

ESTUN CODROID S-SERIES



Robots User Manual

ROBOTS MODEL:

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

CONTROL CABIENT MODEL:

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



NANJING ESTUN CODROID TECHNOLOGY CO. LTD

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

ORIGINAL INSTRUCTIONS

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

Contents

- Chapter 1 Preface 11
 - 1.1 Safety 11
 - 1.2 Nameplate 11
 - 1.3 How to Use This Manual 11
 - 1.4 Copyright and Trademark 12
 - 1.5 Disclaimer of the Manual 12
 - 1.6 Common Terms 12
 - 1.6.1 Robot 12
 - 1.6.2 Maximum workspace 12
 - 1.6.3 Precision 12
 - 1.6.4 Repeatability accuracy 12
 - 1.6.5 Trajectory accuracy 13
 - 1.6.6 Trajectory repetition accuracy 13
 - 1.6.7 Tool Center Point (TCP) 13
 - 1.6.8 Payload 13
 - 1.6.9 Protective stop 13
 - 1.6.10 Singularity (Singular Point) 13
 - 1.7 Revision Record 13
- Chapter 2 Safety Information 14
 - 2.1 Validity and Responsibility 14
 - 2.2 The warning symbols appointed in this manual 14
 - 2.3 Safety Precautions 15
 - 2.4 Safety Requirements 16
 - 2.5 Safety Disclaimer 17
 - 2.6 Limitation of Liability 17
 - 2.7 Stop category 17
 - 2.8 Risk assessment 18
 - 2.9 Safety function 18
 - 2.10 Emergency stop recovery 19
 - 2.11 Forced drive without power 19
 - 2.12 Stopping time and stopping distance 19
 - 2.13 Storage, usage and transportation conditions 20
 - 2.14 Control cabinet and body identification 20
- Chapter 3 Quick Start 23
 - 3.1 Confirmation of packing contents 23
 - 3.2 Robot installation 23
 - 3.2.1 Transportation 24

3.2.2 Handling	24
3.2.3 Installation	27
3.2.4 Operation position layout	29
3.3 Start using	30
3.3.1 Power on and start up	30
3.3.2 Write a program	32
3.3.3 Power off	34
Chapter 4 Mechanical Hardware and Installation	35
4.1 Robot composition	35
4.2 Work Space	35
4.3 Load curve	37
4.4 Flange interface	40
4.5 Installation interface	44
4.6 Robot Specification	46
4.7 Control cabinet	47
4.8 Handle operator	48
Chapter 5 Electrical Hardware and Installation	50
5.1 End Interface	50
5.1.1 Pro terminal interface	50
5.1.2 Meaning of the indicator light strip	50
5.1.3 M8 Interface	50
5.2 Screen information	51
5.3 Control cabinet interface	53
5.3.1 Overview of Electrical Interfaces	53
5.3.2 Safety Interface	54
5.3.3 General Input and Output Overview	55
5.3.4 External power connection method for digital input	56
5.3.5 Internal power connection method for digital input	56
5.3.6 External power connection method for digital output	57
5.3.7 Internal power connection method for digital output	58
5.3.8 Simulation input/output interface	59
5.3.9 CAN/485/IO interface	61
5.3.10 LAN Network Port	62
5.3.11 Communication input	63
Chapter 6 Maintenance and Warranty	64
6.1 Notes	64
6.2 Daily inspection items	64
6.2.1 General cleaning	64
6.2.2 Control box	65

6.2.3 Robot	66
6.3 System update	66
6.3.1 Update steps	66
6.4 Common Mistakes	68
6.4.1 Singularity/Inverse solution failure	68
6.4.2 Trigger collision detection	69
6.4.3 Location/Speed Exceedance	70
6.4.4 Joint tracking error is too large	70
6.4.5 Alarm cleared	70
6.5 Fault code description	71
6.6 Disclaimer	71
6.7 Abandoned robots	72
Chapter 7 Overview of the Teaching Pendant Interface	73
7.1 Login interface	73
7.2 Home page	73
7.2.1 Switch tab area	74
7.2.2 Account Settings Button	74
7.2.3 Error message and real-time log window button	74
7.2.4 Full-screen display button	75
7.3 Project Tab	75
7.3.1 Quick operation area	76
7.3.2 Graphics Programming Area	77
7.3.3 Pose Zone	83
7.3.4 Parameter Area	85
7.3.5 3D Simulation	86
7.3.6 Register	88
7.3.7 I/O	89
7.3.8 Variable Management	89
7.3.9 Project Management Area	91
7.3.10 Speed ratio adjustment area	93
7.4 Settings tab	93
7.4.1 Basic	93
7.4.2 Tools, load, coordinate system	95
7.4.3 Others	96
7.4.4 Sports	98
7.4.5 Register communication	100
7.4.6 IO	101
7.4.7 MODBUS Master	102
7.4.8 Panel IO	104

7.5 Log tab	104
7.6 Management tab	105
Chapter 8 Introduction to Variables	107
8.1 Variable Overview	107
8.2 Variable	108
8.2.1 POSE	108
8.2.2 Basic Data Types	109
8.2.3 SPEED	109
8.2.4 ACC	110
8.2.5 ZONE	110
8.2.6 CLOCK	110
8.2.7 Socket	110
8.2.8 INTERRUPT	111
8.2.9 LsScale	111
8.2.10 LsThresh	111
8.2.11 VibrationSuppression	111
8.2.12 Matrix2	112
8.2.13 Matrix3	112
8.2.14 Matrix4	112
8.2.15 Matrix9	112
Chapter 9 Calibration	112
9.1 Joint coordinate system	112
9.2 World coordinate system	113
9.3 Coordinate System and Calibration	114
9.3.1 Three-point calibration method	115
9.3.2 Use the user coordinate system	117
9.4 Tools and Calibration	118
9.4.1 Four-direction calibration method	119
9.4.2 One-point calibration method (attitude)	121
9.4.3 Use the tool coordinate system	123
Chapter 10 Instruction Introduction	125
10.1 Displacement Instructions	125
10.1.1 MovJ	125
10.1.2 MovL	125
10.1.3 MovC	126
10.1.4 MovCircle	127
10.1.5 MovJRel	128
10.1.6 MovLRel	129
10.1.7 MovLSearch	130

10.1.8 AddDo 131

10.1.9 MovTraj..... 132

10.2 Logical Instructions 132

10.2.1 GoTo 132

10.2.2 If132

10.2.3 Elself 133

10.2.4 Otherwise 134

10.2.5 While 134

10.2.6... =..... 135

10.2.7 RETURN 135

10.2.8 CALL 136

10.2.9 RUN 136

10.2.10 KILL 136

10.2.11 Labeling 136

10.3 Flow Control Instructions..... 137

10.3.1 Wait 137

10.3.2 WaitFinish 137

10.3.3 WaitCondition 137

10.4 IO Instructions 138

10.4.1 SetDO 138

10.4.2 SetAO 139

10.4.3 WaitDI 139

10.4.4 WaitDI8421 140

10.4.5 WaitAI 141

10.4.6 GetDI8421 141

10.4.7 GetDO8421 142

10.4.8 SetDO8421 142

10.4.9 GetDO 143

10.4.10 GetDI 143

10.4.11 GetAO 143

10.4.12 GetAI 144

10.5 Set instructions 144

10.5.1 SetTool 144

10.5.2 SetCoord 145

10.5.3 SetPayload 145

10.5.4 Stop 145

10.5.5 EnaVibraSuppr 145

10.5.6 DisVibraSuppr 145

10.5.7 ClsDectLevel 145

10.6 Position Operation Instructions 146

 10.6.1 GetCurAPos 146

 10.6.2 GetCurCPos 146

 10.6.3 APosToCPos 146

 10.6.4 CPosToAPos 147

 10.6.5 CPosToCPos 148

 10.6.6 ToolOffset 148

 10.6.7 UserOffset 149

 10.6.8 CposOffset 149

 10.6.9 GetAxis 150

 10.6.10 GetCartesian 150

 10.6.11 Position Inverse 151

 10.6.12 PointsDistance 151

 10.6.13 InterpolationCpos 152

 10.6.14 TransformPlane 152

 10.6.15 GetTrajStartPoint 153

 10.6.16 GetTrajEndPoint 154

10.7 Bitwise Operation Instructions 154

 10.7.1 BitAnd 154

 10.7.2 BitNeg 154

 10.7.3 BitOr 155

 10.7.4 BitLSH 155

 10.7.5 BitRSH 156

10.8 Clock Instruction 156

 10.8.1 CLKStart 156

 10.8.2 CLKStop 156

 10.8.3 CLKReset 157

10.9 Socket Command 157

 10.9.1 SocketCreate 157

 10.9.2 SocketClose 158

 10.9.3 SocketSendStr 158

 10.9.4 SocketSendReal 158

 10.9.5 SocketSendInt 159

 10.9.6 SocketReadReal 160

 10.9.7 SocketReadInt 161

 10.9.8 SocketReadStr 161

10.10 Interrupt Instruction 162

 10.10.1 IConnect 162

 10.10.2 IDelete 162

10.10.3 ITimer 163

10.10.4 ICondition 163

10.11 Modbus Commands.....164

10.11.1 GetModConState164

10.11.2 ReadSingleCoilReg164

10.11.3 ReadDiscreteInputReg 165

10.11.4 ReadSingleHoldReg166

10.11.5 ReadInputReg166

10.11.6 WriteSingleCoilReg167

10.11.7 WriteSingleHoldReg 168

10.12 Array Instructions..... 168

10.12.1 SetMatrix2168

10.12.2 SetMatrix3169

10.12.3 SetMatrix4170

10.12.4 SetMatrix9171

10.12.5 GetMatrix2173

10.12.6 GetMatrix3173

10.12.7 GetMatrix4174

10.12.8 GetMatrix9175

10.13 String instructions..... 175

10.13.1 APosToStr175

10.13.2 CPosToStr176

10.13.3 DAPosToStr177

10.13.4 DCPosToStr177

10.13.5 TranStrToIntArray178

10.13.6 TranStrToRealArray179

10.13.7 TranStrToApos179

10.13.8 TranStrToCpos180

10.13.9 TranStrToDApos181

10.13.10 TranStrToDCpos182

10.13.11 IntArrayToString183

10.13.12 RealArrayToString183

10.13.13 BoolArrayToString184

10.14 RS485 Instructions..... 184

10.14.1 RS485Init.....184

10.14.2 RS485Read185

10.14.3 RS485Write186

10.14.4 RS485FlushReadBuffer186

10.15 Mathematical operation functions..... 186

10.15.1 sin	187
10.15.2 cos	187
10.15.3 tan	187
10.15.4 asin	187
10.15.5 acos	187
10.15.6 atan	187
10.15.7 atan2	188
10.15.8 sinh	188
10.15.9 cosh	188
10.15.10 tanh	188
10.15.11 log	188
10.15.12 log10	188
10.15.13 sqrt	189
10.15.14 exp	189
10.15.15 pow	189
10.15.16 deg	189
10.15.17 rad	189
10.15.18 fmod	189
10.15.19 floor	190
10.15.20 random	190
10.16 String Functions	190
10.16.1 byte	190
10.16.2 char	190
10.16.3 find2	190
10.16.4 findEnd	190
10.16.5 format	191
10.16.6 getAt	191
10.16.7 gsub	192
10.16.8 len	192
10.16.9 left	192
10.16.10 lower	192
10.16.11 right	192
10.16.12 reverse	193
10.16.13 strcmp	193
10.16.14 trimLeft	193
10.16.15 trimRight	193
10.16.16 upper	193
10.16.17 IToStr	193
10.16.18 RToStr	194

10.16.19 StrTol194

10.16.20 StrToR 194

10.16.21 Append.....194

Appendix to Chapter 1 195

11.1 Error Codes195

11.2 User Levels and Permissions 199

11.3 Declaration201

Chapter 12 Spare Parts List 203

Chapter 13 Contact Information204

Chapter 1 Preface

1.1 Safety

Thank you for purchasing and using our company's robot, it only passed two functional safety certifications of EN ISO 13840-1:2023:

- 1. the performance level of the emergency stop circuit for the above model of robot is PL d.
- 2. the nennrmance level of the protective stop circuit for the above model of robot is maximum PL d.

1.2 Nameplate

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

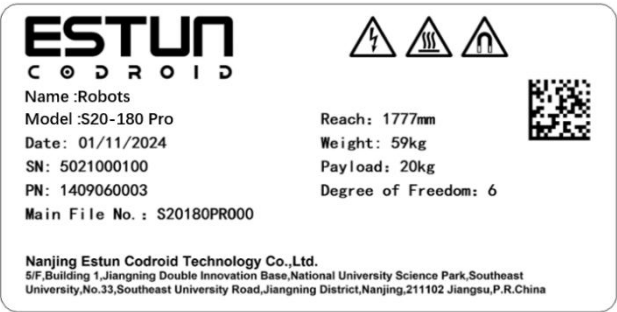


Figure 1-1 Robot Arm Nameplate

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

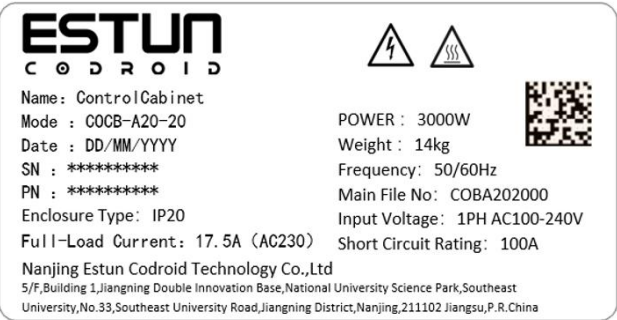


Figure 1-2 Control Cabinet Nameplate

1.3 How to Use This Manual

This manual describes the hardware composition of the Codroid robot and the operation of its teaching control system, which is helpful for users to understand and master the functions, technical specifications, installation and usage of the Codroid robot.

This manual is applicable for customers, sales engineers, installation and commissioning engineers,

technical support personnel, etc.

This manual contains methods on how to protect users and prevent machine damage. Users need to read all relevant descriptions in the manual and be fully familiar with the safety precautions.

In this manual, we have tried to describe various situations. However, due to the vast number of possibilities, it is impossible to record all the situations that should not be done or cannot be done.

1.4 Copyright and Trademark

Estun CoDroid, CoDroid, CoDroid EIP, CoBrain, CoDrive, CoSense, CoSafe, CoTool are registered trademarks of Estun CoDroid. All rights reserved @ Nanjing Estun CoDroid Technology Co., Ltd.

Without the written permission of the Company, no unit or individual may excerpt, copy in part or in whole, or disseminate the content of this document in any form.

1.5 Disclaimer of the Manual

Before using this product, please read this user manual and the relevant technical documents published online in detail and understand the information. Make sure to use the robot only after fully understanding the robot and related knowledge. We recommend that you use this manual under the guidance of professionals. All safety information contained in this manual shall not be regarded as a guarantee by Codroid. Even if you follow this manual and related instructions, harm or loss may still occur during use.

1.6 Common Terms

1.6.1 Robot

Fixed or mobile automatic machinery that can be automatically controlled, repeatedly programmed, multi-purpose, and programmed for three or more axes, used in industrial automation.

1.6.2 Maximum workspace

The space that the robot's moving parts can sweep through, plus the space that can be swept by the end effector and the workpiece during their movements.

1.6.3 Precision

The deviation of position and attitude between the average value of the commanded distance and the actual distance.

1.6.4 Repeatability accuracy

The consistency of the actual distance reached after repeating the movement in the same direction for the same instruction distance n times.

1.6.5 Trajectory accuracy

The maximum trajectory deviation along the obtained trajectory in terms of position and orientation.

1.6.6 Trajectory repetition accuracy

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

1.6.7 Tool Center Point (TCP)

Points set for a certain purpose with reference to the mechanical interface coordinate system. (Ref. GB/T 12643-2013, Definition 4.9)

1.6.8 Payload

It refers to all the loads attached to the robot flange excluding the weight of the tool.

1.6.9 Protective stop

A form of operation interruption that allows for the orderly termination of a process for safety reasons while maintaining the program logic to enable a restart.

1.6.10 Singularity (Singular Point)

The situation where two or more axes of a robot are collinear, resulting in uncertainty in the robot's motion and speed.

1.7 Revision Record

Material Number	Version	Release Data	Description
1210002200	V1.0	20250401	Initial version

Chapter 2 Safety Information

2.1 Validity and Responsibility



The information in this manual does not cover the design, installation and operation of a complete robot application, nor does it include all the peripheral equipment that may affect the safety of this complete system. The design and installation of the complete system must comply with the safety requirements established in the standards and regulations of the country where the robot is installed.

Estun Codroid integrators are responsible for ensuring compliance with applicable laws and regulations of the relevant countries and ensuring that there are no significant hazards in the complete robot application. This includes but is not limited to the following:

- Conduct a risk assessment of the entire robot system.
- Connect other mechanical and additional safety devices defined in the risk assessment together.
- Make appropriate security settings in the software.
- Ensure that users cannot modify any security measures.
- Confirm that the design and installation of the entire robot system are accurate and error-free.
- Good understanding on this instruction
- Mark the relevant logos and contact information of the integrator on the robot.
- Collect all the documents in the technical files; including the risk assessment and this manual.

2.2 The warning symbols appointed in this manual

The following safety warning signs may appear in this manual. Their meanings are as follows:

	<p>Warning</p> <p>This sign indicates a potentially dangerous electrical situation. If not avoided, it may cause death or serious injury to personnel or severe damage to equipment.</p>
	<p>Warning</p> <p>This sign indicates a potentially dangerous situation. If not avoided, it may cause death or serious injury to people.</p>

**Warning**

This sign indicates a potentially dangerous electrical situation. If not avoided, it may cause personal injury or severe damage to equipment.

**Warning**

This sign indicates a potentially dangerous situation. If not avoided, it may cause personal injury or serious damage to equipment.

**Warning**

This sign indicates a potentially dangerous electrical condition. If not avoided, it may cause personal injury or serious damage to equipment.

**Warning**

This sign indicates a potentially dangerous hot surface. Contact with it may cause personal injury.

**Warning**

This symbol indicates a situation that, if not avoided, can lead to serious damage.

2.3 Safety Precautions

- Make sure that the robot arm and the tool/end effector are both correctly and securely fastened in place with bolts. Ensure that the robot arm has sufficient space to move freely.
- Ensure that the safety measures and/or robot safety configuration parameters as defined in the risk assessment have been established to protect programmers, operators and bystanders.
- When operating the robot, please do not wear loose clothing or jewelry. Make sure long hair is tied back when operating the robot.
- If the robot is damaged, do not use it, for example, when the joint cap is loose, damaged or removed.
- Never put your fingers into the control box.
- Do not connect any safety devices to the standard IO interface. Only the safety IO interface can be used.
- Ensure correct installation settings (such as the installation angle of the robot, the weight in TCP, TCP offset, and safety configuration).
- Only after a risk assessment is conducted can the drag-and-drop teaching function be used during the installation process.

- The tools/end effectors and obstacles must not have sharp corners.
- Ensure that people are warned to keep their heads and faces out of the reach of robots that are in operation or about to start operating
- When using the teaching pendant, pay attention to the movement of the robot.
- If a risk assessment has been determined, do not enter the safe confines of the robot or touch the robot while the system is in operation.
- Connecting different machines may increase the risk of danger or cause new hazards. Always conduct a comprehensive risk assessment of the entire installation.
- Do not modify the robot. Any modification to the robot may cause unpredictable dangers. Any authorized reconfiguration of the robot must be carried out in accordance with the latest version of all relevant service manuals.
- Ensure that robot users are aware of the location of the emergency stop button and are instructed to activate it in case of an emergency or abnormal situation.
- The robot and the control box will generate heat during operation. Do not touch the robot when it is running or has just stopped. You can cool down the robot by turning it off and waiting for one hour.
- When the robot is connected to or works together with machinery that could cause damage to the robot, it is strongly recommended to test all the functions of the robot and the robot program separately.
- Do not expose the robot to magnetic fields, fire, explosive hazards, radio interference, liquids, etc. for an extended period of time, as this may damage the robot.
- The robot system is not permitted to be used in explosive or potentially explosive environments.
- When the equipment is in operation, even if the mechanical arm appears to have stopped while waiting for a start signal, it should still be regarded as in motion. Please do not approach the mechanical arm.
- During the processes of transporting, installing, operating and maintaining robots, operators must wear safety gloves, glasses, anti-crush shoes and other safety protective equipment to avoid dangerous injuries.

2.4 Safety Requirements

The safety functions generally comply with the ISO 10218-1 standard, and specifically meet the following requirements.

When safety-related control systems are required, the design of safety-related components should be such that:

- The failure of any single component will not result in the loss of safety functions.
- Where practicable, a single fault shall be detected before or at the next demand on the safety function.

- When a single fault occurs, the safety function should always be in operation and maintain a safe state until the detected fault is repaired.
- All reasonably foreseeable faults should be detected.

This requirement is regarded as equivalent to the Category 3 structure as described in ISO 13849-1. Category 3 is typically achieved through redundant circuits. The safety function and the robot controller comply with Performance Level (PL) d as stipulated in ISO 13849-1.

