Collaborative robots

Basic training

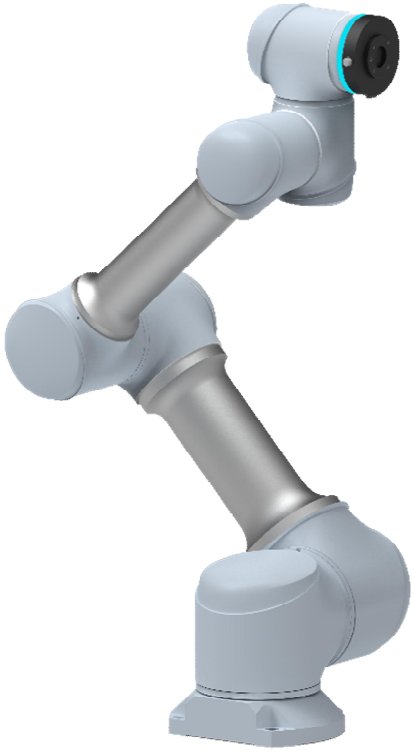
Software Version V3.0.0

START

Part I

Basic introduction to the software

1. Singularity
2. Quick start
3. User level
4. Engineering concepts
5. Overview of the interface
6. Log introduction
7. Multi-terminal connection



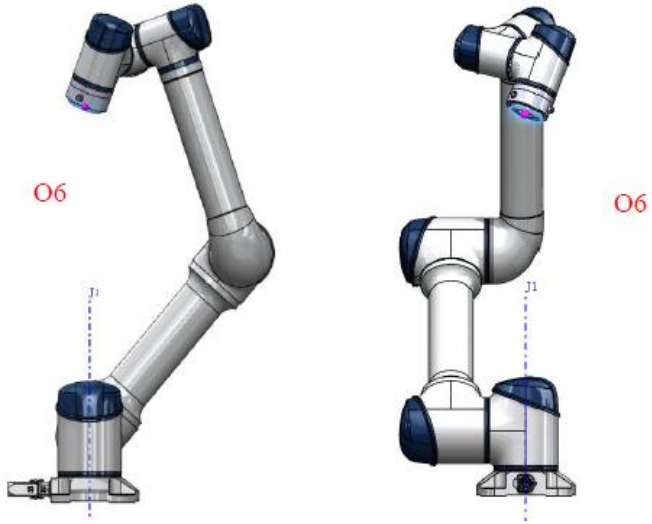
Lecture 1

Singularity

1.1 Shoulder singularity

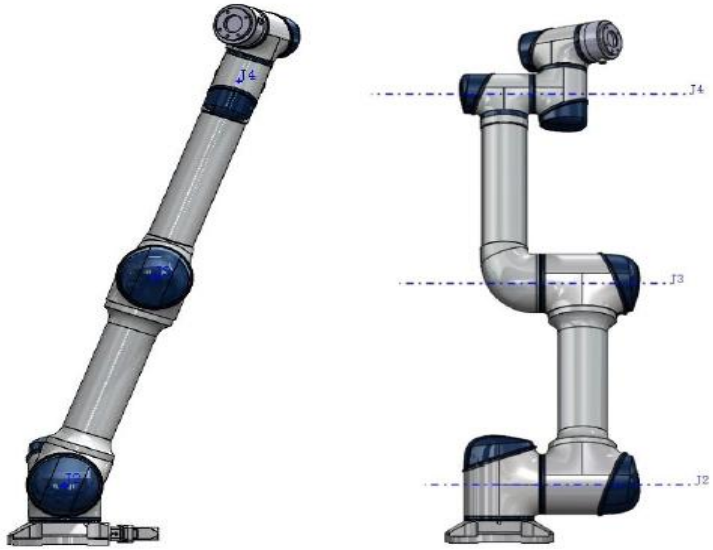
1.For the GCR series configuration, there are shoulder singularities, elbow singularities, and wrist singularities.

2.When the center of the wrist joint O6 is on the axis of one joint J1, the shoulder is singular at this time, resulting in the unsolved joint of 1. O6 is also affected by singularity when it is located very close to J1, where moving the end can cause joint 1 to overdrive. Refer to the following diagram for the singular posture of the adjacent shoulder:



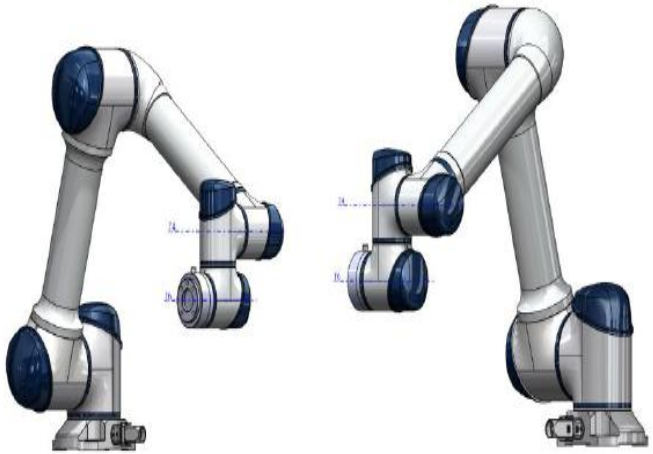
1.2 Elbow singular

When the axes of the second, third and fourth joints J2, J3 and J4 are coplanar, there is no solution for the two joints. Simplely, when the joint 3 are close to 0 degrees and in a near singular position, moving the end may cause joint 2, 3, and 4 to overdrive. Refer to the following diagram for the singular pose of the near elbow:



1.3 Wrist singularity

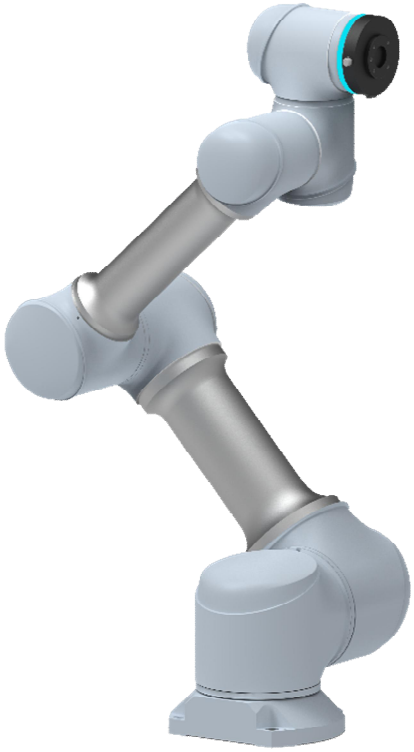
When the joint 5 is 0 degrees, the joint 6 is not solved at this time, resulting in the singularity of the wrist. When the joint 5 are close to 0 degrees, it is a singular posture near the wrist, and moving the end at this time may cause the joint 4, 5, and 6 to overspeed. Refer to the diagram below:



1.4 Robot singularity stall risk

The robot will automatically slow down when performing motion planning (straight line, arc, etc., excluding joint movement) near the singularity, and should avoid the singularity or pass through the singularity with joint motion during teaching.

When the robot runs to or near the above-mentioned singularities (shoulder singularities, elbow singularities, wrist singularities), the planning motion based on Cartesian coordinates cannot be correctly inversely solved into the joint motion of each axis, and the motion planning will not be carried out correctly, and the joint jog motion or move j motion command can be adopted.



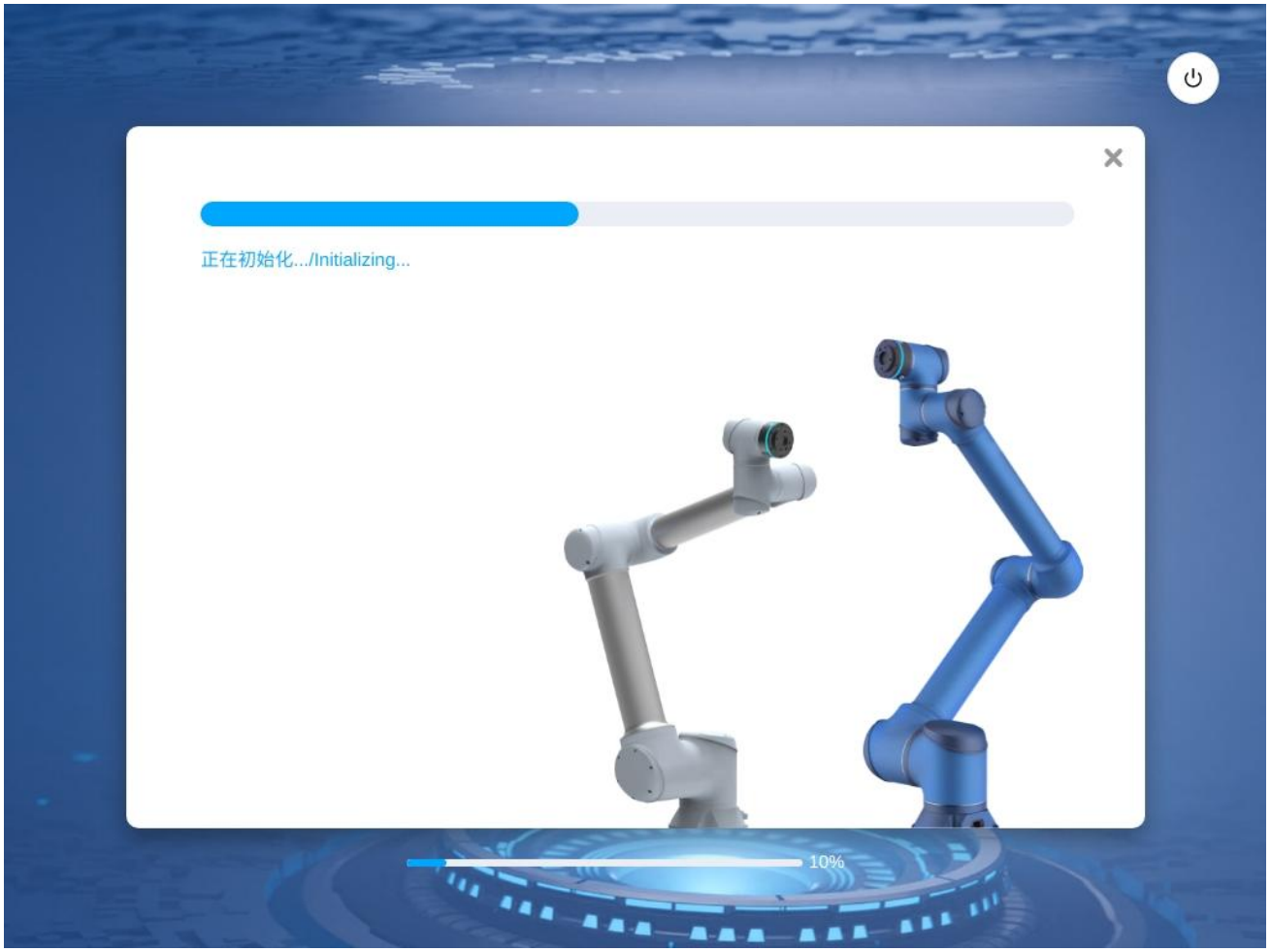
Lecture 2

Quick start

2.1 Turn on for the first time

1. By default, the boot system displays "Welcome to the AUCTECH collaborative robot" to enter the "Start" interface

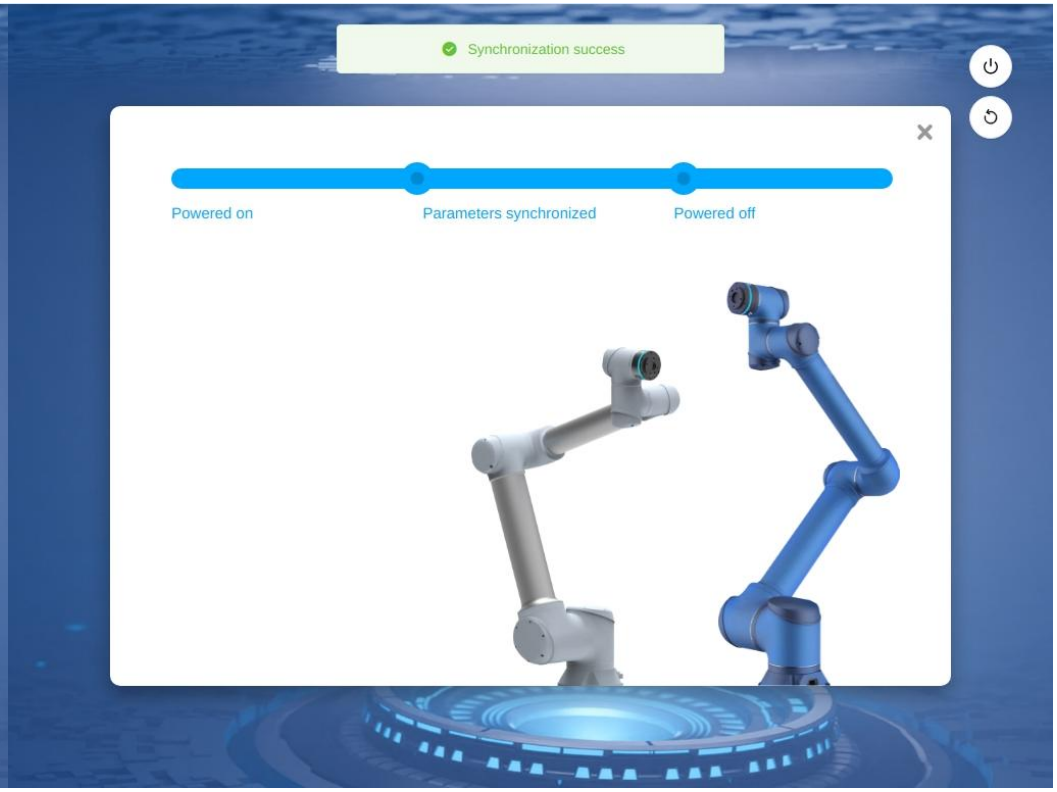
2.1 Turn on for the first time



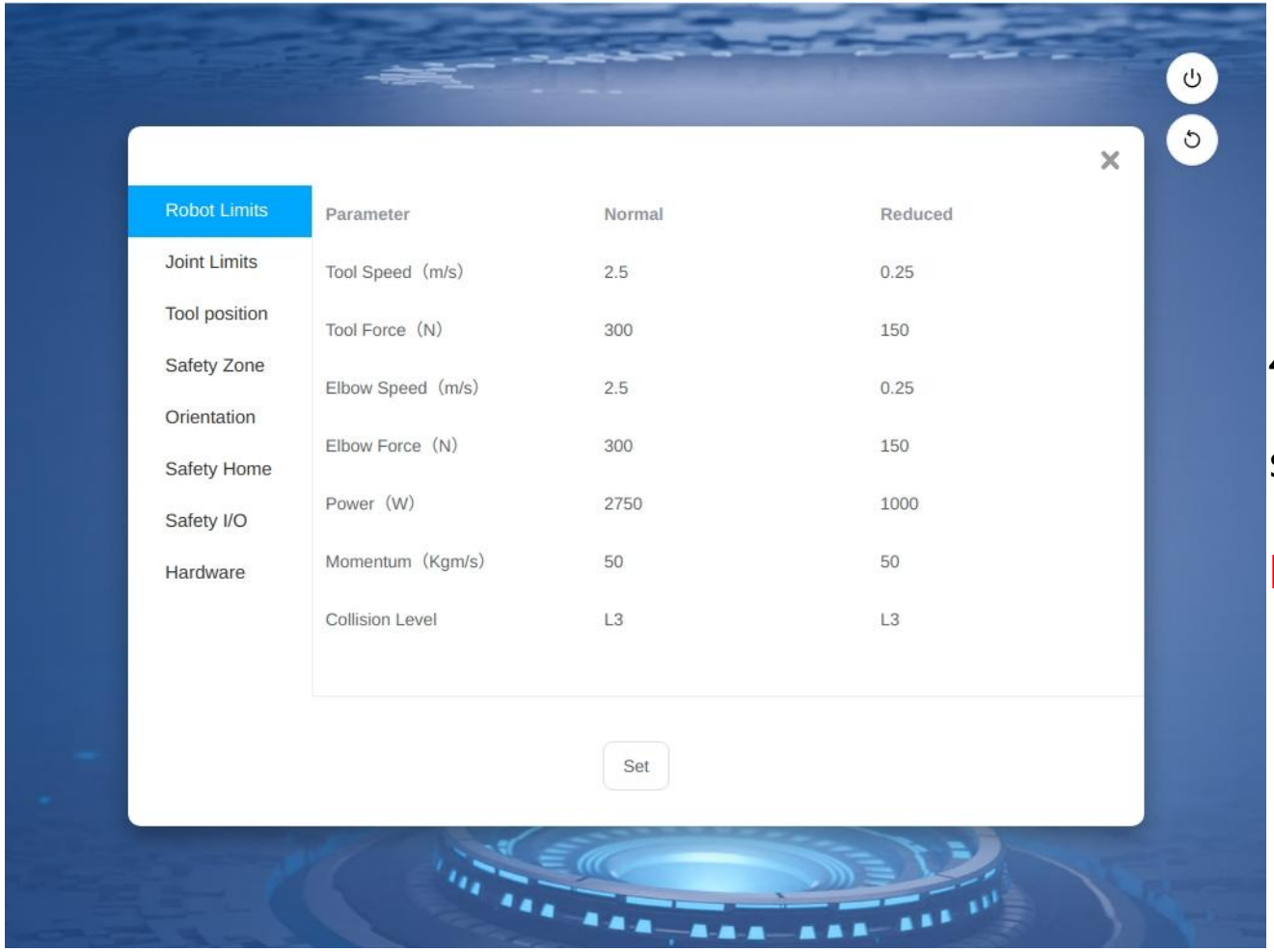
2. Click the "Start" button to initialize the robot. The progress bar of "Initializing..." is displayed, and "Initialization Completed" is displayed after initialization is completed.

2.1 Turn on for the first time

3. The initialization of the robot is completed, and the robot parameter synchronization page is entered. Click "Synchronize Parameters", the robot will power up, parameter synchronization, and power off operations.



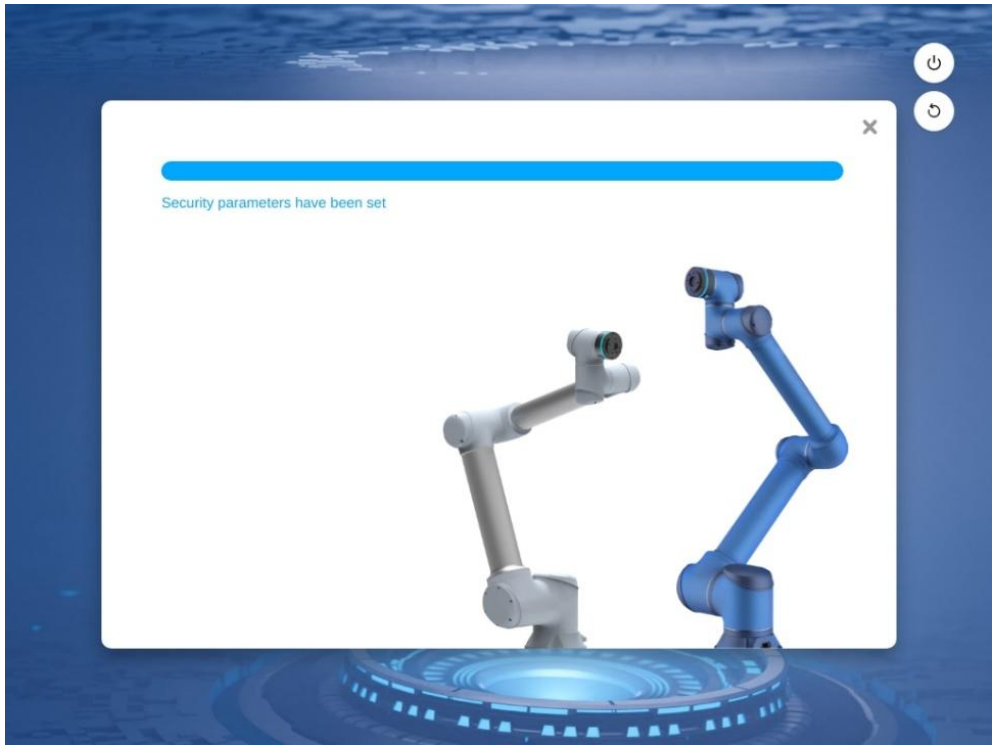
2.1 Turn on for the first time



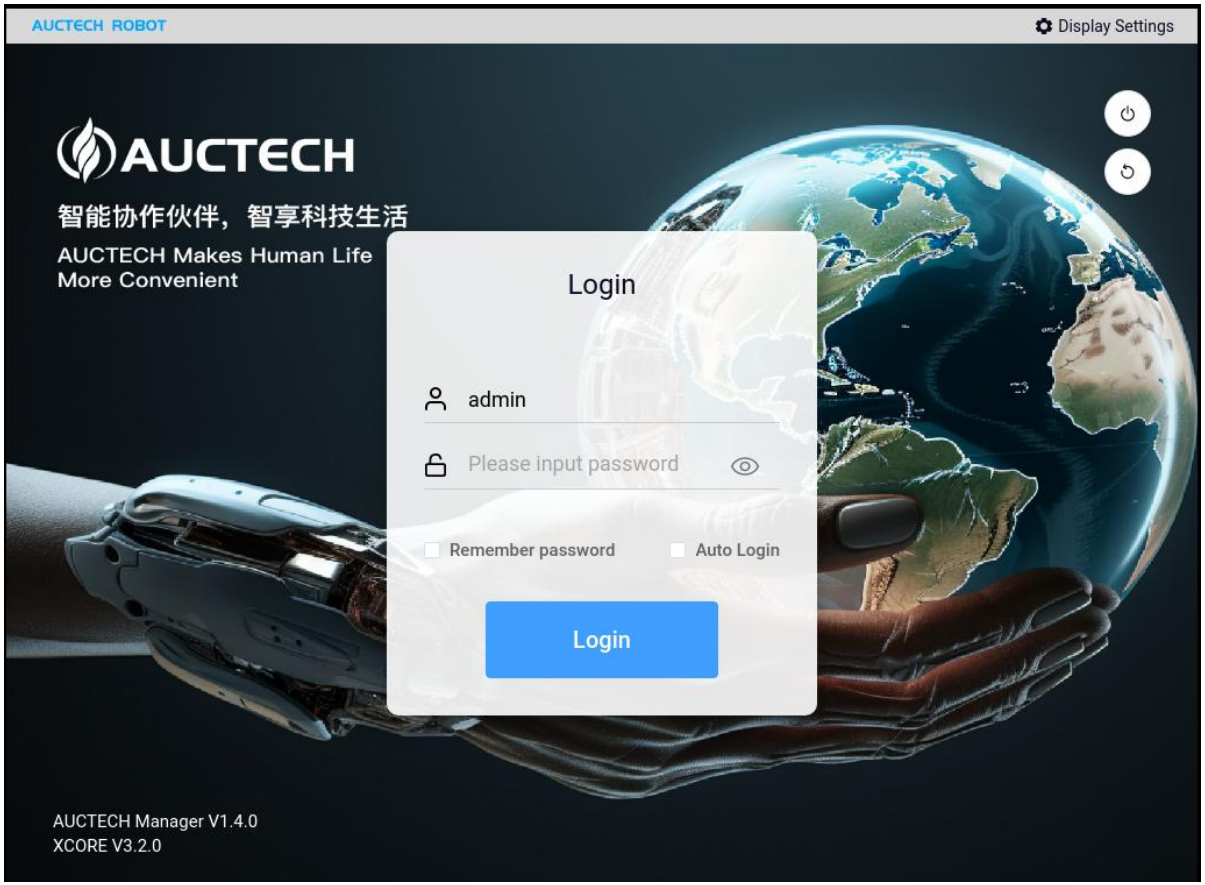
4. After the prompt synchronization is successful, enter the security parameter setting page.

2.1 Turn on for the first time

5. After confirming that the safety parameters are normal, click "Set Safety Parameters", and after the safety parameters are set, the initialization of the robot is completed, and the control cabinet is restarted according to the prompts.

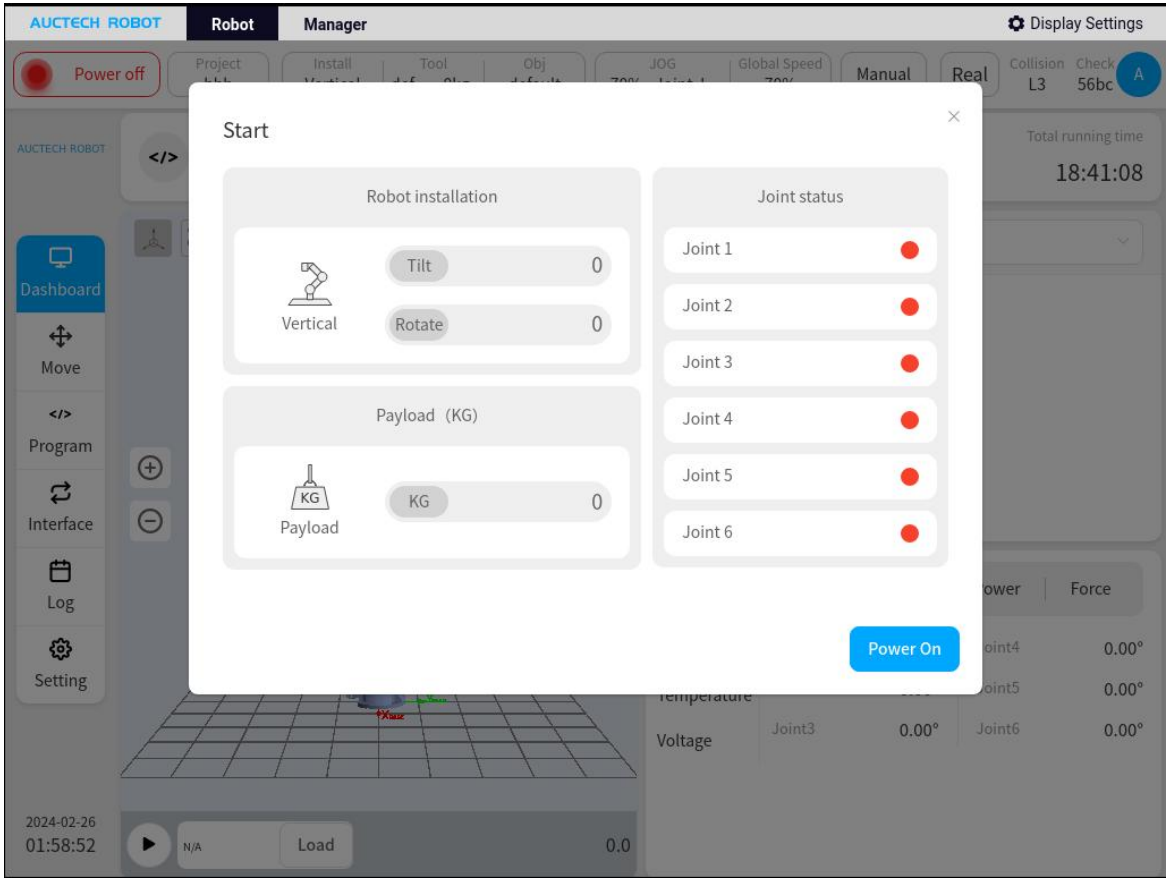


2.2 Log in



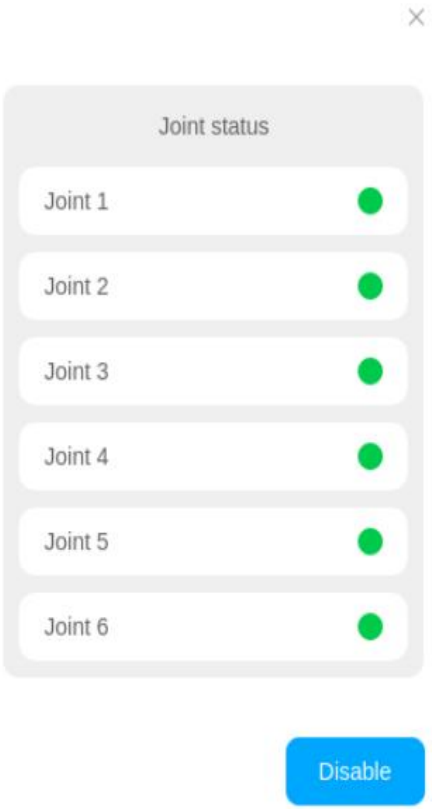
1. After connecting the teach pendant and turning it on, enter the login interface, enter the username and password, and click the "Login" button to log in successfully.
2. At the top right of the login interface is the shutdown button, and in the middle is the user input username and password area.
3. AUCTECH COBOT has two accounts out of the factory, one default account is default, one administrator account is admin, and the password is 123.

2.3 The bot starts



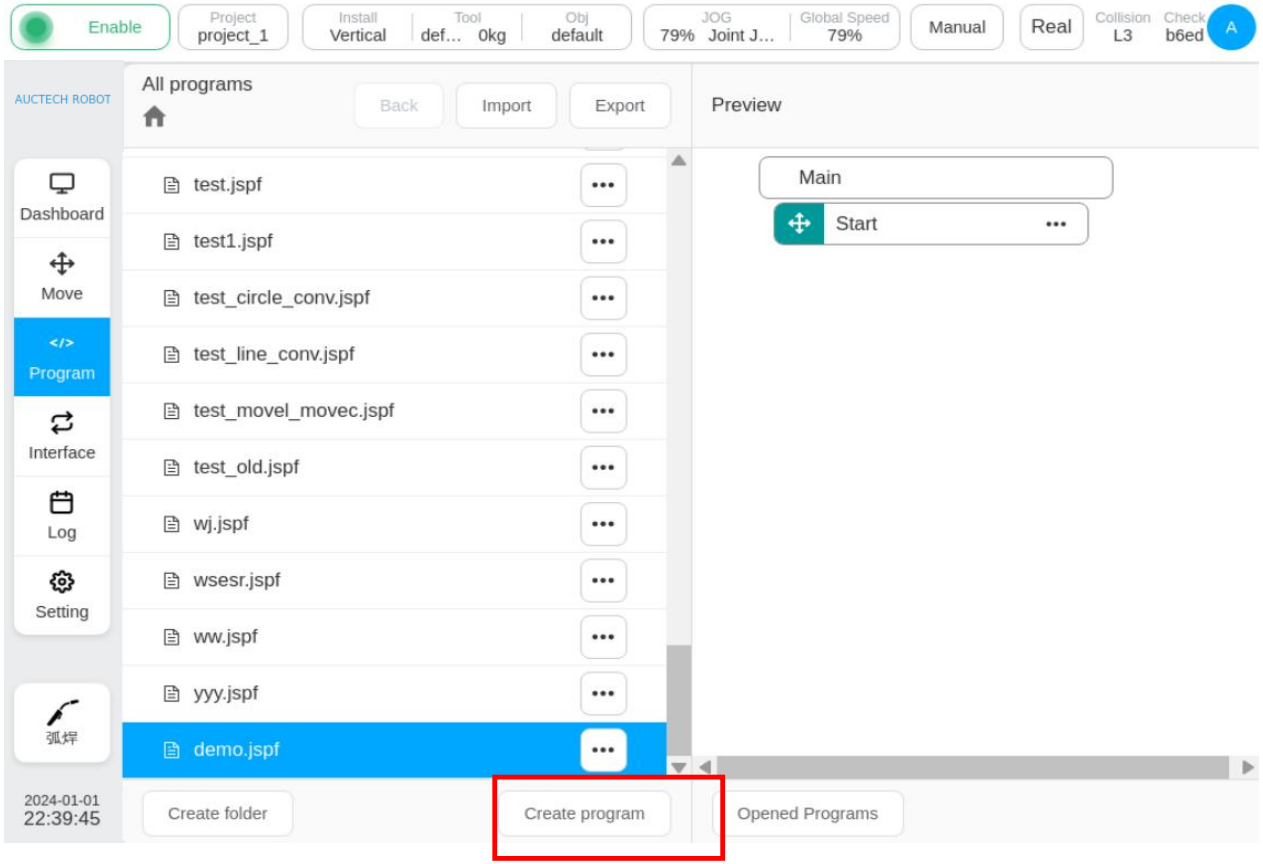
Startup interface (robot is not powered on)

The robot is powered on and not enabled



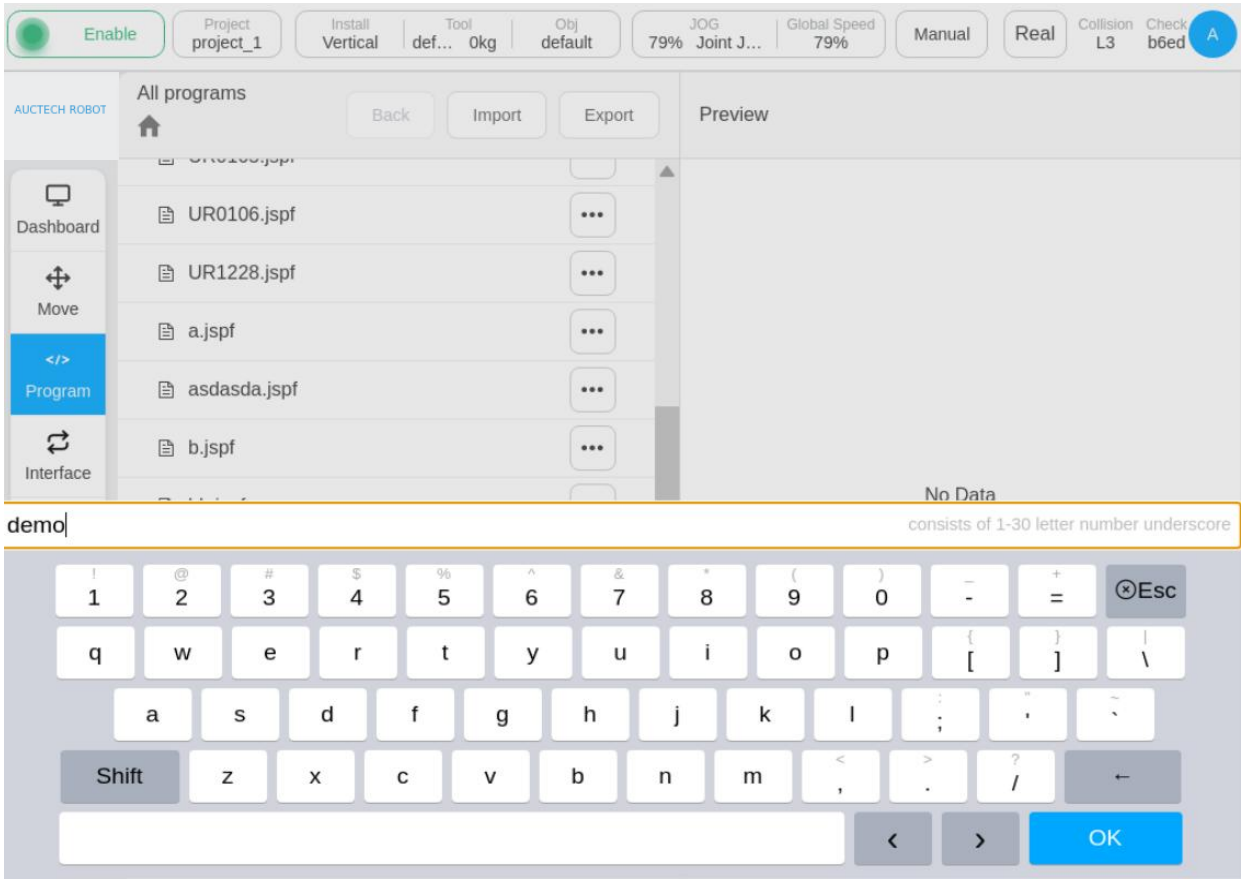
The enabled state of the bot

2.4 Create a program



Note: The program file suffix is .jspf

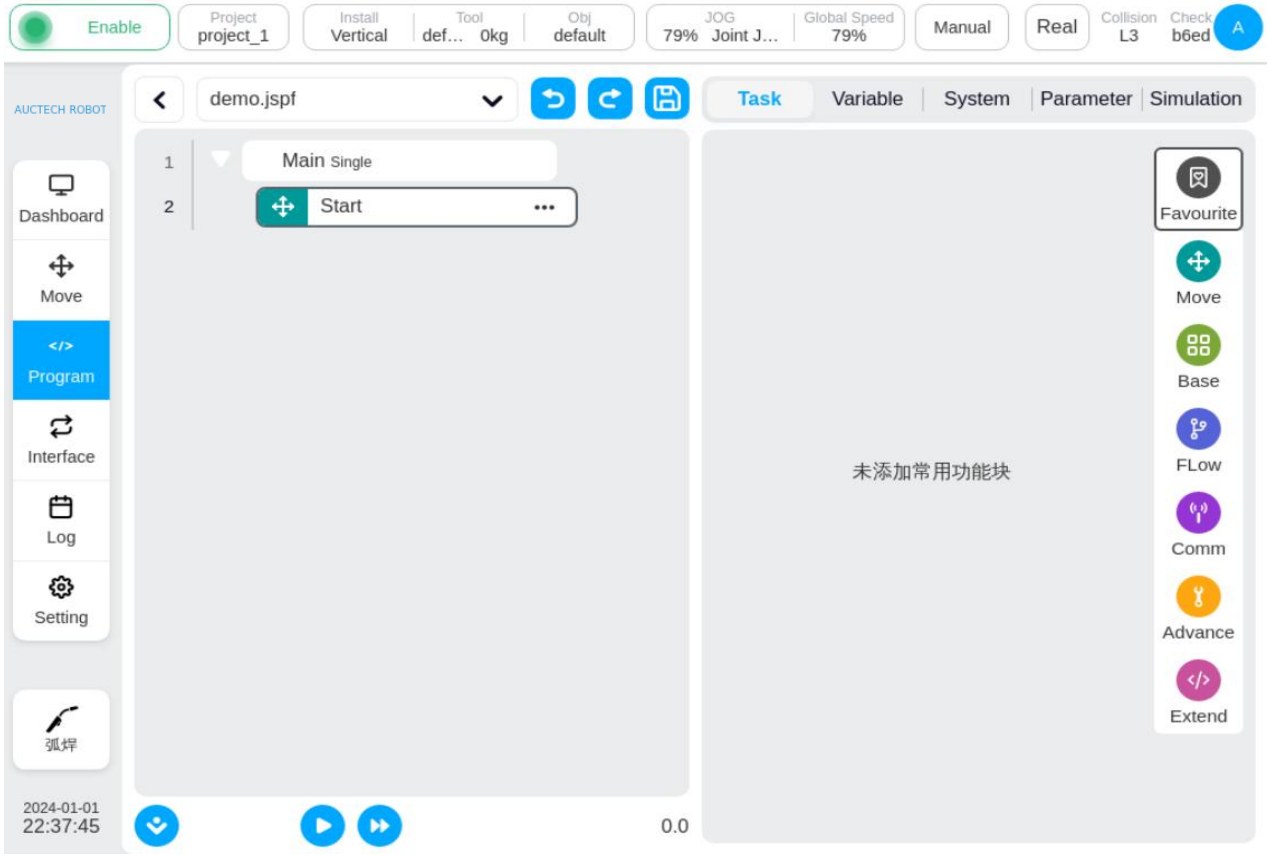
2.4 Create a program



Click "Programs" on the left navigation bar, switch to the program page, click "New Program" at the bottom, and enter the program name: demo. Click "OK" to create a new program demo.jspf and enter the programming page.

Note: The naming convention of program names is composed of 1-30 digits, letters, and underscores

2.4 Create a program



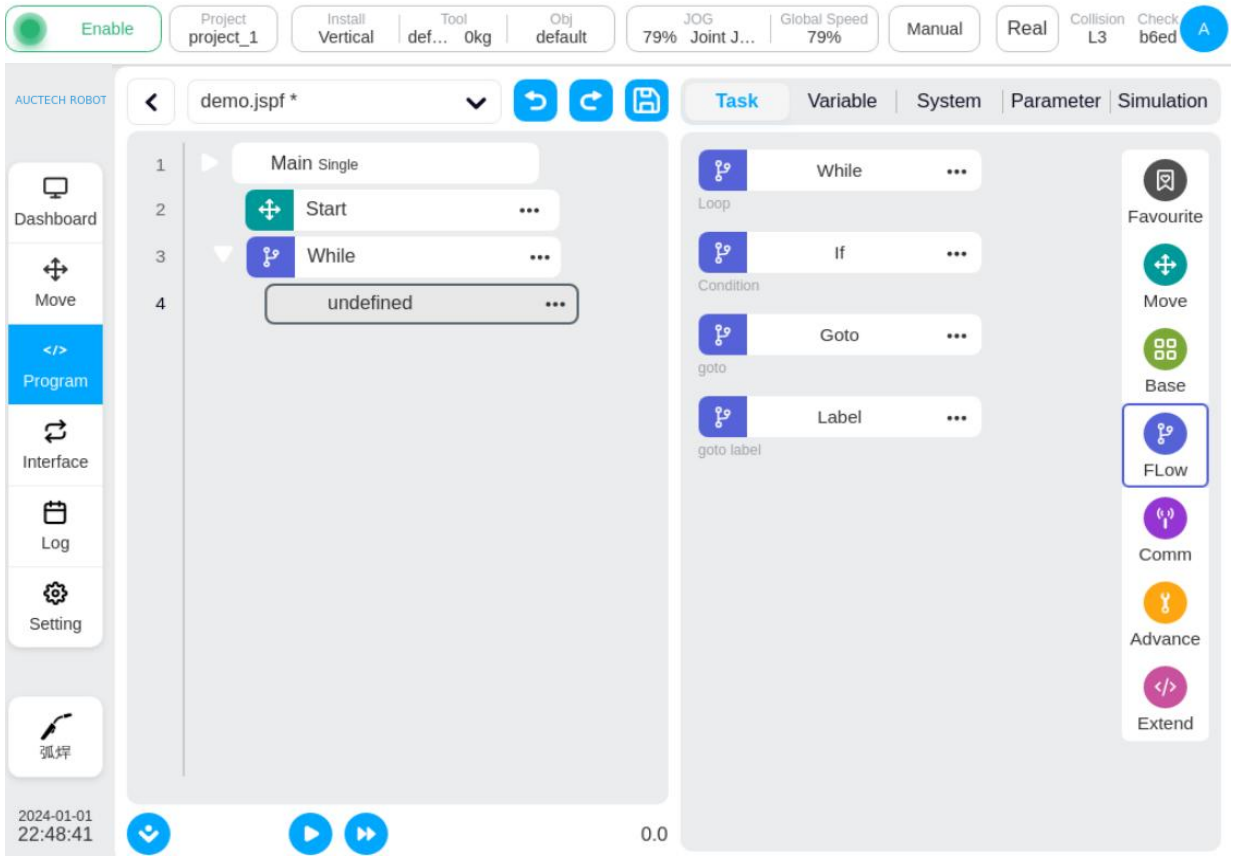
By default, the demo program contains a Start function block to set the starting point of the robot, double-click Start, or click "Parameters" on the right to view and set the starting point.

2.5 Exemple :Write a program

Now let's implement such a feature:

The robotic arm moves to the preparation point, moves down to a working point, waits for 2s, set the digital output port 1 to high, waits for 1s, set the digital output port 1 to low , and the process is cycled 3 times.

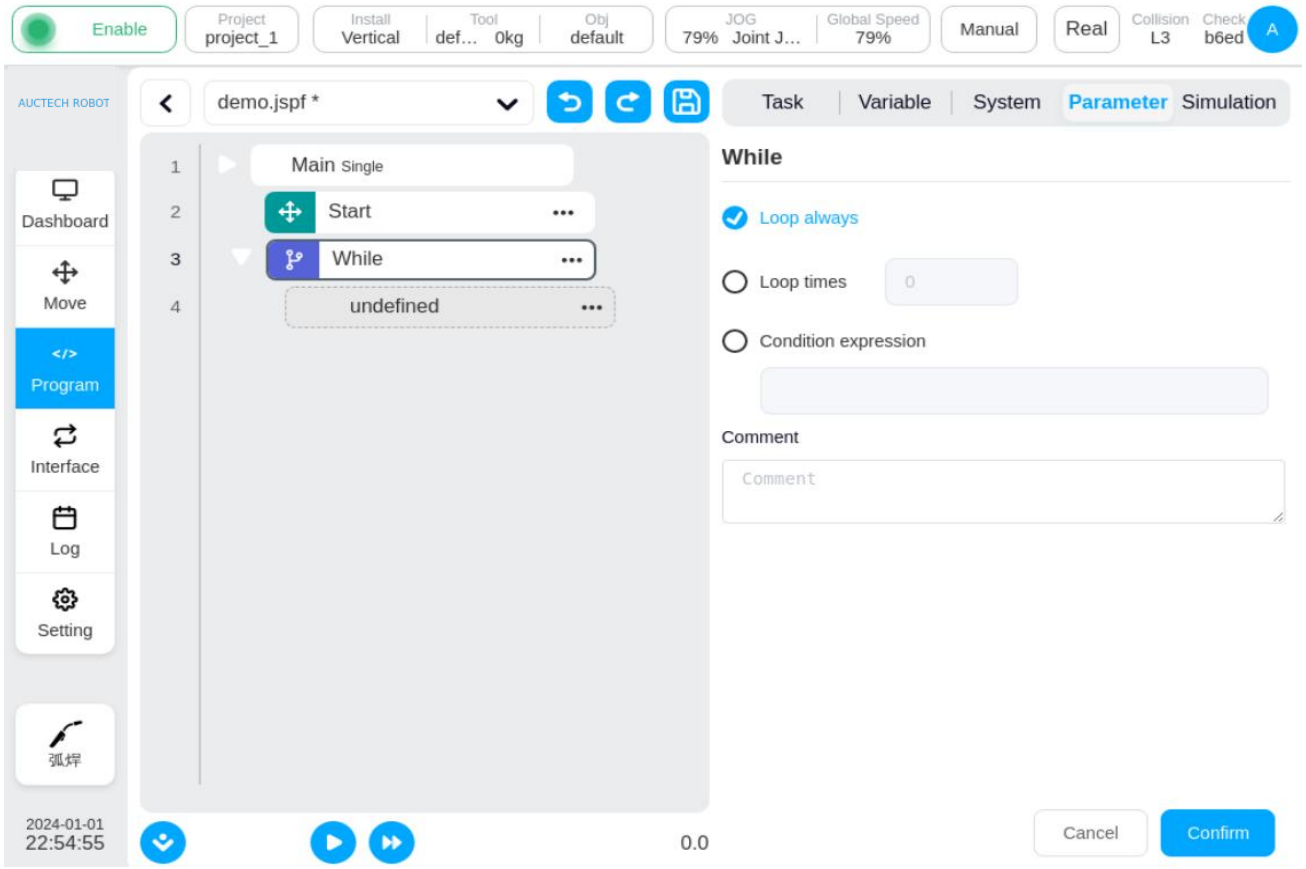
2.5 Write a program



In the “Task" tab - “FLow", find the While function block, which can implement a circular process, hold it down and drag it into the program tree on the left, and release it to add the While function block to the program tree.

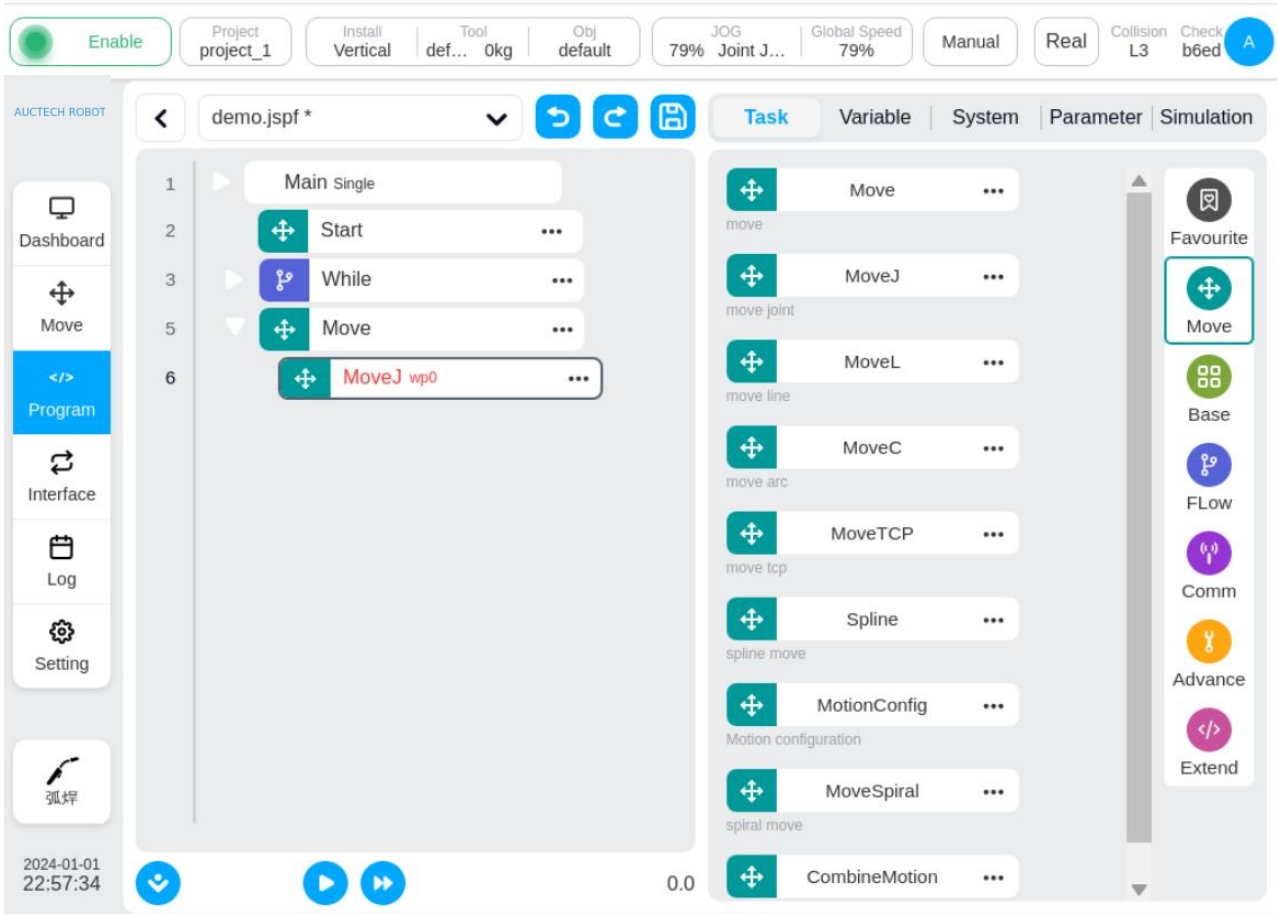
Double-click to add a function block as well.

2.5 Write a program



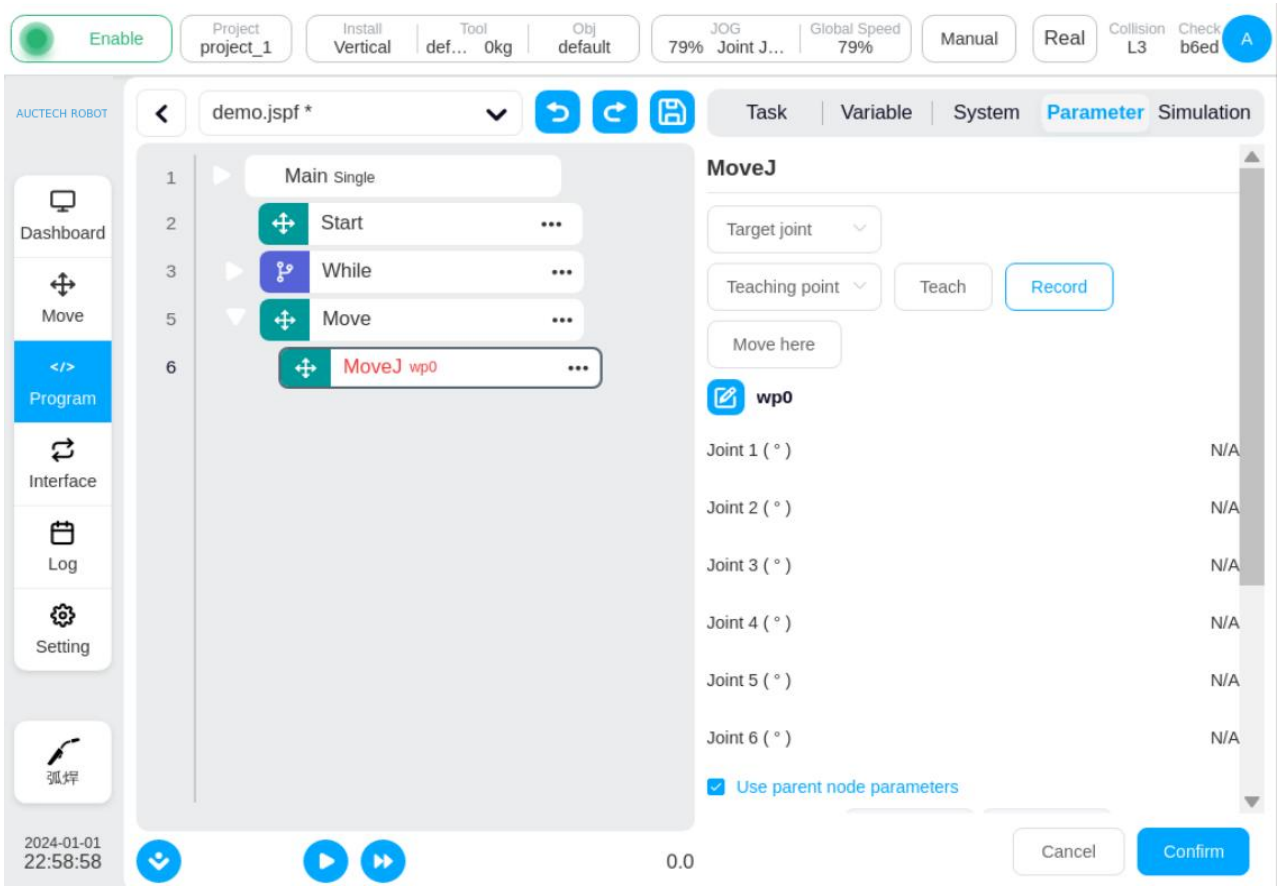
Double-click the While function block to configure its parameters, select Number of Cycles, and enter 3. Indicates that all program segments under the Wh AUCTECH ROBOT block will be execut o three times.

2.5 Write a program



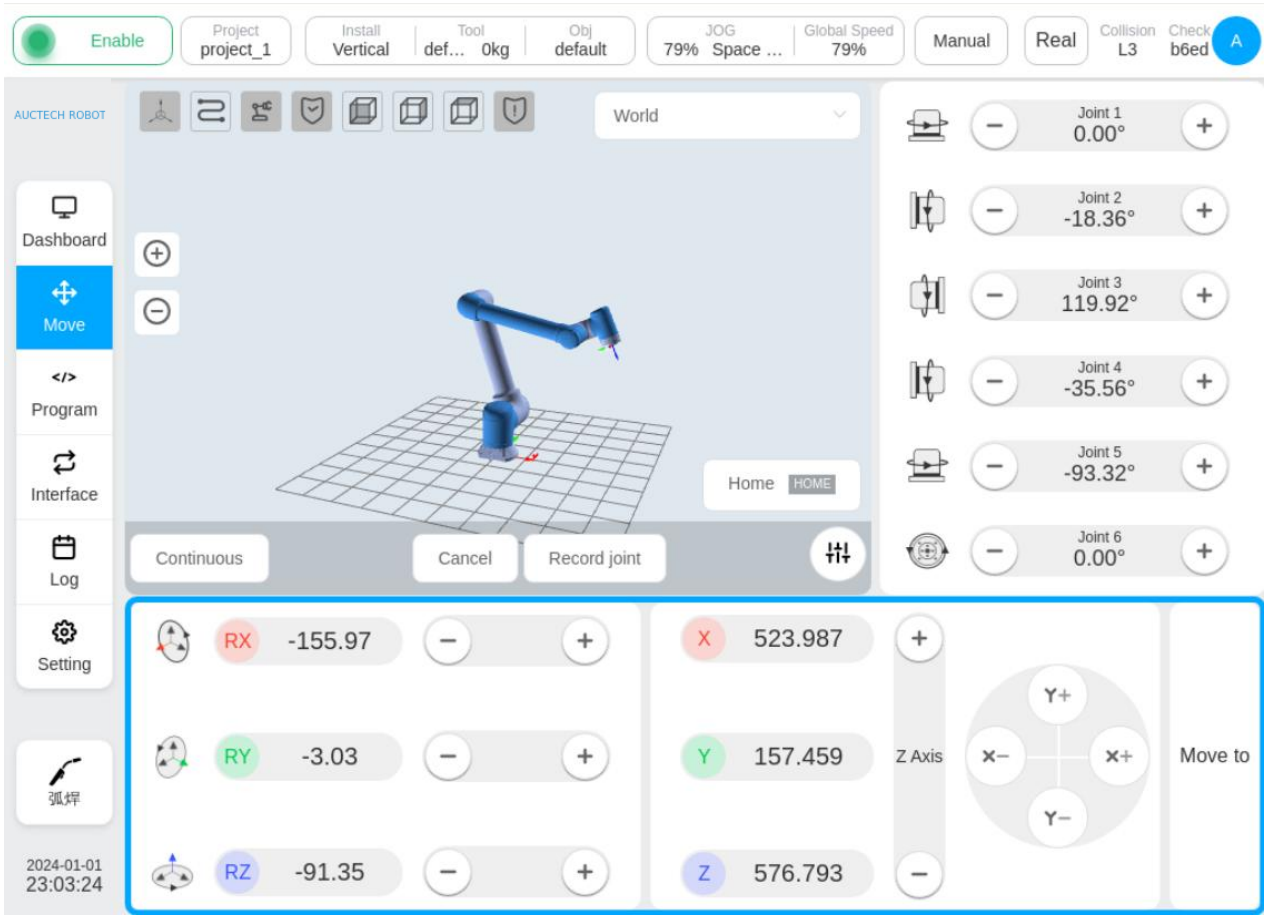
In the “Task" tab - "Move" - find the MoveJ function block and drag it into the While-sub-block in the program tree.

2.5 Write a program



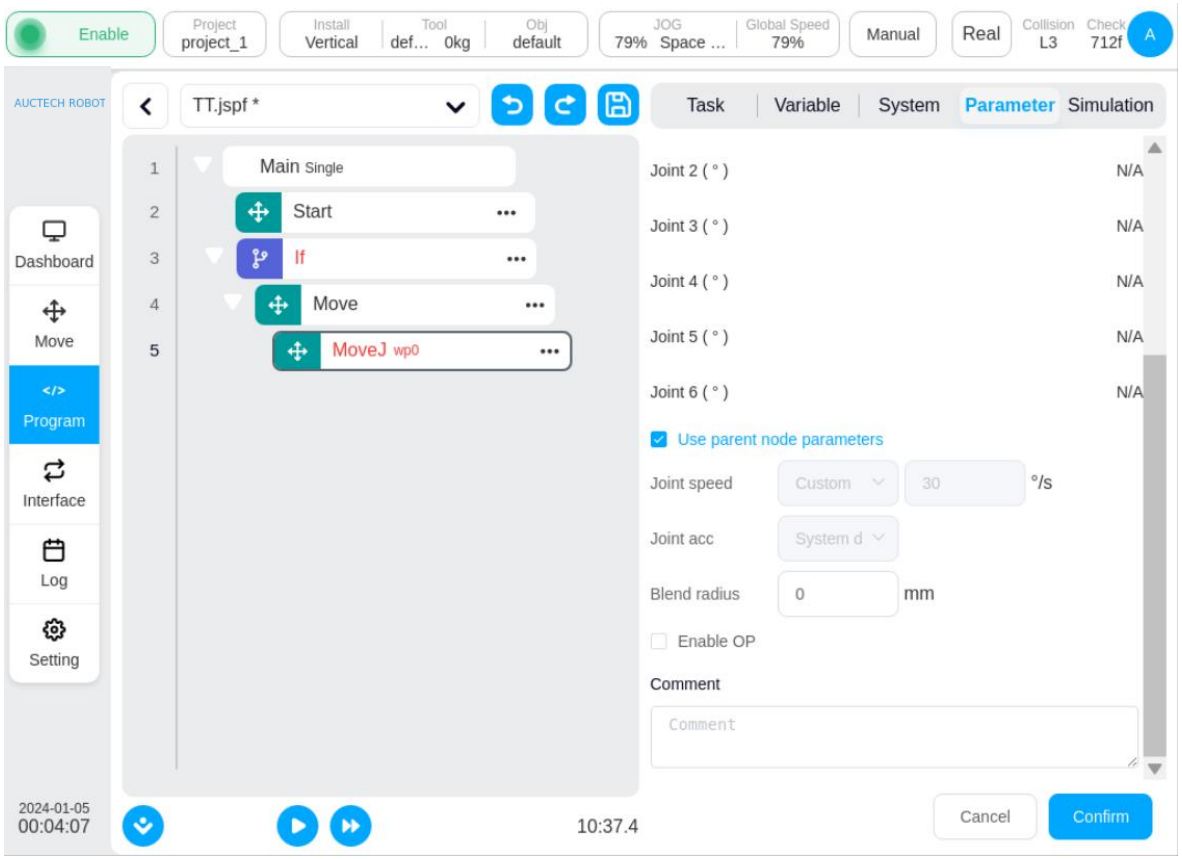
Double-click the MoveJ function block to configure its parameters. Click the "Select Point" button to switch to the point where the robot is taught on the mobile interface.

2.5 Write a program



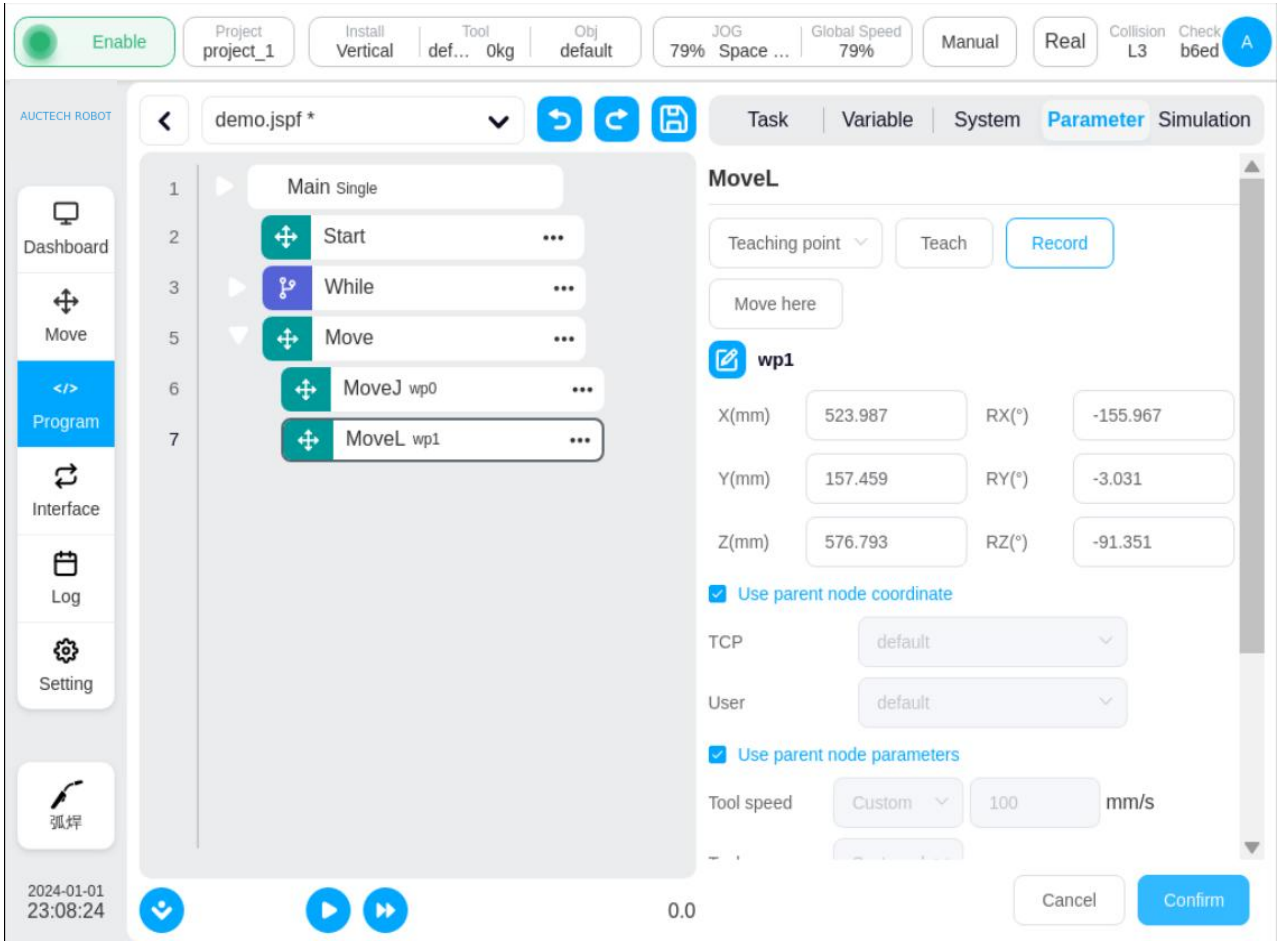
After the point is set, click "Record Joint" to record the current robot posture.

2.5 Write a program



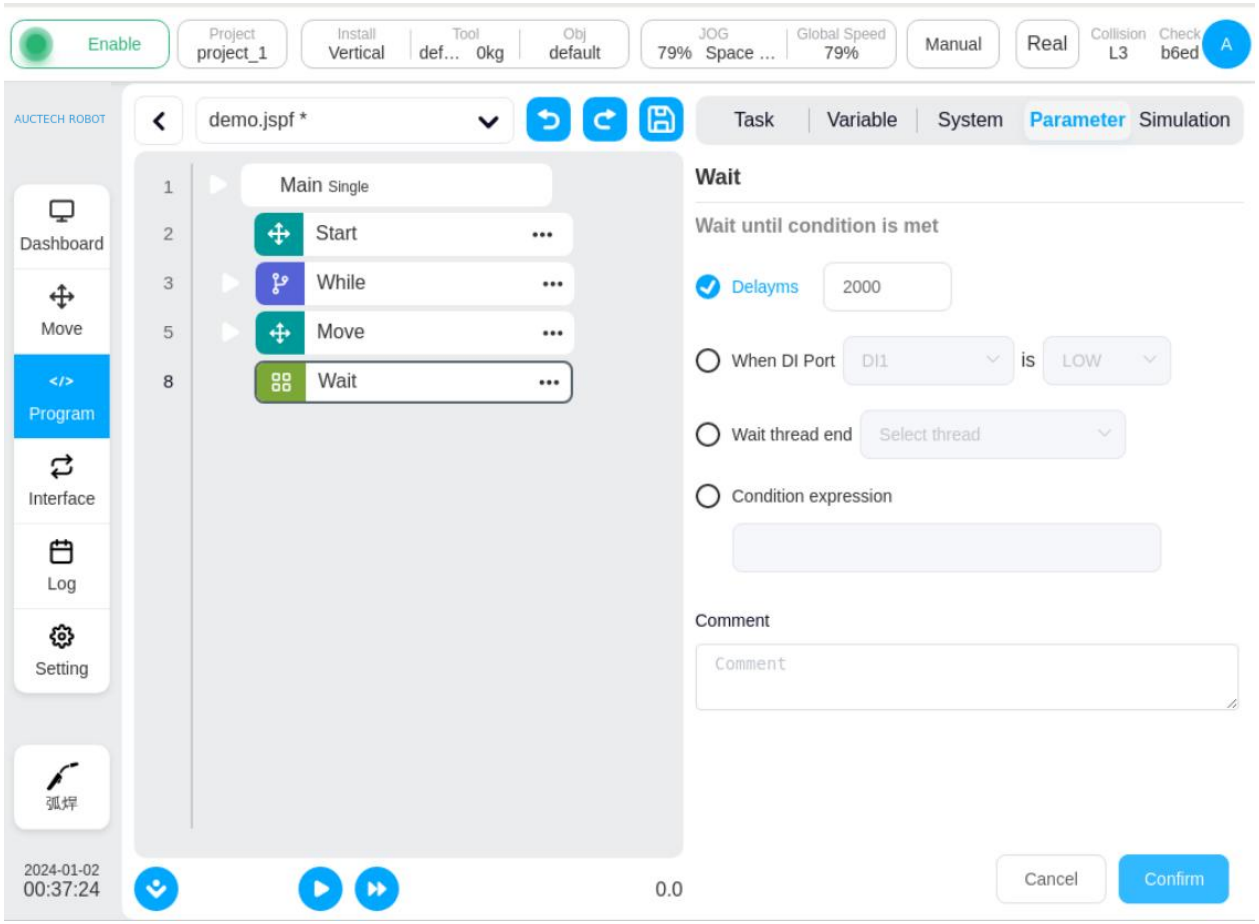
In the parameter configuration area of the MoveJ function block, you can see the set point information. You can further set the angular velocity and angular acceleration of the joint for the movement. Click the “Confirm" button below to save the parameters of the function block.

