77	(Double Word) Floating Point Data	E3 The actual current value of the joint is low	
78		E3 High actual joint current value	
79	(Double Word) Floating Point Data	Current end TCP movement speed	The synthesis speed of XYZ
80			
81	(Double Word)	Current end TCP movement speed	The synthesis speed of ABC
82			

	Floating Point Data		
83	byte	Motion status of robot axis group	0: Initial value 1: Not enabled 2: Stop in progress 3: During exercise 4: Alarm
84	byte	Robot motion program status	0: None 1: Not loaded 2: Already loaded, in preparation 3: Running 4: Pause in progress 5: Error occurred

6.3.2.4 Hold register

Hold registers are readable and writable. Function codes: 0x03, 0x06, 0x10.

The data protocol is shown in Table 3-4

Table 2-4 Maintain Register Data Protocol

HoldReg Address (10 scale)	data type	content	notes
0	Bit	Clear alarm	Rising edge trigger
1	Bit	Enable switch	0: Disconnect Enable 1: Up enable
2	Bit	Operation mode setting	0: None 1: Manual T1 mode 2: Manual T2 mode 3: Automatic mode 4: External mode

3	Bit	Robot manual operation speed	1 to 100 (unit:%)
4	Bit	Robot automatic running speed	1 to 100 (unit:%)
5	Bit	Point motion coordinate system setting	0: None 1: Axis coordinate system 2: World coordinate system 3: Base coordinate system 4: Workpiece coordinate system (user coordinate system) 5: Tool coordinate system
6	Bit	Perform jog based on the currently set coordinate system	0: Stop jogging 1: J1 (X) axis jog forward 2: J1 (X) axis jog in reverse direction 3: J2 (Y) axis jog forward 4: J2 (Y) axis jog reverse 5: J3 (Z) axis point motion positive direction 6: J3 (Z) axis point motion reverse 7: J4 (A) axis jog forward 8: J4 (A) axis jog in reverse direction 9: J5 (B) axis jog forward 10: J5 (B) axis jog in reverse direction 11: J6 (C) axis jog forward 12: J6 (C) axis jog reverse

			13: E1 external axis jog forward 14: E1 External axis jog reverse 15: E2 external axis jog forward 16: E2 External Axis Point Movement Reverse 17: E3 external axis jog forward 18: E3 External Axis Point Movement Reverse
7		Tool coordinate system number setting	0: No operation 1: Select tool coordinate -1 (default) 2: Select tool coordinate 0 3: Select tool coordinate 1 ... 17: Select tool coordinates 15
8		Setting of workpiece coordinate system number	0: No operation 1: Select Workpiece Coordinate-1 (default) 2: Select workpiece coordinate 0 3: Select workpiece coordinate 1 ... 17: Select workpiece coordinates 15
9	Bit	Retrieve the current joint position and save it to the specified JR [x] register	0: No operation 1: Retrieve joint coordinate values and store them in JR [0] 2: Retrieve joint coordinate

			values and store them in JR [1] ... 1000: Obtain joint coordinate values and store them in JR [999]
10	Bit	Retrieve the current Cartesian position and save it to the specified LR [x] register	0: No operation 1: Retrieve Cartesian coordinate values and store them in LR [0] 2: Retrieve Cartesian coordinate values and store them in LR [1] ... 1000: Obtain Cartesian coordinate values and store them in LR [999]
11	Bit	Joint motion to point, joint motion to designated JR [x] register	0: No operation 1: Joint motion to point JR [0] 2: Joint motion to point JR [1] ... 1000: Joint motion to point JR [999]
12	Bit	Straight line motion to point, straight line motion to designated LR [x] register	0: No operation 1: Linear motion to point LR [0] 2: Linear motion to point LR [1] ... 1000: Linear motion to point LR [999]
13	Bit	The index of the R register used by the program configuration	The index number of the program mapping register. For example, if it is R [10], fill in 10

		module in the running configuration function	here
14	Bit	The value of the R register used by the program configuration module in the running configuration function	The value of the program mapping register. For example, if R [10] is configured, when 1 needs to be written into R [10], fill in 1 here
15	Bit	reserve	
16	Bit	reserve	
17	Bit	reserve	
18	Bit	reserve	
19	Bit	reserve	
20	Bit	reserve	
21	byte	(Double Word) Floating Point Data	R[0] data-low
22	byte		R[0] data-high
23	byte	(Double Word) Floating Point Data	R[1] data-low
24	byte		R[1] data-high
...	byte	(Double Word) Floating Point Data	R[...] data-low
...	byte		R[...] data-high
59	byte	(Double Word) Floating Point Data	R[19] data-low
60	byte		R[19] data-high
61	byte	16 bit unsigned shaping	R[20](Only supports read and write within the range of 0-65535)
62	byte	16 bit unsigned shaping	R[21](Only supports read and write within the range of 0-65535)

63	byte	16 bit unsigned shaping	R[22](Only supports read and write within the range of 0-65535)
...	byte	16 bit unsigned shaping	R[...](Only supports read and write within the range of 0-65535)
120	byte	16 bit unsigned shaping	R[79](Only supports read and write within the range of 0-65535)

Tip:

The robot arm can be master and slave at the same time.

CompatibilityInstructions: Since V1.6.11 version, we recommend users to use the way of programming instruction, modbus communication process package is supported in low version.

6.4 SOCKET communication

Socket communication can be implemented in the background program. The robotic arm can be used as a client or a server, and when used as a server it can be realized as a pair of multiple clients.

6.4.1 SOCKET single client

When the robotic arm is a socket client, the communication can be implemented in the background program. The program example is as follows:

```
<program>
R[0]=-1
R[0]=OPENSOCKET OPTIONS=0 HANDLE=1    ' Create a socket with a client handle of 1.
IF R[0]<>0 THEN
THROW "Failed to create client！" LEVEL=WARNING
GOTO LBL[100]       'The communication operation is terminated and program execution is completed.
END IF
```

```
LBL[2]
R[0] = CONNECT(1,"10.10.56.112",30000)      'Connecting to Visual, IP: 10.10.56.112 ,port number is 30000.
do not set the timeout time
IF R[0]<>0 THEN     ' The connection was unsuccessful, probably because the visualization host server is not
yet open
WAIT TIME=3000      ' Delay 3s, then reconnect
GOTO LBL[2]          'Reconnecting the visualization host computer
END IF
SOCKETMONITOR(1,2)   'Setting up to throw only warnings after disconnection, PRG does not stop

R[1]=ISHANDLEOPEN(1)
WHILE R[1]=1
      R[2]= BRECBUFEMPTY(1)         ' Determine if there is data in the receive buffer
      IF R[2]=1 THEN                    ' The RECEIVE function is called only when there is data in the cache.
           RECEIVE(1,SR[0])          ' Receive the data sent by the vision and save it in SR[0].
      END IF
      IF SR[0]<>"" THEN        ' Visual send T1,1.123,2.0,3.0, string; cut to SR[1]=T1 SR[2]=1.123 SR[3]=2.0
SR[4]=3.0
          R[0] = FINDSTR(SR[0],",")              ' SR[0]=T1,1.123,2.0,3.0 R[0]=3
          SR[1]= SUBSTR(SR[0],0,R[0]-1)       ' Intercept T1 on the left, SR[1] = T1
          SR[0]= SUBSTR(SR[0],R[0]+1)          ' Intercept the right part of the string, after the interception
SR[0]=1.123,2.0,3.0
          R[0] = FINDSTR(SR[0],",")         ' SR[0]=1.123,2.0,3.0
          SR[2]= SUBSTR(SR[0],0,R[0]-1)   '    Intercept 1.123 on the left, SR[2] = 1.123
          SR[0]= SUBSTR(SR[0],R[0]+1)      ' Intercept the right part of the string, after the interception
SR[0]=2.0,3.0

           R[0] = FINDSTR(SR[0],",")         ' SR[0]=2.0,3.0
          SR[3]= SUBSTR(SR[0],0,R[0]-1)   ' Intercept left 1.123 SR[3] = 2.0
```

```
        SR[0]= SUBSTR(SR[0],R[0]+1)      ' Intercept the right part of the string, SR[0] = 3.0 after interception


        SR[4]=SR[0]            ' SR[4]=3.0
         SR[0]=""              'empty
         SEND(1,"OK")          'Give the visual uplink OK
       'convert to digital
         R[20]=TOVAL(SR[2])
         R[21]=TOVAL(SR[3])
         R[22]=TOVAL(SR[4])
      END IF
END WHILE
THROW "Server shutdown! Reconnect" LEVEL=WARNING
GOTO LBL[2]
LBL[100]
CLOSESOCKET HANDLE=1       'Close Sockets
<end>
```

6.4.2 SOCKET server

When the robotic arm is used as a Socket server, the communication can be implemented in the background program. The program example is as follows:

```
<program>
R[0]=OPENSOCKET OPTIONS=0 HANDLE=1     'Create creates a server and listens for client connections with handle 1.
IF R[0]<>0 THEN      ' If the creation of the server is unsuccessful, an alarm is thrown, and then the program ends and the program changes to the READY state.
  THROW "Failed to create server!" LEVEL=ERROR
  GOTO LBL[100]      ' Ends the communication operation.
END IF
R[1]=-1              ' R[1] is used to save the handle for communication with the PLC. The unconnected state returned when the ISHANDLEOPEN() function is executed for the first time.
```

```
WHILE 1=1            ' Cyclic listening code, PLC can be disconnected and reconnected to the arm.
    R[10]=ISHANDLEOPEN(R[1])      ' Determining whether the PLC is connected
    IF R[10]=0 THEN      ' 。The PLC is not connected and listens for connections, if the PLC is already connected, there is no need to listen in the call to the ACCEPT function.
        'Listen to the connection, if the PLC connects to the robotic arm, save the fd of communication with the PLC in R[1], and return after 5S of listening.          R[0]=ACCEPT(1,"10.10.56.112",30000,R[1],5000)
    END IF
    R[10]=ISHANDLEOPEN(R[1])
    IF R[10]=1 THEN                    ' R[10]=1 indicates a successful connection to the PLC.
        R[11]= BRECBUFEMPTY(R[1])     ' Determine if there is data in the receive buffer
        IF R[11]=1 THEN                ' The RECEIVE function is called only when there is data in the cache.
            RECEIVE(R[1],SR[0])        ' Receive data sent by the vision.
        END IF
        IF SR[0]<>"" THEN        ' Visual send T1,1.123,2.0,3.0 string; cut to SR[1]=T1 SR[2]=1.123 SR[3]=2.0 SR[4]=3.0
            R[0] = FINDSTR(SR[0],",")            ' R[0]=3    SR[0]=T1,1.123,2.0,3.0 R[0]=3
            SR[1]= SUBSTR(SR[0],0,R[0]-1)       ' Intercept T1 on the left, SR[1] = T1
            SR[0]= SUBSTR(SR[0],R[0]+1)          ' Intercept the right part of the string, after the interception SR[0]=1.123,2.0,3.0
            R[0] = FINDSTR(SR[0],",")         ' SR[0]=1.123,2.0,3.0
            SR[2]= SUBSTR(SR[0],0,R[0]-1)    ' Intercept left 1.123 SR[2]=1.123
            SR[0]= SUBSTR(SR[0],R[0]+1)      ' Intercept the right part of the string, after the interception SR[0]=2.0,3.0

             R[0] = FINDSTR(SR[0],",")          ' SR[0]=2.0,3.0
            SR[3]= SUBSTR(SR[0],0,R[0]-1)     ' Intercept 2.0 on the left, SR[3] = 2.0
            SR[0]= SUBSTR(SR[0],R[0]+1)       ' Intercept the right part of the string, SR[0] = 3.0 after interception
            SR[4]=SR[0]              'SR[4]=3.0
```

```
            SR[0]=""                    'empty
            SEND(R[1],"OK")             'Give the visual uplink OK
            'convert to digital
            R[20]=TOVAL(SR[2])
            R[21]=TOVAL(SR[3])
            R[22]=TOVAL(SR[4])
        END IF
    END IF
    WAIT TIME=10      ' Add 10ms delay to dead loop
END WHILE
R[0]=CLOSESOCKET HANDLE=R[1]   ' After jumping out of the loop, close the socket for communication with the PLC.
LBL[100]
CLOSESOCKET HANDLE=1          ' Close the listening socket
<end>
```

6.4.3 SOCKET server-side one-to-many

```
<program>
  R[0]=OPENSOCKET OPTIONS=0 HANDLE=1     'Create creates a server and listens for client connections with handle 1.
  IF R[0]<>0 THEN
    THROW "Failed to create server!" LEVEL=WARNING
    GOTO LBL[100]      'Ends the communication operation.
  END IF
  R[1]=-1      'Reset first. and R[1] is used to save the communication handle of the first client.
  R[2]=-1      'Reset first. and R[2] is used to save the communication handle of the second client.
  WHILE 1=1     ' Loop listening code, client can disconnect and reconnect with server (robot arm)
    R[10]=ISHANDLEOPEN(R[1])
    IF R[10]=0 THEN          ' If it is not connected, it listens for connections. No need to listen and accept the value of fd if it is already open normally, otherwise it will overwrite the value of R[1].
```

```
    ACCEPT(1,"10.10.57.213",30000,R[1],100)      ' The fd that communicates with the first client is stored in R[1], and is returned after a timeout of 100ms.
    END IF
    R[10]=ISHANDLEOPEN(R[1])
    IF R[10]=1 THEN                    ' R[10]=1 indicates that the handle is open and communication with the first client is started.
      SEND(R[1],"hello world")
      R[11]= BRECBUFEMPTY(R[1])   ' Determine if there is data in the receive buffer
        IF R[11]=1 THEN             ' The cache receives the data only when it's available.
        RECEIVE(R[1],SR[1])
      END IF
     '...          ' Business logic judgment is done here, character cutting and so on.
    END IF

    R[10]=ISHANDLEOPEN(R[2])
    IF R[10]=0 THEN      'If it is not connected, it listens for connections. No need to listen and accept the value of fd if it is already open normally, otherwise it will overwrite the value of R[2].
     ACCEPT(1,"10.10.57.213",30000,R[2],100)      ' The fd for communicating with the second client is stored in R[2], and is returned after a listening timeout of 100ms.
    END IF
    R[10]=ISHANDLEOPEN(R[2])
    IF R[10]=1 THEN                ' R[10]=1 indicates that the handle is open and communication with the second client is started.

      SEND(R[2],"hello world")
      RECEIVE(R[2],SR[2])
     '...          ' Business logic judgment is done here, character cutting and so on.
      END IF
      WAIT TIME=10      ‘ Delay 10ms
  END WHILE
```

```
  R[0]=CLOSESOCKET HANDLE=R[1]    ' After jumping out of the loop, close the socket that is
communicating with Client 1.
  R[0]=CLOSESOCKET HANDLE=R[2]    ' After jumping out of the loop, close the socket that is
communicating with Client 2.
  LBL[100]
  CLOSESOCKET HANDLE=1          ' Close the listening socket
  <end>
```

7 Programming instructions

7.1 Summarize

Terminology and Concepts:

- **program file**

A program, or PRG program, is a text file with a ".PRG" extension. A program file is a collection of system function instructions and statements that can be recognized and executed by the system.

The program file name supports numbers, letters, underscores, and Chinese characters. The program file name should be within 30 bytes, and the program name is case sensitive.

The program file structure is divided into three modules: document attribute module, variable declaration module, program body module. Specific structure and Instructions

As shown in Figure 1-1 and Table 1-1.

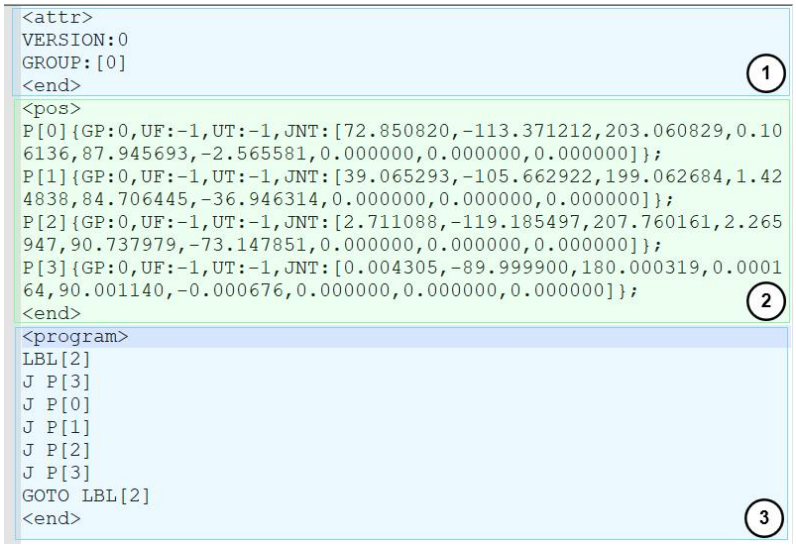


Figure 1-1 Schematic diagram of program file structure

Table 1-1 Instructions for Program File Structure Modules

Label Item	Instructions
1	Document attribute module. Used to record versions and the axis groups bound to them
2	Variable declaration module. Used to define and declare positional variables

3	Main module of the program. Used to add instructions and statements to implement programming logic

- **Main program and subroutines**

  The system supports PRG programs to call other PRG programs, with the main program as the caller and the called program as the subroutine. When calling, use the CALL instruction to make the call. Grammar example: CALL "program name". When the main program reaches this point, it will jump into the program content of the executing subroutine, and after execution, it will return to the main program to continue executing.

- **Front end program and backend program**

  The front-end program, also known as the main motion program, refers to a program with a group mask of 0 and motion instructions.

  A background program, also known as a RUN program, refers to a program with a group mask of * that does not allow motion instructions and is only used for logical processing.

  The system supports running both a front-end program and a back-end program simultaneously, and the two programs can run independently to meet the programming requirements of multitasking.

- **Loading**

  Loading refers to the operation of loading the contents of a program file into memory, and after the program is loaded, it can be executed.

- **Unload**

  Uninstalling refers to the operation of stopping the program from running and clearing its contents from memory.

- **Code annotation**

  Use single quotation marks ['] as comments for program lines in the program.

- **Line break connector**

  Use a reverse slash [\] as the line connector. The line connector is used when a single instruction line is too long to write, written at the end of the program line to connect adjacent lines of the program and concatenate them together to form a complete line of the program.

  For example, there are two lines in the program:

  A= B+ \

  C+D

When executed, it is equivalent to one line of program: A=B+C+D。

- **Instruct**

The instructions of the Type 3 control system can only be capitalized.

The system supports two basic data types: floating-point and string. Integer values will be implicitly converted to floating-point values. Boolean and floating-point conversions follow the mandatory conversion rule of 'non-zero then 1'. Boolean numbers only support ON and OFF, and currently do not support TRUE and False.

7.2 Variable

7.2.1 Global variable

At present, the III system does not support custom variables, but the system predefines digital input/output IO, analog input/output IO, and six global register variables: R SR、JR、LR、UT、UF。 Users can use these global registers in program programming for data computation and storage.

- DI [x]: Digital input IO, boolean type, the system defines 512, namely DI [0]~DI [511]. VDI variables are virtual IO inputs used to switch the state of read-only variable DI [x] and implement virtual inputs.
- DO [x]: Digital output IO, boolean type, the system defines 512, namely DO [0]~DO [511].
- AI [x]: Analog input IO, float type, the system defines 16, namely AI [0]~AI [15].
- AO [x]: Analog output IO, float type, the system defines 16, namely AO [0]~AO [15].
- R [x]: A general-purpose double precision floating-point register variable, of type double, capable of accessing integer and floating-point data. The quantity is 2000, namely R [0]~R [1999]. The precision of variable values is up to 7 decimal places, and if exceeded, it will be rounded off.
- SR [x]: A general-purpose string type register variable, each variable can store 512 characters, with a total of 100 characters, i.e. SR [0]~SR [99].
- JR [x]: A universal joint point register variable used to access joint points. The quantity is 1000, namely JR [0]~JR [999].
- LR [x]: A general-purpose spatial point register variable used for accessing spatial points. The quantity is 1000, namely LR [0]~LR [999].
- UT [x]: Tool coordinate system register variable, used to store the values of the tool coordinate system. The quantity is 16, namely UT [0]~UT [15].
- UF [x]: Workpiece coordinate system register variable, used to store the values of the workpiece coordinate system. The quantity is 16, namely UF [0]~UF [15].

Due to the fact that system variables are all global variables, the system will pre occupy them when implementing functions. Please refer to Appendix 32.2 "System Signal Occupancy Table" for details.

7.2.2 Program point variable

The program point is a point variable within the "<pos>...<end>" section of the PRG program. When the user obtains the point, the system will automatically add the point and record it in the section. The system supports two types of points: joint points and spatial points. The identification of joint points is JNT, and the identification of spatial points is LOC. When the program uses a spatial point, the tool part number specified in that point must be called, otherwise an alarm will be triggered indicating a mismatch in the coordinate system of the point.

The following example records P [0] as the joint point position and P [1] as the spatial point position:

```
<pos>
    P[0]{GP:0,UF:-1,UT:-1,JNT:[3.126, -81.027, 29.643, 0.377, 82.950, -53.426, 0.0, 0.0, 0.0]};
    P[1]{GP:0,UF:1,UT:1,CFG:[0, 0, 0, 0, 0,0],LOC:[98.90, -994.70,296.30, -98.30, -3.60, -114.27, 0.0, 0.0,
0.0 ]};
<end>
<program>
    J P[0]
    UFRAME_NUM =1
    UTOOL_NUM=1
    L P[1]
    ...
    <end>
```

7.2.2.1 Point variable component

The positional variable is a composite data structure, and the corresponding component can be obtained through the square brackets [] to manipulate the value of the corresponding bit. A positional variable contains data for 9 axes, therefore the subscripts are 0~8. The component data of the point is floating-point data, which supports operations such as addition, subtraction, multiplication, and division.

Example:

```
R[1]=JR[1][0]
```

```
LR[1][1]=R[0]*10
P[0][2]=P[0][2]+10
```

7.3 Point operation

Point variables can be understood as vectors, supporting operations such as addition, subtraction, multiplication, division, and remainder of vectors.

Example:

```
LR[1]=LR[2]+LR[3]
```

The above statement adds the components of LR [2] and LR [3] and assigns them to the components of LR [1]. Similarly, support operations such as LR [1]=LR [2] - LR [3], LR [1]=LR [2] * LR [3], etc.

7.4 R register bit operation

The R register stores an 8-bit signed integer, and users can obtain the value of each bit of the binary number corresponding to the integer through the dot. Due to the floating point type of the R register, if the actual value in the R register is not an integer, the system will default to rounding down to an integer before performing a bit operation.

Grammar format: R[x].y

Example:

```
IF R[1].0=ON THEN       'Obtain the value of the 0th digit of R [1]
...
ENDIF
R[1].0=0        'Set the 0th value of R [1] to 0
```

7.5 Operator

The operators supported by the current system are as follows:

Table 3-1 Operators Supported by the System

Operator category	Operator	Describe	Support data types	Driority	Example
Arithmetic operator	+	Addition operation	INT、DOUBLE、STRING、POINT	3	7=5+2
	-	Subtraction	INT、DOUBLE、	3	3=5-2

		operation	POINT		
	*	Multiplication operation	INT、DOUBLE、POINT	2	10=5*2
	/	Division operation	INT、DOUBLE、POINT	2	2.5=5/2
	DIV	The integer whose quotient is obtained by division operation	INT、DOUBLE、POINT	2	2=5 MOD 2
	MOD	Division operation takes remainder (modulo)	INT、DOUBLE、POINT	2	1=5 MOD 2
Logical operator	AND	Logical union (one false is false)	BOOL	5	R[0]=R[1] AND R[2]
	OR	Logic or (one truth is true)	BOOL	5	R[0]=R[1] OR R[2]
	NOT	Logical inversion	BOOL	1	R[0]=NOT R[2]
Comparison operator	=	Equal to	INT、DOUBLE、POINT	4	1=1
	<>	Not equal to	INT、DOUBLE、POINT	4	1<>2
	<	Less than	INT、DOUBLE	4	1<2
	>	Greater than	INT、DOUBLE	4	2>1
	<=	Less than or equal	INT、DOUBLE	4	1<=1
	>=	Greater than or equal	INT、DOUBLE	4	1>=1

Tips:

(1) When using logical operators, floating-point data is implicitly converted to Boolean data, i.e. 'non-zero then 1'.

(2) Combine operators of the same priority from left to right.

(3) NOT: The logical operation is reversed. The highest priority.

7.6 Statement

7.6.1 Process statement

7.6.1.1 LBL[] | GOTO LBL[]

The LBL [] tag statement is used to set the tag position, and the GOTO statement is used to jump the program to the specified tag position. If the GOTO keyword is to be used, the label LBL must be defined in the program first, and GOTO and LBL must be in the same program block.

Grammar format:

LBL“[”<unsigned int num>“]”

...

GOTO LBL“[”<unsigned int num>“]”

Example:

```
LBL[1]
J P[1] VEL=50
J P[2] VEL=50
GOTO LBL[1]
```

Execution Logic Manual: After the robot joints sequentially to P [1] and P [2], it executes GOTO LBL [1], then jumps to LBL [1], and then joints sequentially to P [1] and P [2], continuously reciprocating these two points.

7.6.1.2 CALL

The CALL instruction is used for calling subroutines and executing their program content. Program calls support multiple layers of nesting, up to a maximum of 12 layers.

Grammar format:CALL <string name>

Example, Main Program MAIN PRG calls subroutineSON.PRG：

```
Here is the MAIN.PRG (Main Program)
<program>
    J JR[1] VEL=50
    J JR[2] VEL=50
    CALL "SON.PRG"    'Call subroutine
```

```
    J JR[3] VEL=50
<end>


Here is the SON.PRG (subroutine)
<program>
    DO[1]=ON
    WAIT TIME 500
    DO[1]=OFF
<end>
```

Execution logic Instructions: The robot moves to JR [1], JR [2] in turn, then calls the subprogram, executes DO [1]=ON, after forced waiting for 500 milliseconds, executes DO [1]=OFF, then returns to the main program, the joint moves to JR [3], and completes program execution.

If the subroutine is in another file directory, when the CALL instruction is called, the file directory name must be added, such as CALL "FLOD1/SON. PRG"

7.6.2 Conditional statement

7.6.2.1 IF… GOTO LBL[]

If the condition is met, jump to the corresponding LBL tag for execution; If the condition is not met, the program blocks following IF will be executed sequentially.

Grammar format:

IF <bool expression>, GOTO LBL“[”<unsigned int num>“]”

Example:

```
    IF DI[1]=ON, GOTO LBL[1]     'If the condition is true, jump directly to LBL
[1] for execution; If the condition is false, execute in sequence downwards
    J P[1] VEL=50
    J P[2] VEL=50
    LBL[1]
    VDI[1]=OFF
    ……
```

7.6.2.2 IF… CALL

If the condition is met, the code content of the subroutine will be called and executed before proceeding in sequence; If the condition is not met, execute the next line of instruction content of the IF instruction and ignore the subroutine call.

Grammar format:

IF <bool expression>, CALL<string name>

Example:

```
IF DI[1]=ON, CALL "TEST.PRG"
J JR[1] VEL=50
DO[1]=OFF
```

Execution Logic Manual:

If DI [1]=ON condition is met, execute TEST The program content of the PRG subroutine, after execution, proceeds to execute JJR [1] VEL=50 and subsequent instructions; If the conditions are not met, ignore TEST The subroutine call of PRG directly executes JJR [1] VEL=50 and subsequent instructions.

7.6.2.3 IF... THEN... ENDIF

IF THEN needs to be used in conjunction with END IF instruction, IF... GOTO, IF... CALL instructions do not need to be used in conjunction with END IF. IF THEN can be used in conjunction with the ELSE instruction to execute the ELSE branch when the condition is not met.

Grammar format:

IF <bool expression> THEN

...

[ELSE

...]

ENDIF

Example 1，No ELSE branch:

```
IF DI[1]=ON THEN    'If DI [1]=ON is satisfied, execute the JP [1] instruction; otherwise, do not execute it
J P[1]
END IF
```

Example 2，Used in conjunction with ELSE branch:

```
IF DI[1]=ON THEN    'If DI [1]=ON is satisfied, execute J P [1]; otherwise, execute J P [2]
J P[1]
ELSE
J P[2]
END IF
```

7.6.2.4 SELECT

The SELECT statement compares the value stored in the register selected by SELECT with the value after CASE. If they are equal, execute the CASE flow instruction in that row, and at the same time, the SELECT statement block will pop up. The CASE and ELSE instructions will no longer be executed; If they are not equal, the corresponding CASE for that line will not be executed, and the subsequent CASE option judgment will continue until the entire statement block is completed.

Grammar format:

SELECT <num expression>

CASE<num expression>, (GOTO LBL“[”<unsigned int num>“]”)|(CALL <string name>)

[{CASE<num expression>, (GOTO LBL“[”<unsigned int num>“]”)|(CALL <string name>)}

ELSE, (GOTO LBL“[”<unsigned int num>“]”)|(CALL <string name>)]

Example:

```
SELECT R[0]
CASE 1, GOTO LBL[1]
CASE 2, GOTO LBL[2]
```

```
    CASE 1, CALL"RT.PRG"      'This is a repeated CASE situation. If R
[0]=1, then only execute the previous LBL [1] once
    ELSE, CALL"HS.PRG"     'The ELSE instruction is optional, can be written
or not written
    GOTO LBL[4]
    LBL[1]
    J P[1]
    GOTO LBL[4]
    LBL[2]
    J P[2]
    GOTO LBL[4]
    LBL[3]
    J P[3]
    LBL[4]
```

Execution Logic Manual:

The above program is executed from top to bottom to match R [0]. When R [0]=1, the program moves to point P [1]; When R [0]=2, the program moves to point P2; When R [0] is not equal to 1 or 2, the program executes the subroutine HS of the ELSE instruction part The content of PRG is executed in descending order.

7.6.3 Loop statement

7.6.3.1 FOR... END FOR

The FOR statement needs to define the initial value, final value, and step value of a variable (BY expression is optional, if not written, it defaults to increasing by 1 each time). The system will determine whether the value of the loop variable is less than or equal to the final value. If it is less than or equal to the final value, the loop will be executed. Otherwise, the loop will exit. FOR and END FOR form a loop statement block.

Grammar format:

FOR<num expression1> TO<num expression2> [BY <num expression3>]

[<loop statements>]

END FOR

Example:

```
R[2]=0
FOR R[1]=0 TO 2 BY 1
R[2]=R[2]+1
END FOR
```

Execution Logic Manual:

The initial value of the loop body variable R [1] is 0, the final value is 3, and the step value is 1, so each loop increases by 1. When making the first judgment, R [1]=0, not greater than the final value, perform the first loop, R [2]=1; When making the second judgment, R [1]=1, not greater than the final value, proceed to the second loop, R [2]=2; When making the third judgment, R [1]=2, not greater than the final value, proceed to the third loop, R [2]=3; When making the fourth judgment, R [1]=3, greater than the final value, exits the loop.

So a total of four judgments were made and three cycles were repeated. After the program finishes running, R [1]=3, R [2]=3.

Tips:

If "dead loop" logic is required during the application process, it is recommended to add a WAIT TIME=50/100 delay instruction in the loop statement to avoid possible logic handling exceptions caused by program execution being too fast.

7.6.3.2 WHILE... END WHILE

The WHILE statement determines whether the loop has ended based on a conditional expression. When the condition is true, continue cycling; When the condition is false, exit the loop. The WHILIE statement and END WHILE form a loop statement block.

Grammar format:

WHILE <bool expression>

[<loop statements>]

END WHILE

Example:

```
R[1]=0
WHILE R[1]<3
R[1]=R[1]+1
END WHILE
```

Execution Logic Manual:

When making the first judgment, R [1]=0, less than 3, perform the first loop; When making the second judgment, if R [1]=1 and is less than 3, proceed to the second loop; When making the third judgment, if R [1] is 2 and less than 3, proceed to the third loop; At the fourth judgment, R [1]=3, not less than 3, exits the loop. It cycled three times in total.

Tips:

If "dead loop" logic is required during the application process, it is recommended to add a WAIT TIME=50/100 delay instruction in the loop statement to avoid possible logic handling exceptions caused by program execution being too fast.

7.6.3.3 BREAK

The BREAK statement is used to exit the FOR or WHILE loop.

Grammar format:

<BREAK>

Example:

```
WHILE R[1]=0
....            'program content
BREAK           'Break
END WHILE
```

```
FOR R[1]=0 TO 2 BY 1
....            'program content
BREAK           'Break
END FOR
```

7.7 Motion command

The motion command specifies the type of action for the robotic arm to move. There must be POINT information after the motion command, and the motion parameters of the segment can be specified. If not specified, the system's default motion parameters will be used. Additional motion attributes can be added after the motion command to control the logical operation of the motion segment.

Grammar format:

<Motion Type> <TargetPoint> [Motion Parameter][Additional Property]

Example:

```
J P[1] VEL=100, OFFSET JR[1]
```

Tips:

When the target points of multiple consecutive lines of motion instructions are the same POINT, the system will forward process and ignore these repeated POINT.

7.7.1 J joint motion command

The joint motion J command starts from the current actual position of the robotic arm (or external axis) and moves the robotic arm (or external axis) to the target POINT position. The movement process does not involve trajectory and attitude control, meaning all joints start and stop simultaneously.

Grammar format:

J <TargetPoint> [Motion Parameter][Additional Property]

Example:

```
J P[1] VEL=50 ACC=100 DEC=100    'Specify speed, acceleration, and deceleration
J P[2]                           'Use system default parameters
```

Tips:

The target POINT of joint motion can be selected in two formats: joint point and spatial point, namely J JR [x] and J LR [x]. When executing J JR [x], it will move to the joint angle specified by the target point; When executing JLR [x], due to the multi solution problem of spatial POINT, there will be multiple joint inverse solutions. The system will select the values of joint axes 1, 3, and 5 based on the posture angle specified in the shape position CFG. For the rotation numbers of axes 4 and 6, the system defaults to the shortest path strategy, but users can use the value of the motion parameter TURN to specify the direction of rotation of the end axis.

7.7.2 L Linear motion command

The linear motion L command starts from the current position of the robotic arm and moves in a straight line to the target point, used in situations where trajectory control is required.

Grammar format:

L <TargetPoint> [Motion Parameter][Additional Property]

Example:

```
L P[1] VEL=50 ACC=100 VROT=50    'Specify position speed, position acceleration, attitude rotation speed
```

Tips:

When moving in a straight line, the shape of the robotic arm cannot be changed. If the shape of the starting point is inconsistent with that of the target point, an alarm will be triggered (because shape changes often pass through

singular points), and the motion will stop. But for situations where the wrist shape is inconsistent, the WRISTJNT command can be used to navigate through the wrist singularity zone.

When moving in a straight line, the changes in position and posture are uniform and synchronized, so setting an unreasonable posture speed may affect the final movement rhythm.

When moving in a straight line, the target POINT has two formats for nodes and spatial points, namely L JR [x] and L LR [x]. When executing LJR [x], the spatial point corresponding to the joint position will be moved, so there may be a situation where the actual joint position of the robotic arm when it reaches the target point is inconsistent with the target POINT recorded in the program.

7.7.3 C arc motion command

The circular motion C command takes the current position as the starting point, MIDDLEPOINT as the midpoint, and TARGETPOINT as the endpoint, controlling the robot to perform circular trajectory motion in Cartesian space (three points form a circular arc). The arc command cannot draw a whole circle. If you want to draw a whole circle, you need to use two arc C commands.

Grammar format:

C <MiddlePoint> <TargetPoint>[Motion Parameter][Additional Property]

Example:

```
'A section of circular arc is written as follows
L P[1]          'Starting point of arc
C P[2] P[3]


'The whole circle is written as follows
L P[1]
C P[2]P[3]
```

C P[4]P[1]

The legend for the whole circle is as follows:

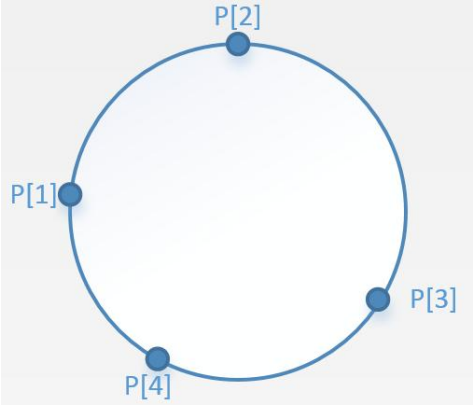


Figure 5-1 Schematic diagram of the whole circle POINT

Tips:

Circular motion is also a spatial motion that satisfies the constraint of linear motion L instruction, that is, the motion does not change the shape, and when the target point is a joint point, it will move to its corresponding spatial point, with synchronized position and attitude velocity.

The circular motion does not consider the attitude constraint at the midpoint, but only considers the attitude of the target point, that is, the attitude at the midpoint during the circular interpolation process may not necessarily be the same as during teaching.