2.5 Safety Disclaimer

This manual does not provide comprehensive information on the design, installation and operation of the robot in conjunction with other equipment, nor does it cover the possibility of the impact of such use on surrounding equipment.

The safety of a robot installation depends on how the robot is integrated, and the integrator needs to conduct a risk assessment of the design and installation of the system in compliance with the laws and regulations of the country where it is installed, as well as safety codes and standards.

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

- ISO 12100:2010 Safety of machinery - General principles of design - Risk assessment and risk reduction;
- ISO 10218-2:2011 Robots and robotic devices - Safety requirements - Part 2: Industrial robot systems and integration
- RIA TR R15306-2014 Technical Report on Industrial Robots and Robot Systems - Safety Requirements, Task-based Risk Assessment Method
- ANSI B11.0-2010 Machinery Safety; General Requirements and Risk Assessment.

2.6 Limitation of Liability

Any safety information contained in this manual should not be regarded as a guarantee for our company's robots. Many matters cannot be described in detail, and there is still a possibility of causing injury or damage.

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

2.7 Stop category

Type 0	Uncontrolled stop, which stops the robot by immediately disconnecting power to the actuator.
Type 1	Controlled stop, where the actuator actively brakes but does not

	ensure that the robot stops on its trajectory. After the robot has stopped, the power supply is cut off.
Type 2	Controlled stop, where the actuator actively brakes and ensures that the robot stops in its trajectory. The robot stops without disconnecting the power supply.

In accordance with the IEC 60204-1 standard, Codroid robots are equipped with three stop categories, namely Stop Category 0 (Cat.0), Stop Category 1 (Cat.1), and Stop Category 2 (Cat.2). Among them, Stop Category 0 is an uncontrolled stop, while Stop Categories 1 and 2 are controllable stops.

According to IEC 60204-1 and ISO 13850, emergency equipment is not a safety guard device. They are supplementary protective measures and are not used to prevent injuries.

In case of an emergency, press the emergency stop button to immediately halt all movements of the robot and lock it in place. The emergency stop function should not be used as a risk reduction measure but can be regarded as a secondary protection device for use only in critical situations.

Under normal circumstances, if it is necessary to stop the robot's movement, please use other methods. After a risk assessment, if an emergency stop button needs to be installed, it must comply with the requirements of IEC-60947-5-5.

When the emergency stop button is pressed, the robot system will cut off the power supply to the robot, and the brake devices between the robot's joints will automatically lock the joints. However, due to the effect of gravity, slight movement of the robot body is a normal phenomenon, but this may also pose a risk of pinching or colliding with the human body.

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

Emergency stop and protective stop functions are implemented through the safety interface. For details, please refer to Section 5.3.2.

2.8 Risk assessment

Before installing or using this product, users must conduct necessary risk assessments based on the usage conditions and carefully read the residual risks that may exist in the company's stated values. For relevant content, please refer to the corresponding software and hardware version manual.

2.9 Safety function

The safety functions of the CoDroid robot are shown in the following table.

Safety function	Description
Emergency stop	When the emergency stop button is pressed, stop category 1 is activated.
Protective stop	When the relevant signal input is low, activate stop category 2. this

	function needs to be manually reset.
Safety rated deceleration control	When the correlation signal input is low, it will reduce the TCP speed to the limit.
Joint position limitation	Sets the limit range of allowable joint positions.
Joint speed limitation	Sets the limit range for the allowable joint speed.
Joint torque limitation	Sets the limit range of allowable joint torque.
TCP location Limitation	Sets the limit range of allowed TCP locations.
TCP speed Limitation	Sets the maximum TCP rate.
TCP torque limitation	Set the maximum torque of TCP.
Robot Power Limit	Limit the maximum power of the robot.
TCP Directional Limit	Sets the direction limits allowed by the tool.
Security-grade monitoring downtime	When the relevant signal input is low, activates Stop Category 2. This function can be reset when the relevant signal input signal is low.
Speed and distance monitoring	Maintain a minimum protective distance between the operator and the robot. The robot system stops when the separation distance decreases below the protective distance. The robot can automatically resume motion when the operator leaves the robot system.
Power Torque Limit	Limit the maximum power and torque of the robot.

2.10 Emergency stop recovery

When the emergency stop button is pressed, it will be locked. To unlock it, rotate the button as indicated on it. Only after unlocking can the alarm be cleared through the control software, and then power on and enable to restore from the emergency state.

2.11 Forced drive without power

In case of an emergency, if it is necessary to move the robot's joints but it is impossible or unnecessary to power on the robot, manual forced drive without power can be used.

To perform a no-power forced drive, you must push or pull the robot arm forcefully to move the joints. Each joint brake has a brake that allows the joint to move under high-torque conditions.

Manual movement without power drive is only for emergency situations and will affect the service life of the brake device.

2.12 Stopping time and stopping distance

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

- Class 0
- Class 1
- Class 2

The test for joint 0 is conducted through horizontal movement, that is, the rotation axis is perpendicular to the ground.

During the tests at joint 1 and joint 2, the robot followed a vertical trajectory with the rotation axis parallel to the ground and performed a stop operation when the robot moved downward.

The robot arm is fully extended horizontally.

The general speed of the robot is set at 100%, and it moves at the maximum speed of the joints.

The maximum effective payload that the robot can handle.

The following table shows the measured stopping distance and stopping time of the 3kg robot when it triggers a Class 1 stop under the above conditions. For test data of other models, please consult our technical staff.

The following data is for reference only. Depending on the application scenarios and usage conditions of the robot, the results of the stopping distance and stopping time may vary.

Position	Stopping distance (rad)	Stopping time (ms)
Joint 1 (base)	0.30	282
Joint 2 (shoulder)	0.29	287
Joint 3 (elbow)	0.29	237






2.13 Storage, usage and transportation conditions

- The ambient temperature during storage and operation should be between 0 and 40°C.
- Places with less humidity and drier. Relative humidity of 10%-90% without condensation;
- Places with little dust, powder, grease fumes and water.
- No flammable materials, corrosive liquids or gases are allowed in the work area.
- For places where the vibration or shock energy on the electrical control cabinet is small (vibration below 0.5G);
- There should be no major electrical noise sources nearby (such as gas shielded welding TIG equipment, etc.);
- There is no potential risk of collision with mobile devices such as AGVs.
- The control box should be installed outside the robot's operating range (beyond the safety fence).
- The control box should be at least 200mm away from the wall to keep the heat dissipation channel unobstructed.

2.14 Control cabinet and body identification

The following signs and nameplates are attached to locations where specific dangers may occur. To prevent accidents, please strictly follow the instructions and contents of

the signs when operating. Do not tear, damage or remove the signs at will. Be especially careful when handling the components or units to which the signs and nameplates are attached and the surrounding areas.

A		The equipment must be operated and maintained by specialized personnel with personal protection. Make sure to follow the hardware setup instructions. Avoid using the product incorrectly and causing damage to the machine or other equipment, or injury to personnel.
B		Do not open the control cabinet and body to touch the internal electronics and circuitry to avoid electric shock. There is a risk of fire or electric shock. Always use appropriate personal protective equipment to protect against the risk of arc flash, failure to follow this code may result in personal injury or death.
C		Hot surfaces that can be hazardous and can cause injury if contact occurs.
D		The robot body has a magnetic field inside, which may be harmful to the body and electronic equipment.
E		Product nameplate to confirm basic product information

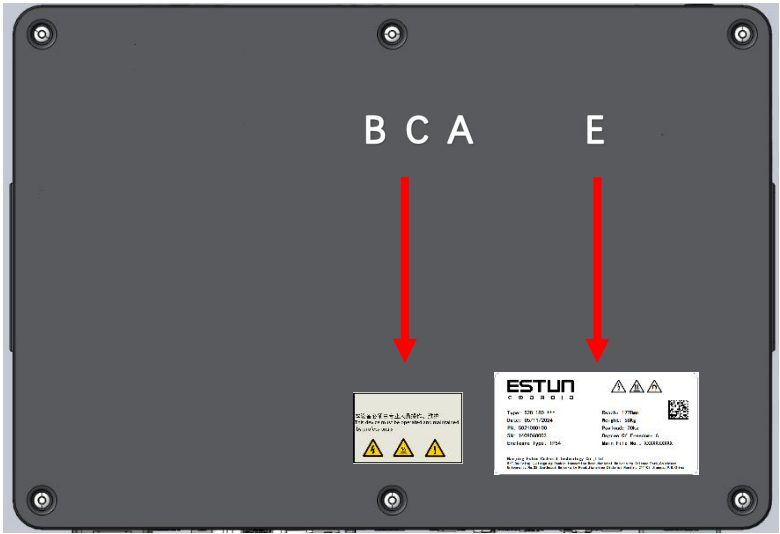


Figure 2-1 Control Cabinet Marking, Nameplate Position

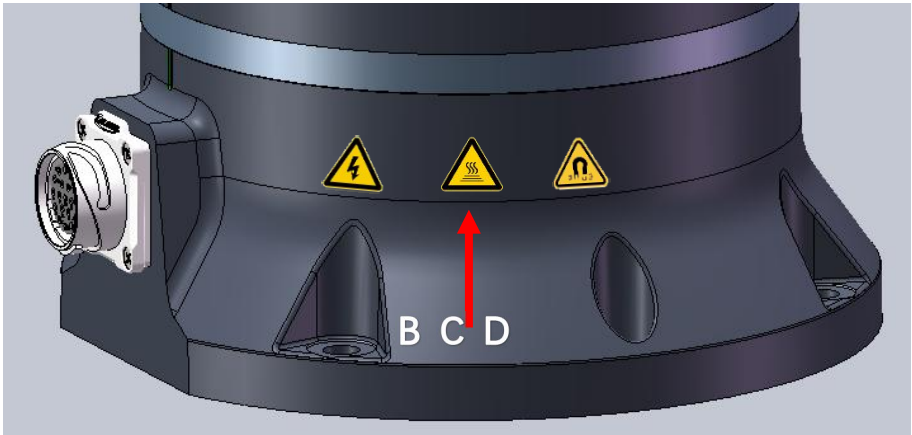


Figure 2-2 Labeling and Nameplate Position for Bodies Weighing 10kg or Less

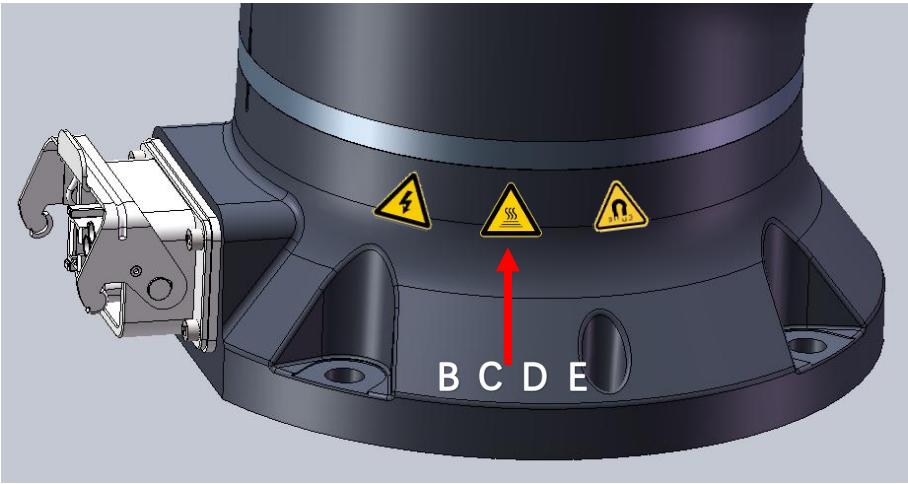


Figure 2-3 20kg Body Marking, Nameplate Position

Chapter 3 Quick Start

3.1 Confirmation of packing contents



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

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



Figure 3-1 Contents of the Carton

3.2 Robot installation

3.2.1 Transportation

Keep the original packaging intact during transportation. Store the packaging materials in a dry place; they may be needed for repacking and moving the robot later. Move the robot from the packaging materials to the installation location:

When installing the robot arms of S3-60, S5-90 and S10-140, the two die-cast connecting rods of the robot arm can be lifted simultaneously. Hold the robot until all the installation bolts of the robot base are tightened. Please refer to 3.2.2 for handling.

When handling the S20-180 robot arm, please refer to 3.2.2 Handling.



WARNING
When moving and handling the equipment, the operator should wear safety gloves, glasses, anti-smash shoes and other safety protection equipment to avoid dangerous injuries during the moving and handling process.



WARNING

- Ensure that no excessive weight is placed on the back or other body parts when lifting the equipment. Use appropriate lifting equipment.

Follow all regional and national lifting guidelines. Universal Robots is not responsible for any damage caused by transportation of the equipment.

- Ensure that the robot is mounted according to the mechanical interface as described in 3.2.3 Mounting and 4.5 Mounting Interface of these instructions.
- If the robot needs to be precisely positioned, it can be positioned by pins through the two pre-drilled holes.



WARNING

- Ensure that the robot is mounted correctly and in a position that avoids vibration.
- Turn off the power to the robot arm during mounting and dismounting to prevent accidents.

Turn off the power:

- Return to the packing position during disassembly.
- Turn off the robot by pressing the power button on the actuator.
- Disconnect the power plug.

3.2.2 Handling



WARNING
When moving and handling the equipment, the operator should wear safety gloves, glasses, anti-smash shoes and other safety protection equipment to avoid dangerous injuries during the moving and handling process.

3.2.2.1 Manner of handling robots weighing 10kg or less

a) Transportation and Unpacking



Figure 3-2 Transportation and Unpacking Diagram for 10kg and blow

b) Install the lifting straps and use the hook to lift the robot arm.

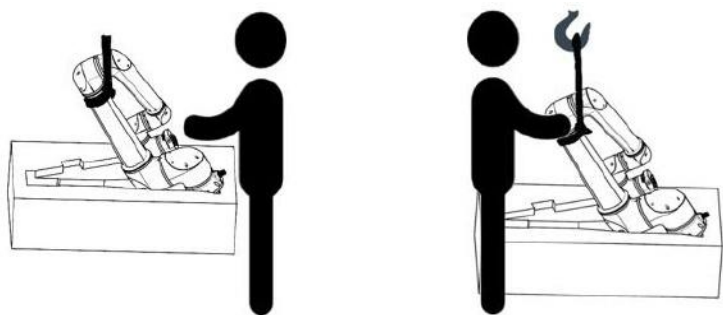


Figure 3-3 Schematic Diagram of the Position of Lifting Straps for 10kg and Below

c) Installation

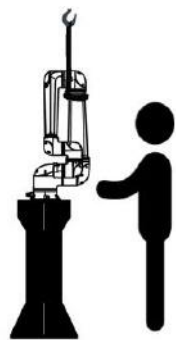


Figure 3-4 Installation Diagram for 10kg and Below

3.2.2.2 20kg robot handling method

a) Transportation and Unpacking

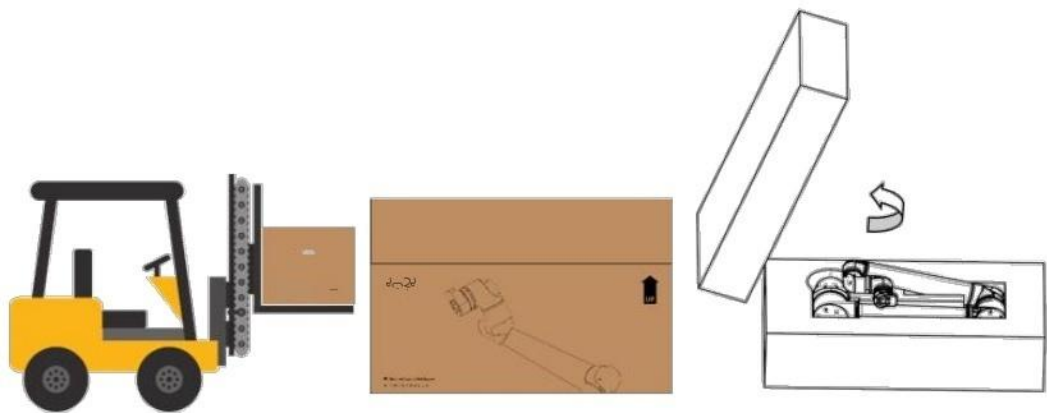


Figure 3-5 20kg Transportation and Unpacking Diagram

b) 2. Install the lifting slings and use the hook to lift the robot arm.

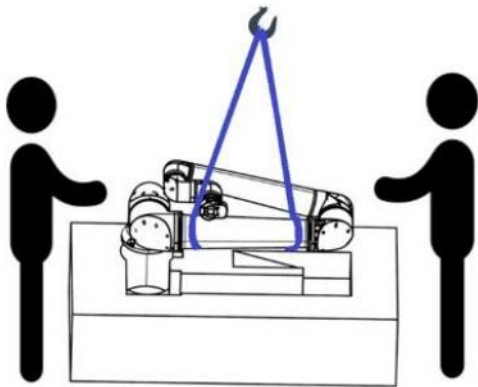


Figure 3-6 Schematic Diagram of the Position of the 20kg Lifting Strap

c) Installation

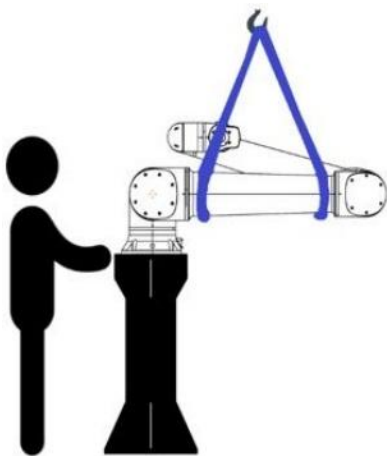


Figure 3-7 Installation Diagram for 20kg Model



WARNING
Lifting or moving heavy parts may cause injury.
- Lifting equipment/lifting aids may be required.



WARNING
Incorrect assembly of components and/or wiring may result in injury.
- Personal protective equipment (shoes, glasses, gloves) may be required.
- Failure to use lifting devices appropriate for the weight of the robot may result in injury to personnel and property damage.
Failure to use a lifting device appropriate for the weight of the robot may result in injury to persons and damage to property.
- The lifting device should be capable of lifting a weight of 59 kg (robot only).
- The lifting device should be able to lift 79 kg (robot and payload).


Lifting sling usage: The lifting sling selected should comply with the following standards under the premise of meeting the load of this product:

European Standard:

- BSEN1492-1:2000 + A1:2008 Textile slings - Safety - Flat webbing slings, made of man-made fibers, for general purposes.
- BS EN 1492-2:2000 + A1:2008 Textile slings - Safety - Round slings made of synthetic fibers for general purposes.

Chinese standard:


- JB/T8521.1-2007 Safety of woven slings - Part 1: Flat web slings for general purposes made of synthetic fibers
- B/T8521.2-2007 Safety of woven slings - Part 2: General purpose synthetic webbing slings for round lifting



WARNING

Using a round sling without inspection may result in injury.

- Inspect slings before and after each use.
- If possible, check the slings during use.




WARNING

Using a damaged round sling may result in injury.

- Inspect slings before and after each use.
- Do not use if sling is cracked, torn or has loose stitching.
- Do not use if sling shows signs of heat damage.
- Protect slings from sharp edges and friction.
- Do not tie knots in the sling.
- If possible, check the sling during use.

3.2.3 Installation



WARNING

Before performing safety sensing on the equipment, confirm that the operator needs to wear safety gloves, glasses, anti-smash shoes and other safety protection equipment to avoid dangerous injuries during installation.

Mount the robot arm using bolts of at least grade 12.9 strength and the mounting holes in the base as shown. See Section 4.5 for robot base mounting dimensions.

The recommended installation torque is as follows:

Item	S3-60	S5-90	S10-140	S20-180
Bolt	M6	M8	M8	M12
Quantity	4	4	4	4
Flat washer	Φ6	Φ8	Φ8	Φ12
Locating pin	Φ4	Φ6	Φ8	Φ8
Torque	>=10N·m	>=20N·m	>=35N·m	>=70N·m

The robot needs to be installed on a solid and vibration-free support surface. The support must be capable of withstanding at least ten times the full torsional force of the

first joint and at least five times the weight of the robotic arm.



Figure 3-8 Ontology Installation

The robot can be installed in any position and orientation, supporting various installation methods such as overhead and side mounting. For non-vertical installations, the installation angle of the robot needs to be set in the robot settings. For the installation method of the robot body and the setting method of the installation angle in non-vertical installations, please consult our technical personnel.

Cable Connections

Before powering up the robot, you need to connect the robot cables according to the cable connection diagram in Figure 3- 9.

Network cable: connects the controller to the tablet for controlling the robot's movements.

Hand controller: used for controlling the emergency stop, enabling and power on/off of the robot.

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

Power cable: Provides power for the robot system.



WARNING
Before energizing the robot, check that the voltage and frequency of the power supply meet the requirements; accessing the wrong voltage can cause the robot to malfunction.



WARNING
A power cord is included in the package, but since single-phase power outlets vary from country to country and region to region, please purchase a power cable that meets the requirements according to the customer's region.