2.5 Write a program



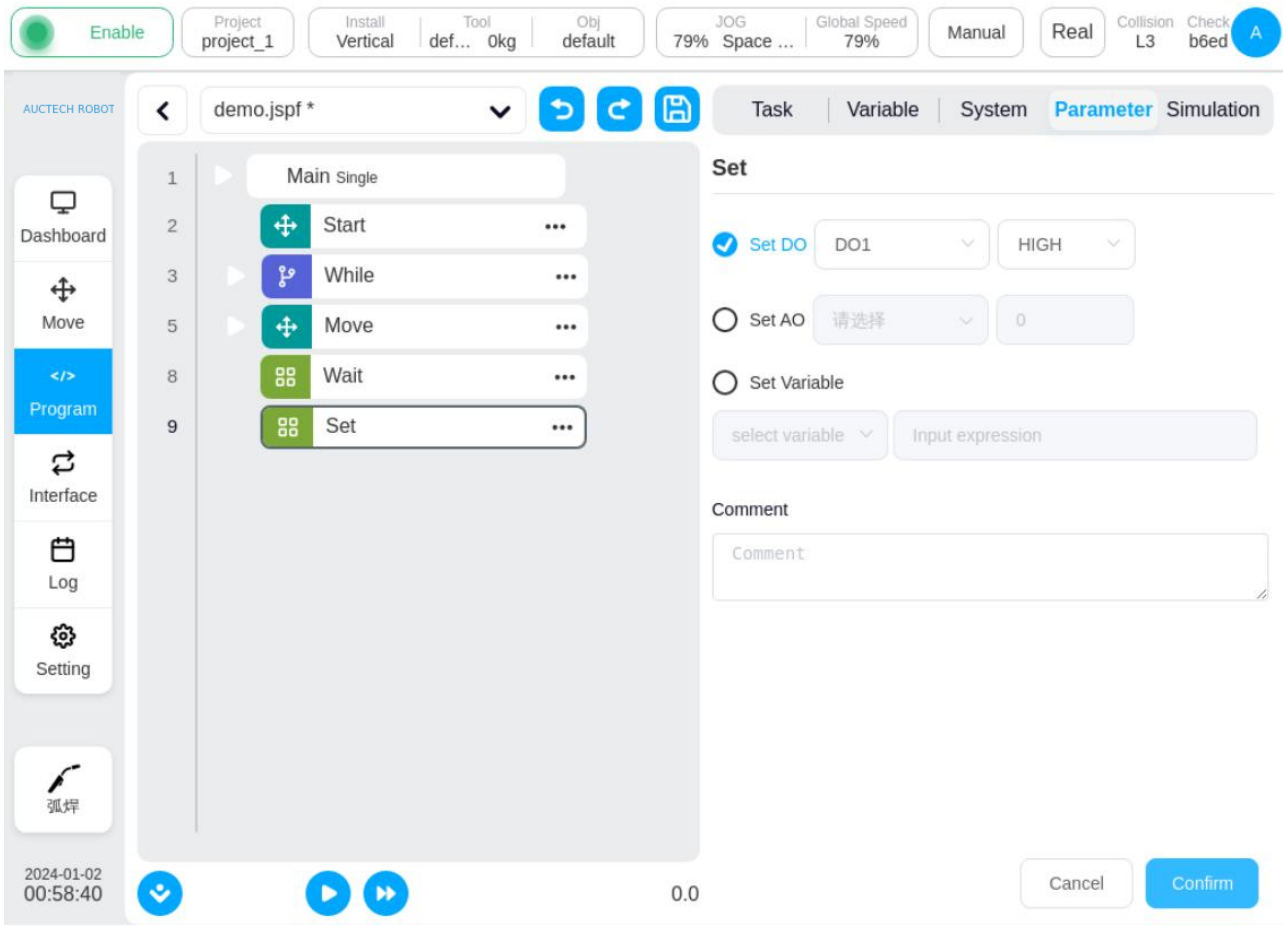
In the same way, add the MoveL function block to the bottom of the MoveJ function block and set the point.

2.5 Write a program



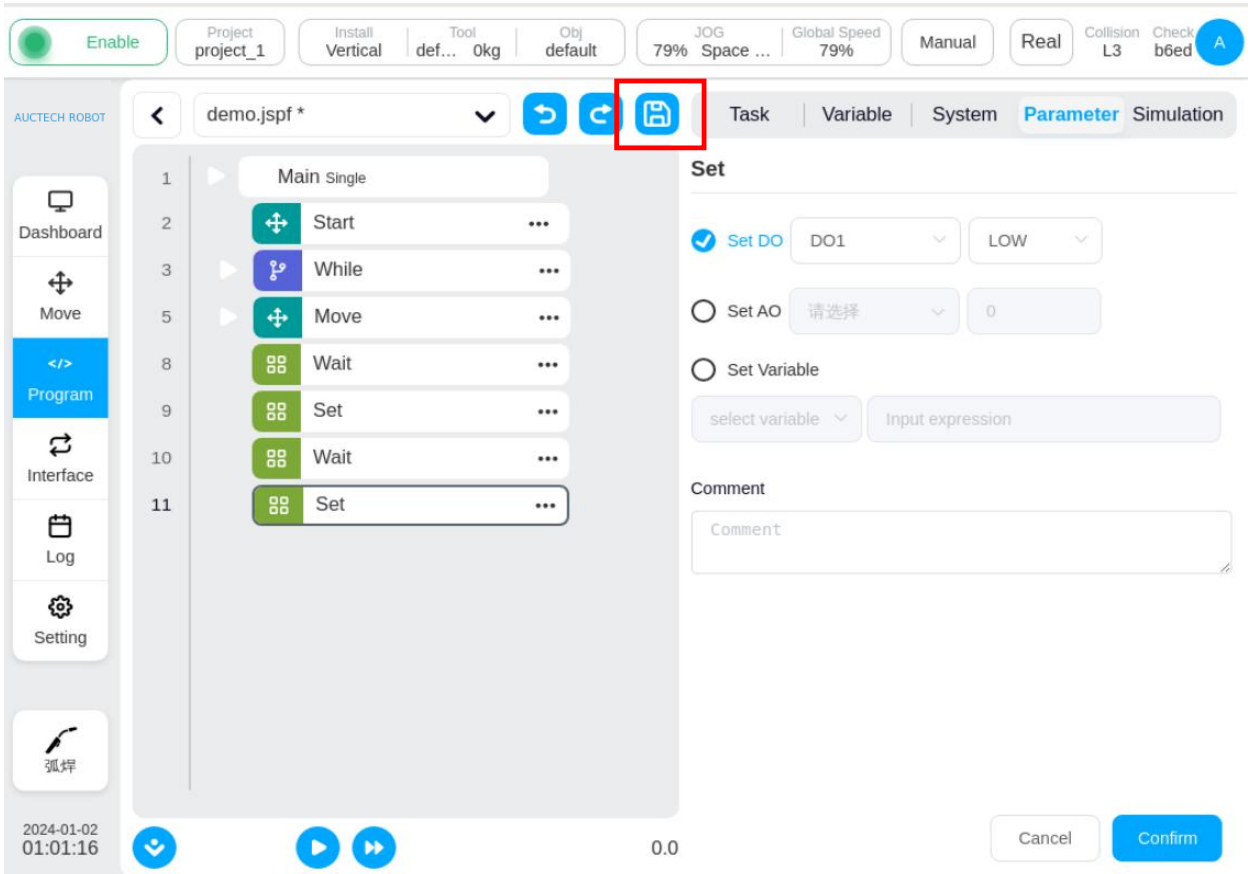
Find the Wait function block in the “Task" tab - "Basic" area, drag it to the position shown in the figure, select "Delay" in its parameter configuration area, and enter the parameter 2000, which means that the program will delay the execution of subsequent function blocks for 2s when the program is executed.

2.5 Write a program



In the “Task" tab - "Basics" area, select the Set function block and drag it below Wait. Select "Set Output" in the parameter configuration area, and select DO1 and HIGH, which means that the digital output port 1 is set high when the program is executed to the function block.

2.5 Write a program

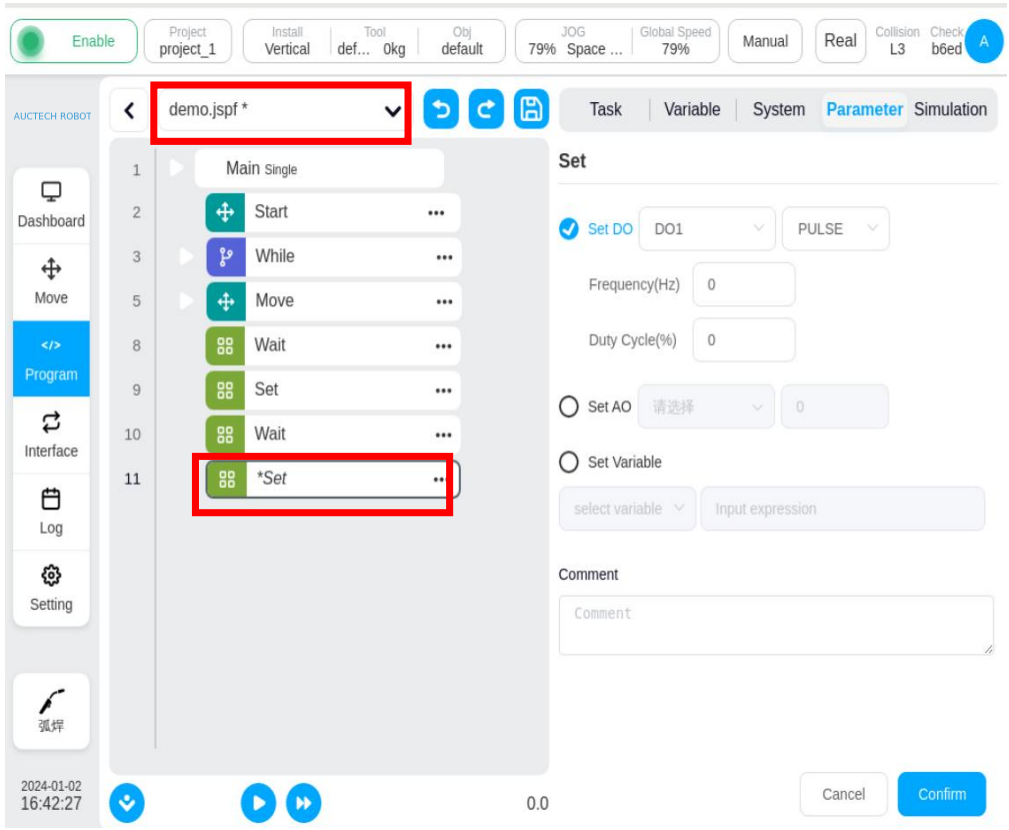


Similarly, add a Wait function block to set the delay of 1s, and add a Set function block to set DO1 to low.

Click the "Save Program" button below to save the program.

2.5 Write a program

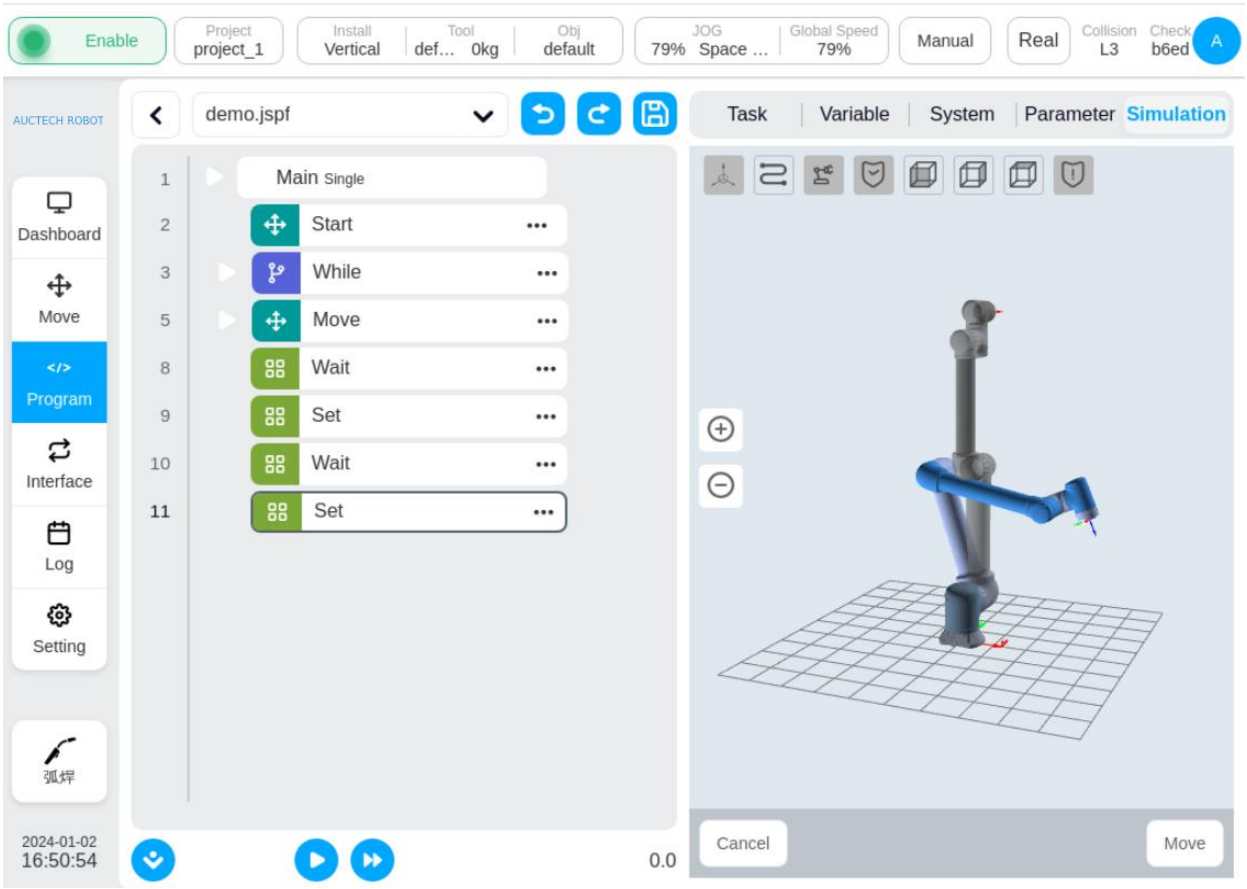
Note: After the function block, program, project, etc. are changed, there is a "*" sign in the corresponding position. Disappear after saving.



1. If the project is not saved, it will not affect the current commissioning and production, but it will be restored to before the change after the power failure.

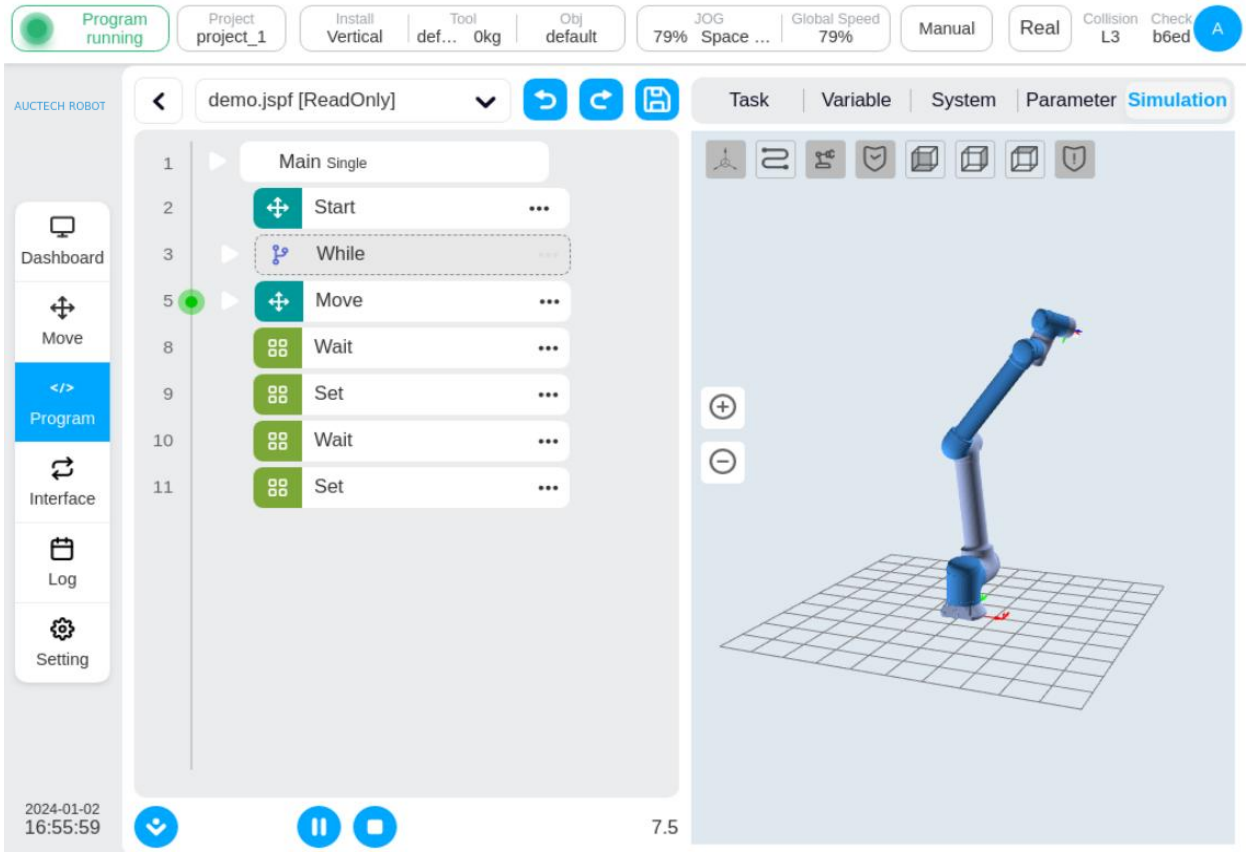
2. After the parameters in the function block are changed, you should click "Confirm" in the lower right corner first, and then perform other operations. If the function block is not confirmed, you cannot click "Save Program".

2.6 Run the program



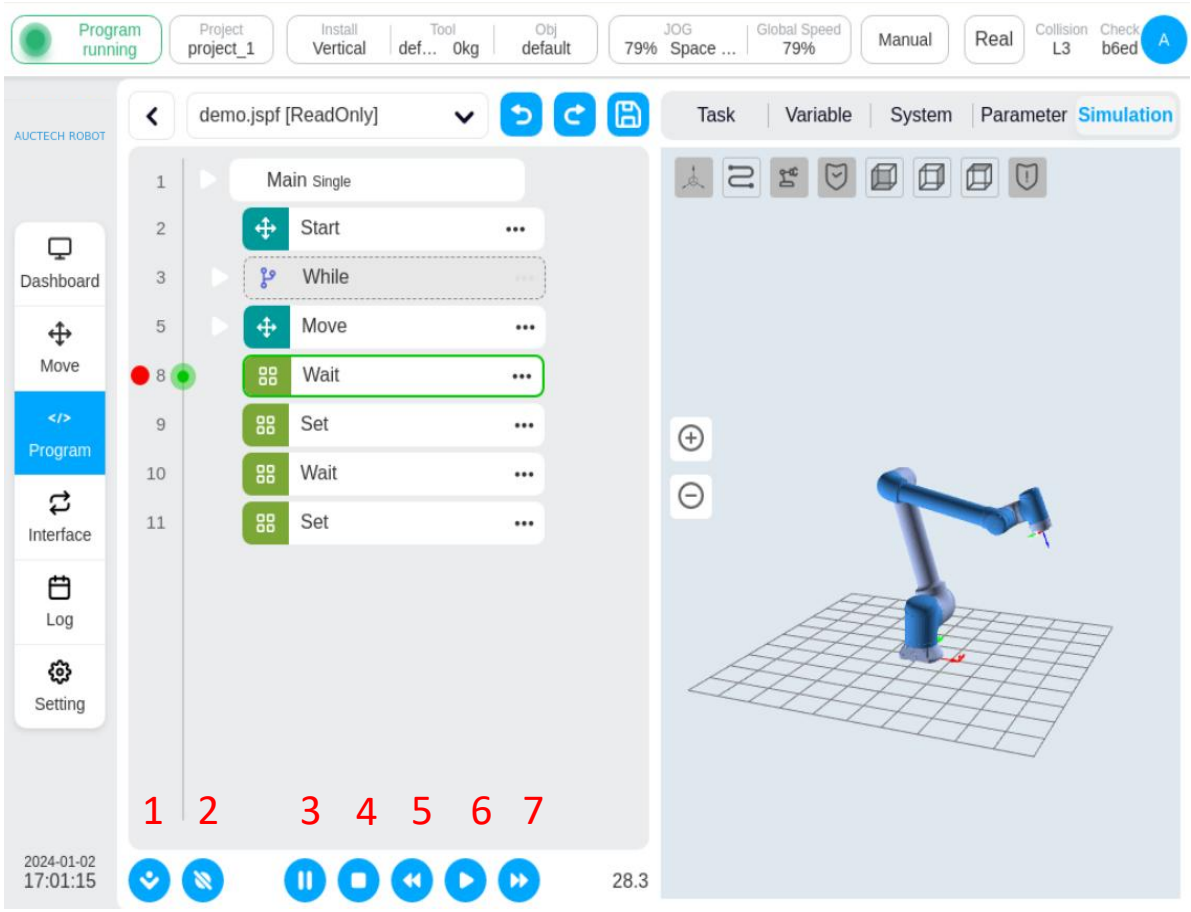
Switch to the "Simulation" tab, click the Run button at the bottom, if the current position of the robot is different from the starting point of the program, the "Move" button will be displayed, and the starting position of the robot will be displayed on the 3D display. After long pressing and Hold "Move" button to move the robot to the starting position. Click the Run button again to run the program.

2.6 Run the program



When the program is executed, the green dot in front of the program tree is used to indicate the function block that is currently being executed, and the real-time posture of the robot can be displayed in the 3D model. You can pause or stop the program while it is running.

2.6 Run the program

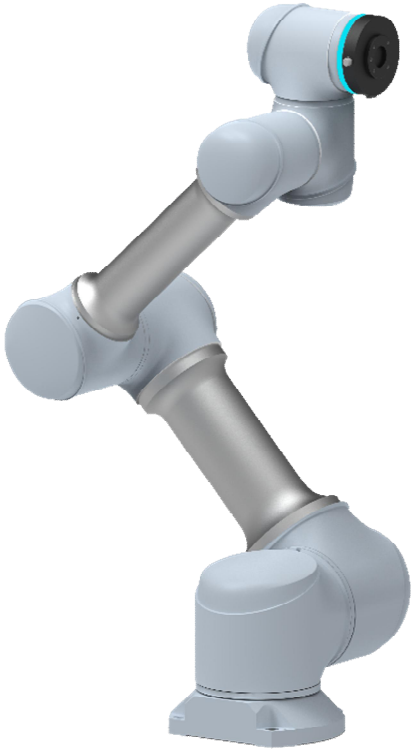


Debugging Function: Breakpoint

You can add a breakpoint before the sequence number of the program tree.

The program runs to a breakpoint and pauses, and in the lower right corner, you can choose to continue or stop.

1: Auto-follow 2: Cancel the breakpoint
3: Pause the program 4: Stop the program
5: Backwards 6: Continue running
7: Forward



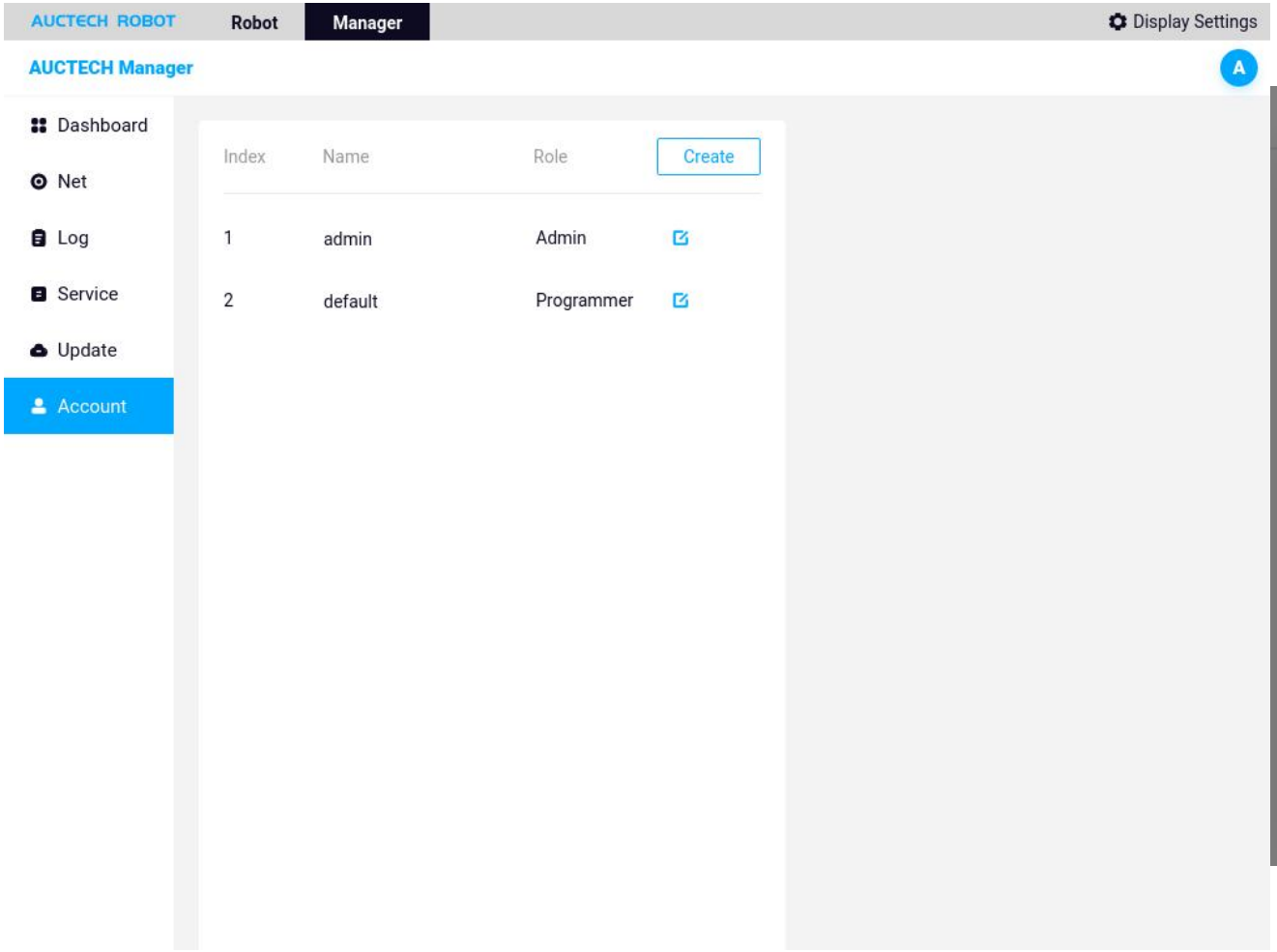
Lecture 3

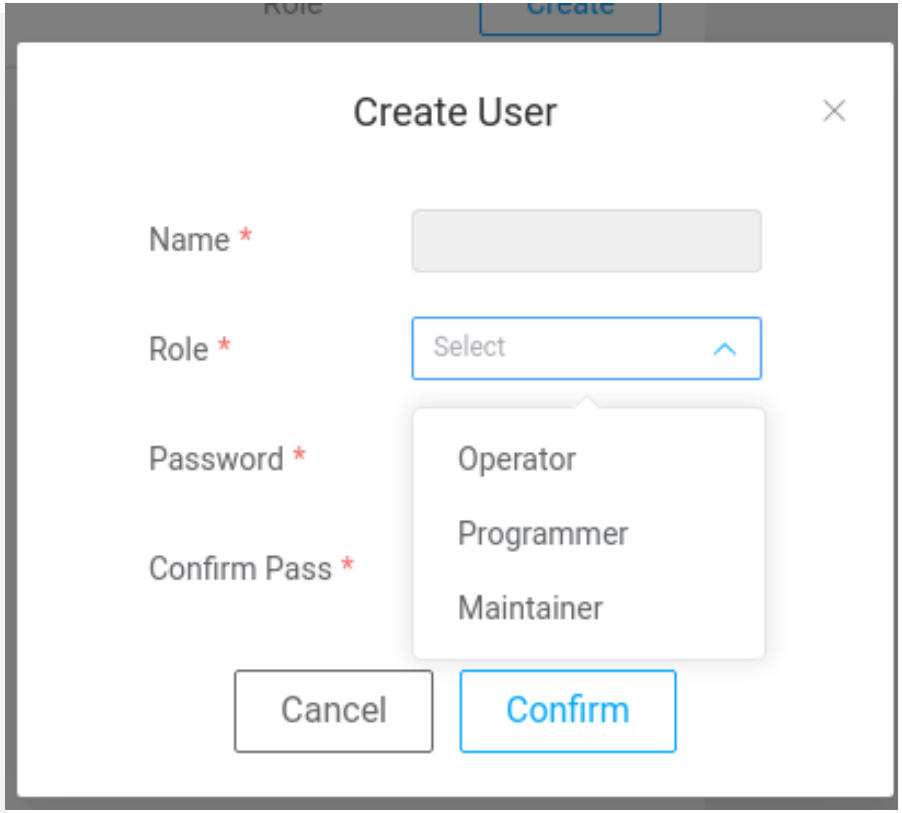
User level

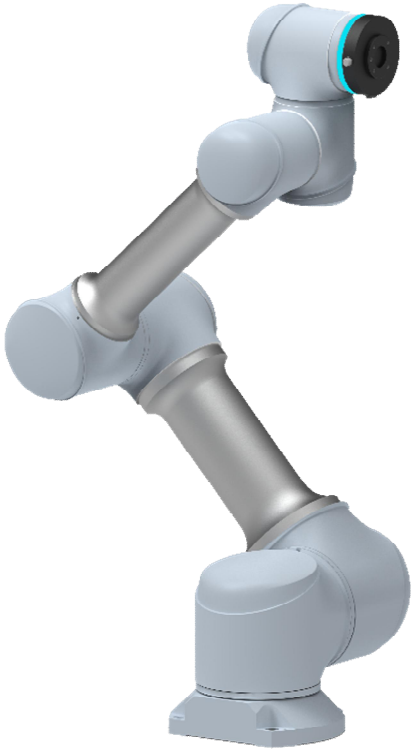
3.1 User level

User permission level	Permission description
Operator	• Allows you to select projects, run programs, manually jog programs, view robot status, and more
Programmer	• Operator user permissions • Programming robot programs, engineering configurations • The default username is default and the initial password is 123
Maintainer	• Operator user permissions • System updates
Admin	• Maintainer user permissions • User management • The single username is admin and the initial password is 123

3.1 User level



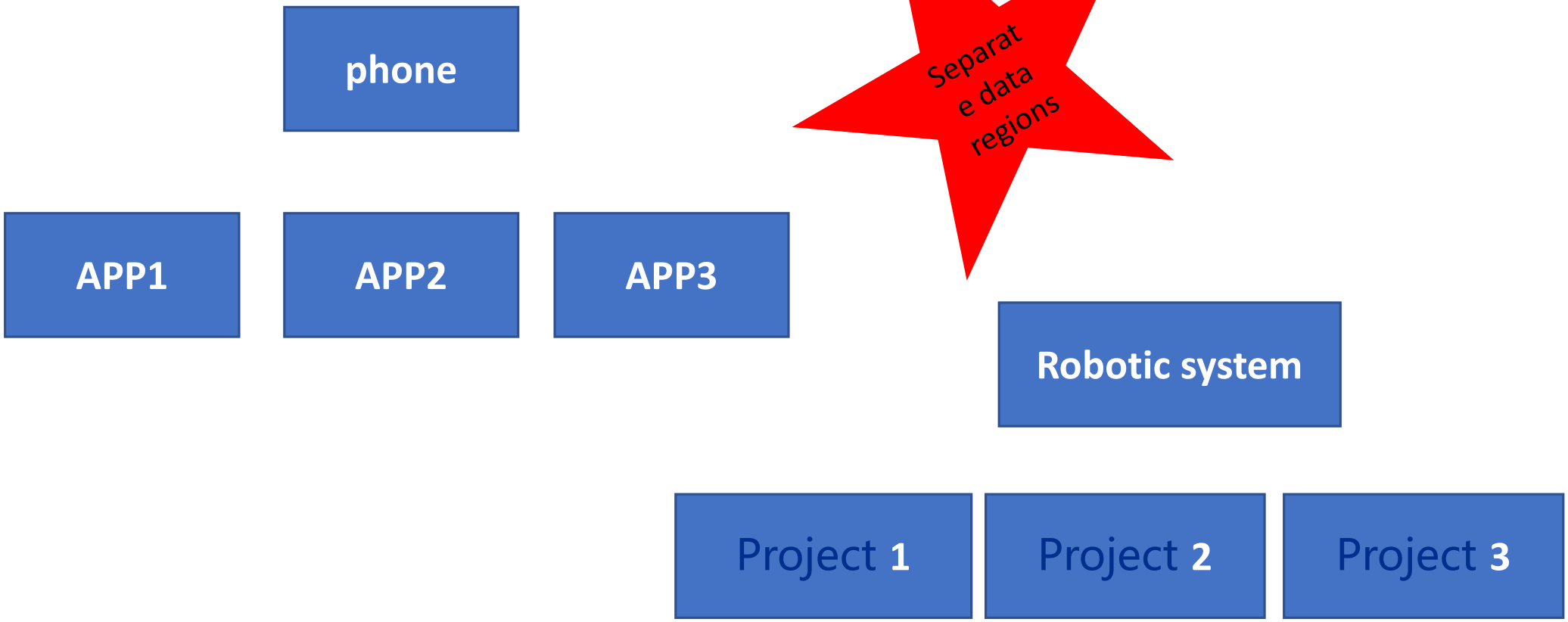




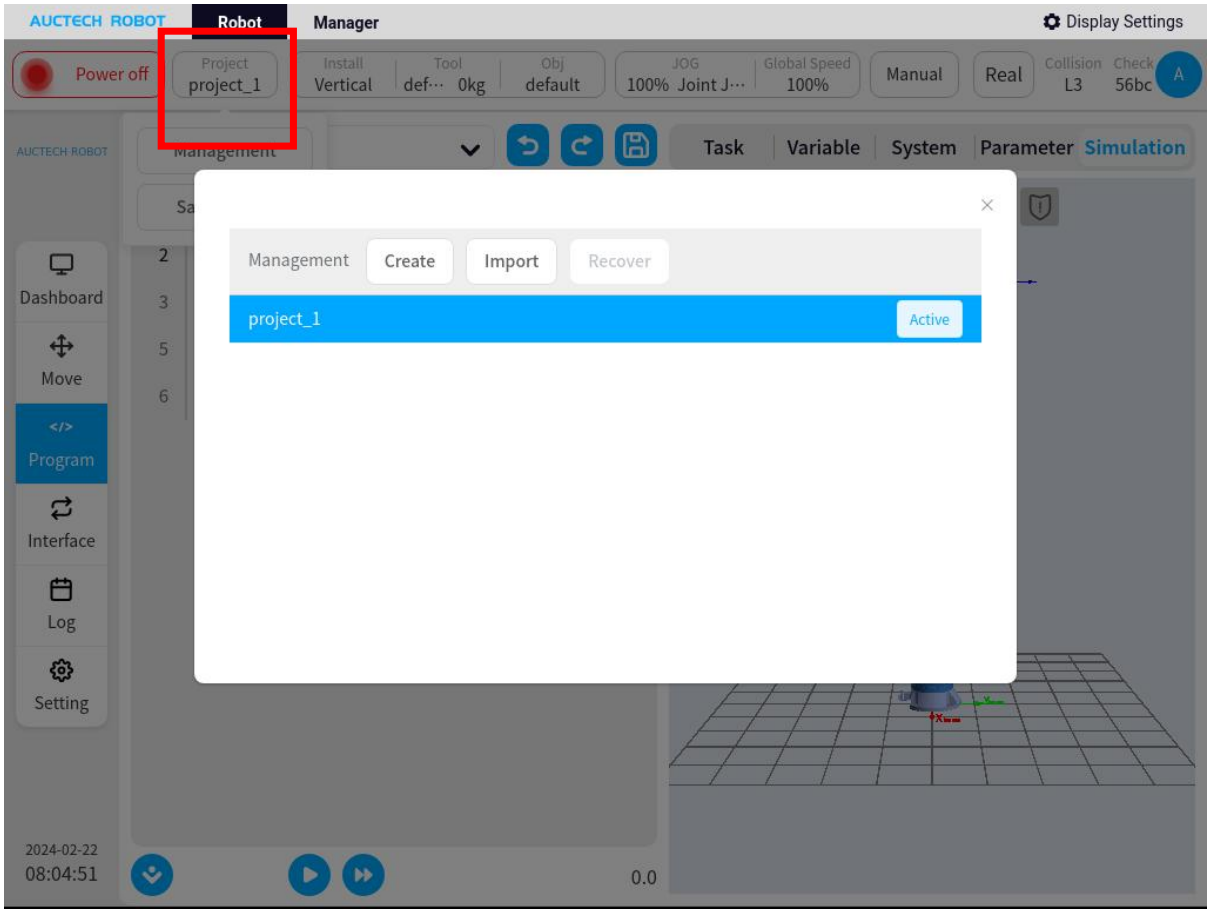
Lecture 4

Project concepts

4.1 Project concepts

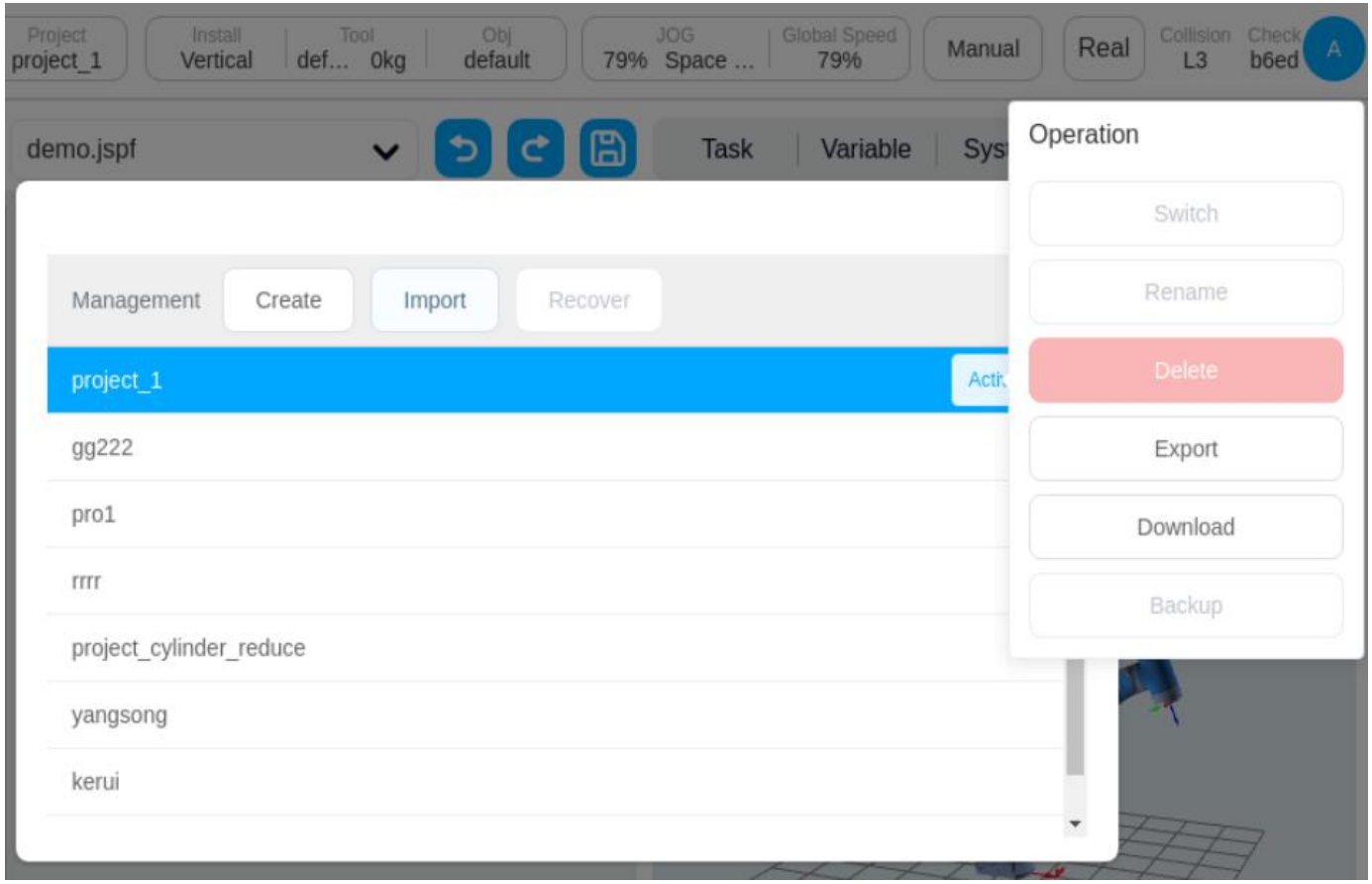


4.2 project management

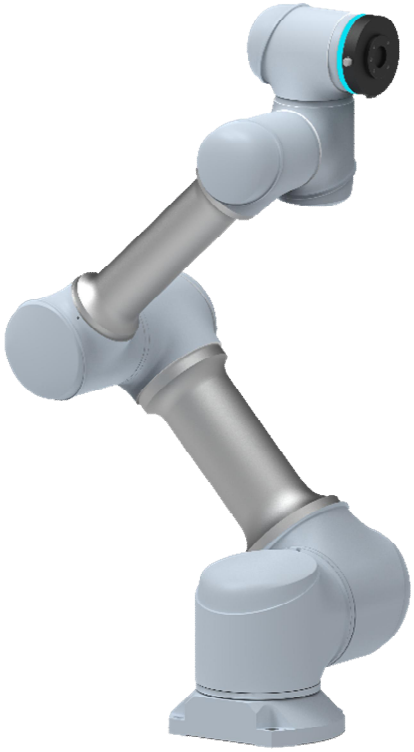


Support multi-process management, engineering data includes robot programs, engineering global variables, engineering settings and other information. When the system starts loading, the current project data will be loaded according to the system settings, and other project data cannot be used in the current project.

4.3 project management



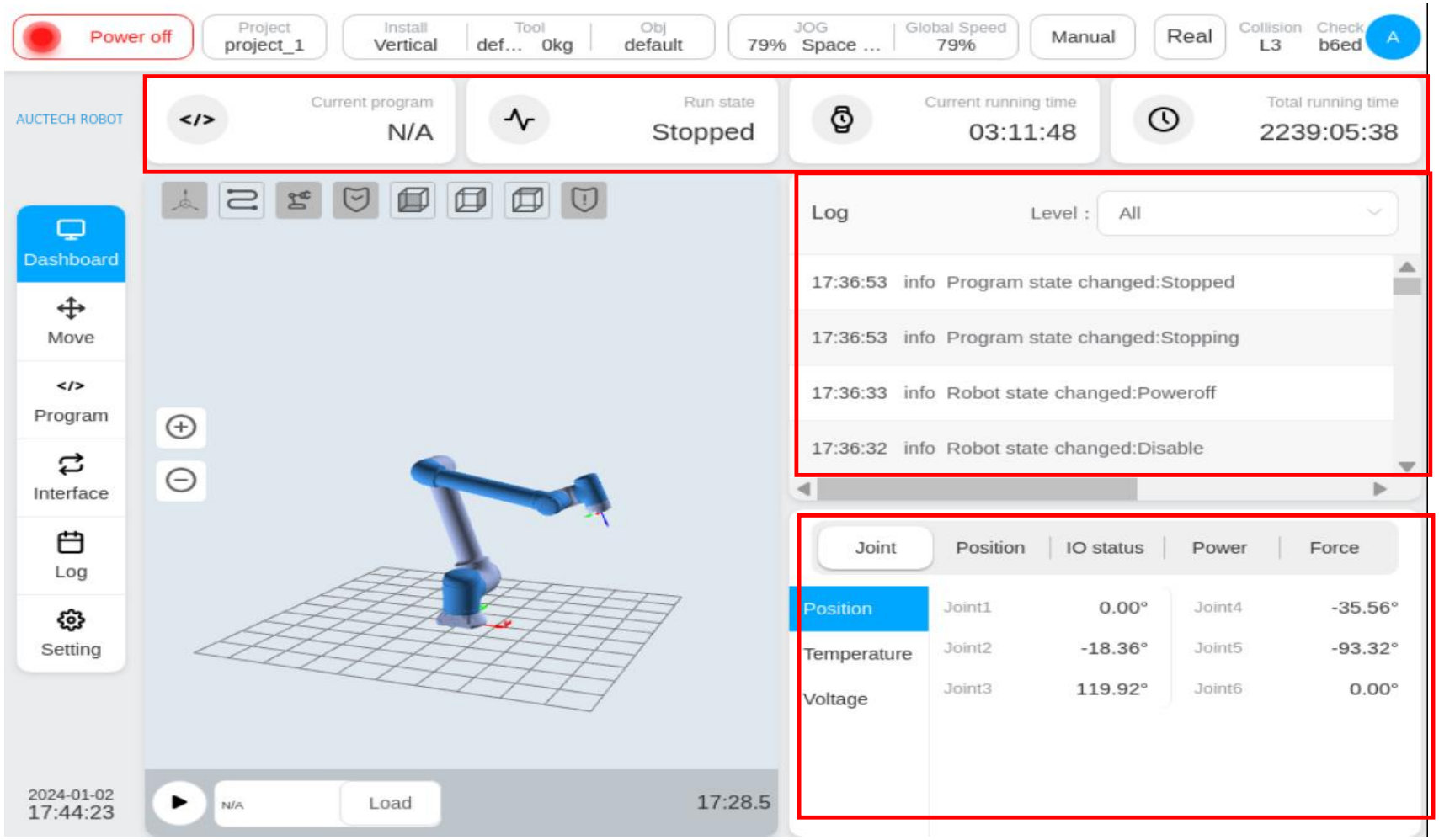
On the project management page, you can create, import, rename, and export projects.



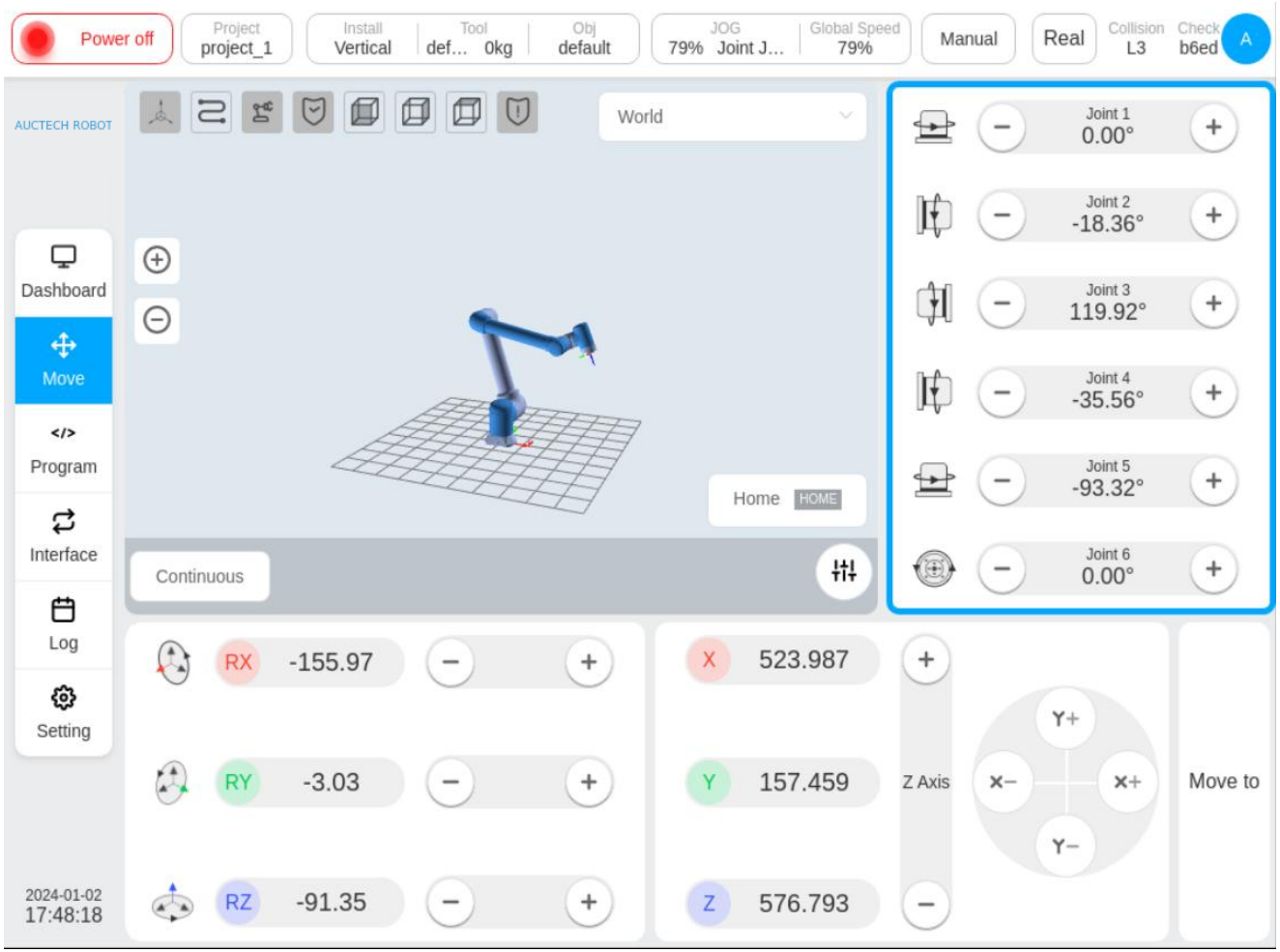
Lecture 5

Overview of the interface

5.1 Dashboard



5.2 Move

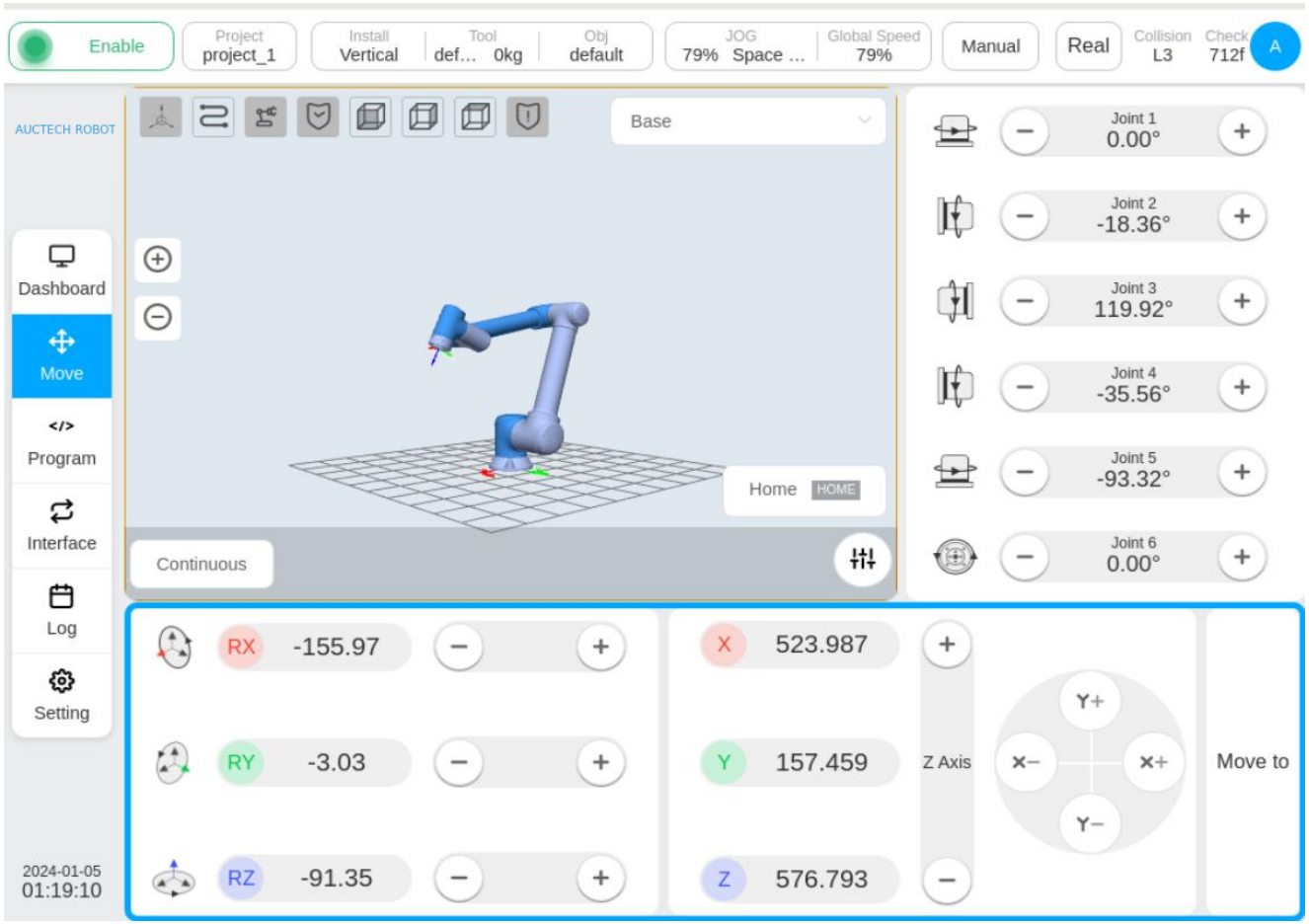


There are two modes of movement when manually moving the robot: joint movement and end movement.

When you click on a joint to move any area, a blue border appears in that area to select for joint movement.

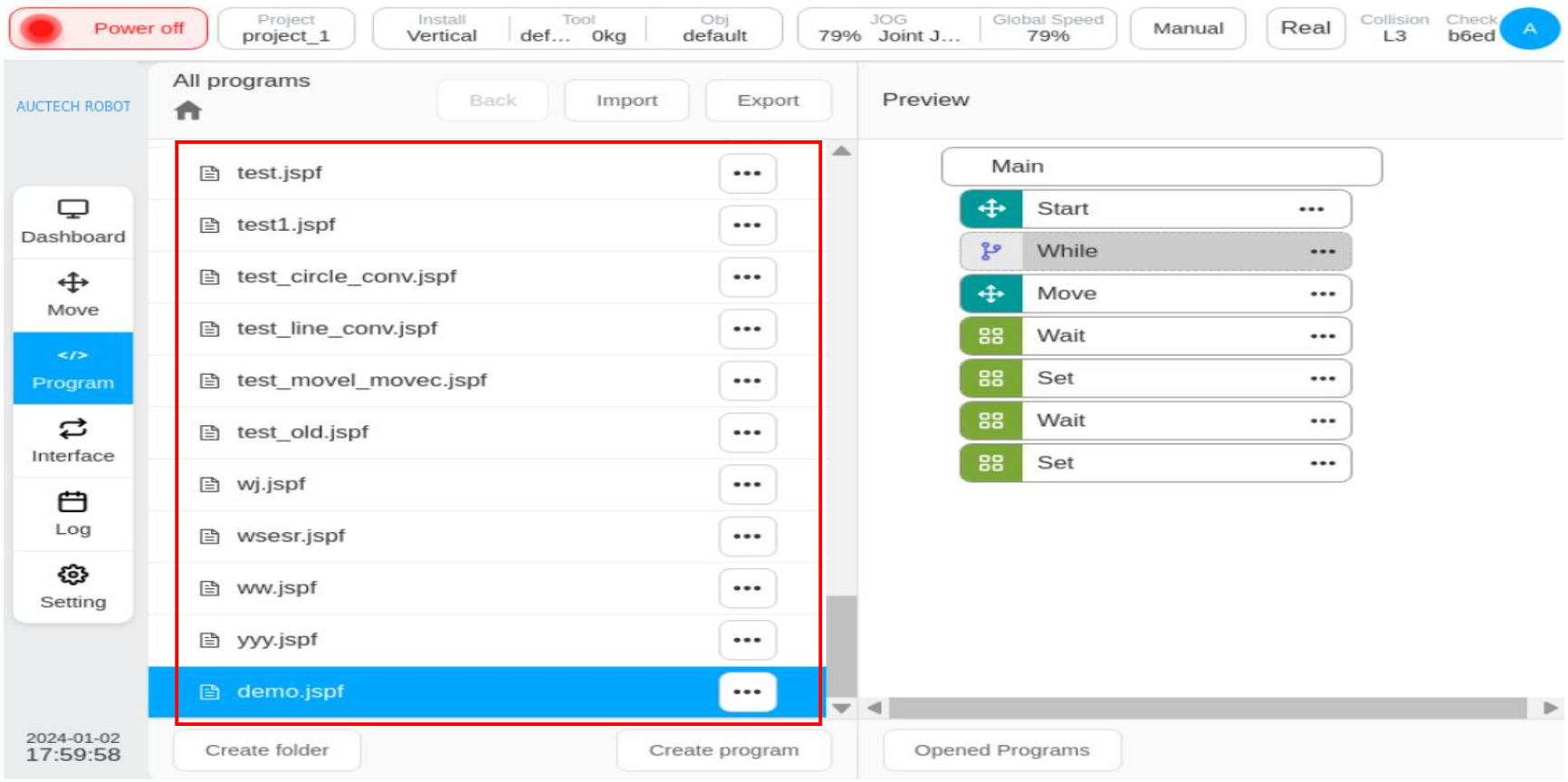
Note: The robot does not refer to the coordinate system when the joints move

5.2 Move

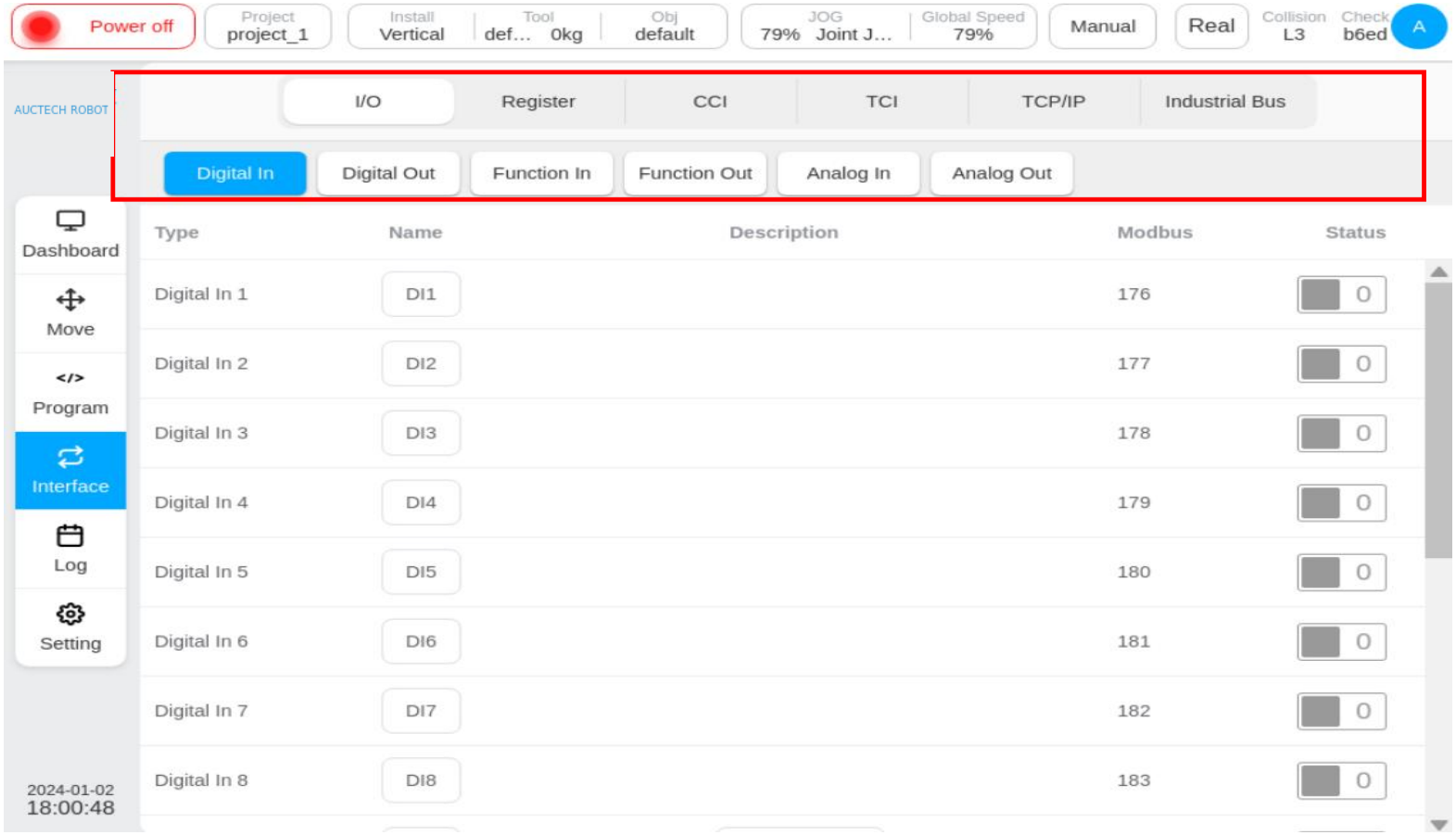


When you click on the end to move any area, a blue border appears in that area, and the end move is selected.