Note: When opened in manual mode or online editing, the [Record] function in the arc C command is invalid. It needs to be modified by clicking on the command and moving the cursor to the input box for recording the required data to record POINT; The point function is to move the arc from the first point to the second point.

7.7.4 A arc motion command

The multi segment arc A instruction can be used to implement multiple arcs of different radii, which are used to splice multiple irregular curves. The positions specified by three consecutive A instructions in the program line can form a circular arc. When using four consecutive A commands, it is possible to achieve arcs with radius X

for the first three points and radius Y for the last three POINT points. POINT addition and deletion can be performed between two arcs, and different motion additional attributes can be set. There is no pause between the movements of the two arcs.

Grammar format:

A <TargetPoint>[Motion Parameter][Additional Property]

Example:

```
J P[1] VEL =100
A P[2] VEL =500
A P[3] VEL =500
A P[4]
A P[5]
L P[6]
```

Execution Logic Manual:

The execution trajectory of the above program is shown in Figure 5-2. When executing the first circular motion A command (line 2), the robot's motion mode is linear motion, that is, from the robot's current position P [1] to the target point P [2], from P [2] to P [3] (circular arc P2P3P4), from P [3] to P [4] (circular arc P3P4P5), from P [4] to P [5] (circular arc P4P5P3), and from P [5] to P [6].

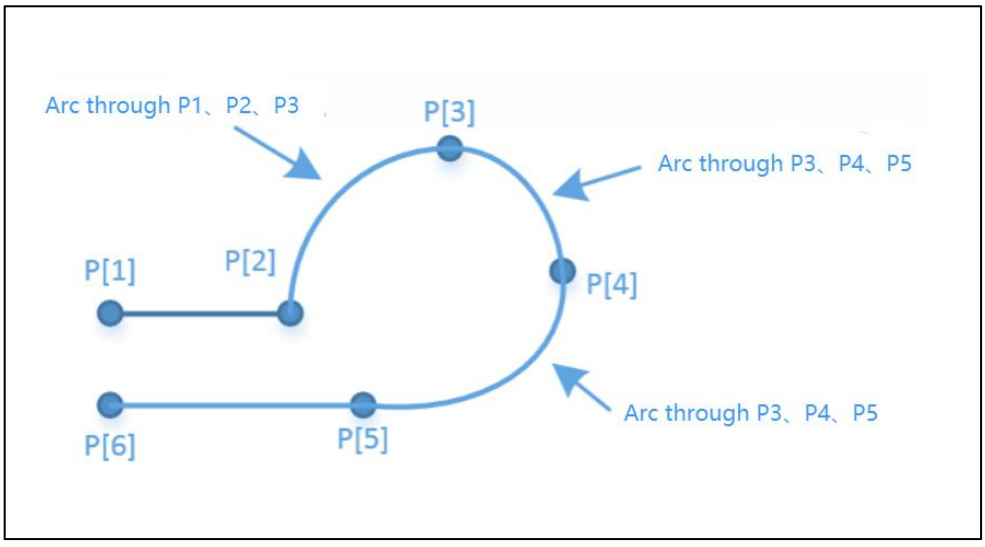


Figure 5-2 Schematic diagram of program trajectory for example A instruction

Tips:

When there are less than 3 consecutive A commands, the robotic arm will move in a straight line to the target point, where the A command is equivalent to the L command.

7.7.5 JUMP gate motion command

The "gate shaped" motion JUMP instruction is to teach only one target point, and the robotic arm can move from the current position "gate shaped" to the target POINT. During motion, the robotic arm will perform three trajectory movements. The first trajectory is a vertical upward movement, the second trajectory is a joint movement on a specified plane, and the third trajectory is a vertical downward movement. The three trajectories are smoothly connected.

Grammar format:

JUMP <TargetPoint> LIMZ=<num expression> ARCH_A=<num expression> ARCH_B=<num expression>

- LIMZ: represents the position value of the SCARA robotic arm's 3-axis (unit: mm), and the robotic arm performs joint motion on the z-plane corresponding to the height of this position value. Due to the value limit of SCARA 3-axis being [A3. PMIN, 0], this value is less than 0.
- ARCH_S: represents the safe ascent distance (unit: mm), which means that the current position of the robotic arm rises the specified distance, then smoothly turns into the plane specified by the LIMZ parameter, and performs the second joint movement.
- ARCH_S: represents the safe descent distance (unit: mm), which means that the robotic arm descends to the specified height and reaches the target point. The robotic arm smoothly turns out from the z-plane and then descends vertically to the target point.

Example:

```
JUMP P[1] LIMZ=-20 ARCH_A=50 ARCH_B=80
```

The execution trajectory of the above program is shown in Figure 5-3.

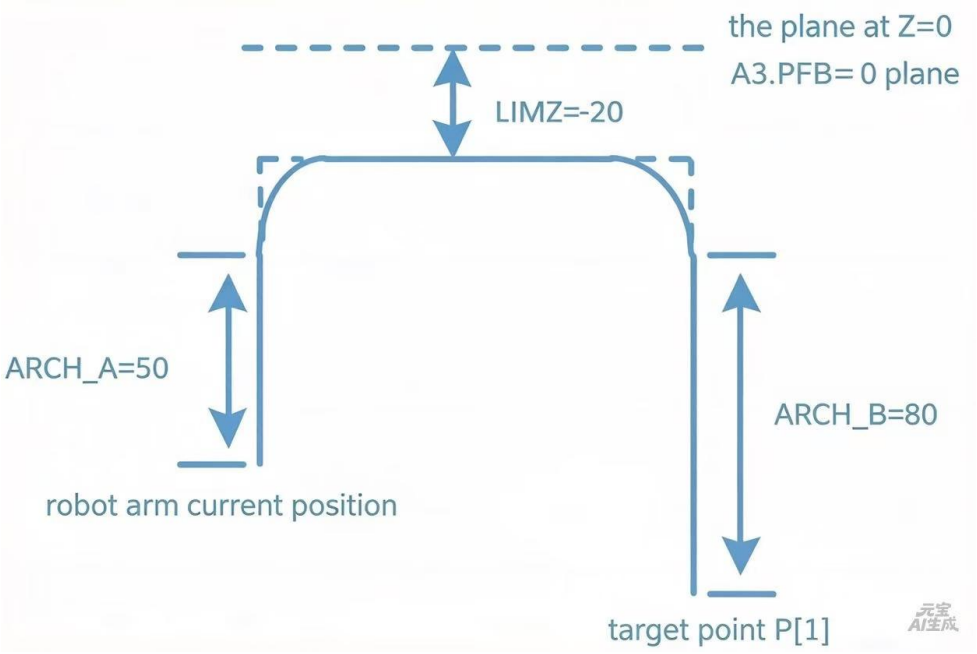


Figure 5-3 Schematic diagram of JUMP instruction example program trajectory

Tips:

The JUMP command can only be used on SCARA robotic arms and cannot be used on six axis robots.

7.7.6 REULMOVE square hole machining motion command

The Lelo triangle motion can be used to achieve mixing operations inside square holes, requiring only one point to be taught, and mixing will almost stick to every corner. When using this command, a stirring fixture must be installed at the end of the robotic arm TCP. The end of the robotic arm TCP will spiral downwards while rotating, and spiral upwards after reaching the specified depth. After reaching the limit, the 6-axis will automatically reverse (i.e. forward and reverse within the limit). After reaching the designated time, the stirring action stops.

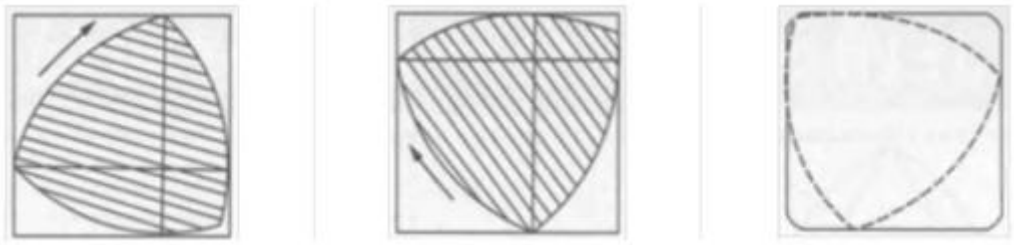


Figure 5-4 Schematic diagram of Leluo triangle motion

Grammar format:

REULMOVE SLEN=<num> DEPTH=<num> VEL=<num> FVEL=<num> TIME=<num>

- SLEN: The side length value of the directional hole, measured in millimeters.
- DEPTH: Depth of the hole, measured in millimeters.
- VEL: End rotation speed, measured in degrees per second.
- FVEL: Feed rate, measured in mm/s.
- TIME: The time of movement, measured in milliseconds.

Example:

```
    L P[1]      'Move in a straight line to point P1, which is the center point of the Lelot triangle
    REULMOVE SLEN=20 DEPTH=10 VEL=10 FVEL=10 TIME=3000      'Start stirring, timed for 3 seconds
```

7.8 Local motion parameters

Local parameters are motion parameters written after the instruction line, which control the characteristics of the trajectory segment. Local parameters are optional. If not written, the system uses default global parameters. If filled in, local parameters are used. The priority of local parameters is higher than that of global parameters.

Table 6-1 List of Local Motion Parameters of the System

Parameter	Meaning	Value Range	Example usage
VEL	Position speed	Floating positive integer. The J command corresponds to percentages ranging from 1 to 100, while the L/A/C command corresponds to spatial velocity values, measured in mm/s	J P[1] VEL=100 L P[2] VEL=500
VROT	Attitude speed	Floating positive integer. Effective only for L/A/C commands, in degrees per second	L P[2] VEL=50
ACC	Acceleration	Floating positive integer. Unit is	L P[2] ACC=50

		percentage	
DEC	Deceleration	Floating positive integer. Unit is percentage	L P[2] DEC=50
SMOOTH	Acceleration compliance level	Unsigned integer. The value range is 1~9, and the larger the value, the smoother the acceleration	L P[2] SMOOTH=50

Grammar format:

<Motion Type> <TargetPoint>{[Motion Parameter]}

Example:

```
J P[1] VEL=100 ACC=100      'VEL=100 and ACC=100 are local parameters
```

7.9 Global motion parameters

7.9.1 Global speed parameter table

The global motion parameters set the characteristics of all motion commands in the program during motion, and the effective range is the entire PRG program. By setting global motion parameters, it is possible to quickly adjust trajectory rhythms and other parameters without the need to input local parameters after each command line for control. The global motion parameters only take effect during the execution of this PRG and do not affect other PRG programs.

Table 7-1 List of Global Motion Parameters of the System

Parameter	**Meaning**	**Value setting range**	**Default value**
J_VEL	Joint speed (effective for J motion command)	1~100, in percentages, and divided according to the pattern: Manual T1:1~10 Manual T2:1~20 Automatic/External: 1~100	100

J_ACC	Joint acceleration ratio (effective for J motion commands)	1~200, expressed as a percentage	100
J_DEC	Joint reduction ratio (effective for J motion command)	1~200, expressed as a percentage	100
L_VEL	Linear velocity (linear L motion command)	According to the mode limitation: Manual T1: 1~125 mm/s Manual T2: 1~250 mm/s Automatic/External: 1-2400 mm/s, maximum value of 2400. The vtran value can be modified in the group speed parameter file group [0] -velocity.data file, which is different for each model	Maximum value
L_ACC	Linear acceleration ratio (linear L motion command)	1~200, expressed as a percentage	100
L_DEC	Linear deceleration ratio (linear L motion command)	1~200, expressed as a percentage	100
L_VROT	Linear attitude velocity (linear L motion command)	According to the mode limitation: Manual T1: 1~30 °/s Manual T2: 1~80 °/s Automatic/External: 1~120 °/s, maximum value of 120. The value of vrot can be modified in the group speed parameter file group [0] -velocity.data file, which is different for each model	Maximum value

C_VEL	Arc velocity (motion commands for arcs C and A)	According to the mode limitation: Manual T1: 1~125 mm/s Manual T2: 1~250 mm/s Automatic/External: 1-2400 mm/s, maximum value of 2400. The vtran value can be modified in the group speed parameter file group [0] -velocity.data file, which is different for each model	Maximum value
C_DEC	Arc reduction ratio (arc C, A motion command)	1~200, expressed as a percentage	100
C_ACC	Arc Acceleration Ratio (Arc C, A Motion Command)	1~200, expressed as a percentage	100
C_VROT	Arc attitude velocity (motion commands for arcs C and A)	According to the mode limitation: Manual T1: 1~30 °/s Manual T2: 1~80 °/s Automatic/External: 1~120 °/s, maximum value of 120. The value of vrot can be modified in the group speed parameter file group [0] -velocity.data file, which is different for each model	Maximum value
SMOOTH	Acceleration compliance level, can be used to adjust rhythm and jitter	Unsigned integer. The value range is 1~9, and the larger the value, the smoother the acceleration	5

Grammar format:

<Global Motion Parameter>=<Num Expression>

Example:

```
J_VEL=100    'The following four are global parameters
J_ACC=100
L_ACC=120
SMOOTH=1
J P[1]
J P[2]
L P[3]
```

7.9.2 SMOOTH flexibility level parameters

To meet the adjustment requirements of rhythm and jitter in different scenarios, the system provides SMOOTH parameters to control the speed of acceleration changes in the robotic arm. SMOOTH value range: 1~9, default value 5. The larger the value, the smoother it is, and the slower the acceleration changes. SMOOTH=1 indicates that the acceleration changes the fastest during the start stop phase, which may cause shaking; SMOOTH=9 indicates that the acceleration changes the slowest and smoothest during the start stop phase.

Example:

```
SMOOTH=1     'Global parameters
J P[1]
J P[2]
J P[3] VEL=100 ACC=100 SMOOTH=5    'Local parameter
J P[4]
```

7.10 Trajectory control parameters

Trajectory control parameters are used to control the motion trajectory and motion strategy of the robotic arm. Similar to motion parameters, trajectory control parameters also have global and local distinctions. Users can choose trajectory parameters according to their actual application needs to meet application requirements.

Table 8-1 Trajectory Control Parameter List

Parameter	Meaning	Value Range	Default value
CNT_TYPE	Smooth type	Unsigned integer 0|1|2	0
CNT	Smooth parameter percentage	Floating point positive integer 0~100	0
CR	Smooth corner radius value, in millimeters	Floating point positive integer 1~100	0
TURN	End axis rotation control options	Unsigned integer 0|1|2|3|4	0
TALARM	End axis rotation control alarm switch	Unsigned integer 0|1	0

7.10.1 CNT_TYPE smooth type parameter

CNTTYPE is a smooth type parameter. There are currently two different smoothing types in the system, and different values of CNTTYPE indicate the use of different smoothing types:

(1) CNTTYPE=0 or 1 is the speed smoothing mode, and the robotic arm adopts a speed mixing method to "turn" into the second trajectory with the current speed as the initial condition.

(2) CNTTYPE=2 is the trajectory smoothing mode, and the system uses Bezier curves to smoothly concatenate two trajectory segments.

The difference between the two smoothing methods is that the CNTTYPE=2 smoothing method does not change the trajectory accuracy of the smooth transition segment in the following three situations:

(1) Modify the speed multiplier (VORD) at which the program runs.

(2) Modify the VEL speed parameter of the instruction.

(3) Pause and restart during smooth trajectory segment.

Smooth speed has the characteristic of wide adaptability, and can have good smoothing effects on joint+line, joint+arc and other situations. However, changing the speed will change the smooth trajectory.

Trajectory smoothing has the characteristic of not deforming the trajectory, but there may be constraints in some cases. Therefore, when debugging or pausing startup, it is required that the smoothed trajectory does not change, and trajectory smoothing can be used.

Grammar format:

<CNT_TYPE>=<0|1|2>

Example:

```
CNT_TYPE=1
CNT=100
J P[1]
L P[2] CNT=50
C P[3] P[4]
J P[5]
```

Tips:

When the CNTVNet instruction is not written to specify the smoothing type, the system defaults to the default value of 0 for CNTVNet, which is the speed smoothing method.

CNT_TPYE specifies the smoothness type, while CR and CNT specify the smoothness value. Therefore, both CR and CNT attributes can be used when velocity smoothing CNTTYPE=1 or trajectory smoothing CNTTYPE=2, and their meanings remain unchanged.

7.10.2 CNT smooth transition parameter

CNT parameters can be used for smooth path stitching of two motion trajectories, reducing motion pauses. Its value is a percentage (ranging from 0 to 100). The meaning of CNT parameter value: the ratio of the time occupied by the smooth trajectory to half of the total motion time T of the trajectory segment. T is the total time required for the trajectory to move. Due to the positive correlation between the length of the trajectory segment and the running time t, the proportional relationship between the trajectory points at a certain moment and the total trajectory can be used to visually describe the CNT properties of the system.

Syntax format 1 (global parameters):

```
CNT=<1~100>
J P[1]
J P[2]
```

J P[3]

Syntax format 2 (local parameters):

J P[1]

J P[2] CNT=100

J P[3]

Example:

```
L P[A]
L P[B] CNT=50
L P[C]
```

Execution Logic Manual:

The trajectory path is A → B → C, as shown in Figure 8-1. Using the smooth transition function at point B, the robotic arm turns in at point M and out at point N.

The attribute value of CNT means CNT=BM/(AB/2) * 100. Then:

CNT=30 = t1/(T/2)

CNT=50 = t2/(T/2)

CNT=100 =t3/(T/2)

The larger the CNT value, the smoother the trajectory of the robotic arm starts from the midpoint of the AB segment trajectory. When the CNT value is maximum, the system starts smoothing from the midpoint of the trajectory. When CNT=0, it indicates that there is no smoothness, and the motion of the robotic arm will pass through point B, and the velocity at point B will decrease to 0 before continuing with the next stage of motion.

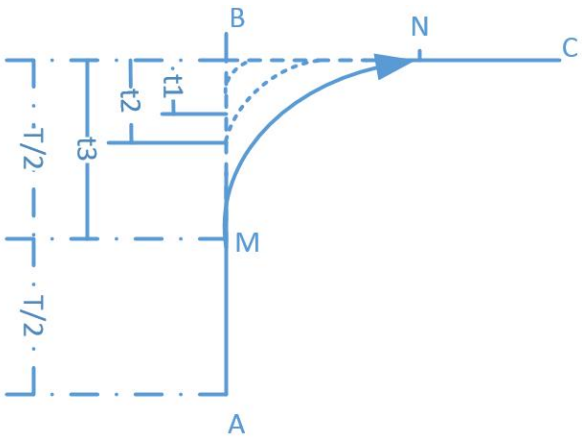


Figure 8-1 Schematic diagram of smooth trajectory of example program

Tips:

When the speed is smooth, the CNT value can be set reasonably to achieve good smoothing effect.

```
L P[1]
L P[2] CNT=20/50/100
L P[3]
```

When a long trajectory is shortened, the CNT value can be set to a smaller value.

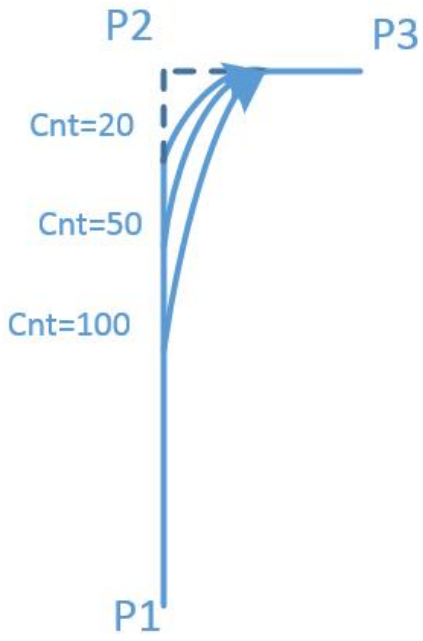


Figure 8-2 Schematic diagram of different CNTs for short and long trajectories

When a short trajectory is extended to a long trajectory, the CNT value should be set to a larger value.

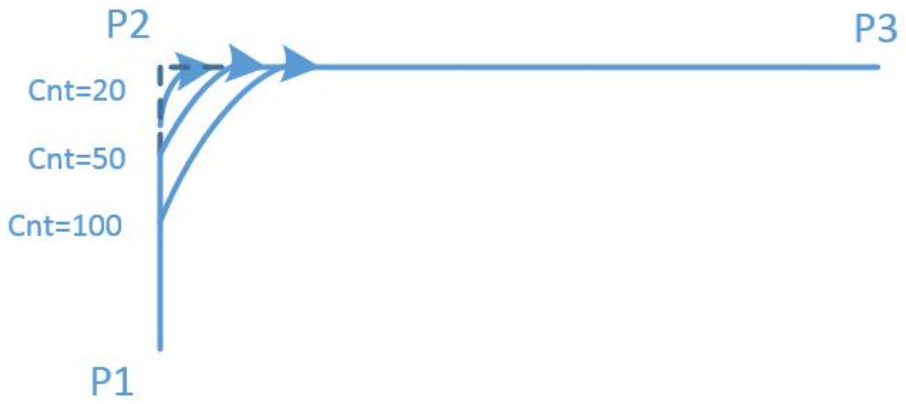


Figure 8-3 Schematic diagram of different CNTs for short and long trajectories

Tips:

The relationship between signal output and smooth transition points suggests:

```
L P[A]
L P[B] CNT=50
DO[1] = ON        'At this point, the signal is output at transition point M
J P[C]
```

The relationship between WAIT waiting conditions and smooth transitions suggests:

```
L P[A]
L P[B] CNT=50
WAIT DO[1] = ON     'If the DO [1] signal is ON before the transition point M, there is smoothing. If the WAIT condition is not met, it will stop at point P [B] and wait for the condition to be met. At this time, there is no smoothing action
J P[C]
```

Boundary Limit Instructions: Typically, when CNT reaches its maximum value, the trajectory transition starts from the midpoint of the trajectory.

Adaptive Instructions: When the trajectory is less than 5mm, the smooth planning performs adaptive processing, and the smooth trajectory first meets

the requirements of high-speed smoothness. It is not strictly constrained by the current CNT smoothing value and may start smoothing from any point on the trajectory.

Pause Start Instructions: Pause during the smoothing process. When starting the motion, it will run directly to the final POINT position and will not follow the old smooth trajectory.

Does not support smoothing between J | L | A | C instructions and JUMP instructions. Example program instructions are as follows:

```
L P[1] CNT=50, PASS          'No smoothness, accurate point P [1]
JUMP P[2] LIMZ=-10 ARCH_A=40 ARCH_B=60
```

7.10.3 FINE attribute

The FINE positioning attribute is used to accurately move the robotic arm to the target POINT position and stop, for precise control of the robotic arm to the point. Due to the delay of servo position feedback, when the position command controls the robotic arm to reach the target point, the actual position of the robotic arm may not be in place yet. Therefore, it is necessary to wait until it is determined that the robotic arm is in place before executing the next movement.

Grammar format:

<Motion Type><Tatget Point>[CNT=FINE]

Example:

```
L P[0]
L P[1] CNT=FINE     'First move in a straight line to P [0], then move in a straight line to point P [1] and wait for the feedback position to be in place, then move in a straight line to point P [2]
L P[2]
```

Tips:

CNT=0 and CNT=FINE both indicate that the speed of the robotic arm at that

point drops to 0. However, CNT=0 means that the commanded speed at that point drops to 0 but does not wait for servo feedback to reach 0 before starting the next stage of motion. CNT=FINE means moving to that point and waiting for position feedback to be in place.

7.10.4 CR Smooth transition parameter

In trajectory based applications such as gluing and polishing, there are geometric requirements for the smooth stitching of two trajectories, which require the ability to specify the position of the turning point. In this case, users can use the CR attribute function, which specifies the distance between the turning point and the target point during smooth transition, with a positive value in millimeters (mm). The larger the value, the larger the corner radius, and the farther the smooth trajectory's turning point is from the target point.

Grammar format:

<Motion Type><Tatget Point>[CR=<num expression>]

Example:

```
L P[1]
L P[2] CR=20
L P[3]
```

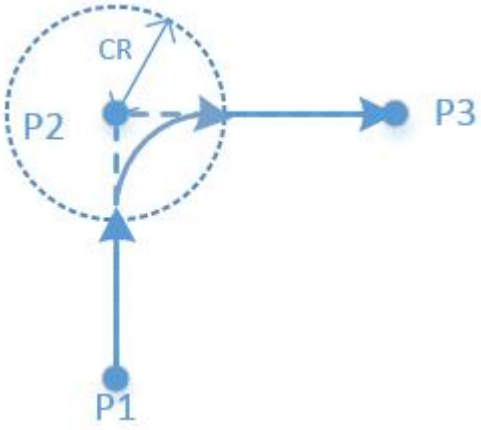


Figure 8-4 Schematic diagram of smooth trajectory of example program

Tips:

CR is a positive number, meaning the value is greater than 0. If the input is

incorrect, an alarm must be triggered.

When the CR value is less than half of the short side, the actual radius value of the corner path is the arc of CR.

When the value of CR is greater than half of the short side, the actual corner radius is equal to half of the short side. However, due to velocity smoothing, the actual turning point may change depending on the velocity value when a long line segment is connected to a short line segment, resulting in corresponding changes in the smooth trajectory.

The CR attribute supports L, C, and A instructions, but does not support JUMP instructions. Writing it will not trigger an alarm, but it will have no effect.

In the case of two-point turning motion or three-point collinearity, the trajectory characteristics will cause the value of CR to be ineffective, and the motion will be precise to the point.

The CR and CNT attributes can be used when the velocity smoothing CNTTYPE=1 or the trajectory smoothing CNTTYPE=2. CNT_TPYE specifies the smoothing type, while CR and CNT specify the smoothness value.

7.10.5 TURN | TALARM end axis control parameters

TURN and TALARM are used to control the rotation direction of the end axes (6-axis of 6-joint robotic arm and 4-axis of SCARA) when the robotic arm executes JLR [x]. Due to the fact that the inverse solution of the system space points can correspond to multiple joint points, there are issues with form positioning and solution selection.

The meanings of the various options for the TURN parameter are as follows:

- TURN=0, which means the direction with the smallest angle rotation change of the spatial equivalent rotation axis. If the inverse solutions of axes 4 and 6 exceed the limit in this direction, the reverse solution will be automatically selected, which is the default option of the system.
- TURN=1, which means the direction where the angle rotation of the spatial equivalent rotation axis changes the most is the opposite direction when TURN=0.
- TURN=2, ensuring that the end axis is within+-180 °, without considering the rotation direction of the equivalent rotation axis. Under this option, there may be forward and reverse rotation of the end axis, but it will

not exceed+-180 degrees.

- TURN=3, the end shaft can only rotate in the positive direction. If the joint of the end shaft rotating in the positive direction exceeds the limit, an alarm will be triggered.
- TURN=4, The end axis can only rotate in the opposite direction. If the joint of the end axis reversing exceeds the limit, an alarm will be triggered.

The TALARM parameter refers to whether the system is allowed to automatically select the option value of TURN to ensure the normal operation of the system, rather than generating alarms.

- The default value of TALARM is 1, indicating that the robotic arm can only operate according to the requirements of the TURN option. If it does not meet the requirements, the system will sound an alarm.
- TALARM=0， The option to turn off TURN requires the system to automatically select the appropriate value of the TURN option to ensure that the movement of J LR [X] does not generate an alarm.

Example:

```
TURN = 0        'Method 1, Global Parameter Setting
TALARM = 0      'Method 1, Global Parameter Setting
J LR[x] TURN=2 TALARM=1        'Method 2: Use local parameter settings
```

Tips:

The TURN and TALARM options only take effect when J<space point>, and have no effect when J<joint point>or L | C<space | joint>. That is, when J JR [x] TURN=1 TALARM=1, the system will not alarm, but the parameters have no actual meaning.

When the input of TURN and TALARM parameter values is illegal (such as parameter values exceeding the range), the value change operation will not take effect.

7.10.6 WRISTJNT singular parameters passing through the wrist

The WRISTJNT instruction can achieve passing through wrist singularities. When passing through the wrist singularity, the 4th and 6th axes do not flip, and the TCP end of the robotic arm maintains a straight trajectory, but the posture position will change. If this instruction is not used, the speed of the robotic arm will gradually decrease and then stop at the boundary of the singular interval when it approaches the wrist singularity.

Grammar format:

WRISTJNT=<num expression>

Example:

```
WRISTJNT=1
L JR[1]    'JR[1][4]=-20
L JR[2]    'JR[2][4]=20
```

7.10.7 SEC specify exercise time parameters

The SEC directive specifies the time for a robotic arm to perform a row of joint movements. The SEC instruction only applies to joint motion J instruction and is invalid for spatial motion L | A | C | JUMP.

The numerical unit of SEC instructions is milliseconds, with a range of 0 to [20, 32000]. If the SEC value exceeds the range during program execution, an alarm will be triggered stating that the value is out of range.

When SEC=0, it indicates that the joint motion instructions after the instruction line are not affected by the effect of SEC instructions.

When using SEC commands, the local velocity parameters of joint motion commands are invalid.

The time specified by the SEC directive is the time it takes to move at a speed of 100% magnification. If the magnification is adjusted, the actual time for the robot to complete the movement will change according to the adjusted magnification.

- For example, if the specified beat is 1000ms, then when the magnification is 100%, the actual time for the robot to complete the movement is 1000ms; When the magnification is 50%, the actual time for the robot to complete the movement is 2000ms.

If the robotic arm can move to the target POINT within the specified SEC time, it will move within the specified time. Otherwise, an overspeed alarm will be reported.

A continuous motion command line with the same specified time is called a segment. In terms of algorithm, the system will perform global planning within each segment and use smooth transition effects between segments to ensure overall continuity of motion. The smooth transition effect is not affected by the CNT attribute values.

Example:

```
VORD=100
```

```
J P[1]          'P [1] is not affected by the effectiveness of SEC directives
R[1]=1000
SEC=R[1]
L P[2] VEL=2000   'P [2] is not a joint motion command, SEC command has no effect, and VEL=2000 is used during motion
SEC=50
J P[3] CNT=0      'During actual exercise, the specified rhythm is performed with SEC=50, and CNT=0 setting is invalid
SEC=10            'SEC instruction value exceeds the range, therefore an alarm is triggered
J P[4] L_ACC=50
L P[5] L_VORT=100    'P[5] is not a joint motion command, therefore SEC command is invalid
SEC=20
VORD=50
J P[6]            'If the magnification is 50% and SEC is 20, then the actual movement will take 40ms to perform the specified rhythm movement
SEC=0
J P[7]
```

7.11 Sports additional attributes

The additional attribute of motion refers to the operation of controlling the robotic arm to change the target point, change the motion state, or output signals during the execution of motion. The instructions and order of effectiveness when using additional attributes for each sport simultaneously are shown in the following table:

Table 9-1 Detailed Description of Attributes

Name	The elements acted upon	Effective order	Notes
INC	Target point	1（Highest）	Cannot be used simultaneously with OFFSET/TOL_OFFSET

			Cannot attach to C/A/JUMP motion command
OFFSET	Target point	1	Cannot be used simultaneously with INC If these two instructions are used simultaneously, when calculating the motion POINT, first calculate the OFFSET offset value on the target point, and then calculate the TOOL-OFFSET tool compensation value
TOOL_OFFSET	Target point/Coordinate System	1	
SKIP	Motion state	2	Motion interruption signal, judged when motion begins to execute
BIND_TR	Motion logic I/O signal	3	Output signal upon reaching the designated location
WAIT	Motion logic I/O signal	3	Wait for the signal to meet the requirements before moving to the target point when reaching the designated position
PASS	Instruction execution effect	4（Minimum）	Independent use

7.11.1 INC incremental motion attribute

Incremental motion is the movement of a robotic arm based on its current position, which means taking the value of the target POINT as the incremental value of the current robotic arm's movement. J P [x], INC, where the incremental value of P [x] can be a joint coordinate value or a Cartesian coordinate value.

Example:

```
J JR[0]          'Move to JR [0]
J JR[1], INC     'Perform incremental motion based on the previous point,
```

```
moving to JR [0]+JR [1]
    J JR[2], INC    'Perform incremental motion based on the previous point,
moving to (JR [0]+JR [1]+JR [2])
```

Execution Logic Manual:

JR[0]={0,-90,180,0,90,0}，JR[1]={10,10,10,10,10,10}，JR[2]={-5,-5,-5,-5,-5,-5}，

The execution process involves joint movement to point JR [0]. When executing J JR [1], INC will perform relative motion based on point JR [0], moving to point (JR [0]+JR [1]). The third row will move to point (JR [0]+JR [1]+JR [2]).

The POINT coordinate values for the three movements are:

JR[0]={0,-90,180,0,90,0}，

JR[0]+JR[1]={0,-90,180,0,90,0}+{10,10,10,10,10,10}={10,-80,190,10,80,10}，

JR[0]+JR[1]+JR[2]= {0,-90,180,0,90,0}+{10,10,10,10,10,10}+

{-5,-5,-5,-5,-5,-5}={5,-85,185,5,75,5}

Tips:

When the incremental value of joint motion is the spatial POINT, the spatial POINT will be solved in reverse first, and then the joint solutions will be selected and overlaid before moving to the target point.

For example:

```
J JR[1]
J LR[1], INC    'LR [1] will first invert to joint values and then superimpose them
```

7.11.2 OFFSET position compensation attribute

The position compensation instruction performs offset compensation on the position, depending on the type of compensation value. There are two forms of compensation. Joint compensation and spatial POINT compensation. The compensation effect varies depending on the POINT type!

Form 1 is the case where the compensation value is joint, i.e. OFFSET JR [x]. The compensation effect is to add the compensated joint values to the joint values set at the target POINT after moving to the target point.

Form 2 is the case where the compensation value is spatial POINT, i.e. OFFSET LR [x]. The compensation effect is to offset the direction of the workpiece coordinate system when moving to the target point and placing it along the target POINT, which is equivalent to multiplying the current point matrix by the compensation value.

Point type Instructions for compensation values.

（1）The compensation value is the joint value, which is the effect instruction of OFFSET JR [x]

```
J JR[1], OFFSET JR[0]    'Final position:JR[1]+JR[0]
J LR[0], OFFSET JR[0]    'LR [0] is first solved as joint value P1, and the final position is P1+JR [0]
```

（2）The compensation value is a spatial value, which is the effect instruction of OFFSET LR [x]

```
L LR[1], OFFSET LR[0]    'Offset compensation is performed along the current workpiece coordinate system, resulting in the final position LR [0] MATMUL LR[1]。
L JR[0], OFFSET LR[0]    'JR [0] is first solved as spatial point P1, and offset compensation is performed along the current workpiece coordinate system, resulting in the final position LR [0] MATMUL P1
```

There are two methods of compensation, global form and local form

（1）Global condition compensation instruction program Example:

```
    OFFSET_CONDITION=JR[2]    'Define global position compensation amount
    J JR[0]                   'Correctly move to the JR [0] coordinate position
    J JR[1],OFFSET            'JR [1]+JR [2] move to a recalculated position
```

As shown in the above program, the program assigns JR [2] to the global position offset and sequentially moves it to JR [0] point (without OFFSET, no offset required). When executed to the last line, it will move to the offset compensated position, that is, JR [3]=JR [1]+JR [2] (with OFFSET, and the

value of OFFSET-CODITION is JR [2]).

（2）Direct position compensation instruction program Example:

```
J JR[0]                          'Correctly move to the JR [0] coordinate position
J JR[1],OFFSET JR[2]   'JR [1]+JR [2] move to a recalculated position
```

As shown in the above program, move sequentially to point JR [0] (without OFFSET, no offset required), and when executed to the last line, it will move to the offset compensation position, that is, JR [3]=JR [1]+JR [2] (with OFFSET JR [2], the direct offset is JR [2]).

Tips:

！！ It should be noted whether the compensation value is in the form of joint points or spatial points, and the compensation effect varies depending on the POINT type!!

If method 1 is used and the OFFSETVNet instruction is not defined before the motion command J JR [1] OFFSET, an alarm will be triggered at runtime stating 'OFFSET value not set'.

If the JR [2] POINT information is modified midway during the execution of J JR [1] OFFSET or J JR [1] OFFSET JR [2], the offset will also change synchronously.

The coordinate representation of compensating POINT can be inconsistent with the target point (such as using joint coordinate system for the target point and Cartesian coordinate system for compensating POINT).

When compensating for the arc command, only the endpoint is compensated, and the transition point POINT of the arc is not compensated.

7.11.3 TOOL_OFFSET tool TCP direction compensation attribute

In practical applications, position compensation needs to be performed along the XYZ direction of the currently used tool coordinate system. When compensating along the tool coordinate system, only spatial POINT compensation is supported,. There are also two compensation methods: global compensation and local compensation.

When compensating for spatial position, it is TOOL_SFFSET LR [x]. The compensation effect is the displacement along the coordinate axis of the tool coordinate system after moving to the target point. It is equivalent to performing a matrix right multiplication operation on the target POINT. Identify new target points.