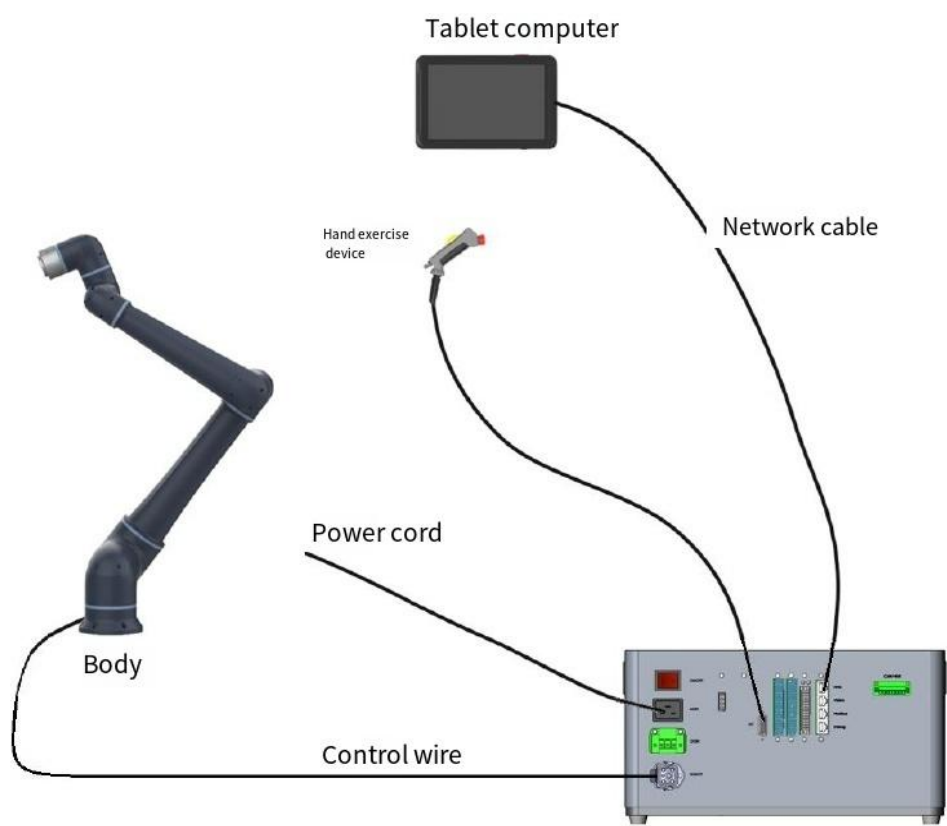


Figure 3-9 Schematic Diagram of Cable Connections

3.2.4 Operation position layout

The positions of the operator, the robot body and the control cabinet equipment are as shown in the operation position layout diagram in Figure 3-10.

Suggestion: When the robot is in operation, the operator should stand outside the reach of the robot's arm to ensure personal safety.

This robot is equipped with a collision detection function and complies with the IOS/TS 15066:2016 standard.

Refer to the content described in Section 4.2 on the workspace to ensure that the operator operates the robot outside the robot's workspace area. Do not operate the robot when there are people inside the robot's workspace area.

Refer to the content described in Section 4.7 for the placement of the control cabinet. Ensure that the control cabinet is installed in a well-ventilated, flat and vibration-free environment.



WARNING
Before energizing the robot, check that the voltage and frequency of the power supply meet the requirements; accessing the wrong voltage can cause the robot to malfunction.

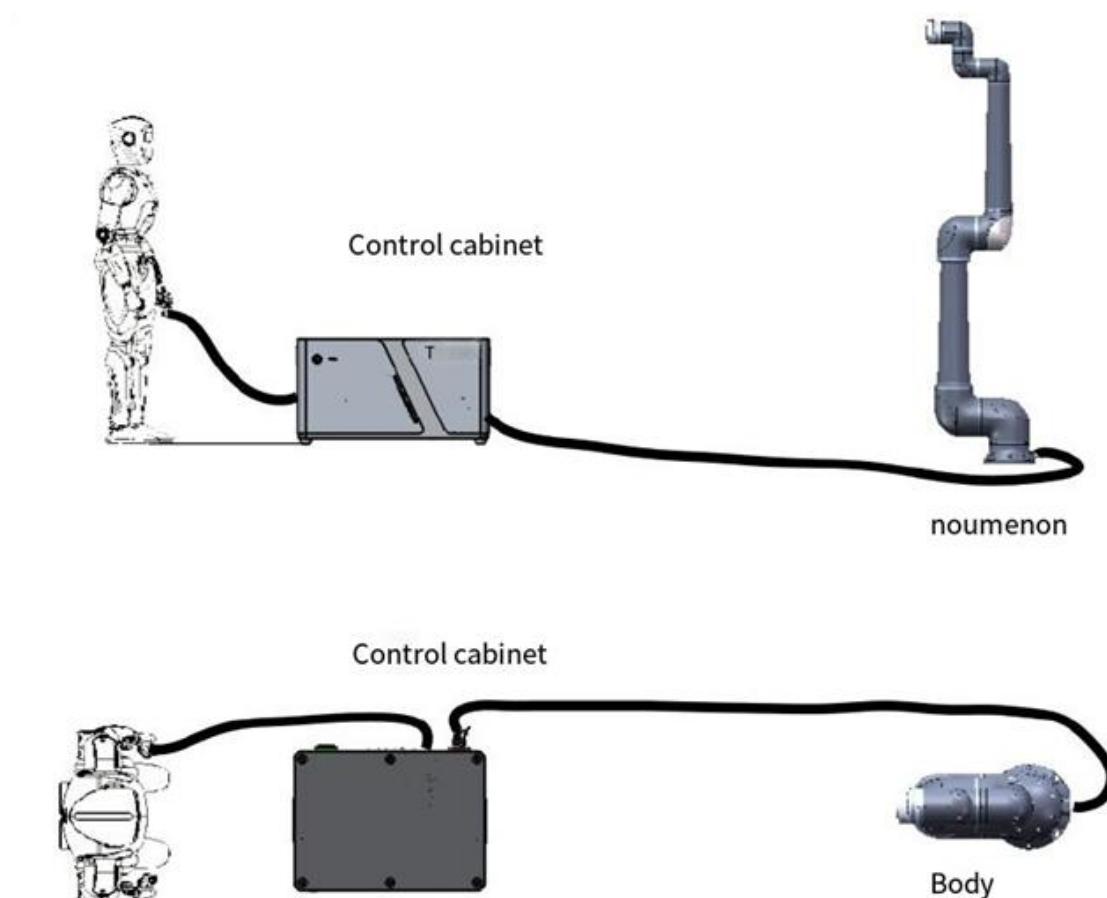


Figure 3-10 Layout Diagram of Operating Positions

3.3 Start using

After completing the above tasks, you can start using the robot.

3.3.1 Power on and start up

When the robot cables are properly connected, power on the system and turn on the switch at the power connection of the controller. At this point, you can hear the controller's fan start working. Then press the power-on button on the front of the control box. The button will turn green and stay lit, indicating that the control box has been powered on.

2. Wait until the light strip at the end of the robot turns into a constant white light and the small screen at the end of the arm shows "Communication [Real-time]" and "Operation [Normal]". This indicates that the controller has been successfully started and the robot body has established communication with the controller. At this point, you can log in to the web page to control the robot.

3. Turn on the tablet and modify its static IP address to 192.168.101.XXX through the settings.

4. Open the browser and enter the robot's IP address: 192.168.101.100:9098 in the address bar. Press Enter to jump to the login page as shown in Figure 3-11. If you cannot jump to the login page, please check the IP address of the tablet. If you still cannot

access the login page, please contact the after-sales service personnel.

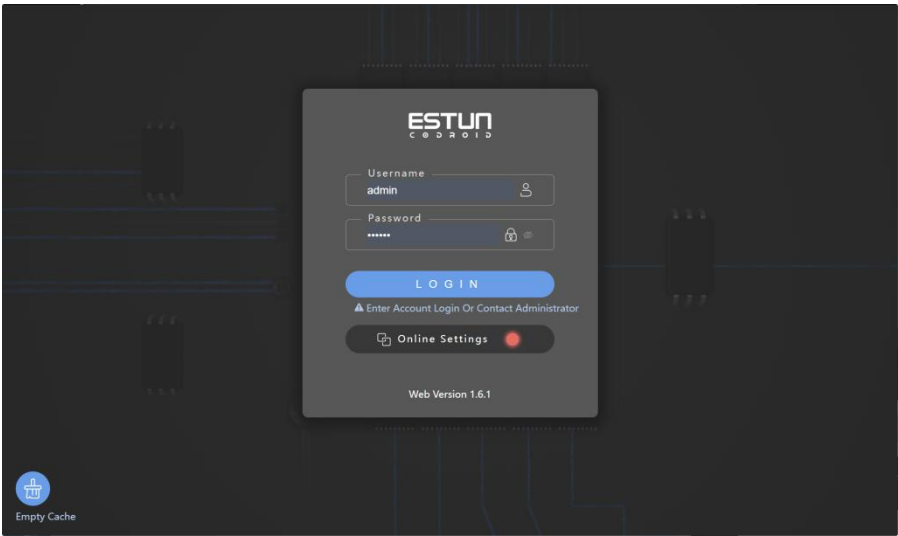


Figure 3-11 Login Page

5. You can log in to the control page by entering your account and password. The initial accounts and passwords available for use are as follows. Different accounts have different permissions. For details, please refer to the appendix.

Account	Password
User	123456
Admin	123456

6. After logging in, you can jump to the robot control page, at this time you can carry out the power-up operation, before powering up, you need to make sure that the emergency stop button on the hand manipulator has been reset, and there are no people and equipment within the robot's range of motion. Click the button



“power on” in the “3D Simulation” view, as shown in Figure 3-12 Robot Control Interface, you can hear the sound of the brake release at the joints, indicating that the joints are powered up.

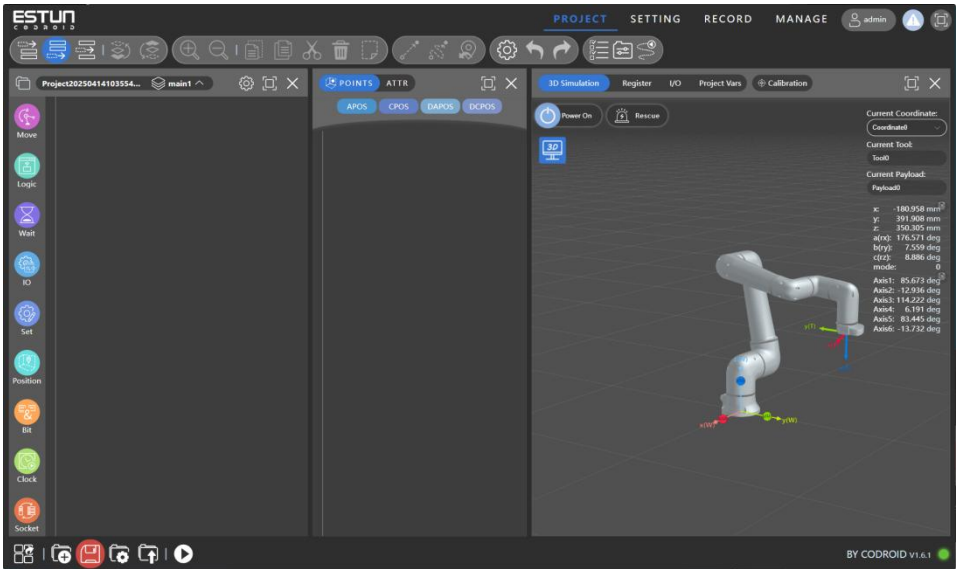


Figure 3-12 Robot Control Interface

7. As shown in the main interface of Figure 3-13, it indicates that the robot has been powered on successfully. Now, you can control the robot to move.

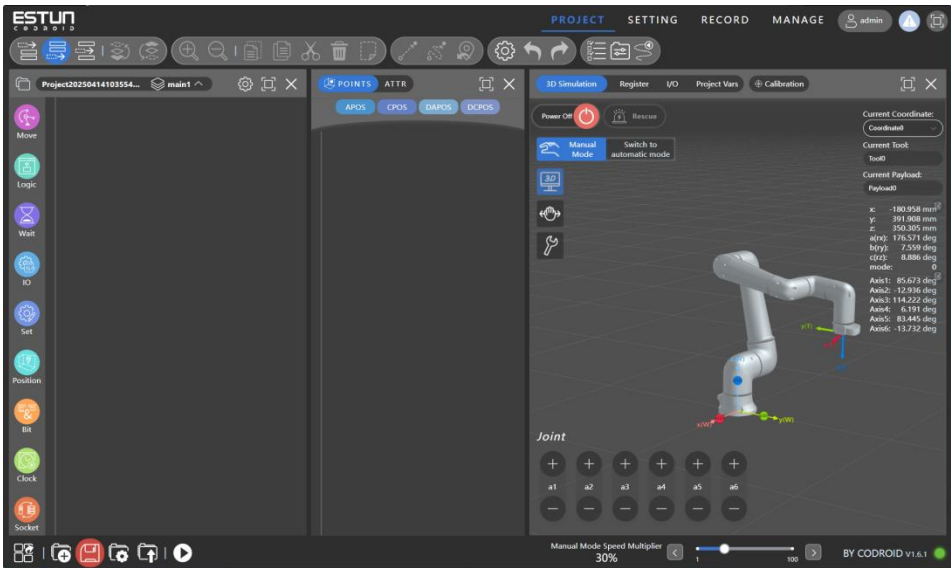




Figure 3-13 Main Interface

3.3.2 Write a program

In manual mode, the robot can perform joint jogging and end-effector jogging.

Joint point movement: It enables the control of the robot to perform single-joint movements. The speed of point movement can be adjusted manually by changing the speed ratio. a1, a2, a3, a4, a5, and a6 respectively represent the six joints of the robot.

Endpoint jogging: It enables the control of the robot's movement in the Cartesian coordinate system. The jogging speed can be adjusted by changing the manual speed multiplier. The reference coordinate system during the robot's movement can be changed by switching between the current coordinate system and the tool coordinate system. x, y, and z represent the directions of the three axes of the reference coordinate system, while a, b, and c represent rotations around the x, y, and z axes of the reference coordinate system, respectively.

- In manual mode, control the robot to move to the target point.
- Click “ POINTS” pose”, click “ ” to record a point P1;
- Jog the robot to another position and repeat steps 1 and 2 to add the second point, as shown in Figure 3-14.

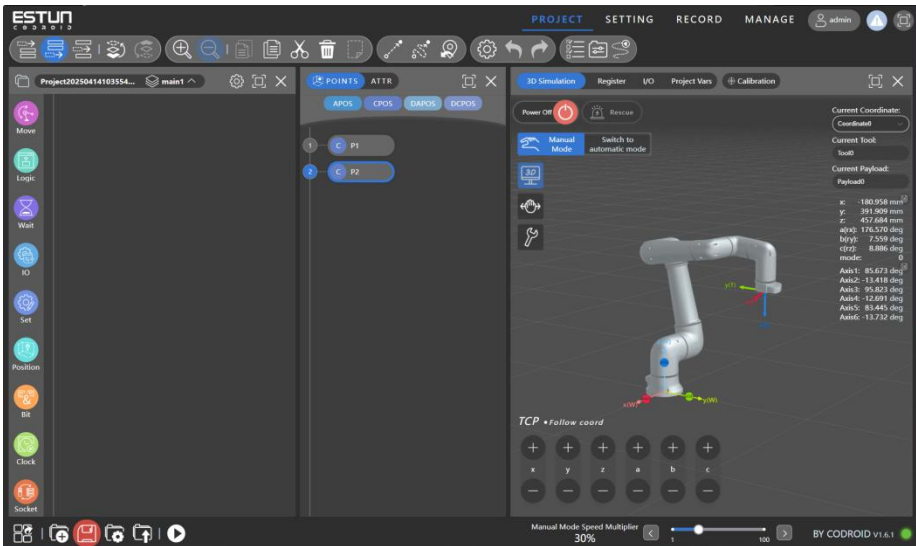




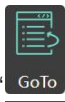



Figure 3-14

- d) Select  "Move jogging" in the left instruction column and click or directly drag  and drop instruction "MovL" to add a motion instruction to the right program tree;

- e) Select  in the program tree, click  parameters", the corresponding parameter column of the instruction will appear, the target position select the point P1 set just now, and the target speed select the system default V100, when the instruction parameter page doesn't show the red error word, it means that the setting of the instruction is completed;6. Repeat Step 5 and add another instruction with its parameters set.
- f) Repeat step 5 to add another command and set the parameters.

- g) 7. Add the  "Label" instruction from  logic" and drag it to the first line of the program.

- h) Adding the  "GoTo" instruction in  logic" and selecting the jump node as Start allows the program to run continuously and repeatedly, as shown in Figure 3- 15 below.

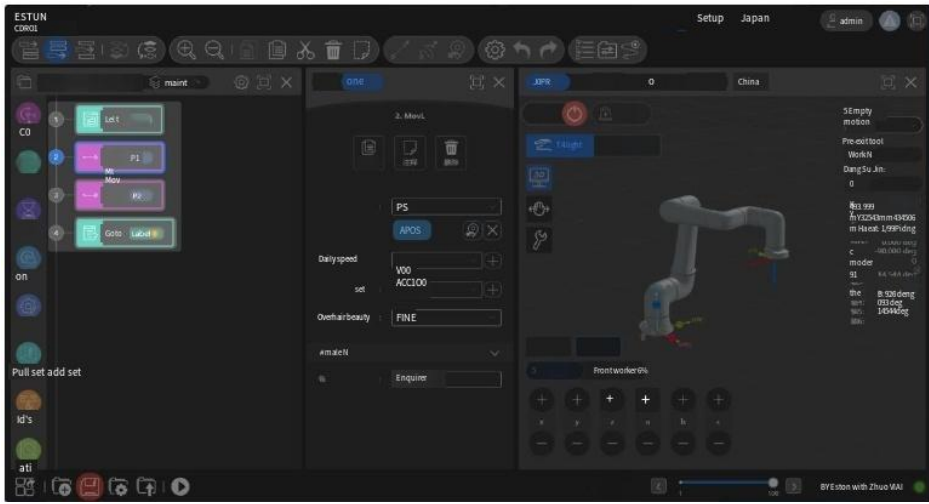

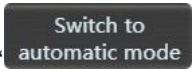




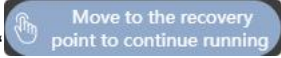



Figure 3-15

- i) Click  to save the program, there will be a pop-up when it is saved;
- j) Click  and select OK to switch the robot to automatic mode
- k) Click  to select Auto Execute and the robot will move from P1 to P2.
- l) Select and confirm to switch the robot to automatic mode.

- m) Click on "" to pause the robot program and the robot will pause its movement at the same time.
- n) After the robot has paused, click on "" to resume running the robot program.
- o) Clicking on "" will stop the robot program and the robot will stop moving.
- p) If the robot is manually moved or stopped after the program has been paused, it will need to be switched to manual mode first, and then switched to automatic mode again after robot " move to running resume point to continue running" before the program can continue to run.

3.3.3 Power off

Adjust the robot's attitude to the proper position, click " shut down" and the robot powers down, then press and hold the power button on the controller until the always-on green light goes out to release the button.

Chapter 4 Mechanical Hardware and Installation

4.1 Robot composition

The Codroid S series robot features six rotating motion joints, a large arm and a small arm as connecting rods. The base of the robotic arm is equipped with an aviation plug, the end of the robotic arm is equipped with a button and an indicator light, and the side of the tool flange is equipped with a button, a screen and an aviation plug.