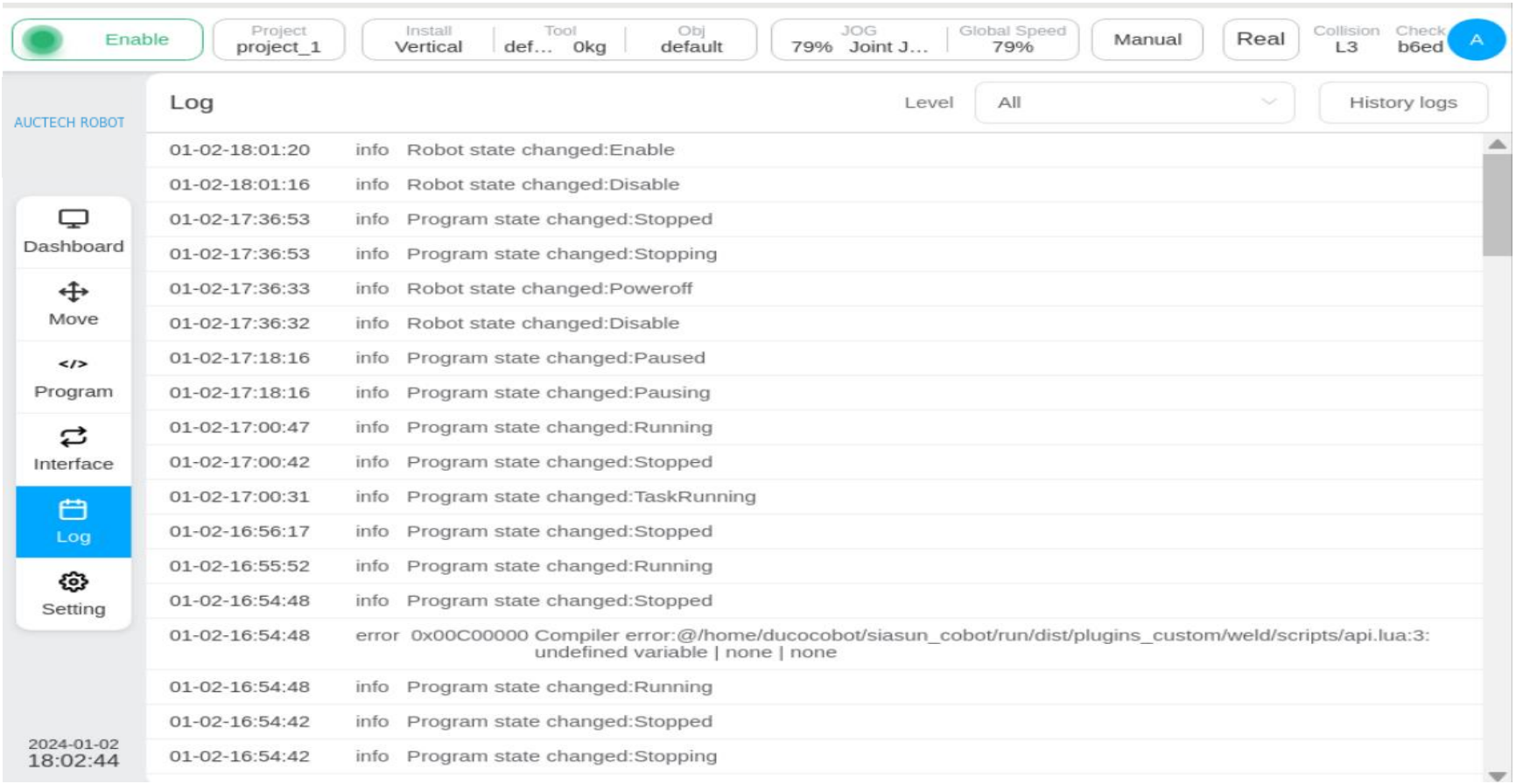
5.3 program list



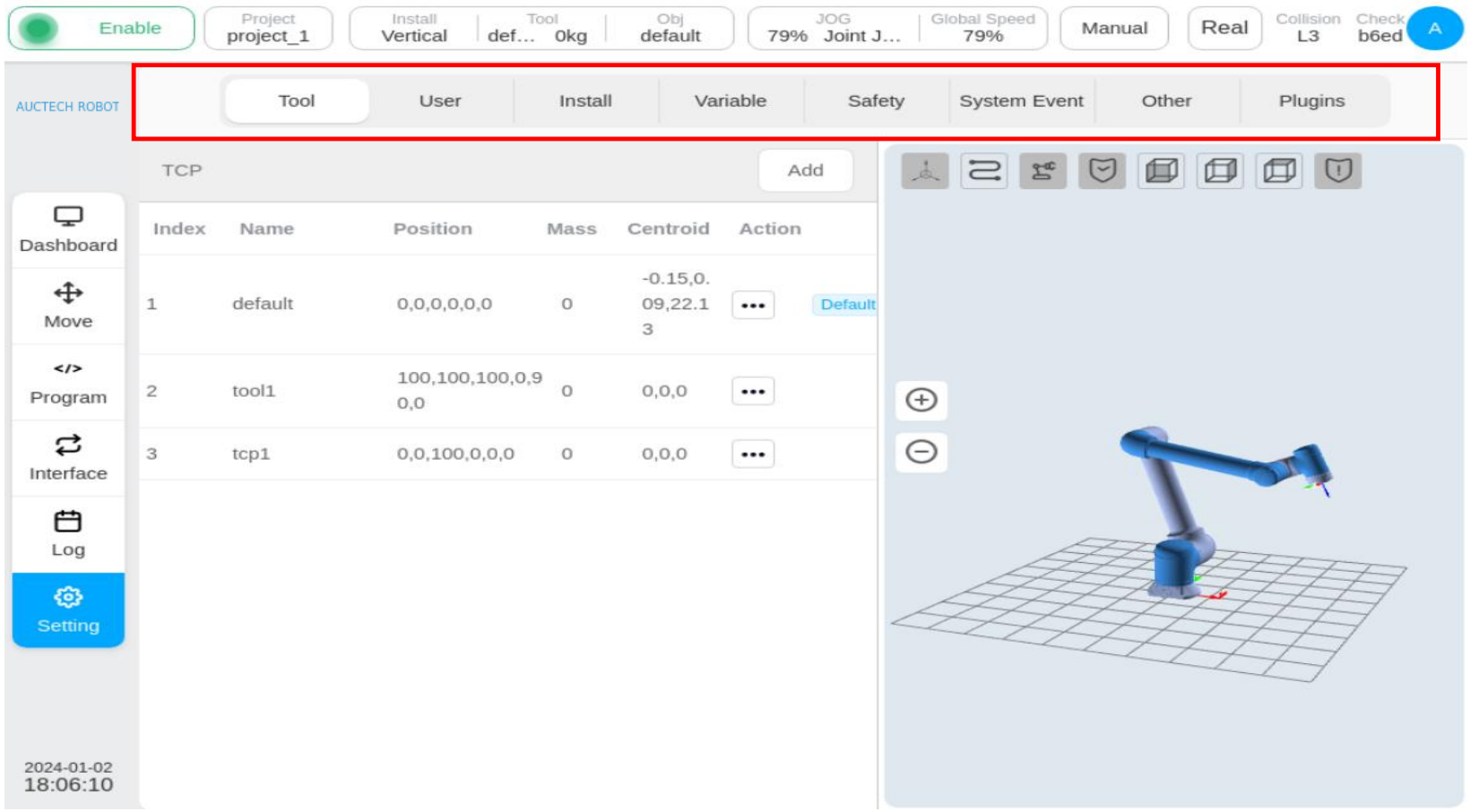
5.4 interface

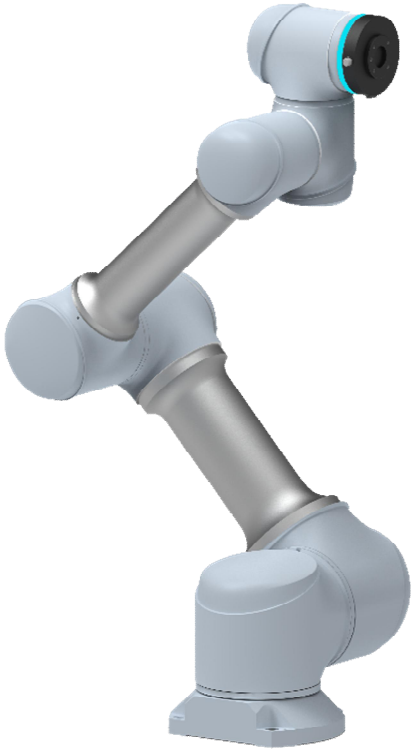


5.5 log



5.6 Setting

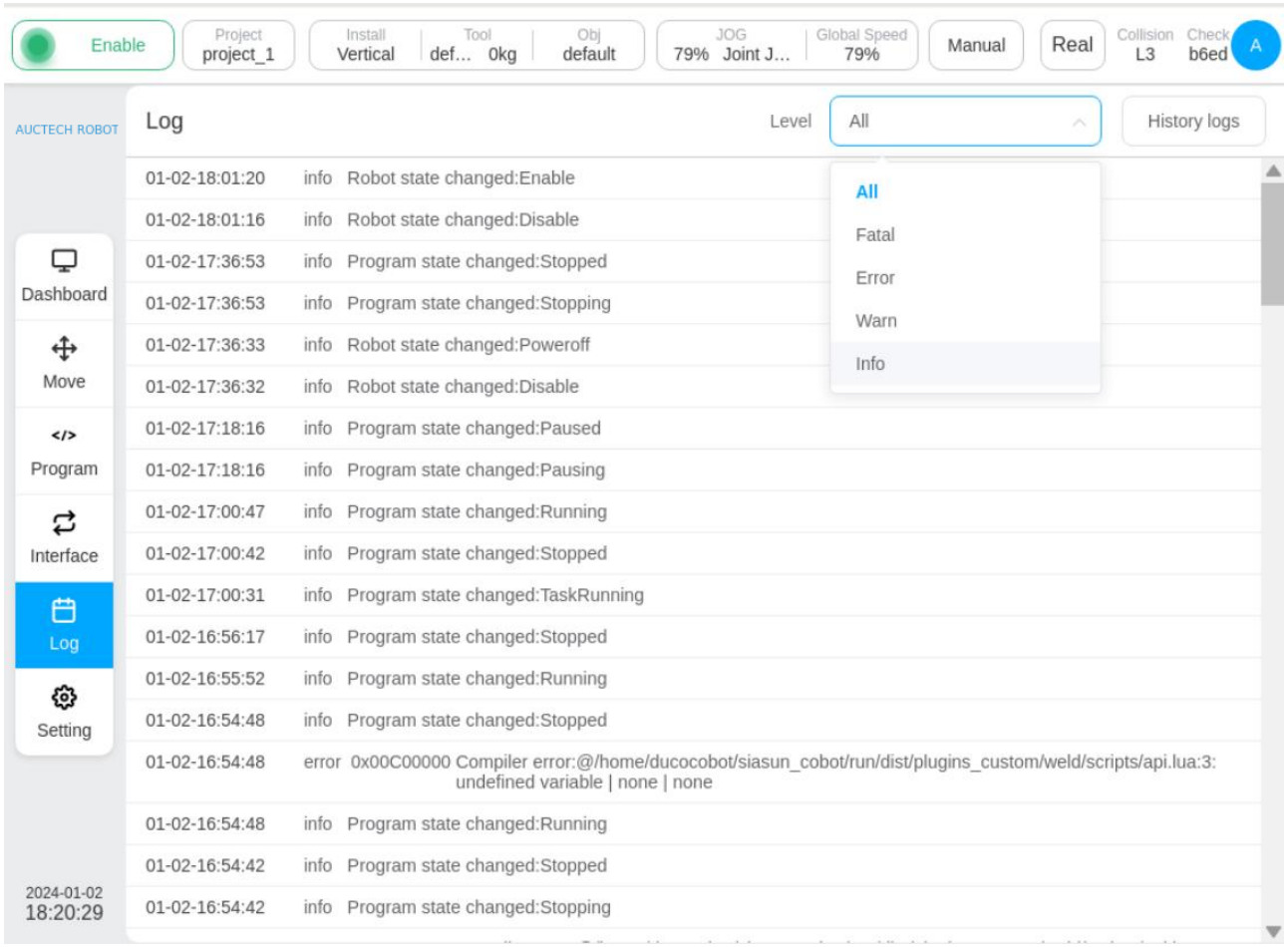




Lecture 6

Log introduction

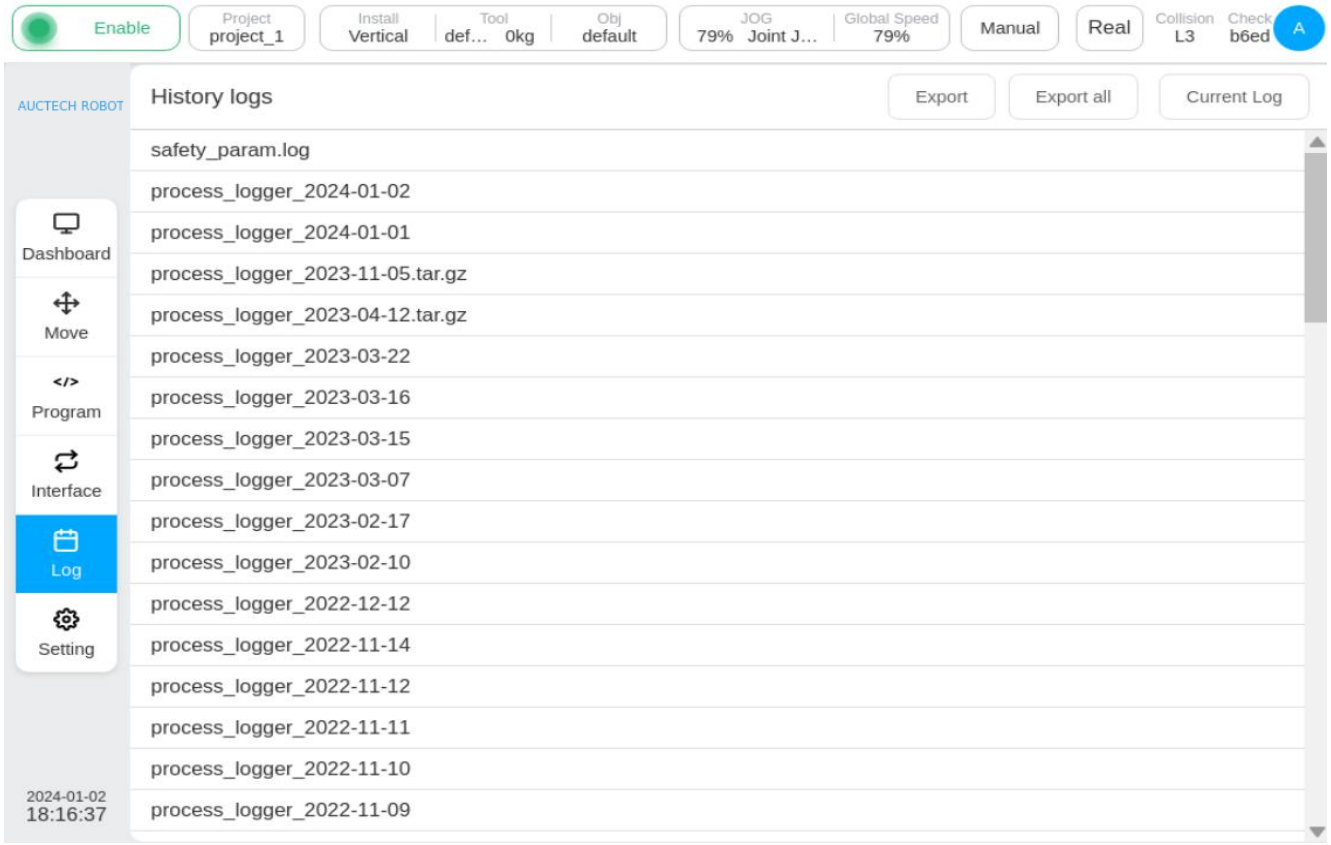
6.1 Log introduction



Click Export Historical Logs to display the list of all historical log files. You can export one of these log files or export all logs.

Note: All historical log files exported to a USB flash drive (Fat32 format) are a compressed package named logs.tar

6.1 Log introduction



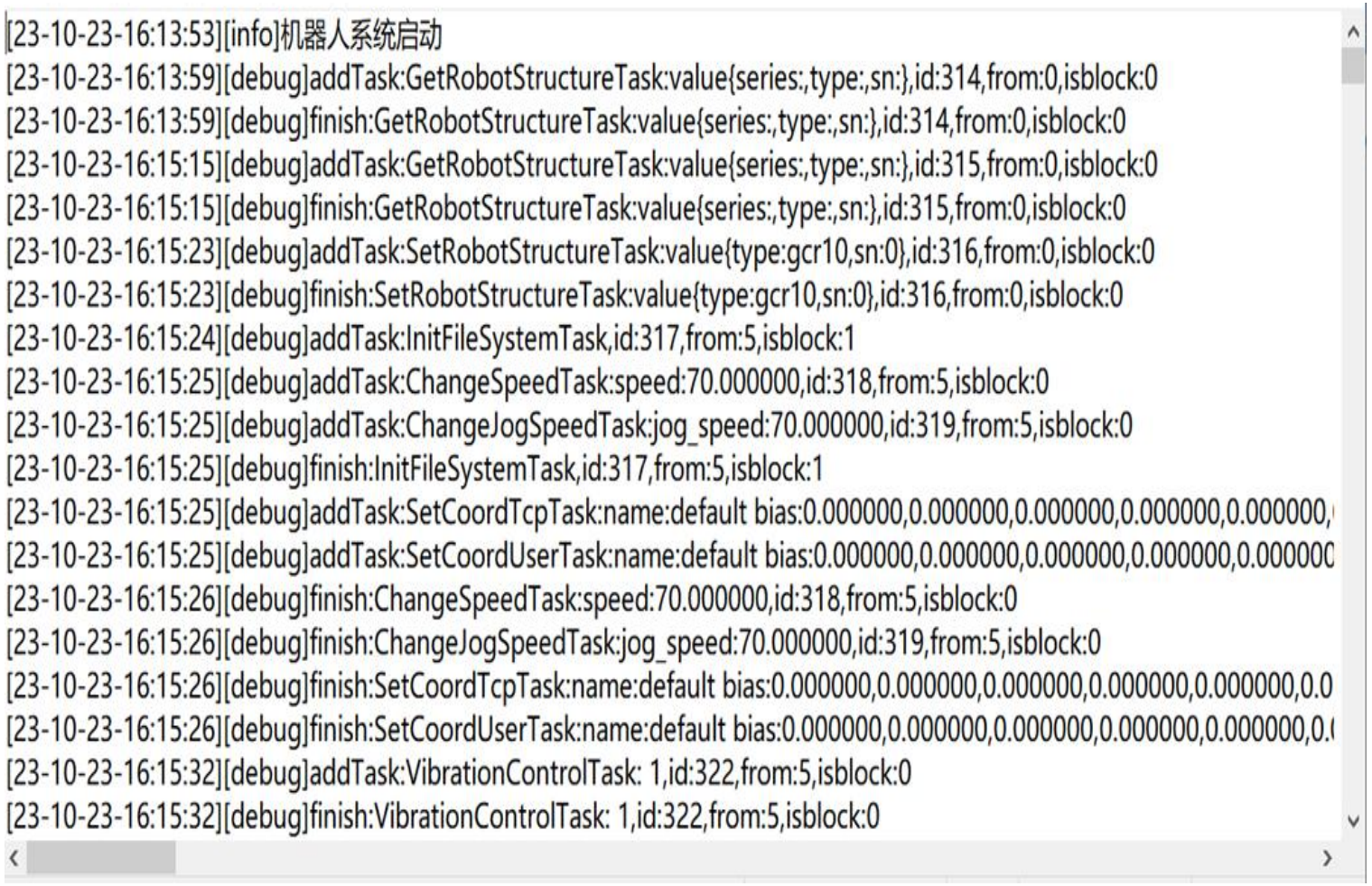
There are three types of log information in the diachronic log:

1.process_logger Process logging

2.ec_logger Error logging

3.brake_test.log Brake detection log

6.2 process_logger



process_logger

Process logging

mainly records

user operation

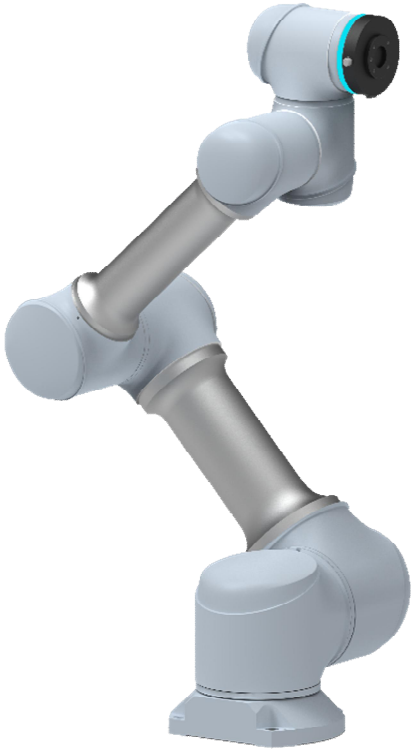
information

6.3 ec_logger

```
[23-11-23-07:30:40][info]0x13000001:  state changed  Master state changed
[23-11-23-07:51:22][info]0x      0:
[23-11-23-07:51:22][info]0x13000001:  state changed  Master state changed
[23-11-23-07:51:30][info]0x      0:
[23-11-23-07:51:30][info]0x13000001:  state changed  Master state changed
[23-11-23-08:06:12][info]0x      0:
[23-11-23-08:08:39][info]0x13000001:  state changed  Master state changed
[23-11-23-08:25:34][info]0x13000001:  state changed  Master state changed
[23-11-23-08:34:39][info]0x13000001:  state changed  Master state changed
[23-11-23-08:37:02][info]0x      0:
[23-11-23-08:37:02][info]0x13000001:  state changed  Master state changed
[23-11-23-08:37:09][info]0x      0:
[23-11-23-08:37:09][info]0x13000001:  state changed  Master state changed
[23-11-23-09:28:09][info]0x13000001:  state changed  Master state changed
[23-11-23-09:31:26][info]0x      0:
[23-11-23-09:31:26][info]0x13000001:  state changed  Master state changed
[23-11-23-09:31:34][info]0x      0:
[23-11-23-09:31:34][info]0x13000001:  state changed  Master state changed
[23-11-23-09:40:45][info]0x13000001:  state changed  Master state changed
```

ec_logger Error Logging:

The main real-time communication data is used for in-depth bug troubleshooting and analysis.



Lecture 7

Multi-terminal connection

7.1 Multi-terminal connection - wired connection

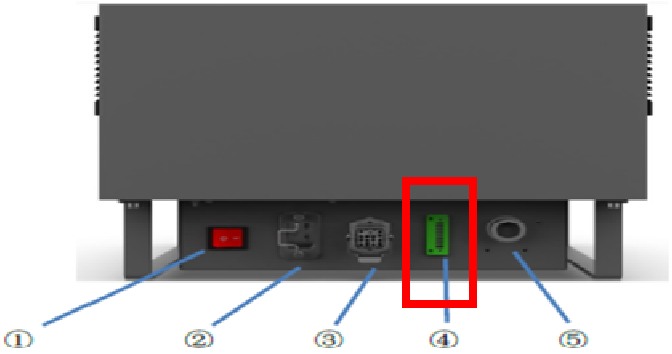
Users can connect the robot by using PC or mobile device with wired or wireless.

Wired Connection:

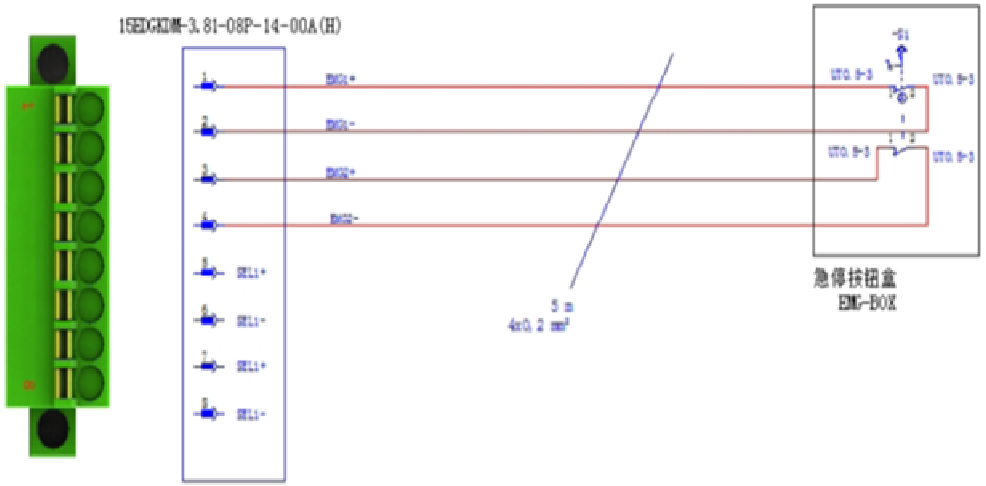
1. Connect the network cable to the "LAN" interface on the control cabinet, and the other end to the computer network port.
2. Set the robot IP according to the requirements, for example: the robot IP is set to: 192.168.1.10. (Enter the system control - network settings page to view the IP address of the robot)
3. Set the computer IP to be in the same network segment as the robot IP, for example, the computer IP is set to: 192.168.1.11.
4. Ping the robot with the computer to test the network on/off.
5. Open a browser, it is recommended to use Google Chrome (Chrome), and then enter the IP address and port number 7000 just set by the robot in the address bar, such as 192.168.1.10:7000 or http:AUCTECH-cobot.com:7000, enter after the input is completed.

7.2 Multi-terminal connection - wireless connection

When using a mobile device (PAD or PC) to wirelessly connect the robot, It is possible to leave the teach pendant unattached, but at this time the robot safety loop is in adisconnected state, and the emergency stop button needs to be connected at the shield interface of the teach pendant.



①:System power switch
②:AC input interface
③:Robotic arm connection interface
④:Shielded teach pendant interface
⑤:Teach pendant connection interface

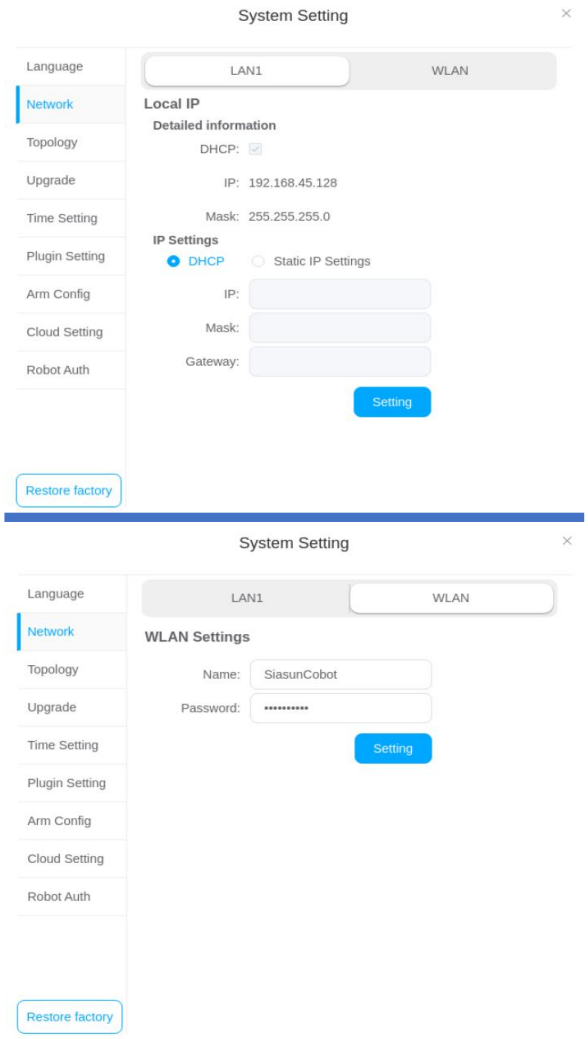


7.2 Multi-terminal connection - wireless connection

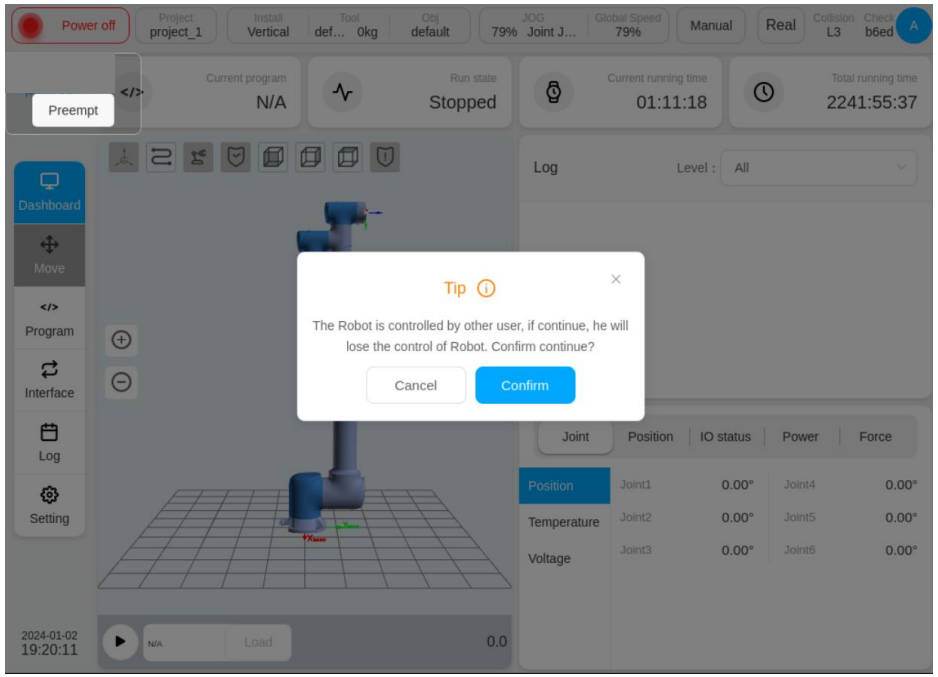
Connection Steps:

1. Start the power supply of the control cabinet, use the mobile device to find the wireless network of the robot, according to the name of the wireless network of the robot control cabinet, click "Connect", and enter the default password: 1234567890.

2. After the mobile device is successfully connected to the wireless network of the robot control cabinet, open the browser (Google Chrome is recommended), enter http:AUCTECH-cobot.com:7000 in the address bar, or 192.168.117.66:7000 (wireless port IP address) to confirm and connect to the robot.



7.3 Multi-terminal connection



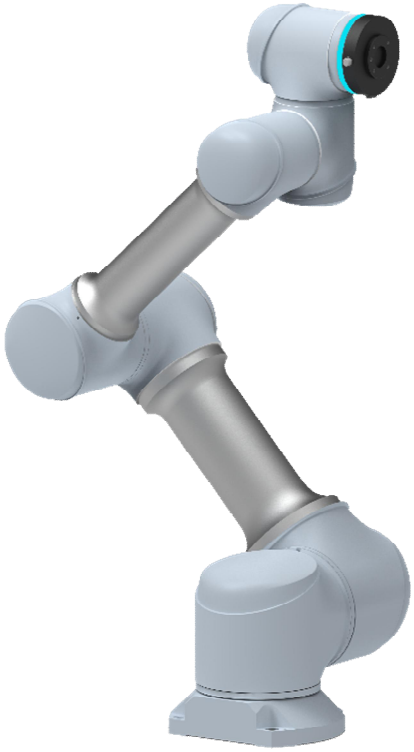
The system supports multiple terminals to be connected at the same time, but only one terminal has the right to control, which can run the program, configure the system and other functions. Other terminals only have the permission to view, and cannot run the robot arm or modify the setting information

When other terminals want to take control, you can click the "preempt" button in the upper left corner. The following dialog box pops up, click the "OK" button to obtain control of the system, and the terminal with control will lose control.

Part Ⅱ

Introduction to software operation

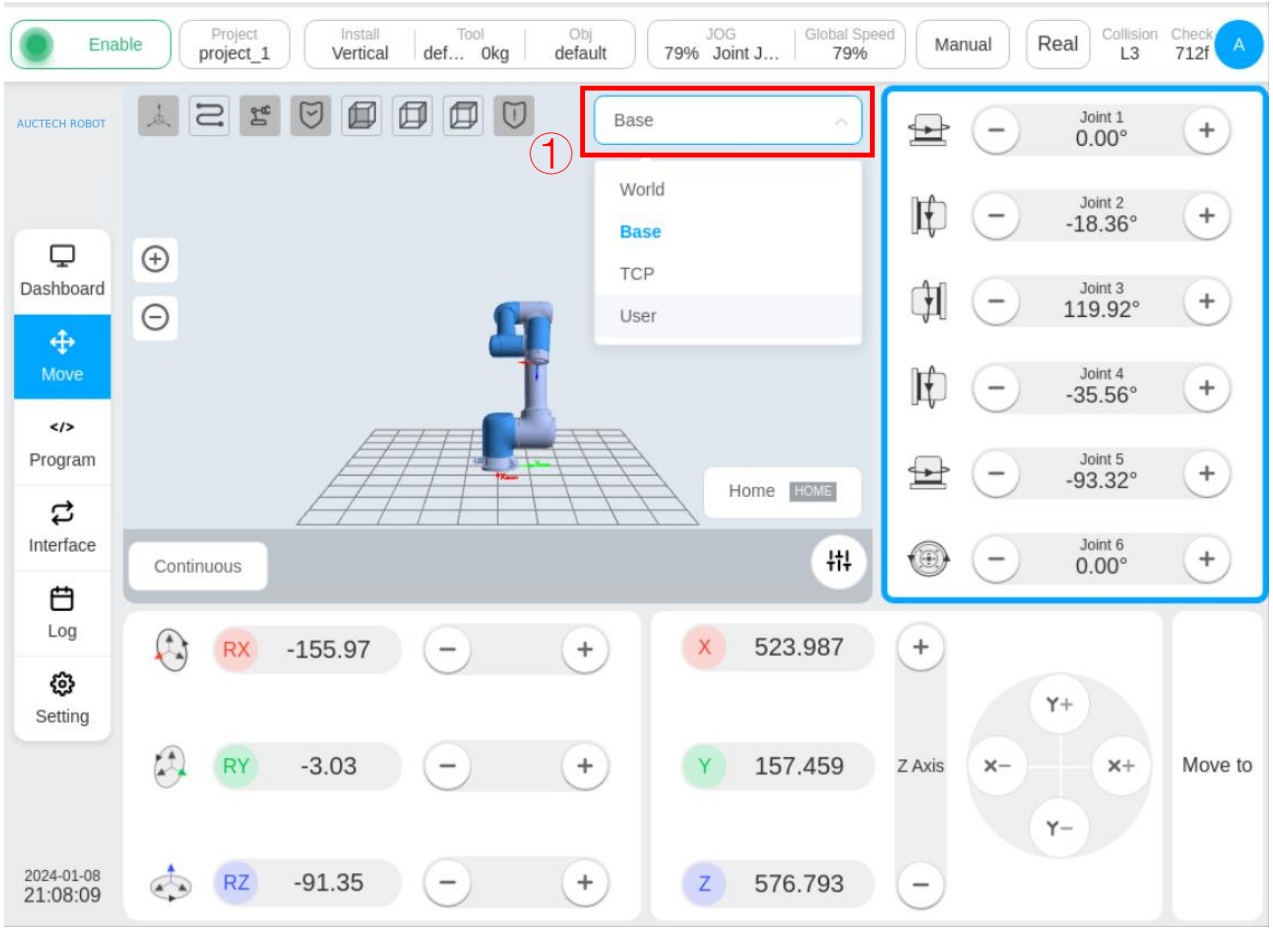
1. **Mobile interface**
2. **Programming**
3. **API introduction**
4. **Set up the information**
5. **Security settings**
6. **Description of the error**



Lecture 1

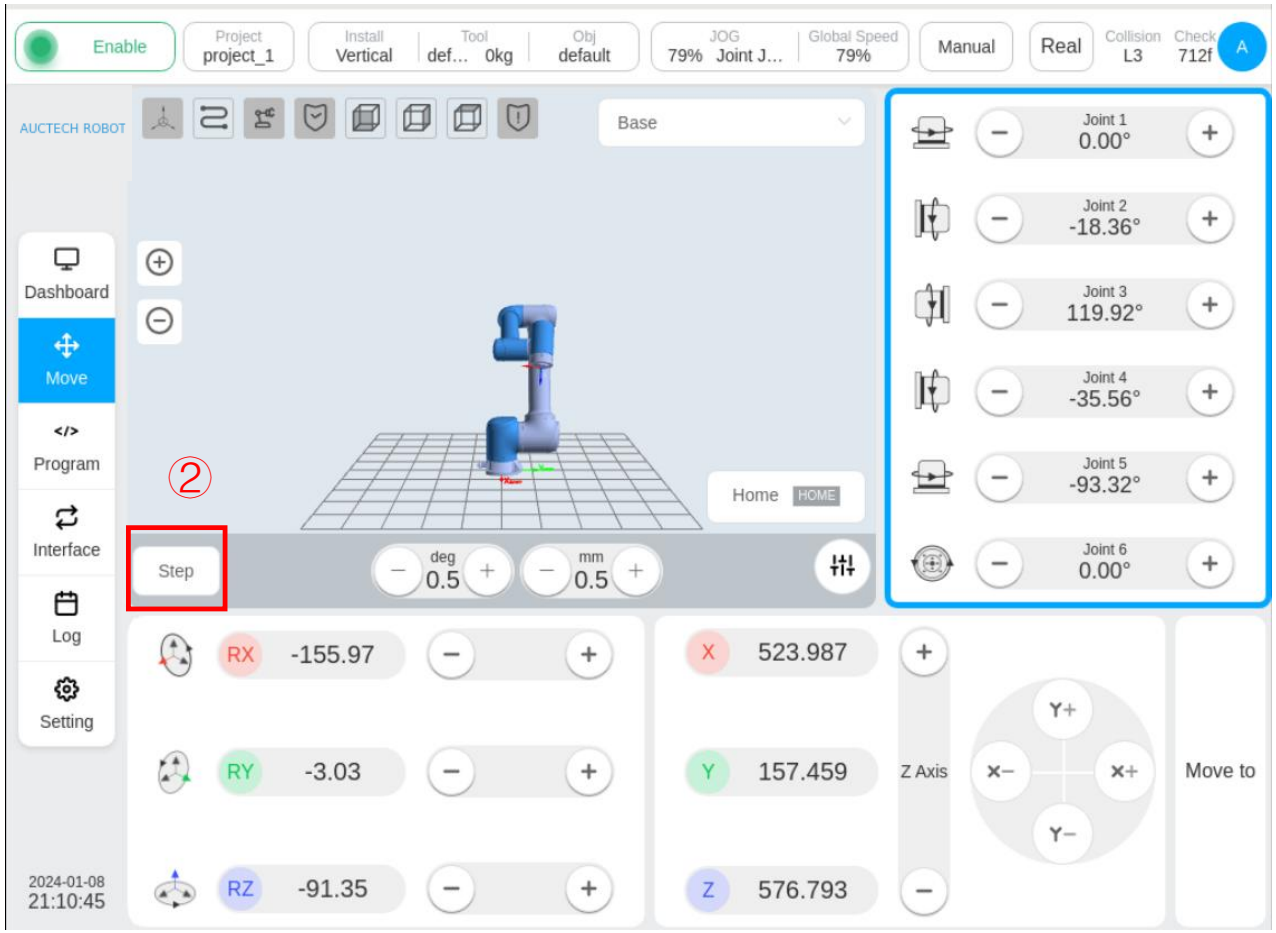
Move Interface

1.1 Coordinate system selection



logo①, You can select the coordinate system used by the current robot and the reference coordinate system of the manually operated robot .

1.2 Sport mode selection



logo②button to switch between "continuous mode" or "step mode" mode.

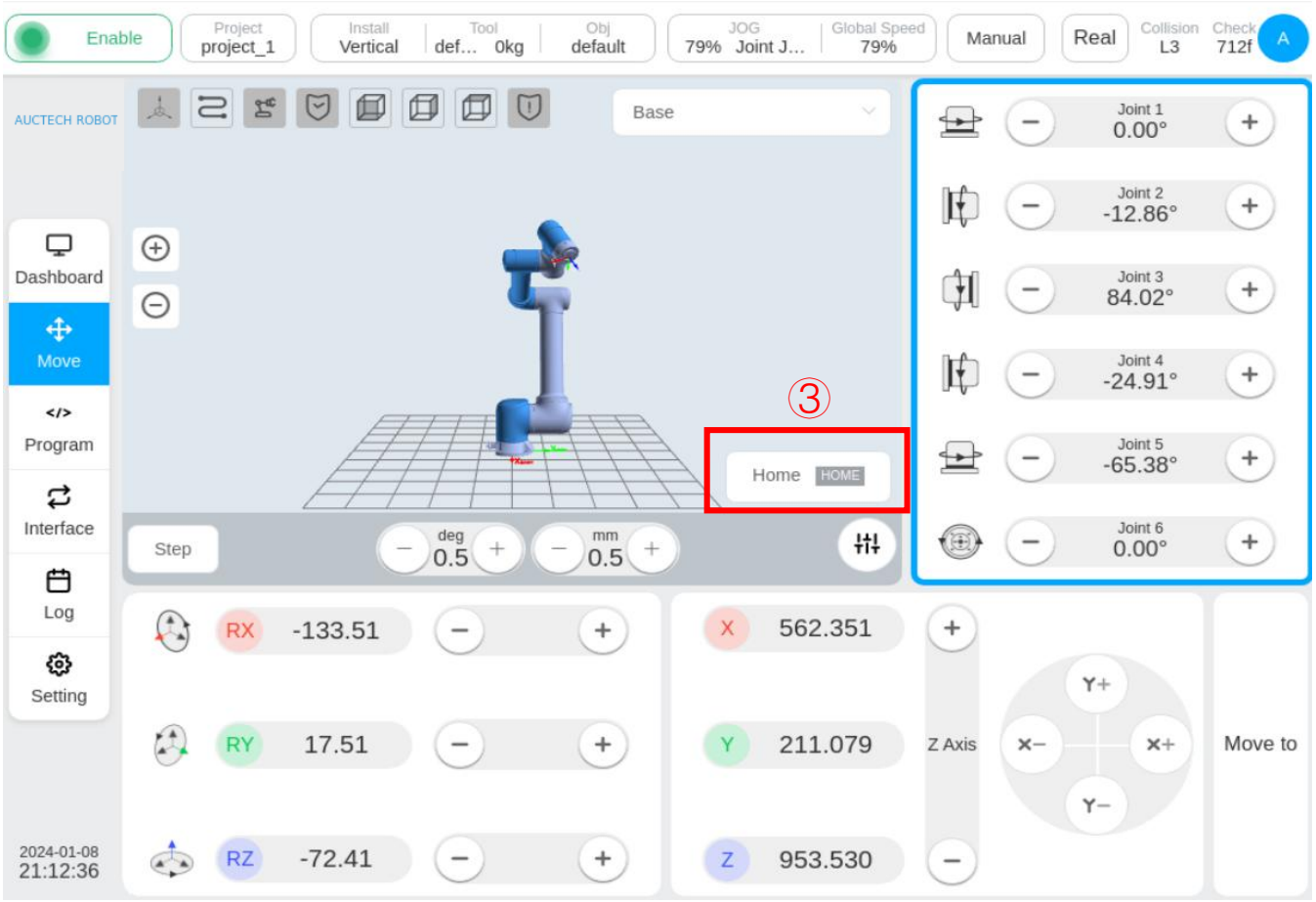
The range of settings for step angle and step distance in Step Mode:

The default values are 0.5deg and 0.5mm

The minimum allowable settings are 0.1deg and 0.1mm

The maximum allowable setting is 5deg and 5mm

1.3 Move to home point



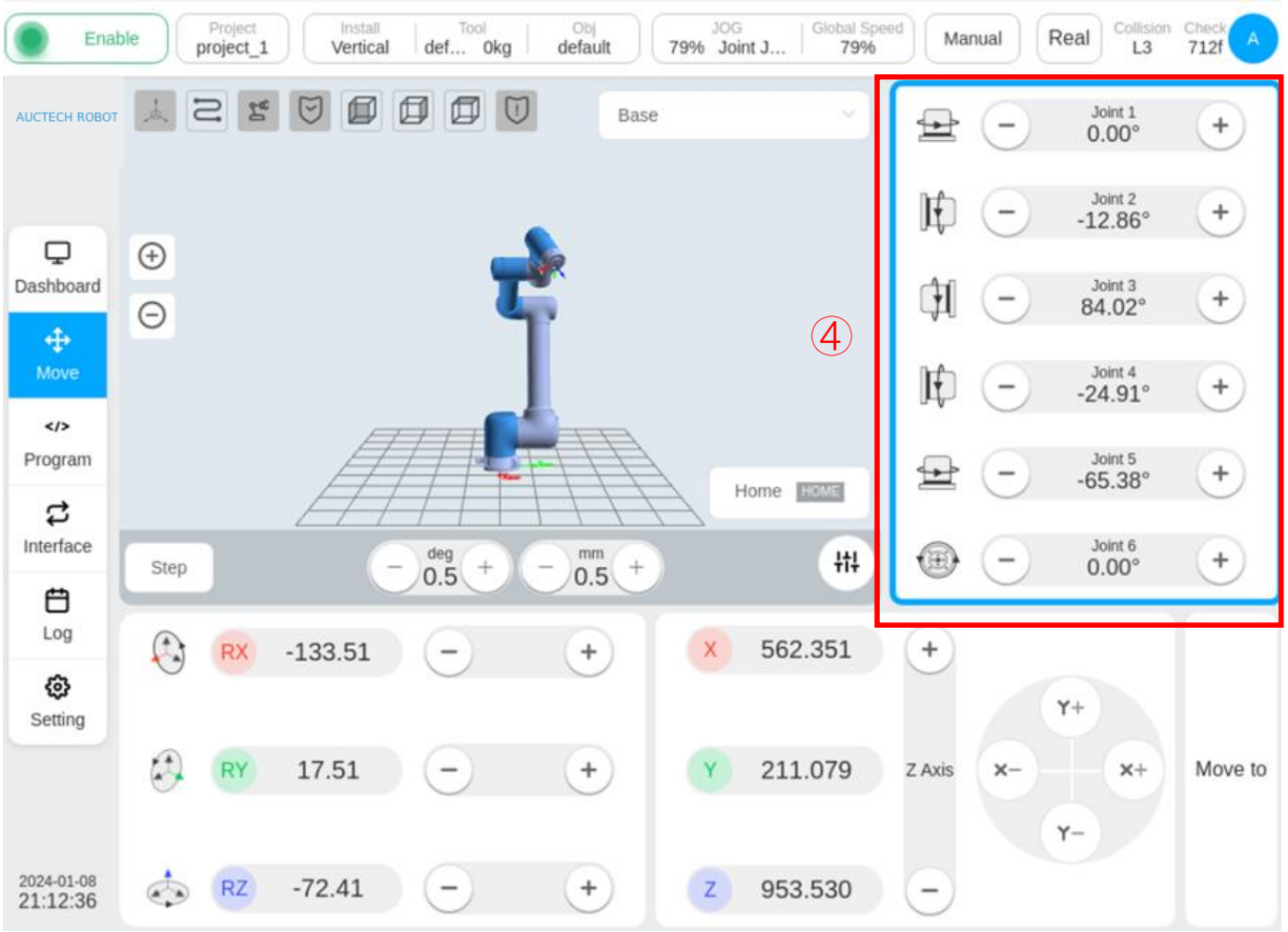
logo③, Pressing and holding the button will move the robot to the set HOME point.

button color

Green at the HOME point

Grey is not at the HOME point

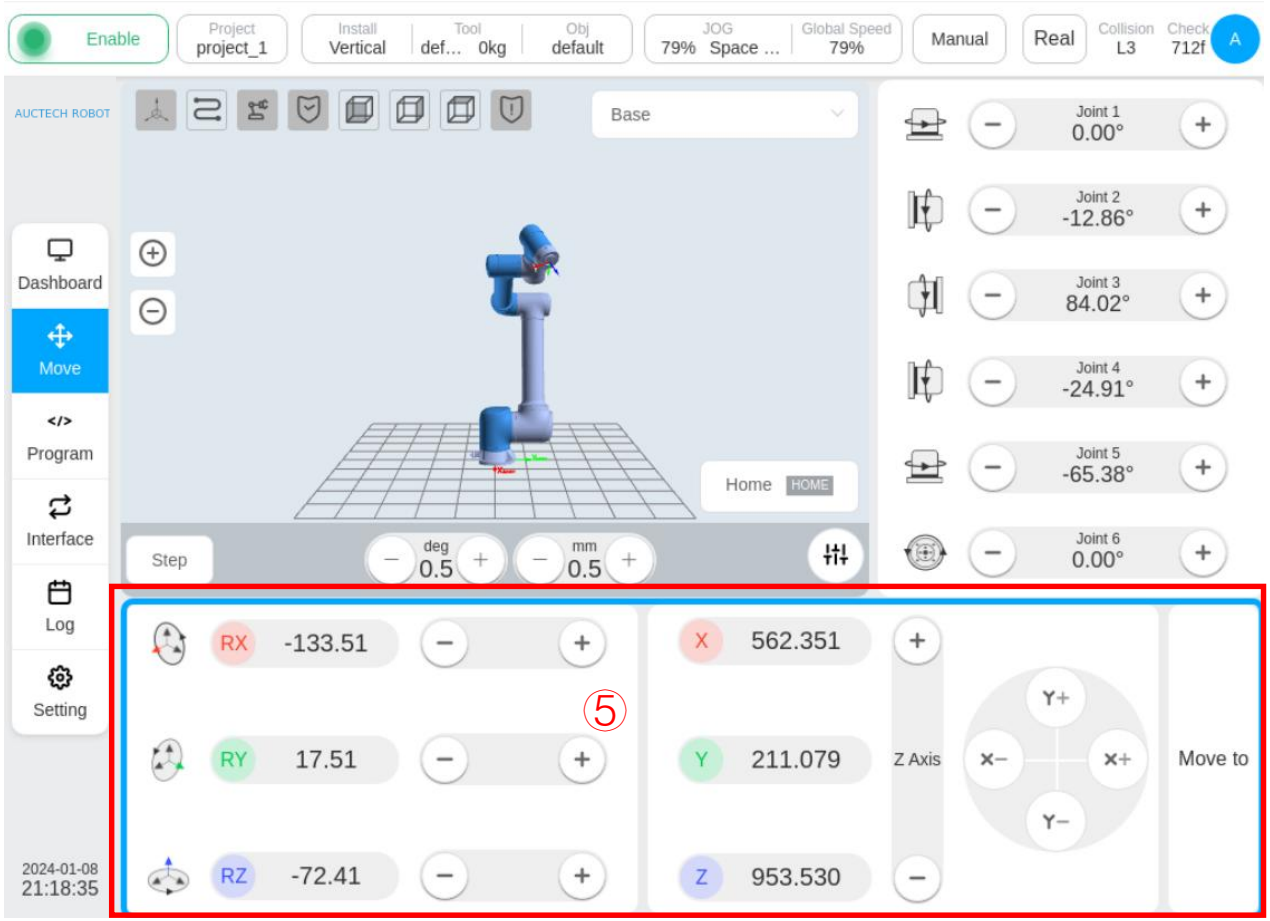
1.4 Joint move



logo ④, Manually operate the area where the robot's joints move.

When you click on a joint to move any area, a blue border appears in that area to select for joint movement.

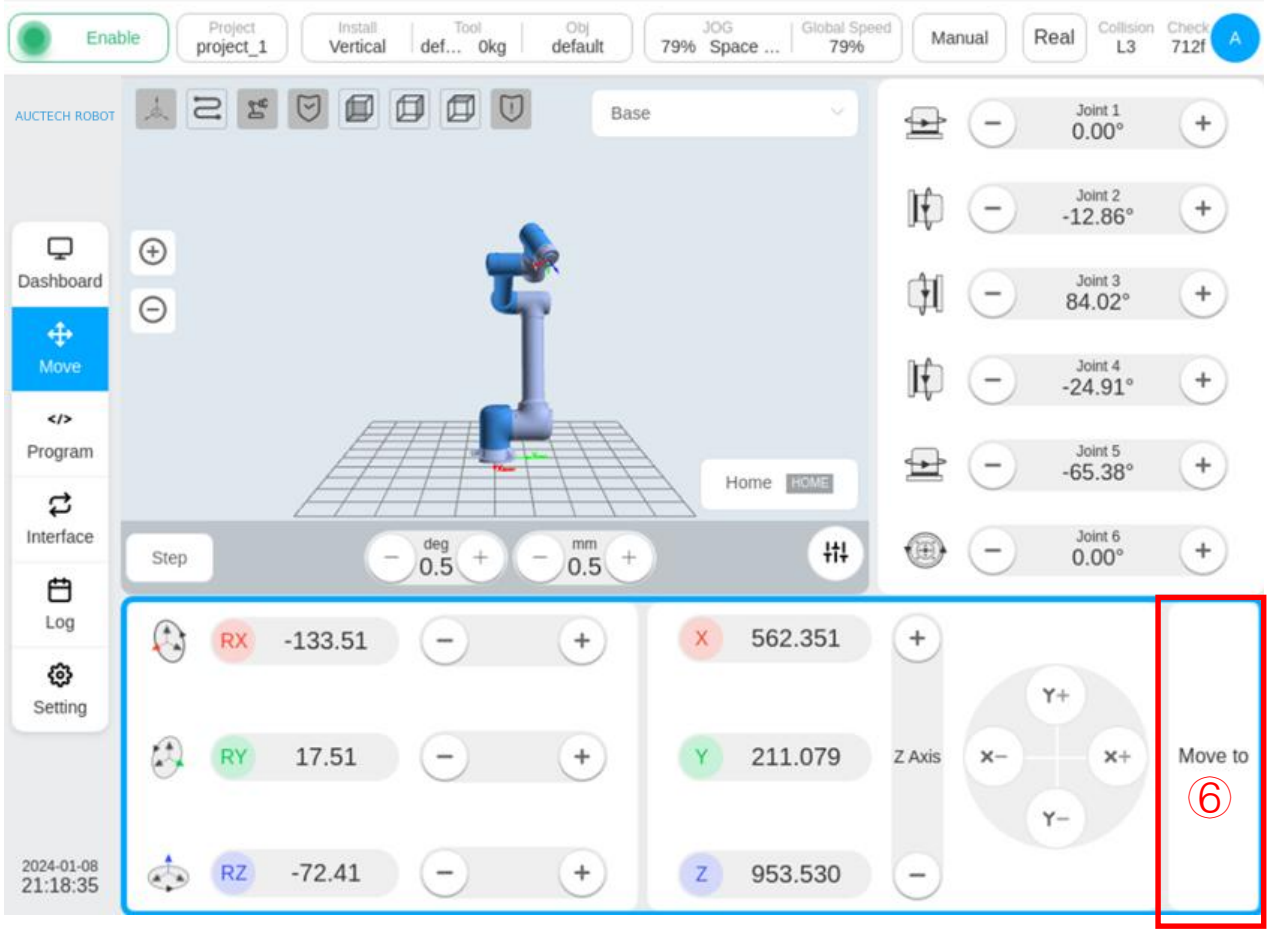
1.5 End movement



logo⑤，Manually operate the area where the end of the robot moves.

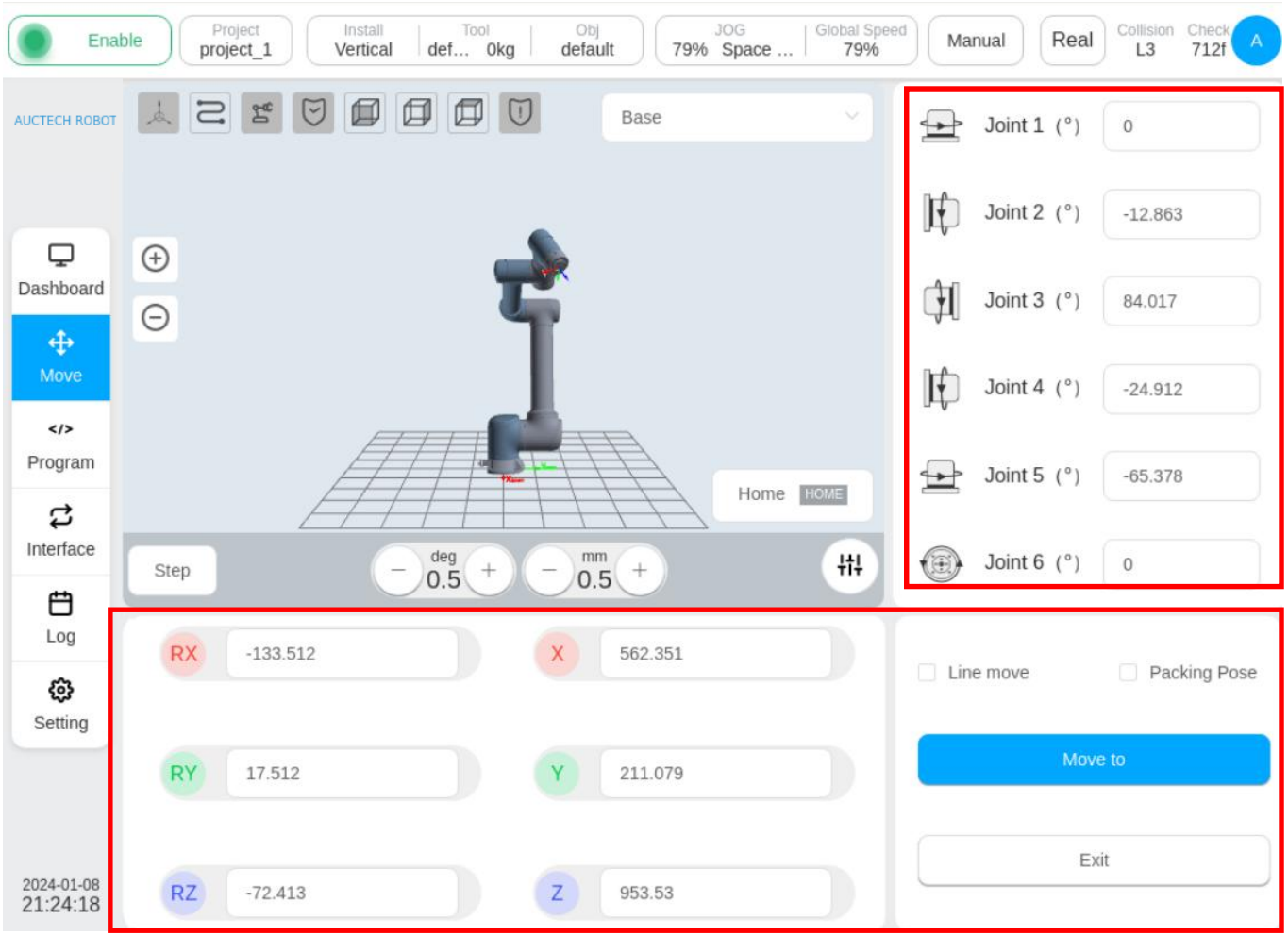
When you click on the end to move any area, a blue border appears in that area, and the end move is selected.

1.6 Move to



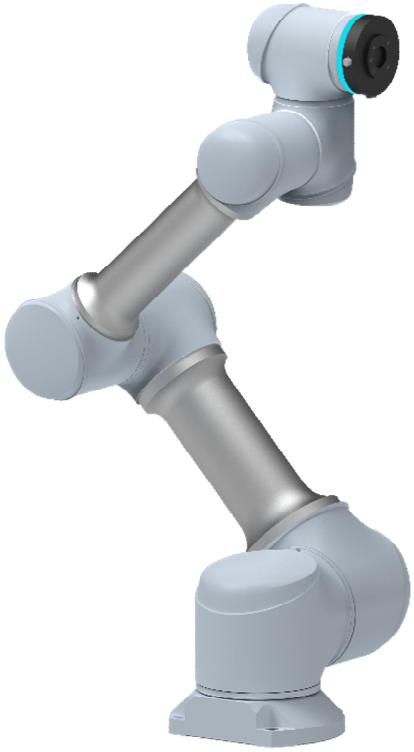
logo⑥， The user manually enters the joint angle or pose value and holds the button to operate the robot to move 。

1.6 Move to



After entering the joint angle or pose value, the position of the input is represented by a ghost in the simulation diagram. After confirming that the location is correct. Long press the "Move to" button.

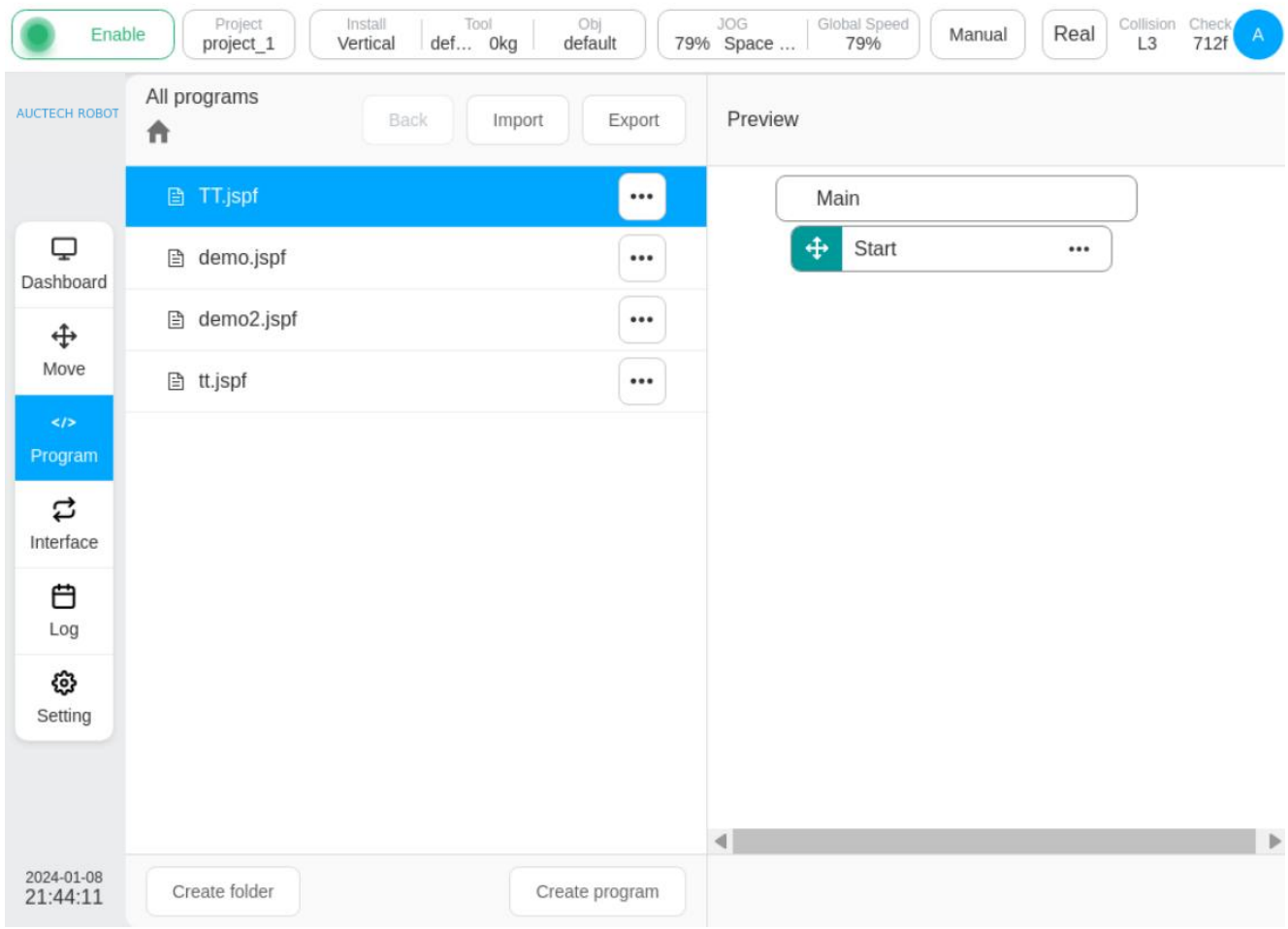
Note: The movement speed of the robotic arm should not be too fast.



Lecture 2

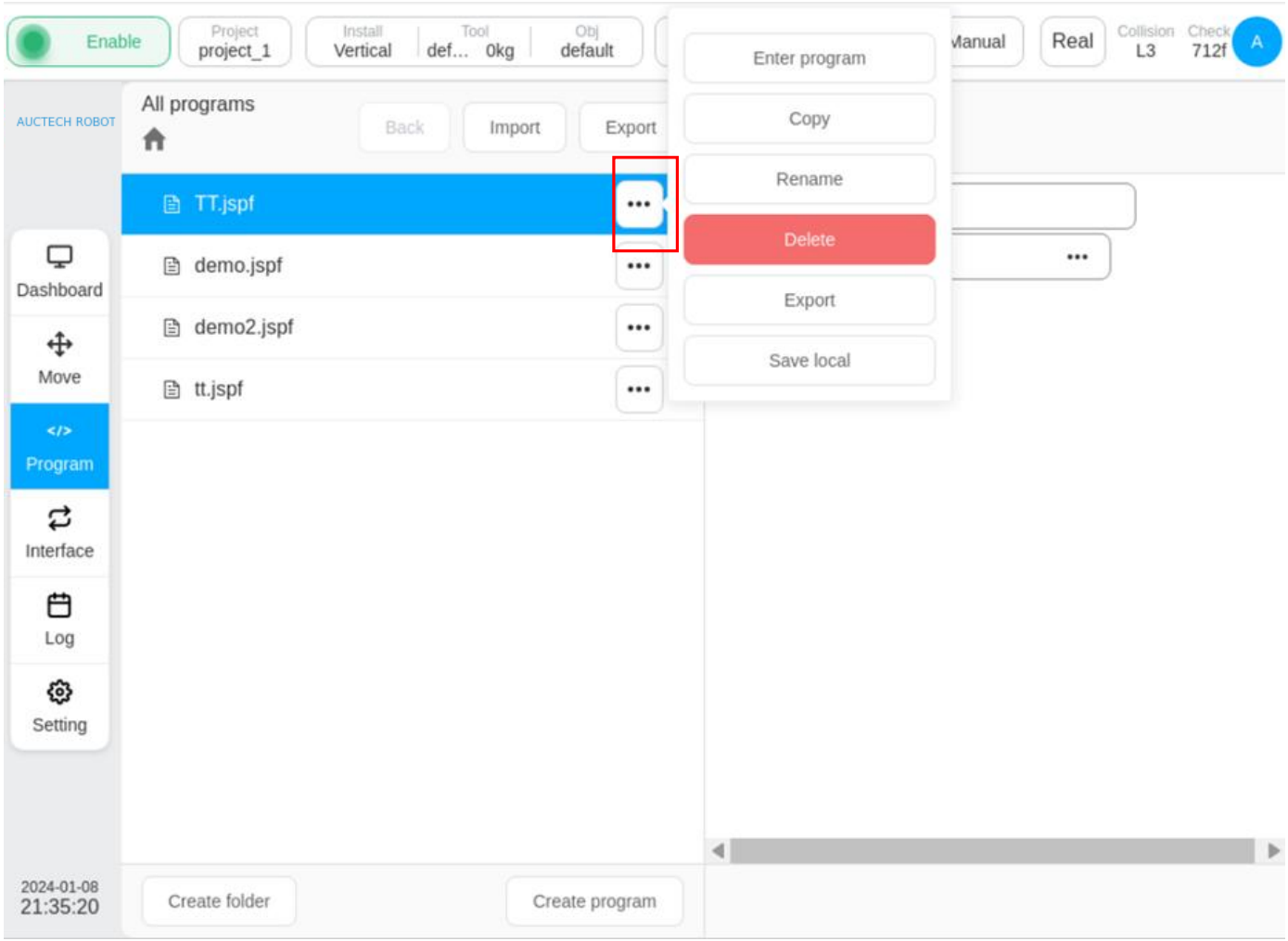
Programming

2.1 List of programs



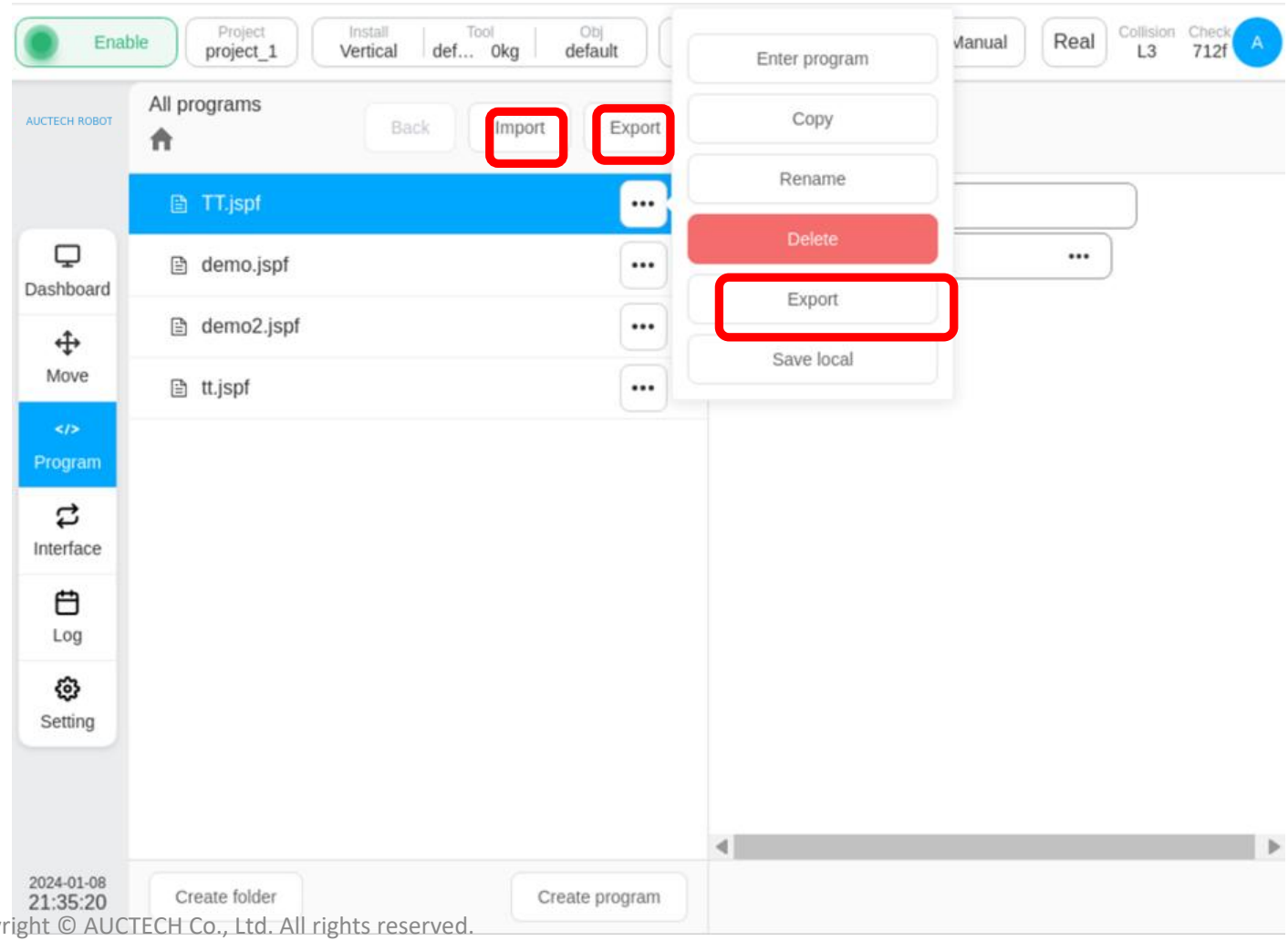
Select: Double-click or check, click "Enter Program" to enter the folder or program.