The compensation value is a spatial value, which is the effect instruction of TOOL-OFFSET LR [x]

```
L LR[1], TOOL_OFFSET LR[0]     'Final position LR [1] MATMUL LR [0].
L JR[0], TOOL_OFFSET LR[0]     'JR [0] is first solved as spatial point P1, and offset compensation is performed along the current tool coordinate system, resulting in the final position P1 MATMUL LR [0].
```

There are two methods of compensation, global form and local form. There are two methods: one is to set the global tool compensation amount CONDITION and use TOOL-OFFSET to implement it, and the other is to directly specify the tool compensation amount after TOOL-OFFSET.

（1）Set global tool compensation amount Example:

```
TOOL_OFFSET_CONDITION=LR[2]     'Define global tool compensation amount
J JR[0]                         'Directly move to JR [0]
J LR[1],TOOL_OFFSET             'Move to the (LR [1] MATMUL LR [2]) position.
```

（2）Directly specify the tool offset compensation amount Example:

```
J JR[0]                         'Correctly move to the JR [0] coordinate position
J LR[1],TOOL_OFFSET LR[2]       'Move to the (LR [1] MATMUL LR [2]) position.
```

Tips:

！！！ Attention: Is the POINT of the compensation value a joint point or a spatial point? The compensation effect for joint points and spatial points is different!!!

The tool compensation quantity TOOL_SOFFSETVNet can only use the spatial POINT LR [x] variable.

If the TOOL-OFFSET-CODITION instruction is not defined before, the runtime alarm will sound "TOOL-OFFSET value not set".

When executing J LR [1] TOOL-OFFSET or J JR [1] TOOL-OFFSET LR [2], if the TOOL-OFFSET offset LR [2] POINT information is modified midway before executing the instruction, the offset will also change synchronously.

The coordinate representation of compensation POINT can be inconsistent with the target point (such as using joint coordinate system for the target point, and Cartesian coordinate system for compensation POINT).

When compensating for the arc command, only the endpoint is compensated, and the transition point POINT of the arc is not compensated.

7.11.4 SKIP interrupt function attribute

The SKIP attribute can implement the function of interrupting the execution of the current motion instruction. During the execution of motion commands, if the configured triggering conditions are met, the robotic arm will immediately end the current motion and start executing the next line of program instructions. If the triggering condition is not met before the robot reaches the target point, the robot will complete the current motion line and execute the next line of instructions in the program after reaching the target point.

Example:

```
J P[1]
J P[2] SKIP DO[1]=ON    'When the DO [1]=ON condition is true, the P2 point is skipped during runtime, and when it is false, the point is completed normally
J P[3]
```

7.11.5 PASS skip waiting for motion execution instruction attribute

The PASS instruction means to skip and not wait for the current line of motion execution to complete. When the

motion instruction starts executing, the non motion statements following the program also start executing at the same time, until the next motion instruction is encountered, blocking and waiting. Reasonable use of PASS attribute can achieve simultaneous execution of motion and logic, and multiple segments of motion can be smoothly spliced to improve production rhythm.

Example:

```
J P[0]
J P[1] CNT=50, PASS   'When starting the motion, DO [1]=ON and subsequent non motion statements begin to execute, with a smooth transition between P [1] and P [2]. If an alarm is triggered when starting execution, such as exceeding the limit, the subsequent logical instructions will not be executed. Only after the motion is successfully started and executed, there will be a PASS operation for that segment of motion
DO[1]=ON
R[1]=1
IF R[1]=0 THEN   'If the conditions are not met, do not enter the branch and continue to execute downwards
LR[1][1]=1
END IF
IF R[1]=1 THEN        'Conditions are met to enter
LR[1][1] =100
L P[2], PASS    'Block, wait until the motion of J P [1] is completed and the motion of L P [2] begins to execute before the following logical statements are executed
ENDIF
DO[2]=ON
```

Tips:

The PASS attribute only supports J | L | A | C instructions and does not support JUMP instructions.

When a motion command has a PASS attribute, the execution pointer will block and wait for the current motion to complete before executing the next motion when encountering motion command J | L | C | A | JUMP.

The DELAY instruction is equivalent to a placeholder for empty motion, and the execution pointer will block on the DELAY instruction line.

7.11.6 SET_TR signal trigger function attribute

The process of a robotic arm completing a motion trajectory is: stationary (starting point) - acceleration - uniform speed - deceleration - stop (target point), with uneven speed during acceleration and deceleration. If the robotic arm outputs a signal or closes the signal after reaching the starting or target point, it will introduce trajectory segments with uneven speed, which will cause uneven coating in adhesive or welding applications and reduce product processing quality. Using SET_TR and BIND_TR signal triggers, signals can be turned on or off at specified trajectory segment positions.

Grammar format:

SET_TR[num], DS|DE|DC=<num>, <signal expression>

<Motion Type><Target Point>, BIND_TR[num1,num2...]

Parameter Instructions：

- Starting point: Under normal circumstances, it is the target point of the previous motion command. When the motion terminates, the calculation needs to be restarted from the current position of the robot as the starting point.
- DS: The abbreviation for DISTANCE START, which refers to the distance from the starting point of the robot's motion.
- DE: The abbreviation for DISTANCE END, which refers to the distance from the moving target point.
- DC: The abbreviation for DISTANCE Complete, which refers to the percentage of completion of the entire distance.

Table 9-2 Detailed Parameter Manual

Trigger condition	DS=xx	DE=xx	DC=xx
Meaning	Robot TCP outputs signals when the distance from the starting point of robot operation exceeds xx mm	When the distance between the robot TCP and the target point of robot operation is within xx mm, output signal	Robot TCP has run the motion command line. When Robot TCP needs to move xx% of the entire distance, output a signal
Trigger condition	The robot leaves the starting point and the distance (instruction) between the robot and the starting point is greater than xx mm	The robot moves towards the target point, and the distance (instruction) between the robot and the target point is less than xx mm	The robot is moving towards the target point, and the percentage of the robot's TCP walking distance to the total length of the path is greater than xx
Output signal	When the robot meets the triggering conditions, signal output or shutdown is performed		

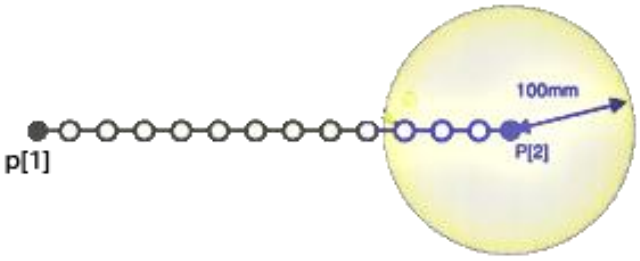


Figure 9-1 Schematic of Triggering POINT

Example:

```
SET_TR[0], DS=50, DO[4]=ON
SET_TR[1], DE=100, DO[4]=OFF    'Set trigger number 1, specify the distance of 100mm from the end point target, and output DO [4]=OFF
J P[0]
L P[1], BIND_TR[0]
```

```
L P[2], BIND_TR[1]    'P2POINT sets trigger 0 or 1, can set single or multiple triggers, separated by commas
```

Execution Logic Manual:

In the example program, the first two lines set triggers 0 and 1, which means that at the end point 100mm, DO [5] outputs ON (note: when there is the same trigger number, execute according to the latest trigger number condition). Trigger 0 is bound at point P1, and trigger 1 is bound at point P2; The robotic arm moves linearly to point P [1], and then moves linearly from point P [1] to point P [2]. The system will periodically detect the position of the robot's TCP distance P1 and P2 points. When the distance P [1] is greater than 50mm, DO [4]=ON signal output; When the distance P [2] is less than 100mm, execute DO [4]=OFF.

Tips:

When the robot starts moving and meets the triggering conditions, it outputs the corresponding signal when it starts moving.

When CNT smoothing exists, the smooth transition does not pass through the target point, and there may be situations where the distance from the target point does not meet the triggering conditions, resulting in no signal output.

When the program pauses midway and resumes running, it will re evaluate the triggering conditions based on the current robot position as the starting point.

The specified distance signal output J joint command is invalid.

In actual operation, the specified distance signal output is related to the operating speed of the robotic arm, and there may be slight errors in the signal output point. This can be adjusted by reducing the speed of the output point or adjusting the specified distance value.

7.11.7 WAIT determine the trigger signal attribute

During the operation of the robotic arm from the current point to the target point, it determines whether the triggering conditions are met at the set distance. If the conditions are met, the robotic arm continues to move forward to the target point. If not met, the robotic arm decelerates and stops, waiting for the signal conditions to be met before starting to run to the target point.

Grammar format:

<Motion Type><Target Point> WAIT <signal expression> <DS|DE|DC=num>

Table 9-3 Detailed Parameter Manual

Trigger condition	**DS=xx**	**DE=xx**	**DC=xx**
Meaning	Robot TCP outputs signals when the distance from the starting point of robot operation exceeds xx mm	When the distance between the robot TCP and the target point of robot operation is within xx mm, output signal	Robot TCP has run the motion command line. When Robot TCP needs to move xx% of the entire distance, output a signal
Trigger condition	The robot leaves the starting point and the distance (instruction) between the robot and the starting point is greater than xx mm	The robot moves towards the target point, and the distance (instruction) between the robot and the target point is less than xx mm	The robot is moving towards the target point, and the percentage of the robot's TCP walking distance to the total length of the path is greater than xx
Output signal	When the robot meets the triggering conditions, signal output or shutdown is performed		

Example:

```
J P[0]
L P[1] WAIT DI[1]=ON DE=100    'Determine whether the signal of DI [1] is ON at a distance of 100mm from the endpoint of P [1]. If it is ON, move to point P [1], otherwise stop waiting.
L P[2]
```

Tips:

When the conditions are not met, the robotic arm decelerates and stops, then waits. At this time, the program is in a running state, and the robotic arm will automatically execute downwards after the signal is met.

7.12 Load setting

The PAYLOAD command is used to set the load number of the current end effector of the robotic arm. Load information includes information such as mass, center of mass, inertia, etc. The system has a total of 16 sets of load numbers, and users can hang different loads in practical applications, estimate and save them separately in each set of numbers.

Grammar format:

PAYLOAD=<-1~15>

Example:

```
PAYLOAD=1
J P[0]
L P[2]
```

Tips:

When PAYLOAD=-1, it indicates no load.

Only robotic arms that support dynamic models can use the load estimation function.

7.13WAIT waiting for instructions

The WAIT instruction is used to wait and determine if a condition is met. If the condition is not met, the program will block on this line until the condition is met before continuing execution. The WAIT instruction can be set to exceed the judgment.

Grammar format:

WAIT <bool expression> [TIMEOUT <int timeout> LBL <int nnum>]

Example 1：

```
    WAIT DI[1]=ON      'When DI [1] is not equal to ON, the program keeps blocking
    J P[1] VEL=50
    J P[2] VEL=50
```

Example 2：

```
    J JR[1] VEL=50
    WAIT DO[4]=ON TIMEOUT 500 LBL 2      'WAIT timeout instruction
    J P[1] VEL=50
    LBL [2]
    J P[2] VEL=50
```

Execution Logic Manual:

The robotic arm joint moves to JR [1], and then waits for the DO [4]=ON condition to be met. If the DO [4]=ON condition is met within 500ms, the instructions will be executed in sequence, and the joints will move to P [1] and P [2] in turn. If the DO [4]=ON condition is not met within 500ms, the timeout will occur, and the pointer will jump to the LBL [2] line and continue to execute from that line, directly moving the joints to point P [2] without executing J P [1].

7.14WAIT TIME delay instruction

The function of the WAIT TIME instruction is to delay the execution of a program (task) in milliseconds. In the background RUN program or the WHILE statement block of an infinite loop, the WAIT TIME instruction should be used for delay, otherwise there may be a problem of high CPU usage.

Example:

```
J P[1]
WAIT TIME=1000     'After moving to P1 and waiting for 1 second for sleep, VDI [1] is set to ON
VDI[1]=ON
J P[2]
```

7.15 Coordinate system instruction

The coordinate system instruction is used to call the coordinate system in the system, mainly to change the values of the coordinate system or make a coordinate transformation relationship effective. The coordinate system conversion diagram of the robotic arm is as follows:

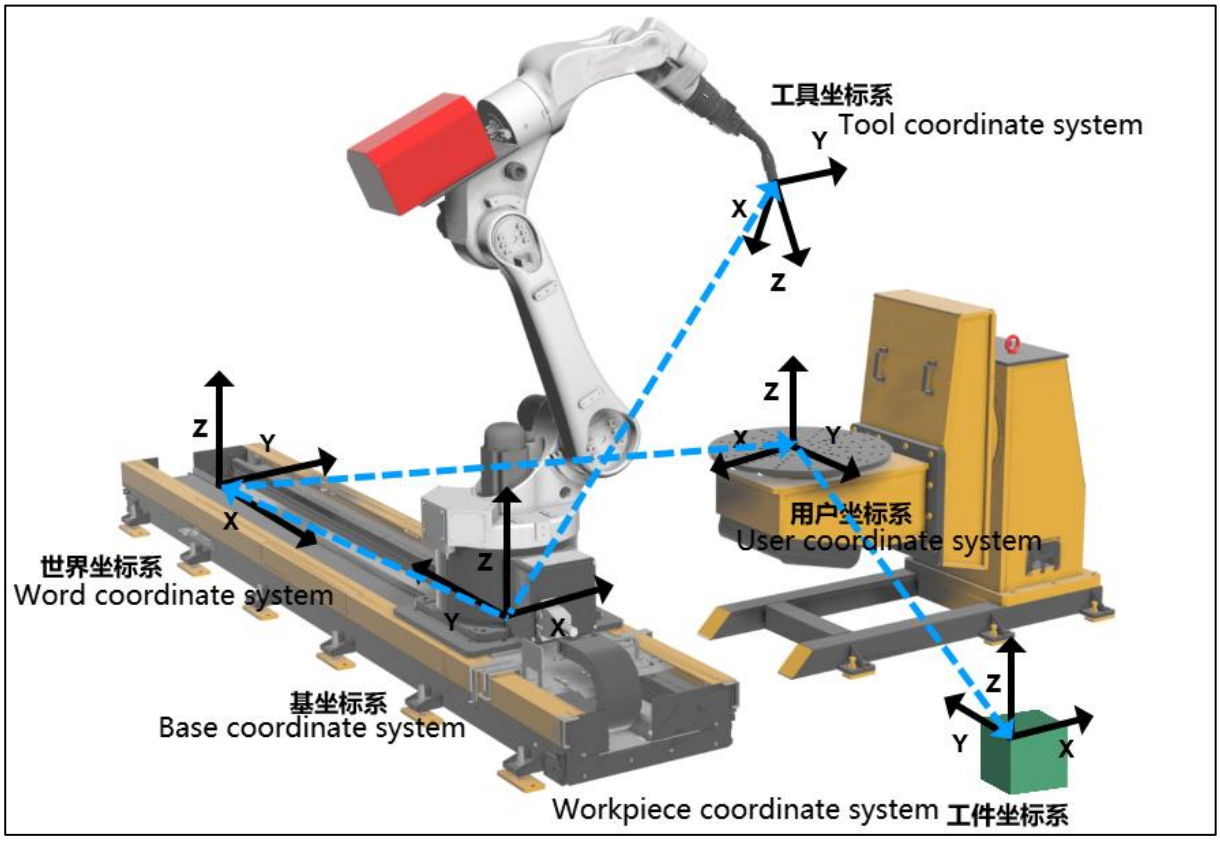


Figure 15-1 Coordinate System Conversion Relationship

7.15.1 UTOOL_NUM tool coordinate system call instruction

This instruction is used to call the TCP tool coordinate system that must be used in the program. When executing a motion to a spatial POINT (where the spatial POINT information includes the value of UT), the system will determine whether the current spatial POINT is consistent with the tool coordinate system being

used. If they are different, an alarm will be triggered stating that the POINT coordinate system is inconsistent. Therefore, before moving to a spatial POINT, the corresponding tool coordinate system must be called first. The value of UTOOL_UM is -1~15.

Grammar format:

UTOOL_NUM=<int expression>

Example:

```
UTOOL_NUM = 0          'Use the value of tool coordinate system 0
L P[1] VEL=50
L P[2] VEL=50
UTOOL_NUM = -1         'Switch the tool coordinate system to the default value of -1
J P[3] VEL=50
J P[4] VEL=50
```

Switching the tool coordinate system to the default value of -1 prompts:

When recording POINT during teaching, the current tool coordinate system used will be saved in POINT, so the corresponding tool coordinate system must be called in the PRG program.

7.15.2 UFRAME_NUM instruction for calling the workpiece coordinate system

This command is used to call the workpiece coordinate system to be used in the program. When executing a motion to a spatial POINT (where the spatial POINT information includes the value of UT), the system will determine whether the current spatial POINT is consistent with the currently used workpiece coordinate system. If they are different, an alarm will be triggered stating that the POINT coordinate system is inconsistent. Therefore, before moving to a spatial POINT, the corresponding workpiece coordinate system must be called first. The value of UFRAME_NUM is -1~15.

Grammar format:

UFRAME_NUM=<int expression>

Example:

```
UFRAME_NUM = 0          'Use the value of 0 in the workpiece coordinate system
L P[1] VEL=50
L P[2] VEL=50
UFRAME_NUM = -1         'Switch the workpiece coordinate system to the default value of -1
J P[3] VEL=50
J P[4] VEL=50
```

Tips:

When recording POINT in teaching, the current workpiece coordinate system used will be saved in POINT, so the corresponding workpiece coordinate system must be called in PRG program.

7.15.3 WORLDFRAME command to call the world coordinate system

This instruction is used to activate or deactivate the world coordinate system. The world coordinate system coincides with the base coordinate without calibration, that is, both are located at the base position of the robotic arm. In the application of ground rail+robotic arm collaboration, it is often necessary to use the world coordinate system. For detailed information on the collaborative motion functions of the robotic arm and ground rail, please refer to documents such as the "Instructions for the Use of Robotic Arm and Ground Rail Collaborative Functions".

Grammar format:

WORLDFRAME=<ON|OFF>

Example:

```
WORLDFRAME=ON       'Enable world coordinate system
UFRAME_NUM=1        'Call Workpiece 1
UTOOL_NUM=1         'Call Tool 1
SYNCEXTAX=ON        'Ground rail collaborative motion activated
```

```
L P[0]
L P[1] VEL=100
WORLDFRAME=OFF          'Close the world coordinate system
J P[2] VEL=100      'Time synchronized motion, POINT is the call to UF1 in the base coordinate system
```

7.15.4 SET_TOOL Tool coordinate system value modification instruction

The SET_TOOL instruction is used to modify the values of the tool coordinate system, and users can use the PRG program to modify the coordinate system of the current tool coordinate system.

Grammar format:

SET_TOOL <0~15> LR[unsigned int num]

Example:

```
UTOOL_NUM =1          'Call tool coordinate system 1
SET_TOOL 1 LR[3]     'Modify the value of tool coordinate system 1 and assign the data of LR [3] to tool coordinate system 1
L P[1]
```

7.15.5 SET_FRAME workpiece coordinate system value modification instruction

The SET_SRAME instruction is used to modify the values of the workpiece coordinate system, and users can use the PRG program to modify the coordinate system of the current workpiece coordinate system.

Grammar format:

SET_FRAME<0~15> LR[unsigned int num]

Example:

```
UFRAME_NUM = 1        'Call workpiece coordinate system 1
SET_FRAME 1 LR[5]    'Modify the value of workpiece coordinate system 1 and assign the data of LR [5] to workpiece coordinate system 1
L P[2]
```

7.15.6 CALIUFRAME calibration instruction for workpiece coordinate system

The CALIUFRAME instruction is used to generate a workpiece coordinate system instruction through three points, and users can calibrate and assign values to the current workpiece coordinate system using the PRG program.

Grammar format:

<location var_uf>=CALIUFRAME(< location origin> ,<location x_pos>, <location y_pos >)

Example:

```
LR[1] = CALIUFRAME(LR[2],LR[3],LR[4])      '3-point method for calibrating coordinate values
SET_FRAME 1 LR[1]                          'Set the calibrated coordinate values to workpiece 1
    L P[2]
```

Tips:

The workpiece coordinate system calibrated by the 3-point method is based on the OX axis. The UT and UF values of origin, x_pos, and y_pos should be consistent.

7.15.7 WCS_TEST coordinate system detection instruction

When the PRG program is executed, if the currently used tool workpiece coordinate system and the tool coordinate system number UT recorded in the spatial POINT are different from the workpiece coordinate system number UF, the system will alarm "POINT coordinate system inconsistency". The function of the WCD_TEST instruction is to check whether the currently used coordinate system is consistent with the coordinate system in the spatial POINT. WCD_TEST=0 means to turn off coordinate system detection and allow the currently used coordinate system to be different from the coordinate system in the spatial POINT.

Grammar format:

WCS_TEST=0|1

Example:

```
    UTOOL_NUM=1
```

```
    L P[1]      'P[1].UT=1, The current tool number used is consistent with the tool number in the spatial POINT, and the system does not alarm
    L P[2]
    UTOOL_NUM=2
    L P[3]      'P[3].UT=1 ,The current tool number used does not match the tool number in the spatial POINT, and the system will sound an alarm
    WCS_TEST=0   'Turn off detection
    UTOOL_NUM=2
    L P[3]      'P[3]. UT=1 , The current tool number used does not match the tool number in the spatial POINT, but coordinate detection is turned off and the system does not alarm. It will move to the point under UT [1]
```

7.16 Matrix arithmetic commands

7.16.1 MATMUL matrix multiplication instructions

This instruction multiplies two chi-square matrices represented by POINT data and returns them. Note that the left and right operands of the two sides of the instruction cannot be exchanged (matrix multiplication does not satisfy the exchange law), i.e., whether a is multiplied right by b, or b is multiplied right by a, is determined by the user's input.

Grammar format:

<target var> = <location var1> MATMUL <location var2>

Example:

```
  LR[1] = LR[1] MATMUL LR[2]    ' LR[1] right multiply by LR[2] and save in LR[1]
```

Limit Instructions:

The UT,UF,CFG of the target point <target var> all use the value of <location var1>, and only the <location var2> POINT value is used in the multiplication, ignoring the coordinate system, morphometric component in the POINT data. The above example assumes LR[1]. UT=1 , LR[1]. UF=1,

LR[2].UT=2, LR[2].UT=2 . When multiplying, only the 'POINT value' of LR[2] is taken, without considering the coordinate system of that POINT. The LR[1] point after multiplication means it is the point under UF1,UT1.

7.16.2 MATINV matrix inverse command

This instruction will invert the chi-square matrix represented by the POINT data and return it.

Grammar format:

<target var> = MATINV <location var1>

Example:

```
LR[1] = MATINV LR[2]
```

Restrictions on Instructions:

The UT,UF,CFG of the target point <target var> take the value of <location var1>.

7.17 Coordinate system conversion case

Case procedure 1: Transformation of space points under the workpiece coordinate system 1 to points under the base coordinate system.

```
UFRAME_NUM=1            ' Switch to use artefact 1. This instruction is not required, and need not be called if artefact 1 has already been called in a previous line of the program.
WAIT TIME=1             ' Waiting for motion to reach position
LR[1] = LPOS            ' Get the current actual position of the robot arm
LR[10]=UF[1]            ' LR[10] holds the values of the workpiece coordinate system 1.
LR[2] = LR[10] MATMUL LR[1]     ' Transforms to a point in base
```

```
coordinates .
```

Case Example Procedure 2: Transform the spatial points of the tool coordinate system 1 with respect to the base coordinates to the spatial points of the end flange with respect to the base coordinate system.

```
UTOOL_NUM = 1                    ' Use of tool 1
WAIT TIME=1                      ' Waiting for motion to reach position
LR[1] = LPOS                     ' Get points under tool 1
LR[11]= UT[1]                    'Getting the value of tool 1
LR[12] = MATINV LR[11]           ' Find the inverse matrix of tool 1
LR[2] = LR[1] MATMUL LR[12]      ' Transforms back to end flange
coordinates. lr[2] is the position of the end flange.
```

7.18 Co-motion commands

7.18.1 COORD_NUM synergy command

The COORD_NUM instruction is used to call the cooperative group number in the co-motion of the indexing machine. When you use the co ordinate function, you have to complete the configuration of the indexer axes, co ordinate group calibration and co ordinate trajectory teaching before you can call the co ordinate group in the programme and start the co ordinate motion. The value range of synergy group number (synergy coordinate system) is -1~4. The use of synergy function can be found in the operation and programming manual of Huazhou 3-type robotic arm.

Grammar format:

COORD_NUM=<int num expression>

Example 1:

```
    COORD_NUM = -1     'No co-ordinated motion, the arm moves in a
straight line to point P[1].
    L P[1]
    COORD_NUM = 3      'Calls synergy group number 3 and starts the
translocator synergy motion
```

```
    J P[2] VEL=50          'Synergistic movement of robotic arms and
displacements
    J P[3] VEL=50
```

Example2：

```
    COORD_NUM=1    'Global parameter, using co-group 1
    J P[1]
    J P[2]                 'Robotic arm and the first positioner working in
tandem
    COORD_NUM=-1    'Close Collaboration
    J P[3]                 'Movement of the robot arm to the transition point
    J P[4] VEL=100 ACC=100 COORD=2    'Local parameter, call synergy
group 2 to synergise with another transformer
```

Hints:

The program execution must call the co-group number correctly, i.e. it must call the co-group number used in the trajectory demonstration.

In the application of one robot arm + two translators, the co-motion supports local attributes for programming convenience.

7.18.2 SYNCEXTAX co-control commands

The simultaneous action of the robotic arm and the ground rail supports both forms:

(1) Time synchronisation, i.e. the ground track and the robot arm start and stop at the same time.

(2) Synergistic movement of the ground track and external axes ensures the trajectory accuracy and running speed of the end TCP.

The time synchronisation function cannot guarantee the accuracy of the end of the robotic arm and is only applicable to production line handling. The synergistic motion can ensure that the trajectory of TCP is straight line, which is suitable for welding and other high-precision requirement scenarios.SYNCEXTAX instruction is used to realise the ground track synergistic function to be turned on and off, and before using the ground track synergistic function, it is necessary to complete the configuration of the axis number of the ground track axes and the calibration of the coordinate system of the ground track first. For more details, please refer to

'Instruction Book of Robotic Arm and Ground Rail Synergy Function'.

Grammar format:

SYNCEXTAX=<ON|OFF>

Example:

```
WORLDFRAME=ON          'Use of geo-orbital coordinate system (world coordinate system)
UTOOL_NUM=1
UFRAME_NUM=1
SYNCEXTAX=ON           'Synergistic movement of the earth's orbit
L P[0]
L P[1]
SYNCEXTAX=OFF          'Conclusion of the ground-orbit concerted motion
J P[2]                 'time synchronous motion
```

7.19 EX_UTOOL external TCP command

The robot arm holds the processed object and moves around a centre point fixed on the ground (e.g., a spray gun or welding torch fixed on the ground). At this time, the centre point of the tool is fixed on the outside of the robot arm rather than at the end of the arm flange, and the robot arm is holding the workpiece in its hand for movement. When using the external TCP function, the calibration of the external TCP must be completed first, and then the external TCP command can be called in the PRG programme to control the movement of the robot arm.

Grammar format:

EX_UTOOL=<ON|OFF>

Example:

```
EX_UTOOL =ON          'Enabling External TCP
UTOOL_NUM =1          'Calling calibrated external tools1
L P[1]
L P[2]
```

```
EX_UTOOL =OFF       'Enabling External TCP
L P[3]              'Use flange tool 1 to move to point P[3].
```

7.20 Process control instruction

The programme control instructions are used for programme execution control, including LOAD load, START start, PAUSE pause, END stop, ABORT unload, as well as RUN instruction, MAINTASK, RUNTASK instruction. Users can control the execution status of foreground and background programs through program instructions. For example, communicating with the host computer in the background programme and controlling the execution of the foreground programme according to the logic requirements of the host computer.

7.20.1 MAINTASK foreground program name get command

The MAINTASK instruction is used to get the name of the foreground programme, the path and suffix are already included in the name of the programme obtained, such as '/MD/QULIAO/TEST1.PRG'. If there is no programme in the current task channel, then the obtained name is an empty string.

Grammar format:

<string var>=MAINTASK

Example:

```
SR[1]=MAINTASK      'Get the name of the frontend application
PAUSE SR[1]         'Suspension of front office procedures
```

7.20.2 RUNTASK background program name get command

The RUNTASK instruction is used to get the name of the background programme, the path and suffix are already included in the name of the obtained programme, such as '/MD/TT1.PRG'. If there is no programme in the current task channel, the obtained name is an empty string.

Grammar format:

<string var>=RUNTASK

Example:

```
SR[1]=RUNTASK       'Get the name of the backend application
```

```
PAUSE SR[1]          'Suspend background processes
```

7.20.3 RUN background program enable command

The RUN instruction is used to turn on the background programme to run. When the user uses the multitasking function, the user can configure the background programme to run automatically on power-up on the oscillator, or use the RUN instruction to turn on the background programme in the foreground motion programme, with the background logic programme group mask as *.

Syntax Format.

RUN <task name>

- The programme name is the absolute path name

Example:

```
J P[1]
RUN "TEST/A1.PRG"       'Enable background program execution
```

7.20.4 LOAD foreground program loading instruction

This directive can only load foreground programs and throws a warning message if the current main program has already been loaded.

Grammar format:

LOAD <task name>

- The programme name is the absolute path name

Example (written in a background RUN programme):

```
LOAD "A1.PRG"           'Load A1.PRG in the root directory
LOAD "TEST/A1.PRG"      'Loading A1.PRG in the TEST folder
```

7.20.4.1 START programme start command

The START instruction is used to start the execution of a foreground or background program that is in the state of 'Ready' | 'Pause', the instruction can be executed in the main program or RUN program, if the specified program is not in the state of 'Ready' | 'Pause', i.e., it cannot start the execution of the specified program, the

system throws an alarm ‘Cannot start xxx program execution’. If the specified program is not in the ‘Ready’ or ‘Pause’ state, it cannot start the specified program, and the system throws an alarm ‘Cannot start xxx program execution’.

Grammar format:

START <task name>

- The programme name is the absolute path name

Example (written in a background RUN programme):

```
SR[1]=MAINTASK     'Get the name of the frontend application
START SR[1]        'Initiate foreground program execution
```

7.20.5 PAUSE programme pause command

The PAUSE instruction is used to pause the program itself, or to pause the execution of the program with the specified program name, and can be used in the main program or in a RUN program.

The syntax is as follows:

PAUSE [programme name]

Example:

```
J P[1]
J P[2]
PAUSE       'Program execution reaches this line and the program status changes to paused
J P[3]
```

7.20.6 END Program Abort Instruction

The END instruction is used to abort the execution of its own programme or another programme, the programme pointer returns to the first line of the programme and the task is in the READY state. This instruction can be used in the main program or in a RUN program.

Grammar format.

END [programme name]

Example:

```
J P[1]
J P[2]
END          'Program execution reaches this line, the program ends, and the program pointer returns to the first line.
J P[3]
```

Hints:

If the END instruction is executed inside the CALL subroutine, the execution of the subroutine is aborted, and the subroutine will be jumped back to the main program to continue to the next execution (if the CALL is multi-layer nested, it will return to the previous one).

If the program specified by the END instruction has been loaded (i.e. in READY state), the execution of the END instruction will have no effect on the program; if the program specified by the END instruction is in RUN, PAUSE, or ERROR state, it can be aborted to return the task to READY state.

7.20.7 ABORT programme uninstallation command

The ABORT instruction is used to force the termination of a task and unload a programme. All movements controlled by the program's task are stopped. the ABORT instruction can abort its own program or other programs.

Grammar format.

ABORT [programme name]

Example:

```
J P[1]
J P[2]
ABORT        'Program execution reaches this line, stops and uninstalls the program
J P[3]
```

Tip:

If the ABORT instruction is executed inside a CALL subroutine, it is terminating the subroutine and the main program.

7.21 VORD trim command

VORD is used to adjust the speed multiplier of the robotic arm, when the programme executes the VORD instruction, no matter which mode the robotic arm is in, the speed multiplier is set to the specified value and maintain this speed multiplier until the outside world modifies the multiplier or until the next VORD instruction modifies it. the VORD value ranges from 1 to 100, and the user can also get the value of the multiplier of the current running speed through this instruction.

Grammar format:

VORD=<num expression>

<var num>=VORD

Example:

```
VORD=10      'Setting the VORD value
J P[1]
J P[2]
R[1]=VORD    'Get VORD value
```

7.22 TIME timing command

The TIME instruction is used to obtain the value of the system's clock, which can be used to calculate the time in ms for a programme to run.

Grammar format:

<var num> = TIME

Example:

```
R[1]=TIME
J P[1]
```

```
J P[2]
R[1]=TIME-R[1]
```

Execution Logic Manual:

The first line assigns the system time to R[1], then the robotic arm moves to point P[1] and P[2], and in the last line, the difference between the current time and the previous saved time is calculated and saved to the R[1] register, and by checking the value of the R[1] register, the length of the running time from P[1] to P[2] can be known.

7.23 JPOS | LPOS position acquisition command

The JPOS|LPOS instructions obtain the current position of the robot and assign it to the corresponding register variables JR[x]|LR[x]. JPOS obtains the current position of the robot in joint coordinates; LPOS obtains the position of the robot in Cartesian coordinates under the current coordinate system of the tool artefact.

Grammar format:

<joint var> = JPOS

<location var> = LPOS

Example:

```
JR[1] = JPOS        'Record the current joint coordinate position in JR[1] register
LR[1] = LPOS        'Record the current Cartesian coordinate position in LR[1] register
```

Hints:

When using the JPOS|LPOS instruction, the JPOS|LPOS instruction can only appear on the right side of an assignment to an equation, and JPOS can only pass values to the JR register and LPOS can only pass values to the LR register.

7.24TRACE conveyor tracking command

The system supports the sensor belt tracking function. To use the sensor belt tracking function, you must configure the parameters of the 7th axis and set the 7th axis as the encoder axis, and then perform the conveyor belt coordinate system calibration, trigger operation settings, and follow area settings. For details, see the Instructions for using the conveyor belt tracking function.

Example:

```
LBL[1]
L P[0]
TRACE NUM=0,SKIP LBL[3]   'Set up conveyor tracking, use group 0, jump to LBL[3] on exception
TPABS=1          'Ways to set the contour offset point to an absolute position
GET TP       'Blocking the acquisition of the triggered location TP point
J TP VEL=800     'Joint movement to TP point, synchronisation process
L P[1]         'reclaim point
L P[2]       'Rising point after pickup is complete
TRACE OFF        'off-track
L P[3]           'feed point
GOTO LBL[1]
LBL[3]
TRACE OFF
GOTO LBL[1]
```

7.25WEAVE swing command

Oscillating motion is an operation in which the end of the robot arm TCP performs a positively rotating oscillating trajectory. When using the swing function, the robot arm establishes a swing coordinate system and performs swing motion in the specified swing plane. Swing commands are divided into swing configuration commands, swing on and swing off commands.

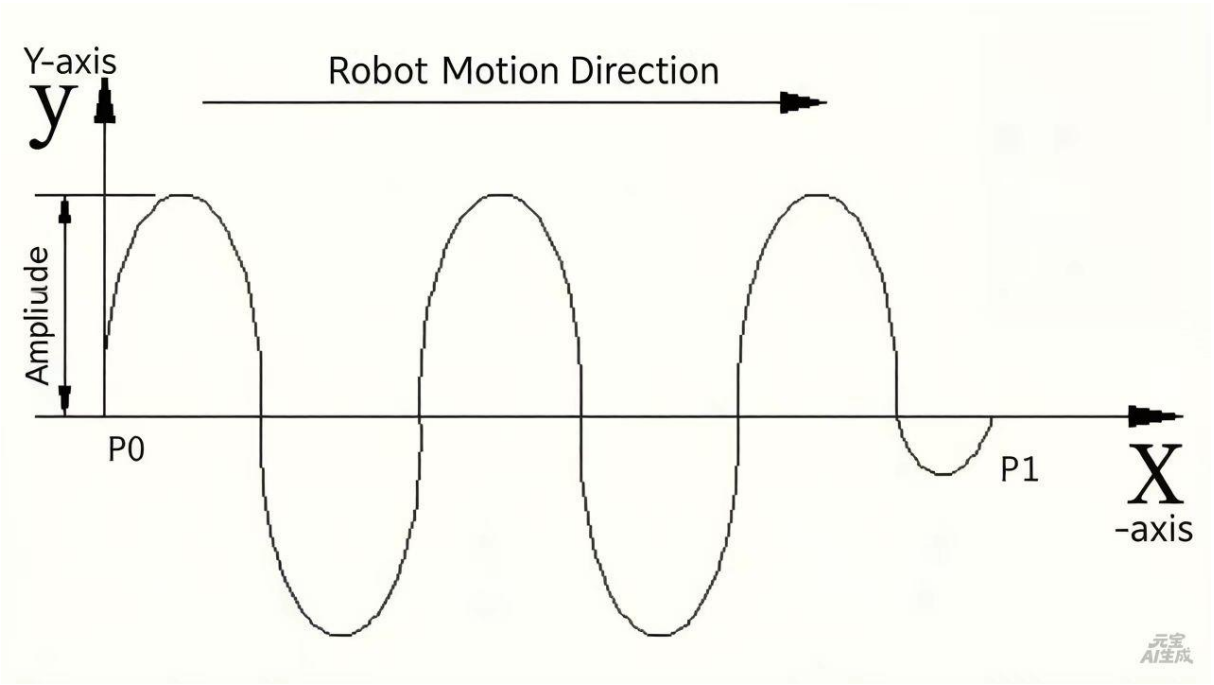


Figure 23-1 Schematic of WEAVE trajectory

Grammar format:

WEAVE NUM=<num > TYPE =<num> PLANE=<num> AMP= <num> FRE=<num>

WEAVE START NUM=<num>

WEAVE END

- NUM：The serial number of the swing group is specified, and multiple groups of swing parameters can be configured, and then the group of swing parameters will be called when the programme starts the swing movement. Currently, the system supports 10 groups of swing parameters, i.e. the value of NUM is 0~9.
- TYPE：Oscillating trajectory type, currently only positive spinning curves are supported, i.e. TYPE=0.
- PLANE：The plane in which the oscillating trajectory of the robot arm is located. 1 denotes the XY plane; 2 denotes the YZ plane; 3 denotes the XZ plane.
- AMP：The amplitude of the oscillation, i.e. the distance of the crest of the wave from the centre axis, in mm mm.
- FRE：The frequency of the oscillation, i.e. how many positive spinning trajectories are travelled in 1 second, in Hz Hz.