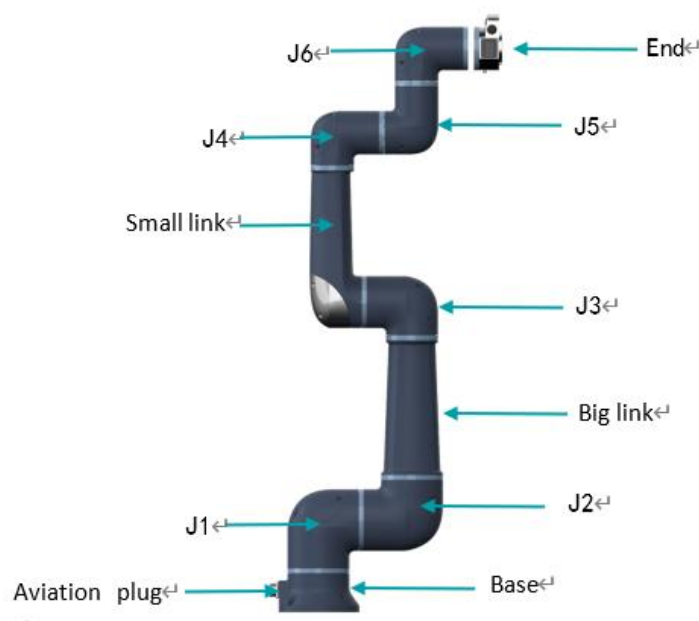


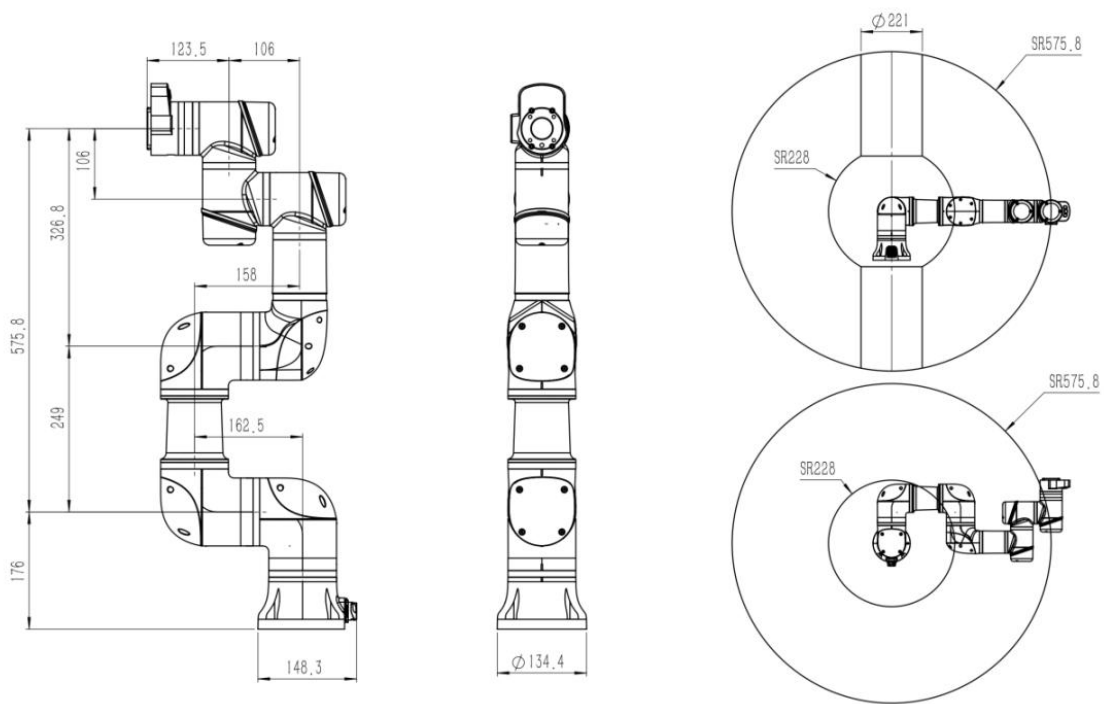
Figure 4-1 Composition of S Series Robots

4.2 Work Space

When choosing the installation location for the robot, it is essential to consider the cylindrical space directly above and below the robot. It is necessary to avoid moving the tool towards this cylindrical space, as doing so would lead to entering the singularity point, causing the joints to move too fast during operation. This would result in low robot efficiency and make risk assessment difficult.



WARNING
When the robot is operated in manual mode (taught), personnel should be outside the safe guarding space. The emergency stop button of the robot hand manipulator must be within reach in manual mode, and at least one emergency stop switch needs to be set outside the robot's range of motion. The robot's range of motion is the maximum range of motion of the body when the robot does not have any motion limits set. Robot movement limits can be set so that all operations do not fall outside the maximum range of motion of the robot body.



Fi

Figure 4-2 Dimensions and Working Space of S3-60

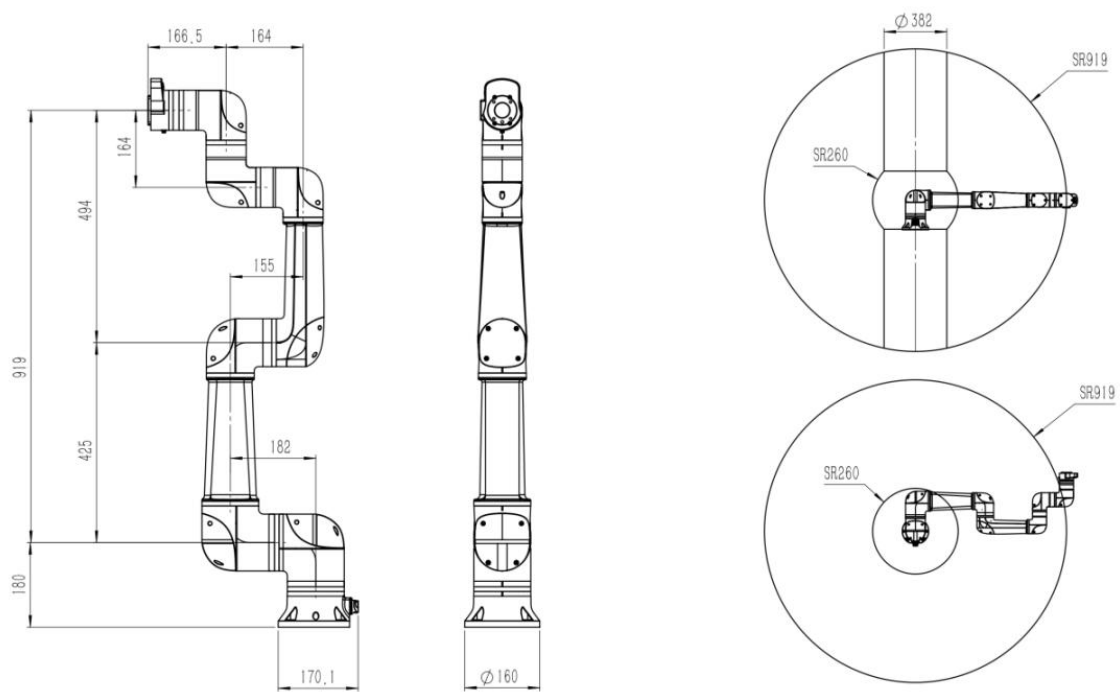


Figure 4-3 Dimensions and Working Space of S5-90

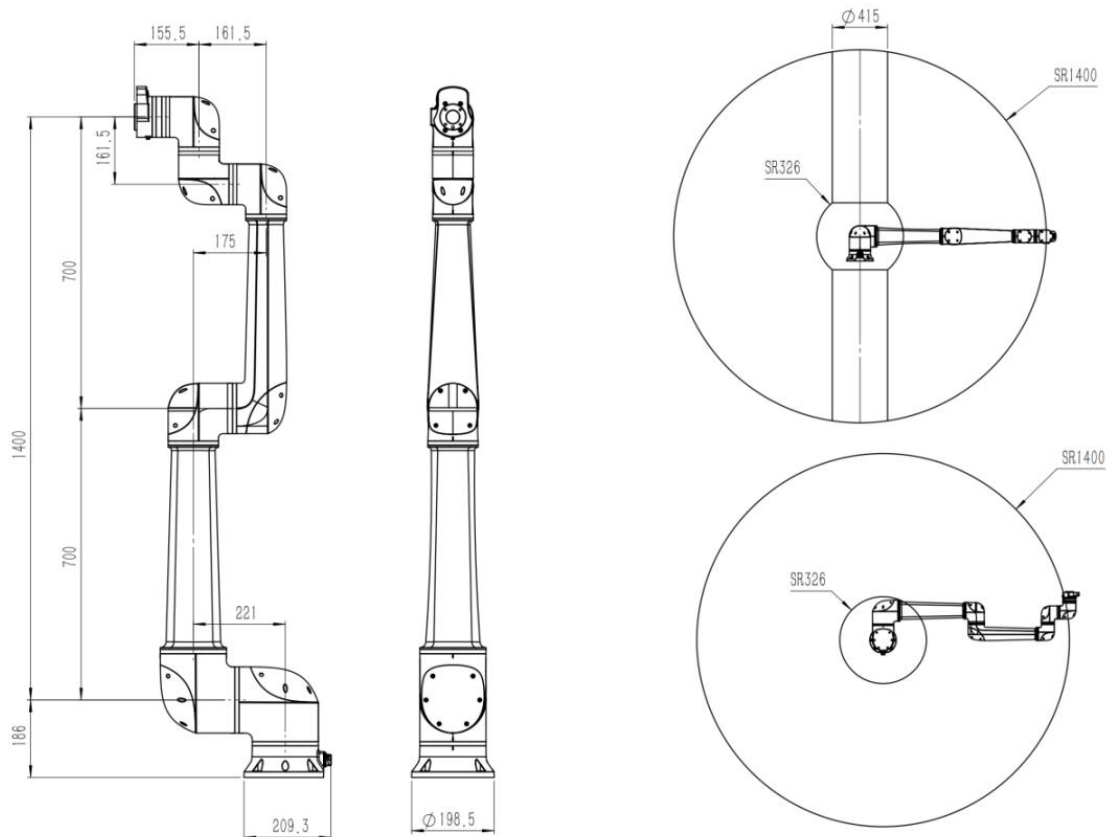


Figure 4-4 Dimensions and Working Space of S10-140

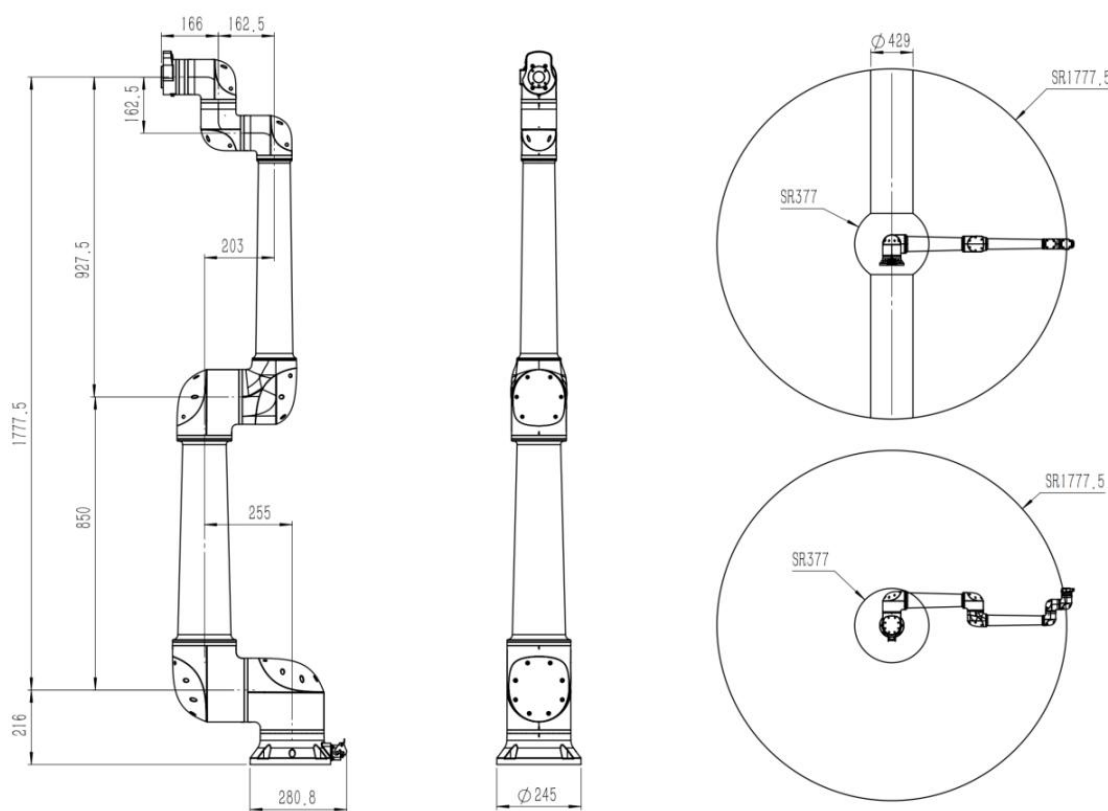



Figure 4-5 Dimensions and Working Space of S20-180

4.3 Load curve

The maximum allowable payload of a robotic arm depends on the offset of the center of gravity. When the distance of the load's center of gravity increases, the load that the robot can bear decreases. According to the eccentric distance of the load, with the eccentric distance on the XY plane as the ordinate and the value of Z as the abscissa, find the corresponding coordinate point of the eccentric load. Observe under which curve this

point lies. The load indicated by that curve is the maximum load that the robot can bear under the current working condition.

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



WARNING

When calculating loads, the weight of the media flange must be included and ensured to meet the robot's load specifications. Ensure that the system never exceeds the maximum allowable load. The user should carry out a full risk assessment of the media flange and the workpiece to avoid shock, vibration, crashing, entanglement, puncture, puncture and other hazards. Ensure the overall safety of the system.