2.1 List of programs



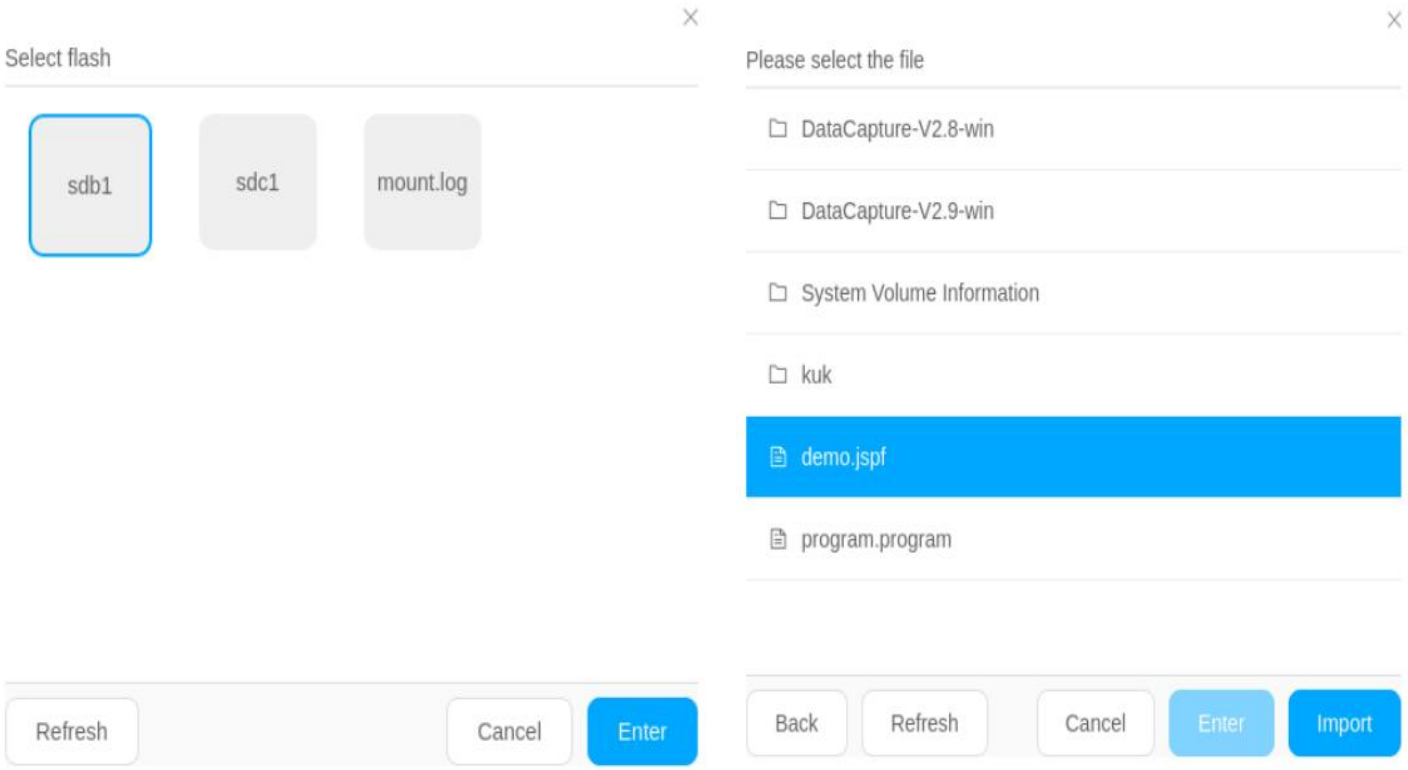
Click on the "...", at the back of the program, to select different operations on the program.

2.2 Importing and exporting programs



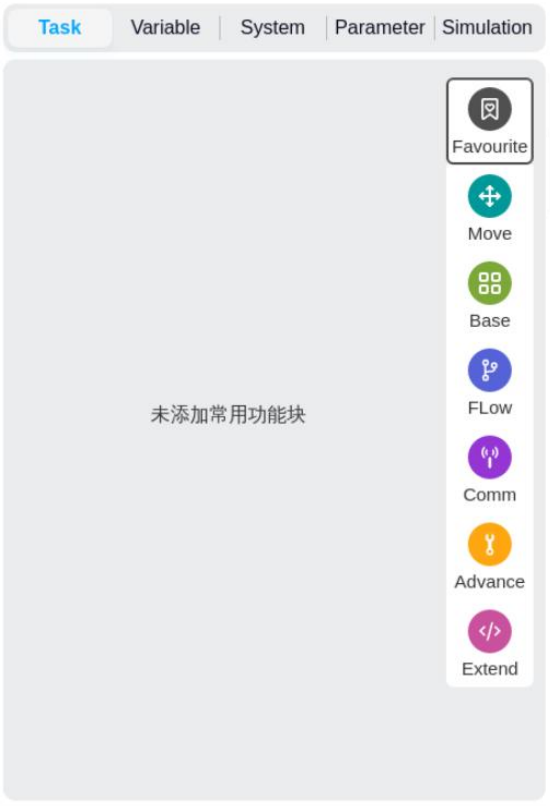
Note: The USB flash drive used for import and export must be in FAT32 format

2.2 Importing and exporting programs

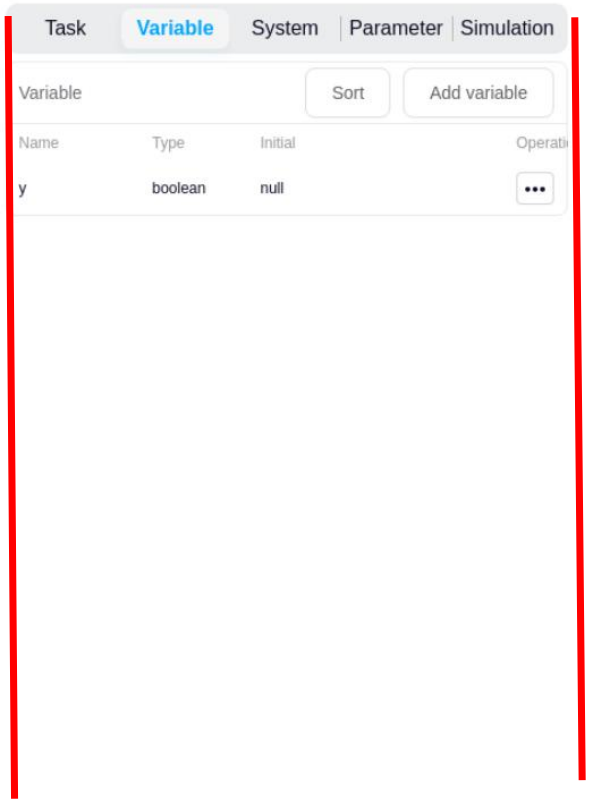


If the name of the program file on the control cabinet is the same as that of the control cabinet during import, the rename keyboard will pop up, and the renamed file will be imported into the control cabinet (the suffix is jspf)

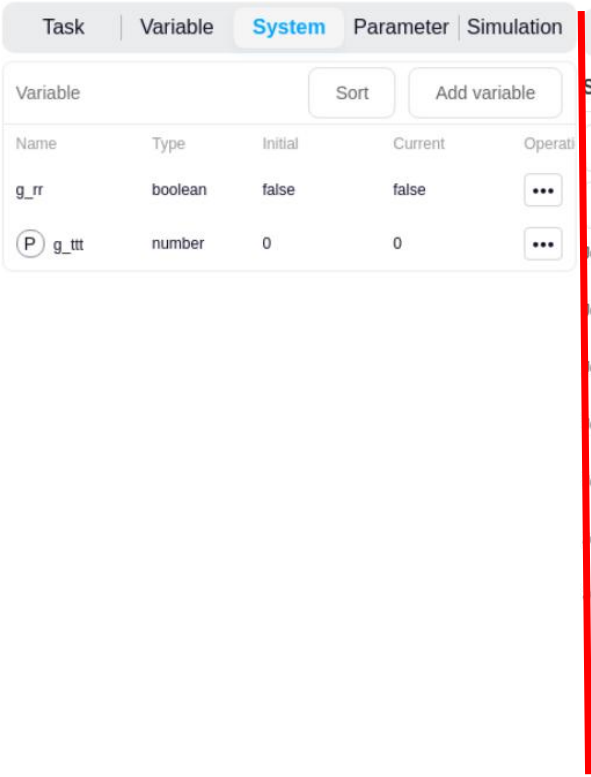
2.3 programming



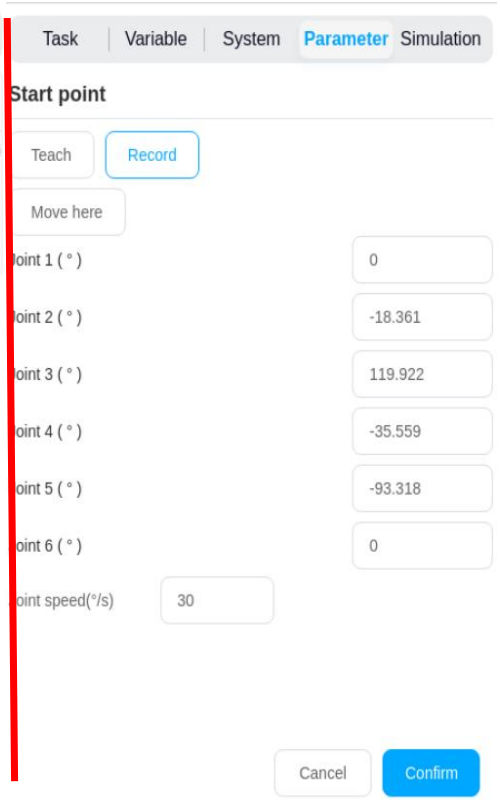
Task



Variable

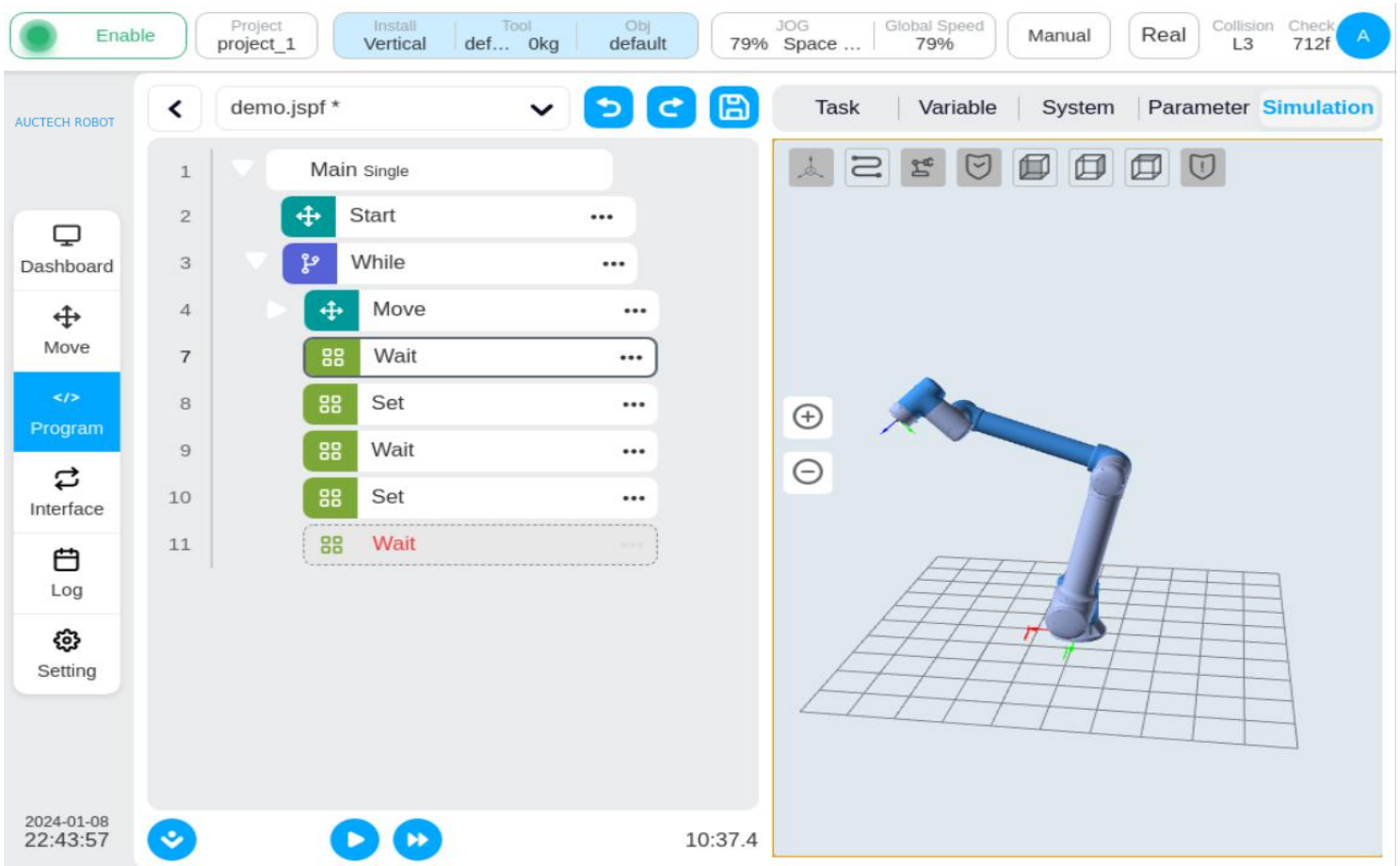


System Variable



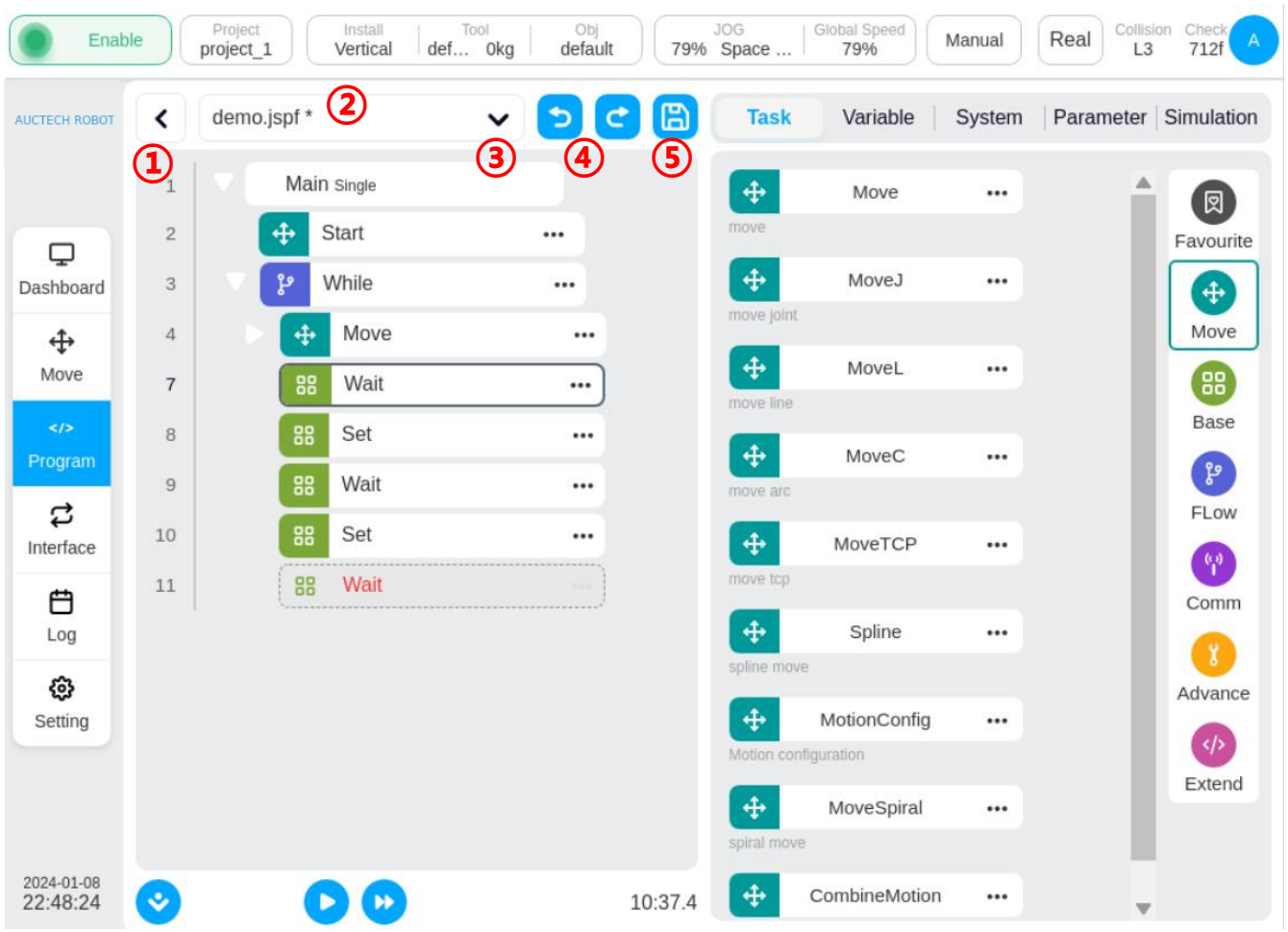
Parameter

2.3 programming



Simulation

2.3 programming



logo①:Returns the list of programs

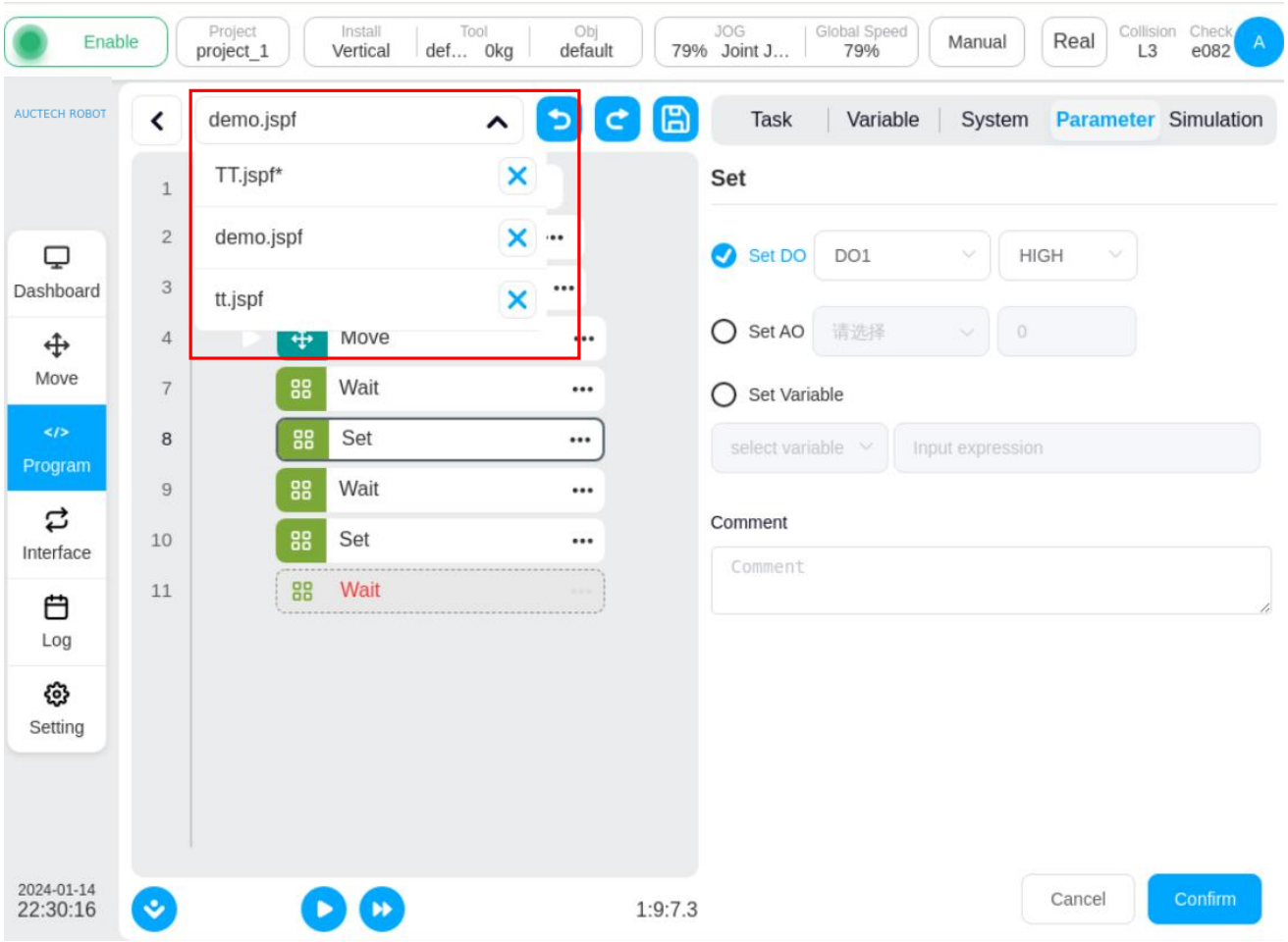
logo② :indicates the current program name

logo③:Expand the list of open programs

Logo④:Undo the changes of the current program

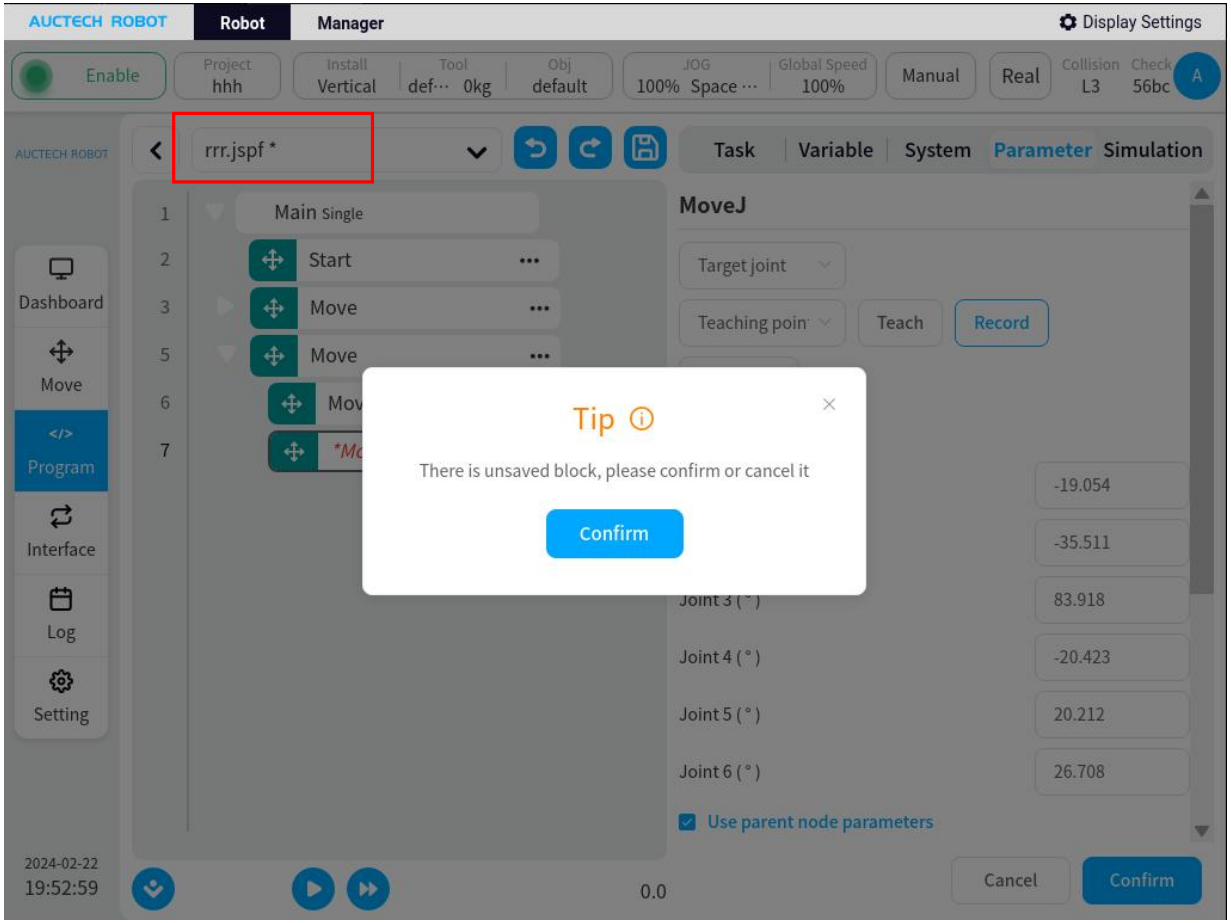
Logo⑤:Save the current program

2.3 programming



Click on the list of open programs and double-click on the program name to switch to that program.

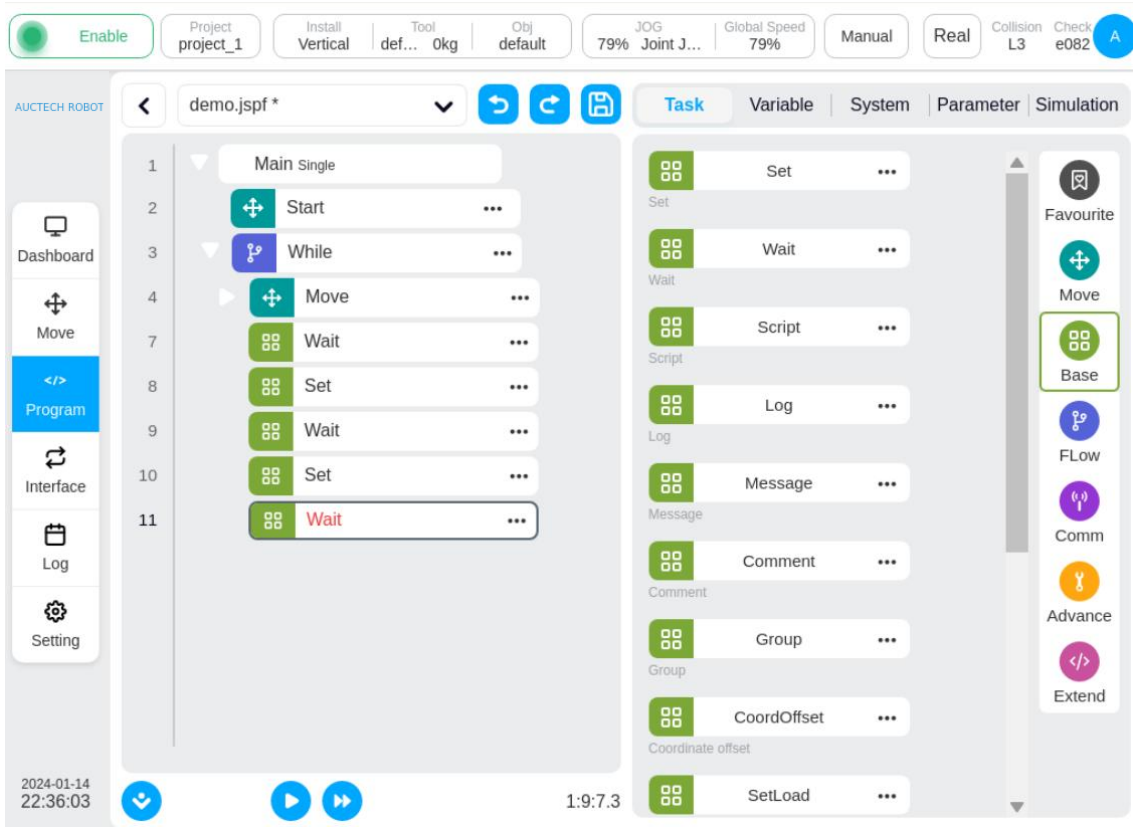
2.3 programming



If there are changes and the program is not saved, it is indicated by an * sign after the program name.

If you select "Close Program", a dialog box will pop up prompting you to save the program or discard the changes

2.3 programming

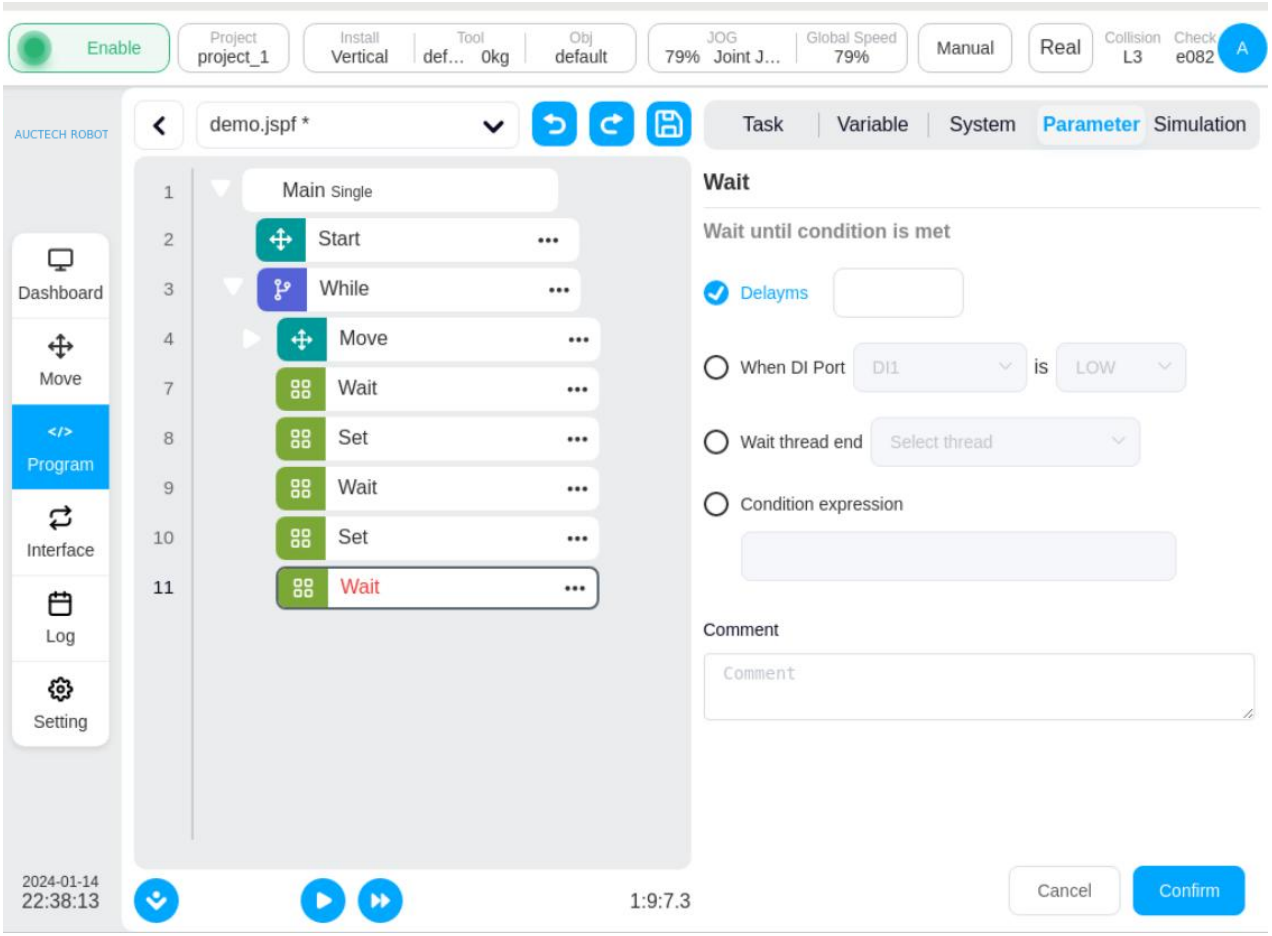


How to add function blocks:

1. Drag and drop the function block to the specified position.

2. Double-click the function block to add

2.3 programming



Configure the parameters of the function block:

(1): Double-click the function block

(2): Select the function block and switch to the parameter configuration area

2.3 programming

The function block in the program tree has the following three states, red indicates that the function block is not configured with valid parameters; The italics plus * indicate that the parameters of the function block have been changed, but they have not been saved to the program tree. The white body indicates that the function block is valid and unchanged

Click on the right side of the function block in the program tree ... The following operation dialog box is displayed, including the following operations:

Cut: Cut the function block. Copy: Copies the function block.

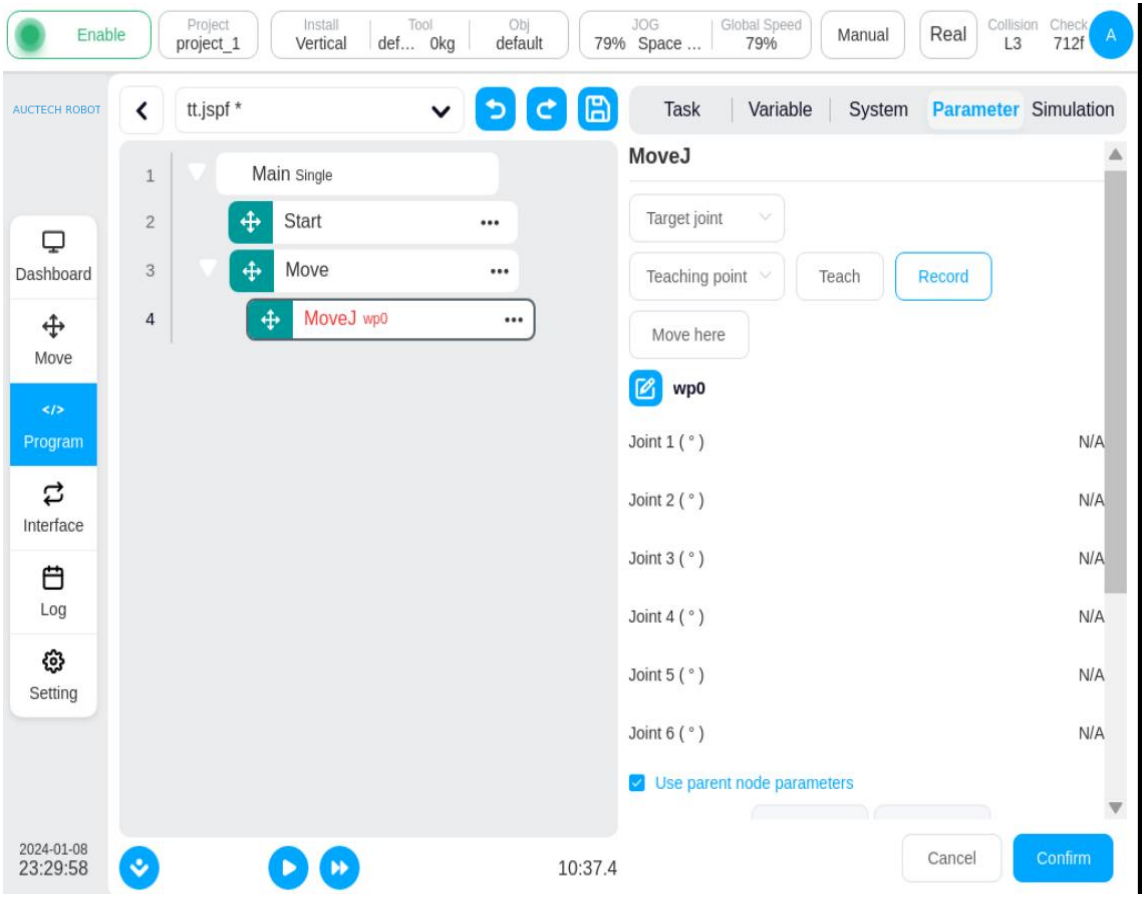
Paste: You can paste the copied function block below the selected function block.

Disable/Enable: Click to choose whether to enable the function block, when you choose not to enable, the function block will be grayed out, and the program will not execute the function block.

Batch Operation: Multiple rows can be selected. Delete: You can delete the function block.

Set as the Start block: Click to select the function block as the start line of run, and there will be an icon in front of the function block

2.4 Instruction blocks -Movej



The robot moves according to the joint movement, and can choose to move to the target joint or target posture.

Configurable parameters:

Target joint: It can be set by teaching or set as a variable, and can be changed manually after teaching.

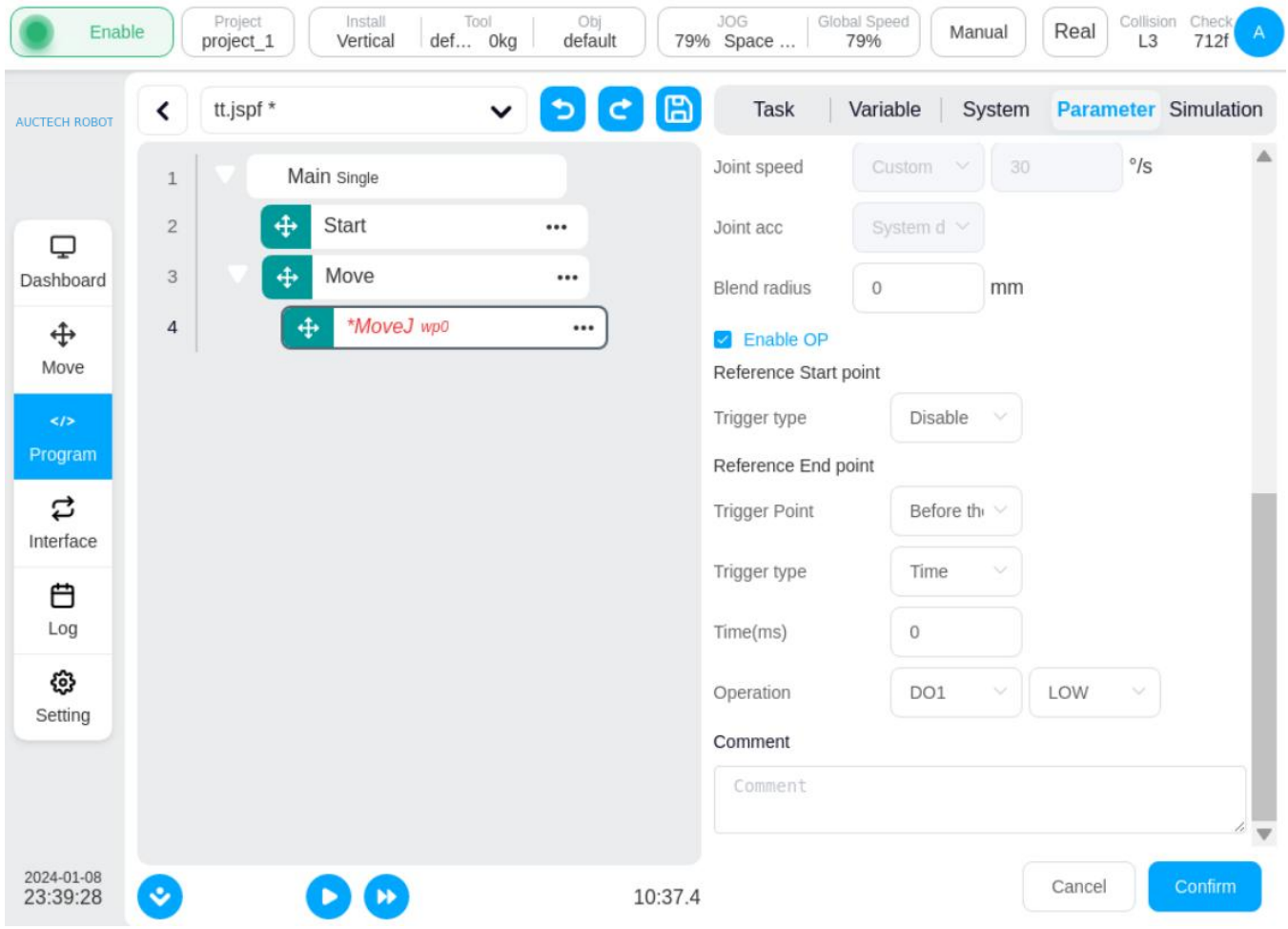
Move a point: You can set the point as a variable.

Use Parent Node Parameters: If selected, the function block uses the joint angular velocity and joint angular acceleration parameters set by the Move function block of the parent node. If this option is not selected, you need to set the joint angular velocity and joint angular acceleration for the function block, which is selected by default

Joint angular velocity: unit°/s; Joint angular acceleration: unit°/s2 Fusion radius: The unit mm0 indicates no fusion

Enable OP: The OP function allows you to set the status of the universal digital output during trajectory execution.

2.4 Instruction blocks -Movej



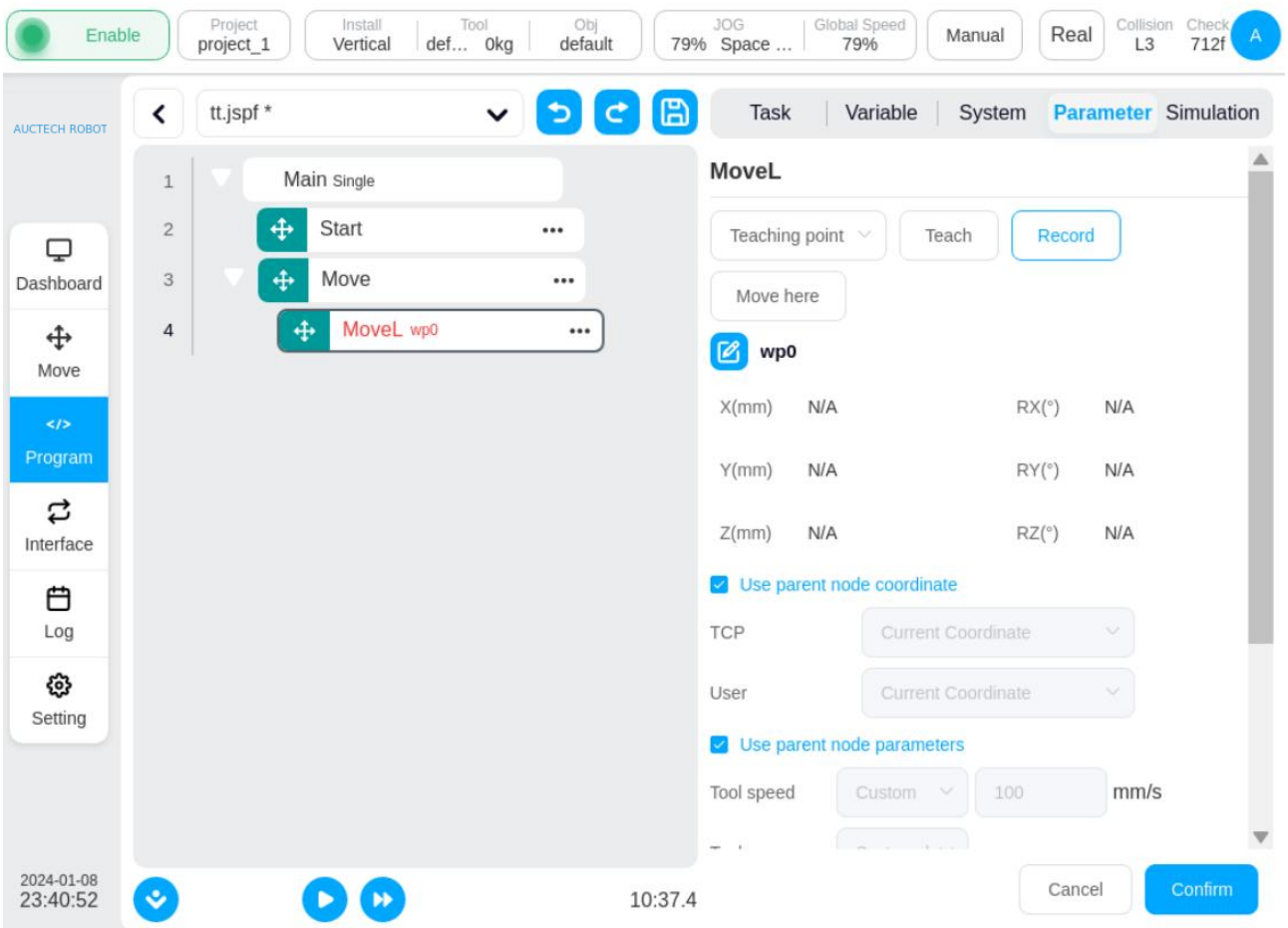
If OP is enabled, you need to configure the following:

It can be triggered after the track starts and before the track ends

Trigger Type: You can select no trigger or time trigger Trigger a delay

Output Port: Select the port and port status

2.4 Instruction blocks -MoveL



The robot moves to the target attitude in a straight line, and the parameters can be set:

Enable OP: The OP function allows you to set the status of the universal digital output during trajectory execution.

Move the setting point: You can set the point as a variable, and you can manually change it after the variable teaching is set.

Use Parent Node Coordinate System: If selected, the parameter uses the reference coordinate system set by the Move function block of the parent node, which is selected by default.

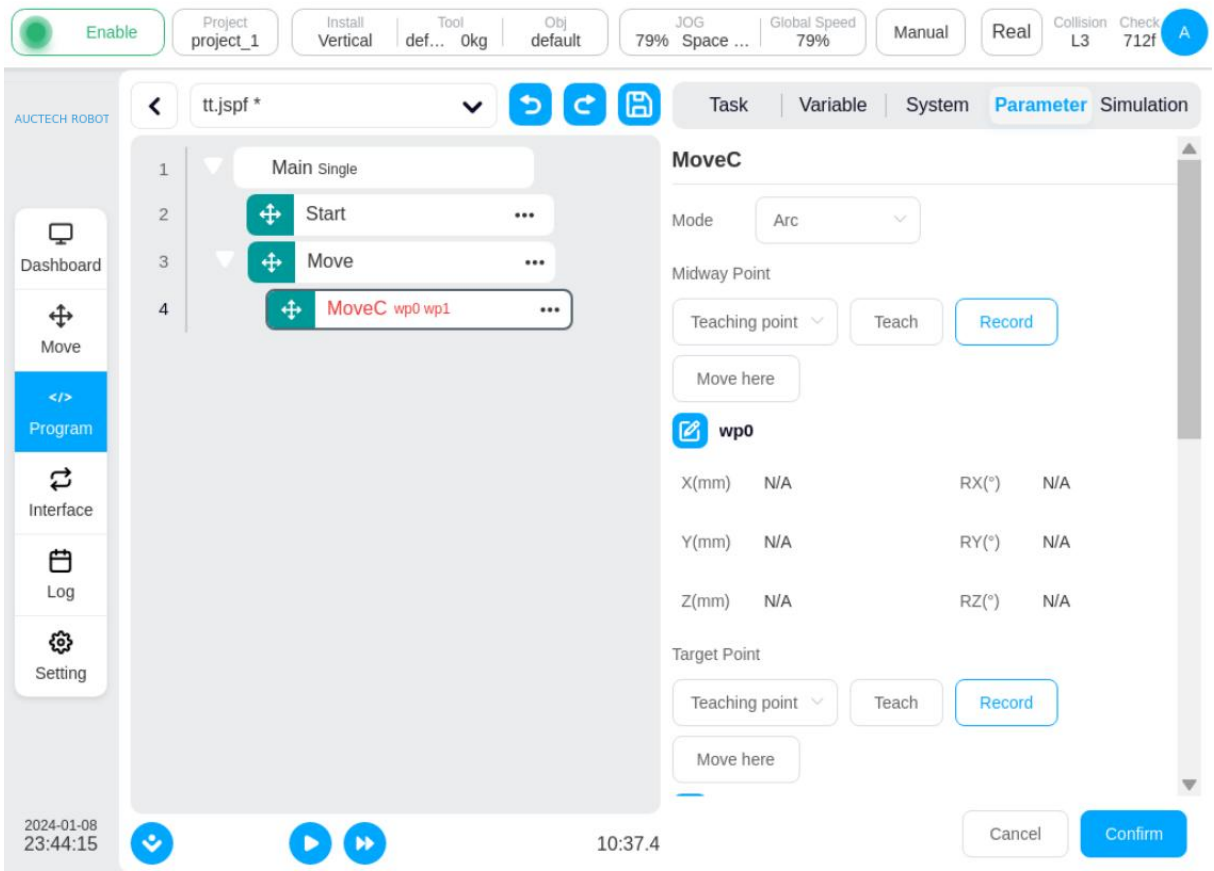
Use the parent node parameter: the same as the description in Movj, and do not repeat it.

End velocity: in mm/s

End acceleration: in mm/s2

Fusion radius: in mm, 0 means no fusion

2.4 Instruction blocks -MoveC



The robot moves in an arc or a circle, and the parameters can be set:

Mode: Arc or Circle

Intermediate point: It can be set or set as a variable by teach-in, and can be changed manually after teach-in.

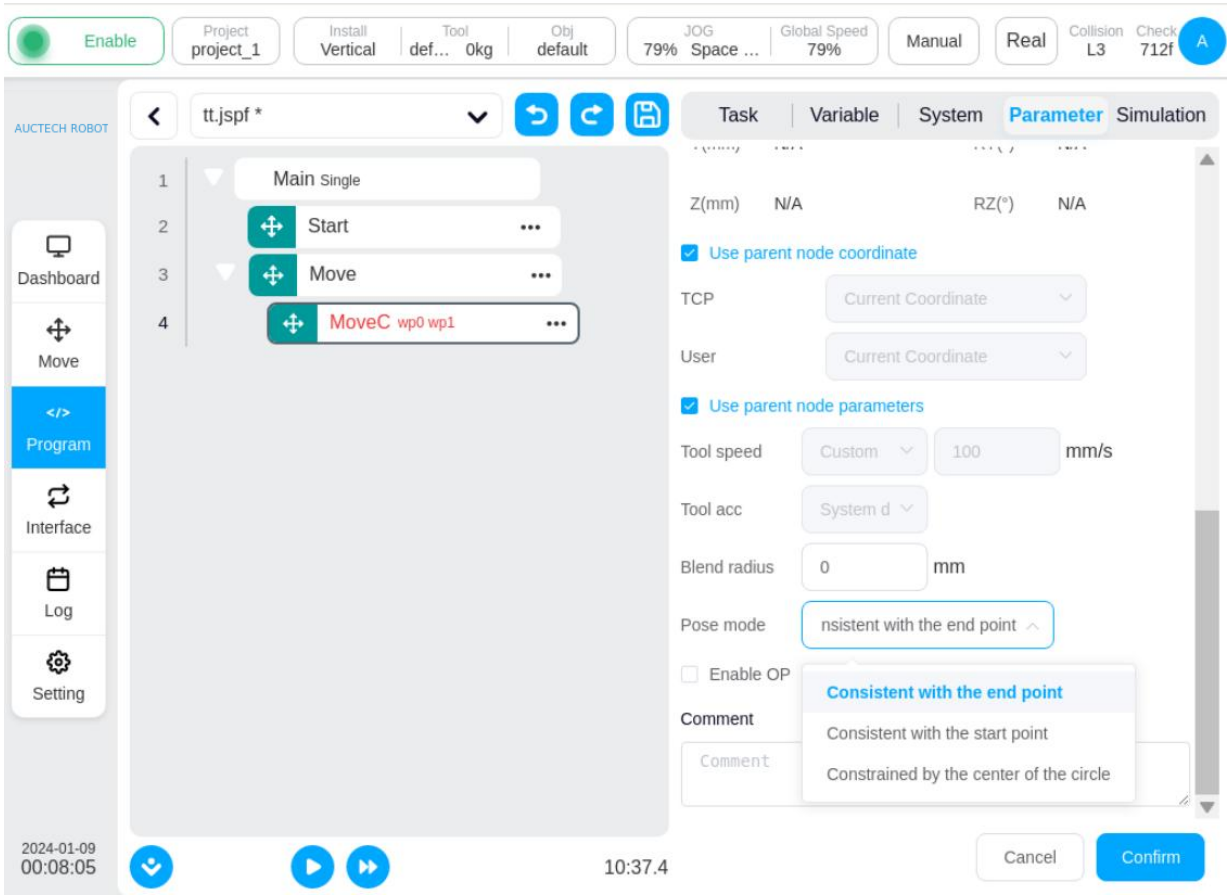
Target point: It can be set or set as a variable by teaching, and can be changed manually after teaching.

Use the coordinate system of the parent node: the same as the description at MovL

Use the parent node parameter: the same as the description in Movj

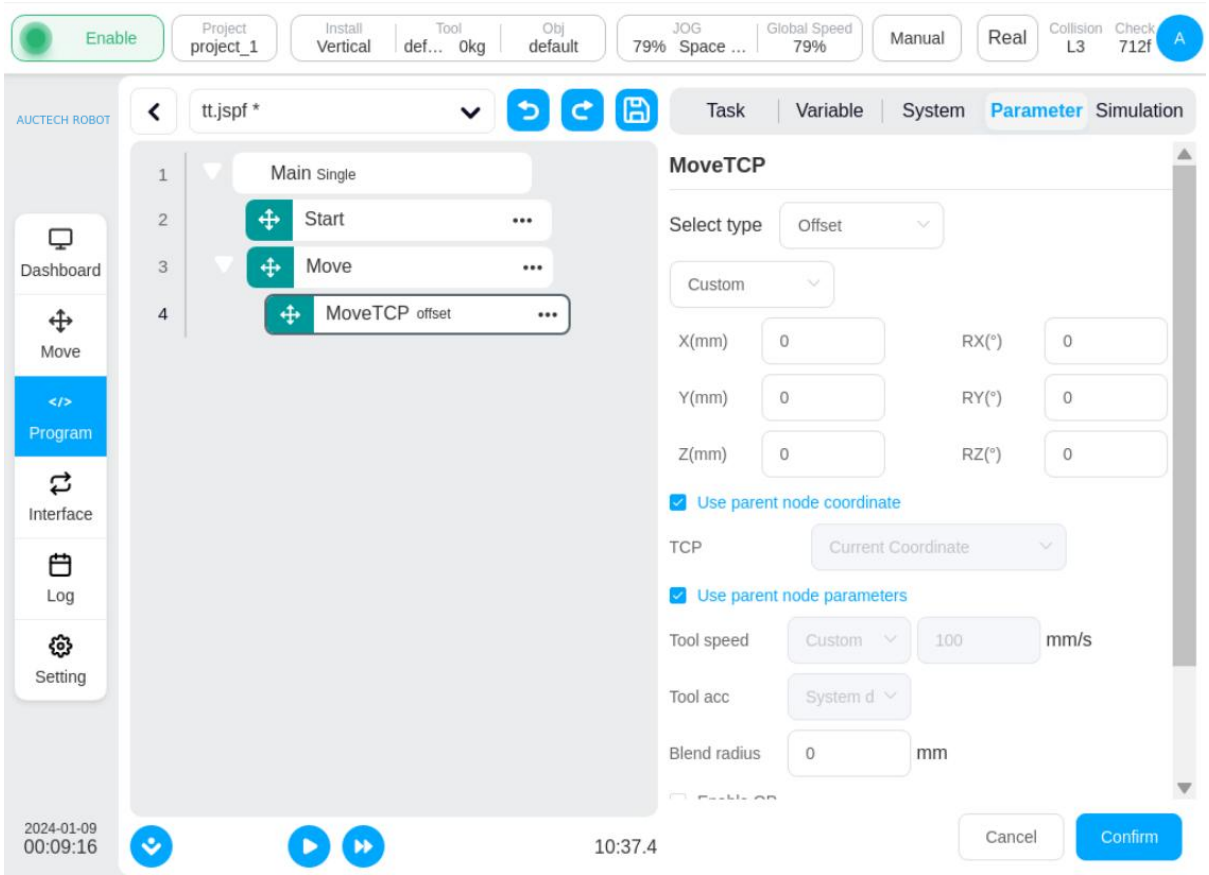
Enable OP: Same as described in Movj

2.4 Instruction blocks -MoveC



There are three modes of attitude control: consistent with the end point, consistent with the starting point, and constrained by the center of the circle.

2.4 Instruction blocks -MoveTCP



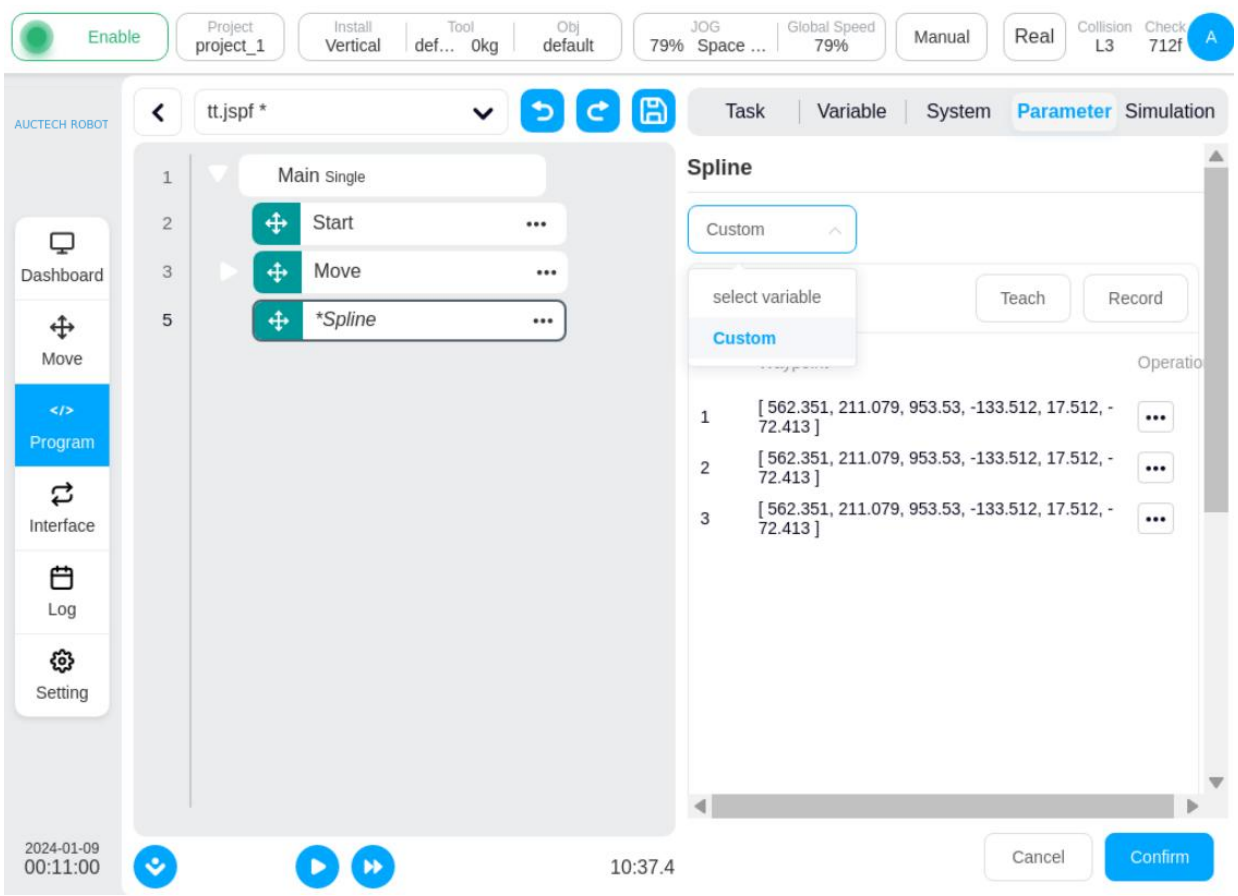
The robot moves along the tool's coordinate system.

You can enter offsets in each direction directly, or you can use variables. You can teach two points and use the offset between the two points as the offset value for the movement. Use the parent coordinate system: Same as described at MovL

Use parent node parameters: same as described at Movj

Enable OP: Same as described at Movj

2.4 Instruction blocks -Spline



Controls the end of the robot to move along a spline curve.

Spline waypoints: Add points by teaching or selecting variables, and generate splines based on points

Reference Coordinate System: When teaching the point, the default is the current tool and workpiece coordinate system, which can be changed manually

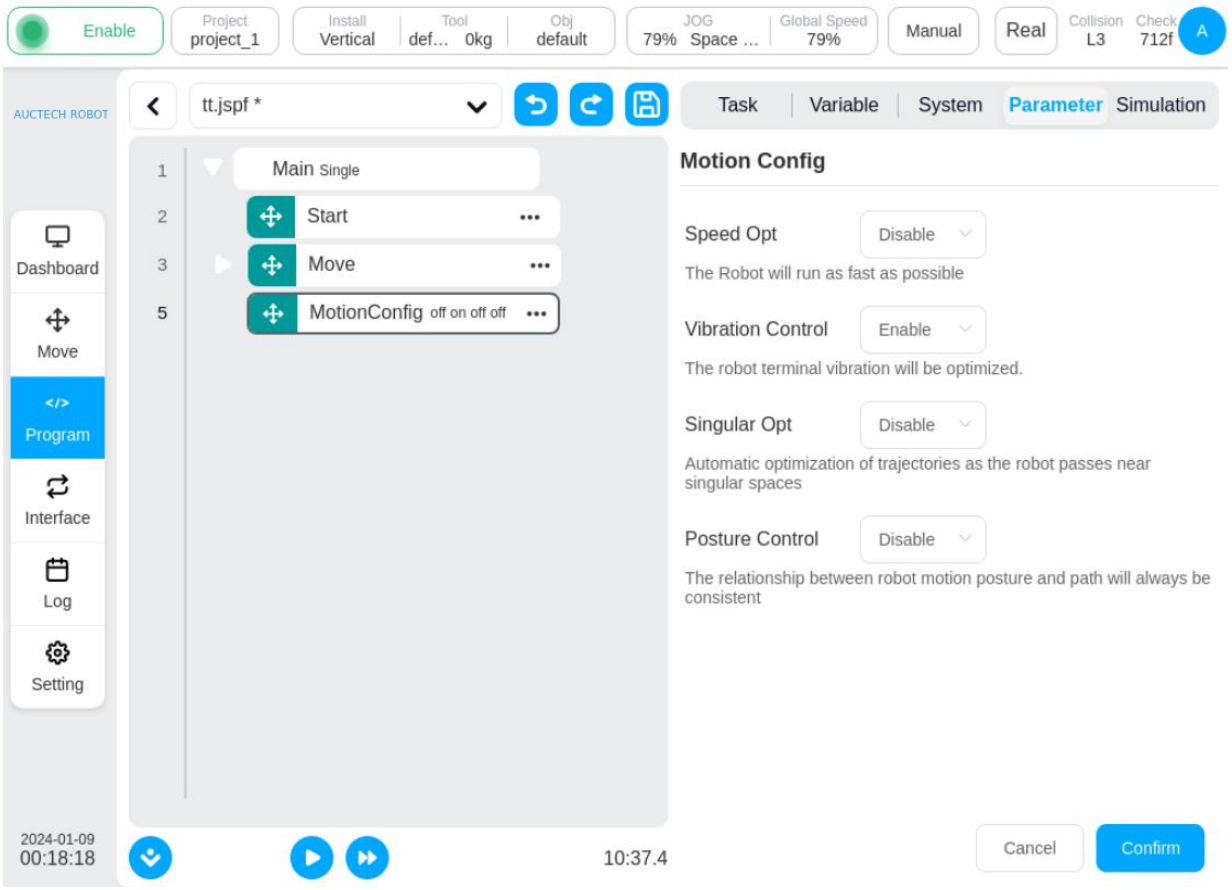
The robot moves along the tool's coordinate system. You can enter offsets in each direction directly, or you can use variables. You can teach two points and use the offset between the two points as the offset value for the movement.

Use the parent coordinate system: Same as described at MovL

Use parent node parameters: same as described at Movj

Enable OP: Same as described at Movj

2.4 Instruction blocks -MotionConfig



(1): Turn motion-related configurations on or off.

Options:

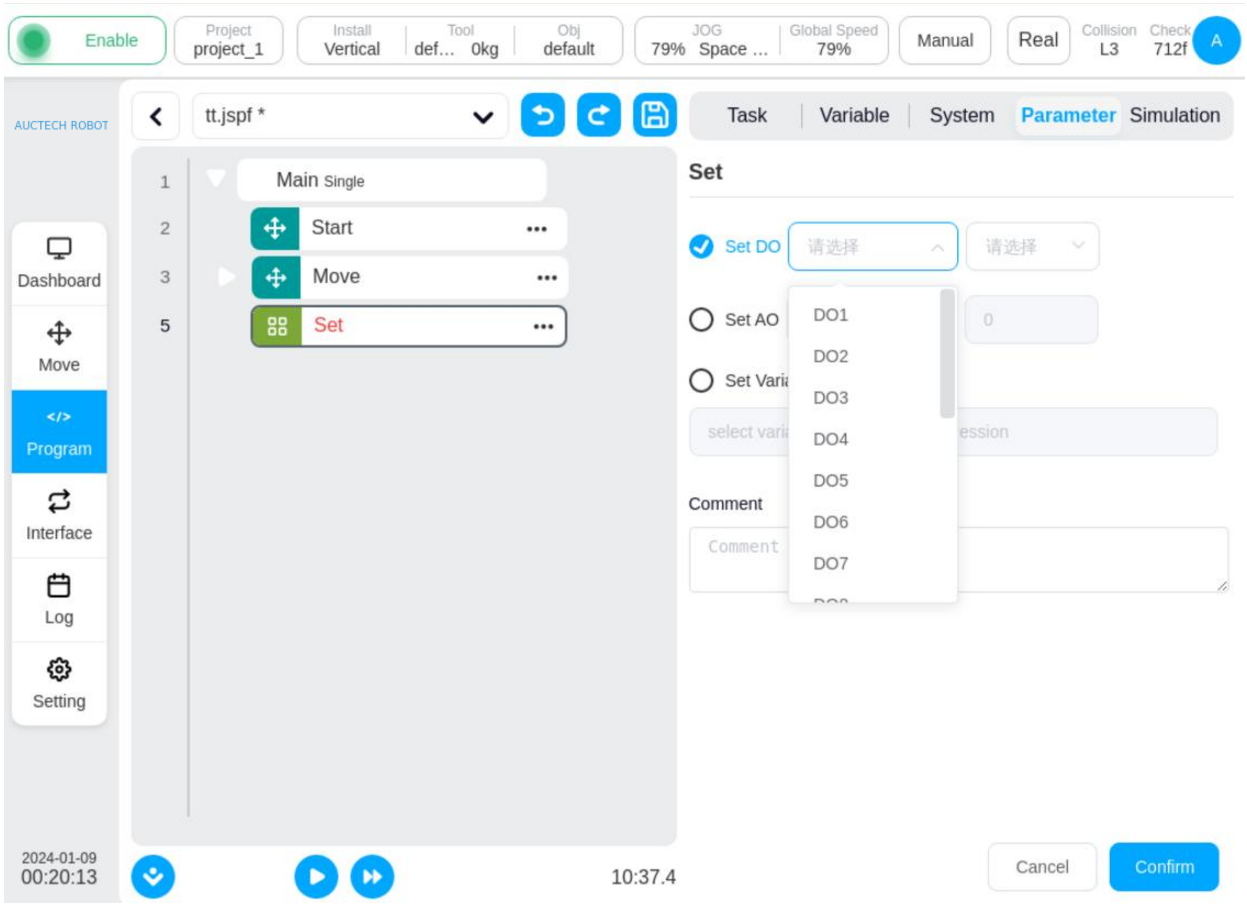
Speed optimization: When enabled, the robotic arm will follow the path at the highest possible speed while meeting the constraints of the system.

(2): Acceleration optimization: After enabled, the acceleration parameters in the script will be automatically optimized, and the acceleration will not be strictly followed.

(3): Vibration control: After it is turned on, it will be optimized for the vibration at the end of the robot.

(4): After enabled, the trajectory will be automatically optimized when the robot passes near the singular space.