Example:

```
WEAVE NUM=0 TYPE=0 PLANE=1 AMP=10 FRE=5      'Configure parameters for swing group 0
```

```
J P[0]          'Movement to the starting point
WEAVE START NUM=0       'Oscillation start, start operation with set
oscillation parameter group 0
L P[1] VEL=100          'oscillatory motion
J P[2]                  'Normal execution of joint movements, no alarms
L P[3] VEL=200       'Oscillating motion, the value of the velocity vector in
the direction from P2 to P3 is 200mm/s
C P[4] P[5]          'oscillatory motion
WEAVE END         'end of swing
J P [6]                 'Return to waiting point L P[1]
```

7.26 SLAVE follow command

The follow-me compensation function is the function of the robotic arm to adjust the position in real time during the movement according to the set compensation value. The follow-me compensation function is, in principle, a synthesis of the main movement and the follow-me movement, and the speed value of the movement command is agreed to be the speed value of the main movement, not the speed value of the final synthesised movement.

Grammar format:

SLAVE MODE=0|1 CYCLE=<num> LR=<num> MASK=255|256

- MODE：Indicates the function is on or off, MODE value equals to 1 means turn on the follower, when the follower function is turned on, the current position will be used as the reference, MODE value equals to 0 means turn off the follower.
- CYCLE：Indicates the period of compensation data sampling issued, the value means a multiple of the interpolation period. For example, CYCLE=2 means 4*2=8ms, and the current compensation data value is sent every 8ms.
- LR：Indicates the LR register index value of the real-time compensation data value, the user calculates the compensation value in real time and writes it to this LR register, and the robotic arm carries out the follow function according to this value.
- MASK：Indicates the mask value. Meaning of the follower function with different MASK values. MASK=256

means position compensation in the direction of the swing coordinate system; MASK=255 means compensation in the direction of the tool coordinate system.

Example:

```
UTOOL_NUM = 1
L P[0] VEL=150 ACC=200 DEC=200 CNT = 0    'Movement to the starting point, using the current point as the reference point
SLAVE MODE=1 CYCLE=4 LR=0 MASK=255     'switch on the follow-through
LBL[1]          'trajectory
L P[1]
L P[2]
C P[3] P[4]
...
GOTO LBL[1]
SLAVE MODE=0 CYCLE=4 LR=0 MASK=255      'Turn off the follower
```

7.27 PALLET tray command

Tray command can be used for loading and unloading of 3C small materials, users only need to create a tray, and then enter the specified number of rows and columns to obtain the corresponding position, without the need to teach programming for each point.

Grammar format:

PALLET ID=<num> <P1> <P2> <P3> ROW=<nnum> COL=<num> LAY=<num> HEIGHT=<num>

- ID: indicates the serial number of the tray, supports unsigned integer values ranging from 0 to 15.
- P1: the origin of the tray; the line connecting P1 and P2 is the row direction of the tray; the line connecting P1 and P3 is the column direction of the tray.
- ROW: indicates the number of rows, supports unsigned shaping values ranging from 1 to 999.
- COL: indicates the number of columns, supports unsigned integer values ranging from 1~999.
- LAY: indicates the number of layers, supports unsigned integer values ranging from 1 to 999.
- HEIGHT: indicates the layer height, supports 64-bit floating point type positive number value range. Note that if

the layer height exceeds the travelling distance of the robot arm, the alarm will be raised at the time of executing the movement to the pallet POINT that the POINT is unreachable. When defining a pallet, it does not detect whether the POINT is reachable or not.

Example:

```
    PALLET ID=0 P[1] P[2] P[3] ROW=5 COL=4 LAY=2
HEIGHT=10    'Defines a 5-row, 4-column, 2-layer tray with an ID number of 0 and a height of 10mm per layer
    LR[1]=PALLET ID=0 ROW=1 COL=2 LAY=1    'Get the POINT of tray 0, row 1, column 2, level 1 and assign to LR[1]
    L LR[1]        'Linear motion to LR [1]
```

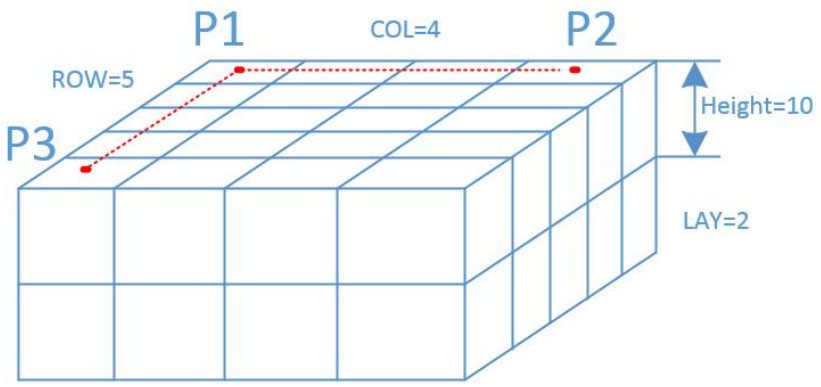


Figure 25-1 Pallet Schematic

Tips:

(1) When defining the pallet, it is required that the coordinate system of the tool artefacts used for the three points P[1], P[2], P[3] is required to be the same, if it is not the same an alarm will be thrown.

(2) single row | single column of the tray, according to the value of ROW and COL to determine, do not need to consider P [1], P [2], P [3] a two-point coincidence between whether or not, for example: a single row of the tray (ROW = 1), this time, do not take into account the P3 point, that is, the P3 point can not be set up or set up and the P1 point coincides. Single row tray is the

same.

Single row tray: PALLET ID=1 P[1] P[2] P[3] ROW=1 COL=4 LAY=2 HEIGHT=10

Single row tray: PALLET ID=1 P[1] P[2] P[3] ROW=4 COL=1 LAY=2 HEIGHT=10

7.28 SAVE variable save command

The SAVE instruction can be used to save the corresponding register variable in the variable list.

Grammar format:

SAVE FILE=<file name>

Example:

```
SAVE FILE= “LR"          'Save LR register file
```

Tip:

The name of FILE supports files for register variables such as UT, UF, R, JR, LR, SR, etc., as well as saving configuration files for other functions of the system.

7.29 PRINTF information printing command

This command prints display information directly on the Oscillator information display bar. When printing multiple messages, use a semicolon to separate them.

Grammar format:

PRINTF <string> <string|double> {;<string> [string|double]}

Example:

```
SR[1]=“tom”
SR[2]=“jerry”
R[1]=10
R[2]=20
```

```
PRINTF "height:" R[1]; "thickness:" R[2]      'Height:10;Thickness:20
PRINTF SR[1]
PRINTF SR[1] SR[2]      ' Syntax error, missing data item
```

7.30 THROW custom alarm commands

The THROW instruction is used to throw user-defined alarm information. When the corresponding alarm level signal is thrown, the system will make corresponding operation according to the alarm level.

Grammar format:

THROW <string> LEVEL=FATAL|ERROR|WARNING|NOTE|INFO

- FATAL: Critical unrecoverable error, the robot stops moving, the motion buffer is emptied, the programme enters an error state and the cache is emptied, unrecoverable (the programme cannot be resumed).
- ERROR: Recoverable error, the robot stops moving, the programme enters an error state, but the motion buffer is not emptied, and the operation can be resumed.
- WARNING: Warning, the system is in a critical state or the user operation may lead to undesirable consequences, such as prompt messages, will not cause the programme or movement to stop.
- NOTE: NOTE, a general message giving guidance to the user.
- INFO: Information, general system status information or user action records that are not prompted by the user.

Example:

```
THROW “Abnormal clamping of jaws” LEVEL=ERROR
```

7.31 Maths function instructions

Maths arithmetic function instructions are included:SQRT、SIN、COS、TAN、ASIN、ACOS、ATAN、ATAN2、ABS、TRUNC、ROUND。

Table 29-1 Detailed instructions for mathematical functions

function name	Instructions	Range of independent variables	give an example	
SQRT	square root	[0,+∞]	R[1]=4 R[2]= SQRT R[1]	R[2]=2
SIN	sine	[-∞,+∞]	R[1]=90 R[2]= SIN R[1]	R[2]=1
COS	cosine	[-∞,+∞]	R[1]=90 R[2]= COS R[1]	R[2]=0
TAN	tangent	Floating point numbers other than 90±180	R[1]=45 R[2]= TAN R[1]	R[2]=1
ASIN	inverse sine	[-1,1]	R[1]=1 R[2]=ASIN R[1]	R[2]=90
ACOS	inverse cosine	[-1,1]	R[1]=1 R[2]=ACOS R[1]	R[2]=0
ATAN	inverse tangent	[-∞,+∞]	R[1]=1 R[2]=ATAN R[1]	R[2]=45
ATAN2	inverse tangent	Except X=0 and Y=0	R[1]=-1 R[2]=0.5 ATAN2 R[1] R[2] , OUT R[3]	R[3]= -63.435
ABS	absolute value	[-∞,+∞]	R[1]=-100 R[2]=ABS R[1]	R[2]=100
TRUNC	Rounding (rounding off decimal values)	[-∞,+∞]	R[1]=5.5 R[2]=-5.2 R[3]=TRUNC R[1] R[4]=TRUNC R[2]	R[3]=5 R[4]=-5
ROUND	Rounding (to the	[-∞,+∞]	R[1]=5.5	R[3]=6

	nearest whole number)		R[2]=-5.2 R[3]=ROUND R[1] R[4]=ROUND R[2]	R[4]=-5
DIV	Left and right division operations to take integers	[-∞,+∞]	R[1]=5 R[2]=2 R[3]=R[1] DIV R[2]	R[3]=2
MOD	Left and right division operations to take the remainder	[-∞,+∞]	R[1]=5 R[2]=-2 R[3]=R[1] MOD R[2]	R[3]=1

Hints:

(1) The independent variable input and output in trigonometric functions are angle values.

(2) If the denominator in the calculation result is 0, or the dependent variable exceeds the range will report an error.

(3) Support and mathematical operators mixed operations, such as R[2] = 30 + ROUND R[3], also supports the function instruction nested operations. For example, it supports R[2]=ABS SIN 60.

(4) When using the ATAN2 instruction, it is necessary to take it out and use it separately as an independent programme line, store the result of the calculation directly into the R register after OUT, and then use the R register to participate in the mixed operation or make a judgement.

Example:

```
ATAN2 R[2] R[1], OUT R[3]
R[4]=R[3]*6+SQRT R[3]
IF R[3]>100，GOTO LBL [1]
```

7.32 String instruction

String instructions can be used to convert, cut, splice, and lookup strings. The current system supports global string variables SR[x], each SR[x] variable can store 512 characters.

7.32.1 TOSTR convert string instruction

The TOSTR instruction enables the conversion of values to strings.

Grammar format:

<string var> = TOSTR(<num expression>)

Example:

```
SR[1] =TOSTR(R[1])
```

Table 30-1 TOSTR Conversion

R[1] value	SR[1] value
1234	“1234”
34.152	“34.152”
23.123456789	“23.123456”

Tip:

Only 6 decimal places are retained when converting floating point numbers.

7.32.2 TOVAL convert numeric instruction

The TOVAL instruction enables the conversion of a string to a numeric value.

Grammar format:

<num var> = TOVAL(<string expression>)

Example:

```
R[1] =TOVAL(SR[1])
```

Table 30-2 TOVAL Conversion

SR[1] value	R[1] value
“1234”	1234

“34.152”	34.152
“98abc”	98
“abcd”	0

Hints:

When SR[i] contains illegal characters, it will convert the correct part before the illegal characters, if all of them are illegal characters, the conversion result will be 0.

7.32.3 STRLEN get string length command

Use the STRLEN instruction to get the length of the string.

Grammar format:

<num var> = STRLEN(<string expression>)

Example:

R[1]=STRLEN(SR[1])

R[2]=STRLEN(“hello”)

Table 30-3 STRLEN Conversion

SR[1] value	R[1] value
“1234”	4
“34.152”	6
“a,,.d;!p”	8
“”	0

7.32.4 FINDSTR lookup string command

The FINDSTR instruction implements a string lookup function. It searches for the target string from left to right in the string, and returns the first occurrence if it is found, or 0 if the string is not found.

Grammar format:

<num var> = FINDSTR (<string1> ,<obj string2>)

Example:

```
R[1]=FINDSTR(SR[1], SR[2])
```

Table 30-4 FINDSTR Conversion

SR[1] value	SR[2] value	R[1] value
“hello world!”	“ell”	2
“hello world!”	“no”	0

7.32.5 SUBSTR intercept string instruction

The SUBSTR instruction enables string interception.

Grammar format:

<string var> = SUBSTR(<obj string>, <start index>, [char length])

- Obj string: the string to be intercepted, this item is mandatory.
- start index: the start index of the string to be intercepted, when its value is equal to 0 or 1, it will start from the first position. If it is negative, the start position is counted backward from the last character, and then the length of the specified character is intercepted from that position in the downward direction.
- char length: character length value, specifies the length of the character to be intercepted, is optional, if not written then the default return from the specified position after the start of all the characters, SUBSTR (‘hello’, 2) ---> return ‘ello ’, SUBSTR (‘hello’, -2) ---> return ‘lo’; if the value of the length of the character if more than the value of the length of the object string, then return all the object string.

Example:

```
SR[1]=SUBSTR (SR[2], R[1], R[2])
```

Table 30-5 SUBSTR Conversion

SR[1] value	SR[2] value	R[1] value	R[2] value
“ell”	“hello world!”	2	3
“world!”	“hello world!”	7	10
“!”	“hello world!”	-1	2
“ld”	“hello world!”	-3	2

7.32.6 String splicing

Grammar format:

<string var1> = <string expression1> + <string expression2>

Use the plus sign to perform string concatenation functions. The operands on both sides of the plus sign are required to be of the same string type.

7.32.7 Character cutting case procedure

Example:Cut SR[0]='T1,1.123,2.0,3.0' to SR[1]=T1 SR[2]=1.123 SR[3]=2.0 SR[4]=3.0

```
R[0] = FINDSTR(SR[0],",")              ' SR[0]=T1,1.123,2.0,3.0     R[0]=3
SR[1]= SUBSTR(SR[0],0,R[0]-1)        ' Intercept T1 on the left，SR[1]=T1
SR[0]= SUBSTR(SR[0],R[0]+1)        ' Intercept the right part of the string

R[0] = FINDSTR(SR[0],",")          ' SR[0]=1.123,2.0,3.0
SR[2]= SUBSTR(SR[0],0,R[0]-1)    ' Intercept 1.123 on the left，  SR[2]=1.123
SR[0]= SUBSTR(SR[0],R[0]+1)      ' Intercept the right part of the string, after the interception SR[0]=2.0,3.0

R[0] = FINDSTR(SR[0],",")          ' SR[0]=2.0,3.0
SR[3]= SUBSTR(SR[0],0,R[0]-1)    ' Intercept 1.123 on the left   SR[3]=2.0
SR[0]= SUBSTR(SR[0],R[0]+1)      ' Intercepting the right part of the string, SR[0] = 3.0 after interception
SR[4]=SR[0]              'SR[4]=3.0
```

7.33 SOCKET communication commands

The system supports the SOCKET communication function, which has two functional forms.

The first is the protocol-free SCOKET character communication, users can call the SOCKET function in the background programme to establish communication, and then complete the operation of sending and receiving data. The system supports network connection status monitoring, and will automatically alarm when communication is abnormal. This form of robotic arm can be used as a client or server; as a server allows multiple clients to connect and respond to multiple client data. The specific functions are described in the following chapters.

The second is the SOCKETCMD network terminal function. The system provides a set of defined SOCKETCMD character stream protocol, the host computer sends corresponding strings to the robotic arm

according to the protocol to control the robotic arm to enable, move to the point, load and run the program and other operations, or get the status data of the robotic arm, such as IO, current position, etc. For details, see the 'V1.6.11_SocketCmd Secondary Development Communication Protocol Manual_A01_CN'. For detailed functions, please refer to 'V1.6.11_SocketCmd Secondary Development Protocol Manual_A01_CN

Instructions: In the following SOCKET function instructions, all of them can accept or not accept the return value of the function. That is, the following programming is correct

R[1]=OPENSOCKET OPTIONS=0 HANDLE=1 'Receiving the return value

OPENSOCKET OPTIONS=0 HANDLE=1 'Does not receive a return value.

7.33.1 OPENSOCKET creation command

Creates a socket. The syntax format is as follows:

INT OPENSOCKET OPTIONS=<int num> HANDLE=<int device number>

- OPTIONS: 0 means the data is buffered until the buffer is full or timeout, this mode will be used to improve network utilisation; 1 means no delay, send the data immediately.
- HANDLE: i.e. device handle, the value range is 1~255.
- Function return value, create success: 0, create failure: different reasons for different return values, see the attached table for details, 'SOCKET error code'.

Example:

```
OPENSOCKET OPTIONS=0 HANDLE=1          'Does not receive a return value
R[0]=OPENSOCKET OPTIONS=0 HANDLE=1     'Receiving the return value
```

7.33.2 CLOSESOCKET shutdown command

Closes the socket. The syntax format is as follows:

INT CLOSESOCKET HANDLE=<device number>

- Function Return Value, Close Success: 0, Close Failure: Different return values for different reasons, see the attached table 'SOCKET Error Code' for details.

Example:

```
CLOSESOCKET HANDLE=1          'Does not receive a return value
```

```
R[0]=CLOSESOCKET HANDLE=1          'Receiving the return value
```

7.33.3 ISHANDLEOPEN judgement instruction

The ISHANDLEOPEN instruction is used to determine whether the socket is open and the communication connection is successful. A return of true indicates that the socket is open and the communication connection is successful, while a return of false indicates that the socket is not open or the communication is not connected. When converting a Boolean variable to floating point, 1.0 means true and 0 means false.

Grammar format:

BOOL ISHANDLEOPEN(int <device number>)

Example:

```
R[1]=ISHANDLEOPEN(1)
```

7.33.4 The CONNECT command

Connection server for situations where the robotic arm is acting as a client.

Grammar format:

INT CONNECT(int <device number>, string <ip address>, int <port number>, {int timeout})

- device number: device handle, the socket created by opensocket.
- ip address: IP address, IP address in string form.
- port number: port number, 30000-65535.
- timeoout: timeout time in milliseconds, the system default connection timeout is 3000 milliseconds (3 seconds), and it will exit automatically after the timeout. This parameter is optional.

Function return value: connection success: 0; connection failure, timeout return: different reasons for different return values, see the attached table 'SOCKET error code' for details.

Example:

```
OPENSOCKET OPTIONS=1 HANDLE=20
R[0]=CONNECT(20,"127.0.0.1",30000)
```

Tip:

The return value of the function is an optional parameter, the user can choose

whether to receive the return value or not, if the timeout does not connect to the server then the function returns.

7.33.5 ACCEPT monitor instructions

ACCEPT instruction is used in the case of a robotic arm as a server, the server side opens a socket (handle=1 in this case), and then blocks waiting for the client to connect (accept), when the client connects, the server side creates a new socket socket and returns to save it in fd, the server side uses this new socket to interact with data from the client that is already connected, and again blocks listening to wait for the next client to connect. The server uses this new socket to interact with the connected client and again blocks listening for the next client to connect. You can send information to the corresponding fd to achieve one-to-many communication.

The syntax is as follows:

INT ACCEPT(int <decvice handle> , string <ip>, int <port>, int& <fd>, {int timeout})

- decvice handle: Device handle, a socket created with the opensocket command to listen for client connections.
- ip: Local IP address string.
- port: the port on which the machine receives data, 30000-65535.
- fd: the handle of the newly created socket socket when the client connects, the server uses this new socket to interact with the connected client.
- timeoout: timeout in milliseconds, the default timeout is 3000 milliseconds. This parameter is optional.

Function return value: client connection is successful and return: 0; listening timeout return: different reasons for different return values, see the attached table 'SOCKET error code' for details.

Example:

```
OPENSOCKET OPTIONS=0 HANDLE=1
R[0] = ACCEPT(1,"127.0.0.1",30000, R[1] )     'Use R[1] to receive connected sockets
If
  Throw
Endif
```

```
SEND(R[1], "hello world”)
```

7.33.6 SEND send command

The SEND instruction is used to send data, both server-side and client-side through this function. Since the maximum user buffer is 512 bytes, the maximum amount of data that can be sent at one time is 512 bytes, and it will be automatically truncated if it exceeds that amount.

Grammar format:

INT SEND(int <decvice handle>, string <data>)

- decvice handle: Device handle, socket created using the opensocket command.
- data: the string to be sent.

Function return value: send successfully returns the number of bytes sent successfully; send failure: different reasons for different return values, see the attached table ‘SOCKET error code’ for details.

Example:

```
OPENSOCKET OPTIONS=0 HANDLE=20
R[0]=SEND(20,"hello world”)   ’Sends hello world , returns 11 if successful.
SEND(20,SR[1])                ’Sends the value inside the SR[1] register variable
```

7.33.7 RECEIVE receive command

RECEIVE function is used to receive data, if receiving data is unsuccessful, such as handle is not created, then the function returns -1, timeout does not receive the data returns 0. Whether it is the server or the client to receive data through this function. Because the user buffer is 512 bytes, so the single reception of up to 512 bytes of data, if the host computer sends more than 512 data, must be received several times. Users can customise the format frame in engineering practice to ensure the reliability of communication.

Grammar format:

INT RECEIVE(int <decvice handle>, string& <data>, {int timeout})

- decvice handle: device handle, socket created with opensocket command.
- data: save the string to be retrieved.

- timeoout: timeout in milliseconds, the default receive timeout is 200 milliseconds. This parameter is optional.

Function return value: the number of bytes received is returned if the reception is successful; reception failure: different reasons for different return values, see the attached table 'SOCKET error code' for details.

Example:

```
OPENSOCKET OPTIONS=0 HANDLE=20
R[0]=RECEIVE(20, SR[1])      'Receive data and save in SR[1] variable
```

7.33.8 BRECBUFEMPTY directive

The BRECBUFEMPTY instruction is used to determine if there is data in the receive buffer.

Grammar format:

INT BRECBUFEMPTY (INT HANDLE)

Function Return Value: 0 means the cache is empty; 1 means there is data in the cache.

Example:

```
OPENSOCKET OPTIONS=0 HANDLE=20
R[0]=BRECBUFEMPTY (20)      'Determine if there is data in the receive buffer
```

7.33.9 SOCKETMONITOR monitor command

SOCKETMONITOR is used to set up asynchronous monitoring of the communication connection status, and the user can set up whether or not to alarm when the connection is disconnected. The system monitors the situation from [Connected] to [Disconnected]. When the connection is disconnected, the system automatically throws an alarm | warning.

Grammar format:

SOCKETMONITOR(int <device handle>, int <leve>)

- decvice handle: device handle, use opensocket command to create socket.
- level: response level, 1: alarm if disconnected, PRG programme stops running; 2: warning is thrown if disconnected, no alarm, PRG programme executes normally.

Example:

```
SOCKETMONITOR(10, 1)
```

7.34 MODBUS master communication command

The system supports MODBUS communication function. When the robot arm is used as a MODBUS master, the user has to use commands to complete the read/write business logic inside the PRG; when the robot arm is used as a slave, the system provides ModbusCmd external control protocol, and the user only needs to turn on the ModbusCmd function switch on the UI of the demonstrator, and wait for the upper computer to connect to establish communication.

The robotic arm can be a master and a slave at the same time. Because the protocol-less universal Modbus function (in the form of commands) and the ModbusCmd external control protocol function are independent functions, the memory opened by the two functions is independent of each other. The robot arm can be used as a master to communicate and interact with the PLC and other devices on the production line using the Modbus function commands, and at the same time, the robot arm can be used as a slave to communicate and interact with the touch screen using the ModbusCmd function, as described in the 'Instructions for the use of the Modbus Slave Control Protocol'.

When the robot arm acts as a master, the user needs to write the PRG programme and load and execute the relevant commands before the communication is established. The instruction is a triggered operation, and the user will only read or write the operation when the instruction is called and executed. Users can use while loop inside the background programme to achieve cycle read/write operation.

Instructions：All of the following function instructions may or may not accept the return value of a function. That is, all of the following programming is correct

```
R[1]=OPEN_MODBUS_TCP("10.10.56.100",502,1,0)     'Receiving the return value
OPEN_MODBUS_TCP("10.10.56.100",502,1,0)          'Does not receive a return value
```

7.34.1 OPEN_MODBUS_TCP create command

Create a MODBUS master, and the robot arm communicates with the slave as the master (client). In case of one-to-many, the corresponding IP address and port number must be called for communication separately, and the port number cannot be the same.

Grammar format:

INT OPEN_MODBUS_TCP(int <SlaveID>, string <IP>, int <Port>, int <HighLowType>)

- SlaveID: Slave ID number, 1~255.
- TCP/IP: IP address of the slave device (i.e. server).
- Port: Port number of TCP communication connection, 502, 503, 504, 505, 506, etc. can be used.
- HighLowType: the order of high and low bits, CDAB, then use 0 to indicate, ABCD, then use 1 to indicate.

The function returns a value of 0 for success, -1 for unsuccessful master creation, and -2 for unsuccessful slave connection. If there is an exception, the controller throws a warning message corresponding to the return value.

Example:

```
    R[1]=OPEN_MODBUS_TCP(1,“10.10.56.100”,502,0)    ’Open and connect slave 1
    R[2]=OPEN_MODBUS_TCP(2,“10.10.56.101”,503,0)    ’Open and connect slave 2
```

7.34.2 CLOSE_MODBUS_TCP shutdown command

The robot arm acts as a master (client) to close the connection with the slave of this ID. The function returns a value, 0 for success, -1 for failure.

Grammar format:

INT CLOSE_MODBUS_TCP(int <SlaveID>）

Example:

```
    R[1]= CLOSE_MODBUS_TCP(1)       ’Close communication with slave 1.
```

7.34.3 OPEN_MODBUS_RTU create command

When the robot arm communicates with multiple slaves as a master, the robot arm communicates with the slaves as a master (client). For one-to-many, the corresponding serial port number and slave station ID must be called up separately.

Grammar format:

INT OPEN_MODBUS_RTU(int<SlaveID>, int<SerialPort>, int<Baud>, int<DataBit>, int <StopBit>, int<ParityCheck>, int <HighLowType>)

- SlaveID: ID number of the slave, 1~255.
- SerialPort: serial port number.
- Baud: Baud rate, can choose 9600, 115200, etc.
- DataBit: Data bit, can be set to 5, 6, 7, 8.
- StopBit: Stop bit, can be set to 1, 2.
- ParityCheck: Parity check. 0 means none, 1 means odd check, 2 means even check.
- HighLowType: the order of high and low bits, CDAB, then use 0 to indicate, ABCD, then use 1 to indicate.

The function returns a value of 0 for success, -1 for unsuccessful master creation, and -2 for unsuccessful slave connection. If there is an exception during creation, the controller throws a warning message corresponding to the return value.

Example:

```
    R[1]=OPEN_MODBUS_RTU(1，3,9600,8,1,0,1)  'As master, connect the slave with ID=1, using serial port 3.
    R[1]=OPEN_MODBUS_RTU(2，4,9600,8,1,0,1)  'As master, connect the slave with ID=2, using serial port 4
```

7.34.4 CLOSE_MODBUS_RTU shutdown command

The robotic arm acts as a master and closes the MODBUS RTU communication with the slaves.

Grammar format:

INT CLOSE_MODBUS_RTU(int SlaveID)

Example:

```
    R[1]=CLOSE_MODBUS_RTU(1)      'Disables communication with slave 1.
```

7.34.5 READ_MDB_COILSTATUS read command

The robotic arm acts as a master to read the coil data from the slaves.

Grammar format:

INT READ_MDB_COILSTATUS(int<SlaveId>, int<StartAddr>, int<CoilNum>, bool <Var>)

- SlaveId: Slave address.

- StartAddr: start address of coil state (0-09999).
- CoilNum: number of coils to be read.
- Var: IO type variable, used to store the value of the read coil, can use DO[x]|R[x] variable to save.

The function returns 0 for a successful read and -1 for a failed read.

Example:

```
READ_MDB_COILSTATUS(1,1,2,R[1])    'Read the values of the 2 coils of slave 1 starting from address 1 and store them in the R[1], R[2] variables
```

7.34.6 WRITE_MDB_COILSTATUS write command

The robotic arm serves as the master station to write data to the slave station coil.

Grammar format:

INT WRITE_MDB_COILSTATUS(int<SlaveId>, int<StartAddr>, int<CoilNum>, bool <Value>)

- SlaveId: slave address.
- StartAddr: start address of the coil (0-09999), unit bit.
- CoilNum: number of coils to be written.
- Value: value of bit type (0-255), set the coil value of ModBus slave according to the value of this variable.

The function returns 0 for a successful write and -1 for a failed write.

Example:

```
R[1]=1
R[2]=1
R[3]=0
WRITE_MDB_COILSTATUS(1,0,3,R[1])    'Set the values in coil address 0, 1, and 2 of slave 1 to 1, 1, and 0, in that order.
```

7.34.7 READ_MDB_INPUTSTATUS read command

The robotic arm acts as a master reading the slave discrete coil data inputs.

Grammar format:

INT READ_MDB_INPUTSTATUS(int<SlaveId>, int<StartAddr>, int<Num>, bool <Var>)

- SlaveId: Slave address.

- StartAddr: start address of the coil state (0-09999).
- Num: number of discrete inputs to be read.
- Var: used to store the value of the read coil, can use DO[x]|R[x] register.

The function returns 0 for a successful read and -1 for a failed read.

Example:

```
READ_MDB_INPUTSTATUS(1,0,2,R[10])    'Read the value of the 2 discrete inputs from slave 1 starting at address 0 and store them in R[10], R[11]
```

7.34.8 READ_MDB_INPUTREG read command

The robotic arm acts as a master to read data from the slave input registers.

Grammar format:

INT READ_MDB_INPUTREG(int<SlaveId>, int<StartAddr>, int<Num>, dWord <Var>, int <DataType>)

- SlaveId: Slave address.
- StartAddr: start address of coil state (0-09999).
- Num: number of input registers to be read.
- Var: used to store the input registers read, can use R[x] variable.
- DataType: data type of input register|holding register, use 0 to indicate 16-bit INT; use 1 to indicate 32-bit DINT; use 2 to indicate 32-bit FLOAT.

The function returns 0 for a successful read and -1 for a failed read.

Example:

```
R[1]=READ_MDB_INPUTREG(1,0,2,R[10],1)    'Read the values of the 2 input registers of slave 1 from address 0 and store them in R[10] and R[11].
```

7.34.9 READ_MDB_HOLDREG read command

The robot arm acts as a master to read the slave save register data.

Grammar format:

INT READ_MDB_HOLDREG(int<SlaveId>, int<StartAddr>, int<Num>, dWord <Var>， int <DataType>)

- SlaveId: slave address.

- StartAddr: start address of the register (0~09999).
- Num: number of read data.
- Var: save the address of the holding register.
- DataType: data type of input register|holding register, use 0 to indicate 16-bit INT; use 1 to indicate 32-bit DINT; use 2 to indicate 32-bit FLOAT.

The function returns 0 for a successful read and -1 for a failed read.

Example:

```
R[1]=READ_MDB_HOLDREG(1,0,2,R[10],1)    'Read the 2 elements starting from 0 of slave 1 and store them in R[10], R[11]
```

7.34.10 WRITE_MDB_HOLDREG write command

The robot arm acts as a master to write to the slave to save the register data.

Grammar format:

INT WRITE_MDB_HOLDREG(int<SlaveId>, int<StartAddr>, int<Num>, dword <Value>，int <DataType>)

- SlaveId: slave address.
- StartAddr: start address of the register (0~09999).
- NUM: number of registers to be written.
- Value: Write this value to the corresponding address of Modbus.
- DataType: data type of input register|holding register, use 0 to indicate 16-bit INT; use 1 to indicate 32-bit DINT; use 2 to indicate 32-bit FLOAT.

Example:

```
R[10]=1
R[11]=1
R[12]=0
R[1] = WRITE_MDB_HOLDREG(1，0, 3，R[10],1)  'Write the data R[10]=1, R[11]=1, R[12]=0 to the 3 elements after the start address of slave 1 is 0
```

7.35 Macro instruction

7.35.1 HANDPICK jaw commands

This instruction is used for the gripper function application, which can control multiple output signals and monitor and judge multiple input signals at the same time to achieve the processing of signal linkage and inspection. The application for when the signal triggered to open or close the gripper jaws, and then judge whether to open or close in place, not in place, then need to be alarmed output of the combination of functions of the process application.

Grammar format:

MAC HANDPICK DO=<num expression> VALUE=ON|OFF DI=<num expression> VALUE=ON|OFF T=<time num>, OUT;

- DO: Output index number indicating the IO number associated with the gripper open/close action. The following VALUE option value indicates that the DO is equal to ON | OFF when the gripper is opened | closed.
- DI: Input index number, indicates the IO number associated with detecting whether or not the gripper open/close action is in place, followed by a VALUE option value that indicates that the gripper open/close is in place when the DI is equal to ON|OFF.
- T: timeout detection time, the unit is ms, that when DO = ON, the system waits for DI = ON signal to detect whether the signal in place, if T = 0, the system has been waiting for the signal, if T > 0, if beyond the set time clamping jaws are not yet in place, the system will issue an alarm, the alarm content for the ‘clamping jaws to open or clamping abnormality’. If T=0, the system waits for the signal.

Instructions Example:

```
MAC HANDPICK DO=0 VALUE=ON DI=0 VALUE=ON T=1000，OUT;
L P[0]
```

The effect of the above statement line is similar to the following program line:

```
DO[0]=ON
WAIT TIME=1000
IF DI[0]=OFF THEN
THROW “Abnormal jaw opening or clamping” LEVEL=ERROR
ELSE
L P[0]
```

```
    END IF
```

Program Example:

```
    L P[0] VEL=1500 CNT=100 ACC=300 DEC=300     'Move to teach point P0
    MAC HANDPICK DO=0 VALUE=ON DI=0 VALUE=ON T=1000, OUT;    'When the target point P[0] is reached, the 'open output signal' output signal DO[0] will be set to TRUE, waiting for the 'open detection signal' input signal DI[0] to be set to TRUE, and the system will alarm if the waiting time exceeds 1 second. 'Jaw opening or abnormal clamping'.
    R[0]=1     'R[0] assigned to 1
    L P[1] VEL=1500 CNT=100 ACC=300 DEC=300 'Move to teach point P1
    MAC HANDPICK DO=R[0] VALUE=ON DI=R[0] VALUE=ON T=2000, OUT; 'When the target point P[1] is reached, the 'open output signal' output signal DO[R[0]] will be set to TRUE, wait for the 'open detection signal' input signal DI[R[0]] to be set to TRUE, and the system will alarm when the waiting time is exceeded by 2 seconds. 'Jaw opening or abnormal clamping'.
```

Tip:

After the alarm is triggered, the alarm will be stopped, the alarm can be reset, the statement will be re-executed after a successful reset, and if the conditions are not met, the alarm will continue to be triggered.

7.35.2 SOFTFLOAT soft float command

The soft float function allows the arm to float in response to an external force in the direction of the force when the arm is subjected to an external force in the stationary state. The position of the arm at rest where the soft float function is turned on is called the reference position. After soft float is turned on, the soft float effect is determined by the parameter configuration item of float. The parameter configuration is done on the Oscillator UI.

The syntax format is as follows:

MAC SOFTFLOAT <n> [(TIMES=numerical value) | (DI_NUM=numerical value DI_VALUE=ON|OFF) | (R_NUM=numerical value R_VALUE=numerical value)], OUT

- SOFTFLOAT<n>: specifies which group of float parameters to use, the value of n ranges from 0 to 7, and the grouping expression specifies the judgement condition to stop the soft float.
- TIMES: specifies the time to float, and stops the soft float when the time is reached.
- DI_NUM and DI_VALUE: It specifies the state of this input IO is equal to DI_VALUE to stop soft floating.
- R_NUM and R_VALUE: Specifies that the soft float stops when the value of the R register is equal to R_VALUE.