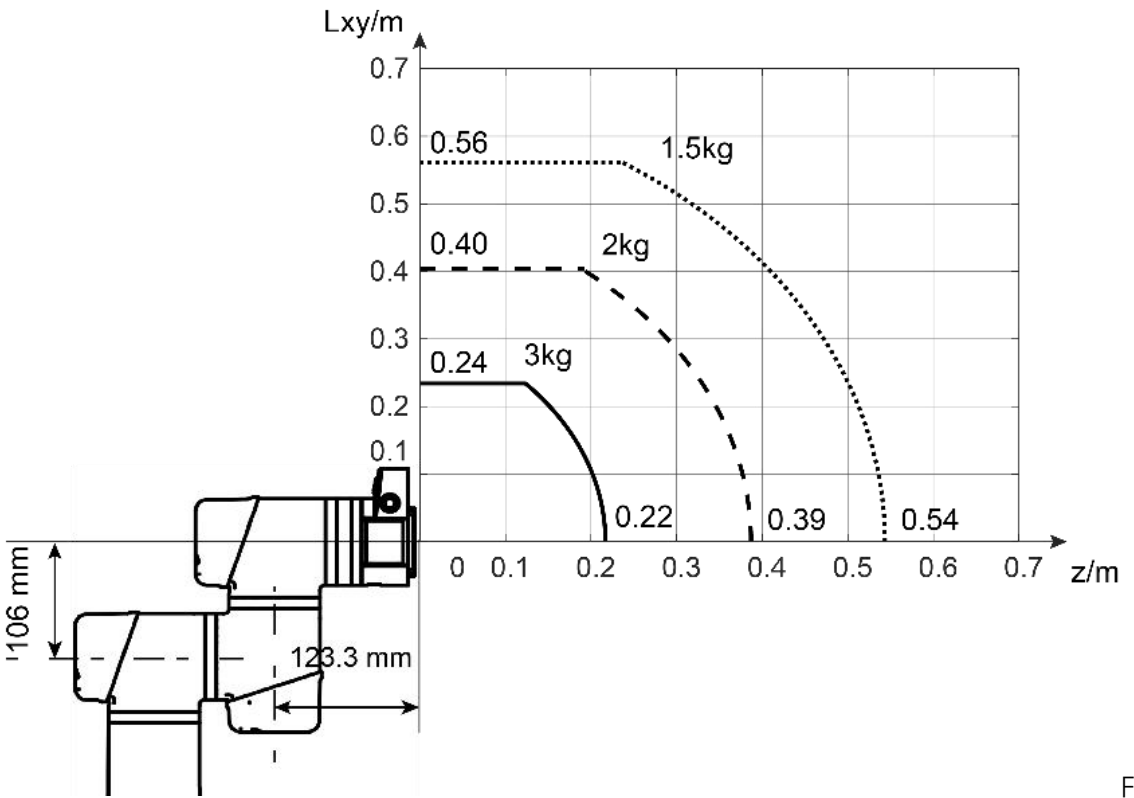


figure 4-6 Payload Curve of S3-60

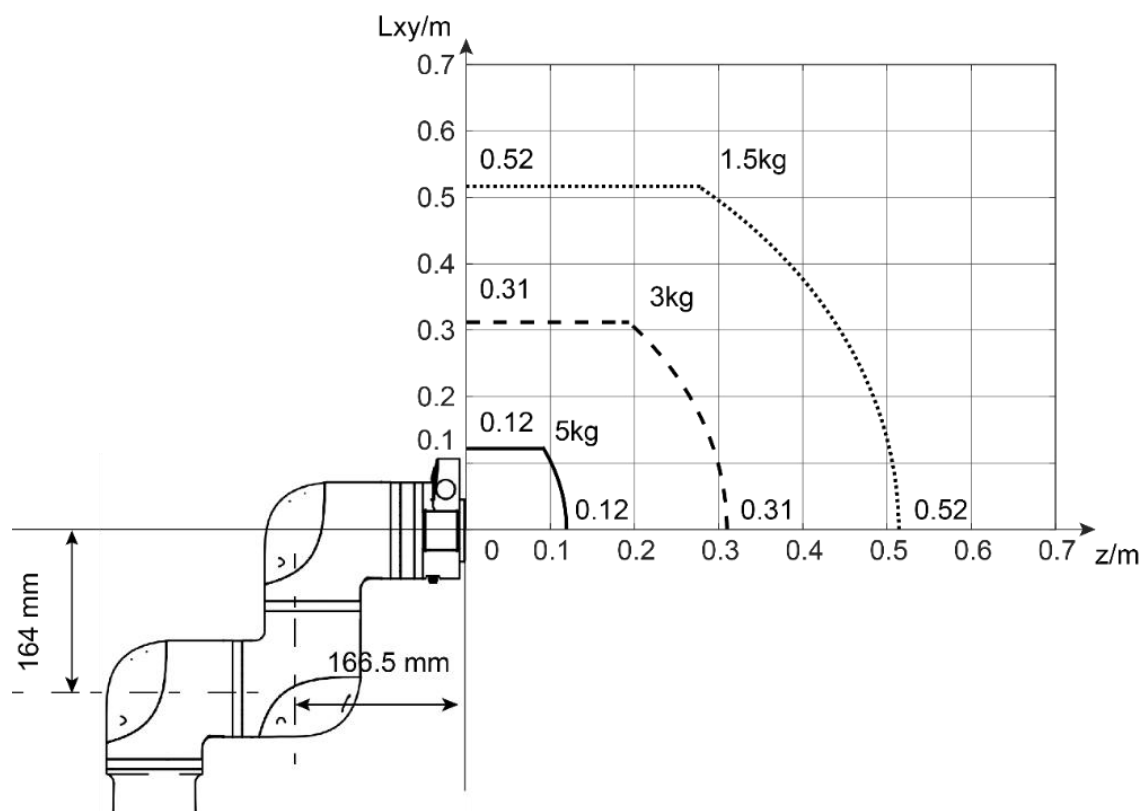


Figure 4-7 Payload Curve of S5-90

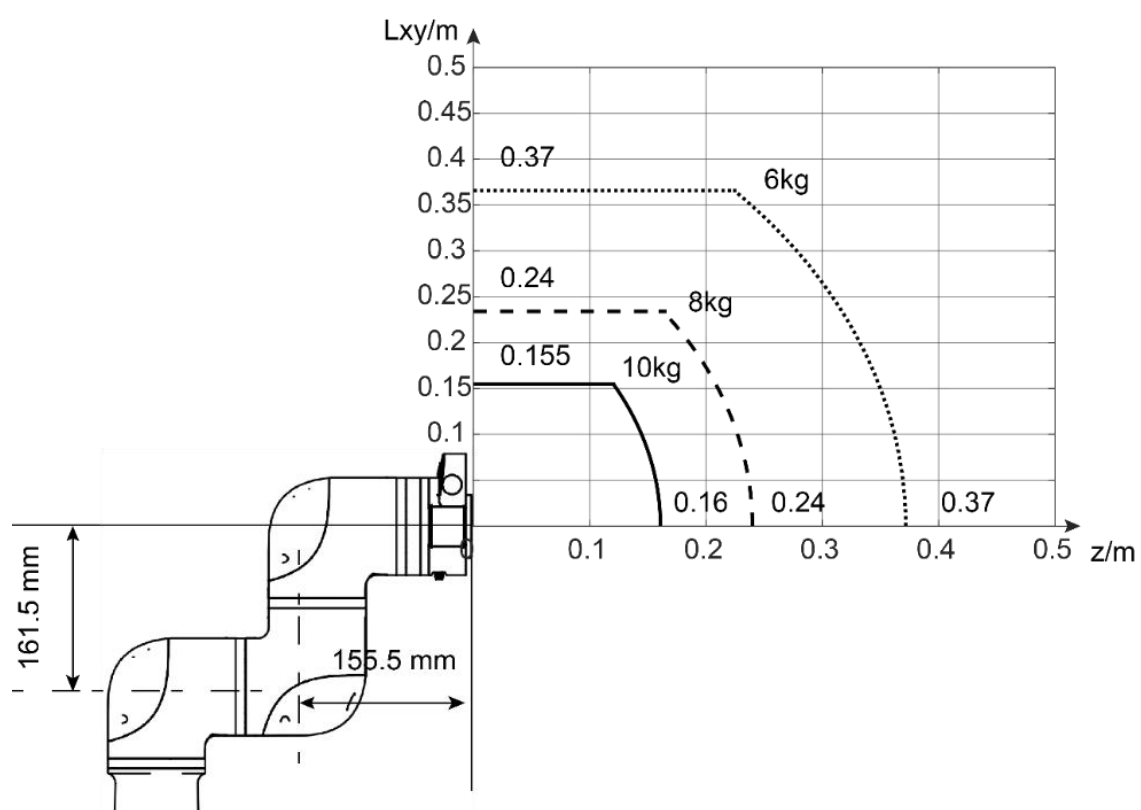


Figure 4-8 Payload Curve of S10-140

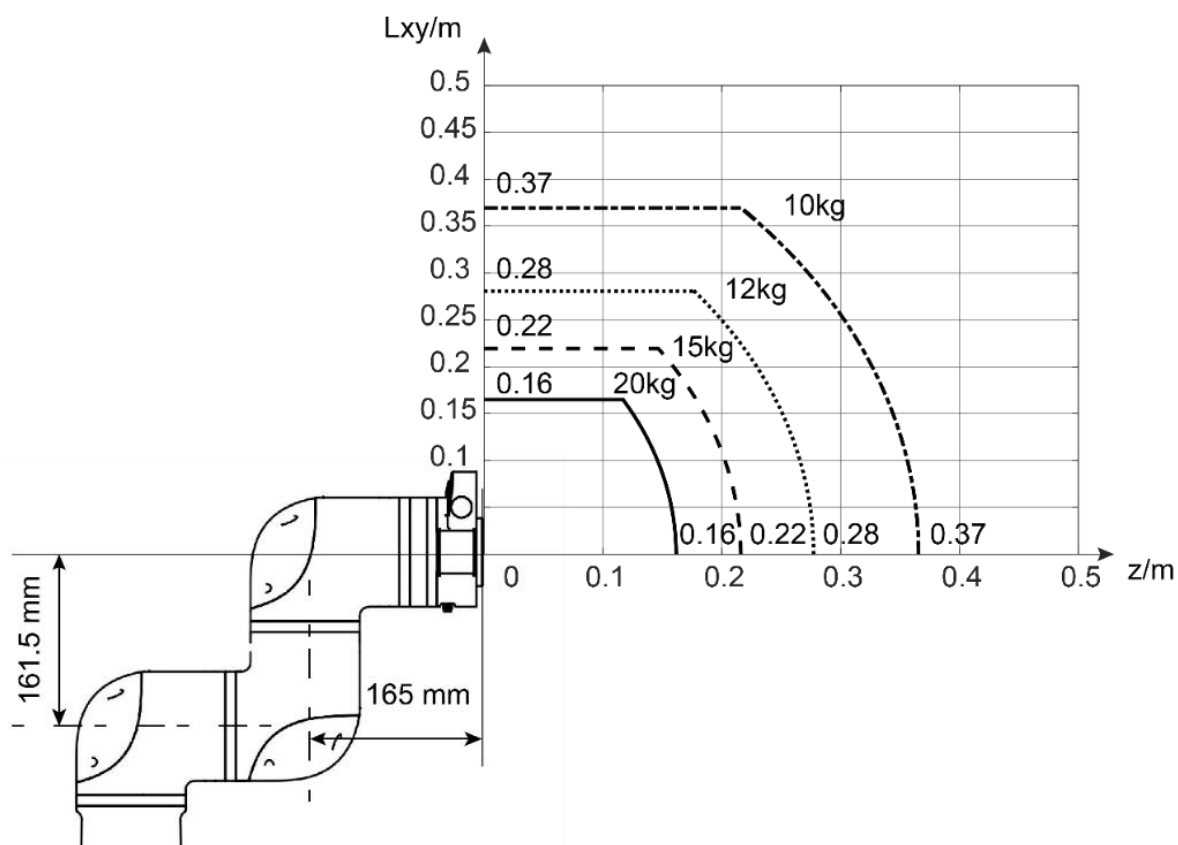


Figure 4-9 Payload Curve of S20-180

4.4 Flange interface

The end flanges of the S series robotic arms all have the same size. Each flange has four M6 threaded holes, which can be used to attach tools to the robot. The flange design complies with the national standard GB/T 14468.1-50-4-M6 (or ISO 9409-1-50-4-M6).

The M6 screws must be tightened with a torque of 12Nm, and their strength grade is 12.9. To accurately reposition the tool, please use a pin in the reserved $\varnothing 6$ hole to maintain the precise position. The screw insertion depth for installing the tool must not exceed 8mm.