2.5 Instruction blocks -Set

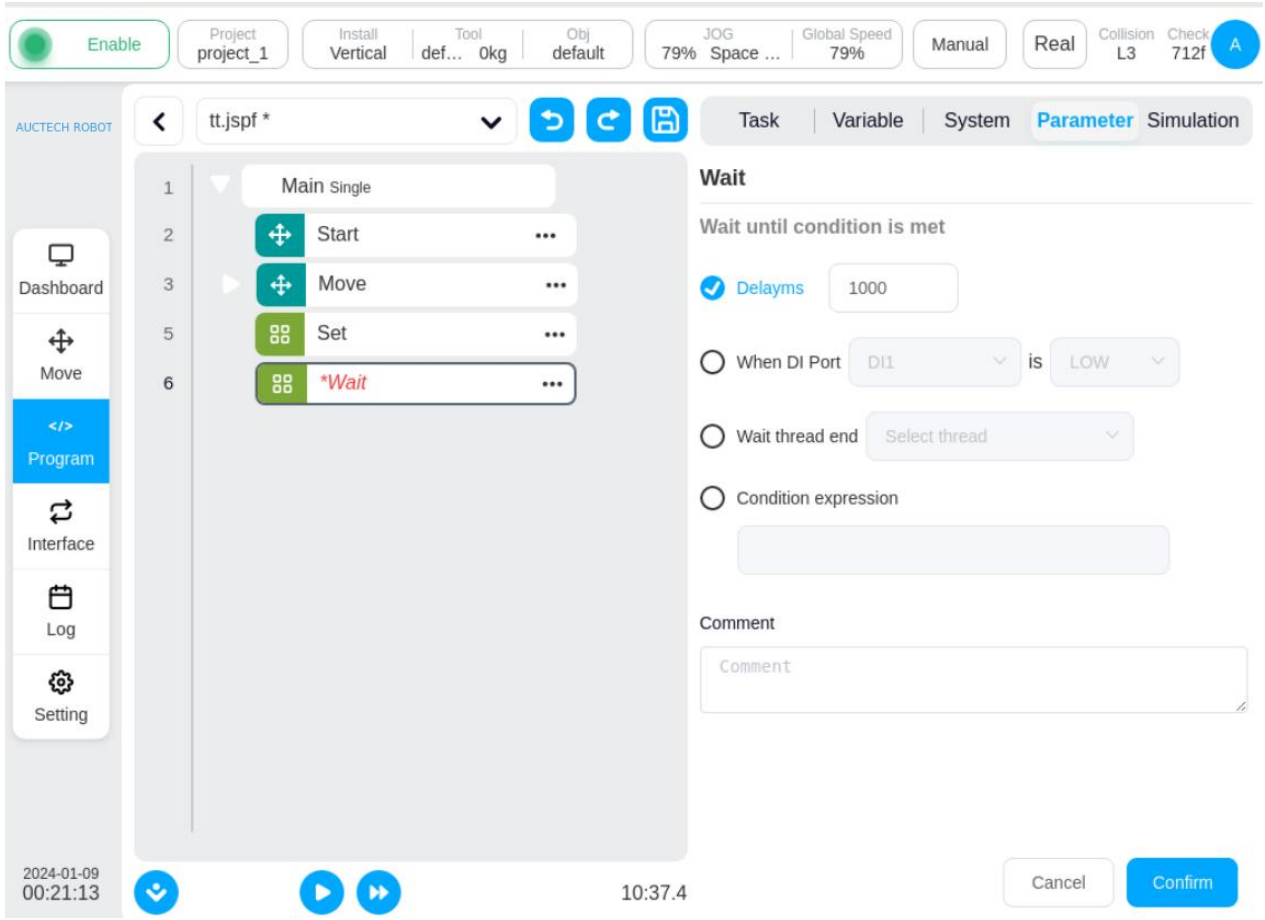


(1): The digital output port of the control cabinet and the digital output port at the end of the robotic arm can be set to high level or low level.

(2): Assign values to program variables or system variables.

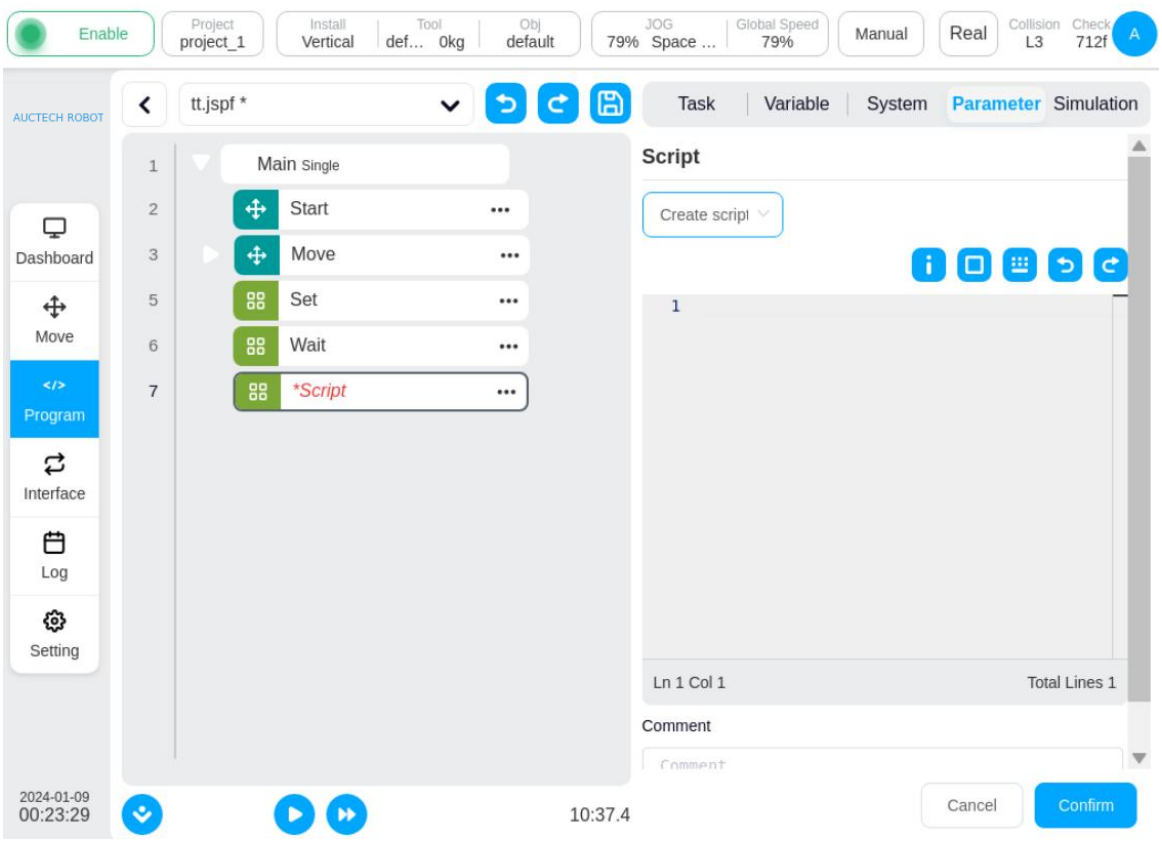
(3): Set the analog output

2.5 Instruction blocks -Wait



Optionally, wait for a period of time, wait for DI signals and expressions, and the program will wait until the set conditions are met when executing to this function block.

2.5 Instruction blocks -Script



You can choose an expression, a script, or a script file.

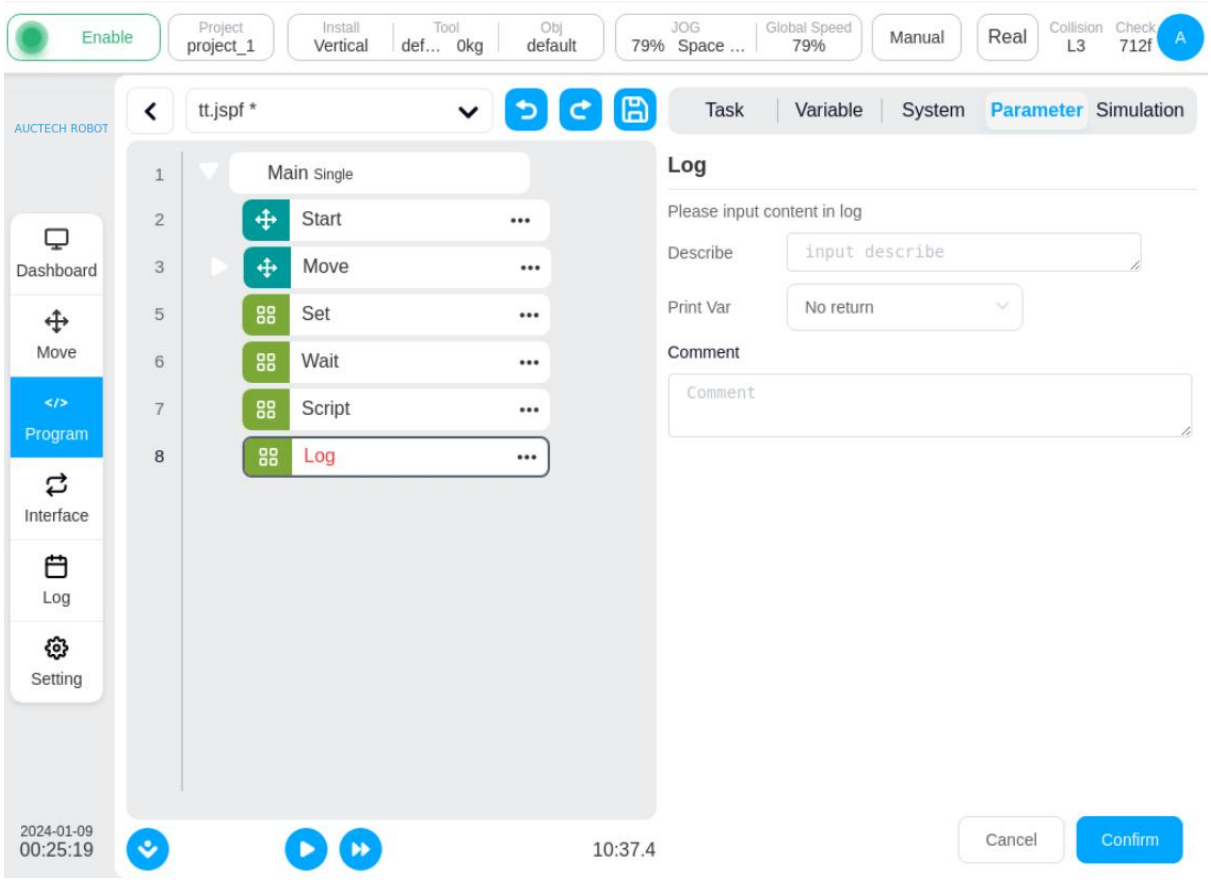
(1): Expressions can be used to create a line of scripts using the expression editor, and for the input of expressions, please refer to Section 8 Input Keyboard (Expression Input Keyboard);

(2): Scripts can be used to write entire paragraphs of script code;

(3): Script file can select a script from the file to a script file.

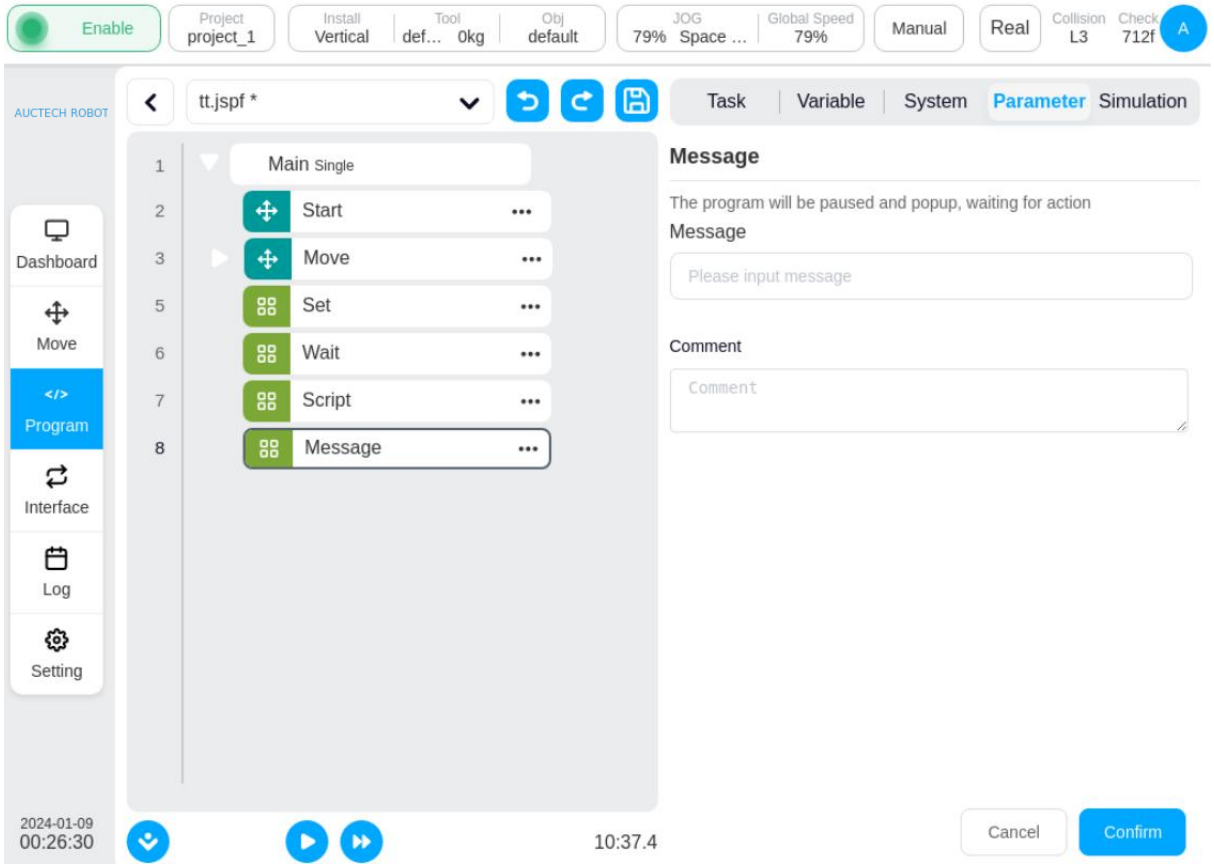
For more information about the script code, see the Scripting Manual.

2.5 Instruction blocks -Log



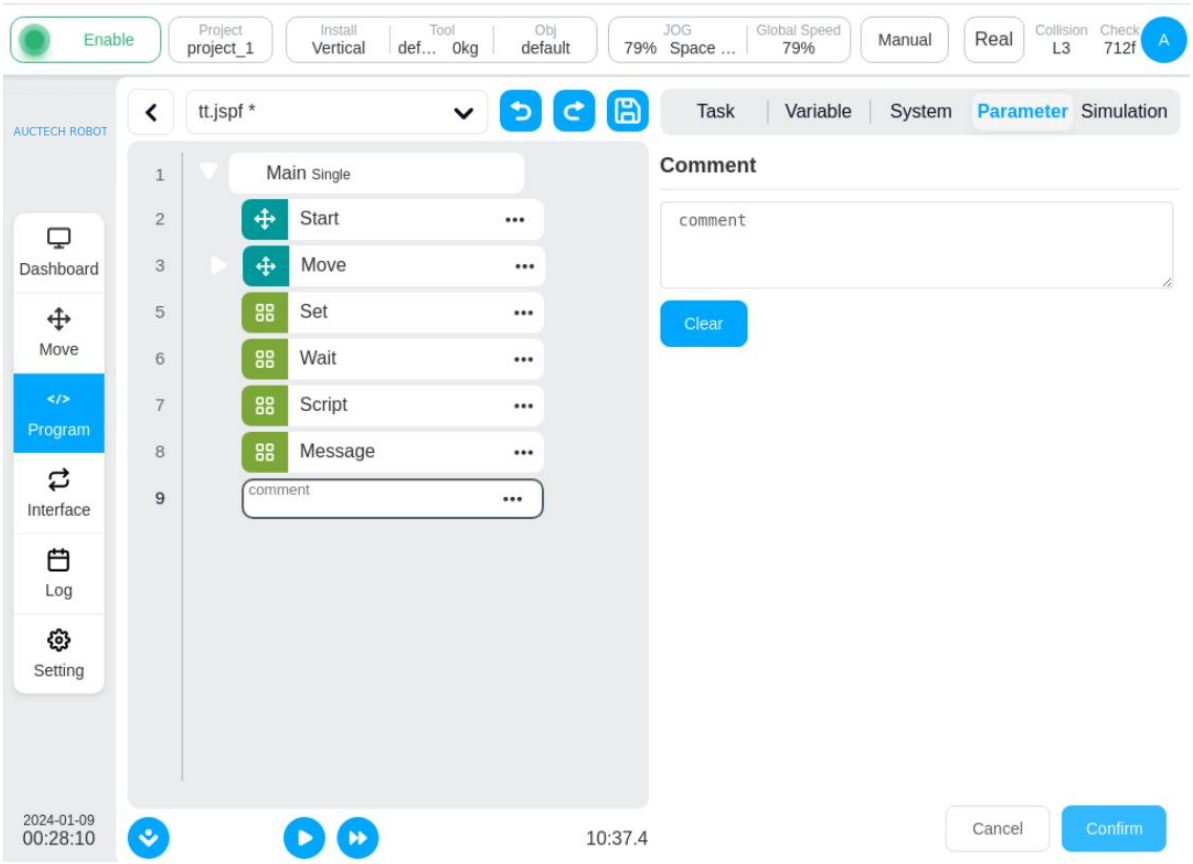
Log function block. You can print the values of messages or variables to a log file.

2.5 Instruction blocks -Messge



Message pop-up function block. A message can be set, and when the program runs to this function block, a dialog box can pop up to display the message and the program is suspended, and the user can choose to stop or continue running the program.

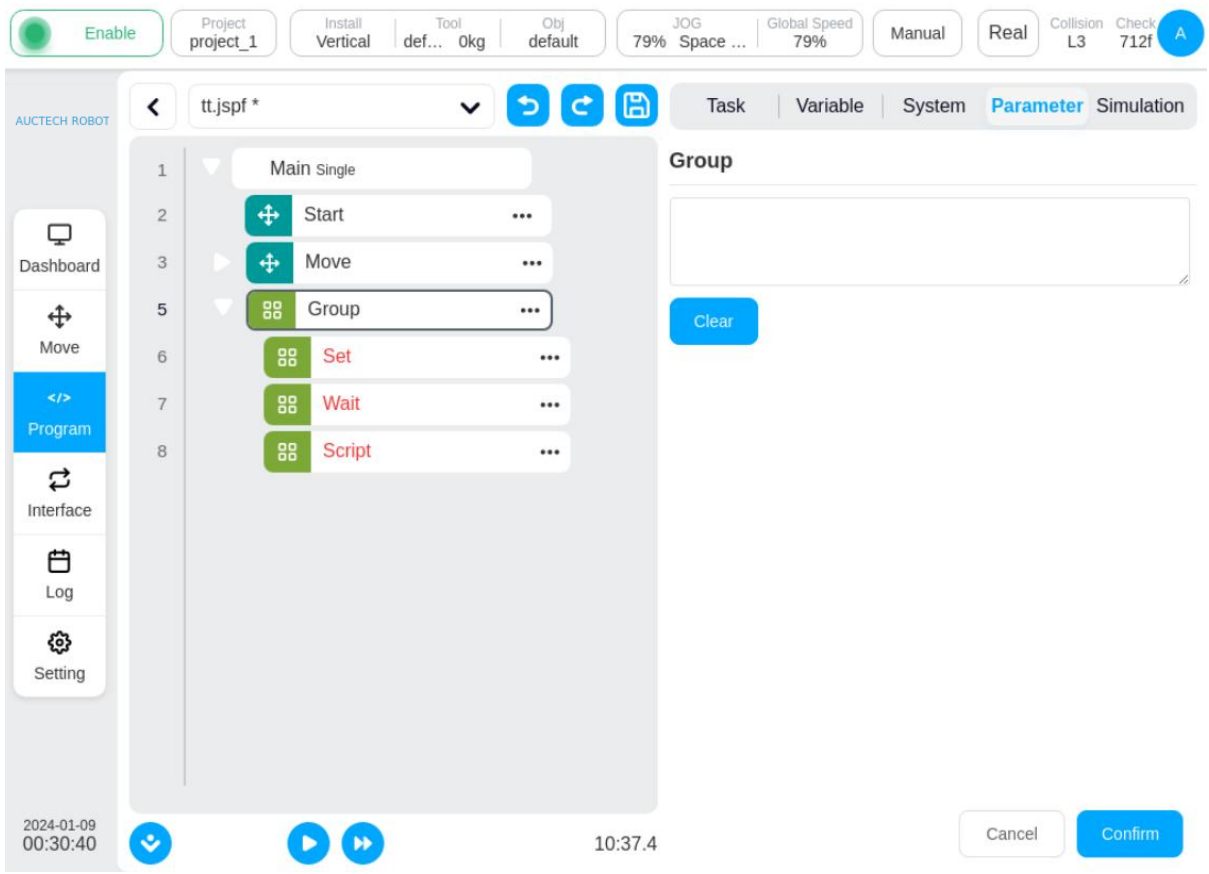
2.5 Instruction blocks -Comment



Annotate function blocks.

Add a comment to the program tree. While the program is running, this function block does nothing.

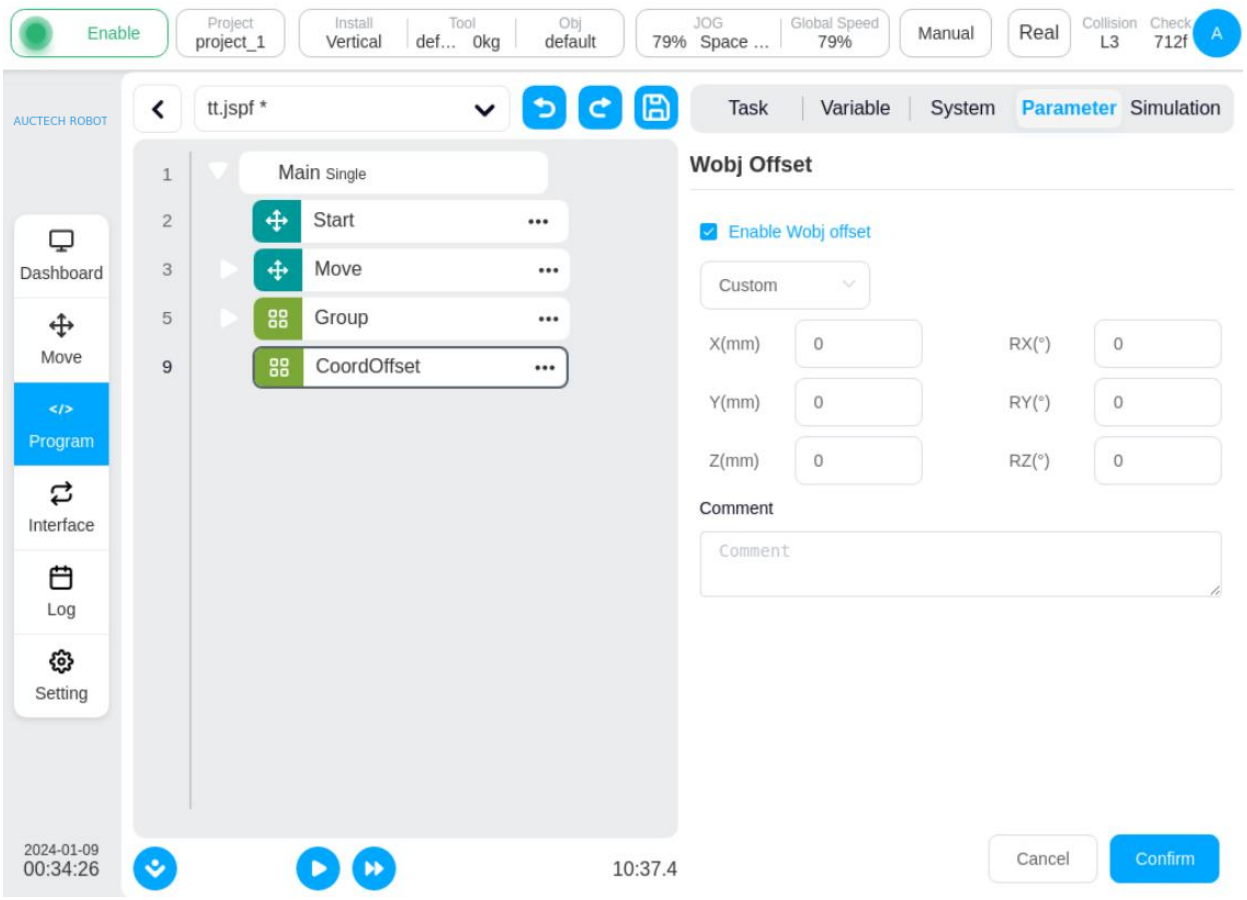
2.5 Instruction blocks -Group



Group function blocks.

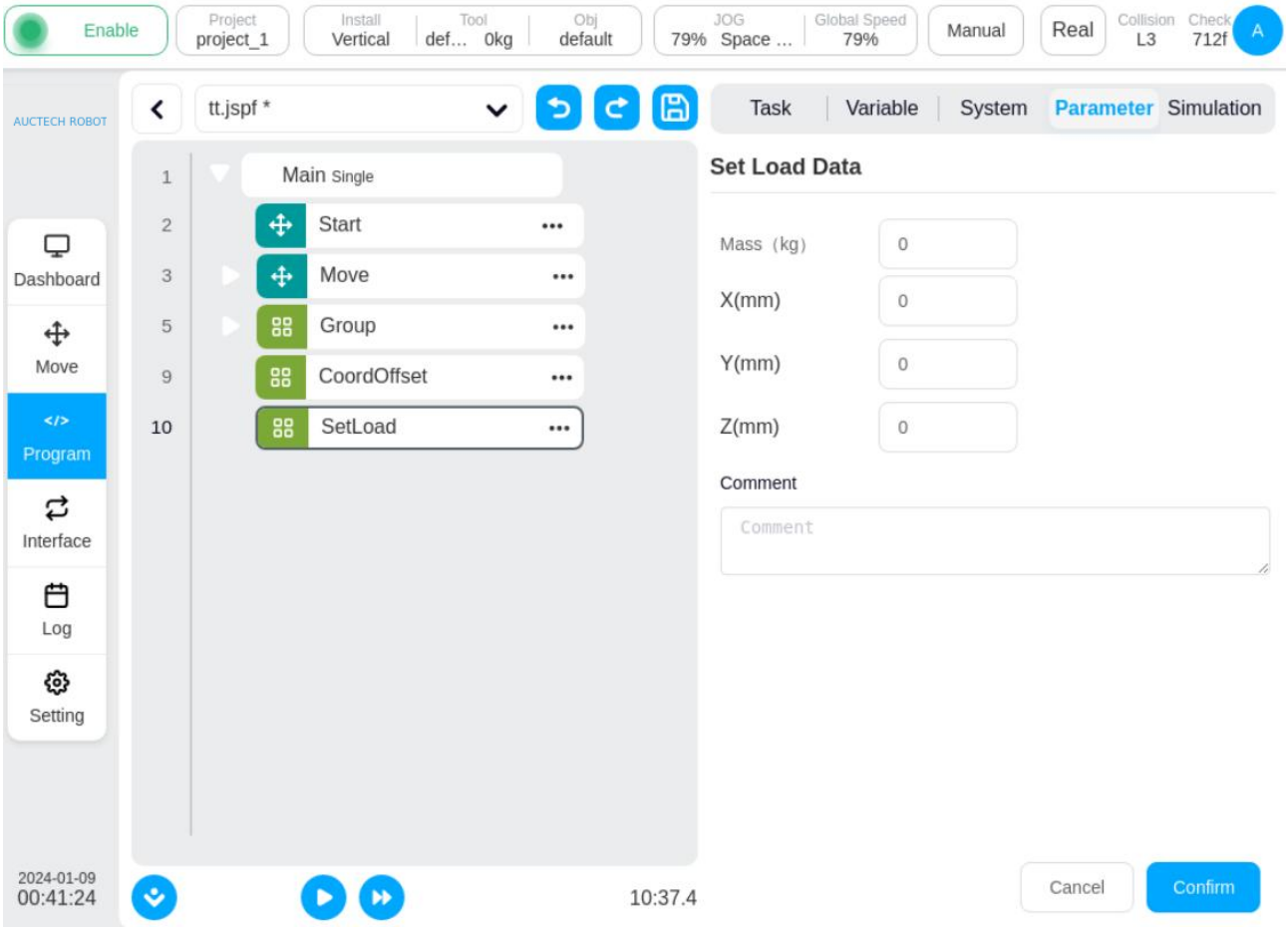
It is used to organize the program, and some function blocks can be placed under a group to facilitate the organization and reading of the program. There is no impact on the execution of the program.

2.5 Instruction blocks -CoordOffset



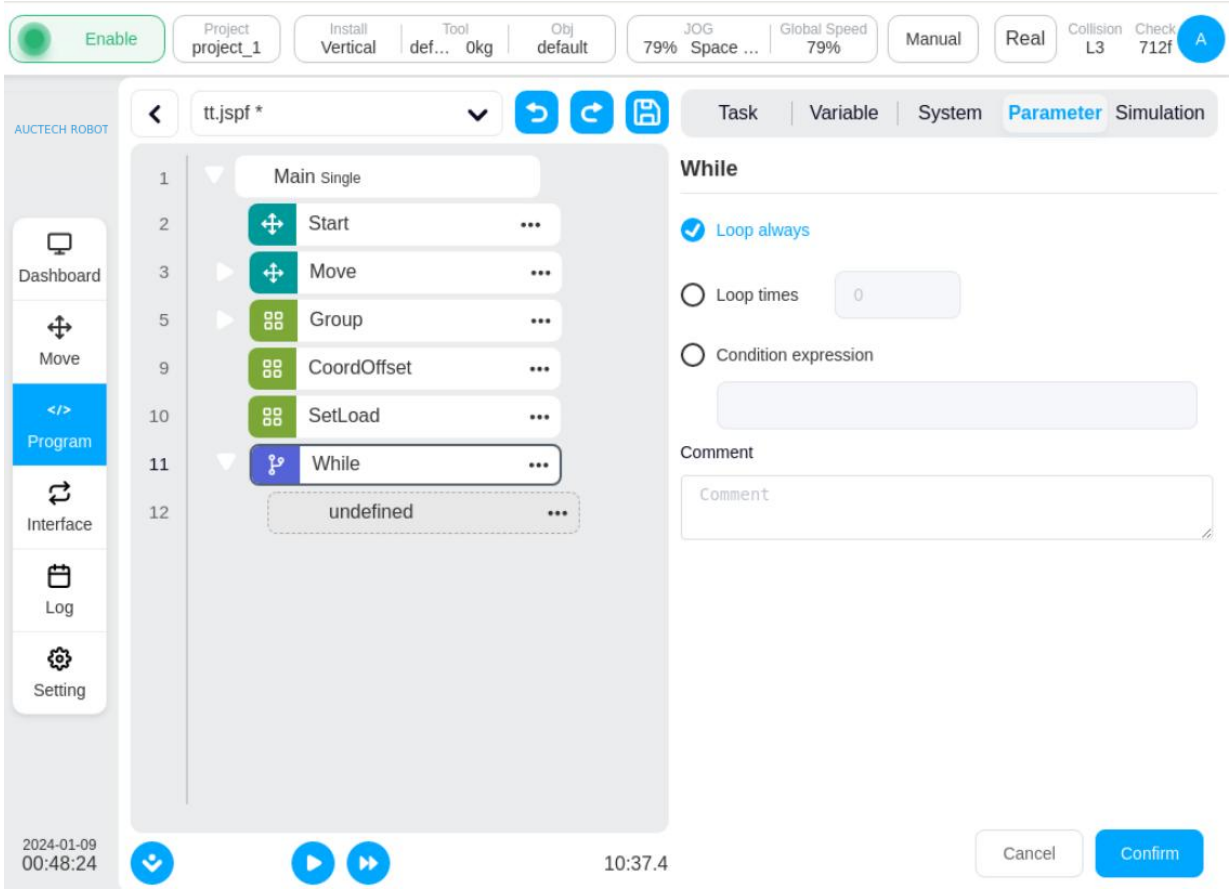
The coordinate system offset function block sets an offset based on the coordinate system of the workpiece. This offset will be added to the reference artifact coordinate system of subsequent Move function blocks. When the program is running, it takes effect, and the program stops and cancels the coordinate system offset.

2.5 Instruction blocks -SetLoad



Set the gripping load function block. The current load of the robot (mass, centroid) can be set during the program run .

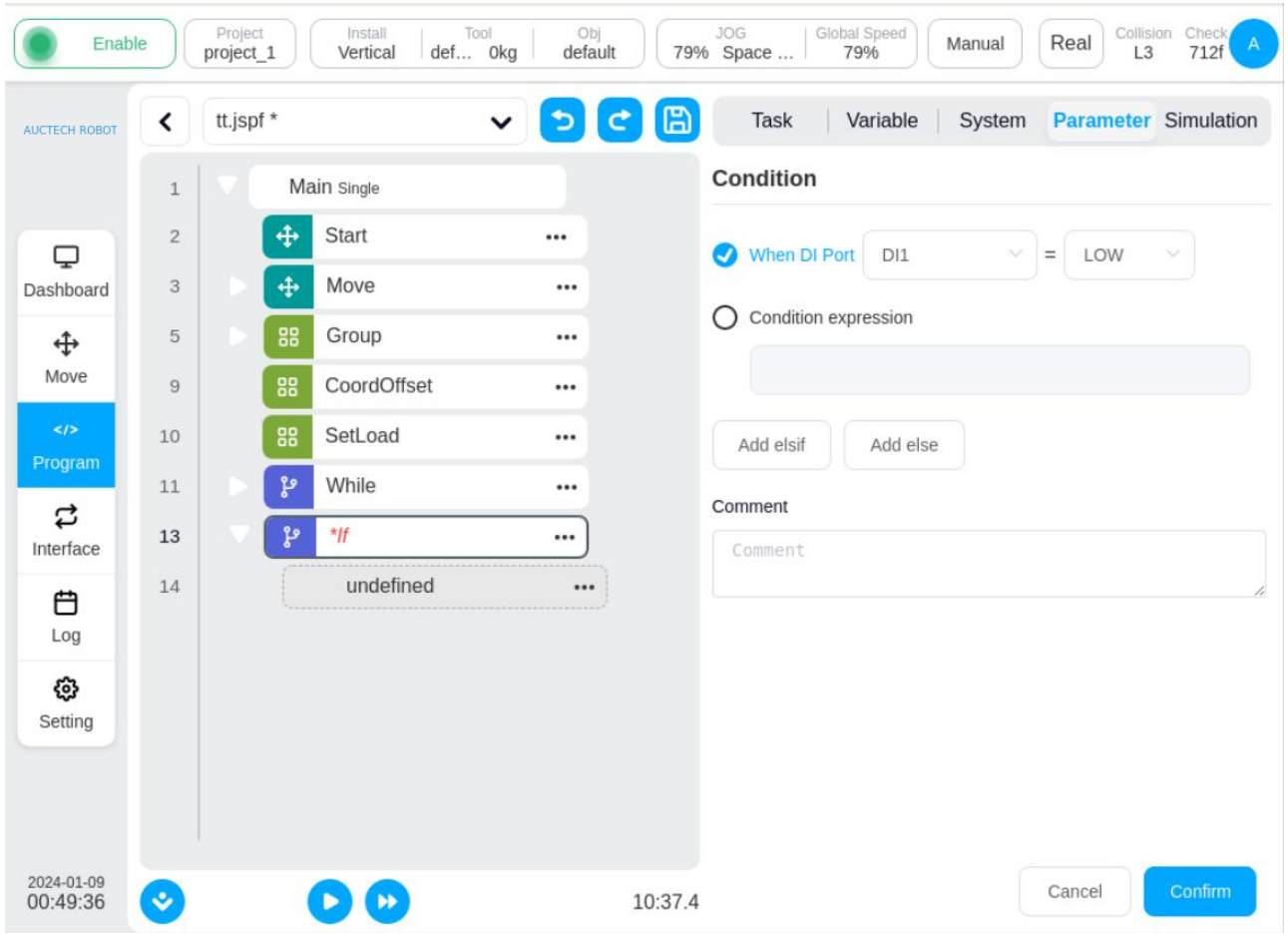
2.6 Instruction blocks -While



Loop function blocks.

The loop executes the function blocks inside it. It can be set to keep looping; Specify the number of cycles; Specify the loop condition, and as long as the loop condition is true, the loop runs.

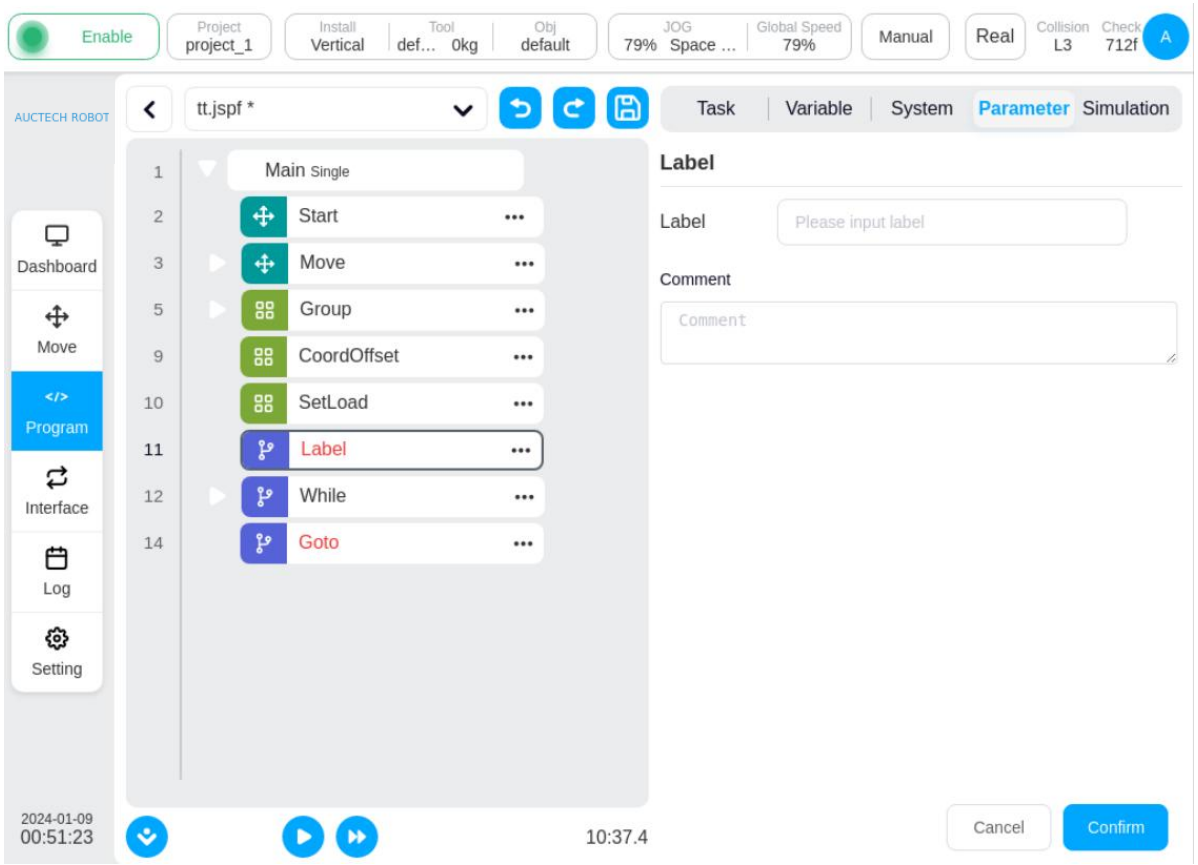
2.6 Instruction blocks -If



Conditional function blocks.

You can set the condition or condition expression of the digital input port, and if the condition is met, the function block in the If will be executed, and the subsequent elsif or else function block can be added.

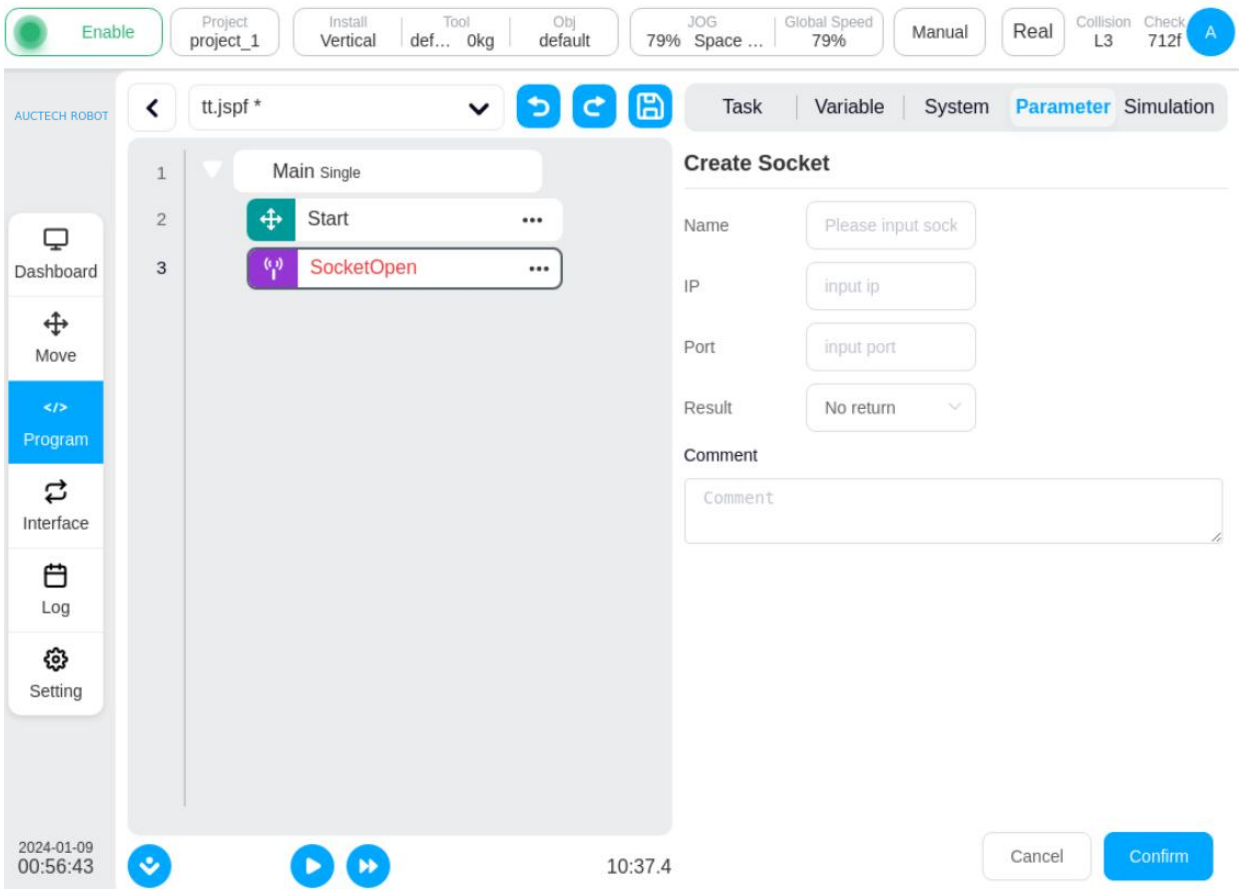
2.6 Instruction blocks -Goto&Label



The Goto function which works in conjunction with Label, allows you to transfer the control point of the program to the Label.

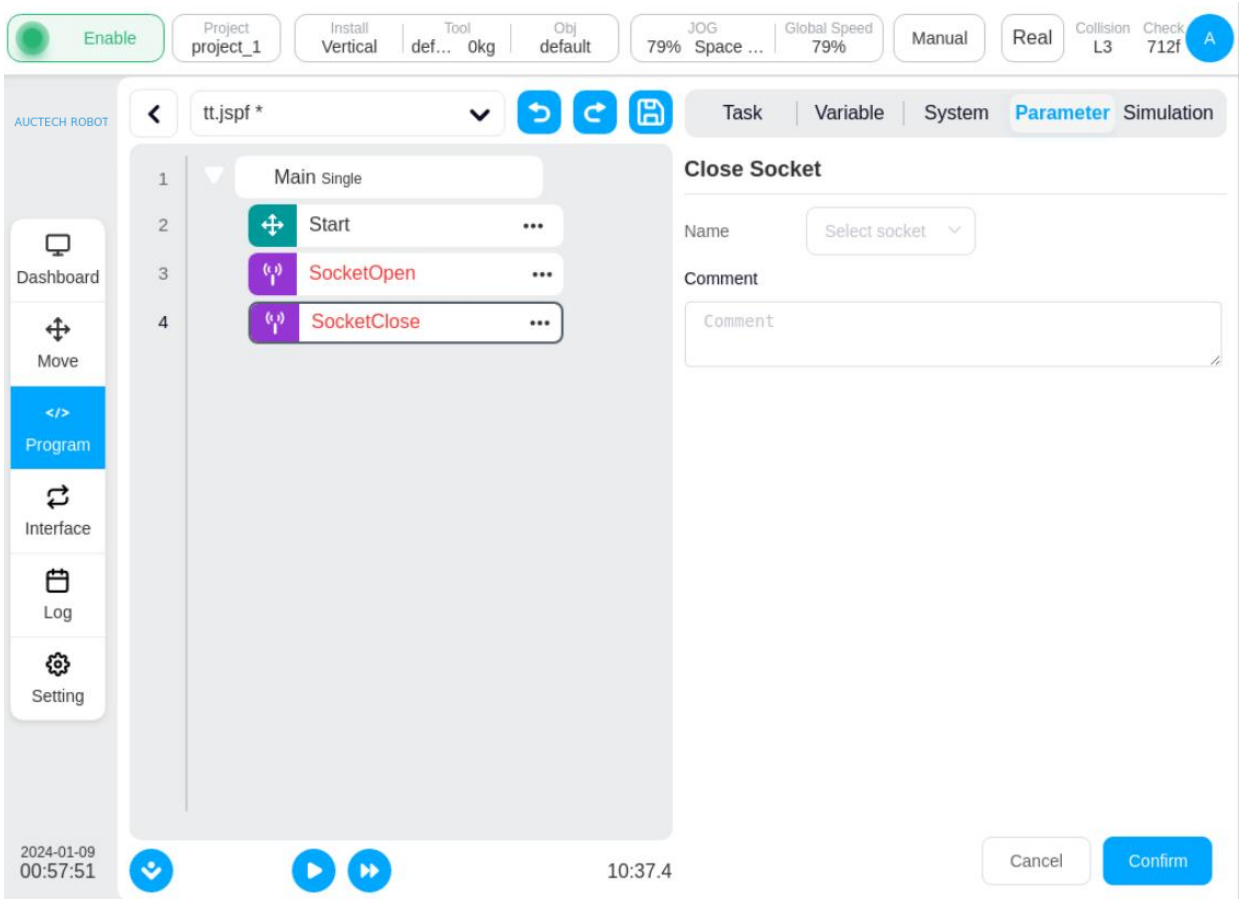
The example program shown in the figure shows that when the program runs to line 14, it will jump to label on line 11 and then run downward.

2.7 Instruction blocks -SocketOpen



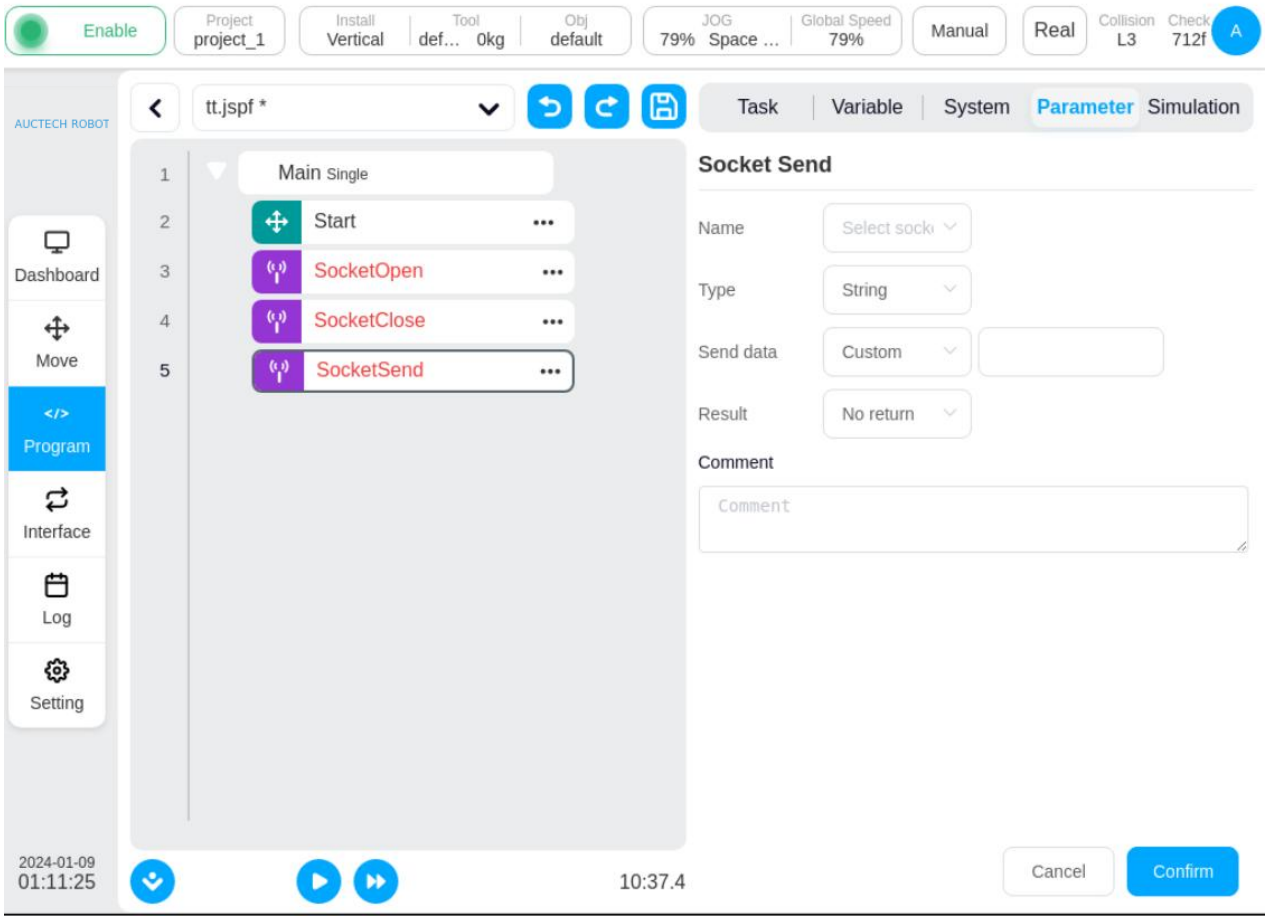
To create a socket, you need to set the name of the connection, configure the IP address and port number of the destination server, and select whether to bind the return value to a variable.

2.7 Instruction blocks -SocketClose



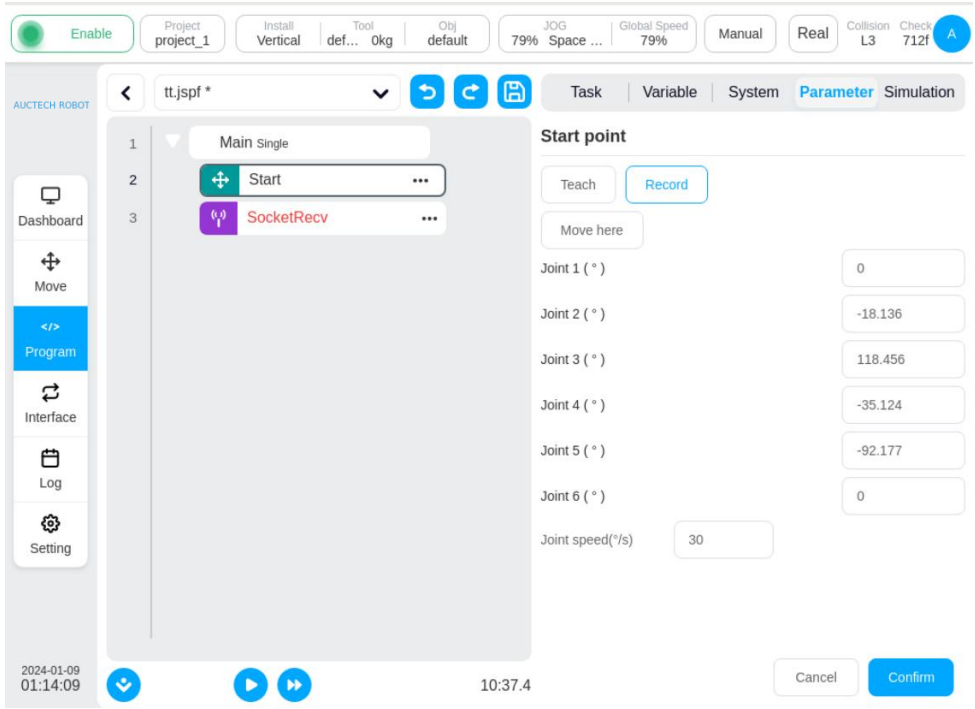
To close a socket connection, you need to select a name for the connection.

2.7 Instruction blocks -SocketSend



socket sends data. To send data to an established socket connection, you can send string or floating-point array as the sending type, you can select a variable or direct input for sending data, and you can bind the return value to a variable to get the sending status

2.7 Instruction blocks -SocketRecv

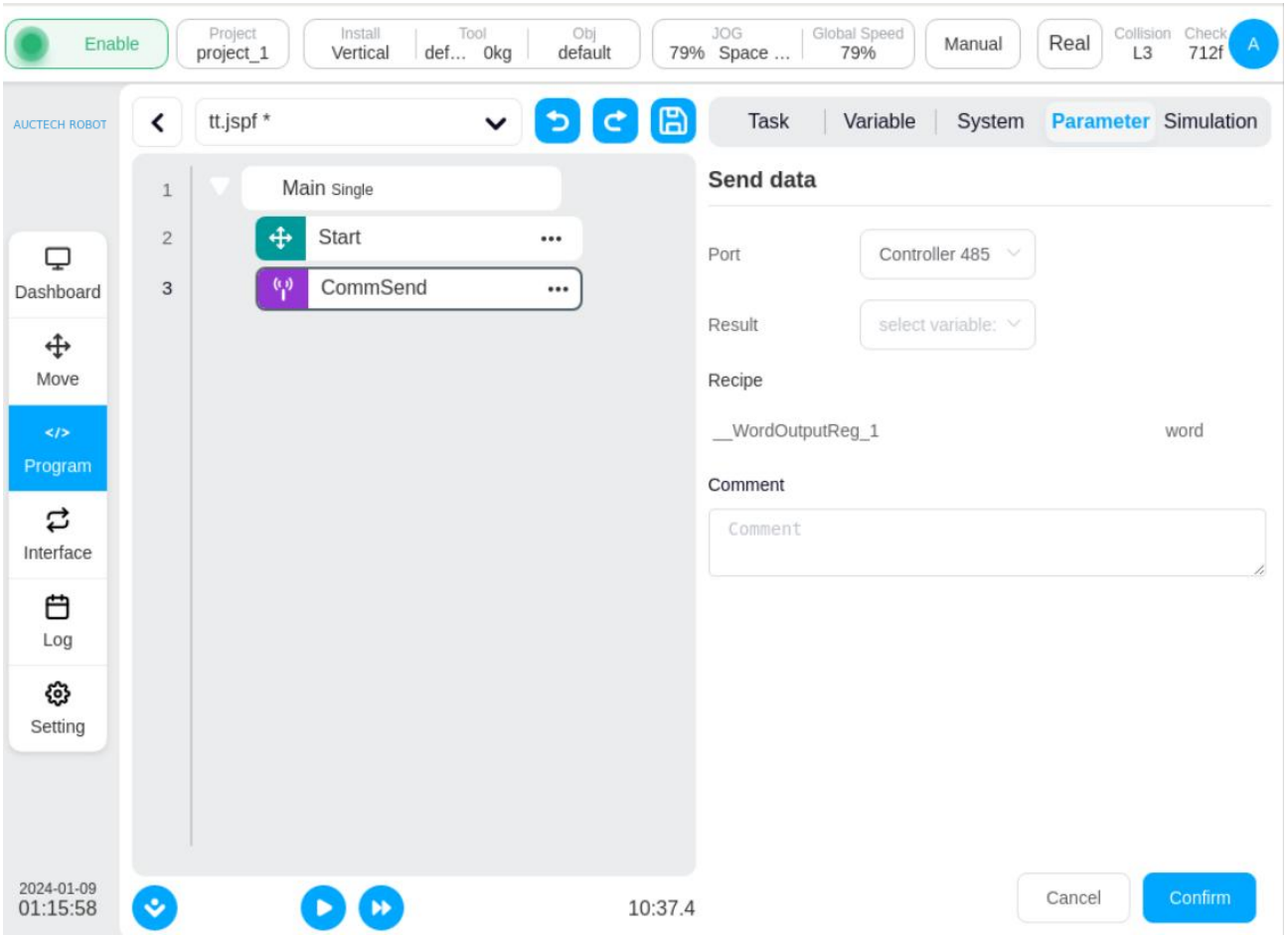


socket receives data

1. Receive data from the established socket connection, the receiving type can be string, string array, floating point array, and the receiving variable is configured for receiving information.
2. When the string type is given, the program will receive the data of the length and process it according to the string, and save it to the variable configured in "Receive Information".
3. When the string array type is used, the program will parse the received string into numeric values, and all the values are in "()", and the values are separated by ",". For example, if a string of strings "(12,1.23)" is received from the socket, the function block converts it to a num_list type with a value of {12,1.23} and saves it to the variable configured for "Receive Information".
4. When the floating-point array type is required, the program will convert the received data into a set of single-precision floating-point numbers and save it to the variable configured in "Receive Information".

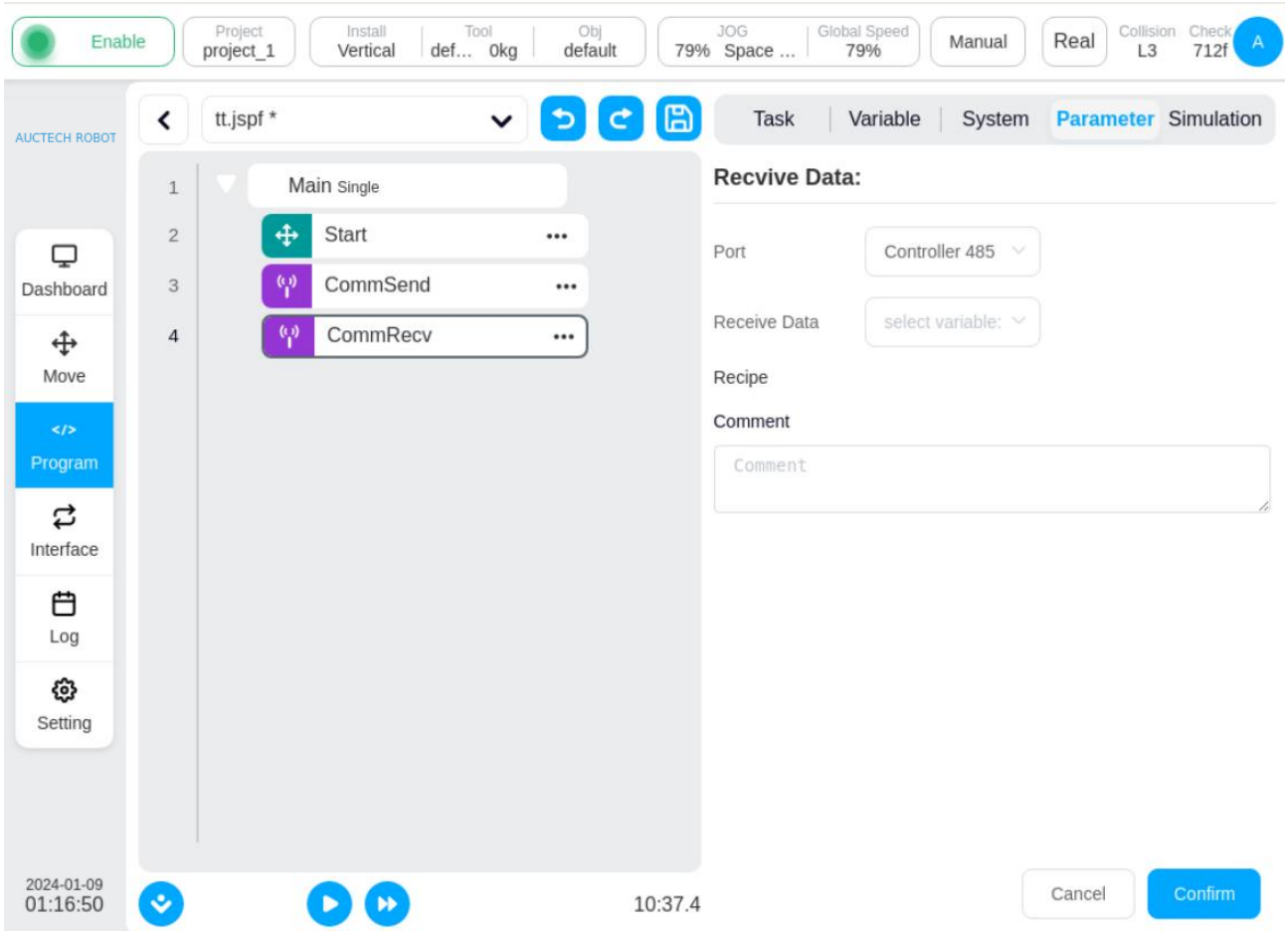
You can configure the receive timeout period, and if the data that meets the rule is not received within the timeout period, the next statement is executed.

2.7 Instruction blocks -CommSend



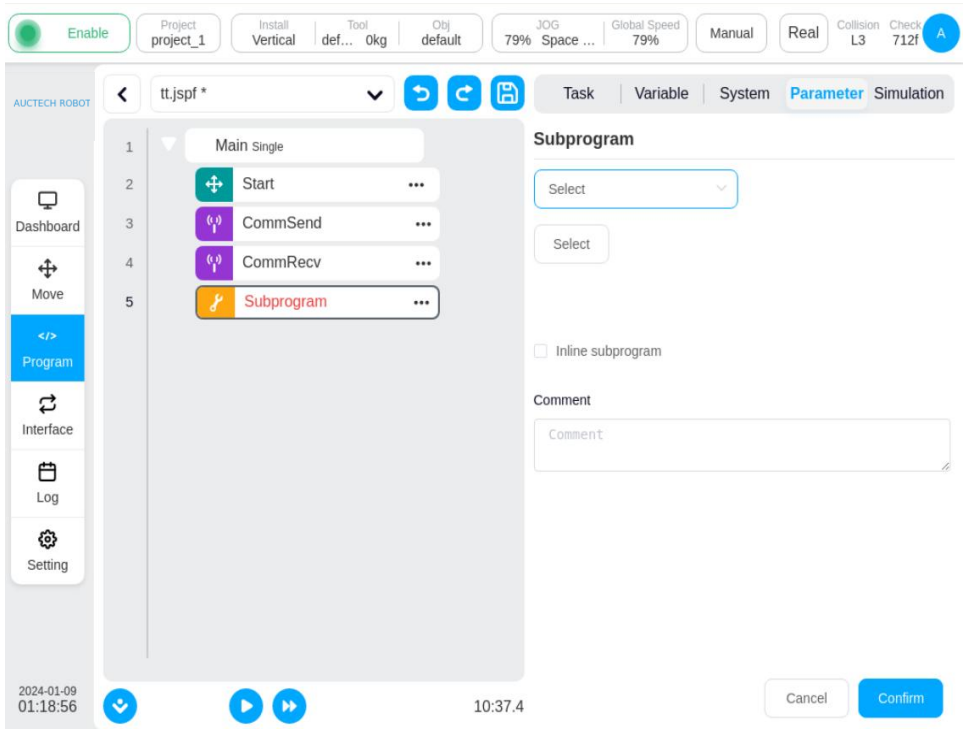
When the robot port 485 or CAN port is set with a recipe, you can use this function block to set the value for the data in the recipe and send it. You can set the status of the return variable to get the recipe data sent.

2.7 Instruction blocks -CommRecv



Recipe data receiving. When the formula is set on port 485 or CAN port of the robot, the function block is used to receive the data, and after processing according to the formula, the obtained num_list data is assigned to the receiving variable.

2.8 Instruction blocks -Subprogram

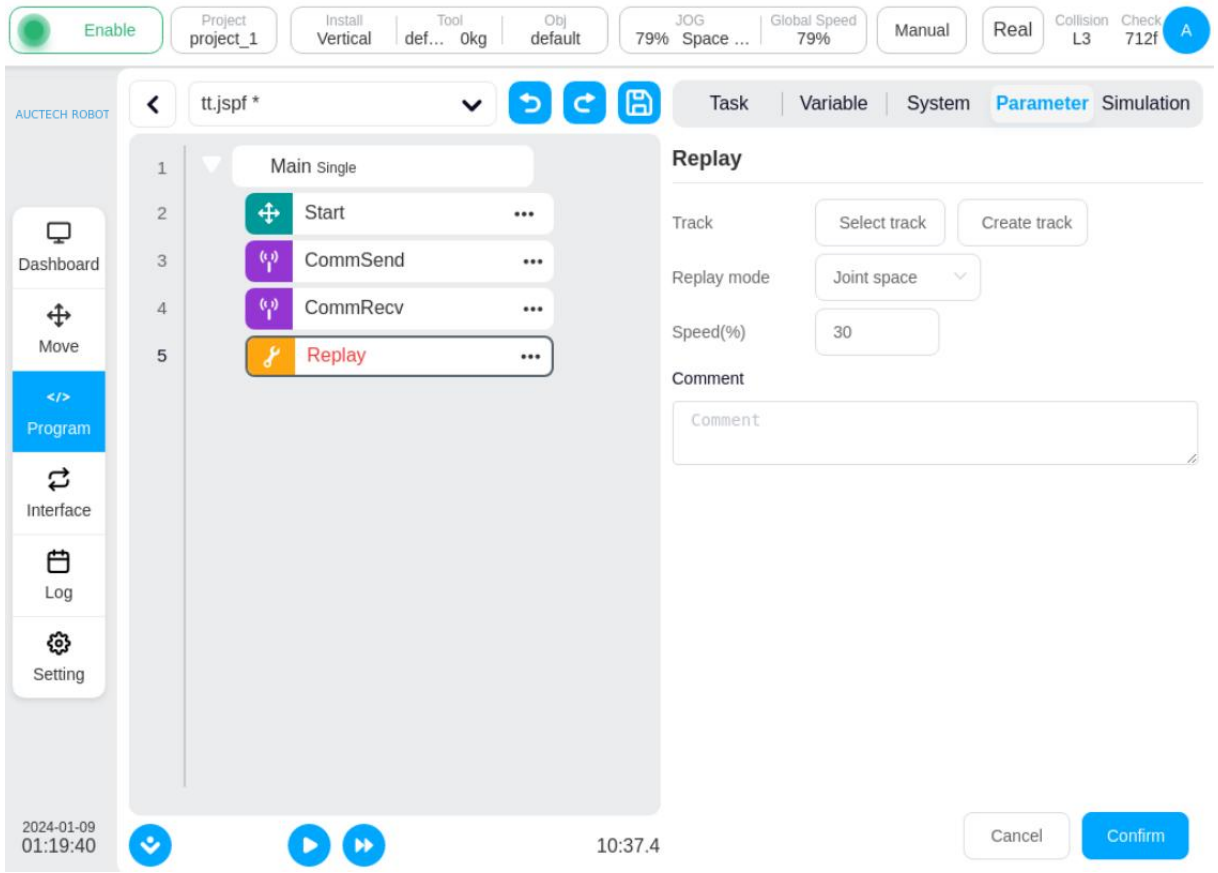


Subroutine.Other programs can be embedded into the current program. There are two ways:

Select the embedded subroutine directly from the program list, if you do not check the “inline subprogram" option, the subroutine will be loaded from the file every time the program is running, that is, the change of the subroutine file will affect the main program. If checked, the subroutine is copied directly to the main program, and the changes to the subprogram file have no effect on it.