The stop condition of soft float can be in various formats, such as time specified, IO state specified, R register value specified, etc., which can be written by the user according to the actual requirements.

Instruction Example:

```
Example1：
MAC SOFTFLOAT 0 ,OUT        'Without specifying the stopping condition of Soft Float, Soft Float will be kept floating after it is turned on, and Soft Float will not be stopped until the uninstallation process.
Example2：
MAC SOFTFLOAT 0 TIMES=2000, OUT      'Soft float is switched on and switched off after 2S of float, the
Example3：
MAC SOFTFLOAT 0 DI_NUM=1 DI_VALUE=ON, OUT     'Enable soft float and block, wait for DI[1]=ON to disable soft float
Example4：
MAC SOFTFLOAT0 R_NUM=9 R_VALUE=10.0, OUT      'Turn on soft float and block, wait for R[9] = 10.0 to turn off soft float
```

Procedure Example:

```
J P[0]        'The soft float can only be switched on after the robot arm has moved to the point and stopped.
MAC SOFTFLOAT 0 TIMES=2000, OUT    'Starts soft float with group 0
```

```
parameters and closes after 2 seconds of float time
    J P[1]        'Movement to point P1
```

Tips:

The softfloat function only supports models that have been adapted to the dynamics model. If the model does not support softfloat, a warning will be thrown when executing the MAC SOFTFLOAT command, and there will be no floating effect.

When soft float is enabled, the current tool, workpiece, and load level parameters must be used correctly.

In the following cases, the soft float function is automatically turned off: when the programme is paused, stopped or unloaded; when single-step debugging, arbitrary line debugging, backward; when the system alarms. The system throws an alert when the soft float function is switched off.

7.36 Introduction to welding instructions

7.36.1 Call the welding channel instruction

Instructions:

Set the welding channel data, which is generally used to call multiple sets of process parameters for a single weld.

Command example:

```
2  J P[1]
3  L P[2]
4  MAC ARC_ON R[500],OUT
5  L P[3]
6  MAC ARC_CHANNEL R[500],OUT
7  L P[4]
8  MAC ARC_OFF R[500],OUT
```

operating steps ：

1. Select the line above the command line to insert.

2. Select the instruction → welding instruction → call the welding channel instruction.

3. Select the channel number for welding data retrieval from the dropdown menu.

4. Click the OK button in the toolbar to add the welding channel command.

7.36.2 Arc start command

Instructions:

Set the welding data and start the arc.

Command example:

operating steps ：

1. Select the line above the command line to insert.

2. Select the instruction → welding instruction → arc initiation instruction.

3. Select the channel number for welding data in the dropdown menu.

4. Click the OK button in the toolbar to add the arc-start command.

Note: Arc-start and arc-stop commands must be used in pairs and placed before the arc-stop command.

7.36.3 Arc-Closing command

Instructions:

End welding.

Command example:

```
1
2  J P[1]
3  L P[2]
4  MAC ARC_ON R[500],OUT
5  L P[3]
6  MAC ARC_OFF R[500],OUT
```

operating steps ：

1. Select the line above the command line to insert.

2. Select the instruction → welding instruction → arc collection instruction.

3. Select the channel number with the closest matching arc command in the dropdown.

4. Click the OK button in the toolbar to add the arc closing command.

Note: The arc closing and arc opening instructions must be used in pairs and placed after the arc opening instruction.

7.36.4 Flight arc parameter setting command

Instructions:

The arc can be started in advance after the distance from the arc setting point, and the production rhythm can be improved by increasing the arc time.

Command example:

```
1
2  MAC FLIGHT DISTANCE=200,OUT
3  J P[1]
4  L P[2]
5  MAC ARC_ON R[500],OUT
6  L P[3]
7  MAC ARC_OFF R[500],OUT
```

operating steps ：

1. Select the line above the command line to insert.

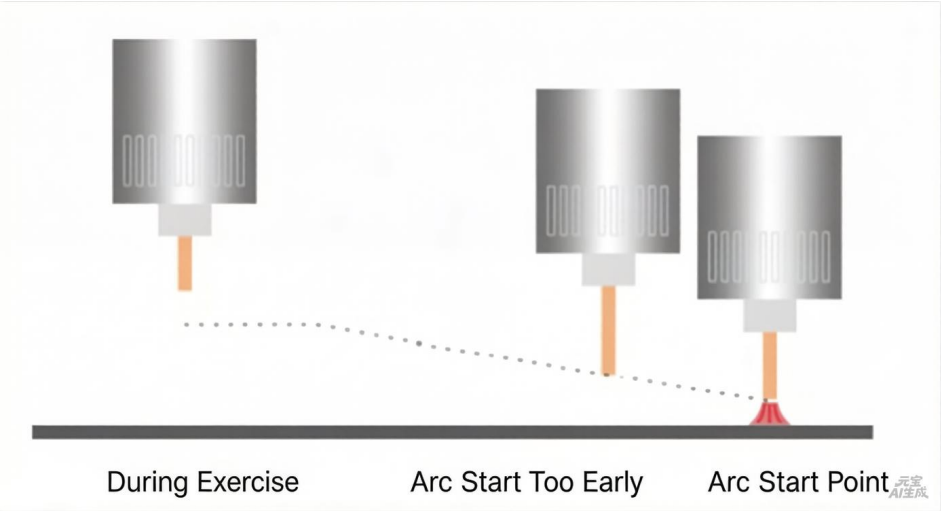
2. Select the instruction → welding instruction → flight arc initiation parameter setting instruction.

3. Enter the distance from the starting point to the arc before the arc begins in the input box.

4. Click the OK button in the toolbar to add the flight arc parameter settings.

5. After adding the flight arc parameter setting command, click Change Properties at the point requiring flight arc, check the Flight Arc checkbox, then click OK. The point will become the flight arc point.

Example of flight arc execution:

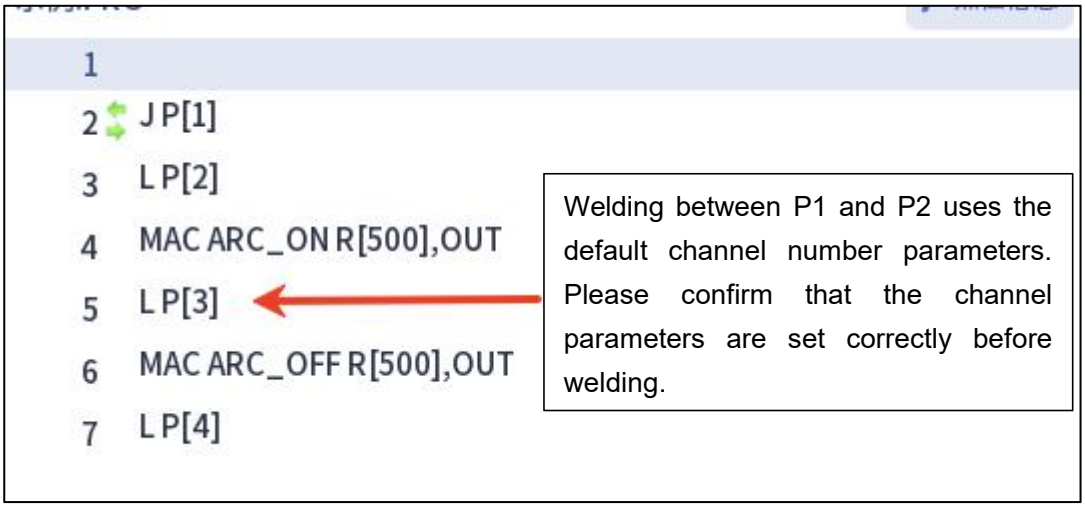


Taking a standard welding machine as an example: With a default slow wire feed speed of 50mm/s during arc initiation, the distance between the wire tip and workpiece after arc termination is approximately 10mm. Without the flying arc function, the wire requires 200ms to contact the workpiece for ignition welding during the arc initiation phase. Since the robot decelerates upon reaching the arc initiation point without a fixed speed, the actual distance traveled during the 200ms run before reaching the arc point cannot be precisely calculated. Therefore, adjustments must be made based on actual usage. Typically, the test distance is preset as the distance from the electrode tip to the workpiece. As shown in the diagram, the flying arc parameter is first set to 10mm in the configuration command, followed by welding, with subsequent adjustments made according to actual production conditions.

Note: Since the robot initiates arc movement before reaching the actual starting point, avoid excessive wire feeding during arc initiation to prevent arc explosion or stretching that may disrupt production timing.

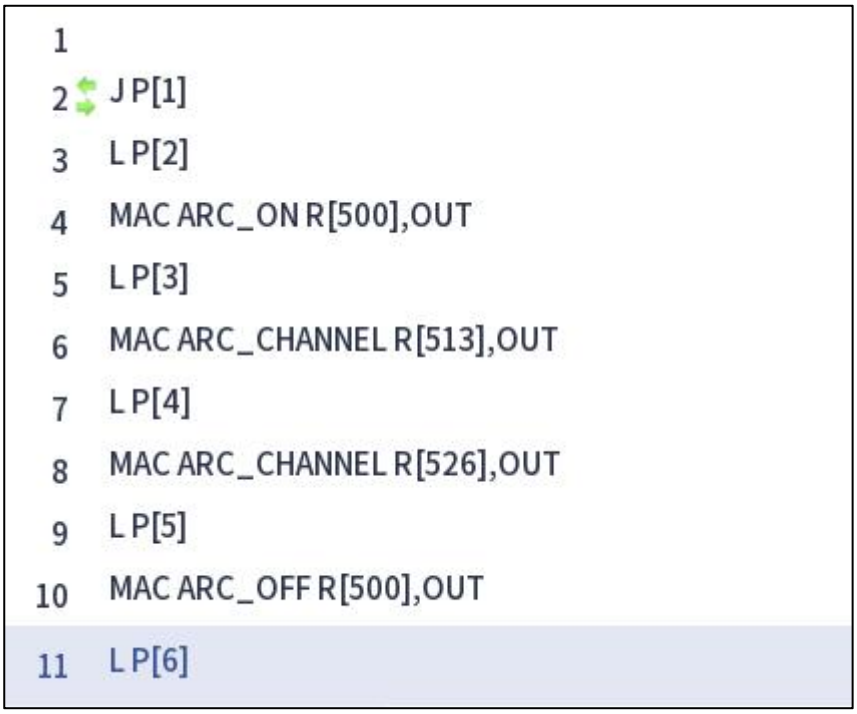
7.36.5 Example welding instruction program

Usage of Welding Instructions with Only One Set of Process Parameters for a Single Weld



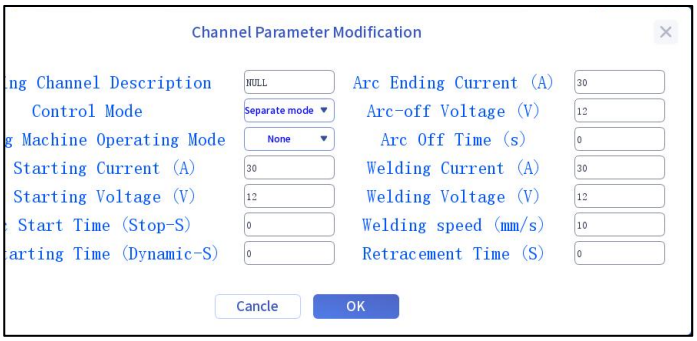
Note: The arc-start command will use the welding channel parameters. Check that the parameters are set correctly before welding.

Usage of Welding Instructions for Multiple Process Parameters in a Single Weld



You can modify the channel call and arc start commands after insertion, as shown in the figure.

After selecting the command line and clicking the Modify button below, a Modify button will appear next to it. Click Modify on the command line to open the dialog box shown in the figure below.

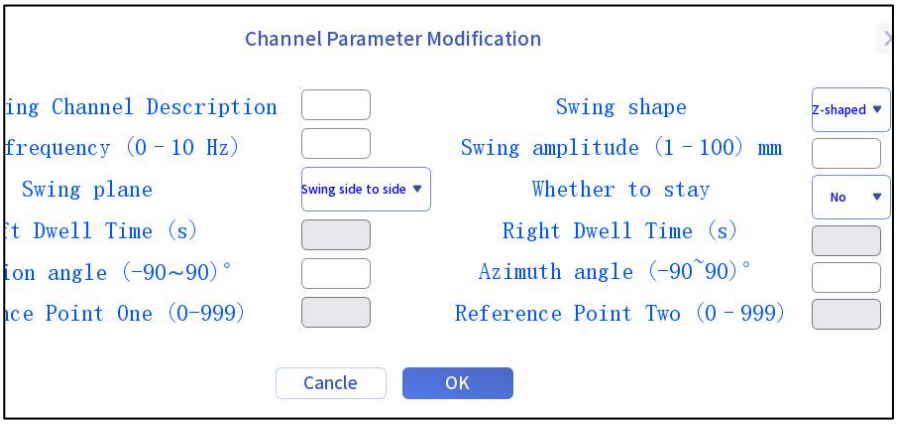


As shown in the figure below, select the channel number to modify. The process parameters will display modification boxes for current, voltage, and welding speed. After setting these parameters, the welding voltage, current, and speed will be updated to the current settings. If no channel is selected, the voltage, current, and welding speed will remain unchanged. Click OK to confirm the changes, which will update the channel number and parameters. Additionally, the arc initiation command now includes an arc detection function for fish-scale welding.

7.36.6 Swing command

The swing function enables the robotic arm to execute linear or circular motions while maintaining a predefined direction, with the final end-effectors TCP (Target Control Point) achieving a specified trajectory. This motion is widely used in welding and grinding applications, featuring common patterns such as sinusoidal Z-swing, circular swing, crescent swing, figure-eight swing, and L-shaped swing.

7.36.6.1 Explanation of parameter settings for swing welding Channel



Parameter meanings are as follows:

Packing channel description: channel notes.

Swing shape: Choose from Z-shaped, round, L-shaped, crescent, inverted crescent, or figure-eight.

Amplitude of oscillation: the distance from the wave crest to the central axis, measured in millimeters (mm).

Swing frequency: the number of swing paths per second, measured in Hertz (Hz).

Swing plane: The plane containing the swing motion, representing its direction. First, establish the first plane using the line connecting the starting and target points as the X-axis and the tool coordinate systems Z-axis. Then, calculate the Y-axis of the swing plane by cross-multiplying the first planes normal vector with the X-axis, followed by the Z-axis through cross-multiplication with the X and Y axes. The XYZ axes form the swing coordinate system. 0: along the Y-axis of the swing coordinate system; 1: along the Z-axis; 2: along the X-axis.

Stay: Yes or No.

Left dwell time: The dwell time of the left endpoint. Measured in seconds. If not an integer multiple of the interpolation cycle, it is rounded to the nearest 4ms. Rounding follows the standard rounding rule.

Right dwell time: The dwell time of the right endpoint. Measured in seconds. If not an integer multiple of the interpolation cycle, it is rounded to the nearest 4ms. Rounding follows the standard rounding rule.

Elevation: The angle of rotation around the X-axis of the swing coordinate system, measured in degrees, with a range of-90 to +90 degrees. A positive value indicates a counterclockwise rotation, while a negative value indicates a clockwise rotation.

Azimuth: The tilt angle of the swing path on the specified plane. A positive value tilts the left endpoint toward the X+ axis, while a negative value tilts the right endpoint toward the X+ axis. The unit is degrees, with a range of ±90 degrees. When ORI equals ±90 degrees, the swing direction is parallel to the X-axis.

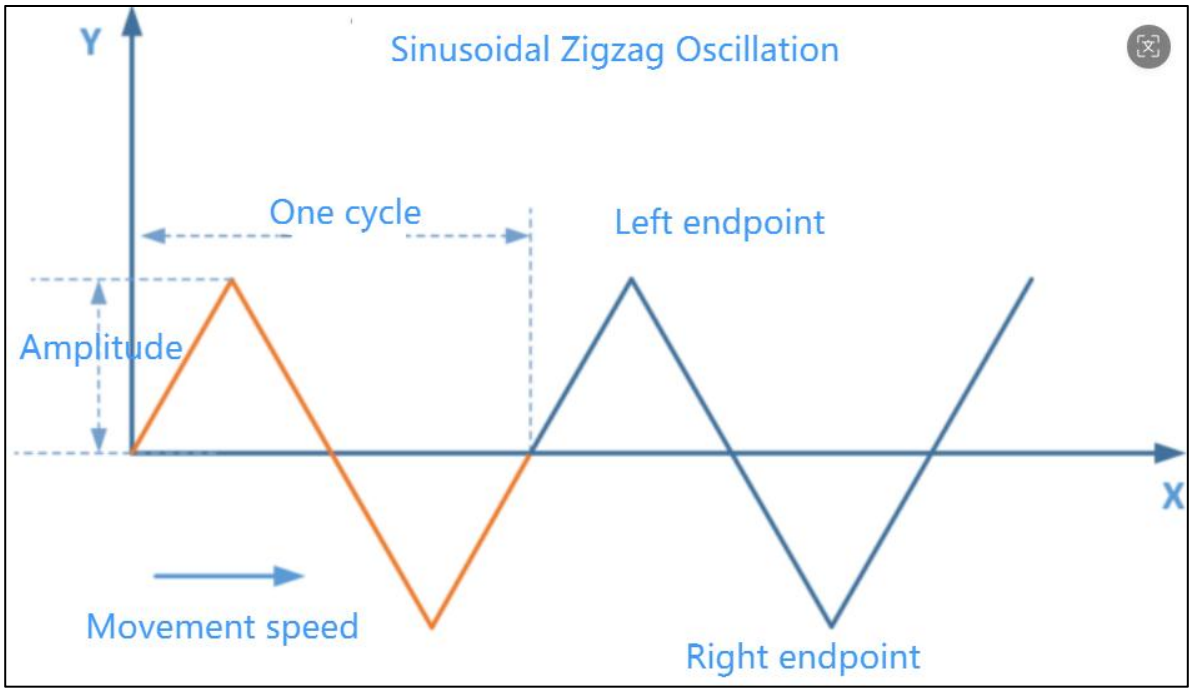
Reference point 1: The index value of the LR register, ranging from 0 to 999. It is the point on the first face of the L-shaped swing and is stored in the LR register.

Reference point 2: The index value of the LR register, ranging from 0 to 999. It is the point on the second surface during L-type oscillation, stored in the LR register.

7.36.6.2 Description of swing welding types

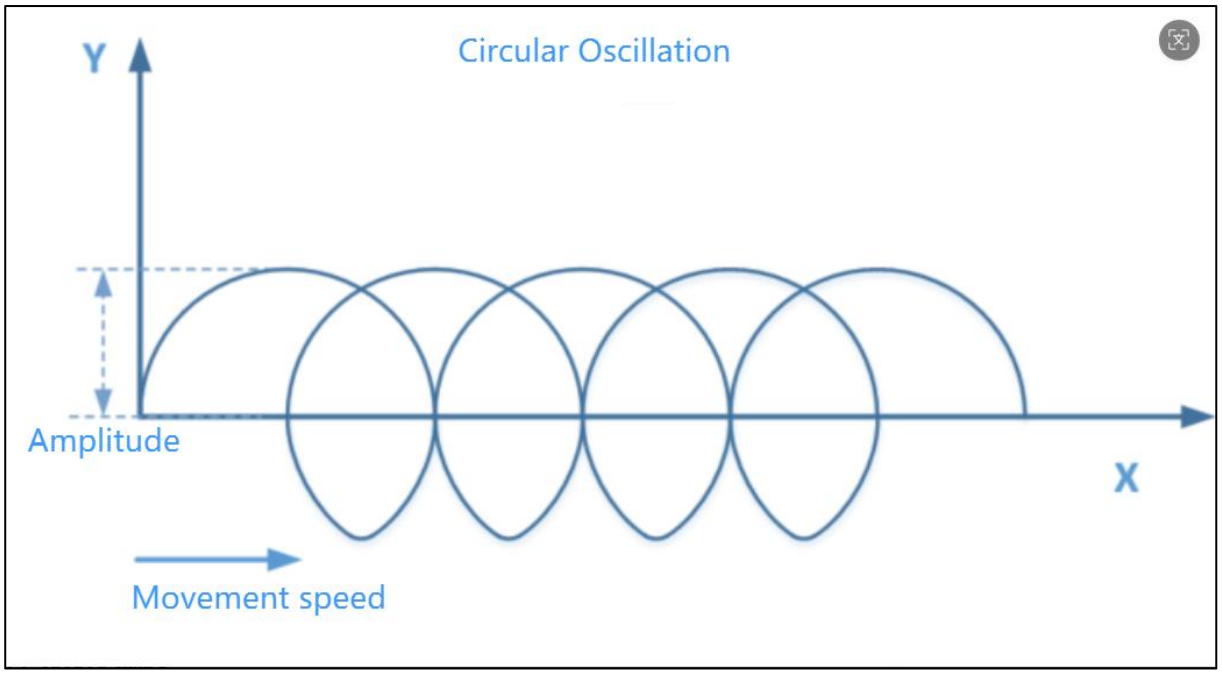
Z-shaped swing

The sinusoidal Z-axis oscillation path is shown below. The robotic arm can perform vertical oscillations on the specified plane when executing linear or circular trajectories.



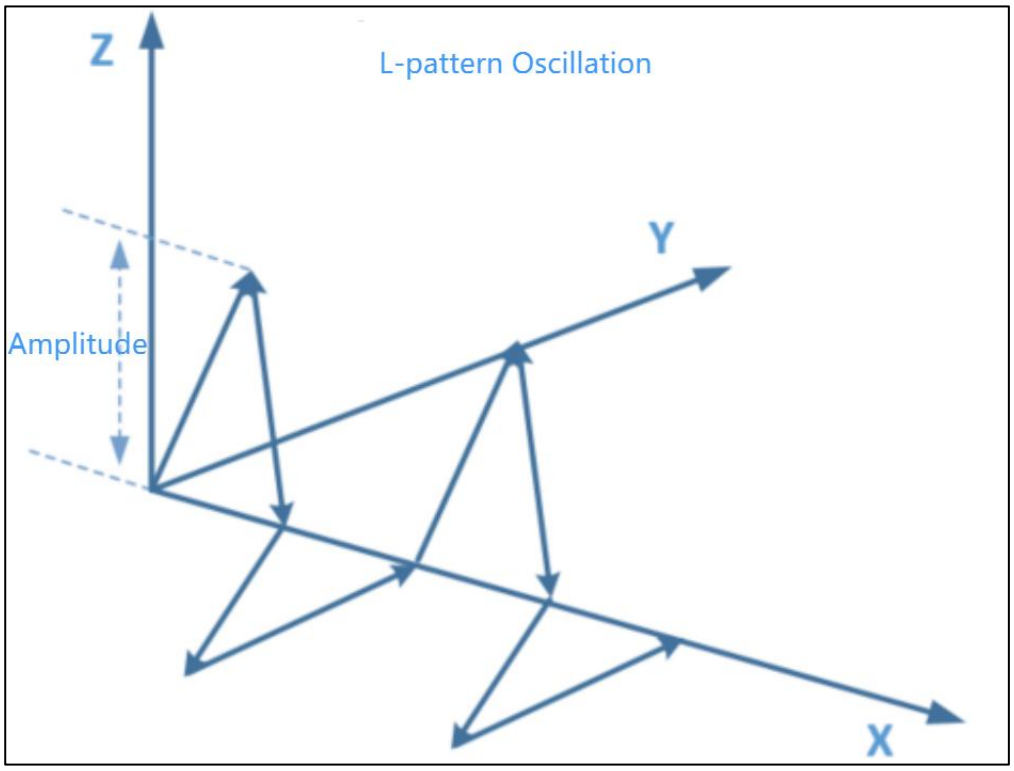
Circular Oscillation

The circular swing trajectory is as follows: when the robotic arm performs linear or circular arcs, it can draw circles while moving forward on the designated swing plane.



L-shaped swing

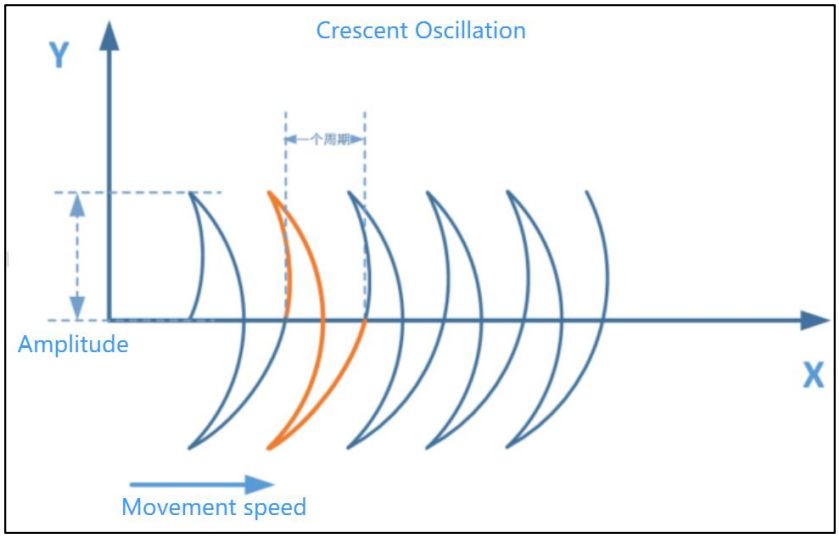
An angle exists between the two plates, and the manipulator alternates swinging motions on both plate surfaces while performing the main motion. The L-shaped swinging trajectory is as follows.



Note: When using an L-shaped swing, in addition to teaching the straight path, you must also teach "Reference Point 1" and "Reference Point 2" on both sheet metal surfaces. In the index of the LR register specified by the swing welding channel number parameter, these two points are used to determine the sheet metal angle with the weld path. The L-shaped swing cannot be used for arc paths.

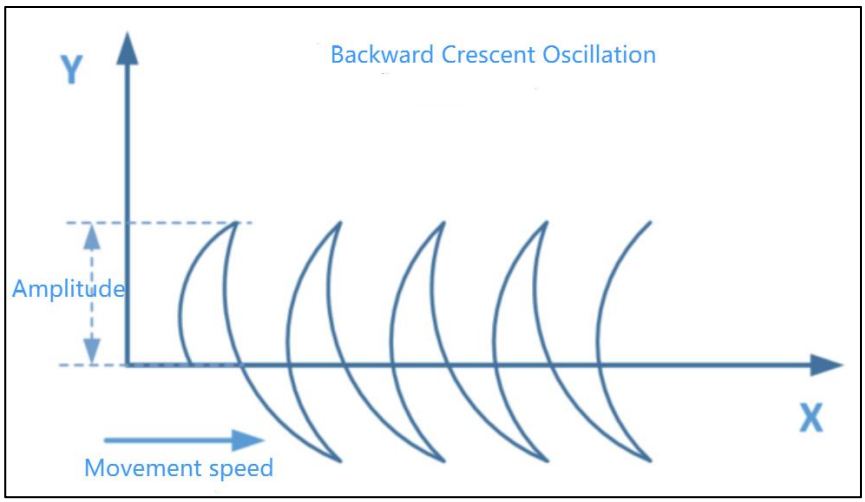
Crescent Moon Swing

The trajectory of the crescent moons swing is as follows:



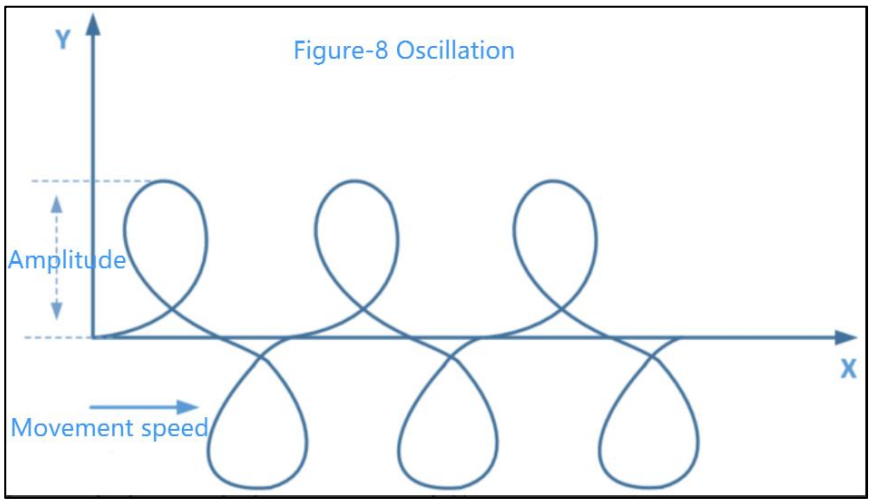
Counter-Crescent Swing

The anti-crescent swing trajectory is as follows:



character swing

The 8-character oscillation path is as follows: When the robotic arm executes linear or arc commands, it can perform a motion that traces an 8-character pattern while moving forward on the designated oscillation plane. As shown in the figure below:



Example of Swivel Welding Channel Command

Instructions:

Set the flip-flop welding parameters for the flip-flop welding program.

Example command: Joint P[1]

Line P[2] Arc Point

Swing Start (1) Swing Start command

Arc Start (1)

Line P[3]

Line P[4]

Arc P[5]

Arc P[6]

Arc P[7]

Arc P[8] Fox Point

Arc Close (1)

End of swing (1) End of swing command

Joint P [9]

operating steps ：

1.Open the welding process package → process parameters → flip welding channel, select channel 1, click the "Modify" button, and set the required flip welding parameters.

2.Create a new program, click the "Open" button, then click the "Edit Program" button.

3.Go to the designated position and select the Insert Joint, Straight, or Arc command.

4.Select the instruction → swing welding instruction → call the swing start instruction.

5.Select the swaging channel number in the drop-down list.

6.Click the OK button in the toolbar to add the call-to-solder channel command.

Example program:

```
J P[1]
L P[2]
MAC WEAVESTART ARCINGWELDCHANNELINDEX=1,OUT
MAC ARC_ON R[500],OUT
L P[3]
L P[4]
C P[5] P[6]
C P[6] P[7]
C P[7] P[8]
C P[8] P[9]
MAC ARC_OFF R[500],OUT
MAC WEAVEEND ARCINGWELDCHANNELINDEX=1,OUT
J P[9]
```

7.36.7 Production order

7.36.7.1 Production statistics command

Instructions:

The instruction is a statistical output instruction, which counts the output of welded workpieces.

> Joint P[1]
>
> Line P[2] normally edited program
>
> Production Statistics (1) Production Statistics Command

Command example:

operating steps ：

1. Select the line above the command line to insert.

2. Select the command → Output command → Output statistics command.

3. Select the production statistics channel number from the dropdown menu.

4. Click the OK button in the toolbar to add the output statistics command.

Note: After executing the yield statistics command, you can view the results in Process Package → Yield Statistics.

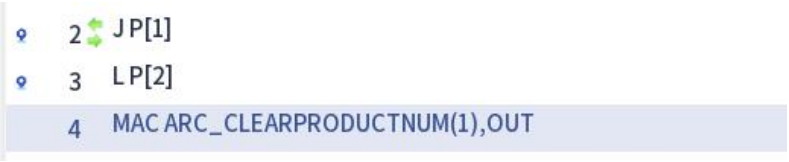
7.36.7.2 Production reset command

Instructions:

This instruction resets the output of the corresponding channel.

> Joint P[1]
>
> Line P[2] normally edited program
>
> Zero production (1) Clear the production of channel 1

Command example:



operating steps ：

1. Select the line above the command line to insert.

2. Select the command → Output command → Reset output command.

3. Select the channel number to reset from the dropdown list.

4. Click the OK button in the toolbar to add the production reset command.

7.36.8 Fish scale pattern command

7.36.8.1 Straight fish scale welding instruction

Instructions:

This instruction is a motion command, and the welding effect after execution is fish-scale.

> Joint P[1] Normal editing program
>
> Arc Start (1) Arc start command. A set of welding channels is called by default before arc start.
>
> Straight line fish-scale welding P[2] Translation speed=8mm/s Time=100ms Interval distance=1mm
>
> Arc closure (1)

Command example:

```
2 J P[1]
3 MAC ARC_ON R[500],OUT
4 MAC INTERMITTENT_WELD P[1] VEL=8 TIME_FLAG=0 SPACE_TIME
  _DISTANCE=100 IDLING_DISTANCE=1,OUT
5 MAC ARC_OFF R[500],OUT
```

operating steps ：

1. Select the line above the command line to insert.

2. Select the instruction → Fish-scale welding instruction → Straight-line fish-scale welding instruction.

3. You can input the translation speed, time, and interval distance in the input box.

4. After completing the input, click the OK button in the toolbar to add the straight-line fish-scale welding instruction.

7.36.8.2 Arc fish scale welding point recording command

Instructions:

The instruction is for recording the arc position of the arc fish-scale welding.

Joint P[1] ‘ transition point

Arc Start (1) Arc start command, calls a set of welding channel data

Record the arc fish-scale weld point position P[2] Record the arc fish-scale weld point position instruction

Record the arc fish-scale weld point position P[3] Record the arc fish-scale weld point position instruction

Record the arc fish-scale weld point position P[4] Record the arc fish-scale weld point position instruction

Execute the arc fish-scale welding motion. Execute the arc fish-scale welding motion command.

Arc Close (1) Arc Close Command

Command example:

```
2  J P[1]
3  MAC ARC_ON R[500],OUT
4  MAC WM P[2],OUT
5  MAC WM P[3],OUT
6  MAC WM P[4],OUT
7  MAC MOVE_WM VEL=8 TIME_FLAG=0 SPACE_TIME_DISTANCE=100
   IDLING_DISTANCE=1,OUT
8  MAC ARC_OFF R[500],OUT
```

operating steps ：

1. Select the line above the command line to insert.

2. Select the instruction → Fish-scale welding instruction → Record the position of the arc fish-scale welding point.

3. Click the OK button in the toolbar to add the arc fish-scale welding point recording command.

Note: Insert at least three arc fish-scale weld point recording commands together to plan the arc path.

7.36.8.3 Arc fish scale welding motion command

Instructions:

The instruction is the arc fish scale welding motion instruction, which is usually placed under the arc fish scale welding point recording instruction.

Joint P[1] ‘ transition point

Arc Start (1) Arc start command, calls a set of welding channel data

Record the arc fish-scale weld point position P[2] Record the arc fish-scale weld point position instruction

Record the arc fish-scale weld point position P[3] Record the arc fish-scale weld point position instruction

Record the arc fish-scale weld point position P[4] Record the arc fish-scale weld point position instruction

Execute the arc fish-scale welding motion.

Arc Close (1) Arc Close Command

Command example:

```
2  J P[1]
3  MAC ARC_ON R[500],OUT
4  MAC WM P[2],OUT
5  MAC WM P[3],OUT
6  MAC WM P[4],OUT
7  MAC MOVE_WM VEL=8 TIME_FLAG=0 SPACE_TIME_DISTANCE=100
   IDLING_DISTANCE=1,OUT
8  MAC ARC_OFF R[500],OUT
```

operating steps ：

1. Select the line above the command line to insert.

2. Select the instruction → Fish Scale Welding Instruction → Circular Fish Scale Welding Motion Instruction.

3. The input box allows you to enter the translation speed, time, and interval distance of the movement.

4. Click the OK button in the toolbar to add the arc fish-scale welding motion command.

7.36.8.4 Example program of fish scale pattern instruction

Use of Straight Line Fish Scale Welding Instructions

Joint P [1] Normal editing program

Arc Start (1) Arc start command. A set of welding channels is called by default before arc start.

Fish-scale welding P[2] translation speed=8mm/s time=100ms interval distance=1mm

Fish-scale welding P[3] translation speed=8mm/s time=150ms interval distance=1mm

Fish-scale welding P[4] translation speed=8mm/s time=80ms interval distance=1mm

Fish-scale welding P[5] translation speed=8mm/s time=50ms interval distance=1mm

Arc closure (1)

Joint P [6]

```
2  J P[1]
3  MAC ARC_ON R[500],OUT
4  MAC INTERMITTENT_WELD P[2] VEL=8 TIME_FLAG=0 SPACE_TIME
   _DISTANCE=100 IDLING_DISTANCE=1,OUT
5  MAC INTERMITTENT_WELD P[3] VEL=8 TIME_FLAG=0 SPACE_TIME
   _DISTANCE=150 IDLING_DISTANCE=1,OUT
6  MAC INTERMITTENT_WELD P[4] VEL=8 TIME_FLAG=0 SPACE_TIME
   _DISTANCE=80 IDLING_DISTANCE=1,OUT
7  MAC INTERMITTENT_WELD P[5] VEL=8 TIME_FLAG=0 SPACE_TIME
   _DISTANCE=50 IDLING_DISTANCE=1,OUT
8  MAC ARC_OFF R[500],OUT
9  J P[6]
```

Application of Arc Scaled Welding Instructions

Joint P [1] ‘ transition point

Arc Start (1) Arc Start command, calls a set of welding channel data

Record the arc fish-scale weld point position P[2] Record the arc fish-scale weld point position instruction

Record the arc fish-scale weld point position P[3] Record the arc fish-scale weld point position instruction

Record the arc fish-scale weld point position P[4] Record the arc fish-scale weld point position instruction

Execute the arc fish-scale welding motion.