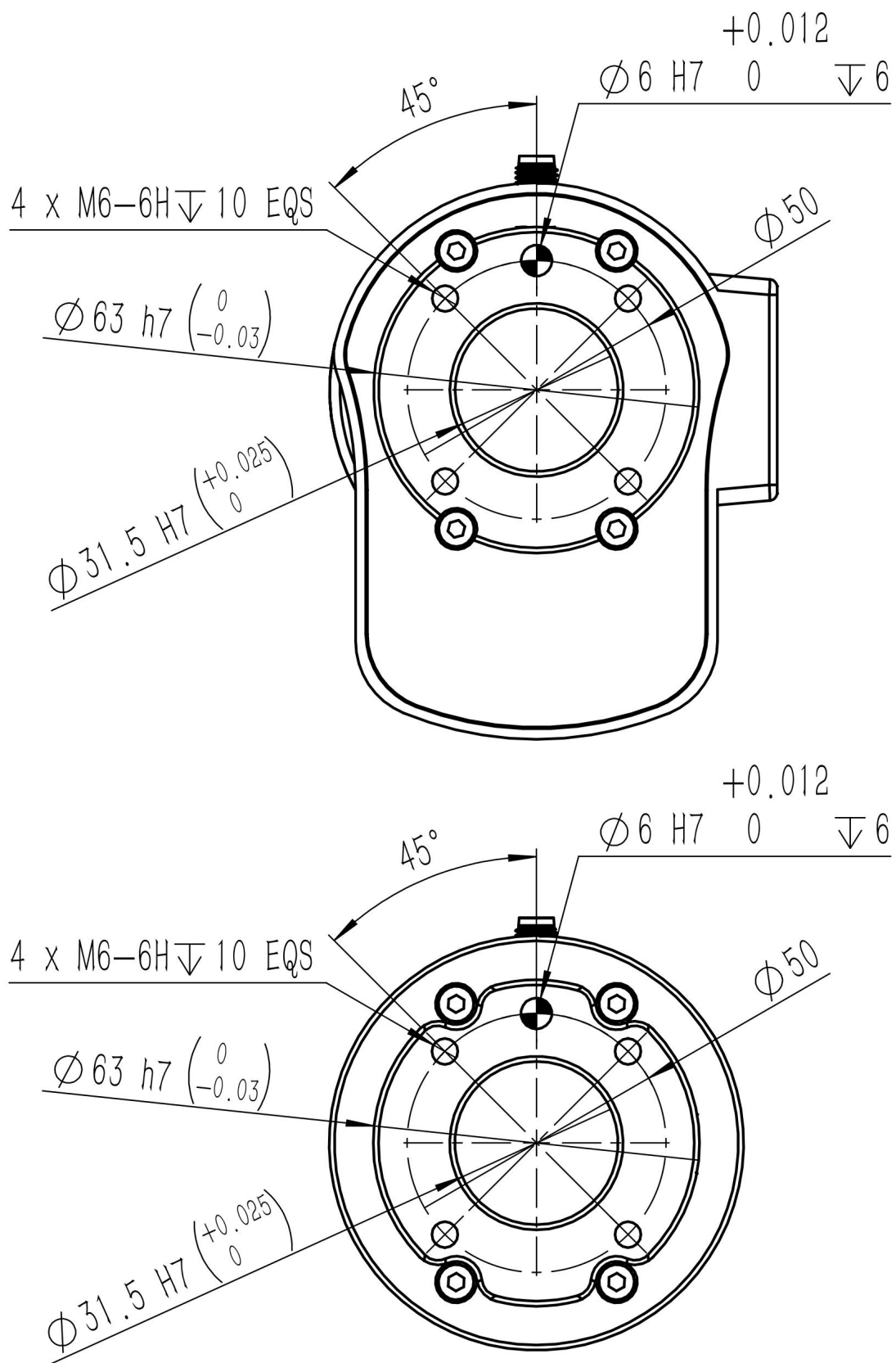


Figure 4-10 Mechanical Installation Interface of S3-60 Pro and Eco Flange

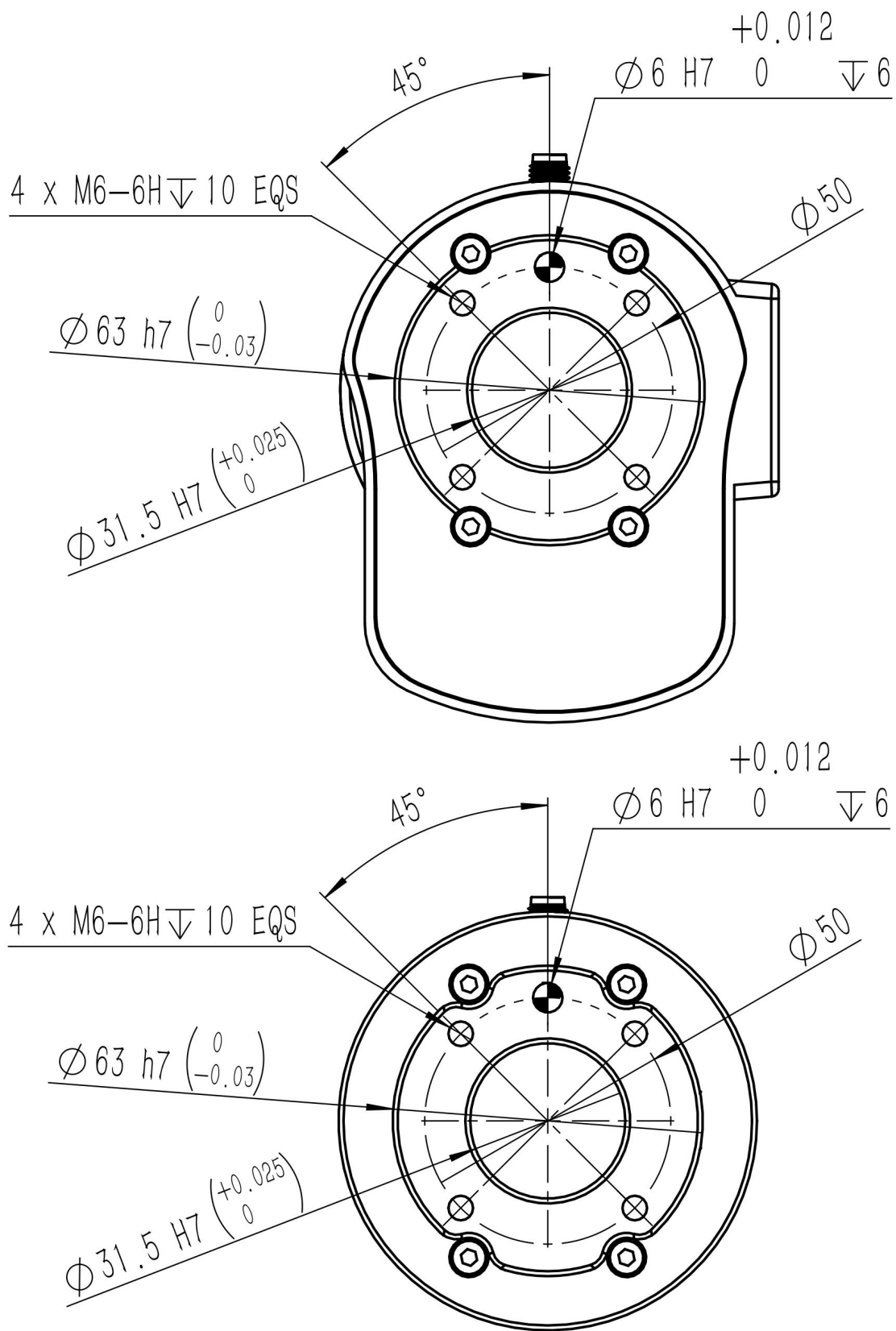


Figure 4-11 Mechanical Installation Interface of S5-90 Pro and Eco Flange

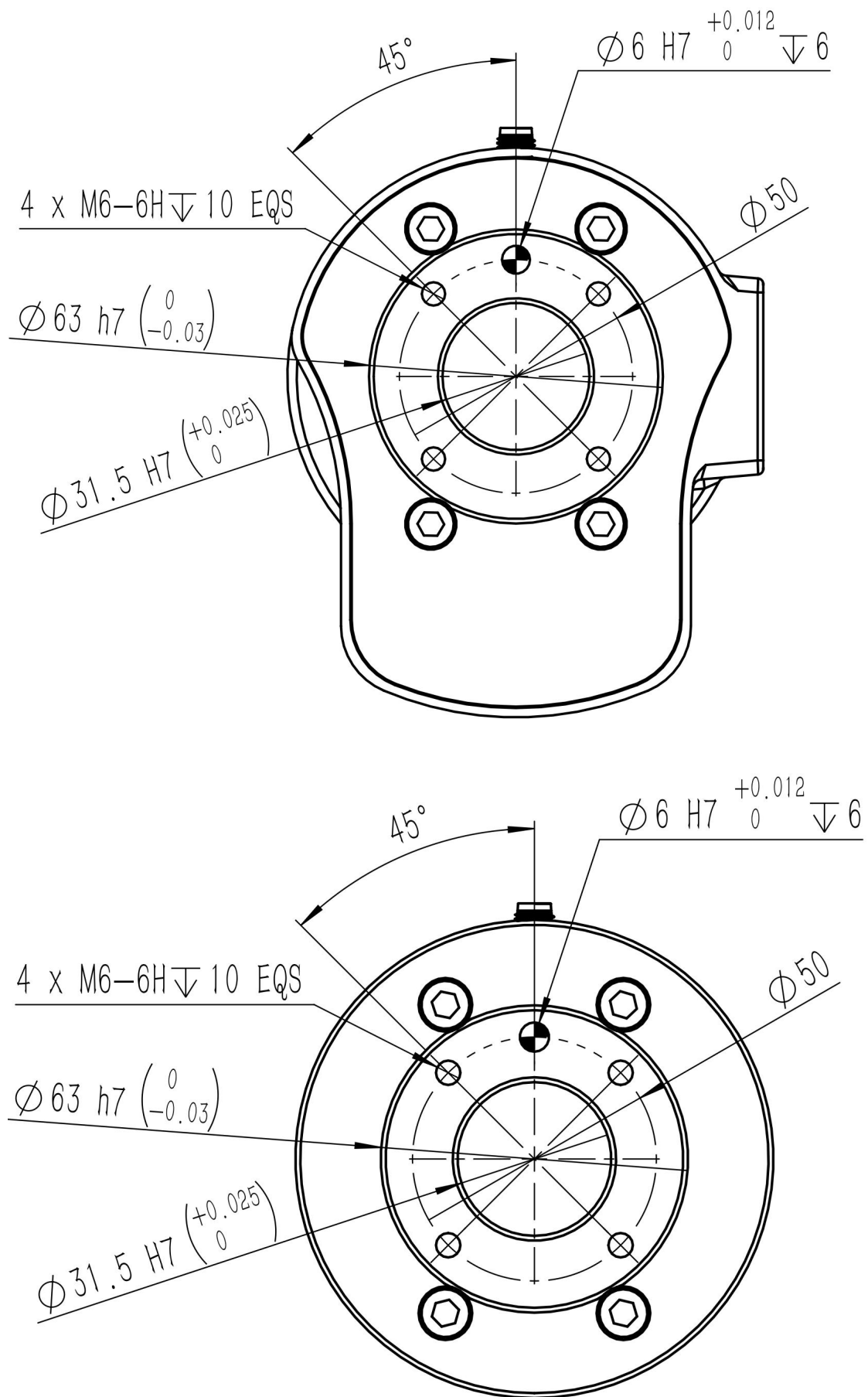


Figure 4-12 Mechanical Installation Interface of S10-140 Pro and Eco Flange

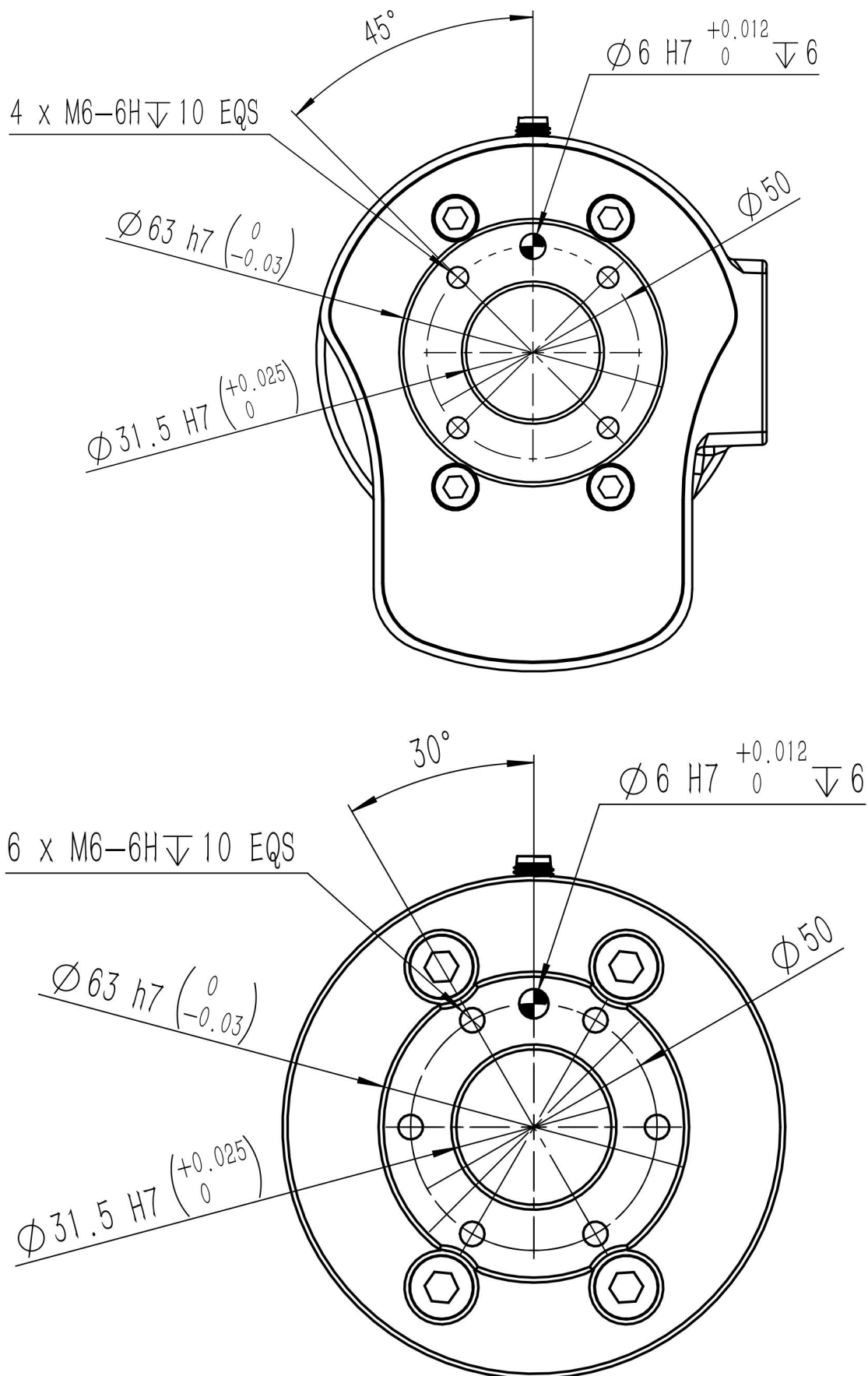


Figure 4-13 Mechanical Installation Interface of S20-180 Pro and Eco Flange

4.5 Installation interface

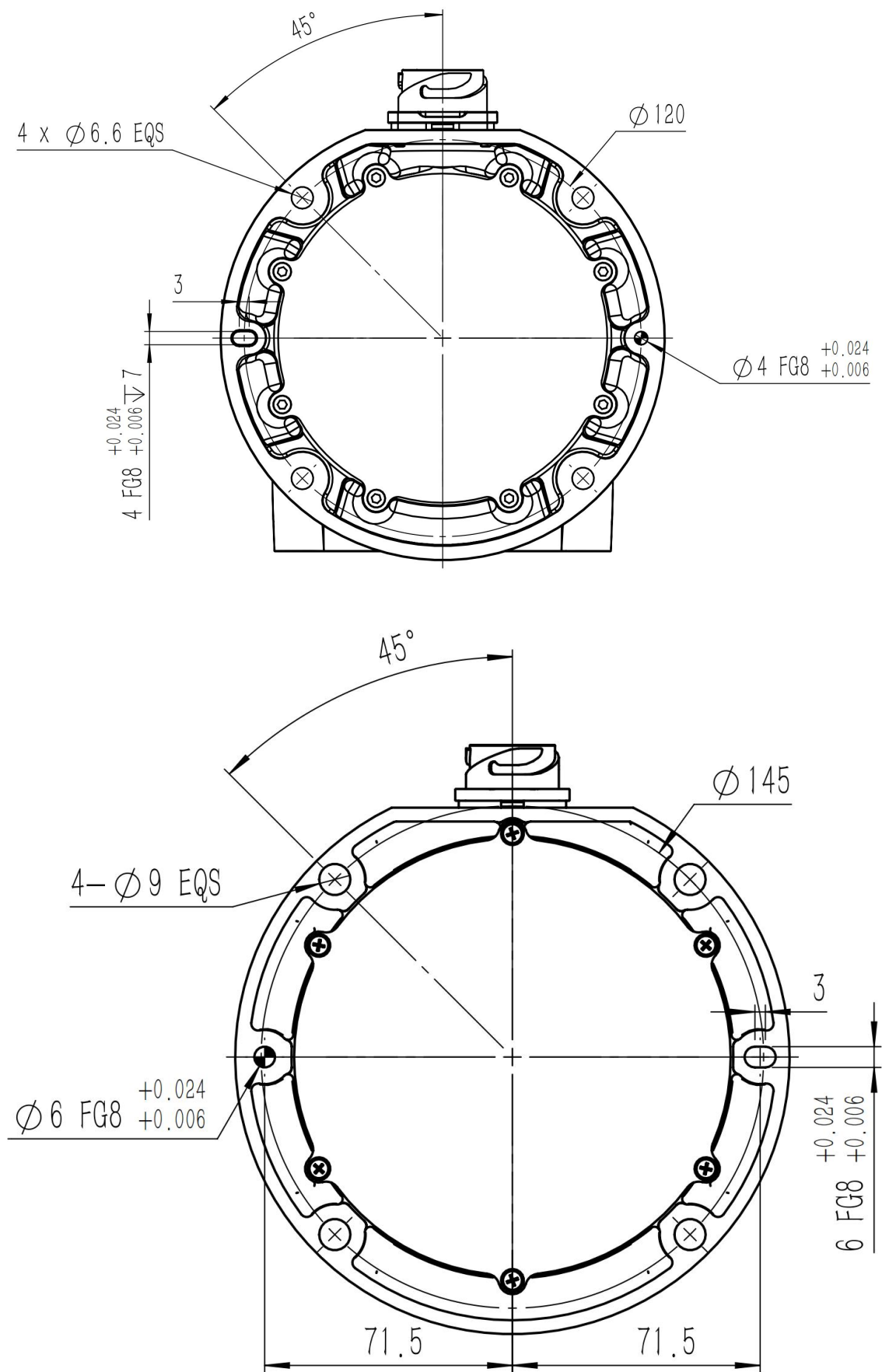


Figure 4-14 Mechanical Installation Interfaces of the Base for S3-60 and S5-90

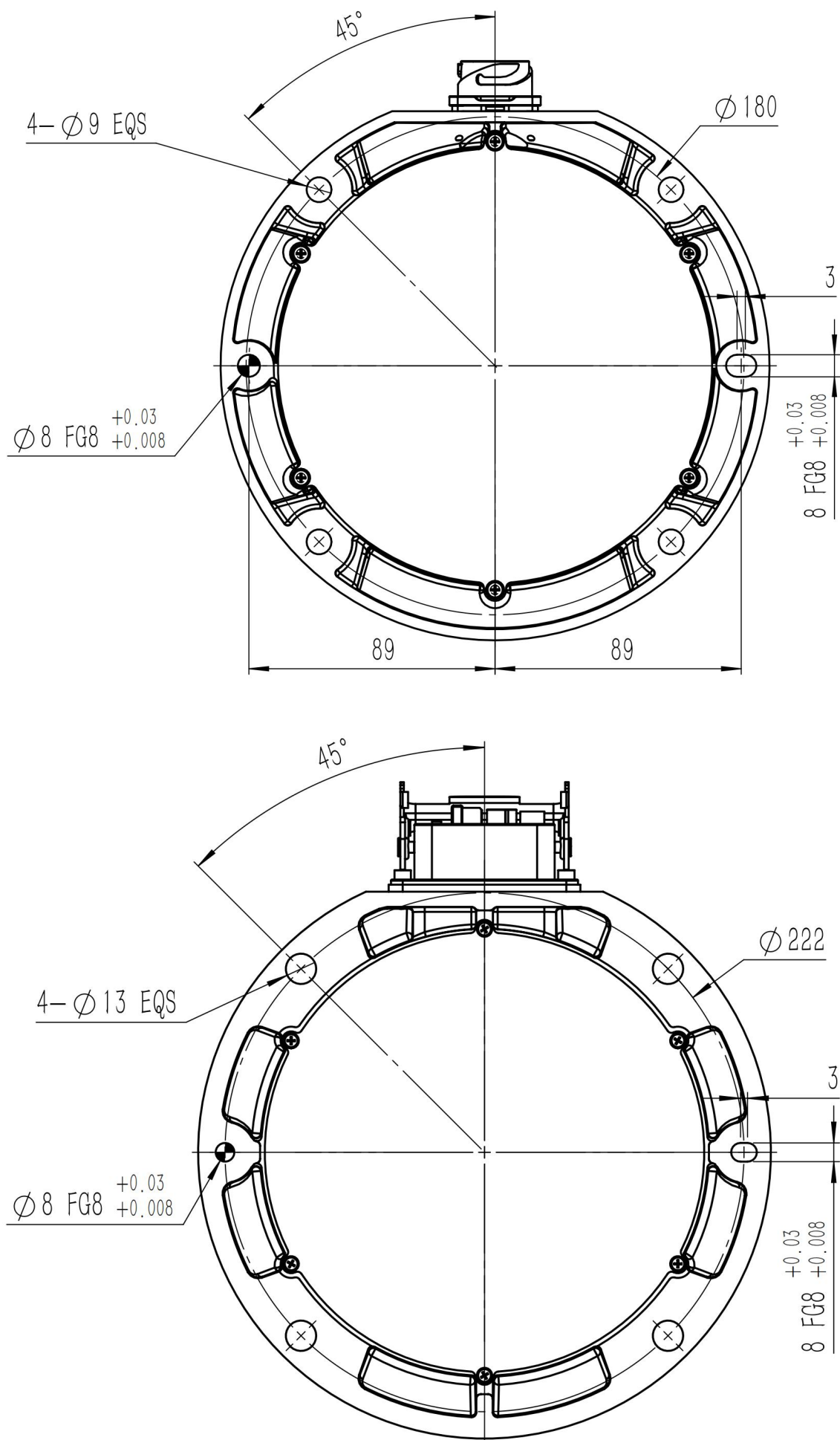


Figure 4-15 Mechanical Installation Interfaces of the Base for S10-140 and S20-180

4.6 Robot Specification

Model	S3-60	S5-90	S10-140	S20-180
DOF	6			
Payload (kg)	3	5	10	20
Reach (mm)	575	919	1400	1777
Repeatability (mm)	±0.03	±0.03	±0.05	±0.1
Weight (kg)	18	22	38	59
Certification	EN ISO 13849-1 PLd Cat.3 & EN ISO 10218-1			
Working range	Axis 1/2/4/5/6: ±360° Axis 3: ±160°			
Max. Speed of axis	【3、5、10kg】 Axis 1/2/3: 150 °/s Axis 4/5/6: 180 °/s			
	【20kg】 Axis 1/2: 110 °/s Axis 3: 150 °/s Axis 4/5/6: 180 °/s			
Max. Speed at Tool End (m/s)	2	2.5	2.5	3.2
Flange Communication	2DI, 2DO, 24VDC, MODBUS RTU, RS485			
Mounting	Any orientation			
Operating Temp.	0 – 40 °C			
Operating Humidity	70% RH			
Operating Noise	≤65dB			

4.7 Control cabinet

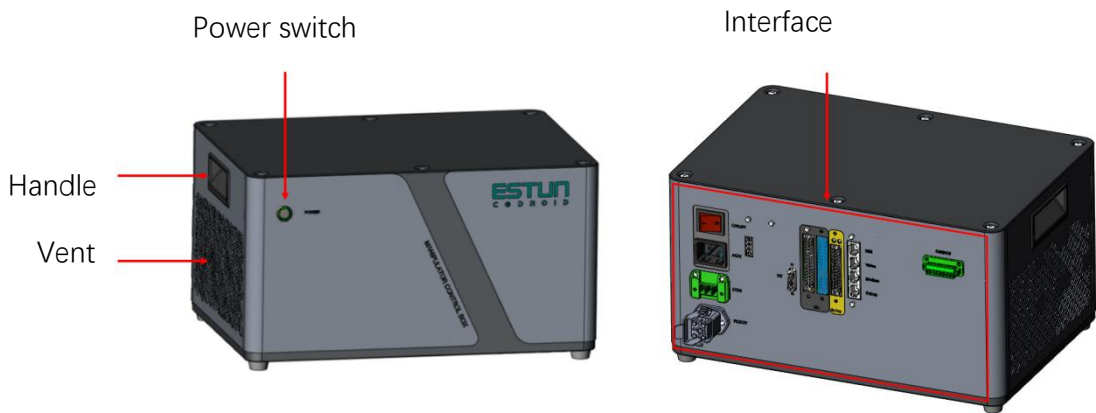


Figure 4-16 Cabinet interface

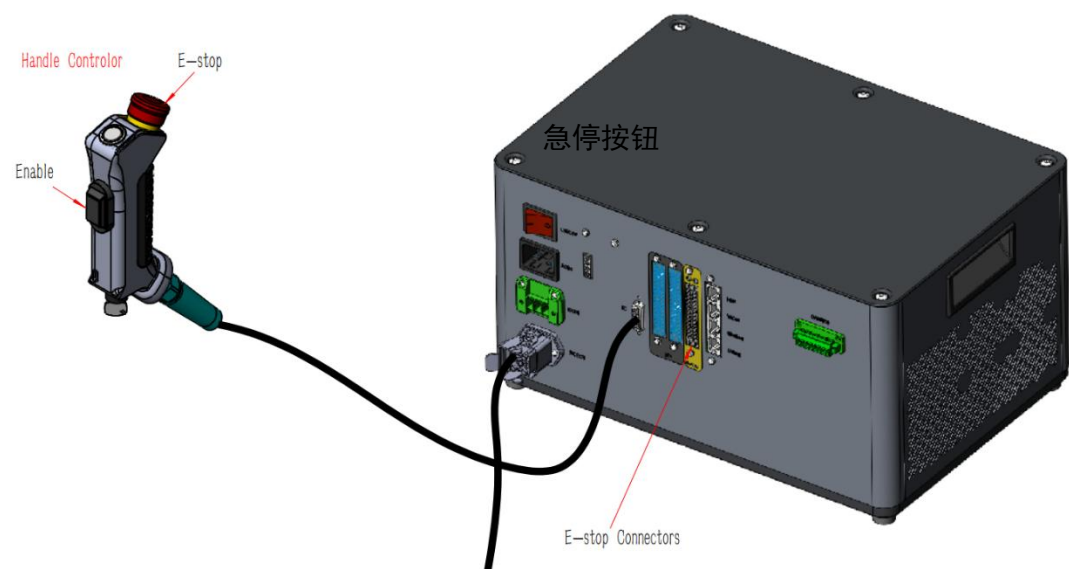


Figure 4-17 Emergency stop and switch position diagrams and safety IO interface position diagrams

Safety device	1 hand-held enable channel, 1 hand-held E-stop channel
IP classification	IP20
I/O ports	16DI, 8DO, 4AI/4AO, 7 stop inputs
I/O power supply	24VDC, 2A
Operating temp.	0~40℃
Operating humidity	10~90%RH, Non-condensing
Noise	≤65dB
Altitude	Below 1000m
Power	AC100- 240V, 50/60Hz
Dimensions	380mm x260mm x 200mm
Weight	14kg (Cabinet of 20kg robot) , 11.8kg (Cabinets of 10kg and below)

4.8 Handle operator

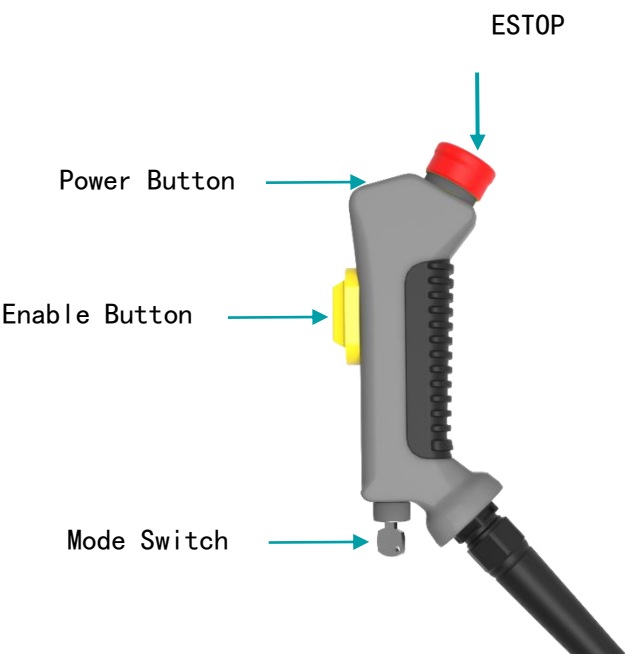


Figure 4-18 Hand Controller Interface


The hand controller contains four switches or buttons, namely the emergency stop button, the power on/off button, the enable switch, and the mode switch. Their positions are shown in Figure 4-18 Hand Controller Interface.

When the robot is off, you can press the power button to turn it on; when it is on, hold down the power button for a long time to turn it off.

In case of an emergency, press the emergency stop switch on the hand controller. The robot will be disabled, stop all movements and lock.

When the emergency stop button is pressed, it will be locked. To unlock it, rotate the button as indicated on it. Only after unlocking can the alarm be cleared through the control software, and then the enable switch can be turned on to restore from the emergency state.

The enable switch is a three-position switch. In manual mode: the robot can move only when the enable switch is in the middle position; when the enable switch is fully released or fully pressed, a Type 2 stop will be triggered.



warn

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

Chapter 5 Electrical Hardware and Installation

5.1 End Interface

The base of the robotic arm is equipped with a heavy-duty interface, the end of the robotic arm is fitted with buttons and indicator lights, and the side of the tool flange is provided with buttons, a screen and an aviation plug. As shown in Figure 5-1, an overview of the end interface.

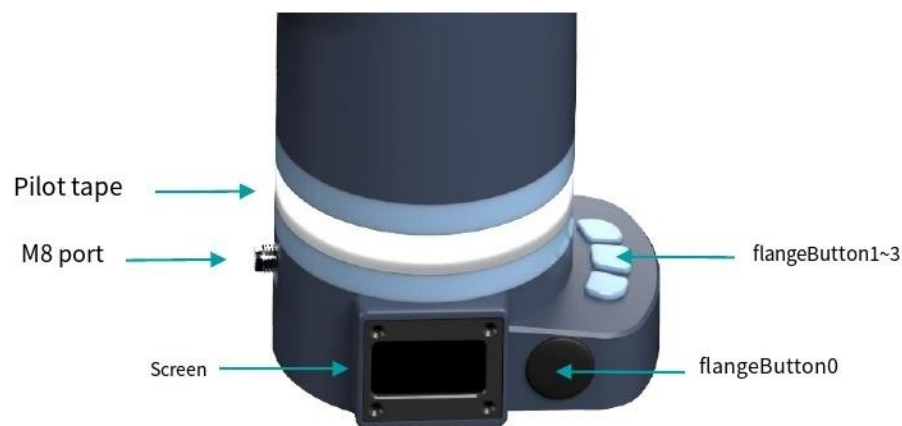


Figure 5-1 Overview of the Terminal Interface

5.1.1 Pro terminal interface

Interface	Description
M8	Power supply, input, output and communication IOs
Flange Button	User function customization buttons, free drag by default
Flange Button1~3	User function customization buttons to set up functions in the setup screen
Screen	Displays robot status, bus communication, inputs and outputs, RS485 baud rate, user-defined button status, etc.
Indicator light	Robot status indicator strip

5.1.2 Meaning of the indicator light strip

Color	Information
Blue stays on	Initialization in progress
White	Turned on but not powered up
Green	Manual mode
Yellow flashing	Auto-run mode
Red flashing	Robot error

5.1.3 M8 Interface

The M8 flange interface of the robot is located at the rear side of the end flange. The pin distribution and definitions are as follows.

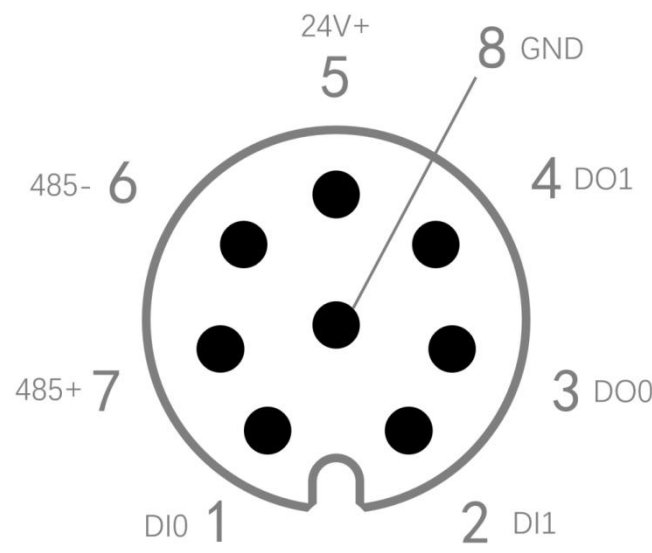


Figure 5-2 Pin Distribution of M8 Interface at the Flange End

Pin No.	Name	Definition
1	DI0	Digital input 0 (flangeDI 0)
2	DI1	Digital input 1 (flangeDI 1)
3	DO0	Digital output 0 (flangeDO 0)
4	DO1	Digital output 1 (flangeDO 1)
5	24V+	Positive 24V supply to external
6	485-	485 communication - (A) of MODBUS
7	485+	485 communication + (B) of MODBUS
8	GND	Flange internal ground; 24V power supply output negative terminal

The cable model for the M8 interface is Lumberg KKMV 8-354 or Lutronic FP-222460. It provides an external 24V power supply with a maximum current of 2A.

The digital output is of PNP type, capable of providing a maximum current of 5mA, and only offers level signals, which cannot be used to drive devices.

The digital input is configured as PNP type. When a switch is used as the DI input source, the wiring method is as follows.

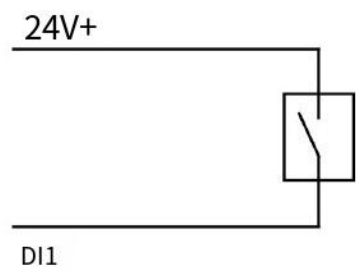


Figure 5-3 Wiring of PNP Type DI Switch at Flange End

5.2 Screen information

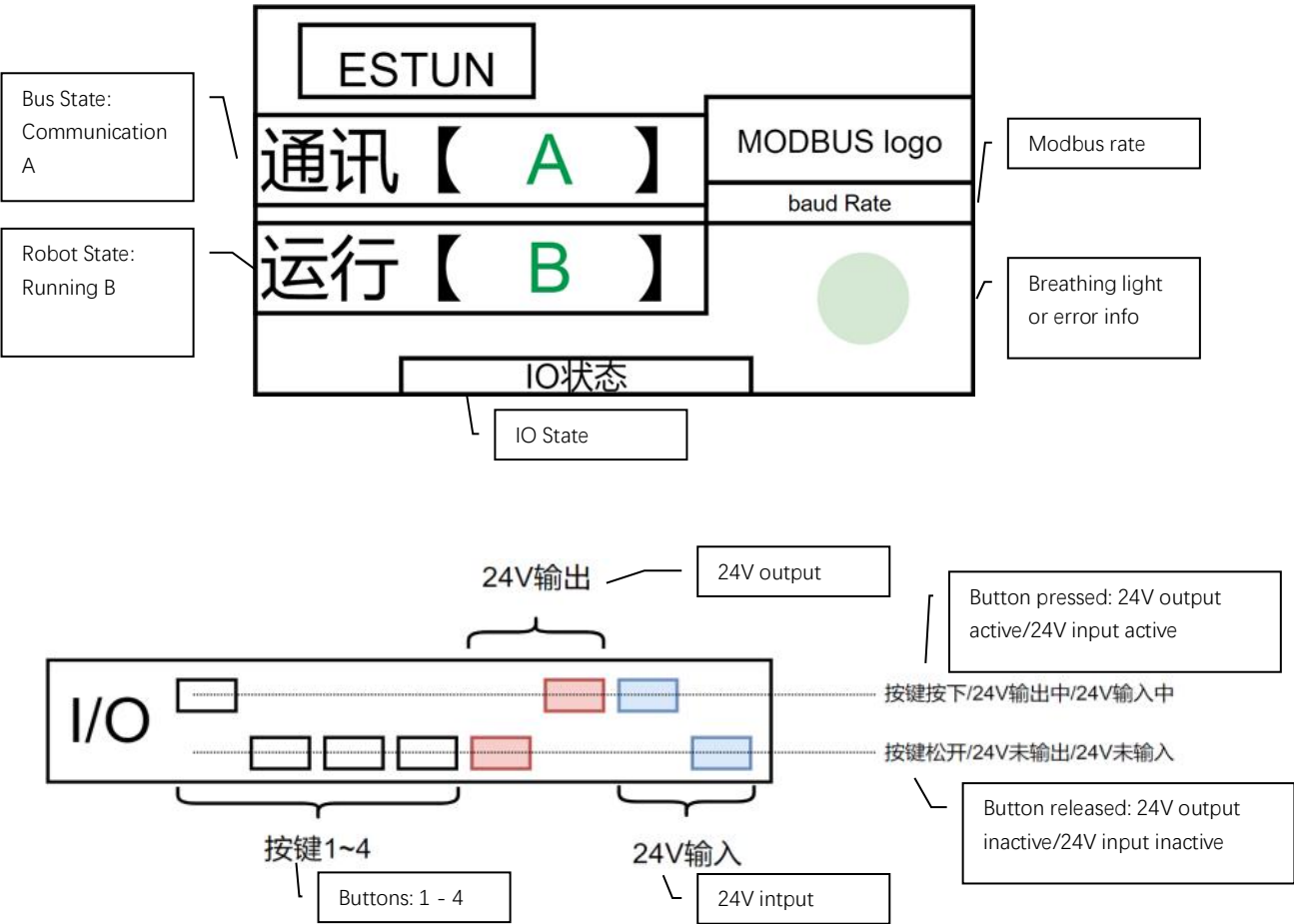


Figure 5-4 Screen Information at the End of the Flange

Info	Instruction
Bus state	<p>“Offline” (red): indicates that the EtherCAT communication state machine is in INIT, which usually occurs at startup or when the communication cable from the controller to the flange is disconnected somewhere.</p> <p>“Mailbox” (yellow): indicates that the EtherCAT communication state machine is in PreOP or BOOT, which occurs when the flange firmware has been updated or when the PDO has not been established.</p> <p>“Live” (green): the EtherCAT communication state machine is in SafeOP, OP, which means that the communication cable is connected properly and the communication with the controller is normal.</p> <p>“Error” (red): indicates that the EtherCAT communication state machine has unexpectedly switched from OP to INIT, which usually occurs when the cable is disconnected during communication, when the controller is suddenly powered down or during a soft reboot.</p>
Robot state	<p>“Normal” (green): the robot has no errors.</p> <p>“Error” (red): the robot is running with errors.</p>
IO state	A high cursor means the corresponding item is active; a low cursor means the corresponding item is not active.