If it is set to a string variable, the program will dynamically load the corresponding subroutine according to the variable value as the subprogram name when it is running.

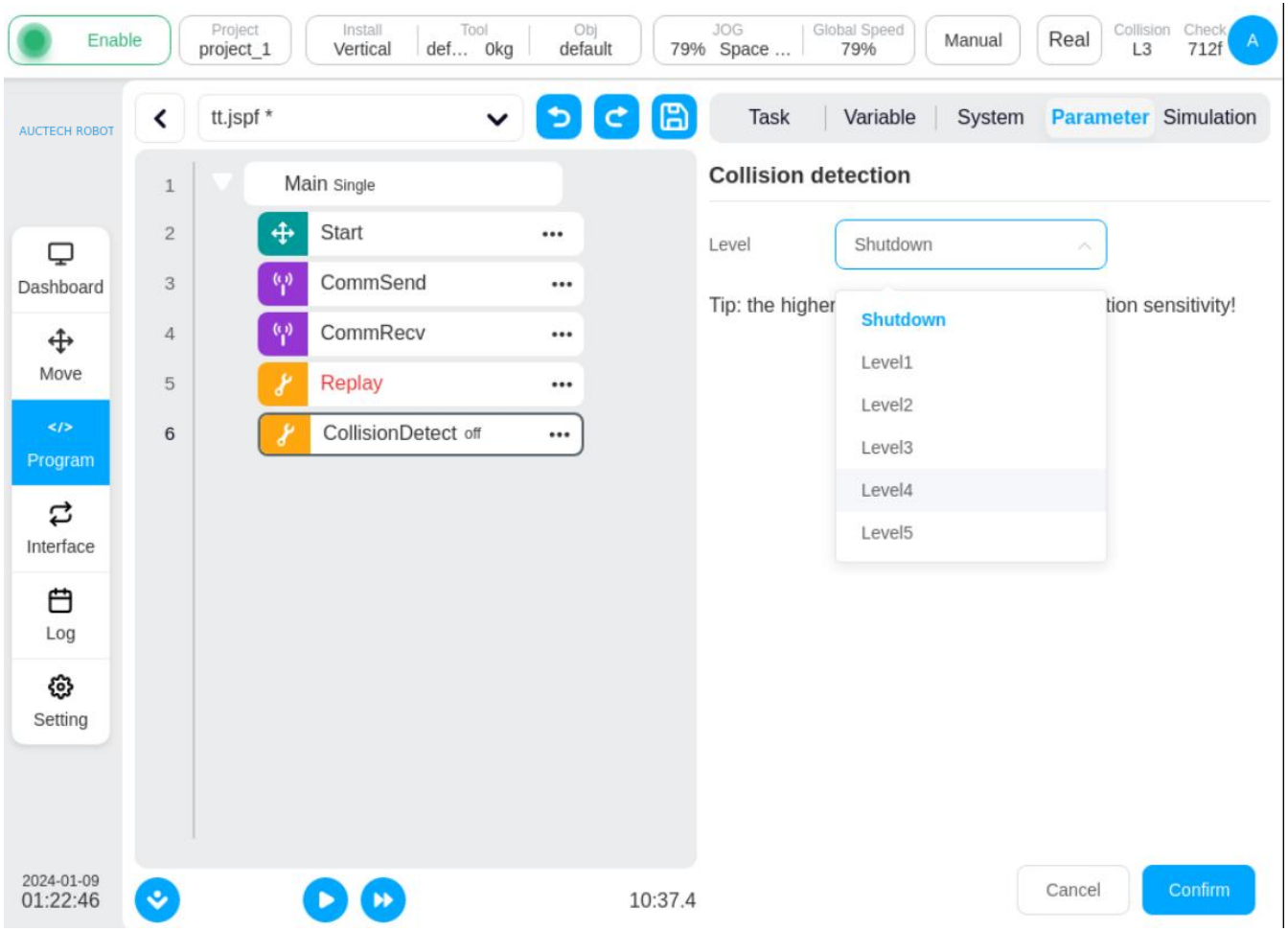
2.8 Instruction blocks -Replay



Trajectory recurrence. You can create a trajectory of the robot's motion, and the robot will move according to the trajectory when the program executes to this function block. You can select an existing track file or create a new track.

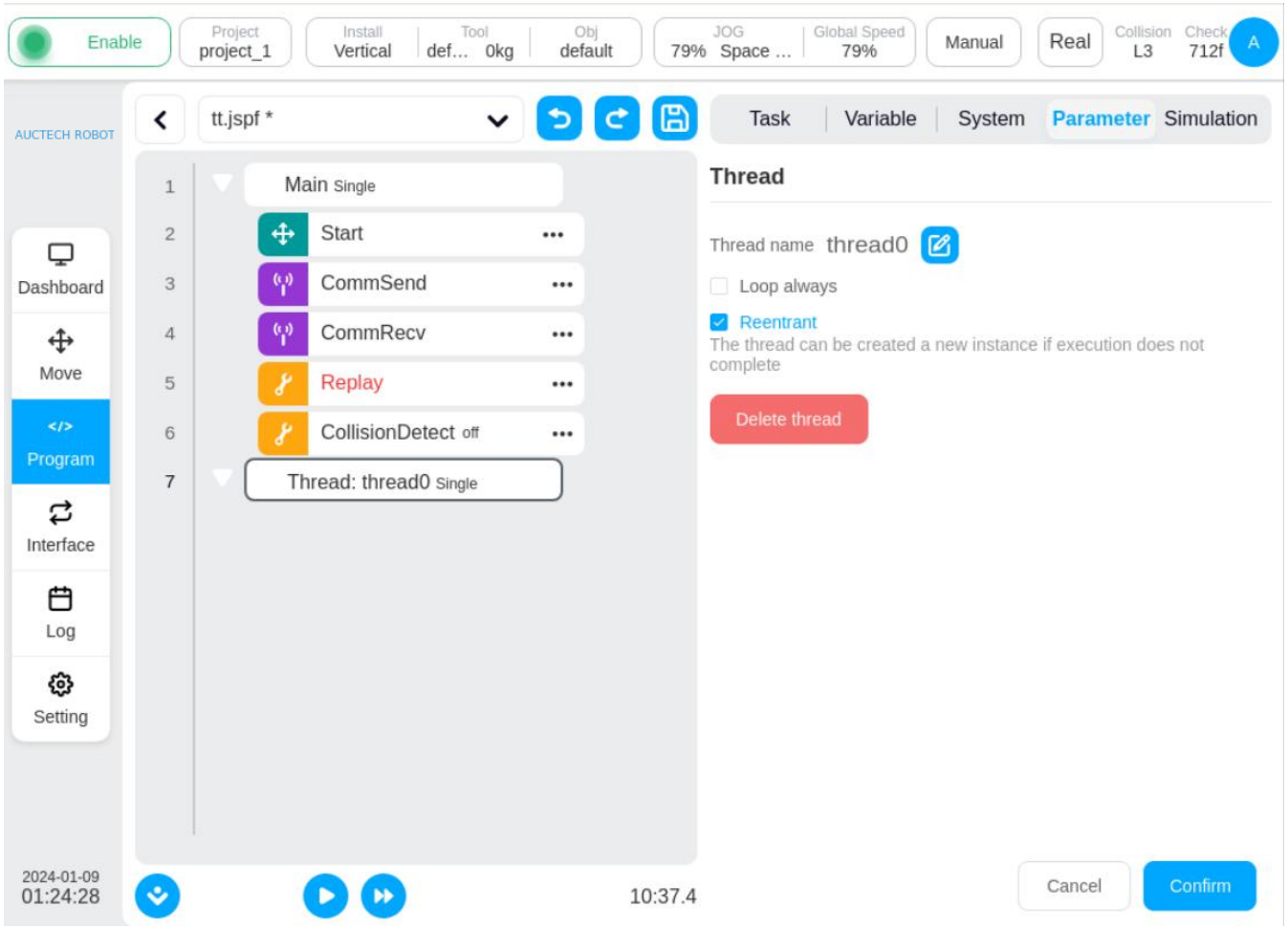
How to create a new track, click "New Track", the "Track Recording" box will be displayed after entering the track name, click "Start Recording" to start recording the trajectory data, at this time, the page will display a translucent floating box indicating that the trajectory is being recorded, the user can use traction or jog to move the robot, click "Stop Recording" or the stop button on the floating frame, then complete the creation of the trajectory file.

2.8 Instruction blocks -CollisionDetect



Collision detection level. This script can be used to set the sensitivity of collision detection during the program run.

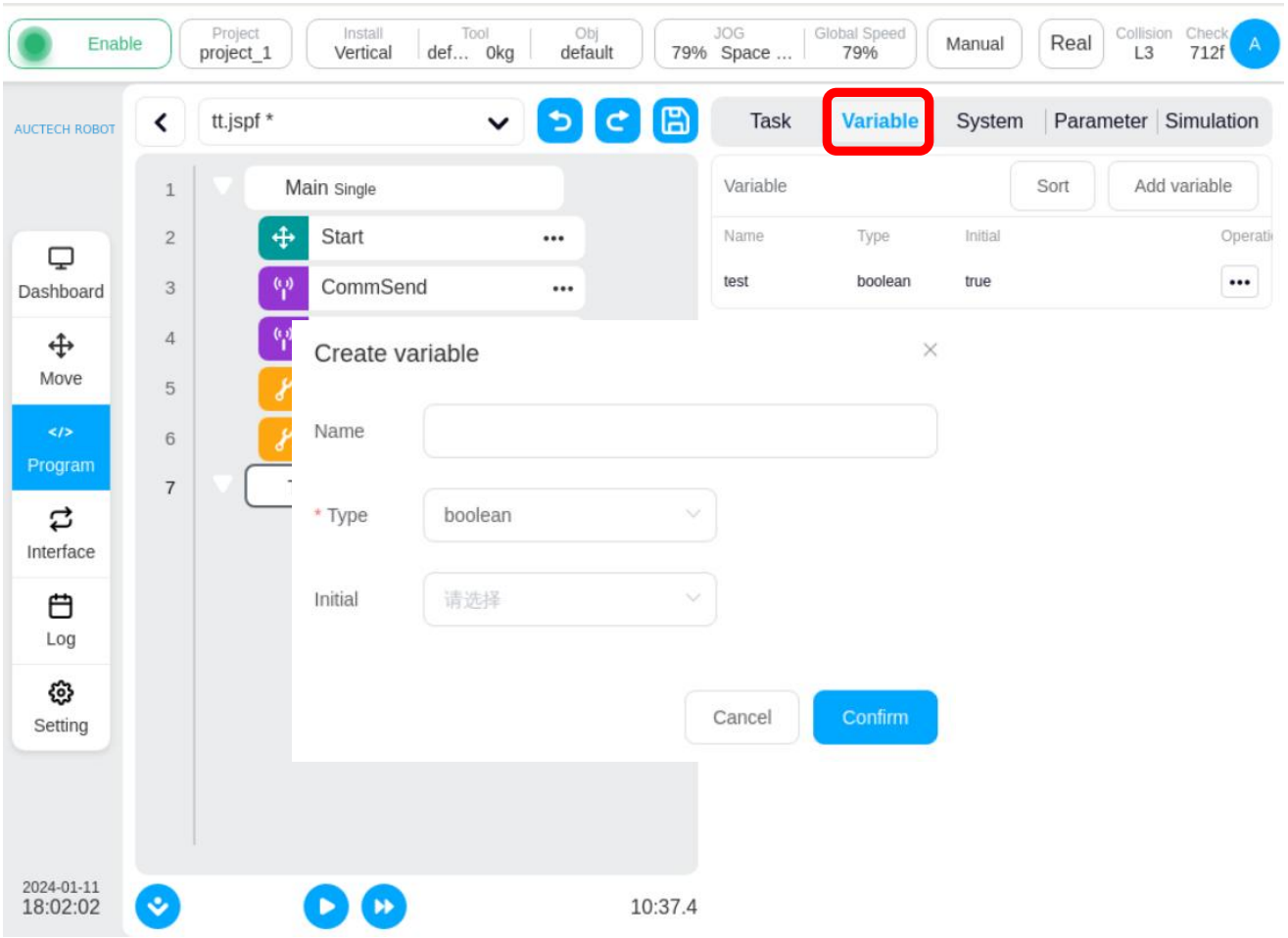
2.8 Instruction blocks -Thread



Threads, executed in parallel with the main program of the robot, interact with the main program through variables. In the parameter configuration area, you can set whether the thread keeps looping. A maximum of 10 threads are allowed to be created in a program.

Note: Robot motion instructions are not allowed in the thread

2.9 Variable



Program variables are of the following types:

boolean: boolean num_list: array type

number: Number type string: String type

pose: The type of data that represents the Cartesian position of the bot

joint: The type of data that represents the position of the robot's joints

pos_ list: Cartesian position array Joint_list: joint position array

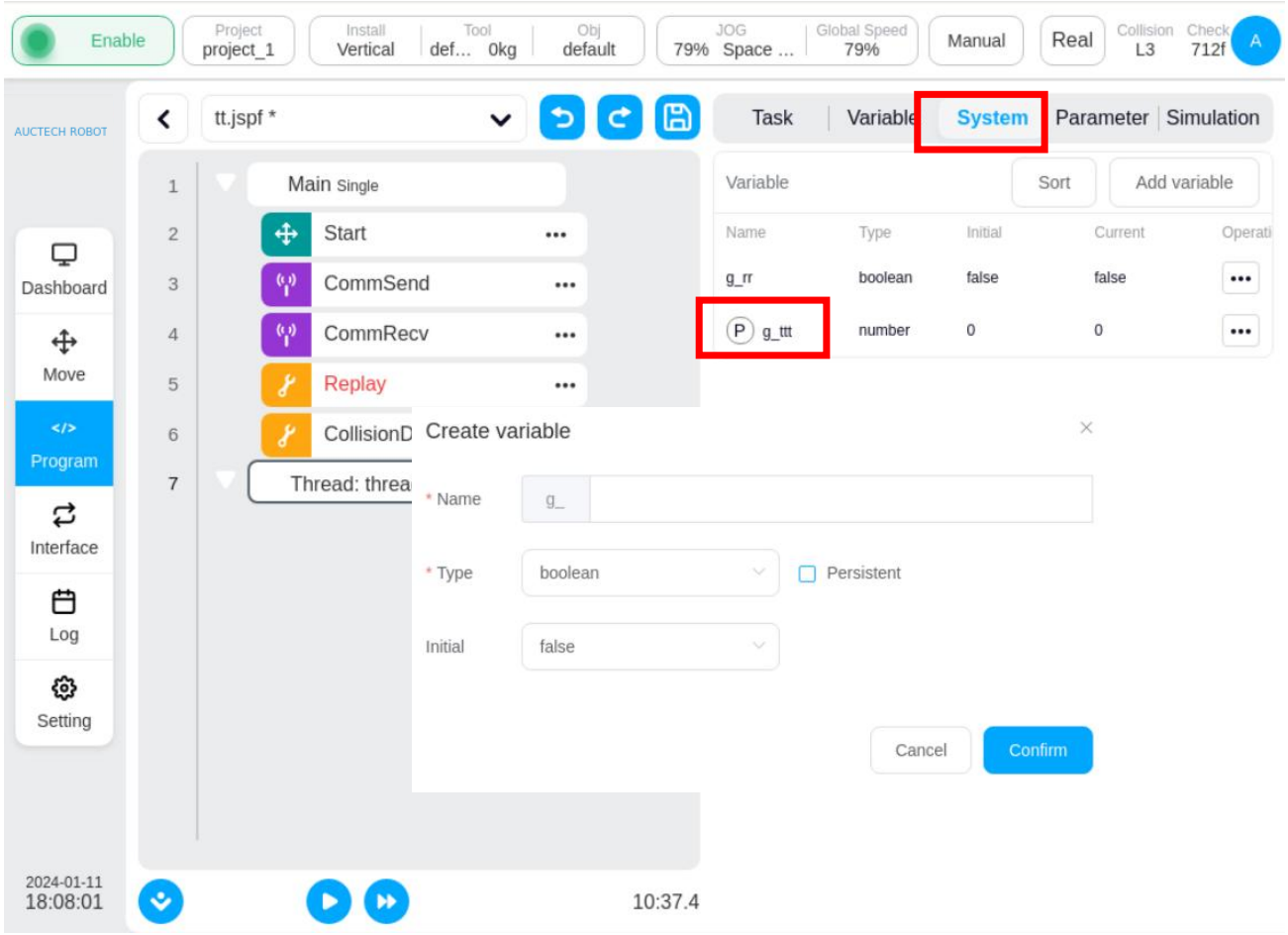
pose_speed: Cartesian position velocity

joint_speed: Joint velocity

pose_acc: Cartesian position acceleration

joint_acc: Joint acceleration

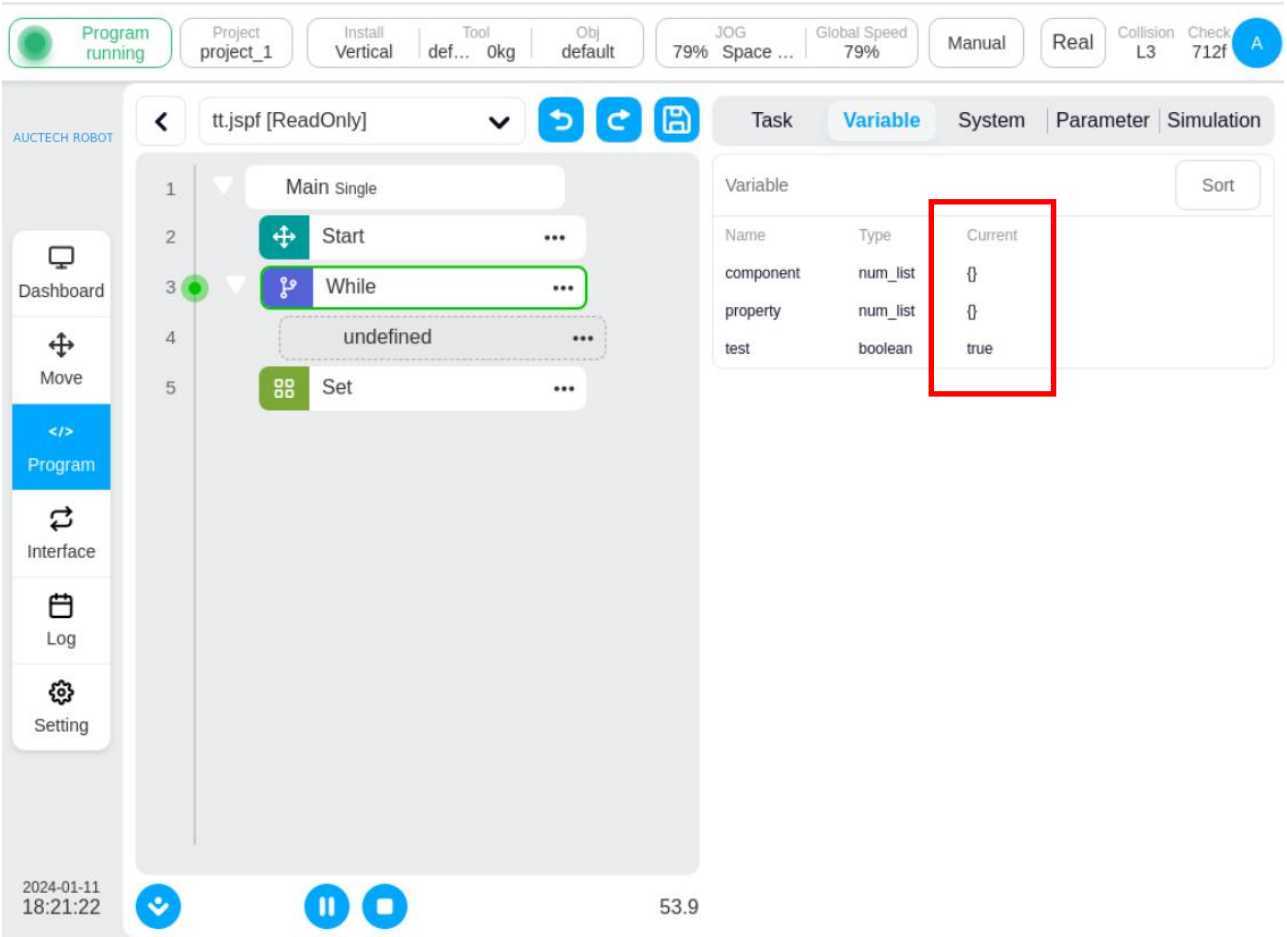
2.9 System variable



The word P is displayed in front of the name of the persistent variable.

The variables added in the settings are system variables, which can be used by each program in the current project.

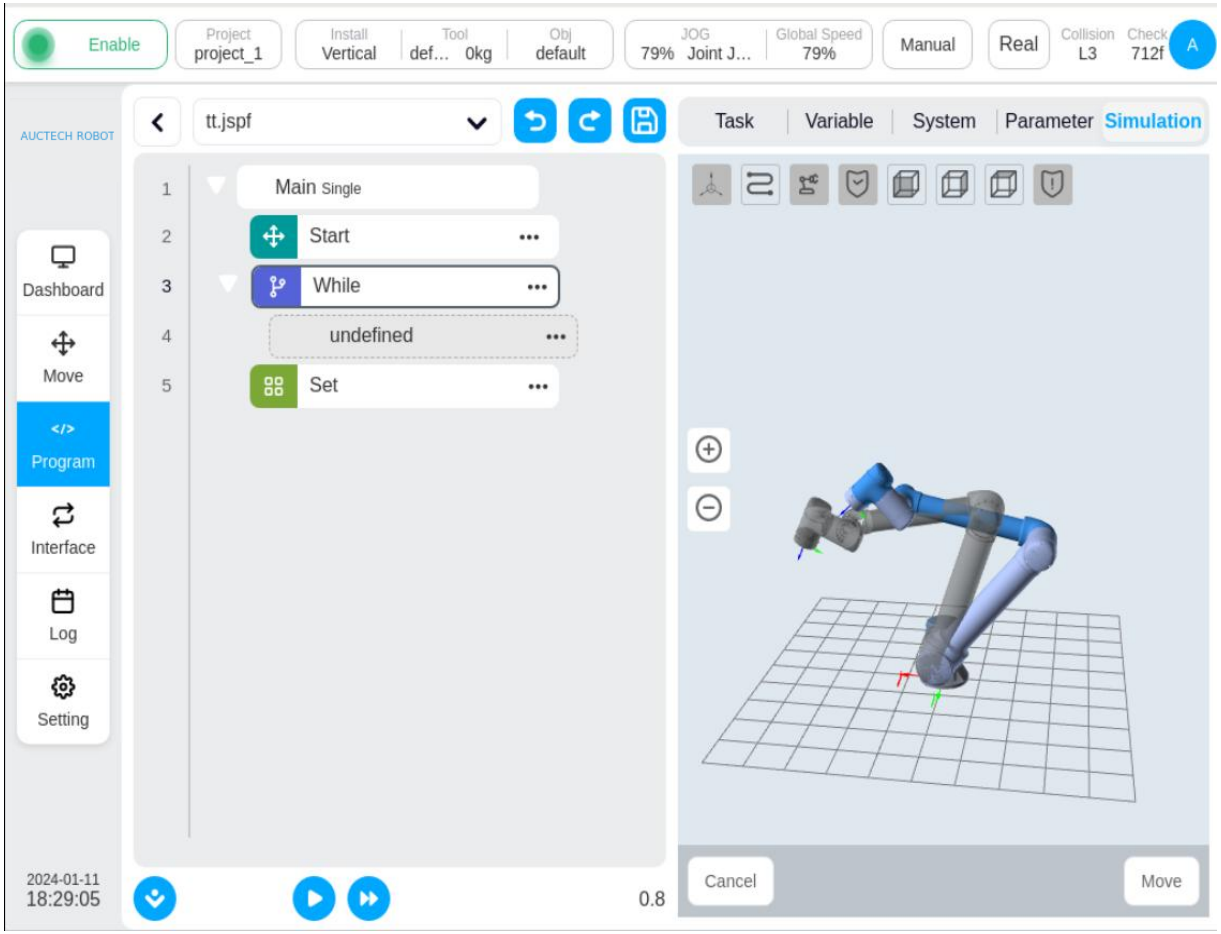
2.10 Variable monitoring



While the program is running, this page can display the current values of program variables and system variables.

Note: Variables of type JOINT and POSE are both displayed as num_list during the monitoring period, and the data unit is displayed in m/rad. If the data is too long, you can use the "log" function block to print the variable to the log for viewing.

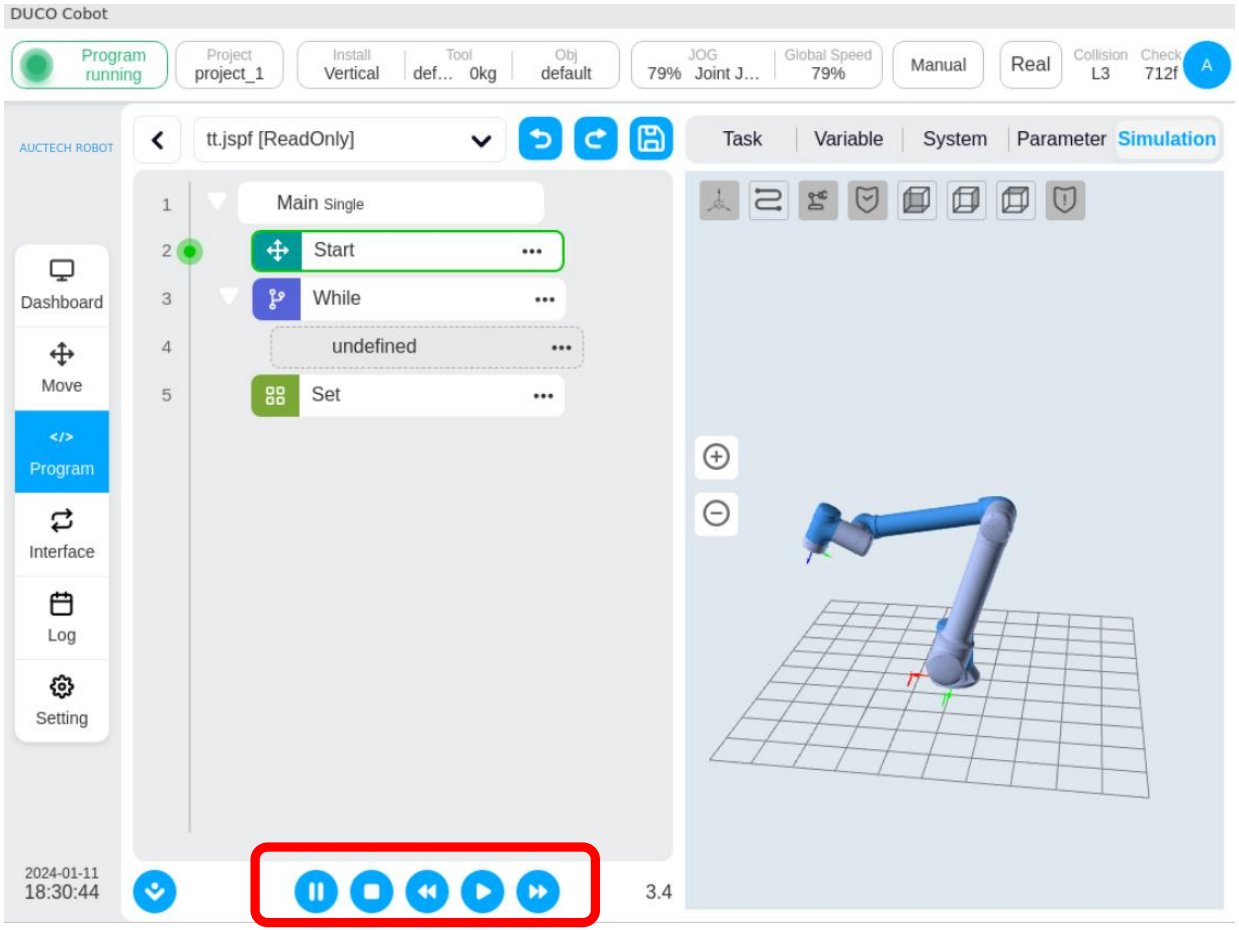
2.11 Run



Click on the "Run" tab to enter the running page, which controls the start and stop of the program and displays the 3D model of the robot. Click the button to run the program, and click the button to run the program in a step-by-step manner. You can pause or stop the program at any time while it is running.

Note: If the actual position of the robotic arm is inconsistent with the Start point, you need to press and hold "Press and Hold" to manually move to the Start point.

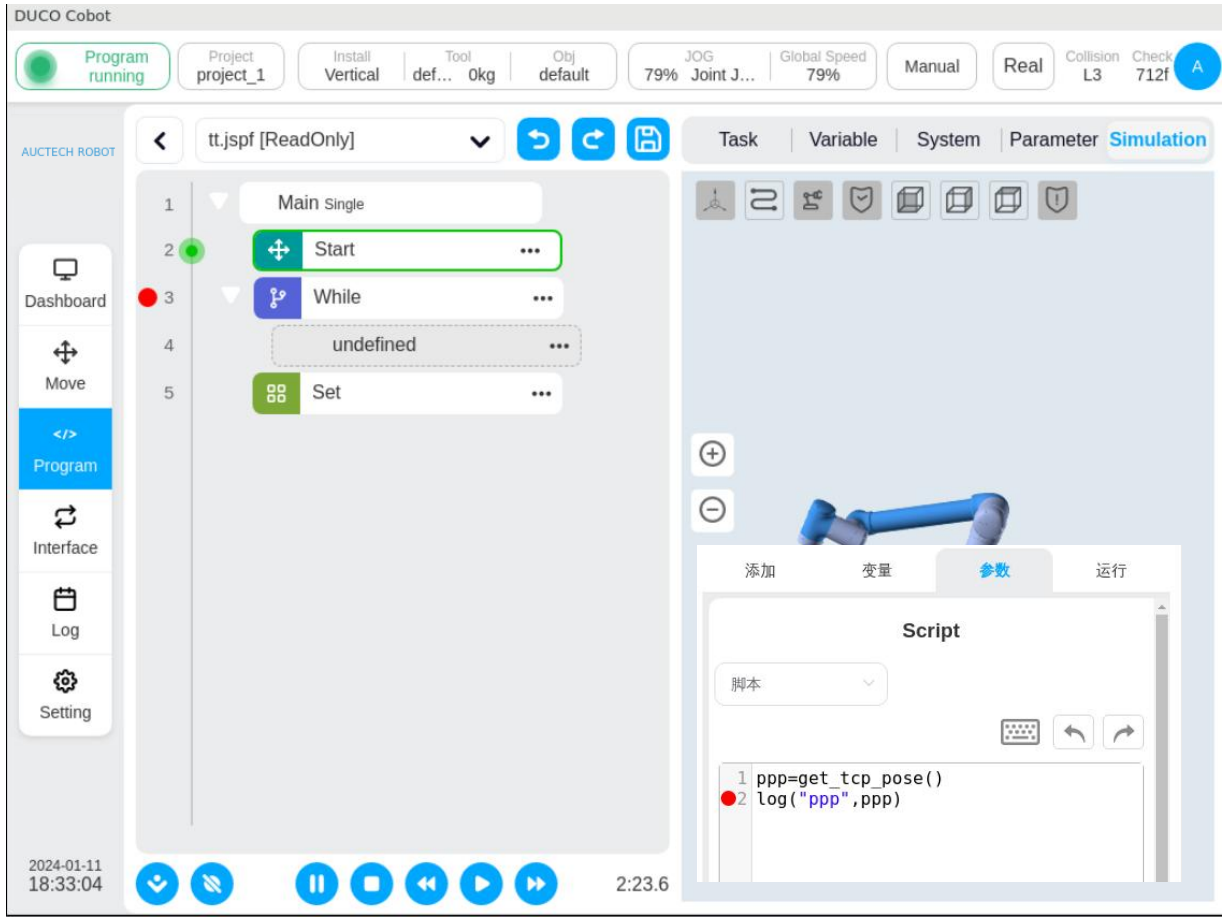
2.12 Run - Single-step



Go back to the previous step

Note: For sports function blocks only

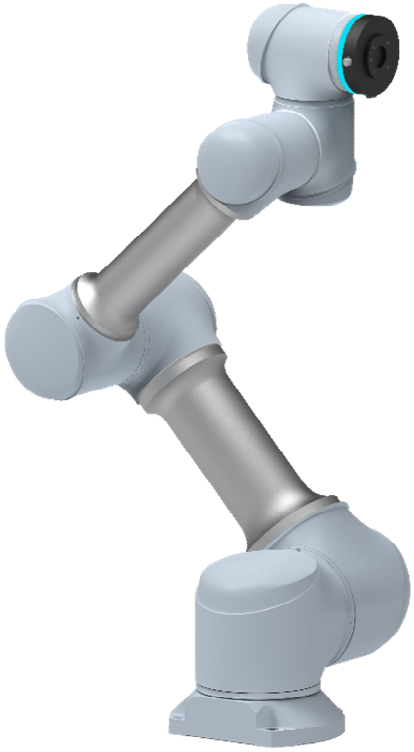
2.13 Breakpoint debugging



When the program runs to a breakpoint, you can choose to continue running, or run forward and backward step by step.

Note:

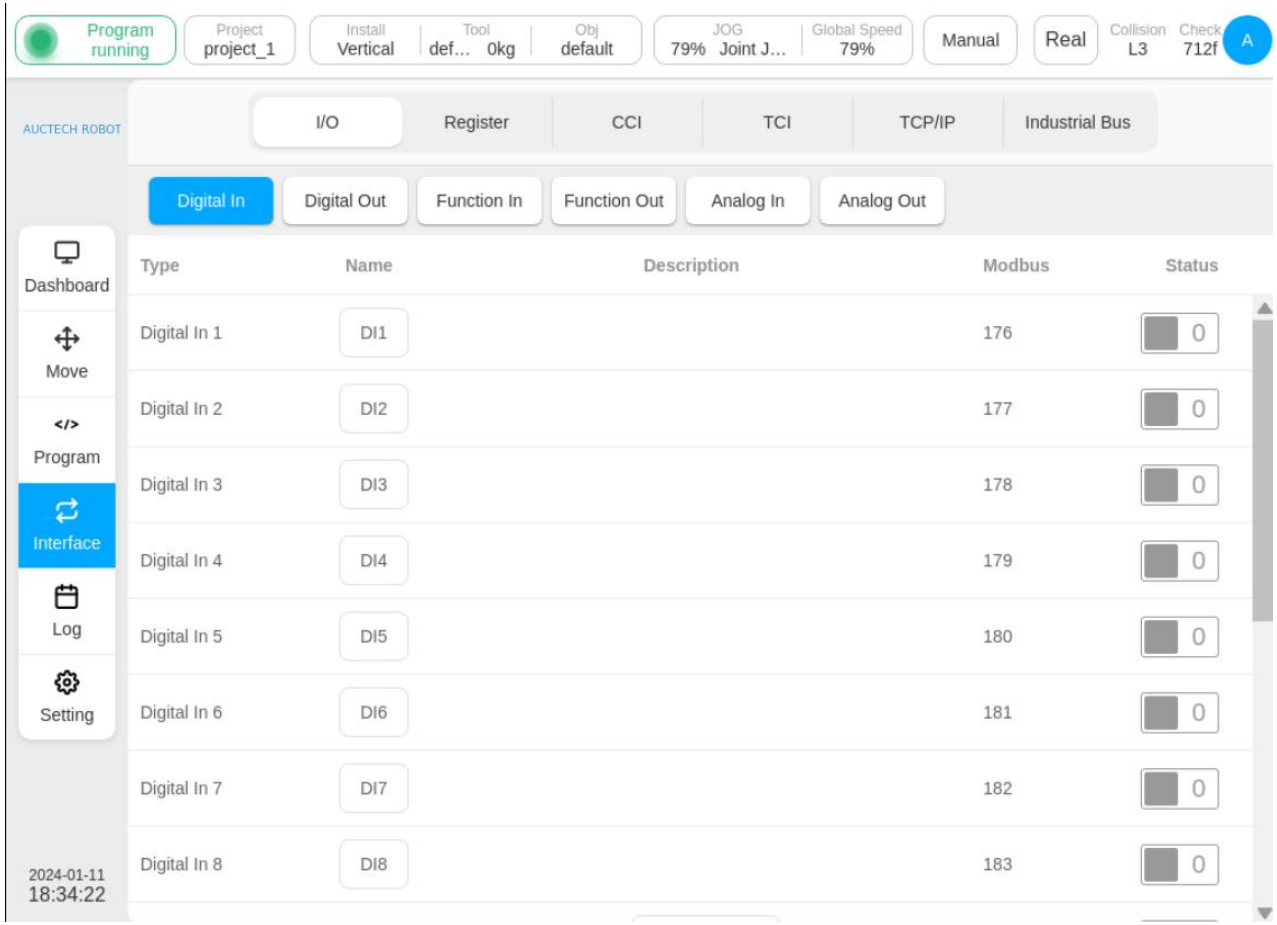
1. "Cancel Breakpoint" is all canceled!!
2. Breakpoints can also be added to the "Script" script function block.
3. When calling a subroutine, adding a breakpoint to the subroutine is invalid.



Lecture 3

Interface

3.1 Digital_I/O

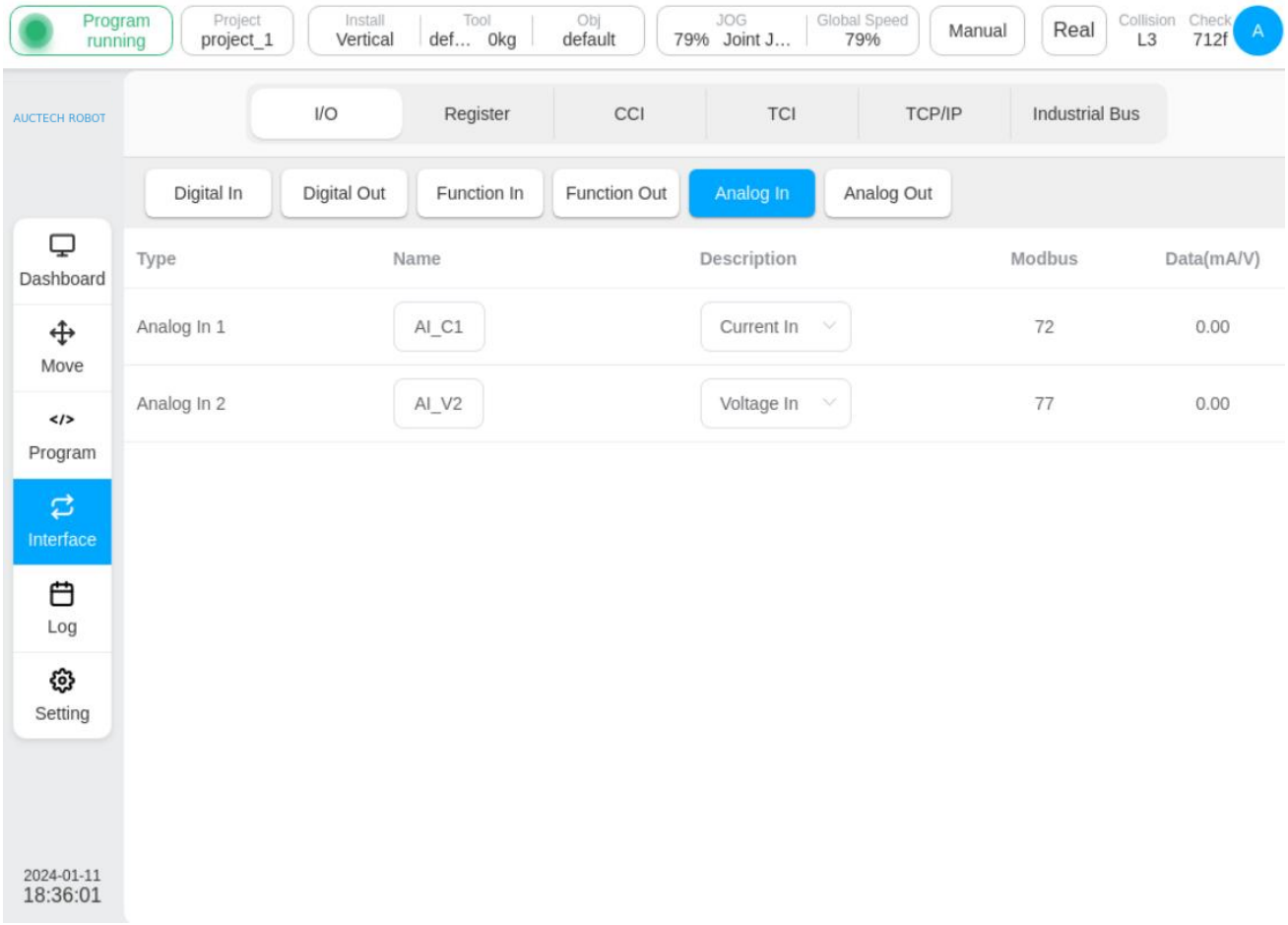


There are a total of 16 sets of input and output IOs, of which 8 groups are configurable IOs, which can be configured as functional IOs. Functional IO currently supports the following functions:

Inputs: Run Program, Pause Program, Stop Program and Traction Teaching, Collision Detection Reset, Robot Power-On, Robot Enable, Robot Disable, Robot Power-off.

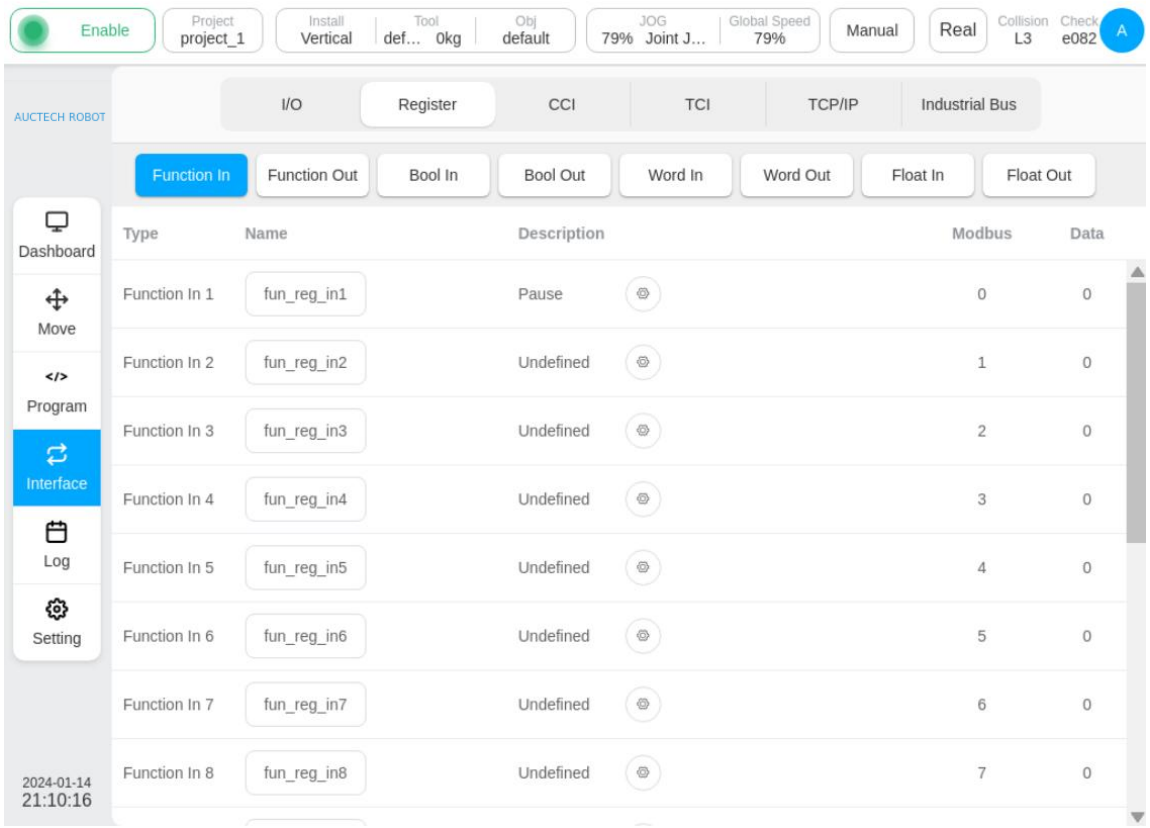
Outputs: Idle Status, Program Running Status, Paused Status, Program End, Profinet Disconnect, Home Position, Collision Trigger, Auto Mode, Robot Power-Off Status, Robot Disable Enabled State

3.1 Analog in/out



2 configurable analog inputs and outputs

3.2 Register-Function in

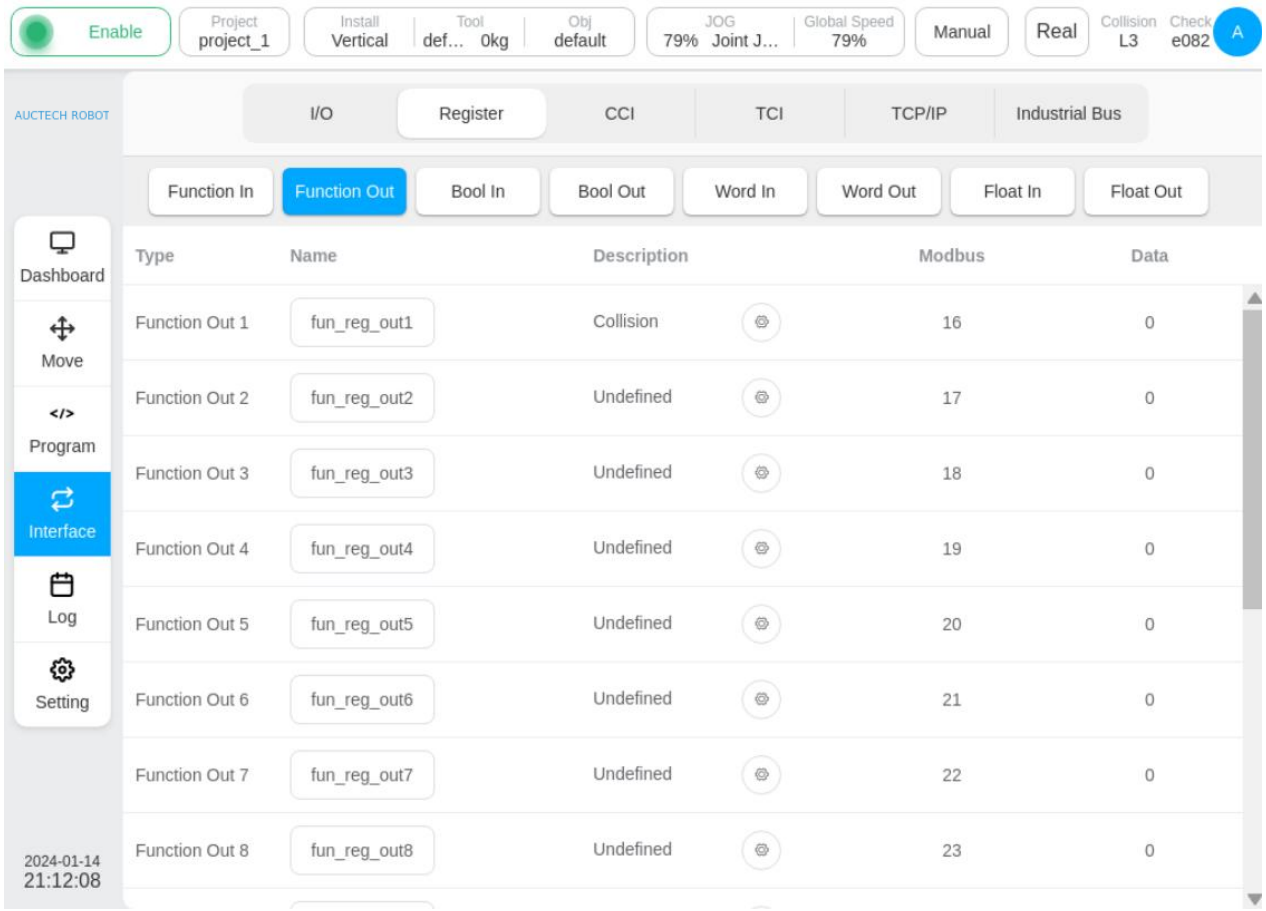


The 16-channel function input of the register currently supports the following functions: running program, pausing program, stopping program and traction teaching, collision detection reset, robot power-on and power-off, robot up and down enablement, system shutdown, and recording point.

Notes:

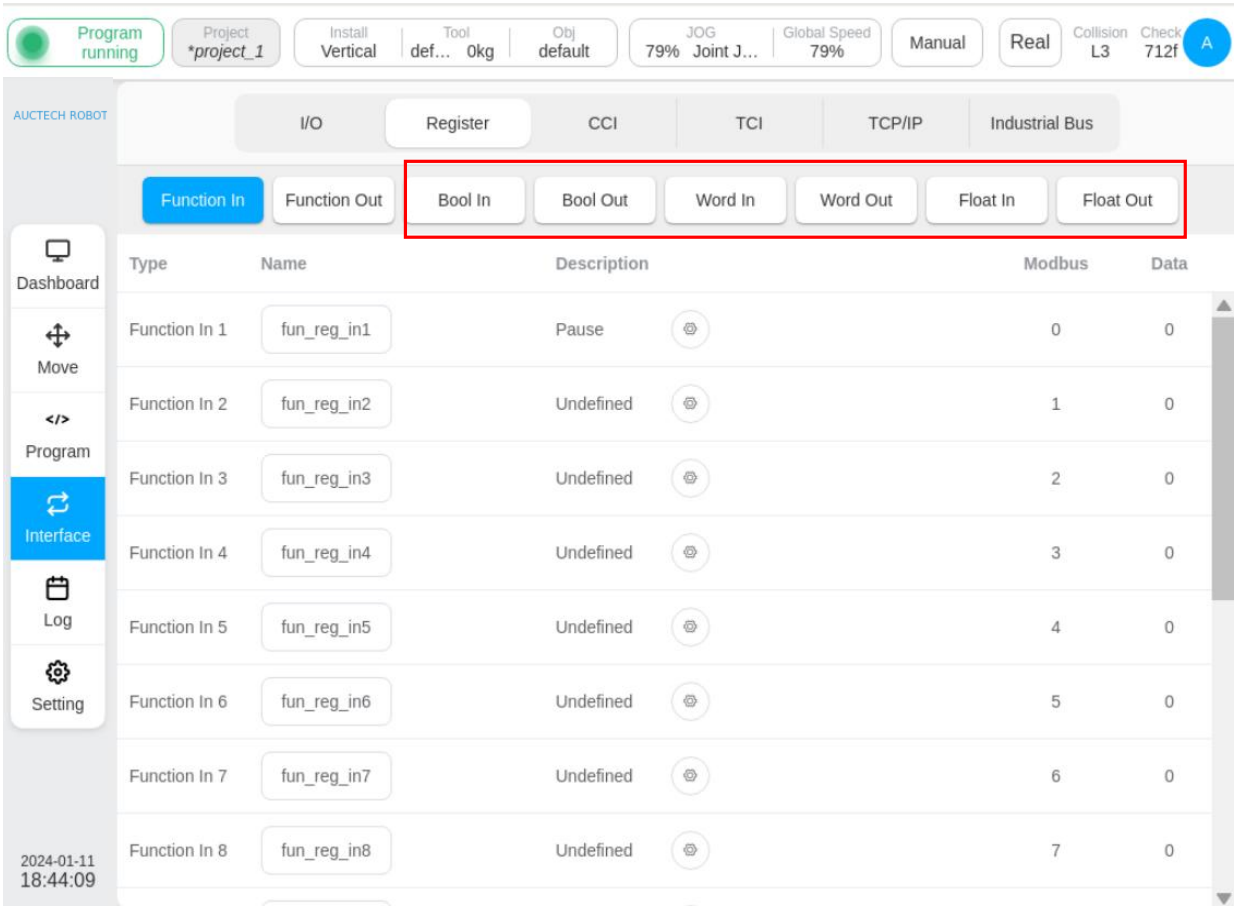
1. Change the register name to conform to the naming rule "2-24 digits starting with a letter. Letter. Underline combination"
2. When defined as "running program", the following selection program cannot be empty
3. The signal does not need to be triggered continuously. For example: "Power off", "Pause program", etc.

3.2 Register-Function out



The 16 function outputs of the register currently support the following states: idle state, program running state, paused state, program end, profinet disconnect, home position, collision trigger, automatic mode, not powered on, enabled, robot arm in motion, and ethernet disconnected

3.2 Register



64-way Boolean type input and output

32 byte input and output

32 floating-point inputs and outputs

3.2 Register_Precautions

1. Change the register name to conform to the naming rule "2-24 digits starting with a letter. Letter. Underline Combination".

2. When the function input is defined as "Run Program":

(1) If the backloading program is empty, then this register will continue to run after pausing, and when the robotic arm is idle, when this register is 1, the robotic arm will not respond.

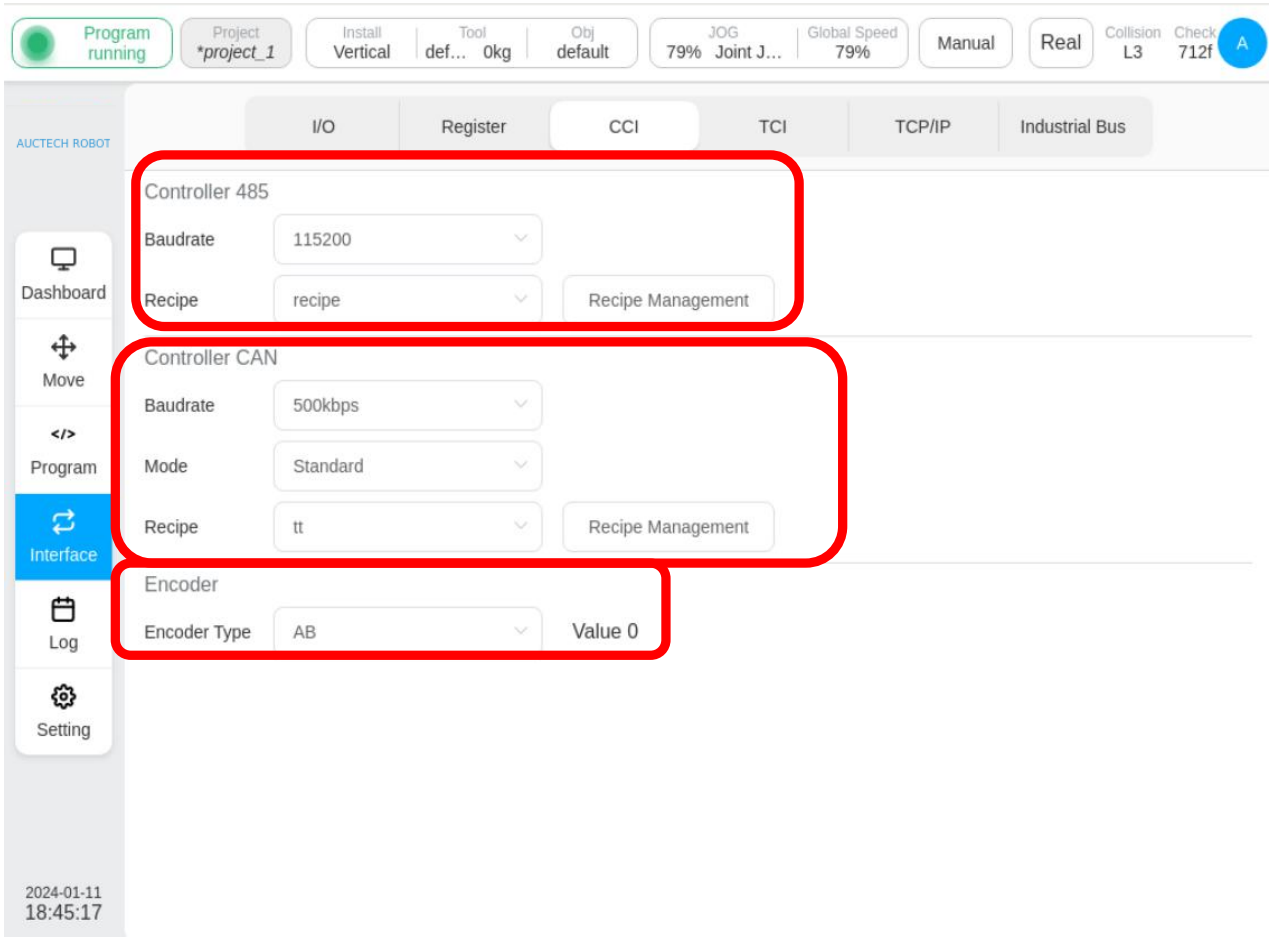
(2) If the loading program is set later. When the robot arm is idle, set this register to 1, the robot arm runs the currently loaded program.

3. The function input (including function IO) signal should not be triggered for a long time, such as triggering "power off" for a long time, "running the program", etc.

4. Be sure to save the project after the change!

5. In the registers and IO signals, the output signal can be manually changed (except for the function output).

3.3 CCI

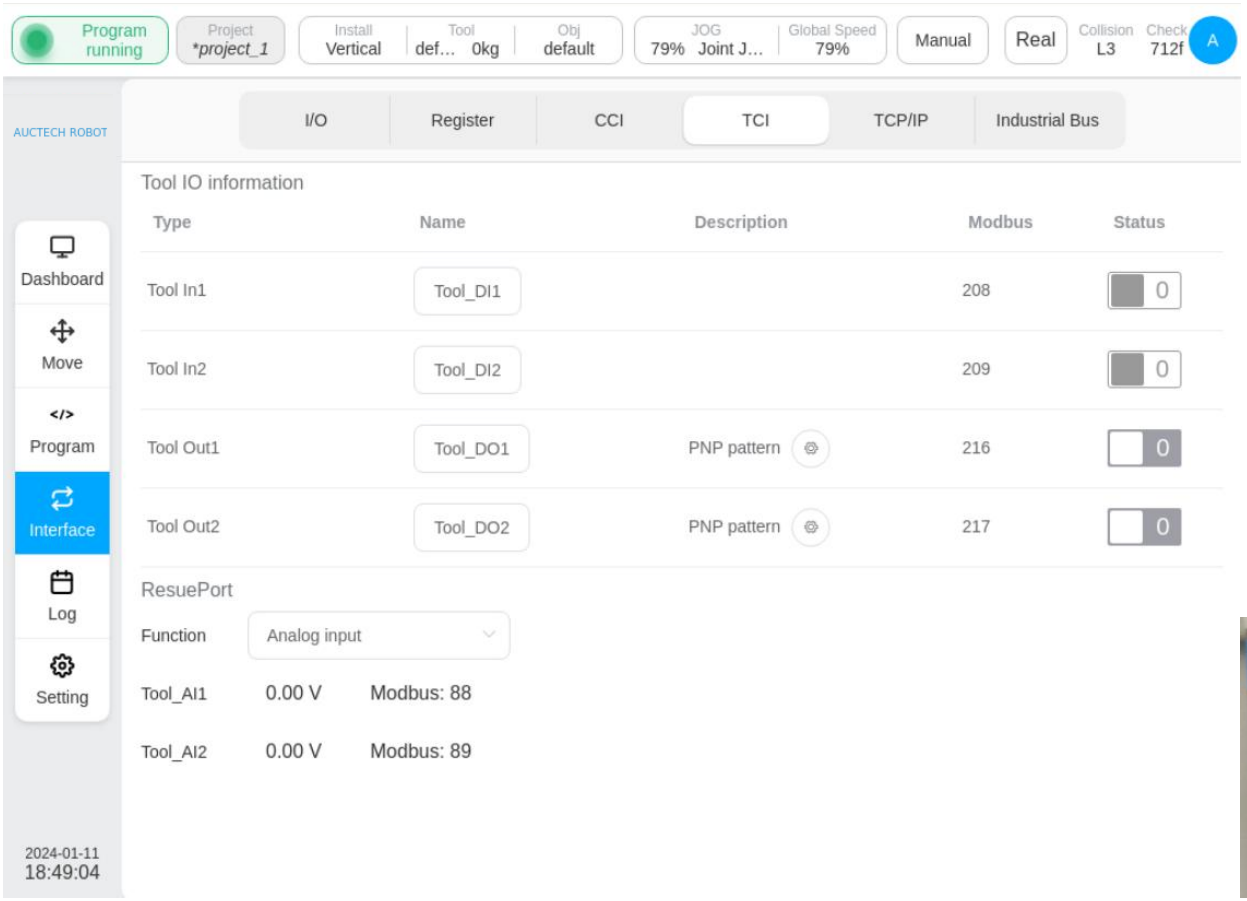


Port 485 supports the following baud rates: 9600, 19200, 38400, 57600, and 115200

The CAN port can be configured with baud rates of 10kbps, 20kbps, 50kbps, 100kbps, 125kbps, 250kbps, 500kbps, and 1000kbps

There are two types of encoder types in the control cabinet, namely AB and ABZ (the number of ABZ encoder lines, default is 0)

3.4 TCI

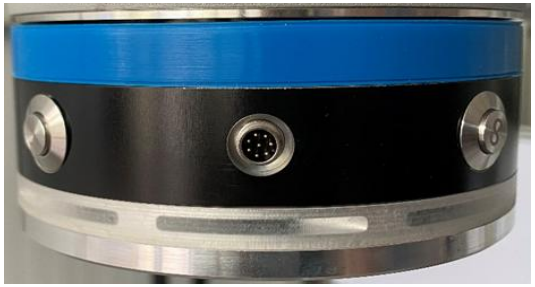


On the end of the robotic arm there are a total of:

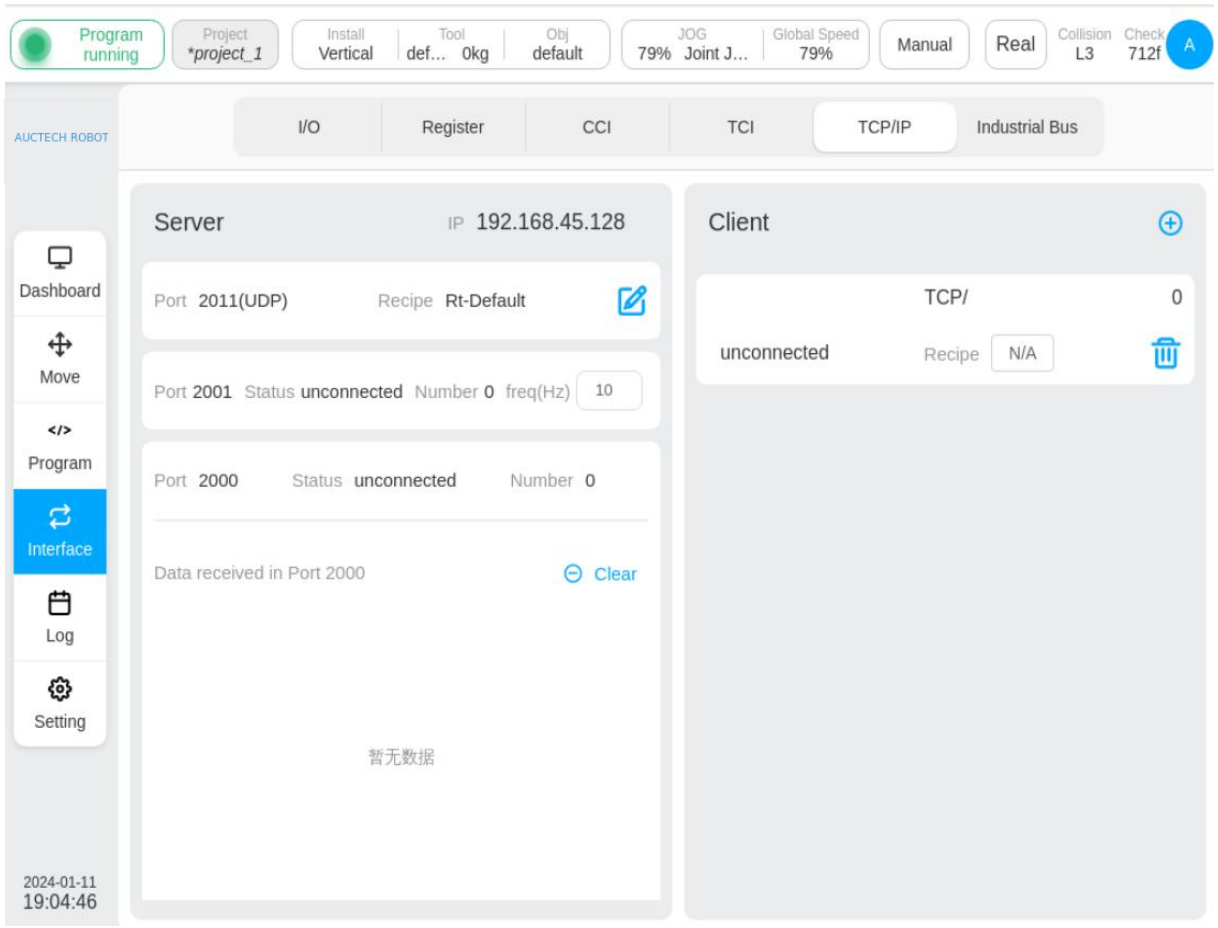
2-way tool inputs

2 tool outputs, PNP or NPN mode can be selected.

Reusable ports that can be used for analog inputs or 485 communication ports.



3.5 TCP/IP



Port 2000 can receive control commands from the robot. The information received by port 2000 is displayed in the lower area of the page.

The 2001 port sends information about the current status of the robot at a frequency of 10Hz.

The 2011 port is a UDP port that allows the robot to be controlled through the data items configured by the recipe.

For details about the use of ports 2000 and 2001, see Cobot External Communication Interface.

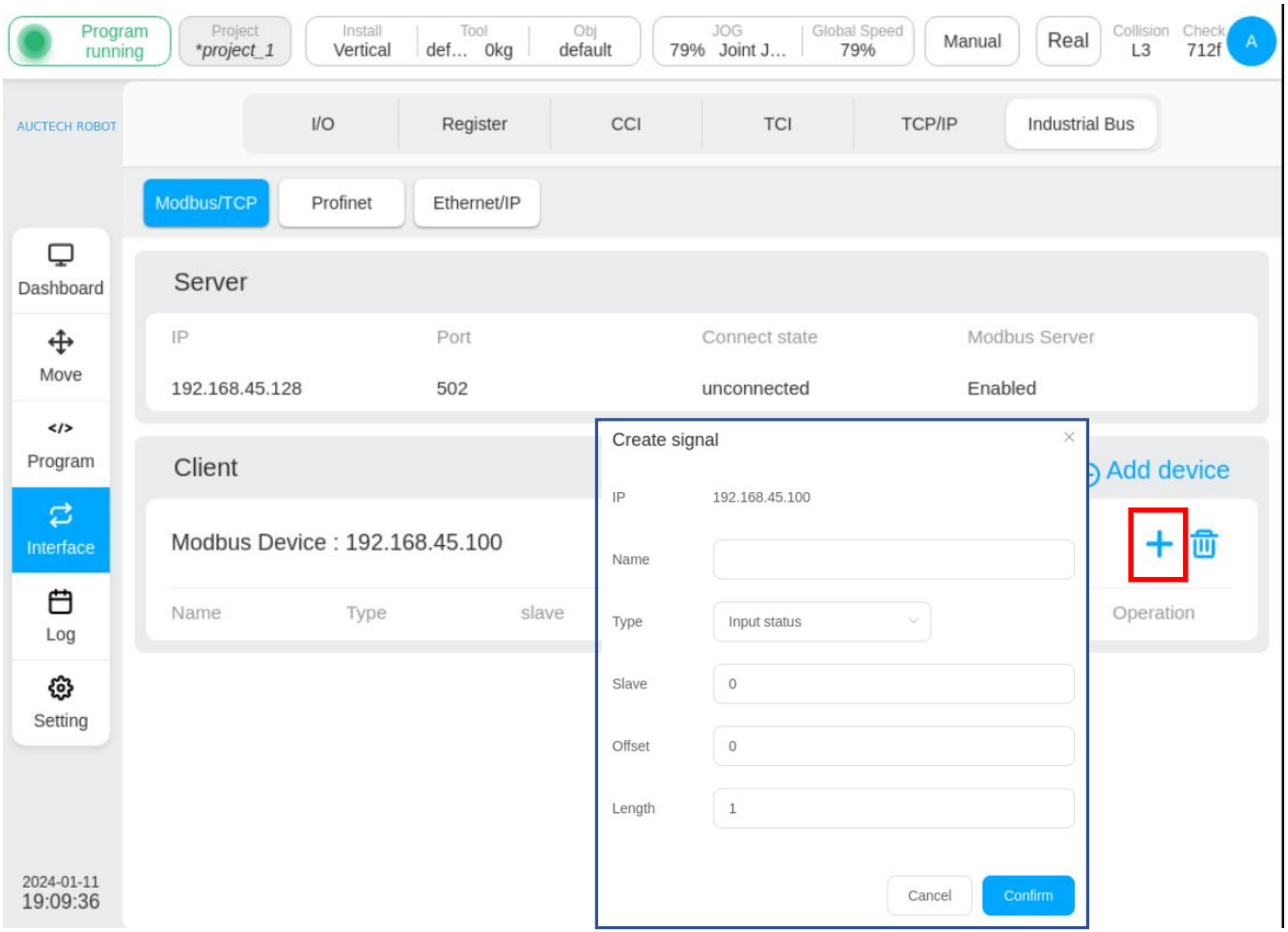
For details about how to use the 2011 port, see Recipes.