Arc Close (1) Arc Close Command

Joint P[5] transition point

```
2  J P[1]
3  MAC ARC_ON R[500],OUT
4  MAC WM P[2],OUT
5  MAC WM P[3],OUT
6  MAC WM P[4],OUT
7  MAC MOVE_WM VEL=8 TIME_FLAG=0 SPACE_TIME_DISTANCE=100
   IDLING_DISTANCE=1,OUT
8  MAC ARC_OFF R[500],OUT
9  J P[5]
```

7.36.9 Laser positioning command

The laser positioning function in robotic welding utilizes a laser to scan the workpiece, capturing data on its position and shape. The robot controller then processes and analyzes this information to determine the optimal welding path and parameters. This feature enables precise positioning and path planning during automated welding, thereby enhancing both welding quality and efficiency.

pay attention to ：

A. The welding surface must be kept smooth, free from dirt and oxides to prevent laser reflection interference, ensuring precise tracking.

B. The laser must remain stable, free from vibrations or interference, to ensure tracking accuracy.

7.36.9.1 Laser tracking parameter adjustment

To configure the laser tracking path, navigate through the following menu sequence: Main Menu → Plugin Package → Welding Process Package → Laser Positioning. This will open the Laser Positioning Parameter Setup interface (as shown below), which includes three sections: Communication Settings, Search Parameter Settings (containing parameters for laser positioning), and Tracking Parameter Settings.

communications setting ：

Laser tracking brand: Select the brand of the laser.

Controller IP address: The controllers IP address is automatically read and does not require configuration.

Laser IP address: Set the lasers IP address.

Port number: Set the port number for laser communication.

Connection timeout: Set the connection timeout time between the robot and the laser.

Delay time: Set the delay time.

Read/Write timeout: Set the read/write timeout time.

Communication status: Displays the lasers connection status. Green indicates a successful connection, while red indicates no connection.

Search parameters:

Search count: Set the number of bit searches.

Search speed: Set the search speed.

Search distance: Set the search range.

Search starting point precision: Set the precision for the starting point of the search.

Tracking parameters:

Weld end point accuracy: Set the weld end point accuracy.

X-axis tracking accuracy: Set the X-axis tracking accuracy.

Y-axis tracking precision: Set the Y-axis tracking precision.

Z-axis tracking accuracy: Set the Z-axis tracking accuracy.

Communication connection: After completing the input communication settings, click the "Communication Connection" button to establish communication with the laser. Check the communication status to confirm the connection.

Laser on: After establishing a communication connection, click to turn the laser on or off. You can also use the auxiliary configuration buttons to control the laser.

Save parameters: After entering search and tracking parameters, click "Save parameters" to save the input parameters.

7.36.9.2 Laser positioning command

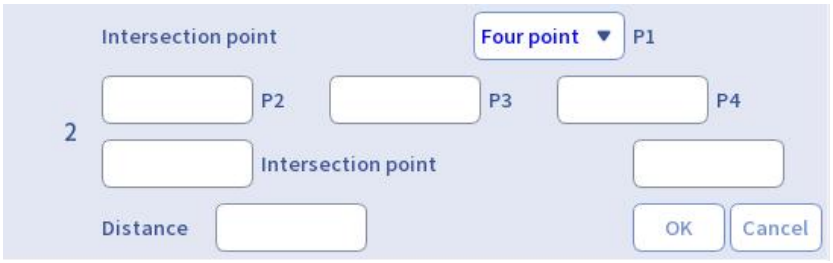
Launch the program and initiate editing. In the laser positioning command options, insert the laser positioning command by selecting the weld type and channel number for laser positioning. Enter the required LR register number for point recording (the corresponding register number must be pre-recorded with the posture). Activate the laser positioning flag switch corresponding to the laser channel number, which can be configured in the laser positioning setup interface of the process package. The following figure shows a template application for single-point laser positioning, including the steps to turn on the laser, perform laser positioning, and turn off the laser.

7.36.9.3 Intersection calculation

The intersection point calculation instruction is inserted after the point data is scanned. The basic principle is to determine a straight line through two points, and then calculate the intersection point position through two straight lines (four points) or three straight lines (six points). Generally, the intersection point position is used as the end position of the weld seam.

As illustrated below: Launch the program → Initiate editing → Welding instruction → Laser positioning instruction. Select the intersection calculation command, which can be configured for either four-point or six-point intersection calculation. Taking the four-point configuration as an example: Points 1, 2, 3, and 4 must be paired with corresponding register numbers 1 and 2 respectively. Points 3 and 4 form a straight line.

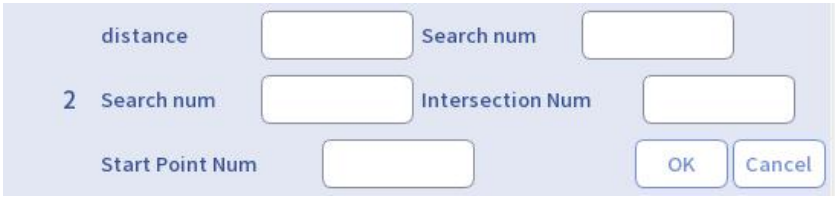
Points 1 and 2 are designated as scanning points on the weld seam. The intersection box requires two entries: the register number for storing the calculated result, and the offset value for the intersection position. The offset direction is determined by the register position of point 2 relative to the intersection point.



7.36.9.4 Starting point calculation

The starting point calculation instruction confirms the welds initial position after completing intersection calculations. The basic principle involves determining the weld direction through two points, then calculating the starting position based on the input distance.

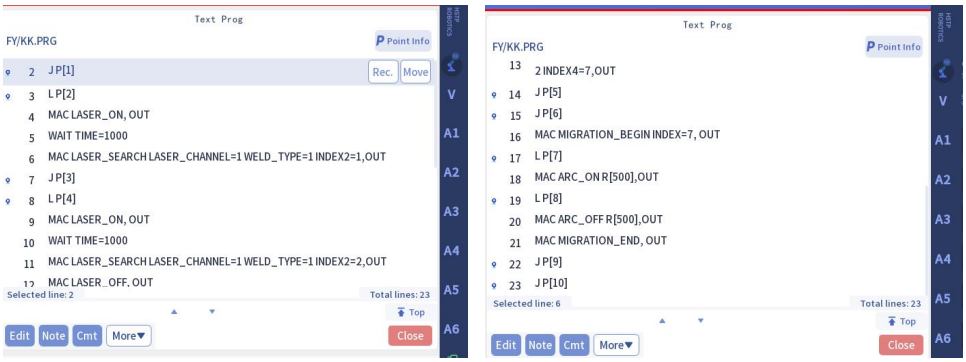
As shown in the figure below: The actual weld length is entered through the distance selection box. The weld direction is determined by the registers corresponding to positioning point number 1 and positioning point number 2. The welding method is determined by the transition from number 1 to number 2. The intersection point number is stored in the intersection register of the intersection calculation instruction, while the starting point number is recorded in the LR register to track the initial position.



7.36.9.5 Calculation of offset

The laser offset command achieves the same functionality as contact-based positioning by invoking the position channel number and selecting an offset type. As illustrated, offset types include 1D,2D,3D+, and arc, with the primary differences lying in the number of scanned points: 1D scans one point, 2D scans two points, while 3D+ and arc scan three points. Position points are sequentially filled into the register numbers corresponding to each positioning command based on the scanning sequence. The offset number must be customized (recommended range: 5-20) and is then passed to the offset command in subsequent contact-based positioning operations.

7.36.9.6 Example of laser offset positioning program



As illustrated in Figure 2D, the laser positioning system comprises three functional modules. The scanning module employs a linear laser sensor to detect the workpieces fixed position, with recorded values stored in corresponding registers. The system determines the required number of scanning points based on the weld seam offset type. The second module performs seam offset calculation by applying offset commands to process the register values, generating precise offset data for the weld seam trajectory. The final welding module inserts the target seam trajectory between the start and end offsets, enabling comprehensive position adjustment for welding operations.

7.36.10 Laser tracking command

Laser tracking in welding robots is a technology that utilizes laser measurement to track welding positions and shapes. By scanning the workpiece with a laser, it captures positional and dimensional data to enable precise welding path tracking and correction. The figure below illustrates a typical laser tracking scenario. Key points to note: 1) Positions P0 to P4 are scanned continuously at preset translation speeds to locate the welding arc initiation point, with P4 serving as the starting point for arc detection and requiring initial trial recording; 2) The movement from P4 to P5 constitutes the actual welding trajectory.

```
2  J P[1]
3  J P[2]
4  MAC LASER_ON, OUT
5  MAC LASER_SEARCH_TRACK P[3] WELD_TYPE=1 VEL=8, OUT
6  MAC ARC_ON R[500],OUT
7  MAC LASER_START_TRACK, OUT
8  L P[4]
9  MAC ARC_OFF R[500],OUT
10 MAC LASER_STOP_TRACK, OUT
11 MAC LASER_OFF, OUT
12 J P[5]
```

7.36.10.1 Laser tracking precautions

A. The welding surface must be kept smooth, free from dirt and oxides to prevent laser reflection interference, ensuring precise tracking.

B. The laser must remain stable, free from vibrations or interference, to ensure tracking accuracy.

C. The current version of laser tracking is only compatible with straight weld tracking without a positioner, and other applications are not supported.

D. The search speed and welding speed should be set to the same.

E. The laser tracking function will only work after configuring the external mode.

F.The laser tracking program needs to record the coordinates of the workpiece.

7.36.10.2 Laser tracking preparation

A. Material Preparation

(1) One laptop

(2) One network cable

(3) Laser Positioning Sensor Software Installation Package

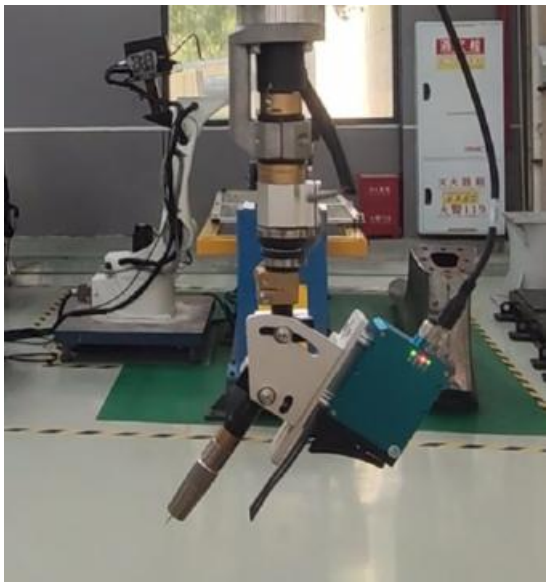
(4) Huashu Robotics

(5) Laser Positioning Sensor Software Tracking Device

(See the laser user manual for installation instructions)

B. Install the laser.

The laser must be mounted on the robot welding gun and positioned in front of it (as shown in the figure below).

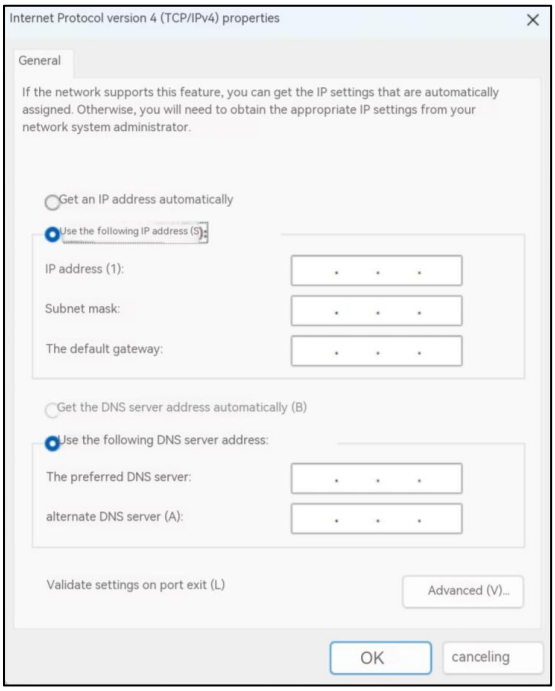


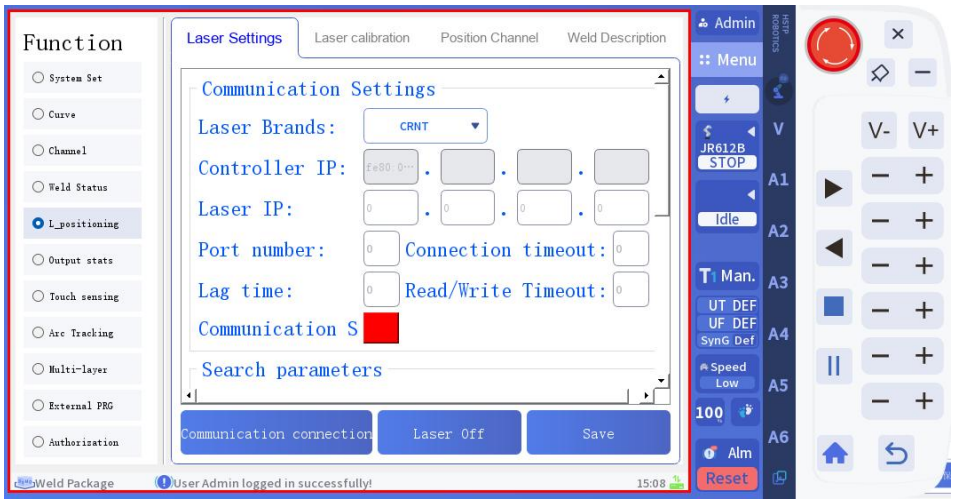
C. Setting up computers, robots, and laser communication

(1) Match the computer IP address with the robot controller

(2) Configure the welding process package at the teaching device end-laser positioning-laser setup, to match with the robot control

After setup, click the teaching device to establish communication. A green status indicates a successful connection. Restart the robot to complete initialization. The specific values are shown in the figure below:



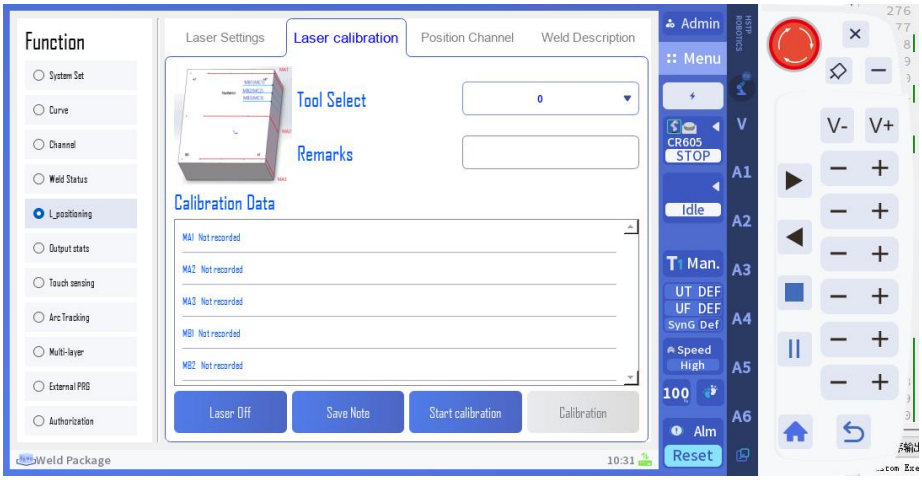


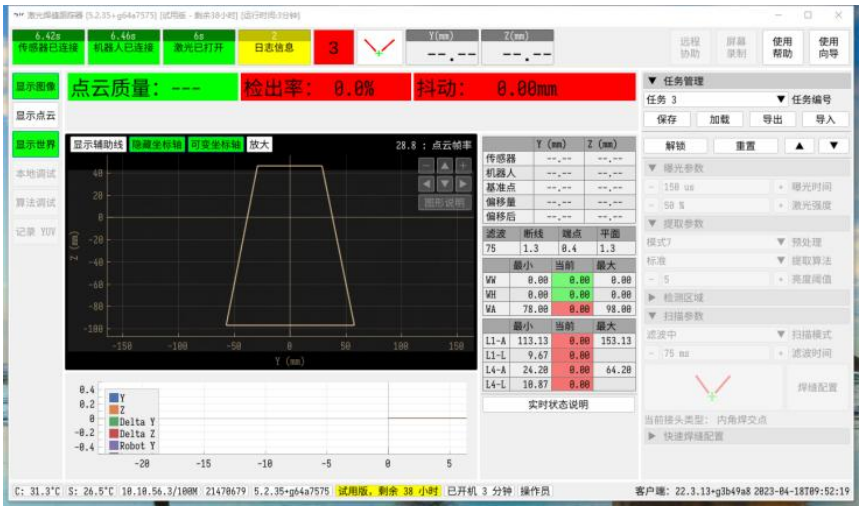
7.36.10.3 Laser calibration

1) Access the calibration interface

(1) The teaching device enters the welding process package-laser positioning-laser calibration-start calibration (select the current tool coordinate number in the tool coordinate system)

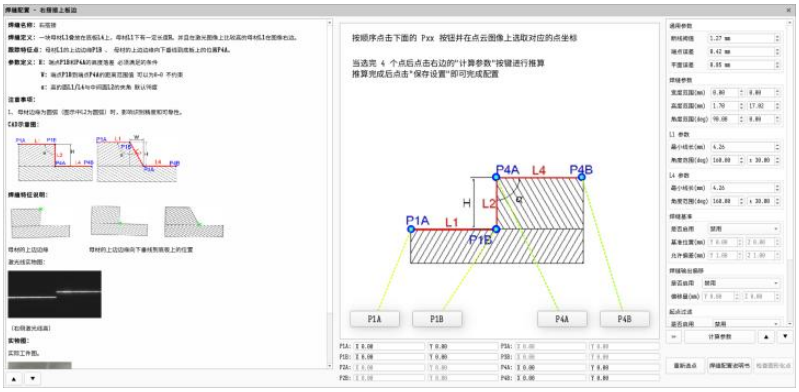
(2) Launch the FULL-V full-view laser positioning sensor software on the computer, connect the sensor to access the laser positioning sensor interface. When the sensor, robot, and laser display are all connected and operational, the hardware connection is confirmed to be normal.





2) Configuring Scan Welds and Preparing Calibration Plates

(1) Open the laser software on your computer. Click the unlock button in the right panel to access the internal parameter settings for the current task. Select the right-side overlap top plate edge as the weld type in the weld configuration section, then save and maintain the settings for the task number.

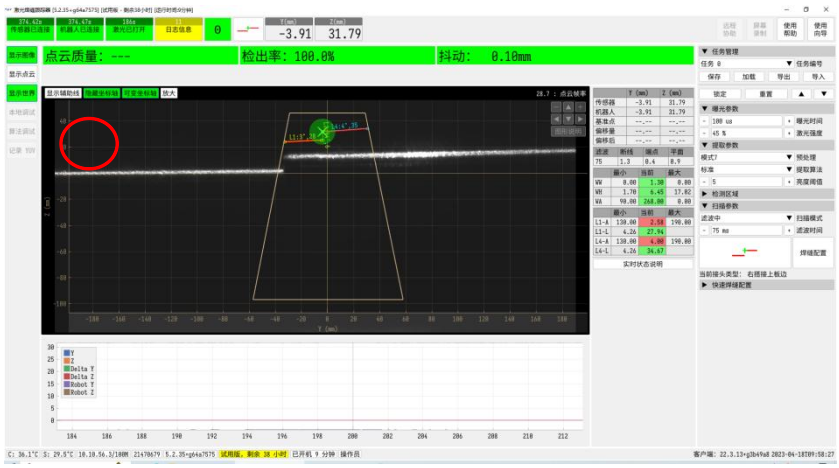


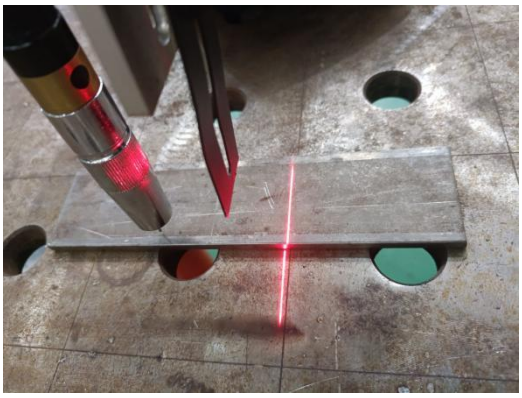
(2) Select a carbon steel plate with a thickness exceeding 3mm for the calibration board. The standard board used on-site measures 200mm×50mm×5mm. After selecting the appropriate carbon steel plate, mark a straight line on it to align with the laser line for calibration (as indicated by the red line in the figure below).

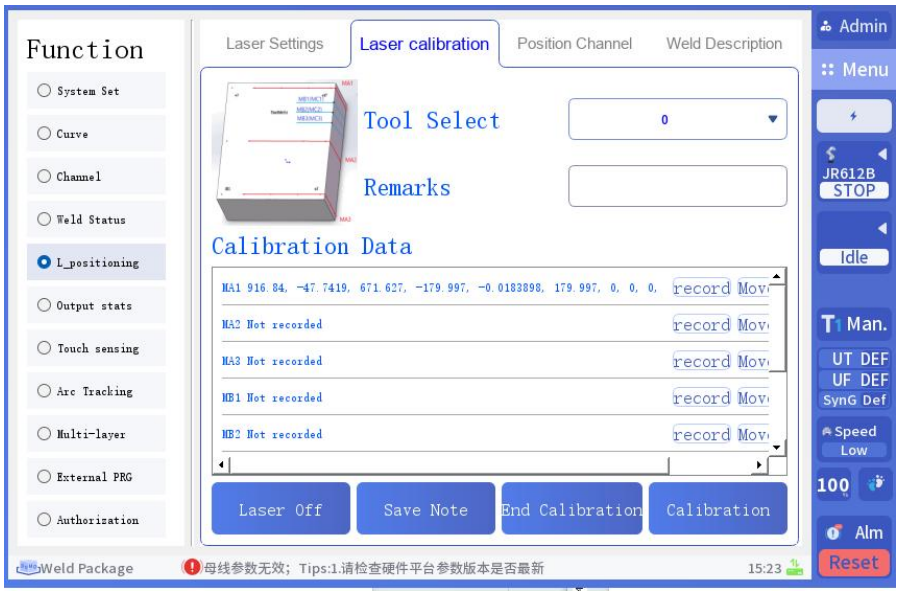
3) Begin calibration

Record the MA1 location

As shown in the figure below: The MA1 position requires moving the laser positioning sensors scanned weld seam to the upper left edge (marked in red), while the actual laser line of the welding gun must align with the straight line marked on the carbon steel plate, recording the current position.

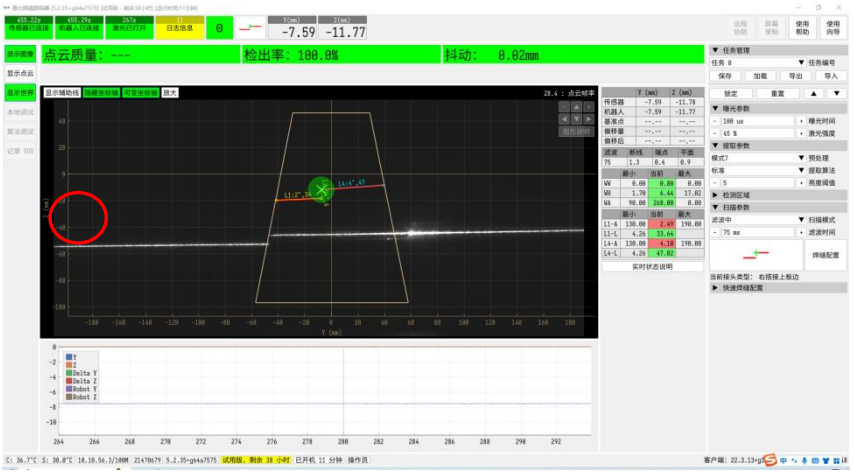


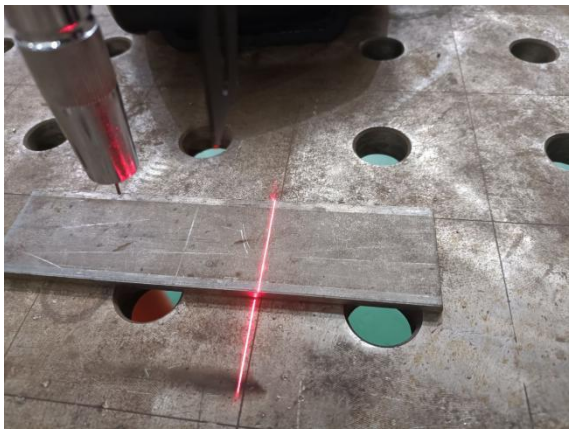


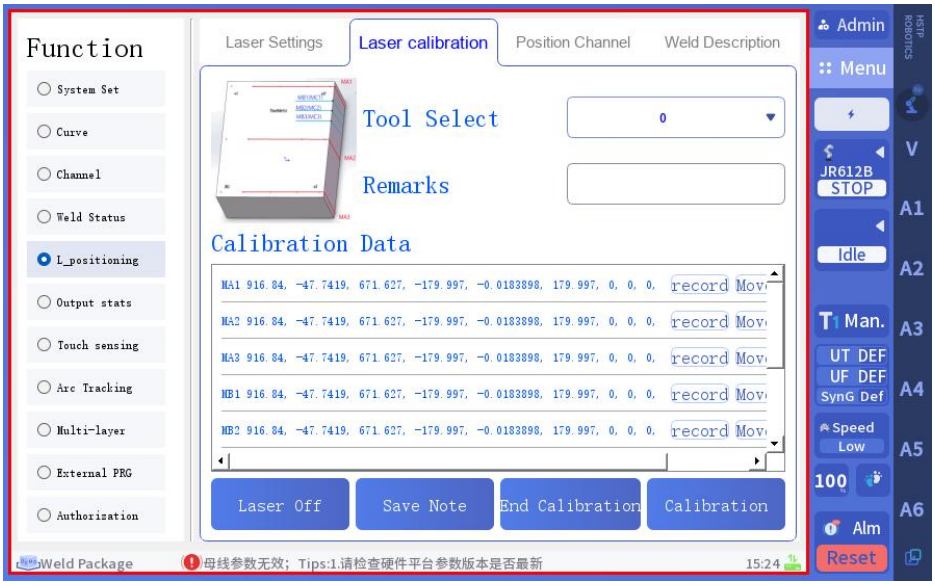


4) Record the MA2 location

As shown in the figure below: The MA2 position requires moving the laser positioning sensors scanned weld seam to the lower left edge (marked in red), while the actual laser line of the welding gun must align with the straight line marked on the carbon steel plate, recording the current position.

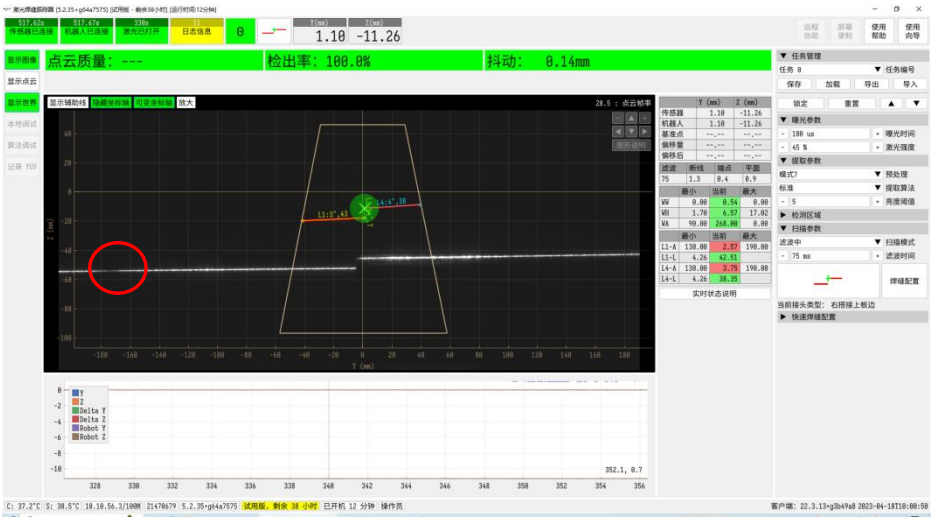


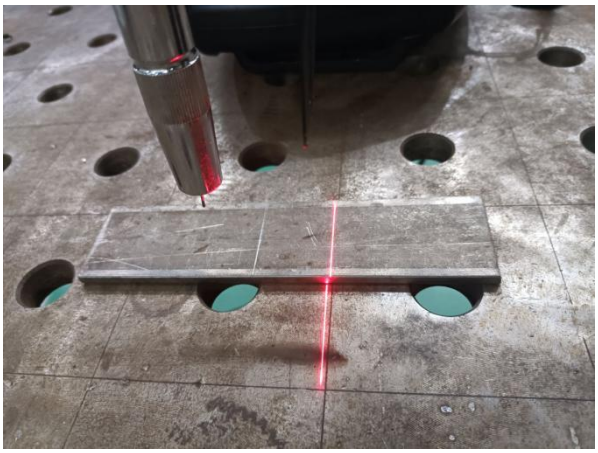


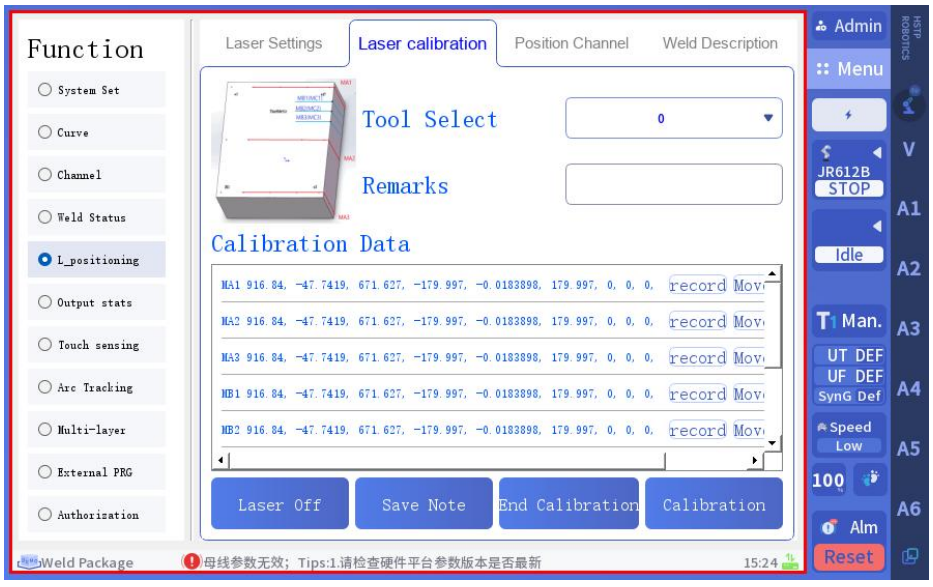


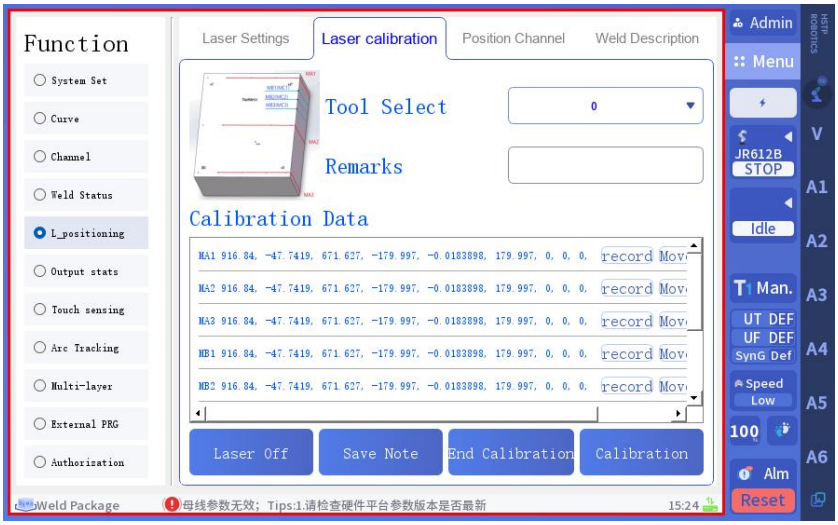
5) Record the MA3 location

As shown in the figure below: The MA3 position requires moving the laser positioning sensor to scan the weld seam to the lower-middle edge (marked in red), and the laser line of the actual welding gun must align with the straight line marked on the carbon steel plate, recording the current position.



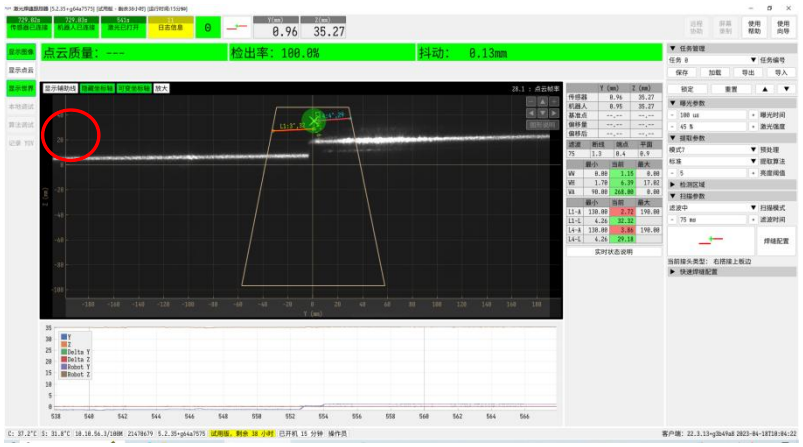


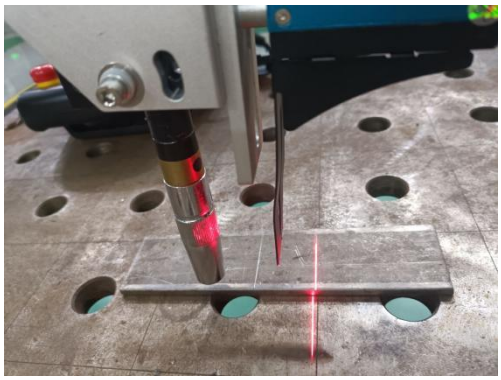


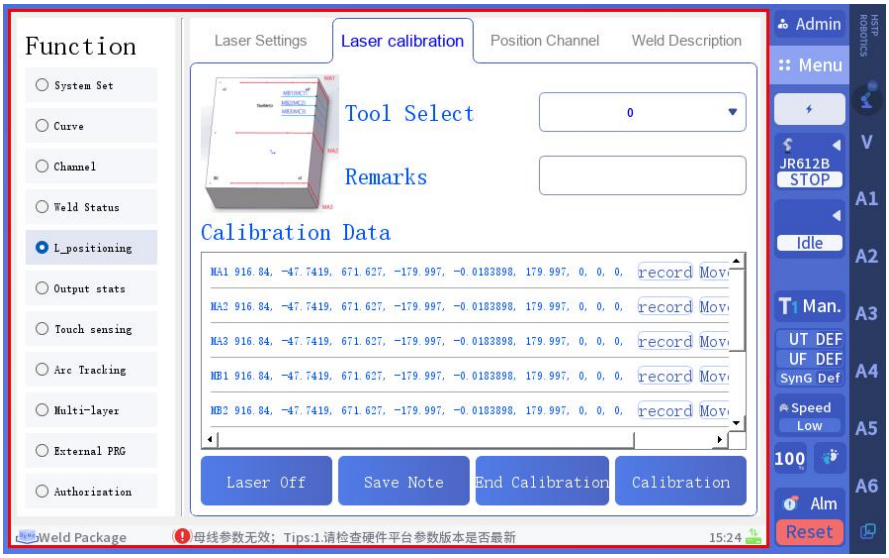


6) Record the location MB1

As shown in the figure below: For the MB1 position, the laser positioning sensor must scan the weld seam to the upper-middle edge (marked in red), and the laser line of the welding gun must align with the straight line marked on the carbon steel plate. The current position should be recorded.