MODBUS rate	MODBUS baud rates include 115200, 57600, 28400, 19200, 9600, 4800, 2400, 1200, 600, which can be configured through parameters.
Breathing light or error info	<p>Slow green blinking (2s): communication not fully established (INIT, BOOT, PreOP)</p> <p>Green fast blinking (0.5s): real-time communication is connected (SafeOP, OP)</p> <p>Red flashing (2s): communication abnormally disconnected OP->INIT</p> <p>Error message: the robot reports an error and displays an error alarm</p>

5.3 Control cabinet interface

There is only one power button on the front of the control cabinet. Press and hold it for a long time when the system is off to start the robot system, and press and hold it for a long time when the system is on to shut down the robot system.

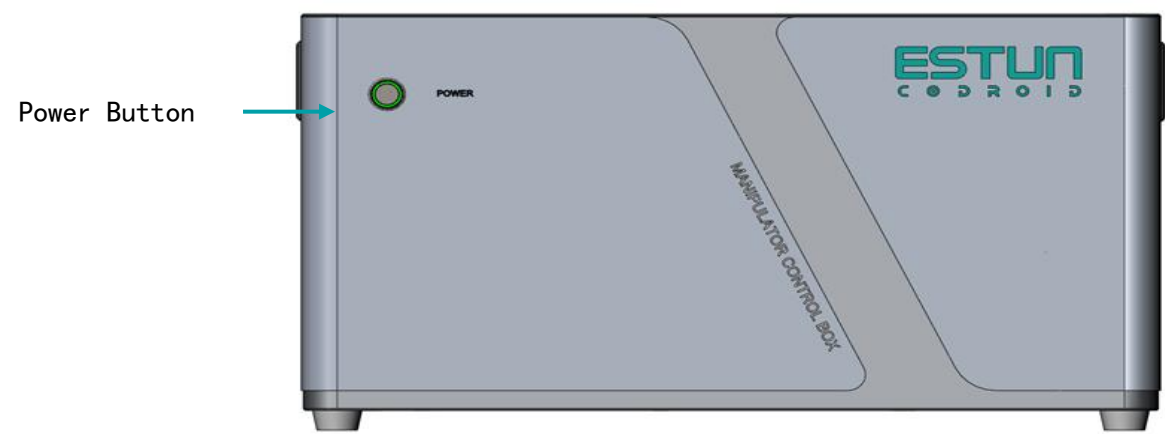


Figure 5-5 Power Button of Control Cabinet

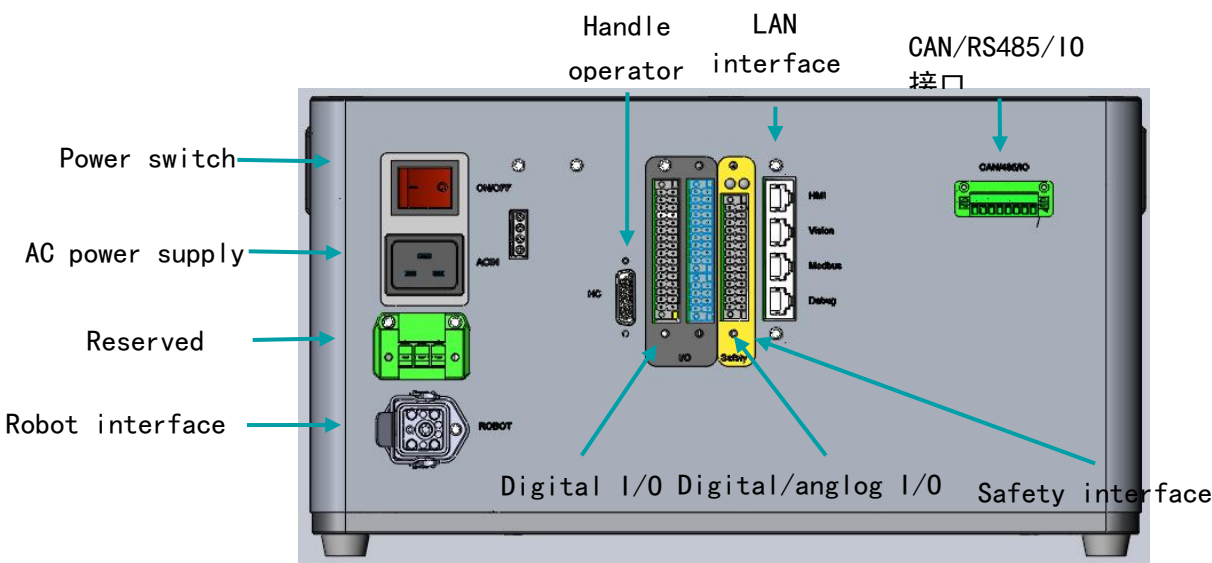


Figure 5-6 Overview of the Control Cabinet Interface

5.3.1 Overview of Electrical Interfaces

Interface	Instruction
AC power supply	For connection to AC100-240V 50/60Hz AC mains power supply
Power switch	Power supply switch
Handle operator	Handle operator
Robot aviation connector	Used to connect the robot to the control cabinet to provide power and communication to the robot.
LAN	For connection to teach pendant, vision, buses and development debugging
CAN/485/IO	CAN/RS485/IO
Safety port	Related ports for safety function
Analog/Digital I/O	Analog/Digital I/O
Digital I/O	Digital I/O

5.3.2 Safety Interface

The safety interface consists of 7 groups of safety dual-channel interfaces. The first to third groups are protective stop interfaces, and the fourth to seventh groups are emergency stop interfaces. The internal safety relays of the protective stop interfaces and the emergency stop interfaces are two independent channels. By default, the connectors are short-circuited horizontally with yellow short wires at the factory. Otherwise, the emergency stop state cannot be released.

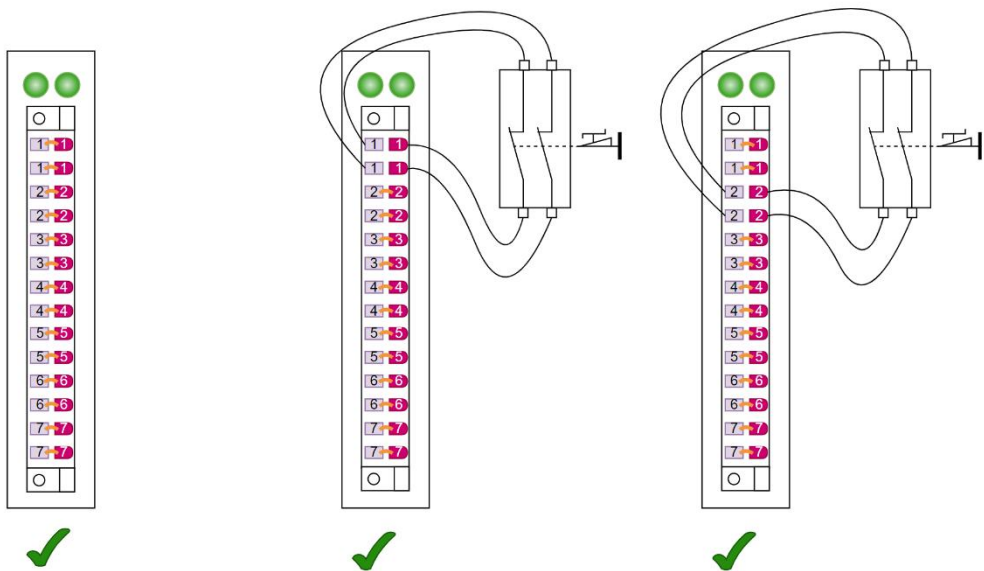


Figure 5-7-1 Correct Wiring Example for Safety Protective Stop

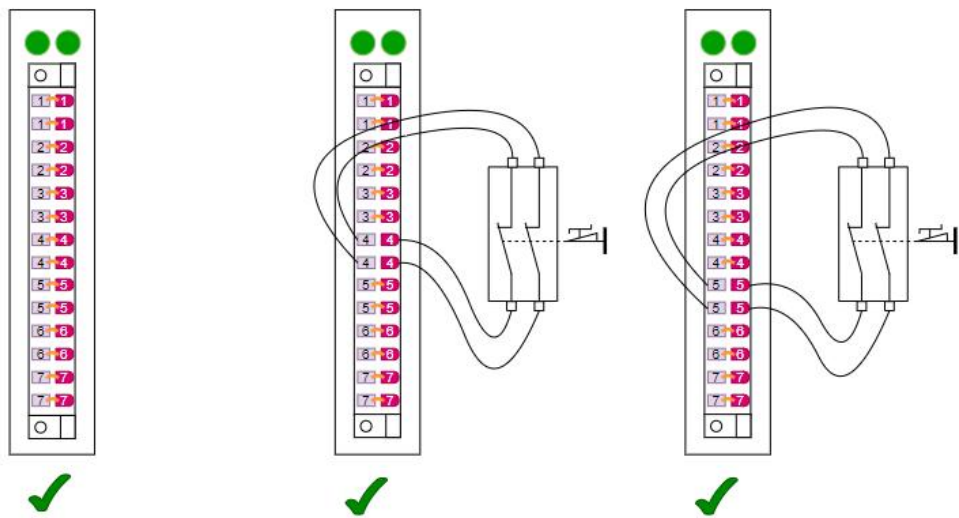


Figure 5-7-2 Correct Wiring Example for Safety Emergency Stop

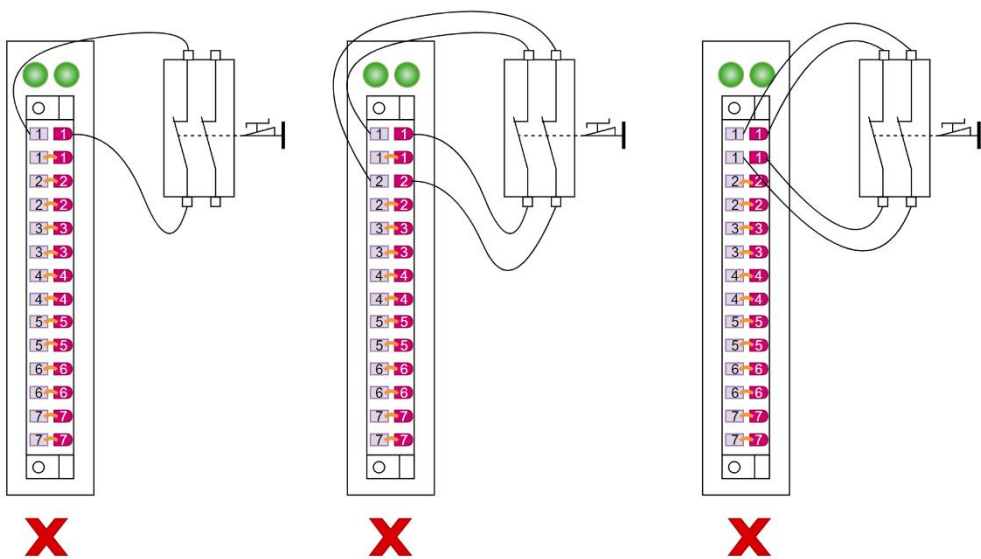


Figure 5-8 Typical Incorrect Wiring Examples of Safety Protective Stop and Safety Emergency Stop

5.3.3 General Input and Output Overview

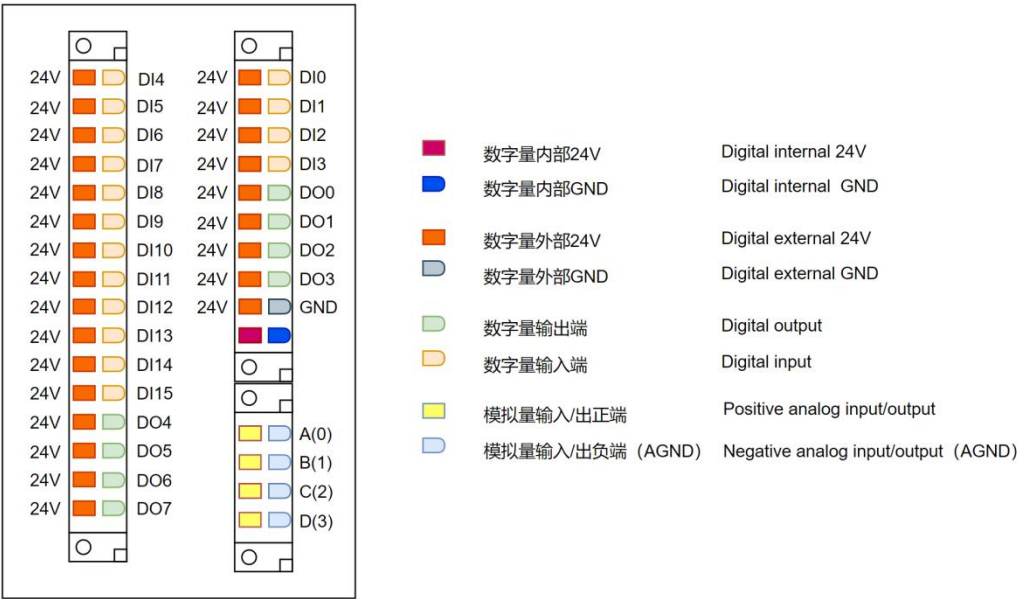


Figure 5-9 Functions of Each Terminal of the Input and Output Module

5.3.4 External power connection method for digital input

When entering digital signals, power supply to the port is required. Use an external power supply and connect the input end with a relay or a PNP-type digital loop. The wiring method is as follows for reference.

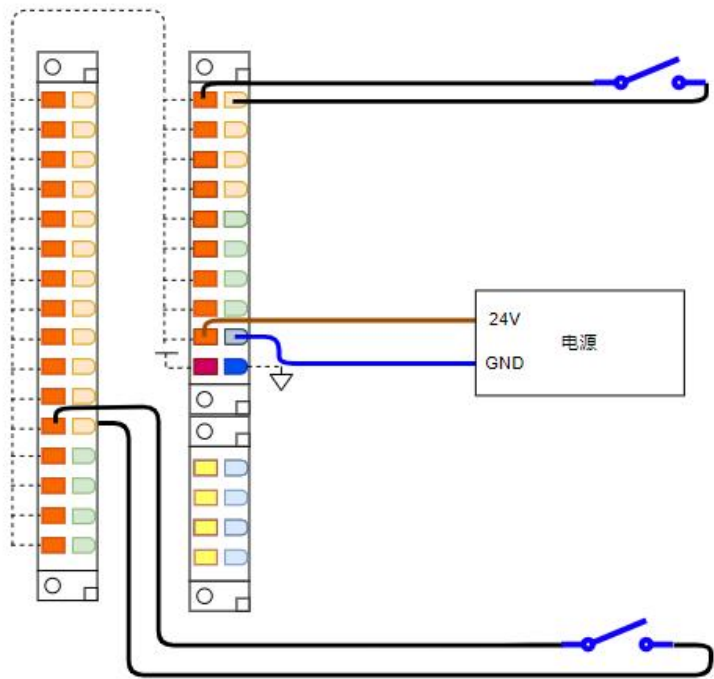


Figure 5-10 Example of Wiring for Relay Digital Input

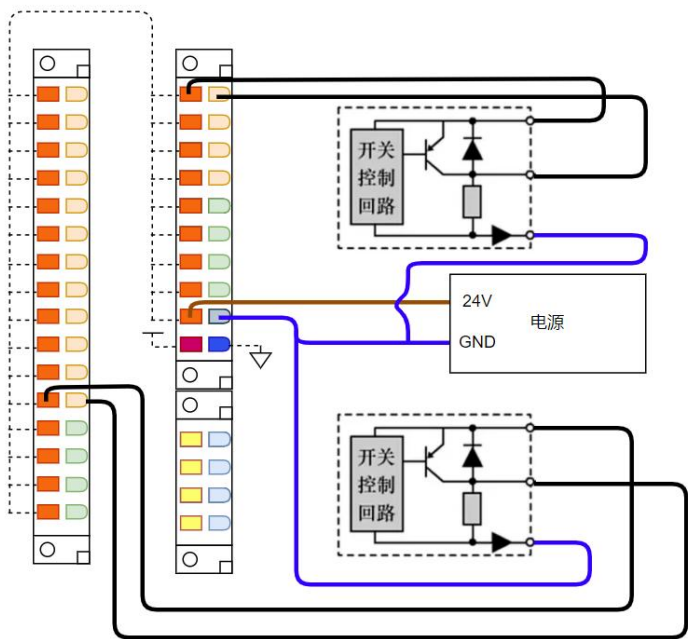


Figure 5-11 Example of Wiring for PNP-Type Digital Input

5.3.5 Internal power connection method for digital input

When the digital input port is powered, the internal power supply of the base plate can be used to power the port. The input end can be connected using a relay or a PNP-type digital loop. The wiring method is as follows:

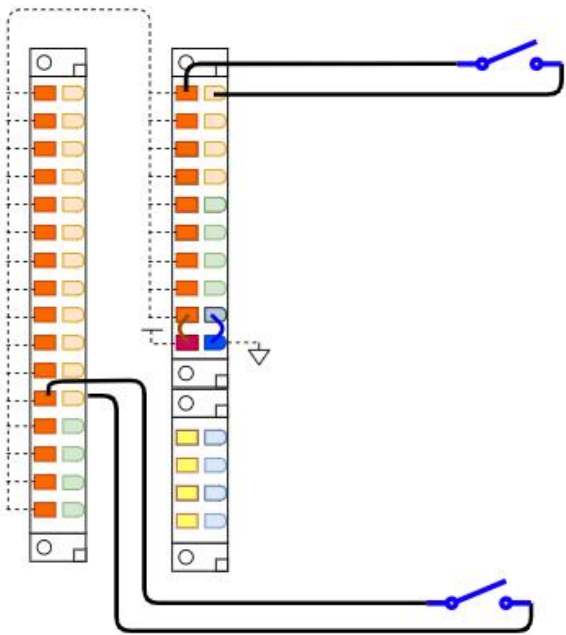


Figure 5-12 Example of Wiring for Relay Digital Input

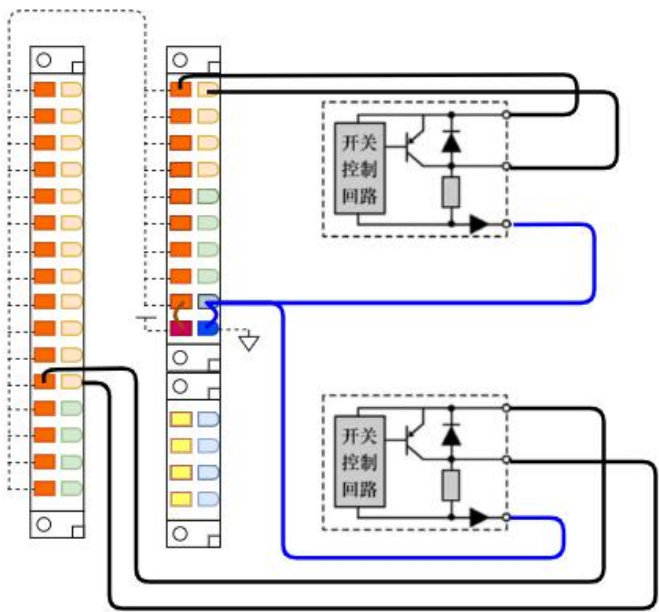


Figure 5-13 Typical Wiring Diagram for External Power Supply Provision

External power supply for digital inputs	Parameter
External supply voltage	Typical 24V
Maximum Output Current	Maximum output 5A per group