3.5 TCP/IP

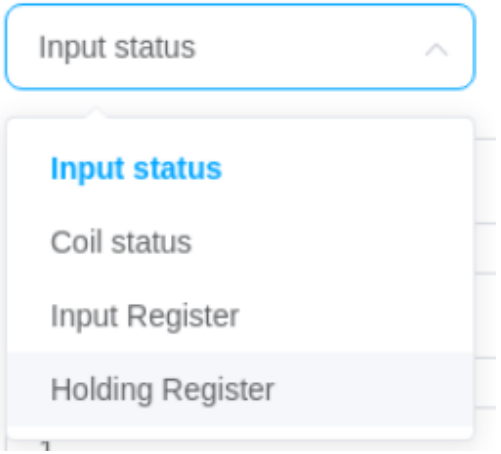
Port 2000

Instructions received	Return value
run(Program name) or run(Program name, speed percentage)	Feedback at the beginning：run start　　Execution failure feedback：run fail
speed	Perform success feedback：set success　　Execution failure feedback：set fail
stop	Perform success feedback：stop success　　Execution failure feedback：stop fail
pause	Perform success feedback：pause success　　Execution failure feedback：pause fail
resume	Perform success feedback：resume success　　Execution failure feedback：resume fail
login	Perform success feedback：login success　　Execution failure feedback：login fail
logout	Perform success feedback：logout success　　Execution failure feedback：logout fail
poweron	Perform success feedback：poweron success　　Execution failure feedback：poweron fail
poweroff	Perform success feedback：poweroff success　　Execution failure feedback：poweroff fail
enable	Perform success feedback：enable success　　Execution failure feedback：enable fail
disable	Perform success feedback：disable success　　Execution failure feedback：disable fail
shutdown	Perform success feedback：shutdown success　　Execution failure feedback：shutdown fail
state	Feedback on the status of each system，format：(1) : (2) : (3) : (4)　　concentrate：(1) Robot status, (2) Program status, (3) Safety controller status, (4) Robot operation mode
isprogfinish	Whether the feedback program runs at the end of the program, the end feedback character is 1, and the unfinished feedback character is 0
getlasterror	Feedback detailed error information for the bot，format: error_id:error_message\n

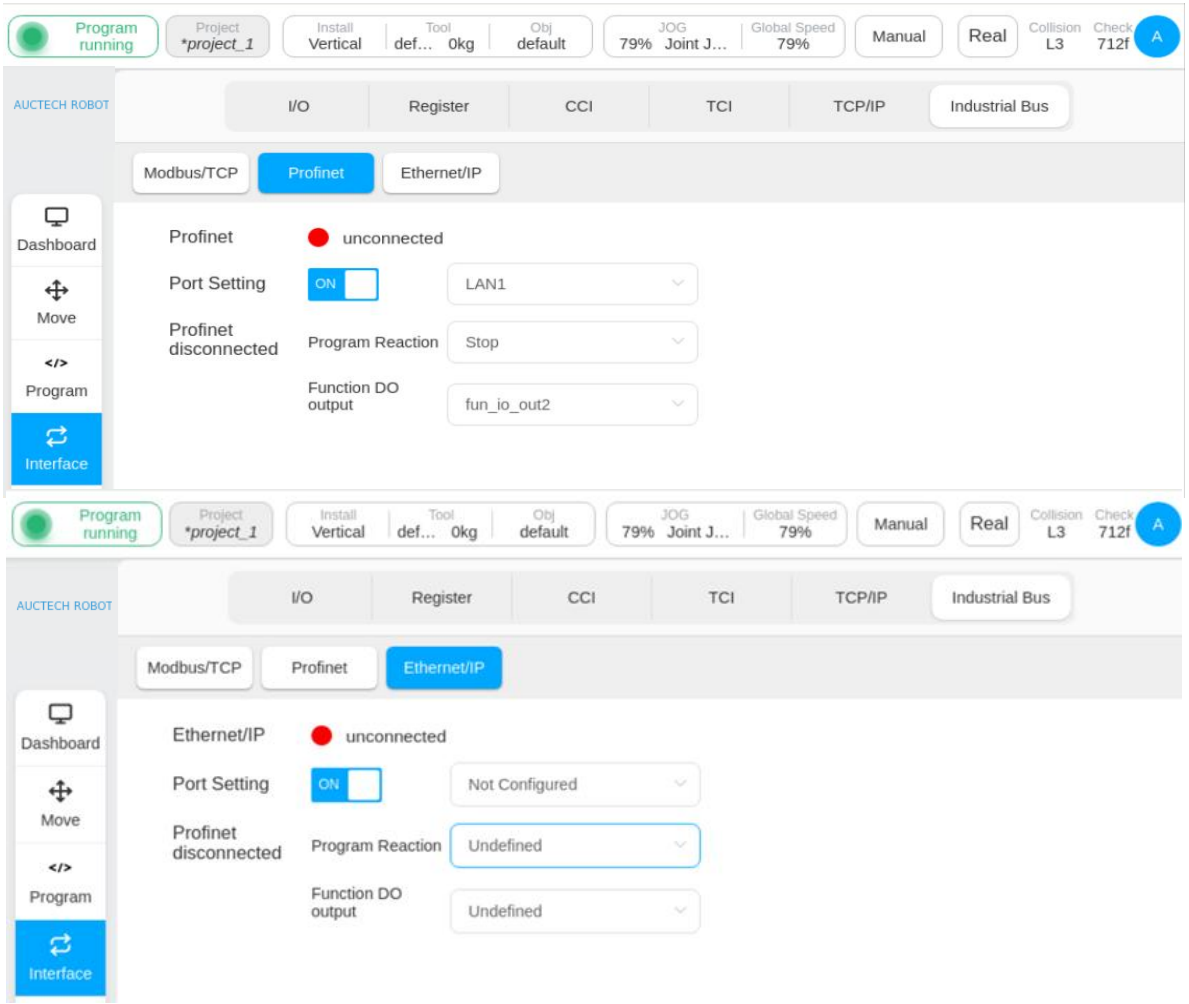
3.6 MODBUS/TCP



Modbus Communication: When the robotic arm is used as a client, external devices can be added, such as modbus to io, etc.

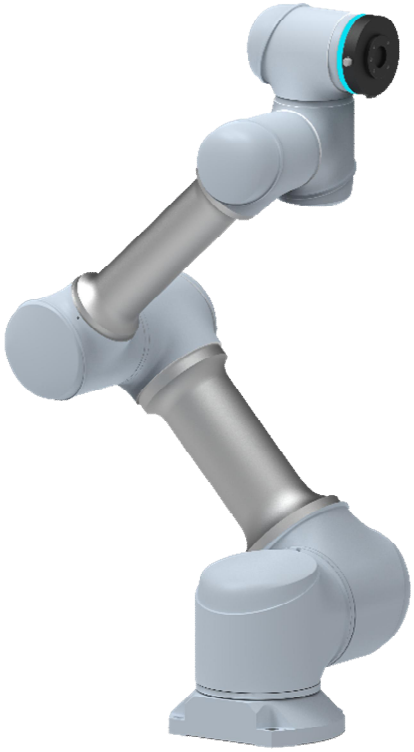


3.7 Profinet & Ethernet/IP



1. Program Response: Undefined, Suspend Program, Stop Program

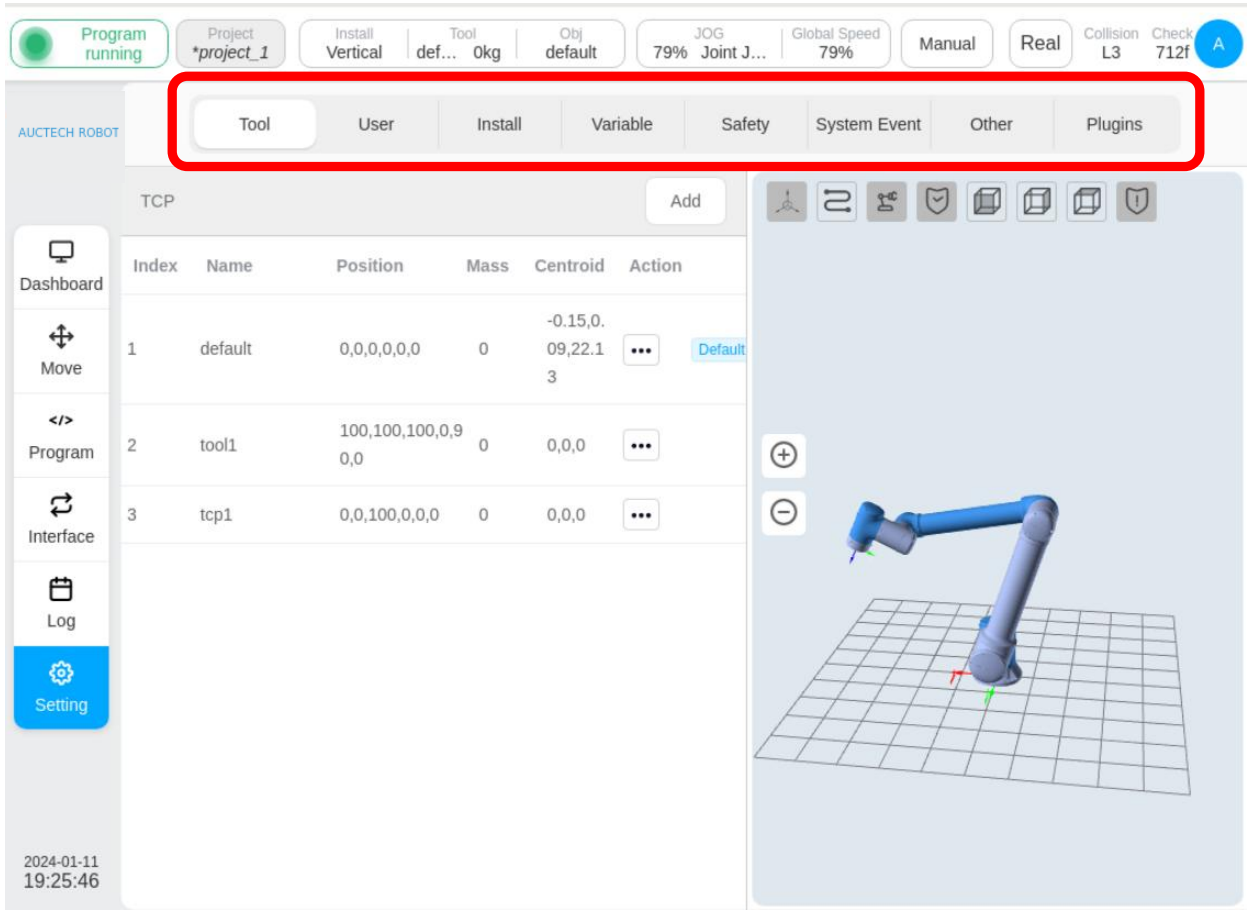
2. The function DO output is synchronized with the function output configuration in the I/O subpage



Lecture 4

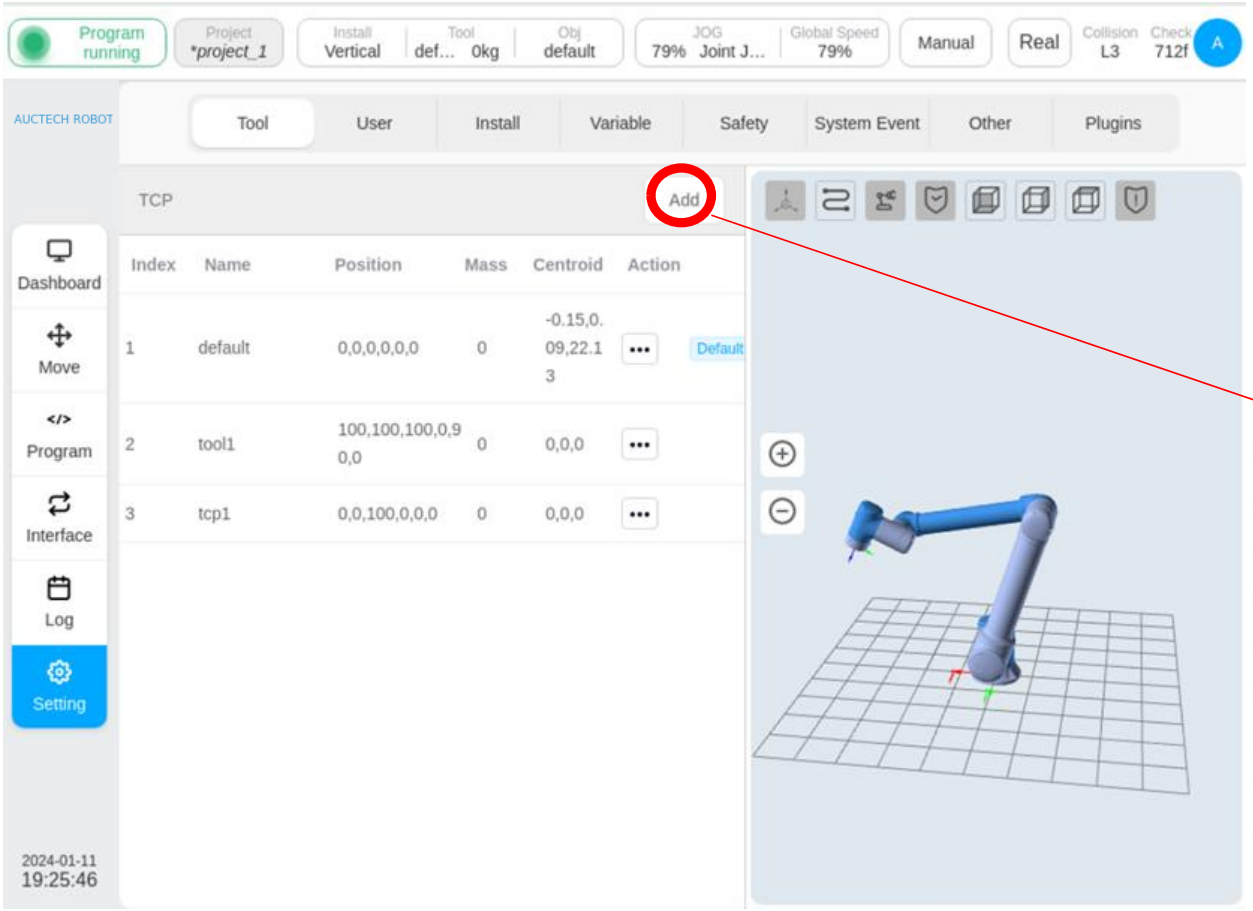
Set up

4.1 Set up

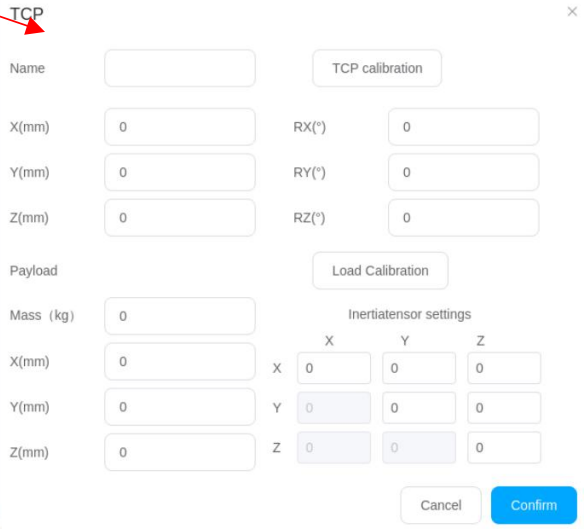


Tool settings, workpiece coordinate system, installation settings, variable settings, security settings, system settings, other settings, plug-in settings

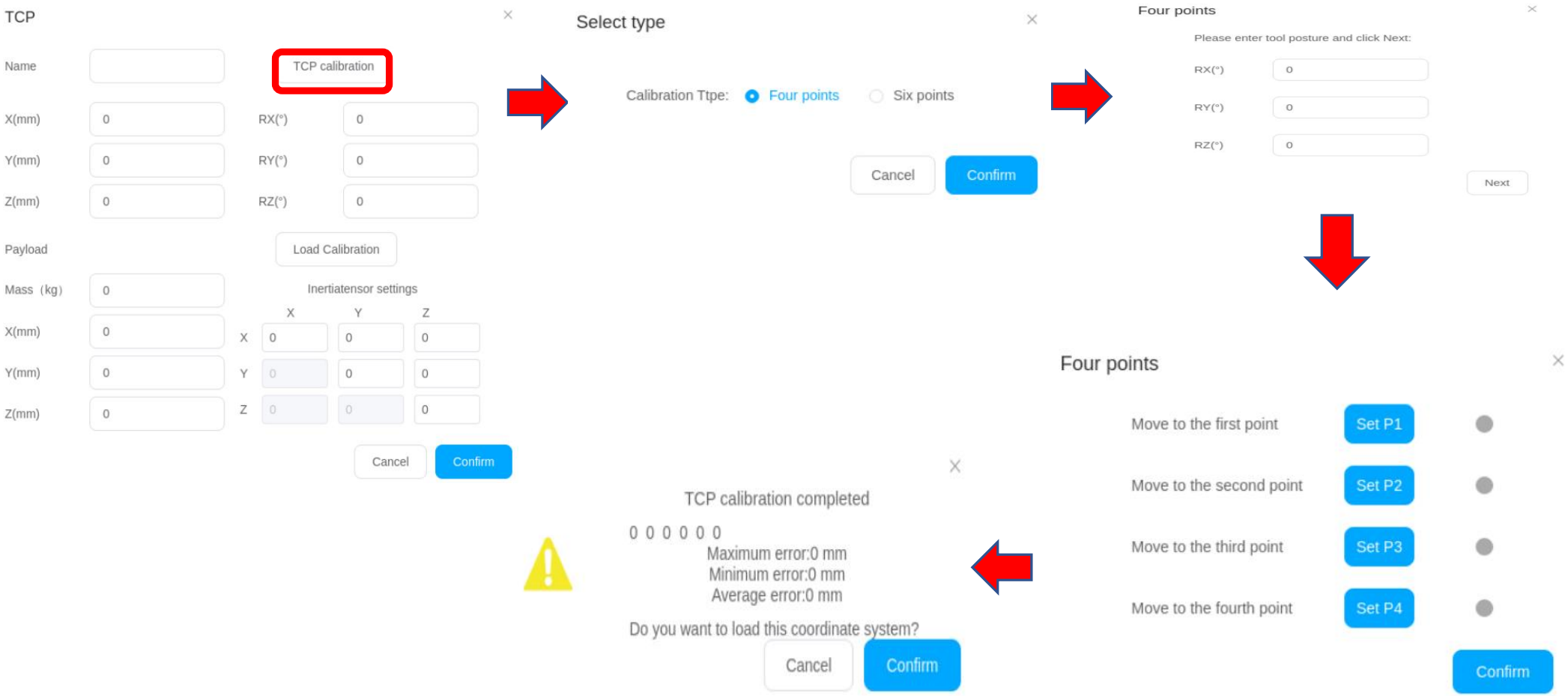
4.1 Tool setting



Click the Add button to create



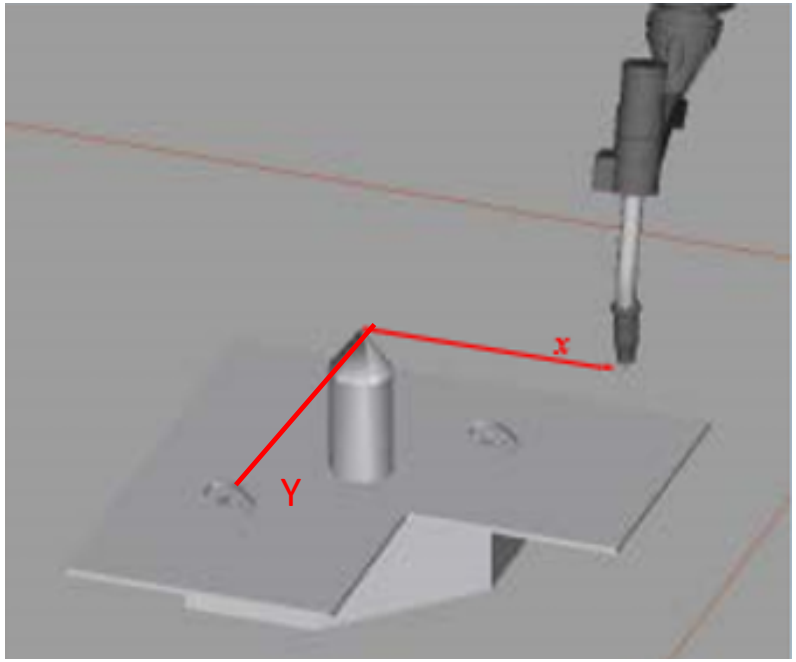
4.1 Tool setting- 4-point calibration process



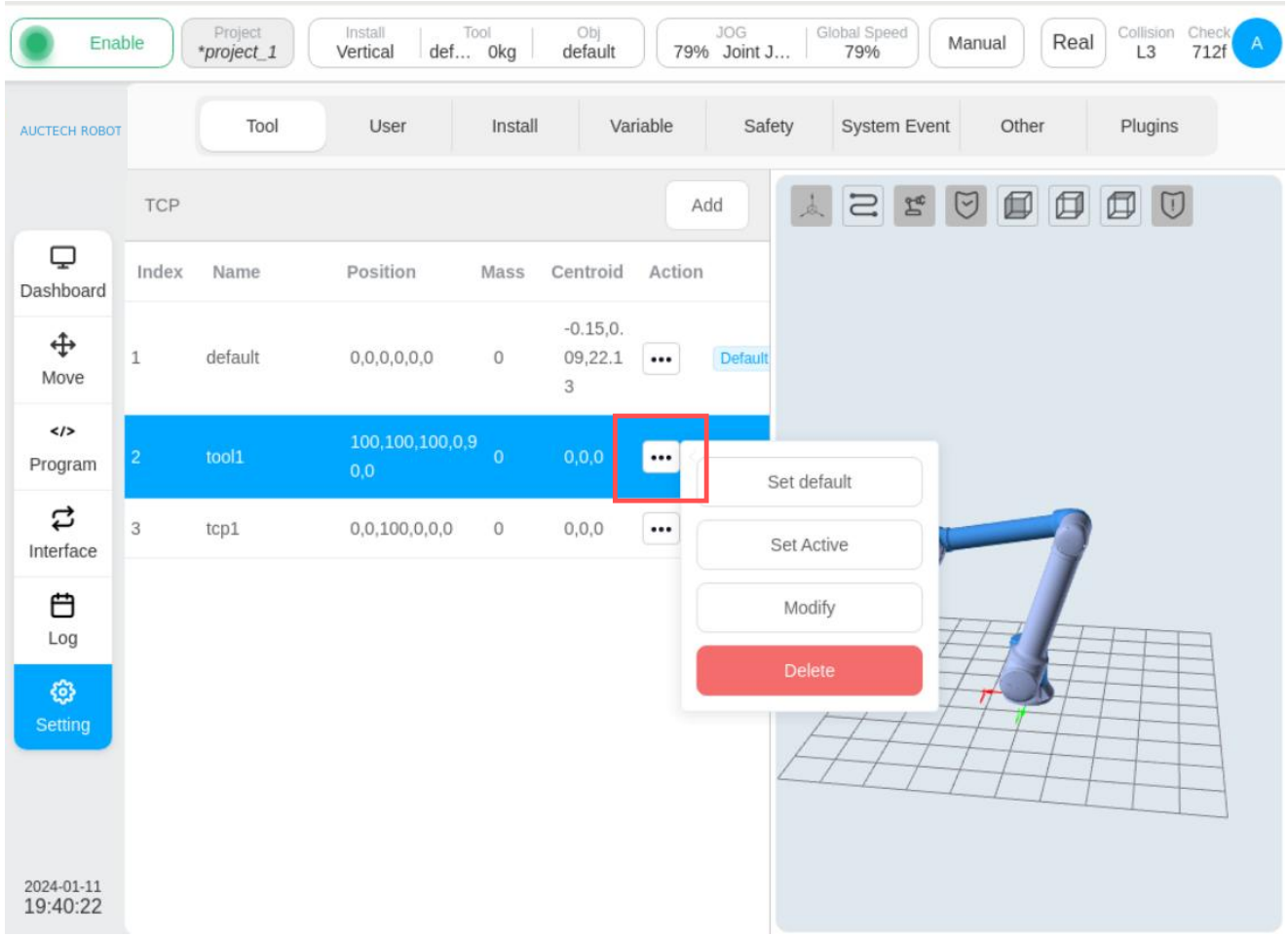
4.1 Tool setting- TCP calibration method

4-point method: take the calibration points P1, P2, P3, P4, and the difference between the 4 points is 90° and not in the same plane.

6-point method: On the basis of the 4-point method, two calibration points in the X+ and Y+ directions are added.

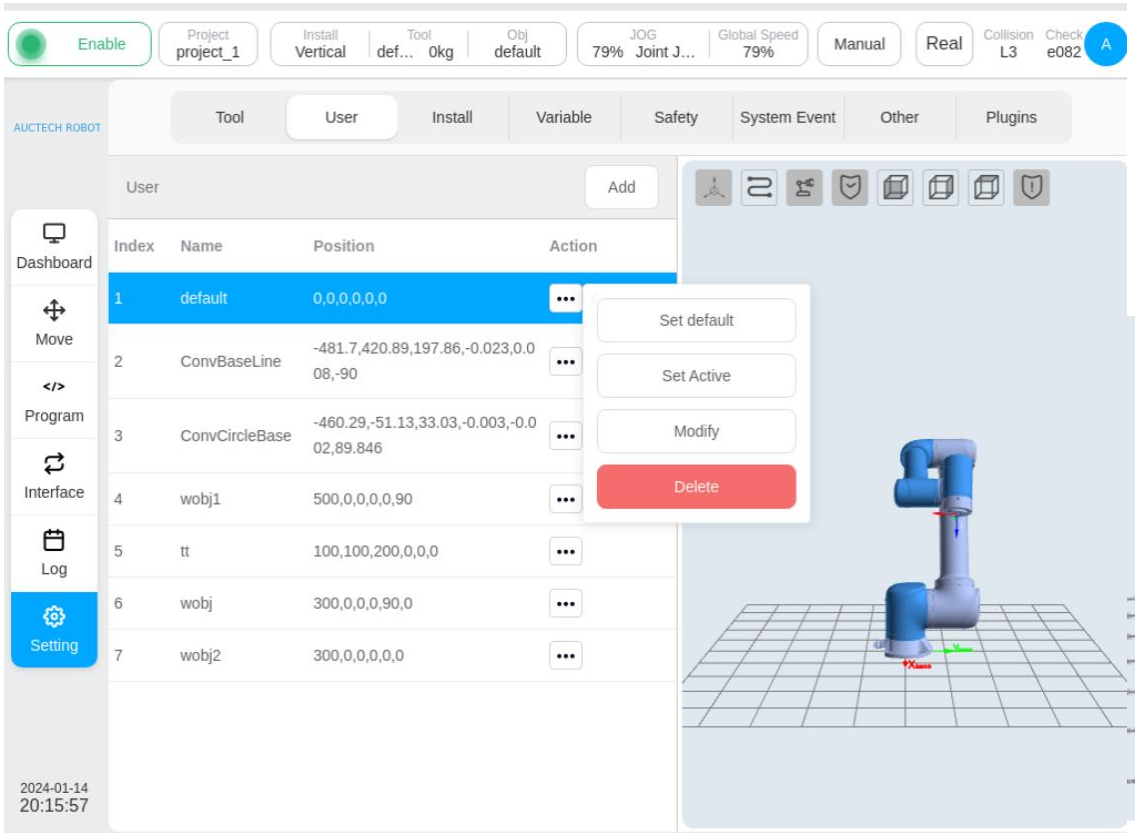


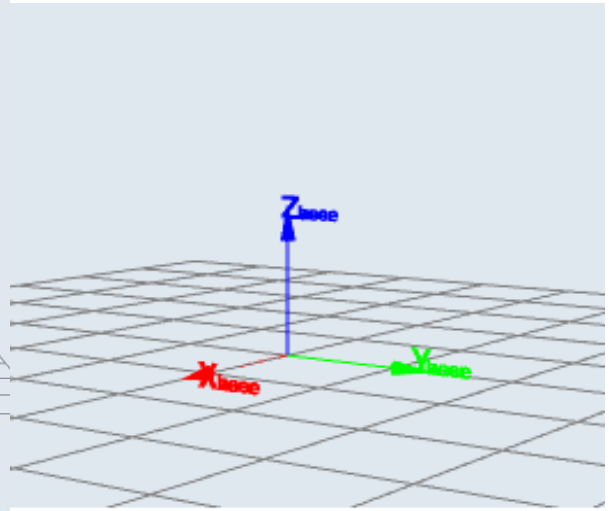
4.1 Tool setting



Note: Tools that have been set as default or current cannot be deleted!

4.2 User setting





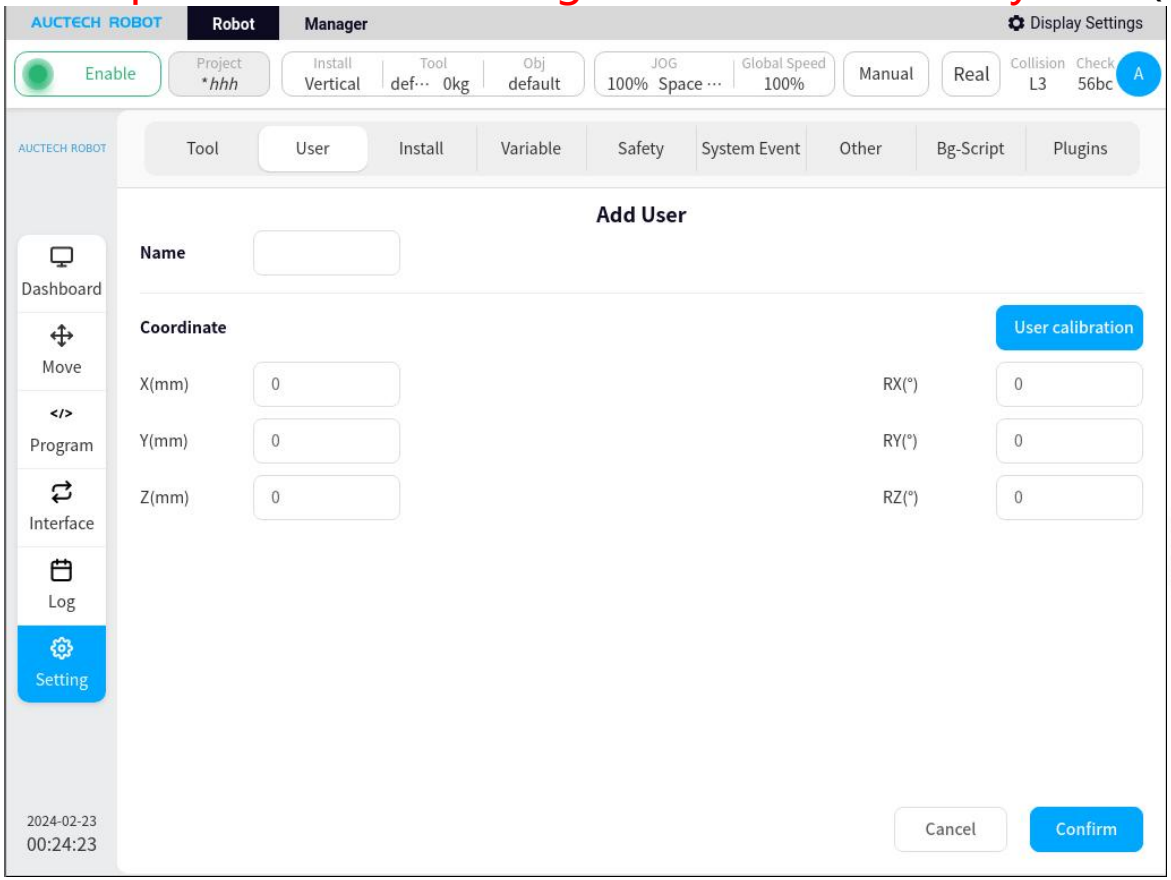
Red indicates the X-axis

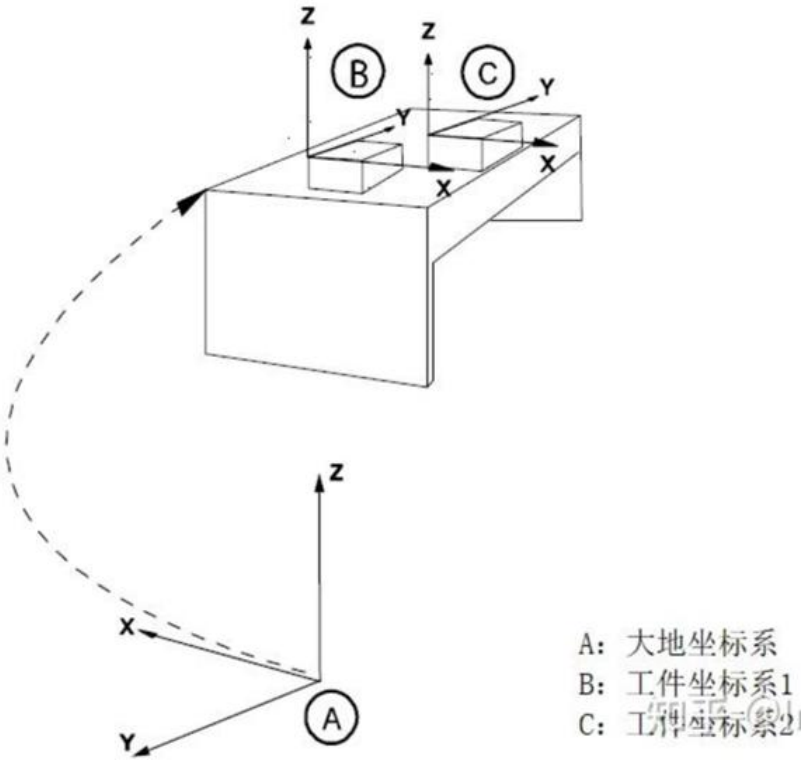
Green indicates the Y-axis

Blue indicates the Z-axis

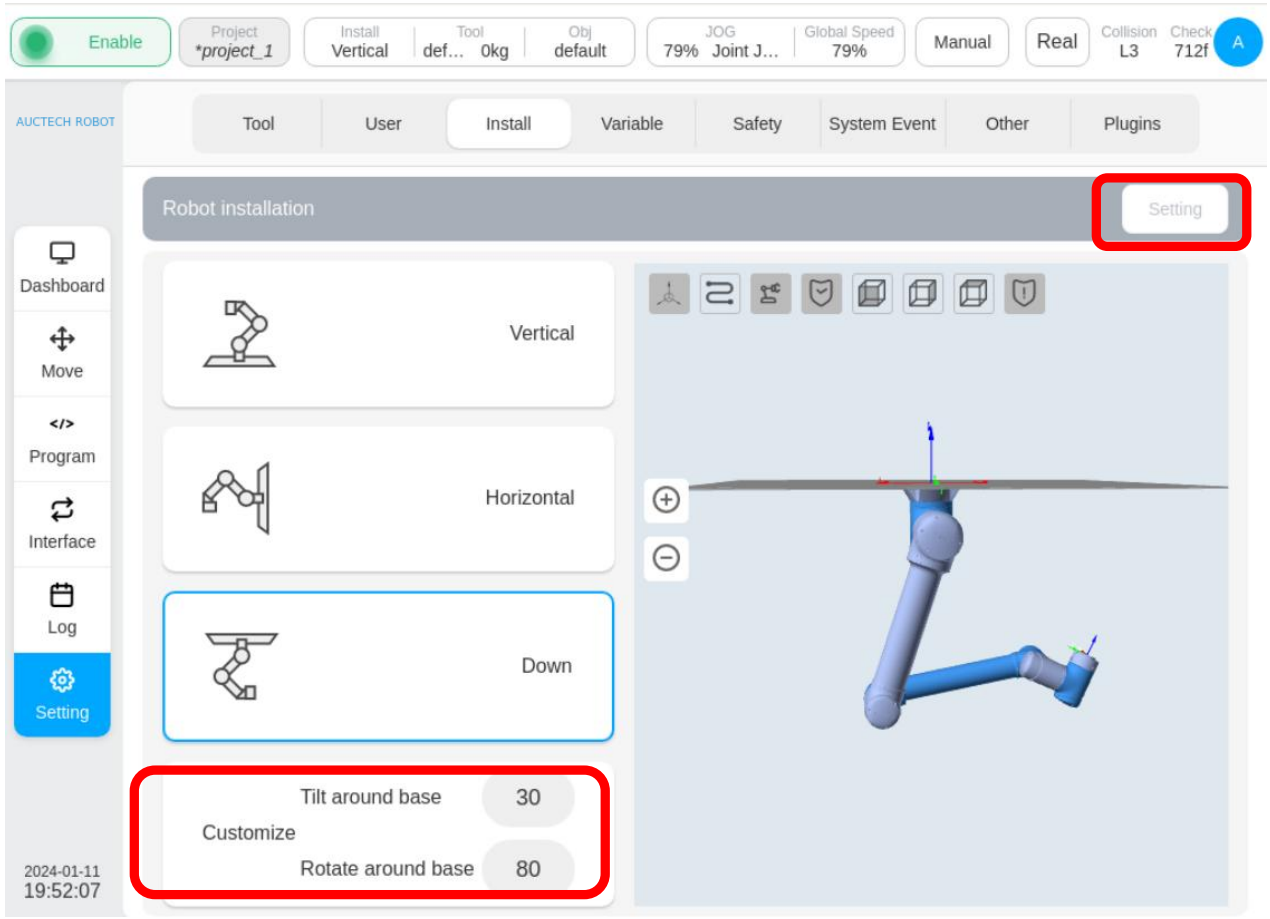
4.2 User setting

The workpiece coordinate system corresponds to the workpiece: it defines the position of the workpiece relative to the geodetic coordinate system (or other coordinate system).



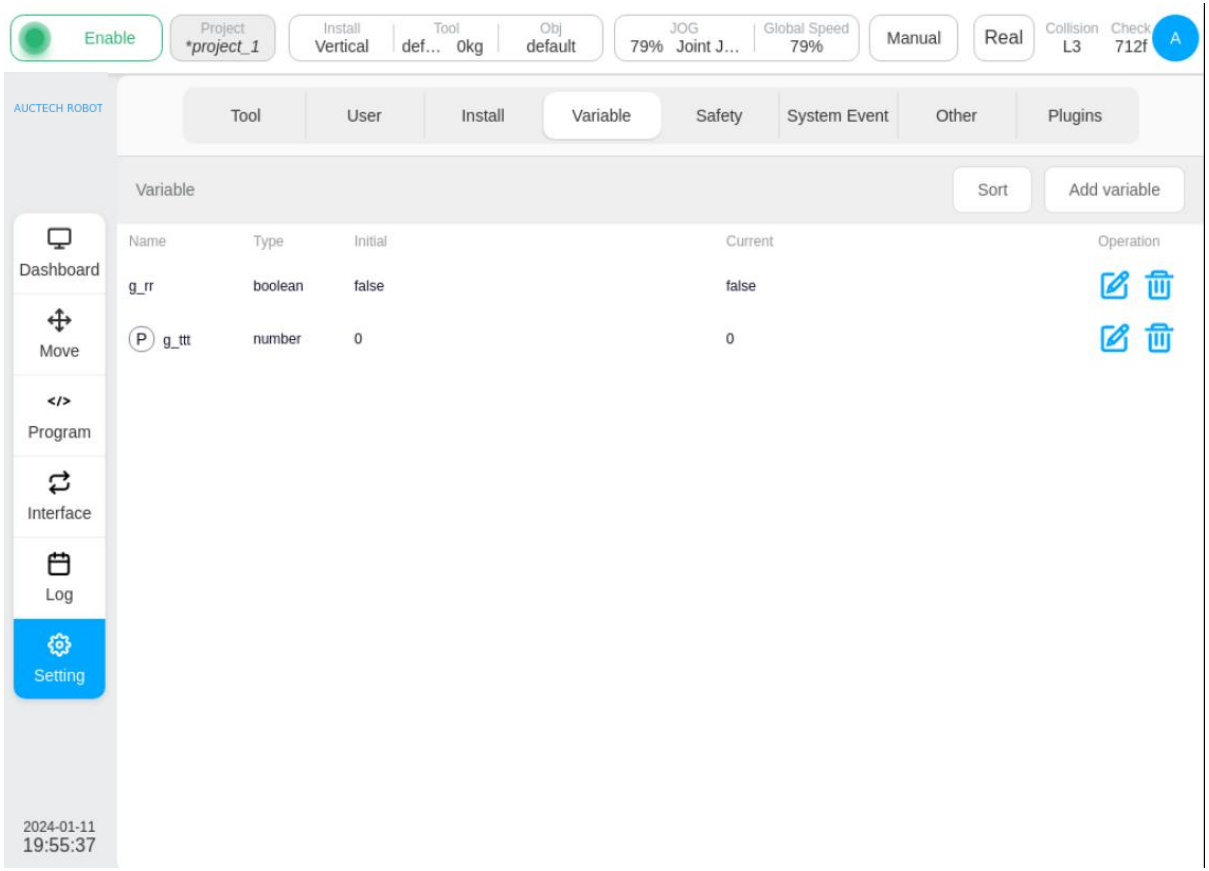


4.3 Install setting



The RY and RZ value setting range is -180°~180°

4.4 Variable settings

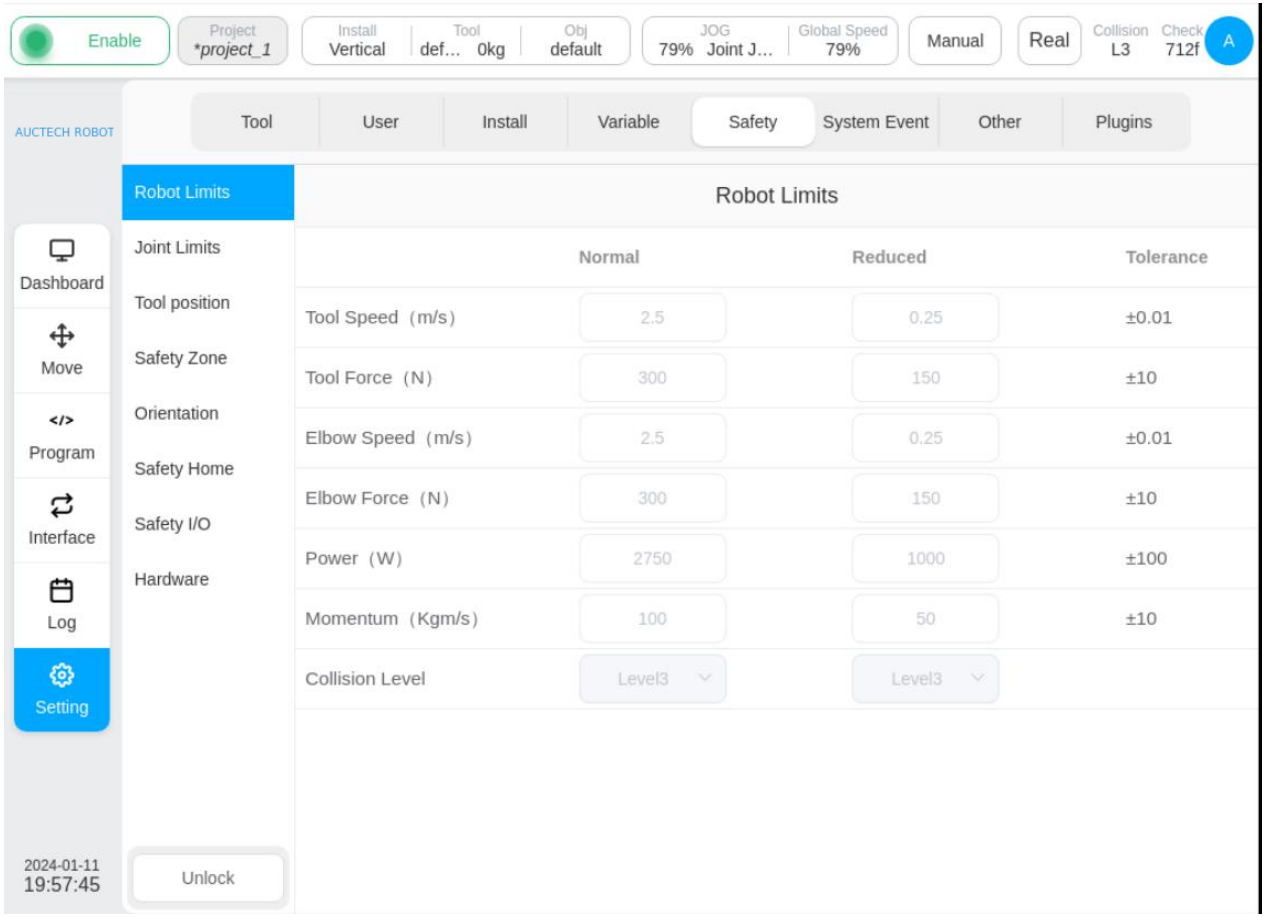


There are 13 types of variables, including boolean, number, string, num_list, pose, joint, timer, pose_speed, joint_speed pose_acc, and joint_acc pose_list and joint_list

The word P is displayed in front of the name of the persistent variable

Note: The variables added in the settings are system variables, which can be used by each program in the current project.

4.5 Security settings



Safe mode:

(1) Normal mode: the security mode activated by default;

(2) Reduction mode: This mode can be activated with Secure Input IO

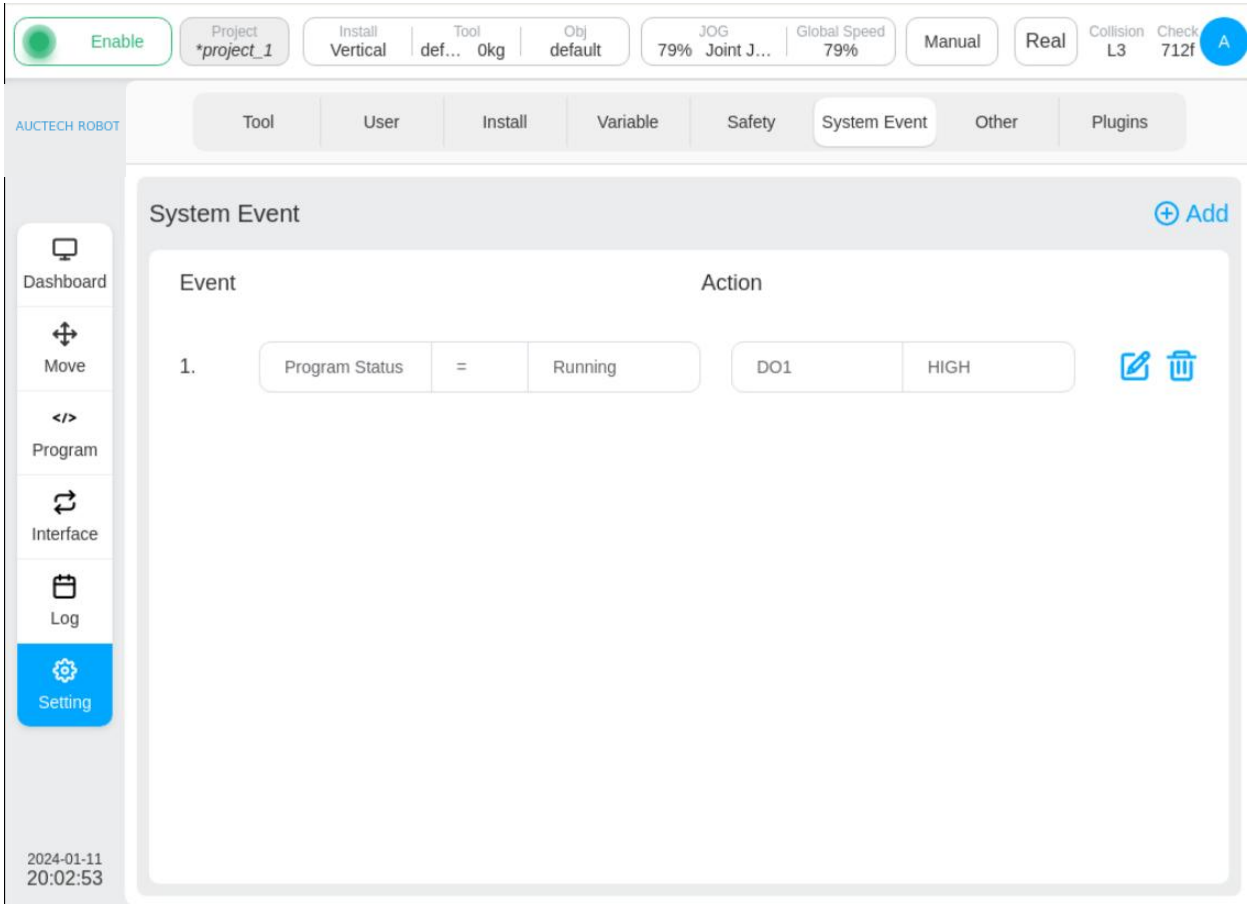
Security Precautions:

1. It must be set when the mechanical arm is powered off.

2. When the project is switched, the parameters in the security settings will be invalid and need to be set anew.

Unlock password: 123

4.6 System Event

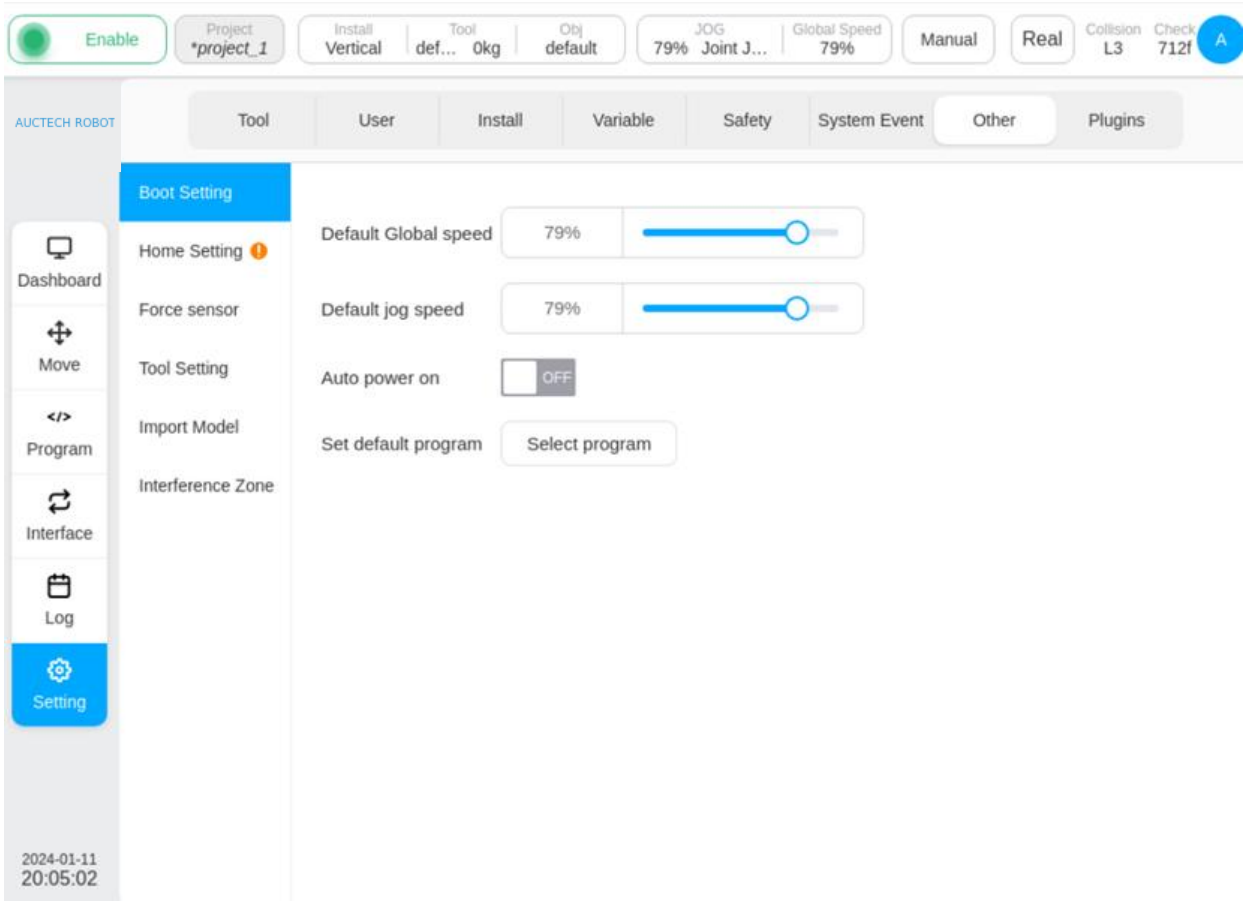


System Event Description: When the event is true, the response will be triggered.

For example, when the **p**rogram is running, the BOOL output register 1 is set to true.

The current system events are not affected by the program.

4.7 Other –Boot Setting

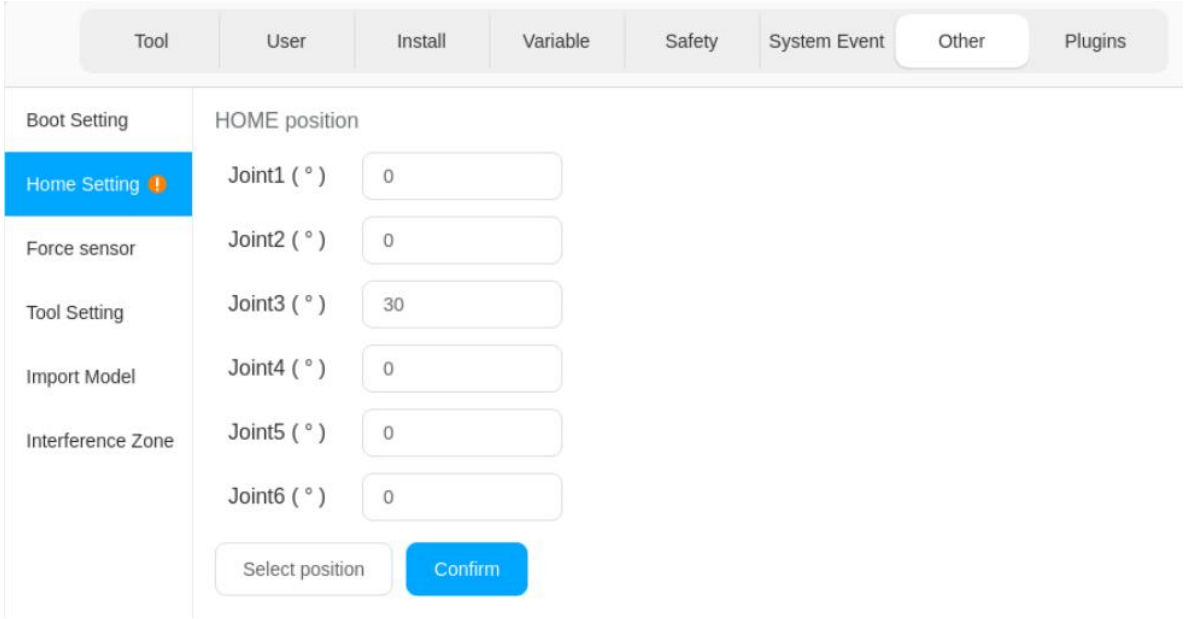


Default Global/JOG Speed: Sets the default speed when the computer is first turned on

Automatic power-on of the robot: After the control cabinet is turned on, the robotic arm body will automatically power on and enable the operation after the control cabinet is turned on.

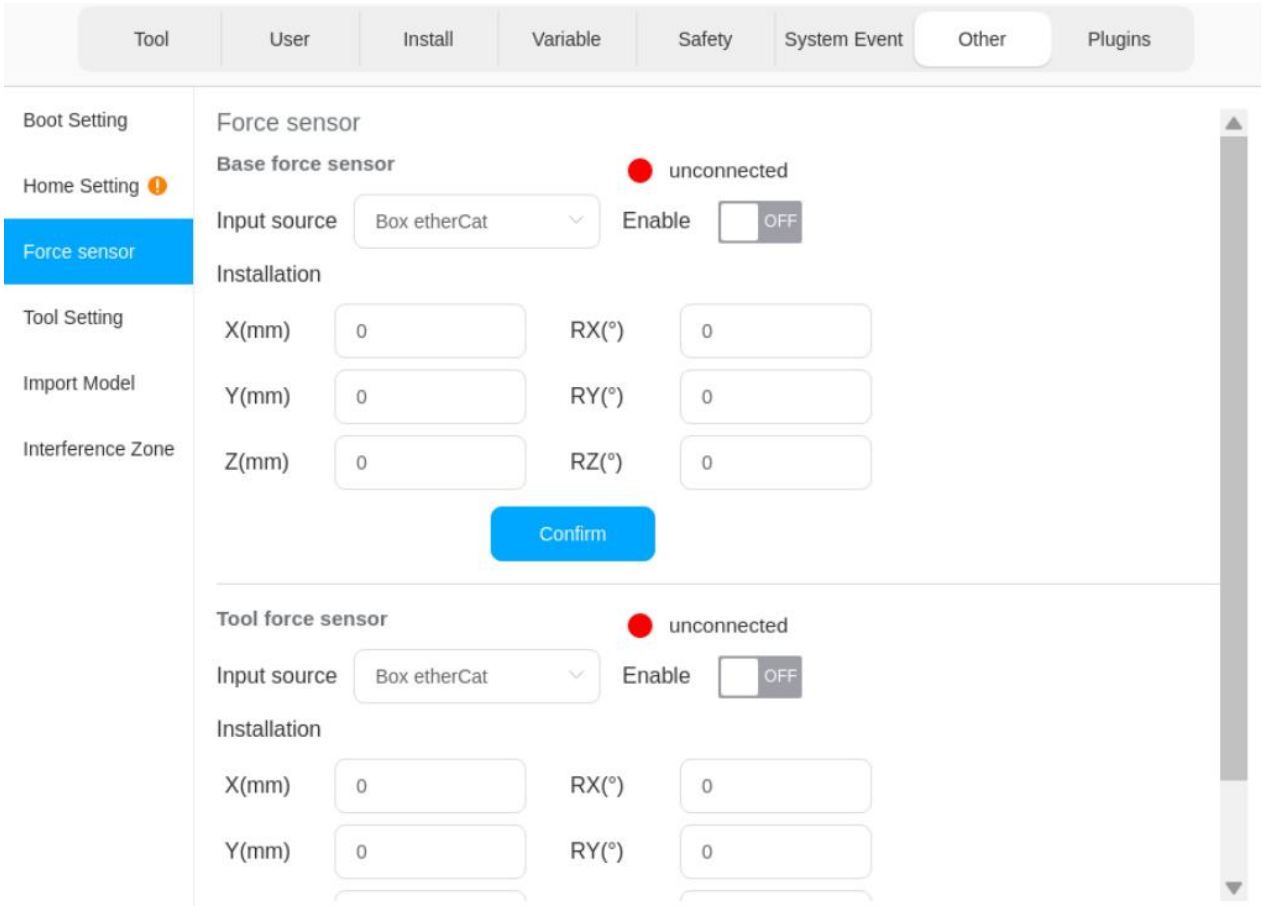
Set Default Program: Set the default program.

4.7 Other – Home Setting



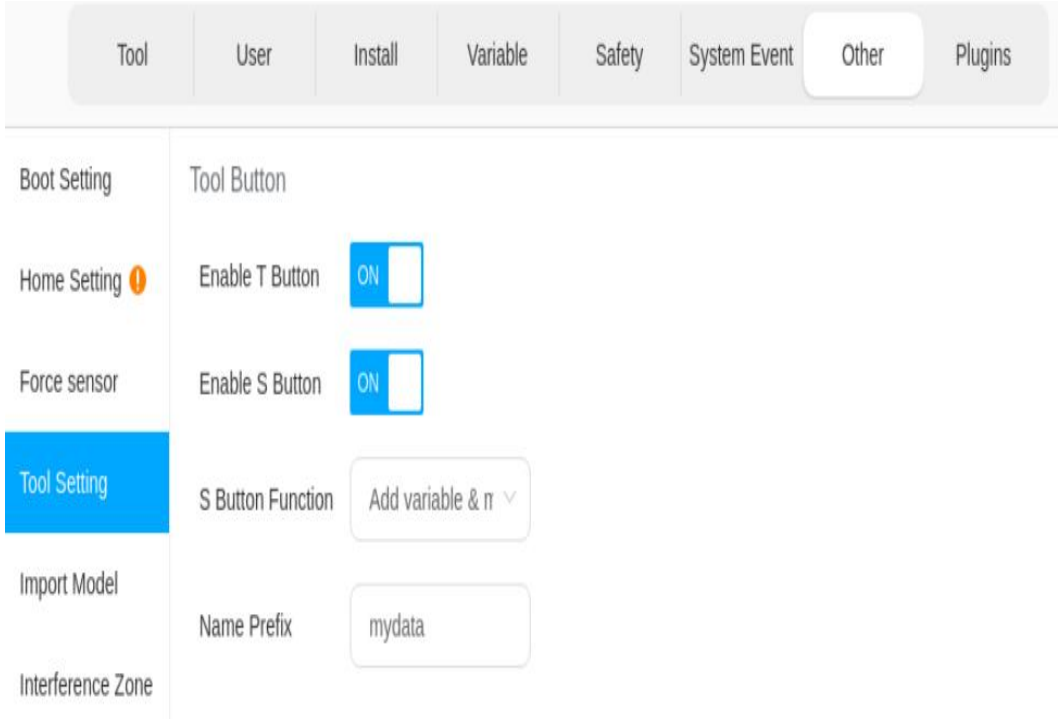
Change or teach the Home point, and after the change is completed, it needs to be synchronized in the safety Home.

4.7 Other –Force sensor



Settings for the force sensor: including settings such as importing topology files.

4.7 Other – Tool Setting

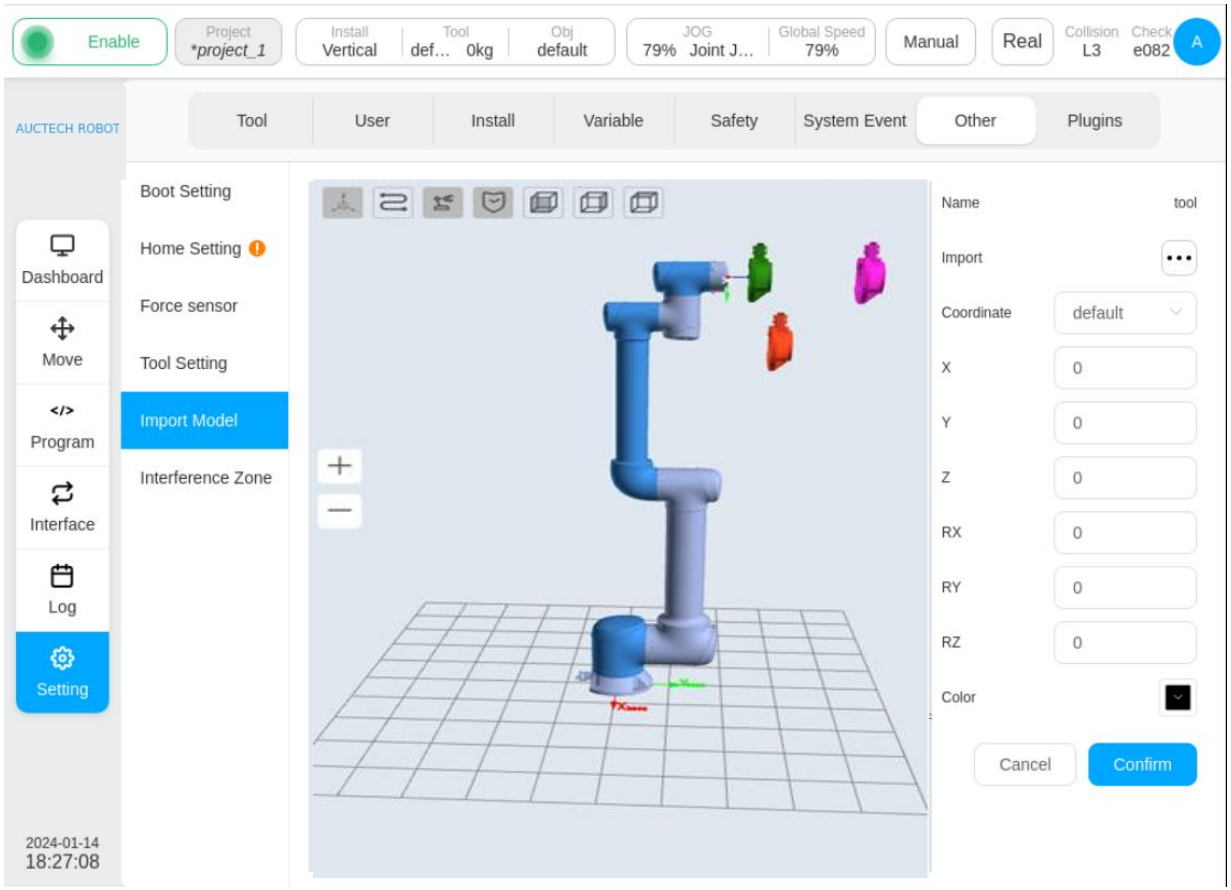


Start the T button: long press this button to drag and teach.