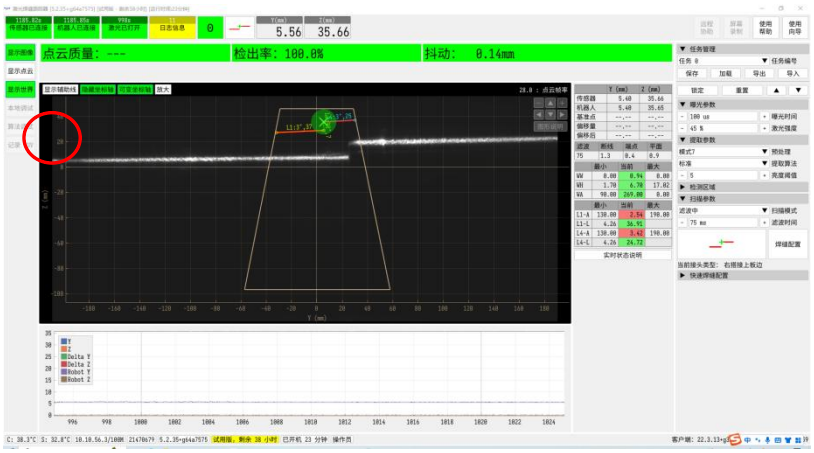


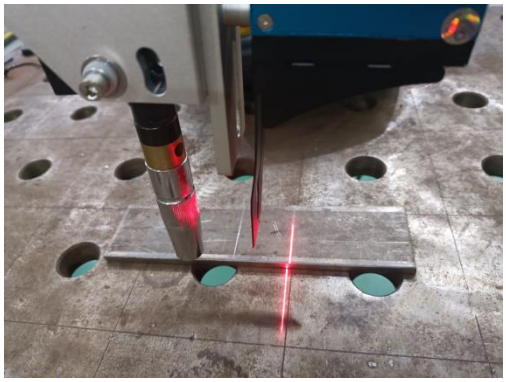


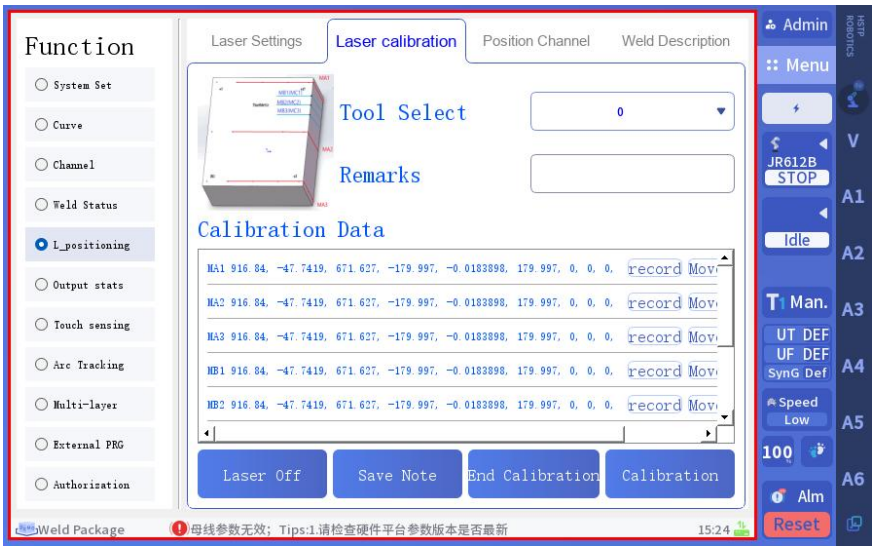


7) Record the location MB2

As shown in the figure below: For MB2 position, the laser positioning sensor must scan the weld seam to the upper right edge (marked in red), and the laser line of the welding gun must align with the straight line on the carbon steel plate. Record the current position.

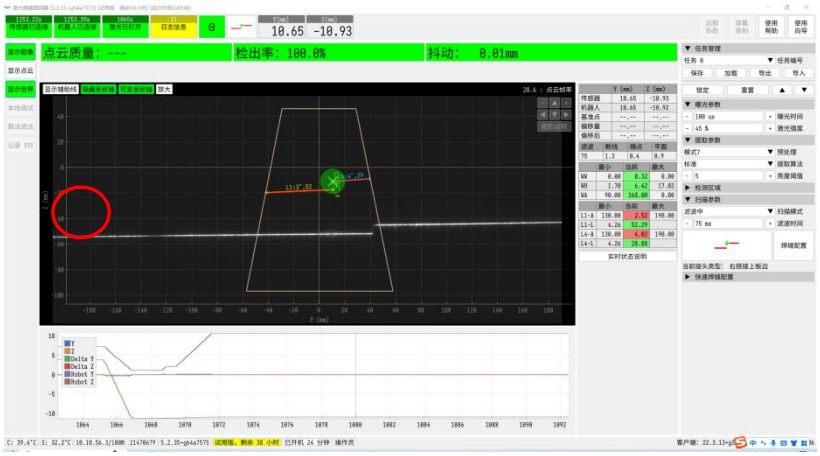


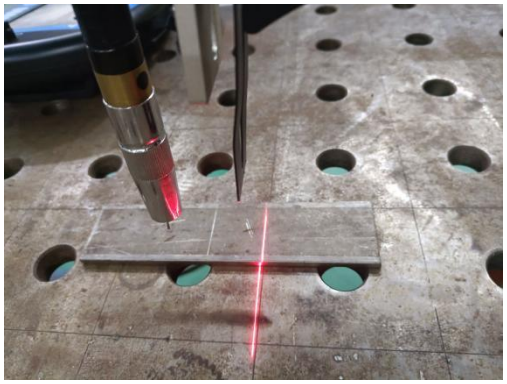


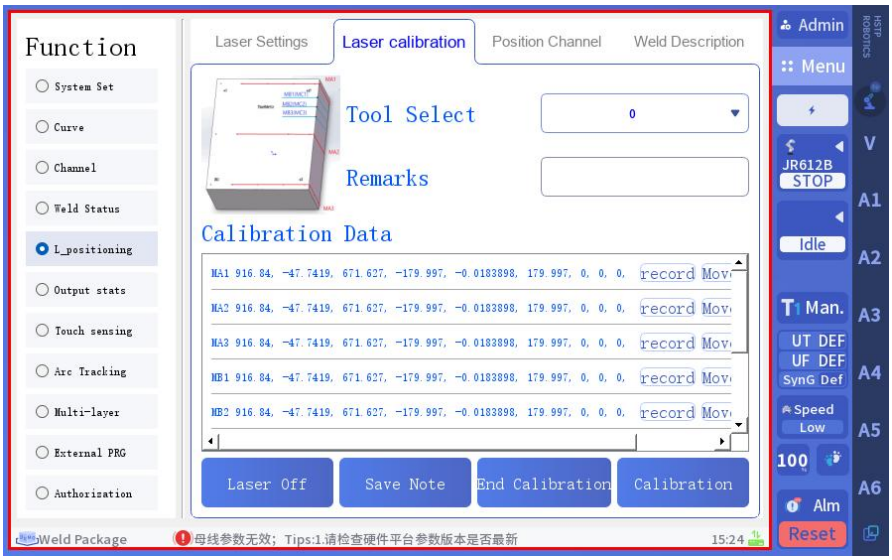


8) Record the location MB3

As shown in the figure below: For MB3 position, the laser positioning sensor must scan the weld seam to the lower right edge (marked in red), and the laser line of the welding gun must align with the straight line on the carbon steel plate. Record the current position.

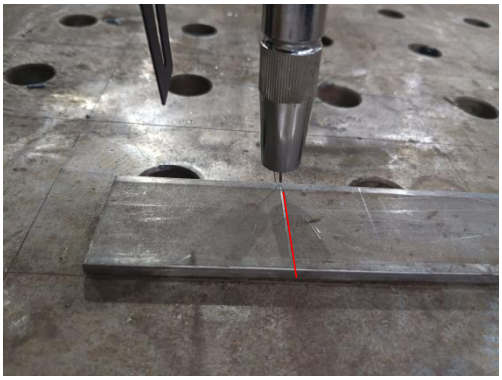


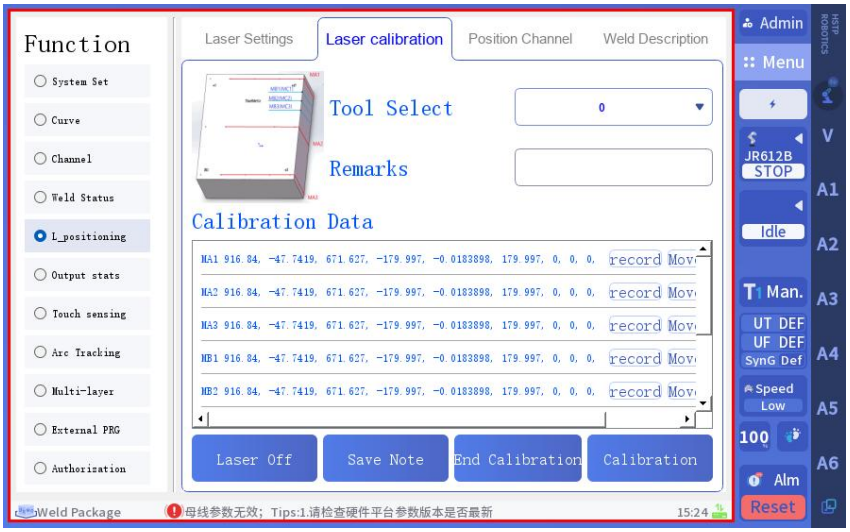




9) Record the locations MC1, MC2, and MC3

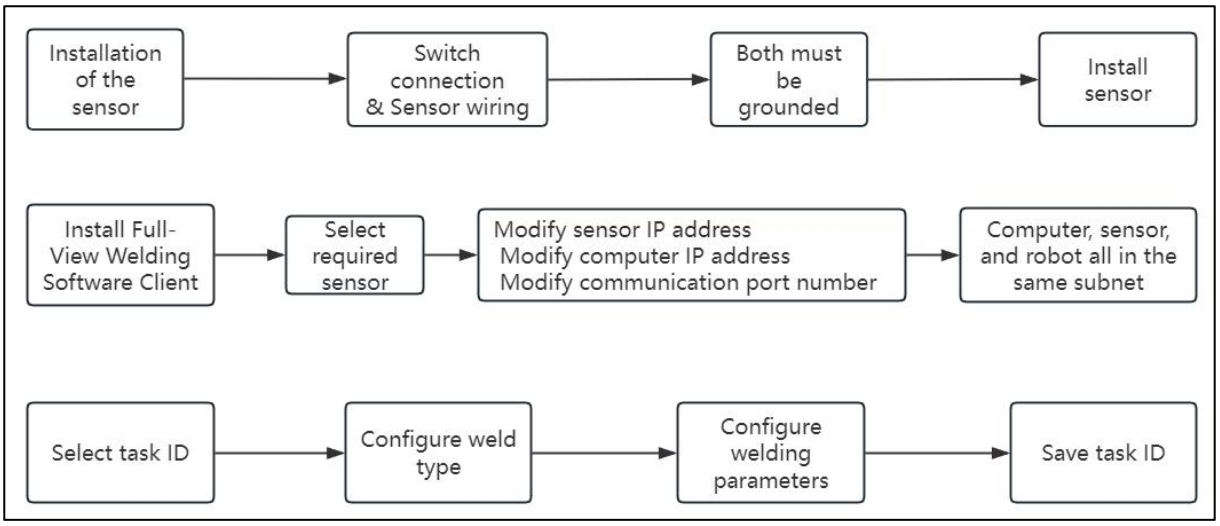
As shown in the figure below: To position MC1, MC2, and MC3, simply move the robot to the weld seam, align the welding wire tip with the marking points on the scanning position, and record the remaining three points consecutively. After recording all nine points, click the calibration button below. The system will confirm calibration success, and the calibration process will conclude.





7.36.10.4 Laser parameter adjustment

Use the operation procedure guide (refer to the Suzhou Quanshi Welding Seam System Configuration Software User Manual for detailed instructions)



1) Laser on/off command

The laser on/off command is inserted directly to turn the laser on or off.

2) Starting point search

The Start Point Search command is used to locate the weld start point. It requires aligning and recording the weld start point, selecting the weld type, and setting the search speed individually.

Start/End laser tracking

The laser tracking command is used to enable laser tracking, usually after the arc initiation command.

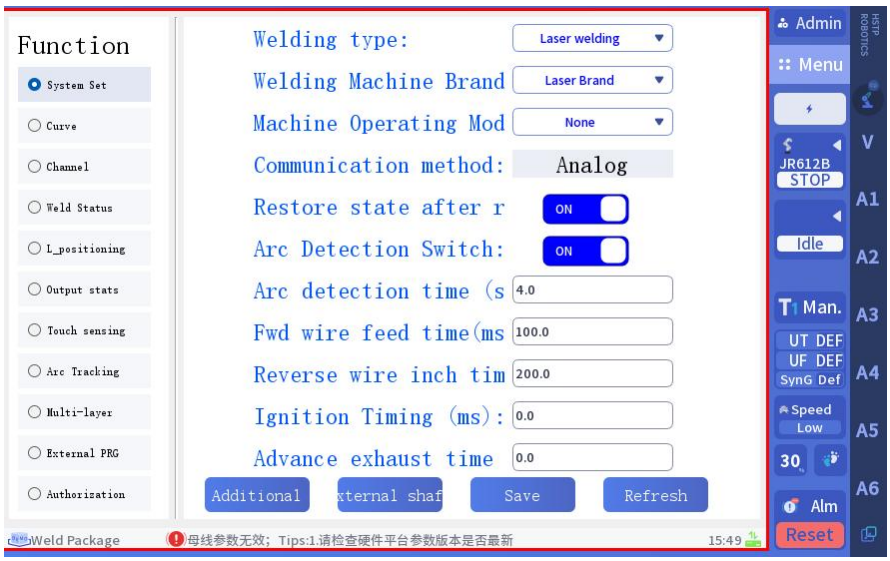
The end laser tracking instruction is to end the laser tracking, which is called after tracking the trace point.

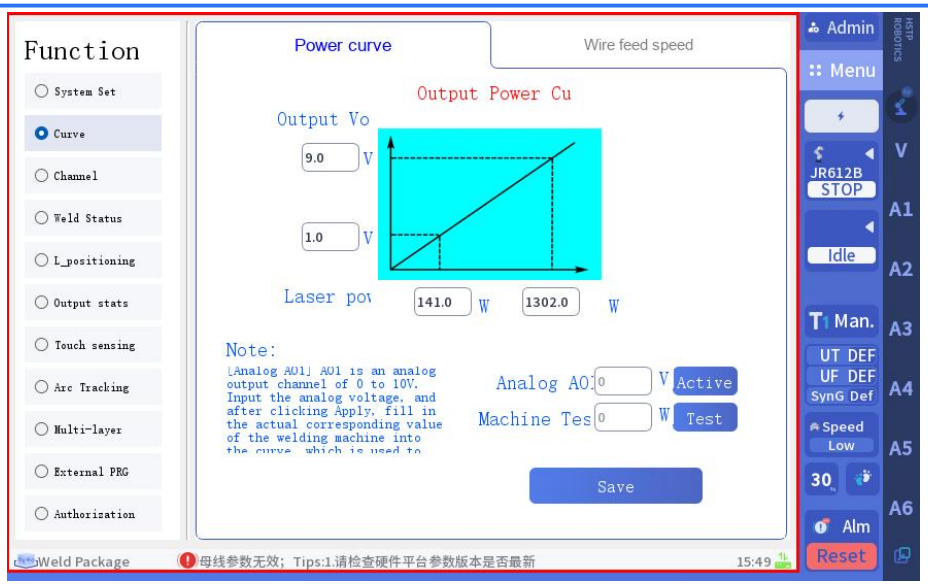
```
2  L P[1]
3  MAC LASER_ON, OUT
4  MAC LASER_SEARCH_TRACK P[3] WELD_TYPE=1 VEL=8, OUT
5  MAC ARC_ON R[500],OUT
6  MAC LASER_START_TRACK, OUT
7  L P[2]
8  MAC LASER_STOP_TRACK, OUT
9  MAC ARC_OFF R[500],OUT
10 MAC LASER_OFF, OUT
11 J P[3]
```

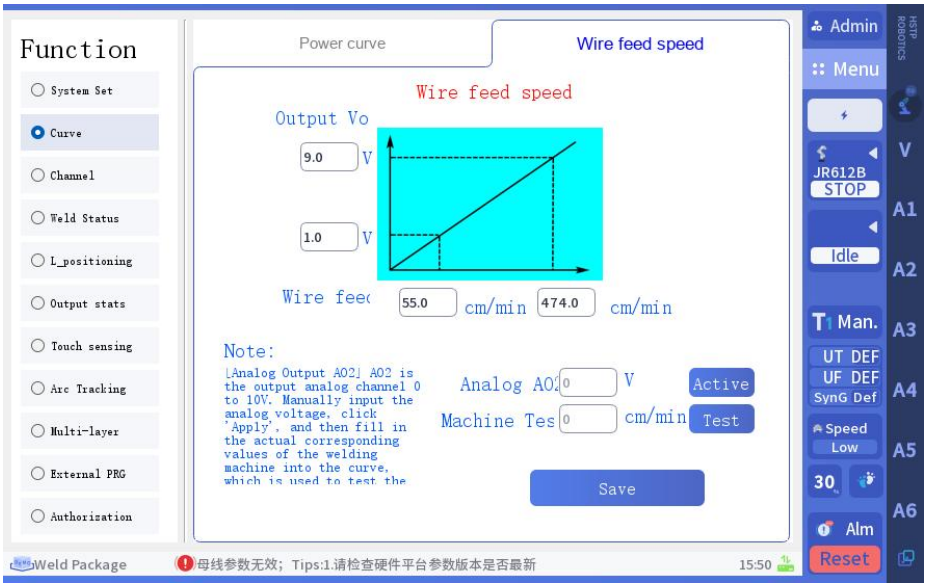
7.36.11 Laser welding instructions

7.36.11.1 Laser welding configuration

First, select laser welding as the welding method in the process package configuration interface. Then, configure the corresponding curve parameters as for conventional welding processes. Specifically, for laser welding, configure the power curve and wire feed speed curve, as shown in the figure below.

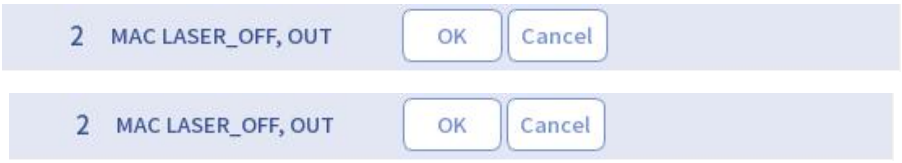






7.36.11.2 Light opening/receiving instructions

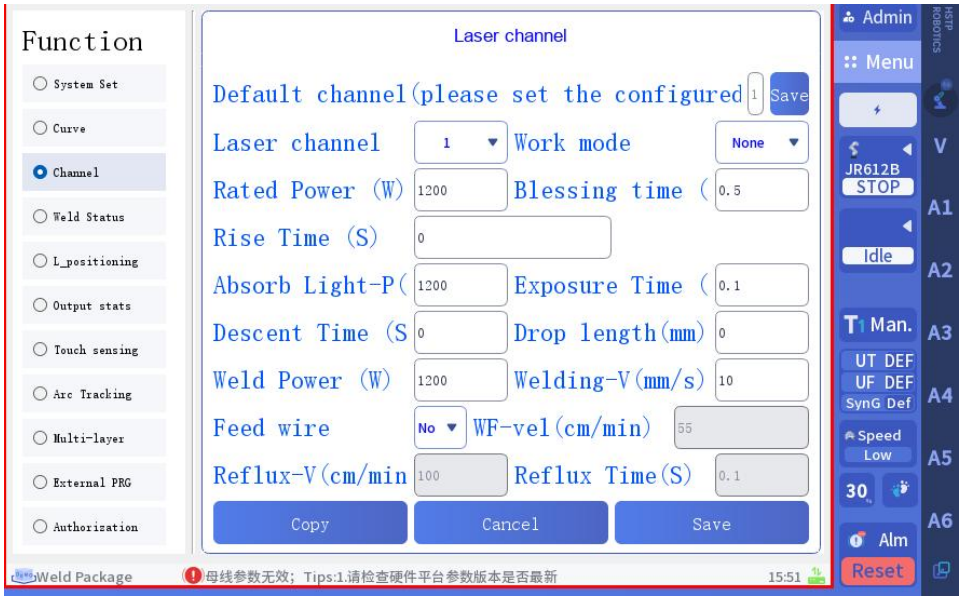
The diagram below shows the laser welding light-on/off command. Inserting it activates the corresponding laser welding channel.



7.36.11.3 Example of laser welding procedure

As shown in the figure below, an example of a laser welding program

```
2  JP[1]
3  LP[2]
4  MAC OPEN_LIGHT ARCINGCHANNELINDEX=1,OUT
5  LP[3]
6  MAC CLOSE_LIGHT ARCINGCHANNELINDEX=1,OUT
7  JP[4]
```



7.36.12 Contact positioning command

7.36.12.1 Start/Stop contact positioning command

The command initiates contact-based positioning, containing a channel number parameter that corresponds to the designated channel in the positioning settings. (During the first run, enable the reference flag in the channel, complete one cycle, then disable it. Move the workpiece to ensure the welding wire contacts it before proceeding.)

The command ends contact positioning. No parameters are required—just insert directly.

7.36.12.2 Contact-based positioning movement command

The instruction is a motion instruction, which needs to record a P point and input a position point number.

7.36.12.3 Calculation offset instructions

The command calculates offset data. You can select offset types: -1D,2D,3D, or 2D+. For 1D, enter the contact-based positioning point number of the motion command. For 2D, enter the positioning point numbers of two contact-based motion commands. For 3D and 2D+, enter the positioning point numbers of three contact-based motion commands. The calculated offset data is stored in the input offset number.

7.36.12.4 Start/End offset instructions

The command calculates the offset data after the start offset. Enter the offset number calculated by the offset command.

The command is an end offset command. No parameters are required. Insert it directly.

7.36.12.5 Offset starting point calculation instruction

The offset calculation instruction supports four welding groove types: right-angle, V-groove, butt, and intermediate point, with three vertical surfaces (triangular). For right-angle grooves, two contact motion position numbers are required; V-grooves need four, butt grooves five, intermediate points two, and vertical surfaces three. The calculated starting coordinates are stored in the input LR register.

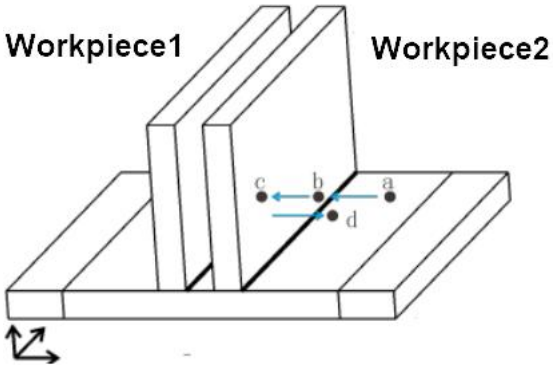
7.36.12.6 Instruction for positioning of corner welds

1. 1D

Service condition ：

The workpiece moves in only one direction, and the positioning direction must be in the moving direction.

Example program:



Workpiece offset in one direction

Note: To locate the initial reference position, the flag must be activated. For 1D,2D, and 3D positioning, incremental positioning can be enabled.

Detailed Description of 1D Positioning Program		
line number	programmed instruction	explain
1	Line P[0]	//a drop
2	Start contact-based positioning. Channel number =1	// Position search start
3	Contact-based positioning motion P[1] Positioning point=1	// Recorded as point b, during operation, it searches for point c from point b and automatically returns to point d direction. The location exists, search point 1
4	end contact positioning	// Position search completed

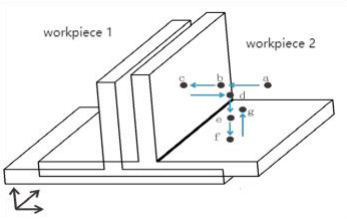
5	Calculate offset Offset type=1D Position point 1=1 Offset number=2	// Calculate offset. The data has an offset number 2
6	Start Offset Offset Number=2	//Start offset number 2 data
7	Line P[2]	// offset point
8	End Offset	// End Offset

2. 2D

service condition ：

In the workpiece coordinate system, translating on any two planes of XYZ is similar to 1D, with one point in each of the two changing directions.

Example program:



Workpiece offset in two directions

Note: To locate the initial reference position, the flag must be activated. For 1D,2D, and 3D positioning, incremental positioning can be enabled.

Detailed Description of 2D Positioning Program		
line number	programmed instruction	explain
1	Line P[0]	//a drop
2	Start contact-based positioning. Channel number =1	// Position search started

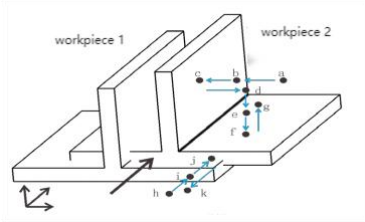
3	Contact-based positioning motion P[1] Positioning point=1	// Recorded as point b, during operation, it searches for point c from point b and automatically returns to point d direction. The location exists, search point 1
4	Contact-based positioning motion P[2] with positioning point=2	// Record as point e, during operation search for point f from point e, automatically return to point g direction, the location exists search point 2
5	end contact positioning	// Position search completed
6	Calculate offset Offset type=2D Position point 1=1 Position point 1=2 Offset number=3	// Calculate offset. The data has an offset number 3
7	Start Offset Offset Number=3	//Start offset number 3 data
8	Line P3]	// offset point
9	End Offset	// End Offset

3. 3D

service condition ：

In the workpiece coordinate system, translating on any two planes of XYZ is similar to 1D, with one point in each of the two changing directions.

Example program:



Workpiece offset in three directions

Note: To locate the initial reference position, the flag must be enabled. For 1D,2D, and 3D positioning, incremental positioning can be activated.

Detailed Description of 3D Positioning Program		
line number	programmed instruction	explain
1	Line P[0]	//a drop
2	Start contact-based positioning. Channel number =1	// Position search started
3	Contact-based positioning motion P[1] Positioning point=1	// Recorded as point b, during operation, it searches for point c from point b and automatically returns to point d direction. The location exists, search point 1
4	Contact-based positioning motion P[2] with positioning point=2	// Record as point e, during operation search for point f from point e, automatically return to point g direction, the location exists search point 2

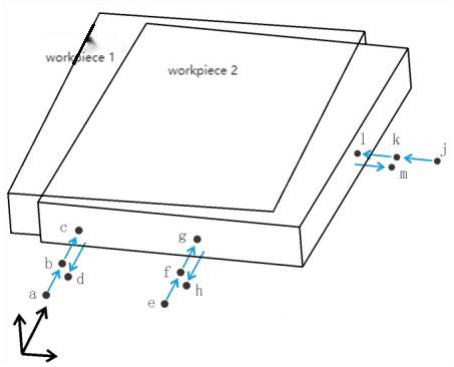
5	Line P[3]	//h drop
6	Contact-based positioning motion P[4] Positioning point=3	// Recorded as point i, during runtime it locates point j from point i and automatically returns to point k direction. The location exists to locate point 3.
7	end contact positioning	// Position search completed
8	Calculate offset Offset type=3D Position point 1=1 Position point 1=2 Position point 1=3 Offset number=4	// Calculate offset. The data has an offset number 4
9	Start Offset Offset Number=4	//Start offset number 4 data
10	Line P[5]	// offset point
11	End Offset	// End Offset

4. 2D+

service condition ：

Rotate around any X, Y, or Z axis (or the Z axis in Cartesian coordinates) of the workpiece, or move in any two directions.

Example program:



The workpiece is offset in two directions and rotates in a third direction.

Detailed Analysis of 2D+ Positioning Program		
line number	programmed instruction	explain
1	Line P[0]	//a drop
2	Start contact-based positioning. Channel number =1	// Position search start
3	Contact-based positioning motion P[1] Positioning point=1	//Recorded as point b, during operation, it searches for point c from point b and automatically returns to point d direction. The location exists, locate point 1
4	Line P[2]	//e drop
5	Contact-based positioning motion P[3] Positioning point=2	// Recorded as point f, the system searches for point g from point f and automatically returns to point h. The location exists, and point 2 is being searched.

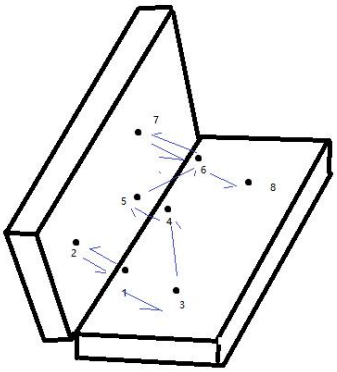
6	Line P[4]	//j drop
7	Contact-based positioning motion P[5] Positioning point=3	The system records k points, searches for point l from point k during operation, automatically returns to point m direction, and exists a search point 3 at the location.
8	end contact positioning	// Position search completed
9	Calculate offset Offset type=2D+ Position 1=1 Position 1=2 Position 1=3 Offset number=4	// Calculate the offset. The data has an offset number 4
10	Start Offset Offset Number=4	//Start offset number 4 data
11	Line P[6]	// offset point
12	End Offset	// End Offset

5. V

service condition ：

Rotate or move around any Y or Z axis (or the Z axis in geographic coordinates) of the workpiece. The X direction and rotation offset around X cannot be calculated.

Example program:



As shown in the figure, the V-shaped weld seam requires locating five points: 1 at 2, 3; 4 at 5; 6 at 7, 8. The points 2, 5, and 7 must not lie on the same line to define a plane.

The workpiece is offset in two directions and rotates in a third direction.

Detailed Explanation of V Positioning Program		
line number	programmed instruction	explain
1	Line P[1]	// 1 point
2	Start contact-based positioning. Channel number =1	// Position search started
3	Contact-based positioning motion P[2] Positioning point=1	//Set to 2 points. During operation, it searches for position 1 from 2 points and can be configured to return automatically. Position 1 exists.
4	Line P[1]	//1 point
5	Contact-based positioning motion P[3] with positioning point=2	//Recorded as point 3. During operation, it searches for position 2 from point 3

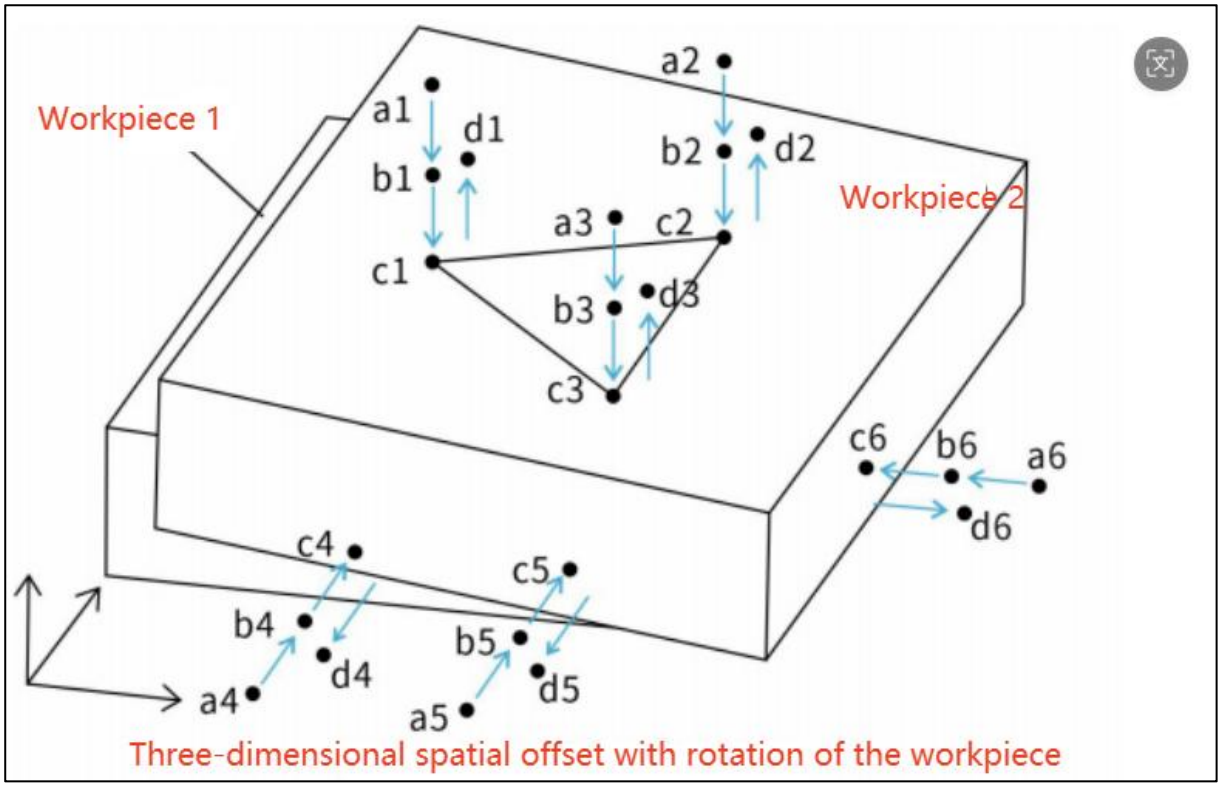
		and can be set to auto-return. Position 2 exists.
6	Line P[4]	// 4 points
7	Contact-based positioning motion P[5] Positioning point=3	//Recorded as point 5. During operation, it searches for position 3 from point 5 and can be set to auto-return. Position 3 exists.
8	Line P[6]	// 6 oclock
9	Contact-based positioning motion P[7] Positioning point=4	//Recorded as 7. During operation, it searches for position 4 at 7 and can be set to auto-return. Position 4 exists.
10	Line P[6]	// 6 oclock
11	Contact-based positioning motion P[8] Positioning point=5	//Recorded as 8 oclock. During operation, it searches for position 5 at 8 oclock and can be set to auto-return. Position 5 exists.
12	end contact positioning	// Position search completed
13	Calculate offset Offset type=V Position 1=1 Position 2=2 Position 3=3 Position 4=4 Position 5=5 Offset number=1	// Calculate the offset. The data has an offset number 1
10	Start Offset Offset Number=1	//Start offset number 1 data

11	Line P[9]	// offset point
12	End Offset	// End Offset

3D+

service condition ：

Rotate around any X, Y, or Z axis (or the Z axis in Cartesian coordinates) of the workpiece, or move in three directions.



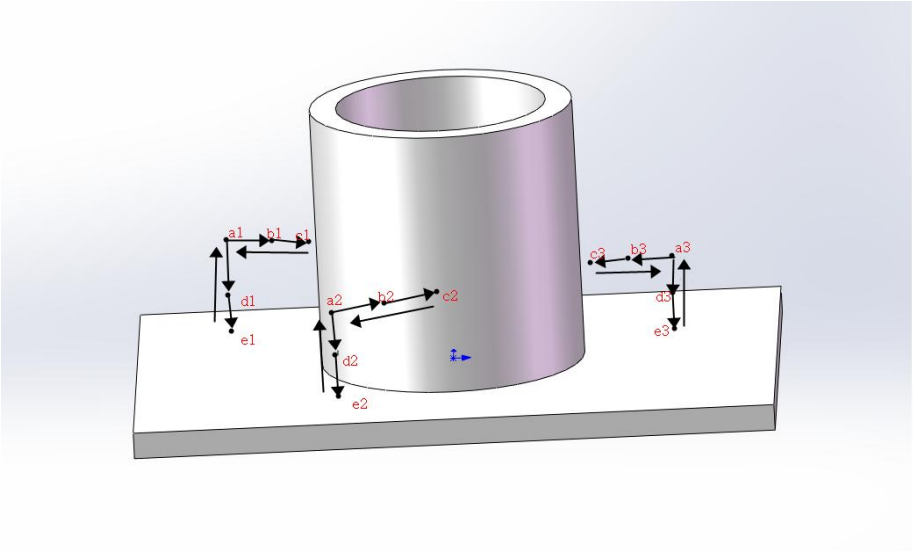
Detailed Explanation of 3D+ Positioning Program		
line number	programmed instruction	explain
1	Line P[0]	// Point a1
2	Start contact-based positioning. Channel number =1	// Position search start
3	Contact-based positioning motion P[1]	//Recorded as point b1,

	Positioning point=1	during operation, it searches from point b1 to point c1 and automatically returns to point d1 direction. The location exists, search point 1
4	Line P[2]	//Point A2
5	Contact-based positioning motion P[3] Positioning point=2	// Recorded as point b2, during operation, it searches from point b2 to point c2 and automatically returns to point d2 direction. The location exists, search point 2
6	Line P[4]	// Point A3
7	Contact-based positioning motion P[5] Positioning point=3	// Recorded as point b3, during operation it searches from point b3 to point c3, automatically returning to point d3 direction, with point 3 being the search point
8	Line P[6]	transition point
9	Line P[7]	transition point
10	Line P[8]	// A4 point
11	Contact-based positioning motion P[9] Positioning point=4	// Recorded as point b4, during operation it searches from point b4 to point c4, automatically returning to point d4 direction, with point

		4 being the target location
10	Line P[10]	// at 5 a.m.
11	Contact-based positioning motion P[11] Positioning point=5	// Recorded as point b5, during operation it searches from point b5 to point c5, then automatically returns to point d5 direction, with point 5 being the target location
12	Line P[12]	transition point
13	Line P[13]	transition point
14	Line P[14]	// Point A6
15	Contact-based positioning motion P[15] Positioning point=6	// Recorded as point b6, during operation it searches from point b6 to point c6, automatically returning to point d6 direction, with point 6 being the search target
16	end contact positioning	// Position search completed
17	Calculate offset Offset type=3D+ Position 1=1 Position 2=2 Position 3=3 Position 4=4 Position 5=5 Position 6=6 Offset number=4	// Calculate the offset. The data has an offset number 4
18	Start Offset Offset Number=4	//Start offset number 4 data
19	Line P[16]	// offset point
20	End Offset	// End Offset

7.arc

Usage conditions: The workpiece can move in any direction (X, Y, Z) or in multiple directions. The system will locate three points from the workpieces exterior and three points along the Z-axis.