Start the S button: S button function: add variables; Add motion commands, as well as add variables and motion commands

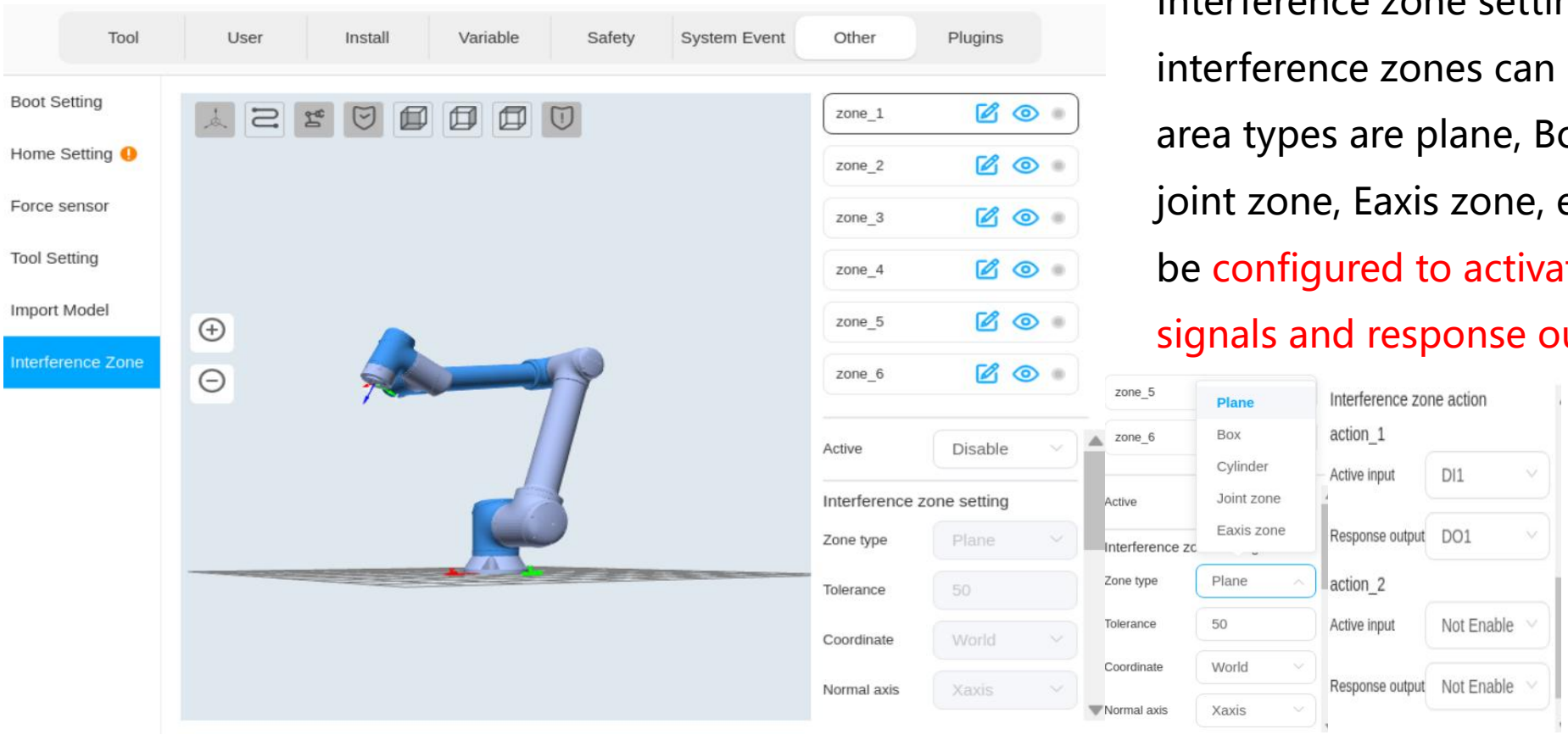
When selecting the s button to add a variable or adding a variable and motion command, you also need to configure the variable name prefix.

4.7 Other – Import Model



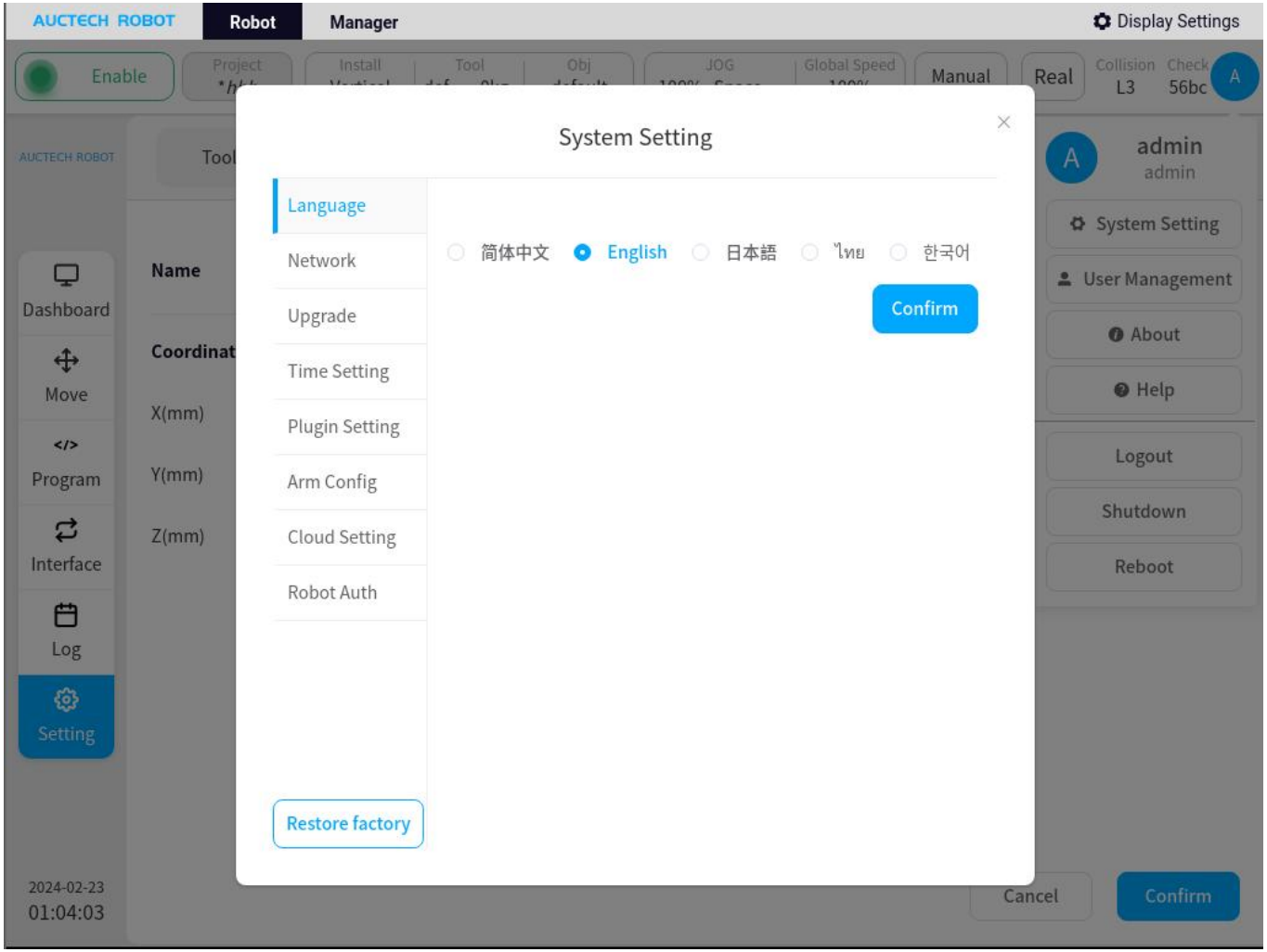
Model Import Settings: To import a model in a 3D image, the model must be a TSL file.

4.7 Other – Interference Zone



Interference zone setting: 6 interference zones can be set, the area types are plane, Box, cylinder, joint zone, Eaxis zone, each area can be configured to activate by input signals and response output signals

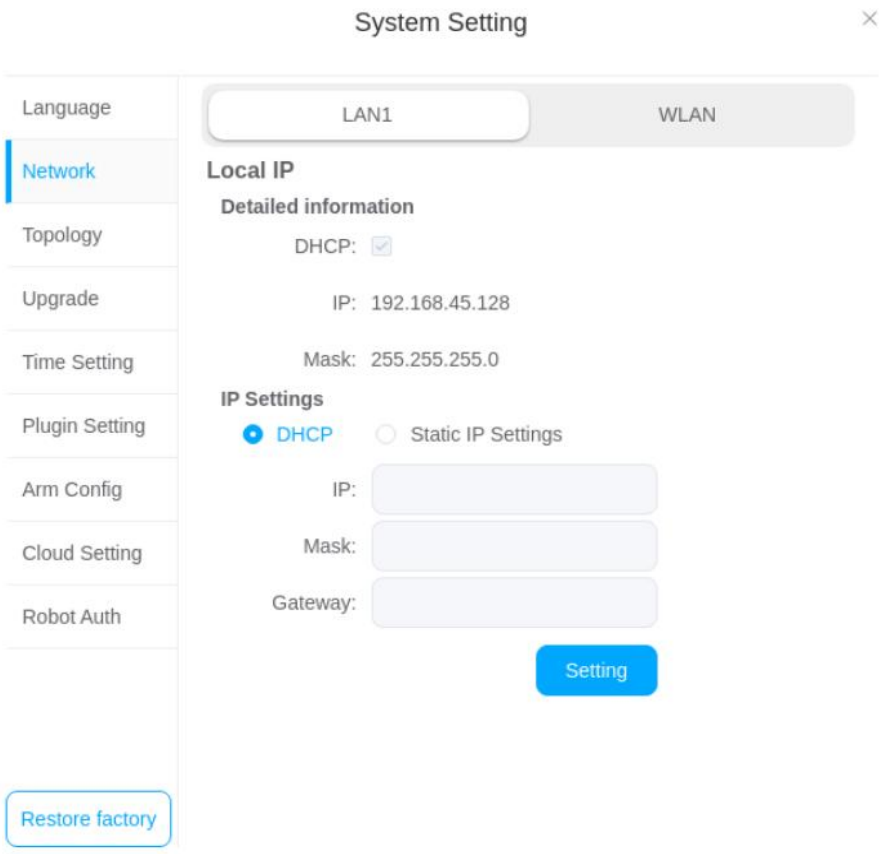
4.8 System Settings - Language

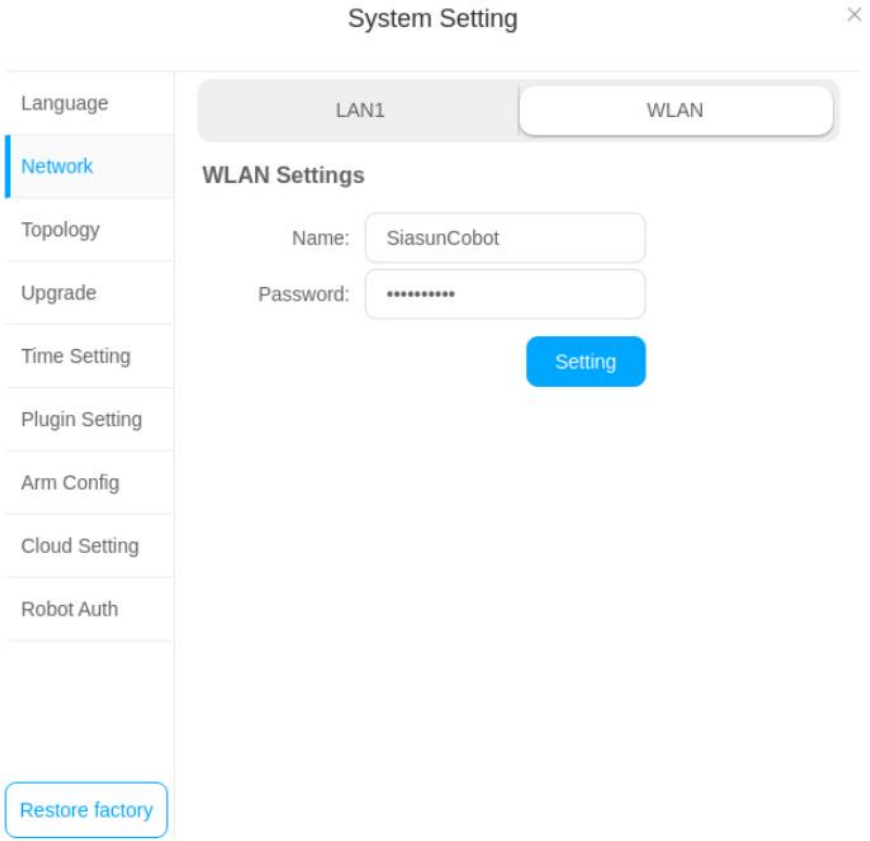


Supported languages include: Chinese, English, Japanese, Thai

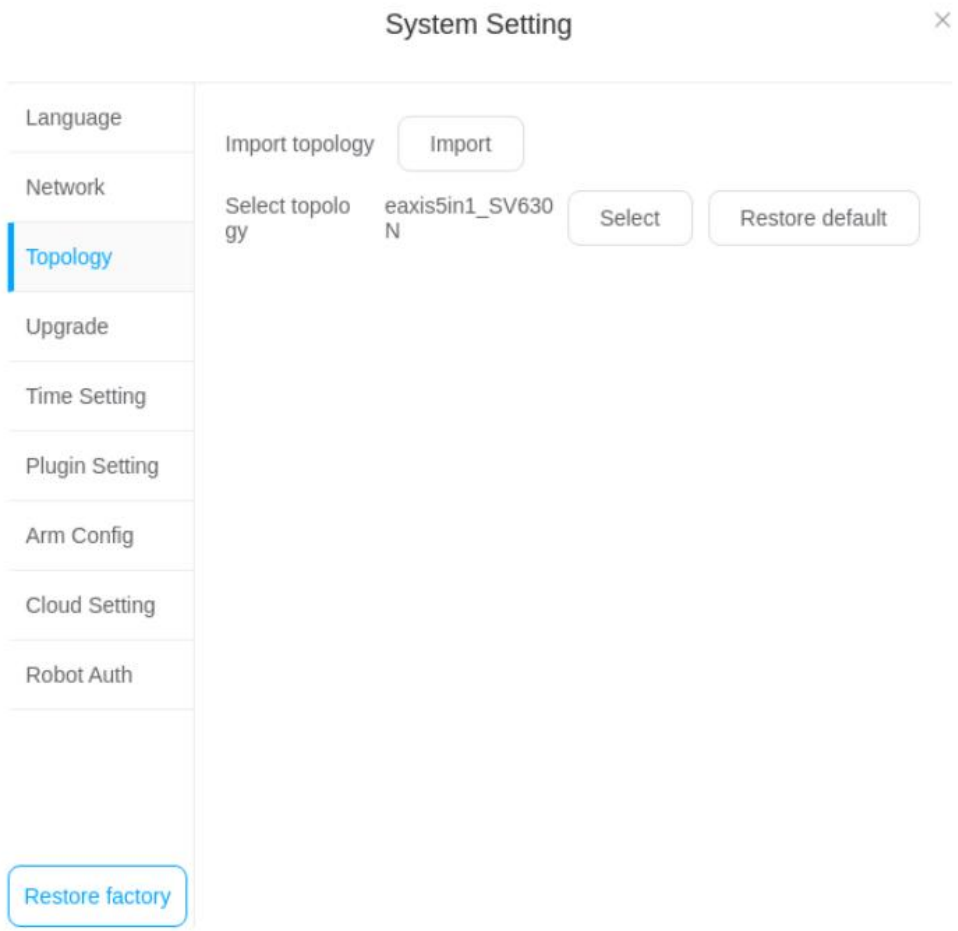
4.8 System Settings - Network

The network settings include LAN, WLAN settings.



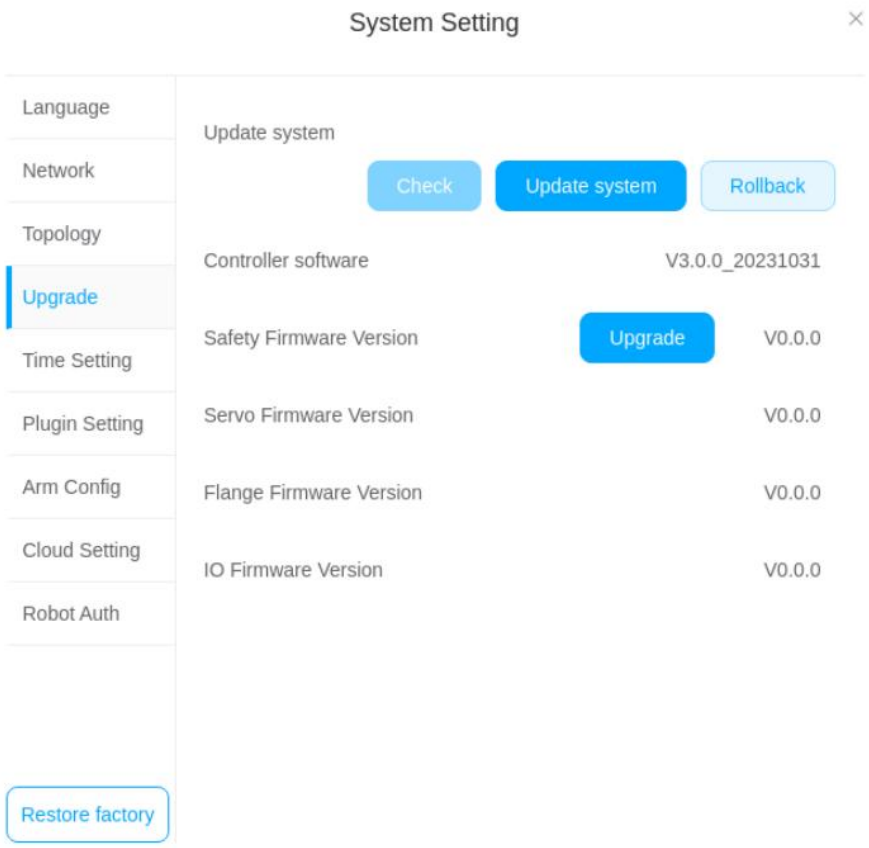


4.8 System Settings - Topology



Topology management is used to import topology files, such as expanding external axes and 6D force control sensors.

4.8 System Settings - Upgrade

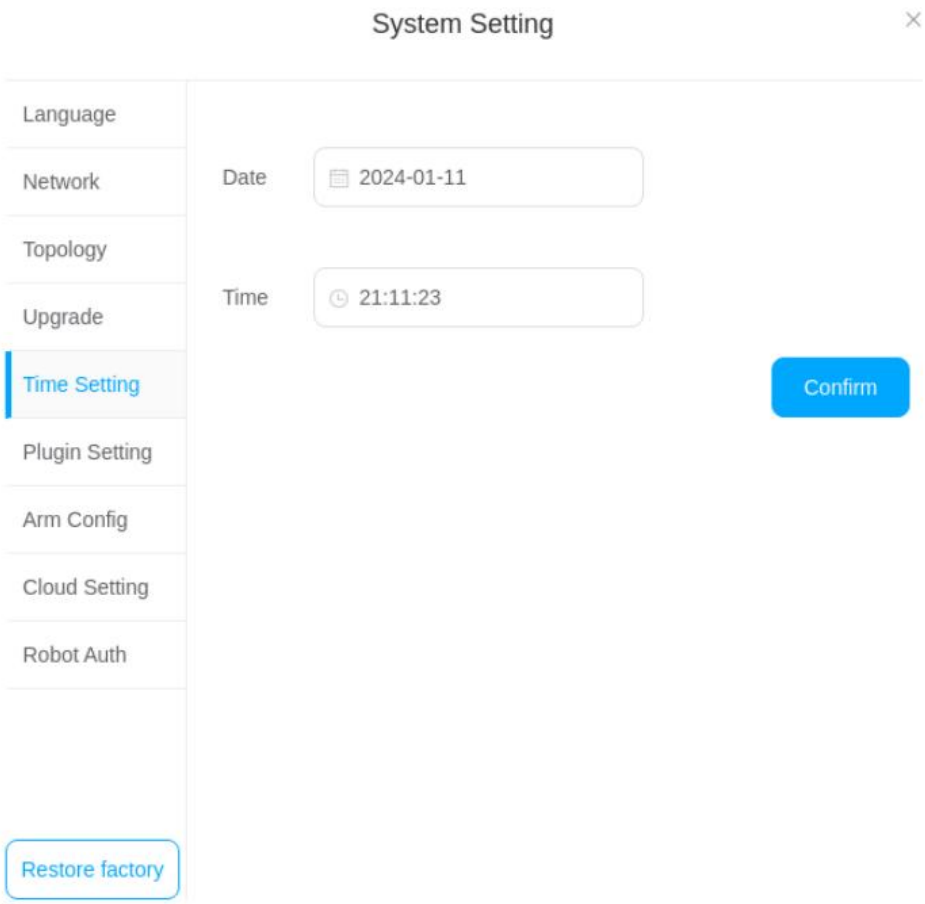


Put the upgrade package in a USB flash drive in FAT32 format and insert it into the USB port of the control cabinet. Select Local Update. Find the file on the USB flash drive and OK.

Software version update or rollback files end with the .update suffix, select the required file and click the "Rollback" button will prompt the reboot to take effect (make sure the robot arm is in a power-off state)

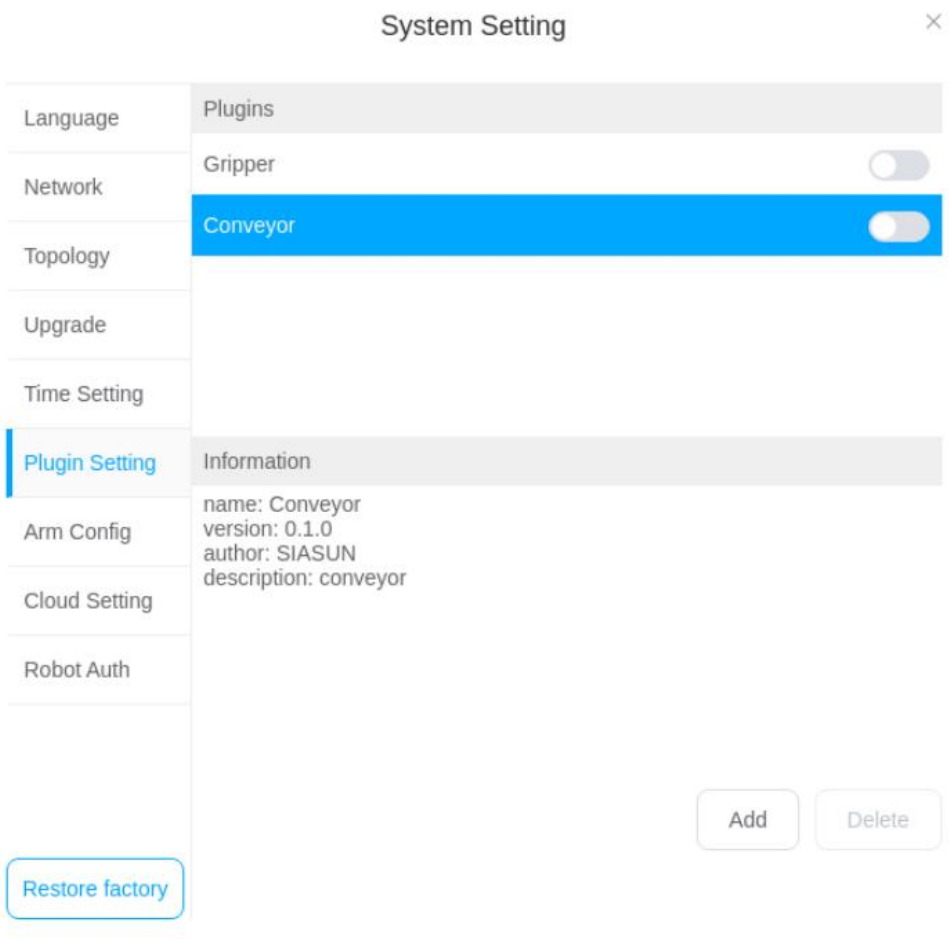
Note: The control cabinet needs to be restarted after the update, and the USB flash drive should not continue to be plugged into the USB port.

4.8 System Settings – Time Setting



Time Setting: Set the current time (the log records are based on this time)

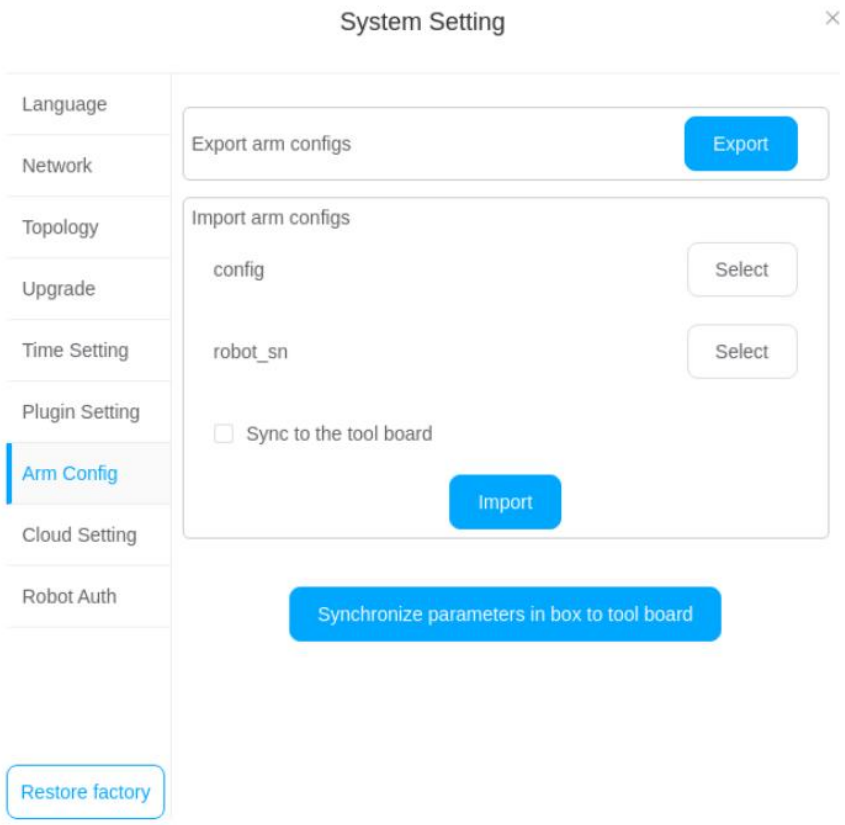
4.8 System Settings – Plugin Setting



Plugin management: Manage the plugins used.

The supported plug-ins are visual, welding, palletizing, and external shafts.

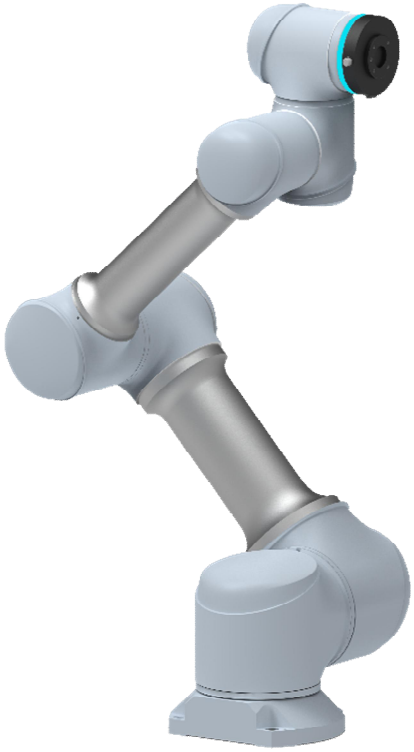
4.8 System Settings – Arm Config



Robot parameters:

1. The configuration of the robotic arm can be imported and exported

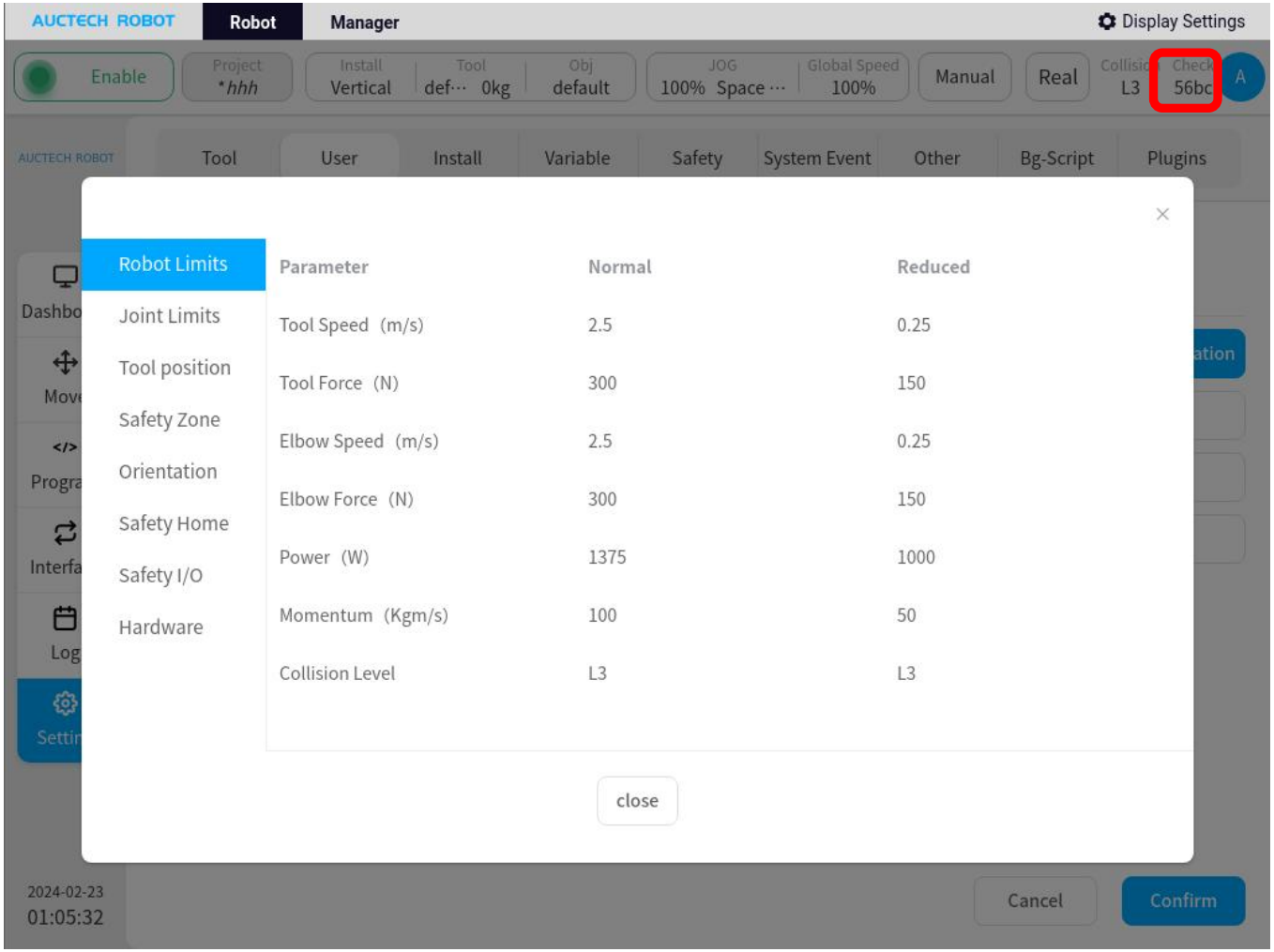
2. Synchronize the parameters of the control cabinet to the terminal version



Lecture 5

Security settings

5.1 Modify the security configuration



Click "Security Check" on the upper status bar, and the following dialog box will pop up, you can view the currently activated security configuration parameters.

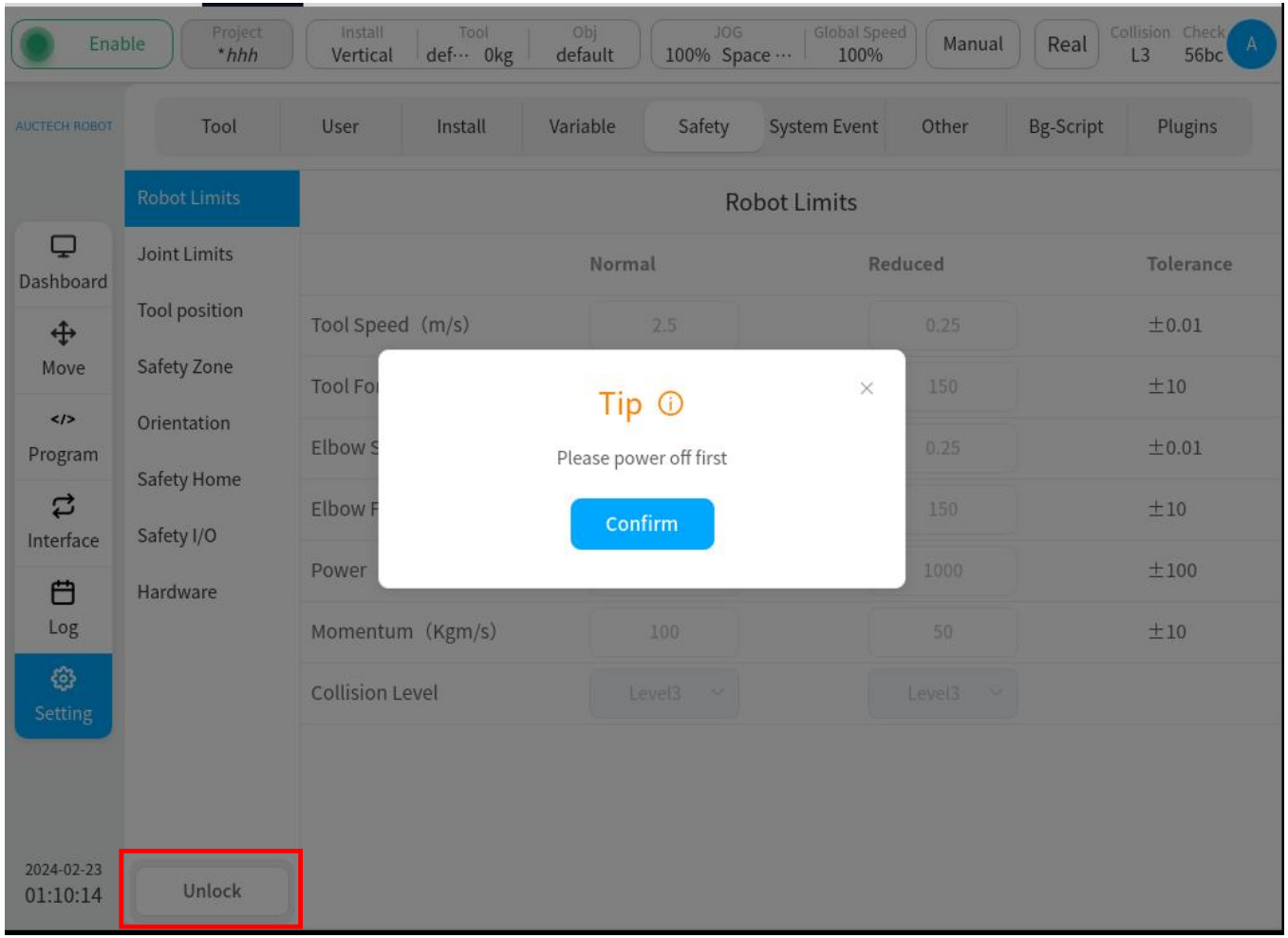
Security configuration parameters can also be viewed in the settings page - security settings.

5.2 Modify the security configuration

Normal Mode: The security mode that is activated by default

Reduced Mode: This mode can be activated using the Safe Input IO

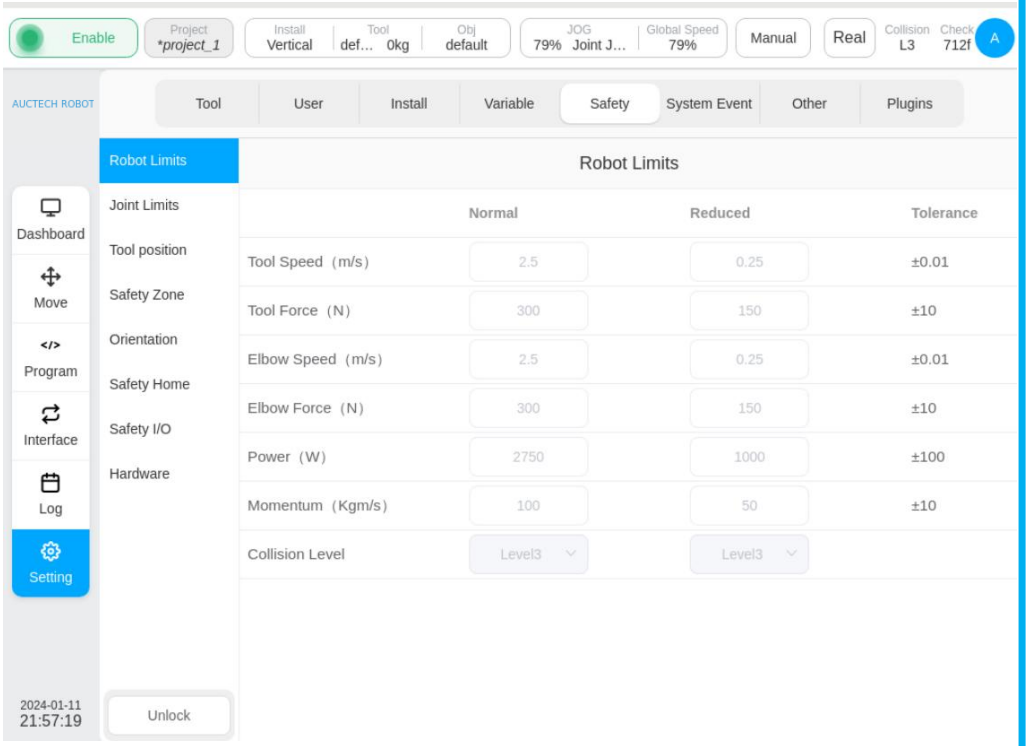
5.2 Modify the security configuration



Before changing the safety configuration, you must use the password to unlock when the power of the robotic arm body is powered off, and the unlock password: 123

5.3 Modify the security configuration –Robot Limits

Robot safety parameters are used to limit general robot movements. It is possible to configure its parameter values in normal and reduced modes.



Maximum speed at the end: Limit the maximum speed at the end of the robot.

Maximum Force at the End: Limit the maximum force exerted at the outermost end of the robot.

Maximum elbow speed: Limit the maximum elbow speed of the robot.

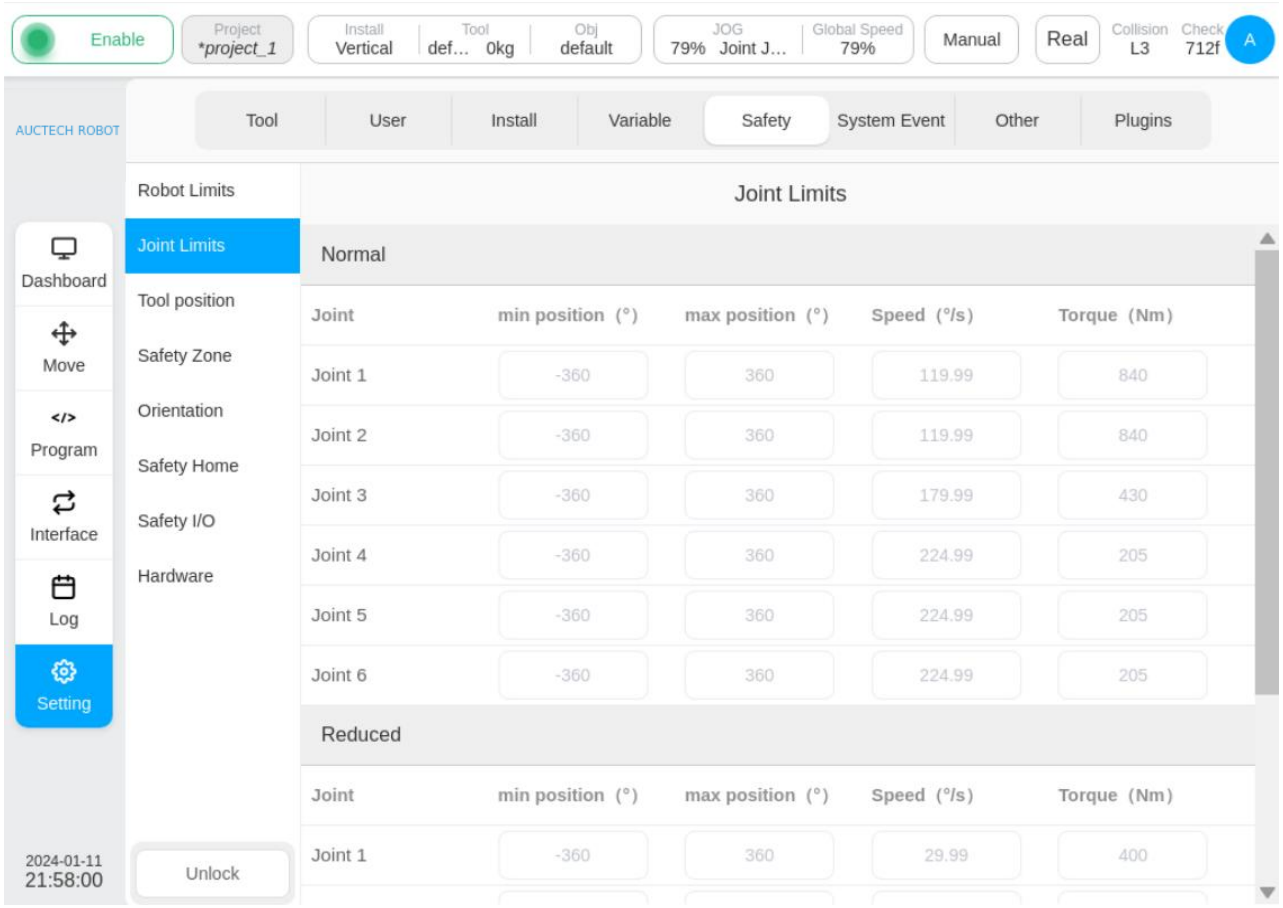
Elbow Maximum Force: Limits the maximum force exerted by the outermost part of the robot's elbow.

Power: Limit the maximum mechanical work that the robot can do to the outside.

Maximum Momentum: Limit the maximum momentum of the robot.

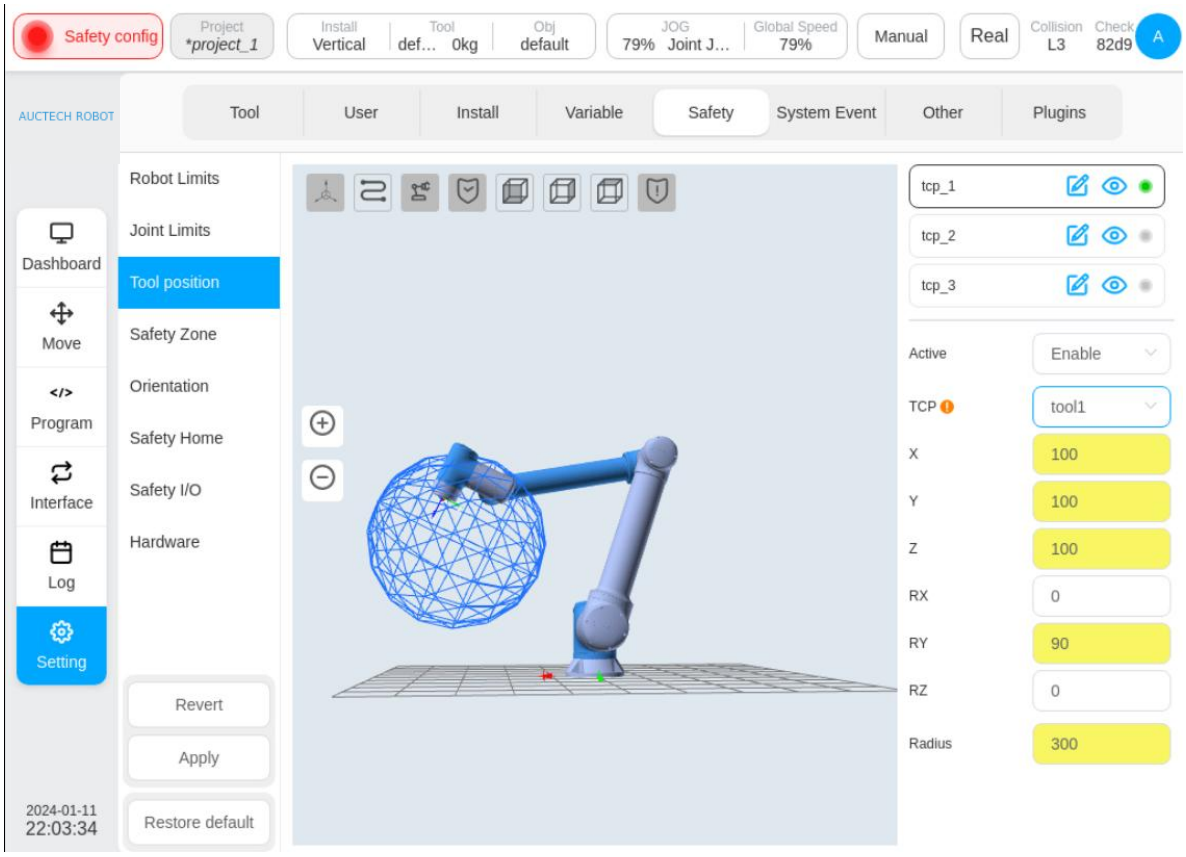
Collision Level: Set the level of the robot collision, the higher the level, the more sensitive.

5.4 Modify the security configuration - Joint Limits



Joint safety parameter limits are used to limit the position range, maximum speed, and maximum torque of each joint of the robot. It is possible to configure the values of the parameters in normal and reduced modes.

5.5 Modify the security configuration-Tool position



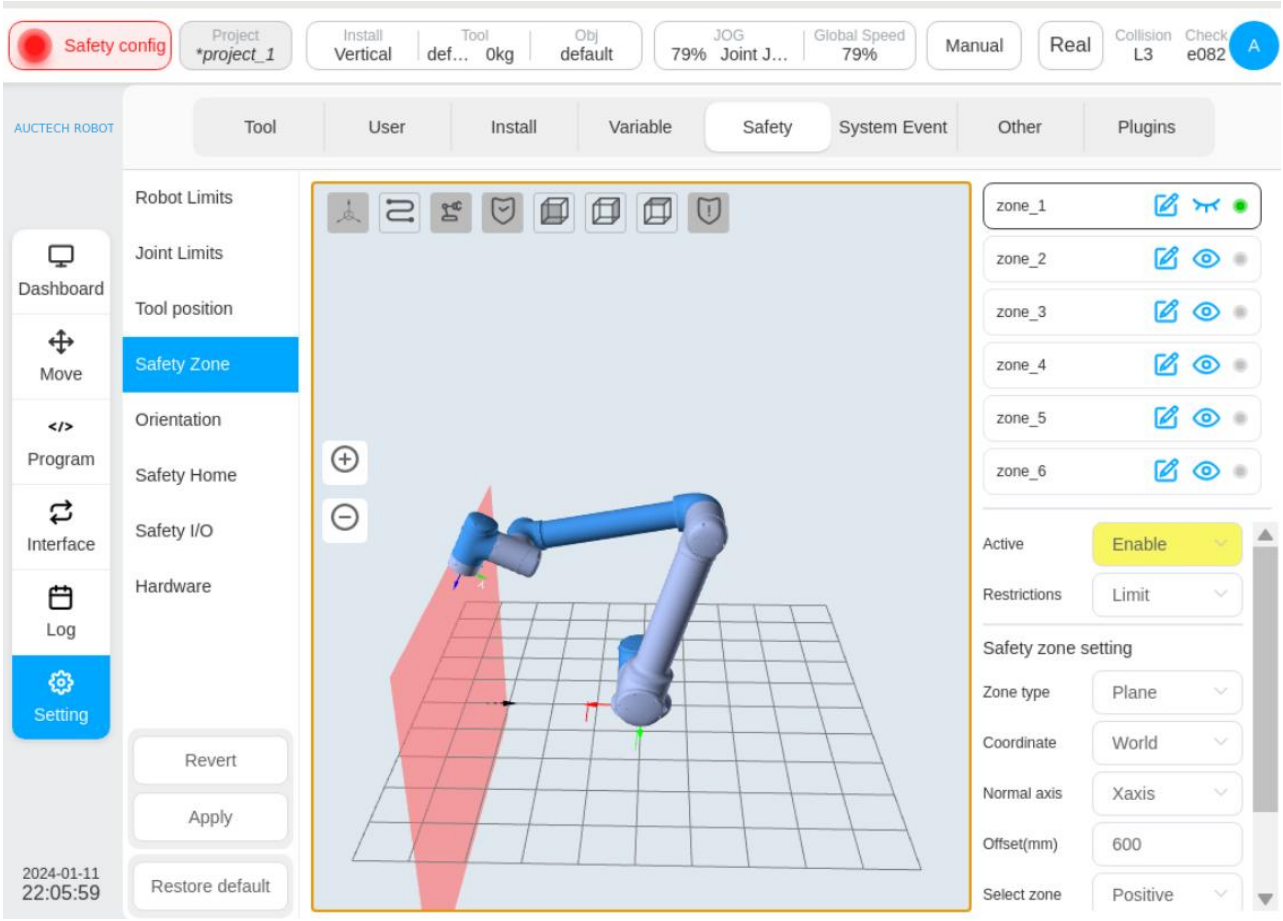
The security system can define three sets of TCP offsets, and after configuration, the robot will use these three sets of TCP to do speed and position monitoring. Any TCP location where the speed exceeds the security setting will trigger a security violation.

Note:

(1) The inactive configuration is indicated in gray

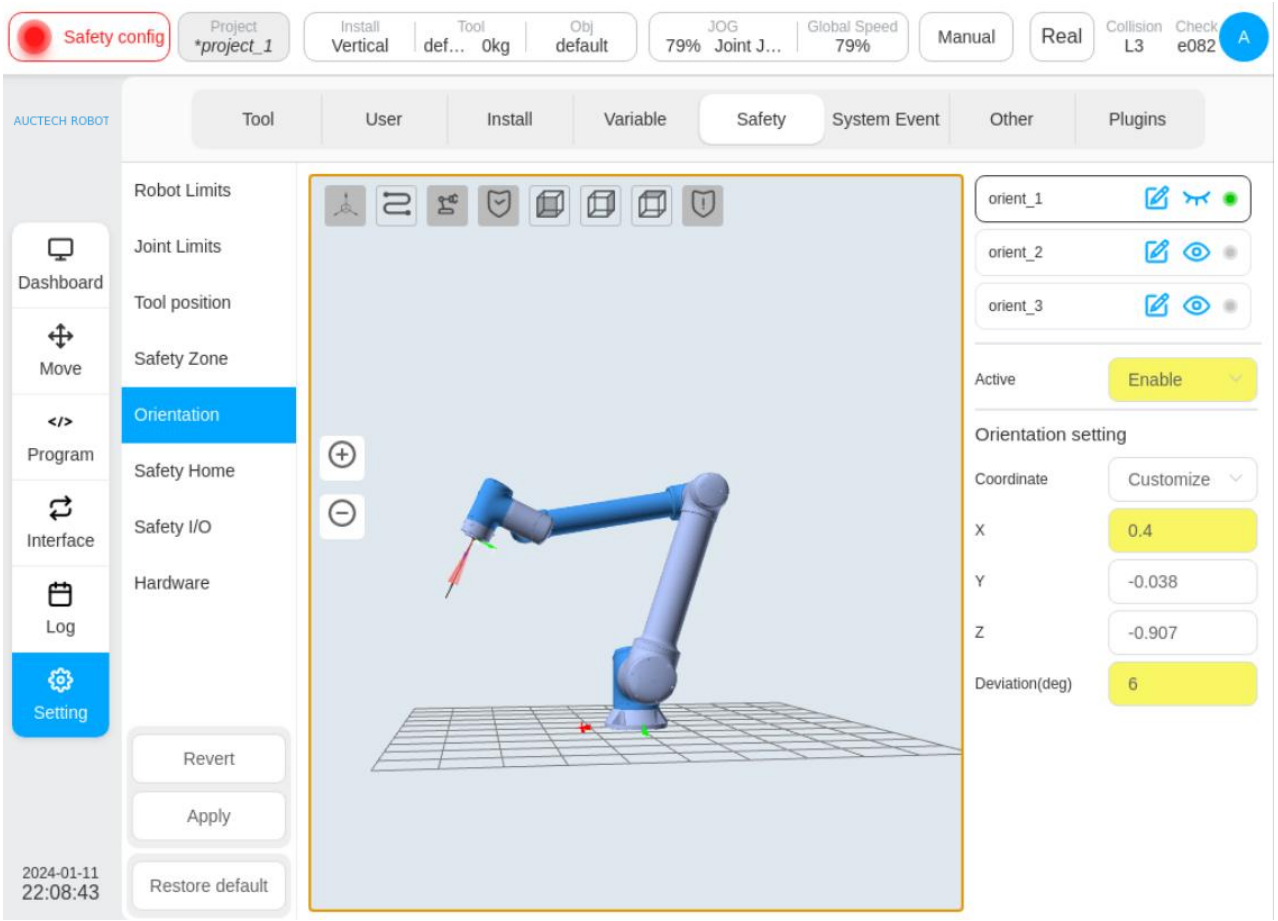
(2) This safety setting should be used in conjunction with the "safe area".

5.6 Modify the security configuration-Safety Zone



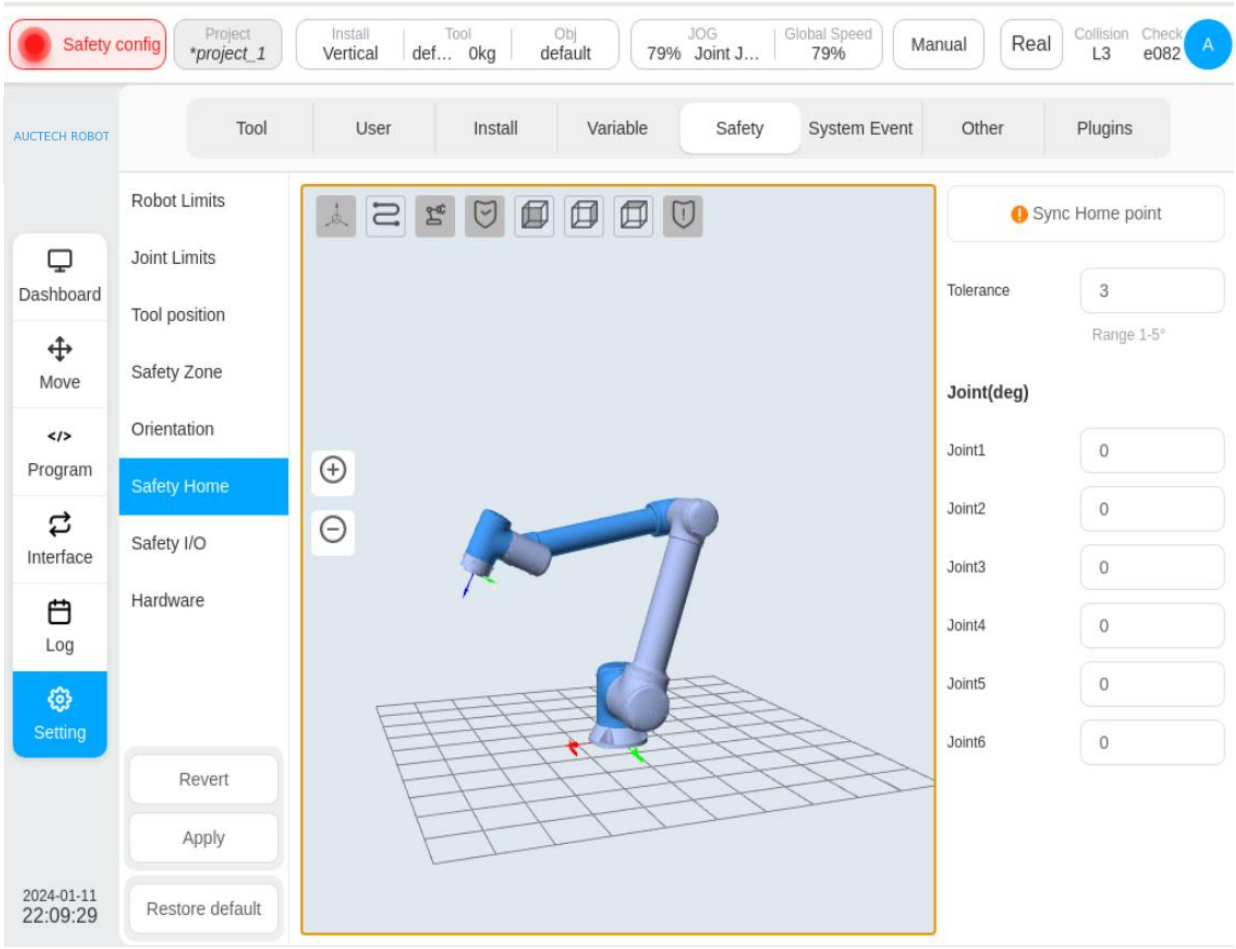
The definition of safety zone includes three forms: planar, rectangular and cylindrical. Up to 6 independent space areas can be set up.

5.7 Modify the security configuration-Orientation



The safe attitude area is set up, and the attitude area refers to the formation of a conical angle around a certain direction vector under the robot base coordinate system. The attitude restriction only restricts the Z-axis of the robot TCP to the range of the attitude region. Triggering a protective stop beyond the area.

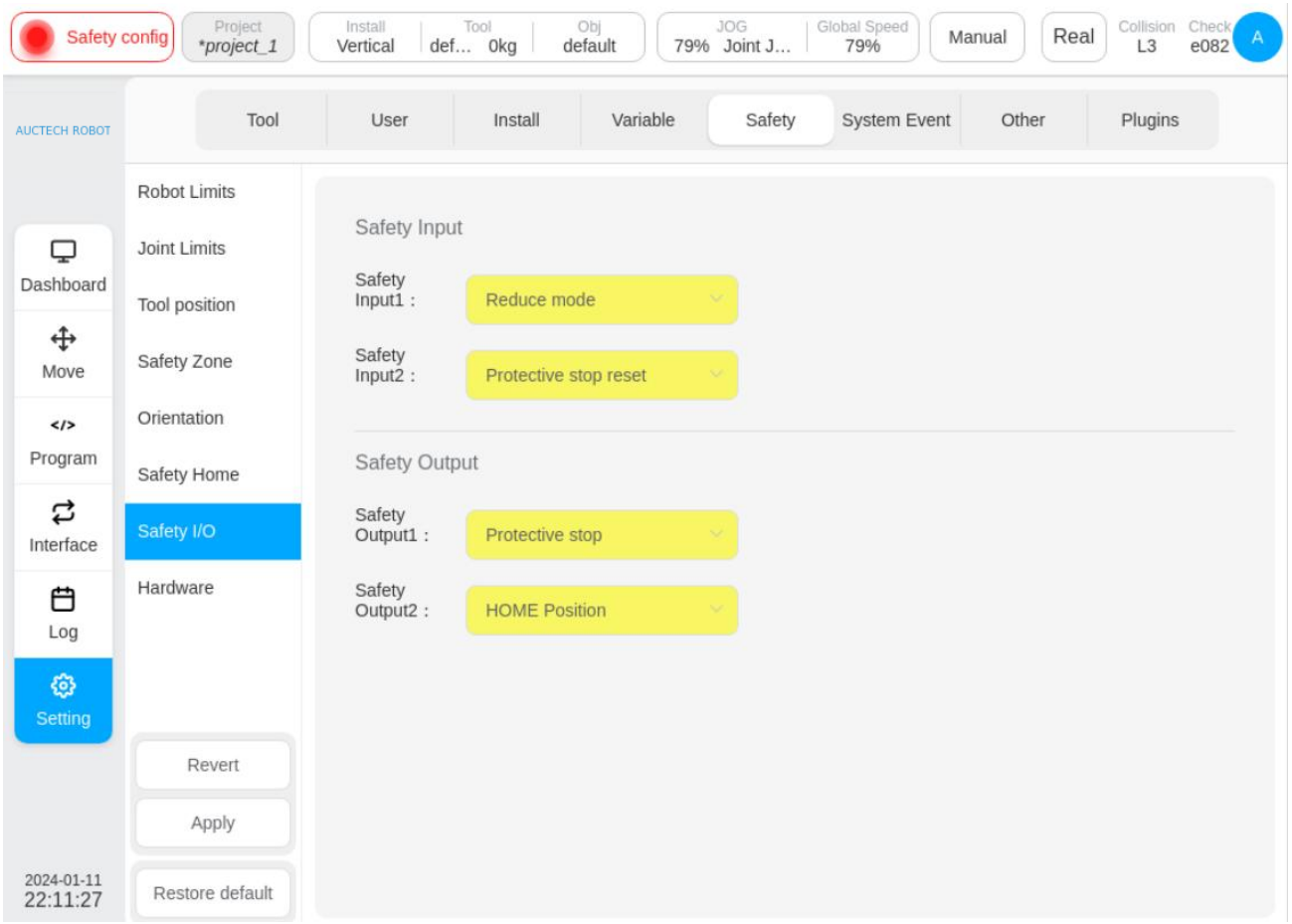
5.8 Modify the security configuration – Safety home



If the location in the security home settings is inconsistent with the home position in the "Other settings", a reminder icon will be displayed in the "Sync home settings".

If the settings are not synchronized, the home location in the system is subject to the security settings. The operation of "press and hold home" back to the home point in the mobile interface, and the output of the home signal of other ports of the robot (such as 2001) are subject to the safety settings.

5.9 Modify the security configuration - Safety I/O



There are two configurable inputs and outputs in the safety I/O.

5.10 Introduction to security features

Security features	description	Trigger an action	Security level
Joint position limitations	Set the upper and lower limits of the allowed joint position, which will be triggered when the current position of the robot exceeds the limit.	Safe stop 1	PLd
Joint speed limits	Set the allowable joint speed limit, which is triggered when the robot joint speed exceeds the limit.	Safe stop 1	PLd
Joint acceleration limitations	Set the allowable joint acceleration limit, which is triggered when the robot joint acceleration exceeds the limit.	Safe stop 1	PLd

5.11 Introduction to security features

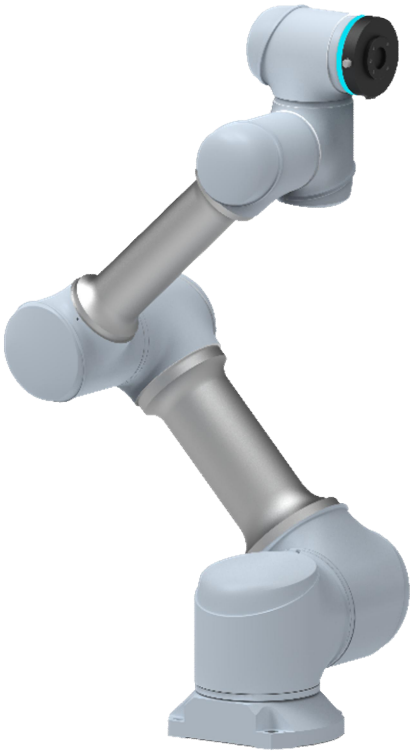
Secure input	description	Trigger an action	Security level
Emergency stop	There are three emergency stop inputs: emergency stop button for control cabinet, emergency stop button for teach pendant, and emergency stop interface reserved for control cabinet. Triggered when any input source is disconnected.	Perform a Type 1 shutdown first, trigger a timer at the same time, and perform a Type 0 shutdown after 500ms.	PLd
Configurable safety input protective stops2	There are two configurable security inputs, which need to be configured on the interface to take effect after being connected. Takes effect only in automatic mode.	Safe stop 2	PLd
Configurable safety input protective stops1		Safe stop 1	PLd
Mode of operation	You can choose to enter manual mode or automatic mode. The speed is 250 mm/s in manual mode, and the safety function is triggered by switching between the operating modes while the robot is moving.	Safe stop 2	PLd
3-position enabling device	When the robot is in manual mode, you need to press the 3rd gear to the middle position to move the robot. The safety function will be triggered when the 3rd gear is enabled at any time during the robot movement and is in a non-intermediate gear. You can choose to turn this feature off.	Safe stop 2	PLd

5.12 Introduction to security features

Secure output	description	Security level
Emergency feedback	When any E-stop is triggered, the path is disconnected.	PLd
Configurable safety outputs	There are two configurable security outputs, which need to be configured on the interface to take effect after being turned on. Takes effect only automatic mode.	PLd

5.13 Downtime category

Stop category	description
0	Immediately cut off the power of the robotic arm
1	Immediately reduce the speed of each joint to 0 with the fastest acceleration, and after the joint is stationary, the brake will be snapped up, and the motor will be enabled
2	While maintaining the trajectory, the robot is slowed down to a standstill, and each joint remains enabled after stationation, and there is no action in holding the brake.



Lecture 6

Script Example

6.1 Example1

Socket communication

```
ret= false
a="123"
b=""
socket_close("aa")
while(ret== false ) do
  ret=socket_open("192.168.233.1",50000,"aa",3000)
  sleep(200)
end
while(1) do
  socket_write(a,"aa",1000)
  sleep(1000)
End
```

6.2 Example2

Posture millimeters, degree units converted into meters, radians

```
a={100,200,300,90,0,45}
b={0,0,0,0,0,0}
function fun(data)
  local a={0,0,0,0,0,0}
  for i=1,3 do
    a[i]=data[i]/1000
  end
  for i=4,6 do
    a[i]=deg2rad(data[i])
  end
  return a
end
b=fun(a)
log(  “b”  ,b)
```

6.3 Example3

Pose meters, radians converted to millimeters, degrees

```
a={0.100,0.200,0.300,1.57,0,0.785}
b={0,0,0,0,0,0}
function fun(data)
  local a={0,0,0,0,0,0}
  for i=1,3 do
    a[i]=data[i]*1000
  end
  for i=4,6 do
    a[i]=rad2deg(data[i])
  end
  return a
end
b=fun(a)
log("b",b)
```

6.4 Example4

Empty list expressions

```
a=1
b=nil ---- equated b{}
if next(b)==nil then
  a=a+1
end
log("a",a)
```

6.5 Example5

Palletizing and depalletizing procedures

```
--movej({0,0,1.57079601,0,-1.57079601,0},90,90)
a={4,3,2}                        -- Layers, rows, columns
b={0.1,0.1,0.1}                  --Workpiece dimensions, length, width and height (in mm)
c={0,0,0}                        --Record Layer Rows and Columns
while(c[1]<a[1]) do              -- Judgment layer
start_dian={0.69200003,0.16400003,0.6600002,-3.14159226,3.2737059996179596e-7,-1.57079637}
            if(c[2]<a[2]) then               -- Judgment line
             if(c[3]<a[3]) then
movej({1.57079601,0,1.57079601,6.442085975777445e-9,-1.57079601,3.947477900112517e-9},90,90)
start_dian[1]=start_dian[1]-b[1]*c[3] --column
start_dian[2]=start_dian[2]-b[2]*c[2] -- row
start_dian[3]=start_dian[3]-b[3]*c[1]
movel(start_dian,1,1)
      c[3]=c[3]+1
else
      c[3]=0
     c[2]=c[2]+1
End
else
     c[2]=0
     c[1]=c[1]+1
end
End
```

6.6 Example6

String data is extracted based on the head and end of frames

```
data="152,,295,26262,aaa,100,200,300,400,500,600,bbb,152,295,26262,aaa"
tou="aaa,"
wei=",bbb,"
qqq=""
a=0
b=0
a=str_find(data,tou)+4
b=str_find(data,wei)
qqq=str_substr(data,a,b-a)
log("qqq",qqq)
```

6.7 Example7

Add parentheses to the string and turn it into an array

```
data="100,200,300,400,500,600"
cc={0,0,0,0,0,0}
data=str_cat("(",data)
data=str_cat(data,")")
cc=str2list(data)
log("cc",cc)
```

Thank you for using Auctech Cobots

AUCTECH

THANKS