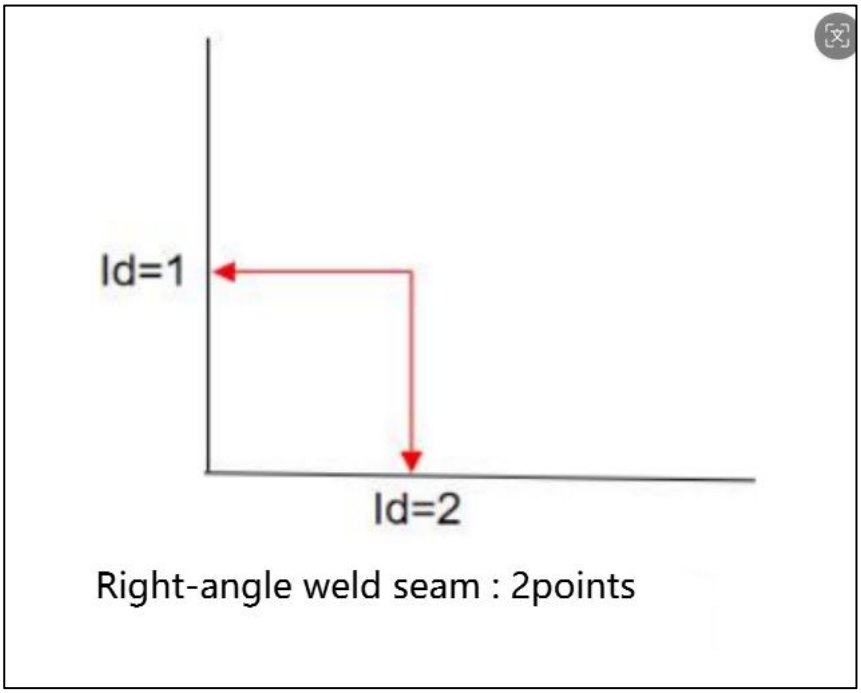
Detailed Explanation of 3D+ Positioning Program		
line number	programmed instruction	explain
1	Line P[0]	// Point a1
2	Start contact-based positioning. Channel number =1	// Position search started
3	Contact-based positioning motion P[1] Positioning point=1	//Recorded as point b1, it moves from b1 to point c1 during operation and automatically returns. The location includes a positioning point 1.
4	Line P[0]	//Point A1
5	Contact-based positioning motion P[2] with	// Recorded as point a1,

	positioning point=2	the system automatically returns from point d1 to locate point e1 during operation, with point 2 being the reference point
6	Line P[3]	// transition point
7	Line P[4]	// Point A2
8	Contact-based positioning motion P[5] Positioning point=3	//Recorded as point a2, it moves from point b2 to point c2 during operation and automatically returns. The location includes a positioning point 3.
9	Line P[3]	// A4 point
10	Contact-based positioning motion P[6] Positioning point=4	//Recorded as point a2, it moves from point d2 to point e2 during operation and automatically returns. The position includes location point 4.
11	Line P[7]	// transition point
12	Line P[8]	// Point A3
13	Contact-based positioning motion P[9] Positioning point=5	//Recorded as point a3, it moves from point b3 to point c3 during operation and automatically returns. The position includes location

		point 5.
14	Line P[10]	transition point
15	Line P[8]	// Point A3
16	Contact-based positioning motion P[11] Positioning point=6	// Recorded as point a3, during operation, locate point e3 from point d3, automatically return to point 6 for position search
17	end contact positioning	// Position search completed
18	Calculate offset Offset type=Round Position point 1=1 Position point 2=2 Position point 3=3 Position point 4=4 Position point 5=5 Position point 6=6 Offset number=7	// Calculate the offset. The data has an offset number 4
19	Start Offset Offset Number=7	//Start offset number 4 data
20	Line P[12]	// offset point
21	Arc Start (1)	// scratch start
22	Arc P[13]	//Same as P[12]
23	Arc P[14]	// offset point
24	Arc P[15]	// offset point
25	Arc Close (1)	//Arc closure
26	End Offset	// End Offset

8. Right-angle weld

APPLICATION CONDITION: Vertical weld.

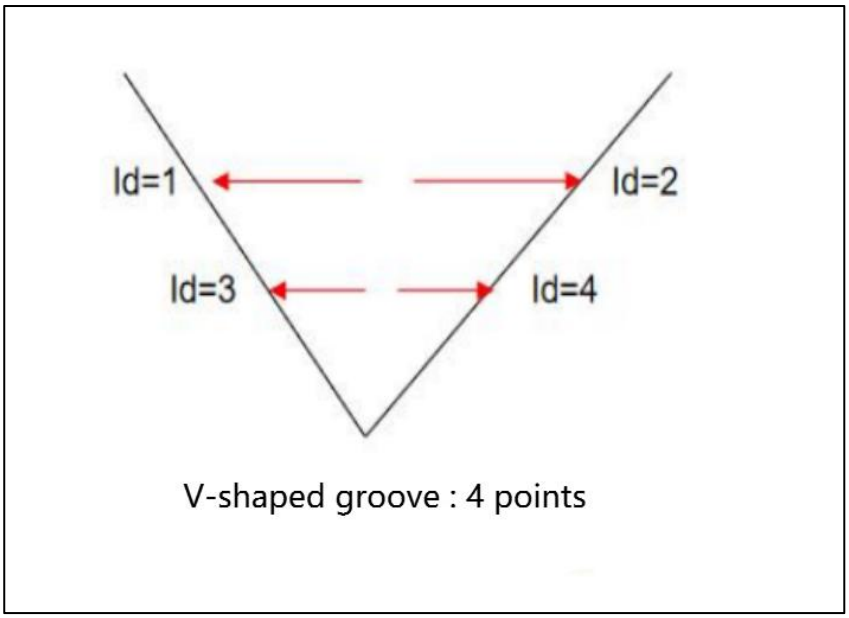


Detailed Description of the Positioning Program for Right Angle Welding		
line number	programmed instruction	explain
1	Line P[0]	//
2	Start position search channel number=1	//
3	Contact-based positioning motion P[1] Positioning point=1	
4	Line P[0]	
5	Contact-based positioning motion P[2] with positioning point=2	
6	end contact positioning	// Position search completed

7	Calculation point Weld type=1 Register number=3	//
8	L LR[3]	

9. V-shaped groove

Usage condition: V-groove.

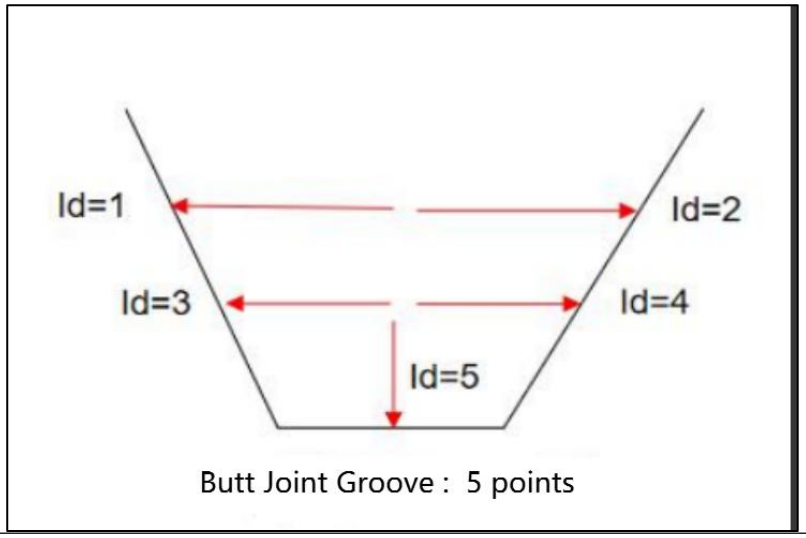


Detailed Description of V-shaped Groove Positioning Procedure		
line number	programmed instruction	explain
1	Line P[0]	//
2	Start position search channel number=1	//
3	Contact-based positioning motion P[1] Positioning point=1	
4	Line P[0]	
5	Contact-based positioning motion P[2] with positioning point=2	

6	Line P[1]	
7	Contact-based positioning motion P[3] Positioning point=3	
8	Line P[1]	
9	Contact-based positioning motion P[4] Positioning point=4	
10	end contact positioning	// Position search completed
11	Calculation starting point Weld type=2 Register number=3	//
12	L LR[3]	

10. Bevel joint

Usage condition: For butt bevel.

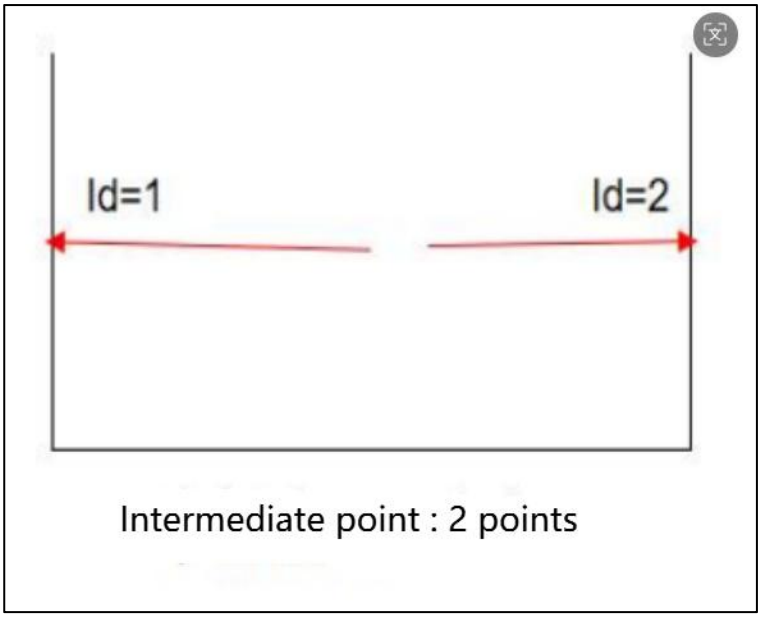


Detailed Analysis of Positioning Procedure for Jointed Groove		
line number	programmed instruction	explain

1	L P[0]	//
2	Start position search channel number=1	//
3	Contact-based positioning motion P[1] Positioning point=1	
4	L P[0]	
5	Contact-based positioning motion P[2] with positioning point=2	
6	Line P[1]	
7	Contact-based positioning motion P[3] Positioning point=3	
8	Line P[1]	
9	Contact-based positioning motion P[4] Positioning point=4	
10	Line P[1]	
11	Contact-based positioning motion P[5] Positioning point =5	
12	end contact positioning	// Position search completed
13	Calculation starting point Weld type=3 Register number=3	//
14	L LR[3]	

11.intermediate points

Usage: Midpoint.



Detailed Description of Midpoint Positioning Procedure		
line number	programmed instruction	explain
1	Line P[0]	//
2	Start position search channel number=1	//
3	Contact-based positioning motion P[1] Positioning point=1	
4	Line P[0]	
5	Contact-based positioning motion P[2] with positioning point=2	
6	end contact positioning	// Position search completed
7	Calculation point Weld type=4 Register number=3	//
8	L LR[3]	

12. Triangular projection plane

Usage: Triangular projection.

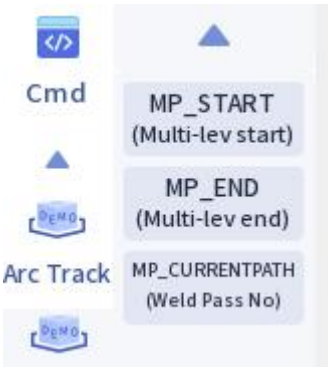
7.38.3.8.6 Incremental Positioning

Incremental positioning extends standard positioning by dynamically adjusting the reference point based on the workpieces actual offset. Unlike fixed-point positioning, it recalibrates the reference point according to the offset detected from the previous position. To activate this feature, simply turn on the corresponding increment switch in the positioning channel. All other settings remain identical to conventional positioning.

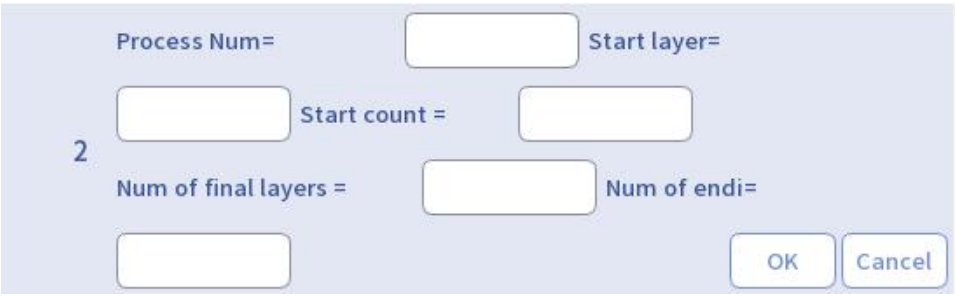
7.36.13 Multi-level and multi-path instructions

7.36.13.1 Start/End multi-layer multi-channel

Open the program and select the welding instruction-Multi-layer and multi-pass instruction. You can insert the start and end points of the multi-layer and multi-pass instruction. The multi-layer and multi-pass sequence number instruction is not in use.



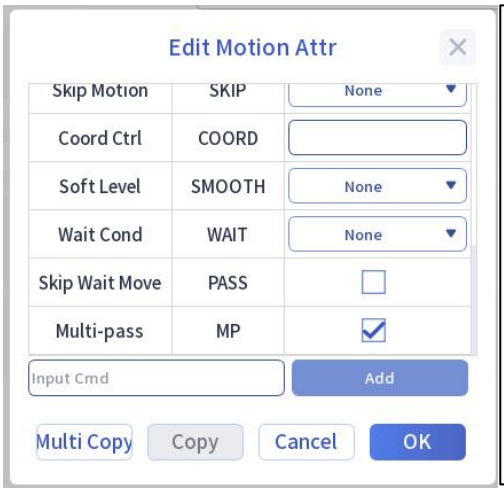
The instruction is a multi-layer and multi-track starting point. After selecting the process number, the program uses the corresponding process number parameters and places them before the multi-layer and multi-track trajectory. By modifying the corresponding layer and track numbers, you can start welding from any layer and track number.

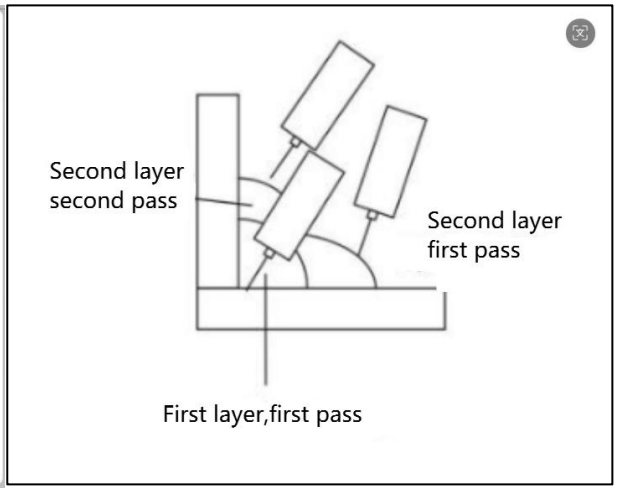


The instruction is a multi-level, multi-channel endpoint, placed after the multi-level, multi-channel trajectory.

7.36.13.2 Multi-layer multi-channel program example

As illustrated below, the key distinction between multi-layer multi-track programs and standard programs in multi-layer multi-track trajectory programming lies in the additional elements: the multi-layer multi-track start command, multi-layer multi-track end command, transition point MP, and arc start point MP. The multi-layer multi-track start and end commands can be identified through the multi-layer multi-track instruction set within the welding command. For the transition point MP and arc start point MP, users must first select the motion command requiring modification, then click the "Change-Add Instruction-MP" option in the lower-left corner, followed by clicking the small box next to the selection.





7.36.14 Arc tracking command

7.36.14.1 Introduction to arc tracking function

Arc tracking technology in welding robots is an automated parameter adjustment system that monitors real-time arc signals during welding to dynamically track and optimize the welding path, thereby enhancing both quality and precision. The oscillatory arc tracking method regulates welding heat input by controlling the arc to oscillate periodically over the molten pool, while simultaneously adjusting the welding path through these oscillations. This technology improves the robots adaptability to varying welding conditions and welding accuracy, serving as a key solution for achieving high-quality, high-efficiency automated welding.

7.36.14.2 Precautions for arc tracking

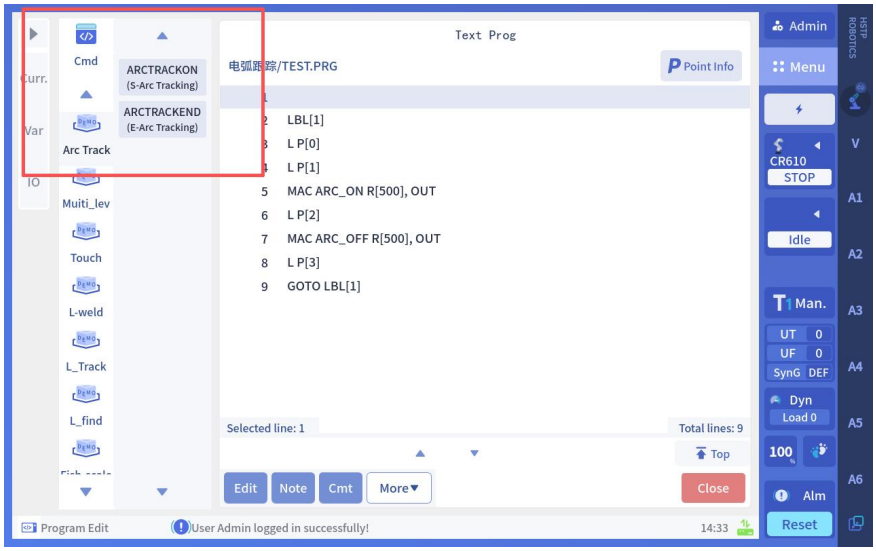
A. The arc tracking function must be taught to the swing welding program to effectively collect the current.

B. The arc tracking function is designed for V-groove or fillet welds, which are symmetrical in axial orientation.

C. The current version of arc tracking is compatible only with the Pulse mode of the Ota Pulse-MIG 630RPH. For other welding machines or modes, the arc tracking function is not recommended for use.

Arc Tracking Command and Program Example

After opening the program, click Start Editing → Welding Instruction → Arc Tracking Instruction. The arc tracking instruction includes Start Tracking and End Tracking commands.



7.36.14.3 Start tracking instruction

The command initiates arc tracking and includes a channel number parameter. This parameter corresponds to the channel in the arc tracking settings (shared by upper, lower, and left/right channels).

7.36.14.4 End arc tracking command

The command is a tracking end command. No parameters are required. Insert it directly.

Follow-up instructions

7.36.14.5 Demonstration of the teaching procedure

Line P[0]	arc starting point of spot welding
Start Arc Tracking (1)	Start tracking
Arc Start (1)	Arc-start command, channel 1
Start soldering (1)	call the swing welding channel
Line P[1]	end of flip-chip placement
Arc Close (1)	Arc closing instruction, call channel 1
End of pick-and-place	
end arc tracking	end arc tracking

Joint P[2]	Departure point

```
2  L P[1]
3  MAC ARCTRACKON ARCTRACKCHANNEL=1,OUT
4  MAC ARC_ON R[500],OUT
5  MAC WEAVESTART ARCINGWELDCHANNELINDEX=1,OUT
6  L P[2]
7  MAC ARC_OFF R[500],OUT
8  MAC WEAVEEND ARCINGWELDCHANNELINDEX=1,OUT
9  MAC ARCTRACKEND, OUT
10 J P[3]
```

8 Client service

8.1 Technical support enquiry

Introductory

This document provides information about the operation and handling of the robotic arm and can help you troubleshoot problems.

Information

The following information is required to provide technical support:

- Description of the problem, including:
  - ■ Steps of operation performed some time before the alarm fault.
  - ■ Information about the duration and frequency of the fault.
- As much detailed information as possible about the hardware and software components of the overall system, such as:
  - ■ Robot model, nameplate number
  - ■ Controller model, factory number
  - ■ System software name and version
  - ■ Diagnostic data package
    - ◆ The diagnostic data export in the system software allows you to consult your technician about exporting streamlined data.
  - ■ Existing installed plug-in packages
  - ■ Program files for existing applications
  - ■ Additional axis cases for existing configurations

9 Appendix

9.1 Glossary

name	Instructions
flangeless	● The end of the arm, i.e. the end of the 6-axis robot, is flanged.
singularity	● During the motion of a six-axis robot, if the angle of the fifth axis is 0 (J5=0) and the fourth axis is co-linear with the sixth axis, the angle of the two axes of the fourth and sixth axes will produce an infinite number of sets of solutions when the kinematic inverse solution is calculated. Such positions are called singularities in robotics. ● In practice, when the six-axis robot moves in a spatial POINT coordinate system, it is most likely to pass through the singularity, and rapid rotation of the joints will occur, so it should try to avoid any singular POINT in the motion planning.
Euler's angle	● Euler angles are a method of spatial coordinate attitude in a robotic arm, which determines the orientation of the coordinates through three angles of rotation. The Euler angles for the Wachovia III use the ZYX Euler angle type, which means that the object is first rotated by one angle around the z-axis, then by one angle around the rotated y-axis, and finally by one angle around the rotated x-axis.

9.2 Global motion parameters command table

Parameter name	Value range	Default value	MeaningInstructions
CNT_TYPE	0/1/2	0	Smoothing type parameters
CNT	0~100	0	Global Smooth Transition Parameters
J_VEL	The unit is percentage and is divided according to the mode: Manual T1: 1~10 Manual T2: 1~20 Automatic/External: 1~100	100	Joint velocity
J_ACC	1~200	100	Joint acceleration ratio in per cent
J_DEC	1~200	100	Joint reduction ratio in per cent
L_VEL	Limited according to mode: Manual T1: 1~125 mm/s Manual T2: 1~250 mm/s Auto/External: 1~2400 mm/s, max.	Maximum values	Linear velocity

	2400 but different for each model.		
L_ACC	1~200	100	Linear acceleration ratio in per cent
L_DEC	1~200	100	Linear reduction ratio in per cent
L_VROT	Limited according to mode: Manual T1: 1~30 °/s Manual T2: 1~80 °/s Auto/External: 1~120 °/s , max. 120 but different for each model.	Maximum values	Linear Attitude Velocity
C_VEL	Limited according to mode: Manual T1: 1~125 mm/s Manual T2: 1~250 mm/s Auto/External: 1~2400 mm/s, max. 2400 but different for each model.	Maximum values	Circular arc velocity
C_DEC	1~200	100	Circular reduction ratio in per cent
C_ACC	1~200	100	Circular acceleration ratio in per cent
C_VROT	Limited according to mode: Manual T1: 1~30 °/s Manual T2: 1~80 °/s Auto/External: 1~120 °/s , max. 120 but different for each model.	Maximum values	Circular Attitude Velocity
SMOOTH	1~9	5	Parameters of softness level

9.3 Table for assigning operating privileges of the demonstrator

First level menu	Secondary Menu	Third level menu	User group		
			Production staff	Debugger	administrators
File management	Programme data file management	——	√	√	√
	User PLC file management	——	View/Exportable	√	√
	System parameter file management	——	View/Exportable	√	√
File operation	newly built	——	Check out	√	√
	load	——	/	√	√
	save	——	√	√	√
	removing	——	/	√	√
	backing up	——	/	√	√
	resumption	——	/	√	√

	multiple choice	——	/	√	√
	rename	——	/	√	√
	lock	——	/	√	√
	Switch on/off Instructions	——	√	√	√
	On/Off status	——	√	√	√
	Deactivate Waiting	——	√	√	√
	optimisation	——	√	√	√
	turn on	——	√	√	√
configuration	Demonstrator Configuration	user group	√	√	√
		Alternate Key Configuration	Check out	√	√
		Lock Password Setting	——	√	√
		Robot Communication Configuration	Check out	√	√
	Controller Configuration	Running configuration	Check out	√	√
		Encoding/Decoding Configuration	Check out	√	√
		Regional configuration	Check out	√	√
		Controller IP Configuration	Check out	√	√
		Axis Group Configuration	Check out	Check out	√
		Co-group configuration	Check out	√	√
		Timeout setting	√	√	√
		Oscillating configuration	/	√	√
		Back to Zero Procedure Configuration	/	√	√

		Speed Configuration	Check out	√	√
		Back end Programming	Check out	√	√
		path-holding function	Check out	√	√
	Kinetic function	Mounting Attitude Setting	Check out	√	√
		Ontological friction identification	Check out	√	√
		Load Configuration	/	√	√
		Collision Detection Configuration	/	√	√
		Drag-and-drop instructional settings	/	√	√
		Drag Track Recording	/	√	√
Demonstrate	Digital inputs/outputs	——	/	√	√
		——	/	√	√
	Analogue inputs/outputs	——	Check out	√	√
	Physical location	——	√	√	√
	Variable list	——	√	√	√
Diagnostic	Running log	——	Check out	√	√
	Diagnostic data export	——	√	√	√
Put into operation	User tool calibration	——	√	√	√
	User workpiece calibration	——	Check out	√	√
	Co-group calibration	——	Check out	√	√
	External shaft reduction ratio calibration	——	Check out	√	√
	20-point calibration	——	Check out	√	√

	Zero calibration	——	Check out	√	√
	Software Limit Switches	——	Check out	√	√
Information	Version information	——	Check out	√	√
	System Information	——	Check out	√	√
	Robot parameters	——	Check out	Check out	√
Systems	Language Settings	——	/	/	
	Reveal the Teachers	——	/	/	√
	Clearance system	——	/	/	√
	Shut down the system	——	/	/	√
	Reboot	——	/	/	√
	System upgrades	——	Check out	√	√
	Authorisation	——	Check out	√	√
Interface buttons	Activation button	——	Check out	√	√
	Pause button	——	√	√	√
	Uninstall button	——	√	√	√
	Rewind button	——	/	√	√
	Switching between manual and automatic modes	——	/	√	√
	Tool workpiece switching	——	√	√	√
	The way the programme runs	——	/	√	√
	Incremental manual movement setting	——	/	√	√
	Manual magnification adjustment	——	/	√	√
	Automatic magnification adjustment	——	/	√	√
	Select Coordinate System	——	/	√	√

	Programme Control Soft Buttons	——	/	√	√
	Auxiliary Button Operation		/	√	√
	Manual movement operation	——	/	√	√
	One button to return to zero	——	√	√	√
	——	——	/	√	√
Craft kit	Process package management	——		√	√
Programmes After loading	Online Editing	——	Check out	√	√
	Switching Mode	——	/	√	√

9.4 List of external signals

Table of signals in relation to modes of operation:

	Manual T1/T2 mode	Automatic mode	External mode
External Input Signal	Null	Valid (except loader signal iPRG_LOAD)	Efficiently
External Output Signal	Efficiently	Efficiently	Efficiently
Oscillator button	Efficiently	Partially valid	Partially valid

System input signal table:

Signal Name	Instructions	Effective method	range of action
iPRG_START	Start program signal.Starts the loaded user programme running.	Decline in effect along	Automatic mode, external mode
iPRG_PAUSE	Suspend programme signals.Suspends user programme operation.	Decline in effect along	Automatic mode, external mode
iPRG_STOP	Stop programme signal.Stops the user programme from	Decline in effect along	Automatic mode,

	running.		external mode
iPRG_LOAD	Load program signals.Loads the specified user programme.	Rising edge in effect	External mode
iPRG_UNLOAD	Unload programme signal.Uninstalls the program in the ready state.	Decline in effect along	Automatic mode, external mode
iENABLE	System enable signal.	Rising Edge Enable, Set 0 Break Enable	Automatic mode, external mode
iCLEAR_FAULTS	Clear the error signal.	Rising edge in effect	Automatic mode, external mode
iSAFE_SPEED	Safety speed trigger signal.When triggered, it causes the arm to reduce its operating speed.	Decline in effect along	Automatic mode, external mode
iSAFE_DOOR	Safety door trigger signal	Decline in effect along	Automatic mode, external mode
iSHARED_EN[0]	Shared Area [0] Enable Switch	Rising Edge Enable, Set 0 Break Enable	Manual mode, automatic mode, external mode
iSHARED_EN[1]	Shared Area [1] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[2]	Shared Area [2] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[3]	Shared Area [3] Enable Switch	Enable on rising edge, set to	As above

		0 to close	
iSHARED_EN[4]	Shared Area [4] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[5]	Shared Area [5] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[6]	Shared Area [6] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[7]	Shared Area [7] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[8]	Shared Area [8] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[9]	Shared Area [9] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[10]	Shared Area [10] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[11]	Shared Area [11] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[12]	Shared Area [12] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[13]	Shared Area [13] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[14]	Shared Area [14] Enable Switch	Enable on rising edge, set to 0 to close	As above
iSHARED_EN[15]	Shared Area [15] Enable Switch	Enable on rising edge, set to 0 to close	As above

System output signal table:

Signal Name	Instructions	Note	Scope of action
oROBOT_READY	Robot ready signal.This signal is output when it is also satisfied that the system initialisation is complete, the user program is in the loaded state, and it is enabled.	The signal will not be output while the programme is running	All modes

oROBOT_STANDBY	Robotic arm standby signal: the signal is output when the system initialisation is completed, the robotic arm is not enabled, the main program is not loaded and there is no alarm.		All modes
oEMERGENCY_STOP	Emergency stop circuit on/off status signal	This signal is ON when the emergency stop circuit is ON and OFF when the emergency stop button is pressed.	As above
oFAULTS	Error.		As above
oENABLE_STATE	Enable state.		As above
oPRG_UNLOAD	The user program is in an unloaded state.	At the same moment in time, there is one and only one signal output from these signals	As above
oPRG_READY	User program loaded status.		As above
oPRG_RUNNING	User program running status.		As above
oPRG_ERR	User program alarm status.		As above
oPRG_PAUSE	The user programme is suspended.		As above
ols_Moving	The robot is in motion.		As above
oMANUAL_MODE	The system is in manual mode.	At the same moment in time, there is one and only one signal output from these signals	As above
oAUTO_MODE	The system is in automatic mode.		As above
oEXT_MODE	The system is in external mode.		As above
oHOME	Currently at zero POINT		As above
oMD_ENABLED	MoDBuS Enable Switch	**Reserved, not**	As above

		available at this time	
oMD_CONN	ModBus connection status	**Reserved, not available at this time**	As above
oCOL_STATE	Collision detection enable state		As above
oCOLLA_MODE	The system is in a collaborative mode		As above
oREF[0]	Reference point [0]		As above
oREF[1]	Reference point [1]		As above
oREF[2]	Reference point [2]		As above
oREF[3]	Reference point [3]		As above
oREF[4]	Reference point [4]		As above
oREF[5]	Reference point [5]		As above
oREF[6]	Reference point [6]		As above
oREF[7]	Reference point [7]		As above
oAREA_OUT[0]	Area [0] output signal		As above
oAREA_OUT[1]	Area [1] output signal		As above
oAREA_OUT[2]	Area [2] output signal		As above
oAREA_OUT[3]	Area [3] output signal		As above
oAREA_OUT[4]	Area [4] output signal		As above
oAREA_OUT[5]	Area [5] output signal		As above
oAREA_OUT[6]	Area [6] output signal		As above
oAREA_OUT[7]	Area [7] output signal		As above
oAREA_OUT[8]	Area [8] output signal		As above
oAREA_OUT[9]	Area [9] output signal		As above
oAREA_OUT[10]	Area [10] output signal		As above
oAREA_OUT[11]	Area [11] output signal		As above
oAREA_OUT[12]	Area [12] output signal		As above
oAREA_OUT[13]	Area [13] output signal		As above
oAREA_OUT[14]	Area [14] output signal		As above
oAREA_OUT[15]	Area [15] output signal		As above
WELD_GUN_COLLISION	Torch collision alarm	/	As above

WELD_ALARM	Welder Alarm	/	As above
WELD_MACHINE_ALARM	Welding equipment alarms	/	As above

9.5 SOCKET error codes

Serial number	Error code	Error code resolution
1	-1	Handle does not exist
2	-2	Using connectionless handles
3	-3	Illegal ports
4	-4	Server Listening Failure 1
5	-5	Server Listening Failure 2
6	-6	Client connection failure
7	-7	The client has been connected
8	-8	Type Illegal
9	-9	Listen timeout
10	-100	The network is down.
11	-101	Network unreachable
12	-102	Network disconnected due to reset
13	-103	Software Causes Connection Termination
14	-104	The connection was reset by the other party.
15	-105	No available cache space
16	-106	Transmission Endpoint Connected
17	-107	Transmission endpoint not connected
18	-108	Cannot be sent again after the transmission endpoint is closed
19	-109	Too many references: can't splice
20	-110	Connection timeout
21	-111	Connection denied
22	-112	The mainframe is shut down.
23	-113	Unable to route to host
24	-114	The operation is already in process
25	-115	In-process operations
26	-125	Connection timeout

9.6 System signal occupancy table

Serial number	Flag position	DO No.	Instructions
1	GC_oROBOT_READY	100	Robot readiness
2	GC_oFAULTS	101	Robot Alarm Status
3	GC_oENABLE_STATE	102	Robot Enable State
4	GC_oPRG_UNLOAD	103	Program uninstallation status
5	GC_oPRG_READY	104	Program readiness
6	GC_oPRG_RUNNING	105	Program operating status
7	GC_oPRG_ERR	106	Program error status
8	GC_oPRG_PAUSE	107	Suspended state of the programme
9	GC_oIS_MOVING	108	Robot Motion Status
10	GC_oMANUAL_MODE	109	Manual Mode Status
11	GC_oAUTO_MODE	110	Auto Mode Status
12	GC_ oEXT_MODE	111	External Mode Status
13	GC_oREF_0	112	Reference 0 state
14	GC_oREF_1	113	Reference 1 state
15	GC_oREF_2	114	Reference 2 state

Serial number	Occupied functions	DO No.	Instructions
1	Coordination	0	Emergency stop
2	CO610 Strip Driver	16	Light Strip Drive Low
3		17	Light Strip Drive High
4	Socket communication	200	Communication 1 Communication Success Status
5		201	Communicating a Communicating Waiting for Connection Status
6		202	Communication 1 Communication Success Status
7		203	Communication 2 Communication Waiting for Connection Status
8		204	Communication 3 Communication Success Status

9		205	Communication 2 Communication Waiting for Connection Status
10	Codesys	511	Occupancy rate of a heartbeat

Serial number	Flag position	DI number	Instructions
1	GC_PRG_START_N	100	Triggering procedure
2	GC_PRG_PAUSE_N	101	Suspension of proceedings
3	GC_PRG_STOP_N	102	Stopping procedure
4	GC_PRG_LOAD_P	103	Loading Procedures
5	GC_PRG_UNLOAD_N	104	Uninstaller
6	GC_ENABLE	105	Enabling robots
7	GC_CLEAR_FAULTS_P	106	Clear Alarms
8	/	Default 129(virtual)/ 128(real)	Drag Trigger Input
9	/	Default 128(virtual)/ 129(real)	Drag-and-drop instructional trigger inputs

Serial number	Occupied functions	DI number	Instructions
1	Drag	300（virtual）	Default Drag Trigger Input (Drag Button hardwired depending on model)
2		301（virtual）	Default Teach-In Trigger Input (Drag Button hardwired depending on model)
3	CO605 Drag	16（real） Default value	Drag the button to hardwire the trigger input (this IO is the servo's virtual IO, in descending order depending on the number of physical IOs configured)
4		17（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo

			virtual IO, in descending order depending on the number of physical IOs configured)
5		18（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo virtual IO, in descending order depending on the number of physical IOs configured)
6		19（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo virtual IO, in descending order depending on the number of physical IOs configured)
7	CO610 Drag	16（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo virtual IO, in descending order depending on the number of physical IOs configured)
8		17（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo virtual IO, in descending order depending on the number of physical IOs configured)
9		18（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo virtual IO, in descending order depending on the number of physical IOs configured)
10		19（real） Default value	Teach-In Button Hardwired Trigger Input (this IO is a servo virtual IO, in descending order depending on the number of physical IOs configured)

11	CR605/CR607 Drag	6（real）	Drag button hardwired trigger input
12		7（real）	Teach-In Button Hardwired Trigger Input
13	CO610	0	Emergency stop

Guangzhou Auctech Automation Technology Ltd

Add: BuildingE1, Design City, No.7, Hexian North Street, Helong Street, Baiyun District, Guangzhou City, CHINA

Tel:+862084898493

E-mail:info@auctech.com.cn

Web:www.auctech.com.cn