5.3.6 External power connection method for digital output

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

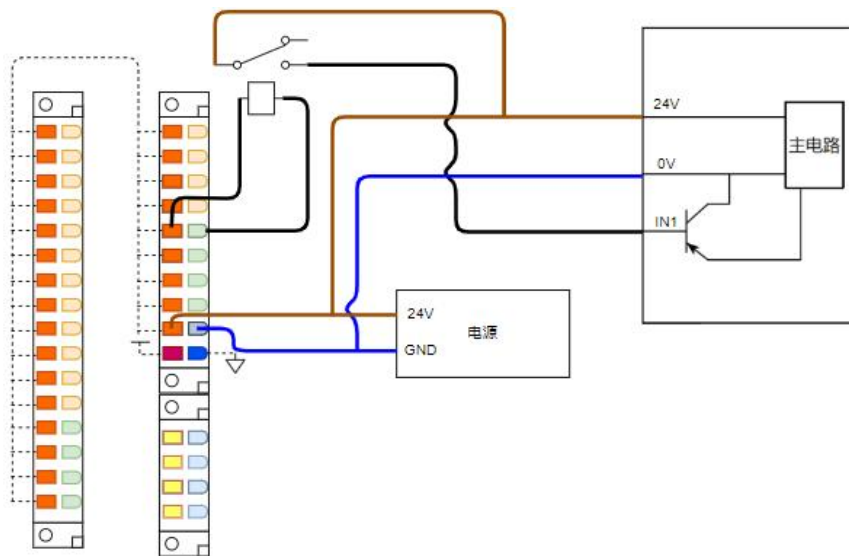


Figure 5-14 Example of Wiring for Digital Output Using a Relay

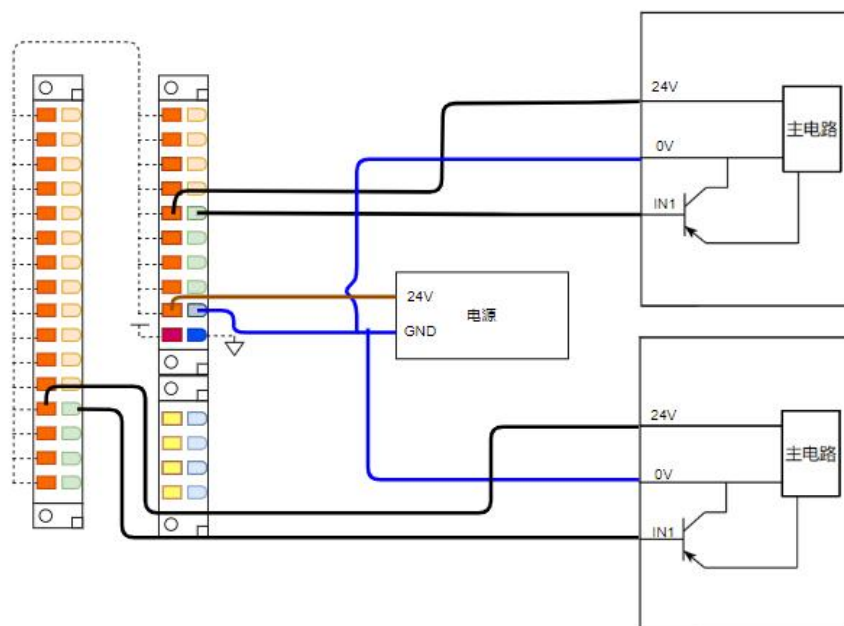


Figure 5-15 Example of Wiring for Digital Output Using PNP Circuit Type

5.3.7 Internal power connection method for digital output

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

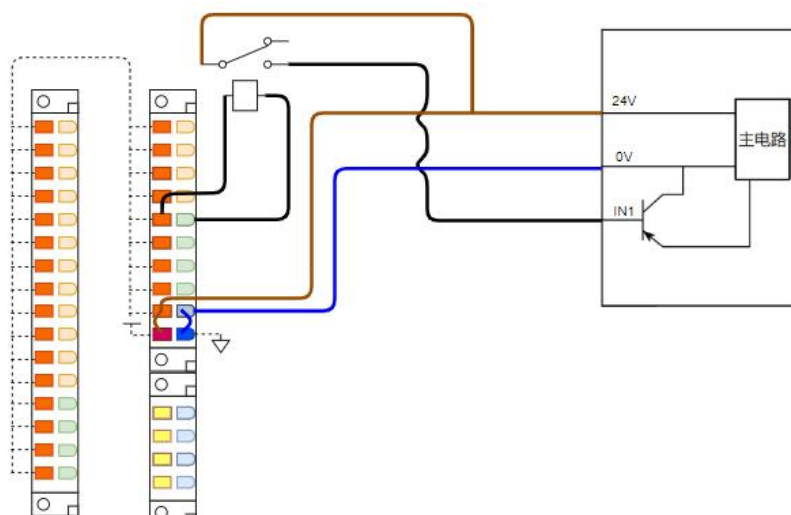


Figure 5-16 Example of Wiring for Digital Output Using Relays

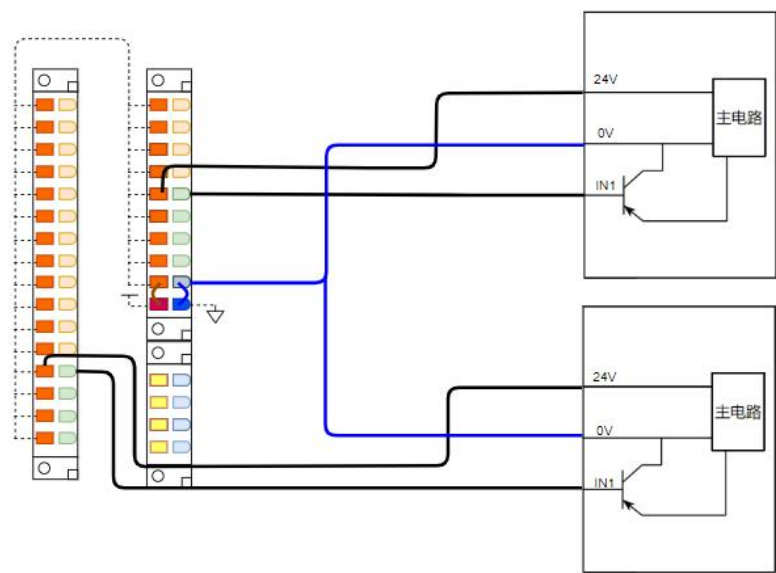


Figure 5-17 Example of Wiring for Digital Output Using PNP Circuit Type

digital output	Parameter
Interface type	PNP
Output voltage	Typical 24V; Maximum 30V
Max output current	125mA max. for single group

5.3.8 Simulation input/output interface

The analog input supports both voltage-type and current-type sensors. The input type of voltage or current needs to be set in the robot settings options; the analog output only supports current type.



Warn
The analog output port must be connected to a load; otherwise, the robot will report an error. You can turn off the corresponding analog output port from the robot's operation interface.

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

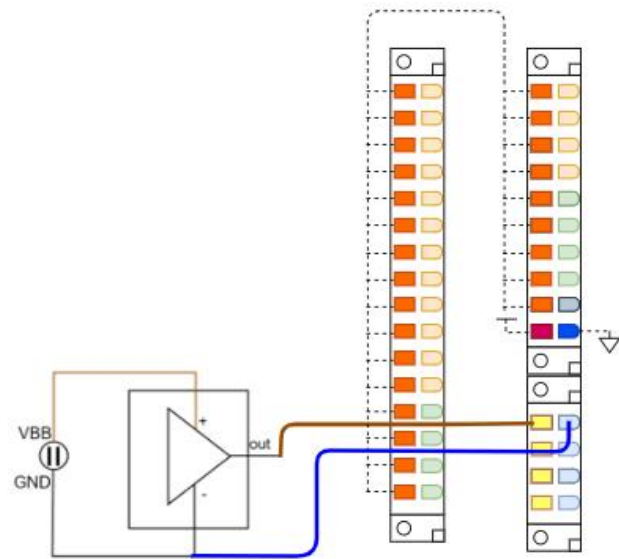


Figure 5-18 Correct Wiring Example for Analog Voltage Input

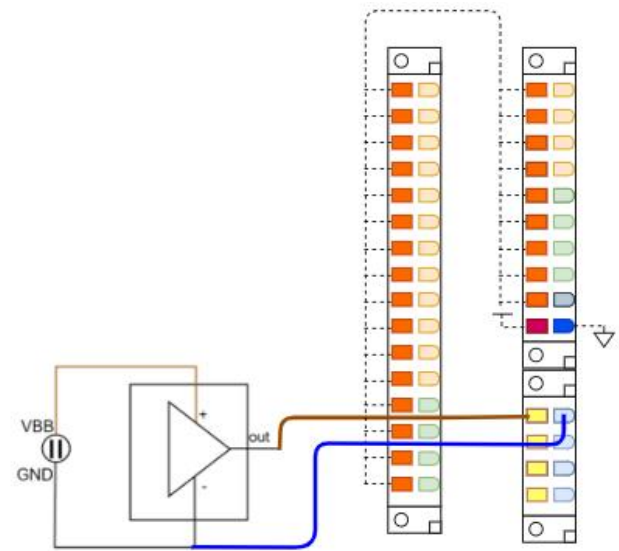


Figure 5-19 Correct Wiring Example for Analog Current Input Type

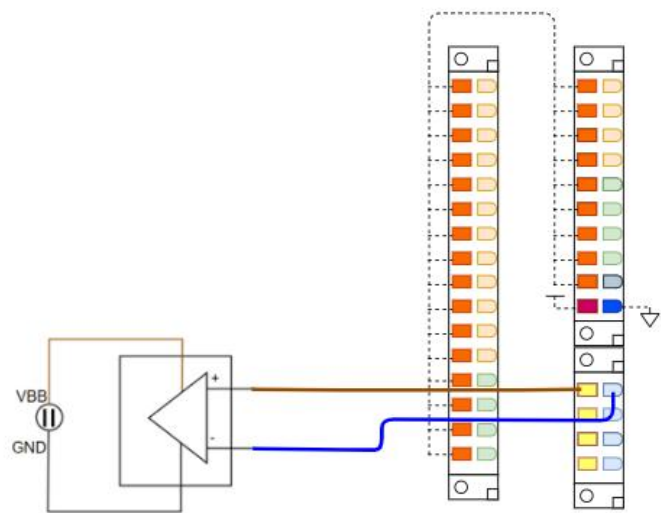


Figure 5-20 Correct Wiring Example for Analog Current Output

Analog Inputs	Parameter
Resolution	12bit
Measurement range	In voltage mode: 0-10V; In current mode: 4-20mA;

Input Impedance	In current mode: 20Ω;
-----------------	-----------------------

Analog Outputs	Parameter
Resolution	12bit
Onput Impedance	Current mode: 4 - 20mA;

5.3.9 CAN/485/IO interface

The pin definitions of the interfaces including CAN, 485 and IO on the control cabinet are as follows:

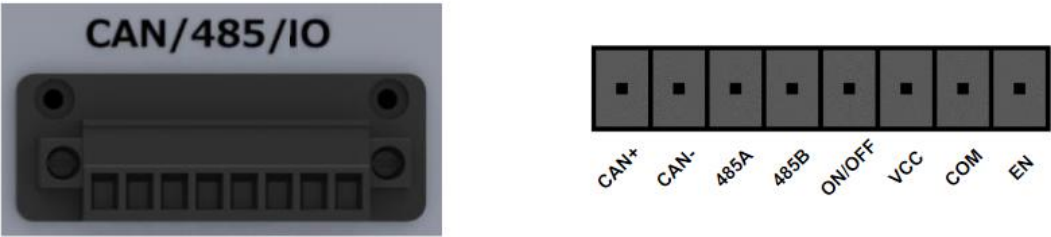


Figure 5-21 Definition of CAN/485/IO Interface

Ports	Instruction
CAN+	CAN+
CAN-	CAN-
485A	485A/485+
485B	485B/485-
ON/OFF	External start/stop button
VCC	Start-stop signal transmission
COM	Start-stop signal receiving
EN	Internal start/stop button

Different wiring methods for power supply on and off:

Method ①: Use the power button on the control cabinet and hand controller to turn on and off the machine.

(Short-circuit the COM and EN interfaces with a jumper wire.)

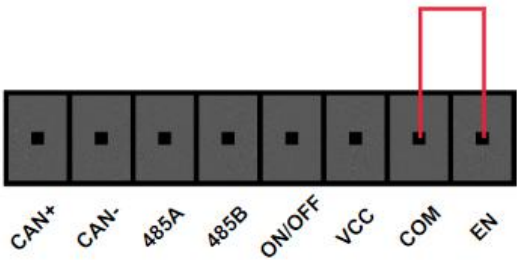


Figure 5-22 Wiring Diagram for Power Supply Startup Method ①

Method ②: External power supply start/stop button

Short the COM and EN interfaces with a jumper wire, and connect the ON/OFF and VCC interfaces to an external self-resetting normally open stop button.

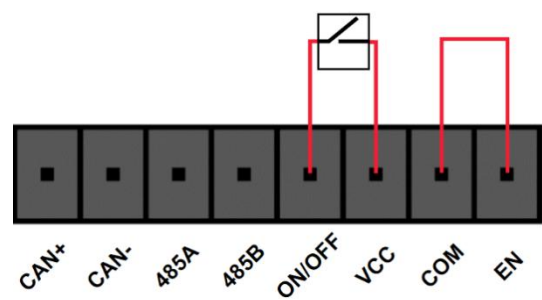


Figure 5-23 Wiring Diagram for Power Supply Startup Method ②

Method ③: Self-starting after the control cabinet is powered on

(Short-circuit the VCC and 24VEN interfaces with a jumper wire.)

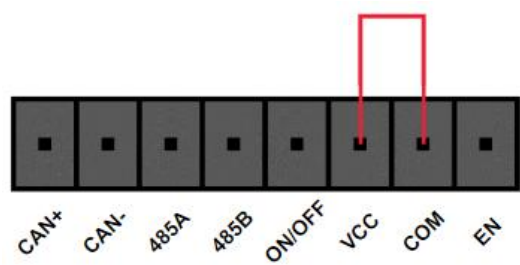


Figure 5-24 Wiring Diagram for Power Supply Startup Method ③

5.3.10 LAN Network Port

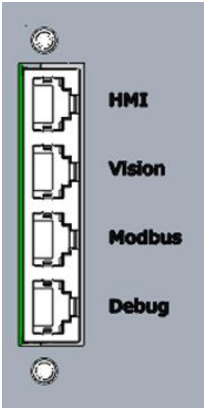


Figure 5-25 Network Interface

LAN ports	Instruction
HMI	Connect to a demonstrator or tablet. Direct connection to internal router, router connected to keba's ETH0 port.
Vision	Connecting Visuals. Directly connect to the internal router, which is connected to the keba's ETH0 port.
Modbus	Bus connection port. Directly connect to the internal router, which is connected to the keba's ETH0 port.

Debug	Debugging, socket port. Direct connection to internal keba controller ETH1 port.
-------	--

5.3.11 Communication input

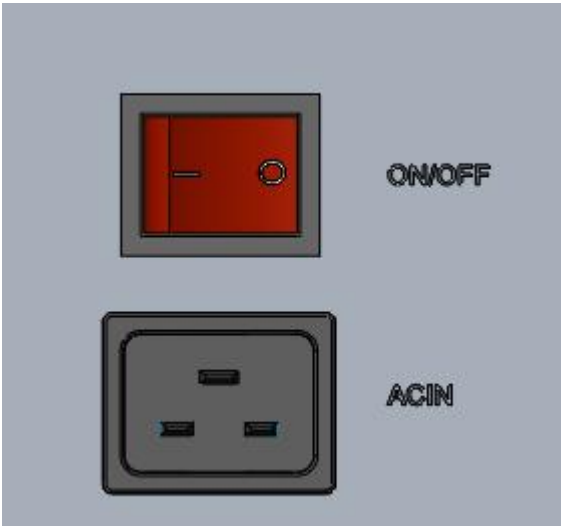


Figure 5-26 AC Input Interface and Switch

The AC input range is: AC100~240 V 50/60 Hz. AC power supply and DC power supply cannot be connected simultaneously. When in use, a magnetic ring needs to be placed on the ACIN power cord to eliminate EMC interference.

Chapter 6 Maintenance and Warranty

6.1 Notes

- Maintenance work can only be carried out by Codroid or authorized system integrators.
- Always perform any visual or workplace inspections for maintenance or repair in accordance with all safety instructions in this manual.
- The change control system and robot joints require recalibration of the robot. The calibration operation and result judgment method are described in the zero-point verification manual. Also, the parameter settings need to be checked. If there is a parameter backup, it can be imported. If not, the parameters need to be reset.

When operating the robot body or control cabinet, the following safety tasks must be followed:

- Remove the main input cable from the back of the control cabinet to ensure that the system is completely de-energized. Necessary precautions should be taken to prevent others from re-energizing the system during maintenance. After de-energizing, re-check the system to ensure it is de-energized.
- Please check the grounding connection before restarting the system.
- When disassembling the robot body or control cabinet, please comply with the ESD (Electrostatic Discharge) regulations.
- Avoid disassembling the power supply system of the control cabinet. Even after the control cabinet is turned off, its power supply system can still retain high voltage for several hours.
- Avoid water or dust from entering the robot body or control cabinet.

6.2 Daily inspection items

6.2.1 General cleaning

If dust/dirt/motor oil is observed on the controller or the robotic arm, it can be wiped clean with a cloth dampened with a cleaning agent. Cleaning agents: water, isopropyl alcohol, 10% ethanol or 10% naphtha.

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

Do not use compressed air to clean the controller or the mechanical arm; otherwise, the seals and internal components may be damaged.

6.2.2 Control box

Inspection plan

Inspection item	Method	Monthly	Semi-annual	Annual
Emergency stop button for handle operator	Functional test	X		
Free Drive Mode	Functional test		X	
Safety inputs and outputs	Functional test	X		
Teach pendant cables and adapters	Visual inspection		X	
Terminals on the control box	Functional test		X	
Control cabinet main power and switches	Functional test			X

Highlight the safety features of the robot and recommend monthly testing to ensure proper functioning.

The following tests must be carried out:

6.2.2.1 Test the emergency stop button on the handle operator

- Press the emergency stop button.
- Observe the robot stop and then turn off the power supply of the joints.
- Restart the robot again.

6.2.2.2 Test free drag mode

- According to the tool specifications, remove the accessory devices or set the TCP/load.
- Hold down the free drag button at the end of the robot to set the robot to free drag mode.
- Move the robot to a position where it is horizontally extended to the edge of its workspace.
- While holding down the free drag button, monitor the robot to maintain its position without support.

6.2.2.3 Test safe input and output

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

6.2.2.4 Visual inspection

- Unplug the power cord from the controller.
- Check if the terminals are correctly inserted and if the wires are loose.

- Check if the network cable inside the controller is loose.
- Check if there is any dirt/dust inside the controller. If necessary, clean it with a vacuum cleaner that prevents static discharge.

6.2.3 Robot

Inspection plan

Inspection item	Method	Monthly	Semi-annual	Annual
Check the joint cover	Visual inspection		X	
Check cover screws	Functional test		X	
Inspection of flat rings	Visual inspection		X	
Check robot cables and connections	Visual inspection		X	
Checking the robotic arm mounting bolts	Functional test	X		
Check tool mounting bolts	Functional test	X		
Check the screws connecting the joints	Functional test		X	

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

6.3 System update

This chapter explains how to update the CoDroid robot software. The information in this manual is accurate at the time of writing. Users will not be notified in advance of updates for subsequent products.

Before starting the update, please confirm the following update precautions.

Please ensure that the power supply will not be turned off or cut during the update.

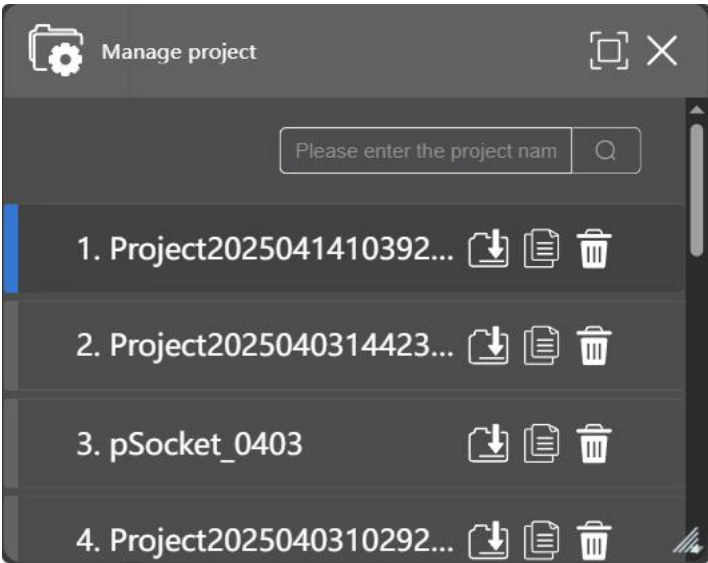
Confirm that the correct version of the update compressed file has been obtained.

All the programs of the robot have been backed up.

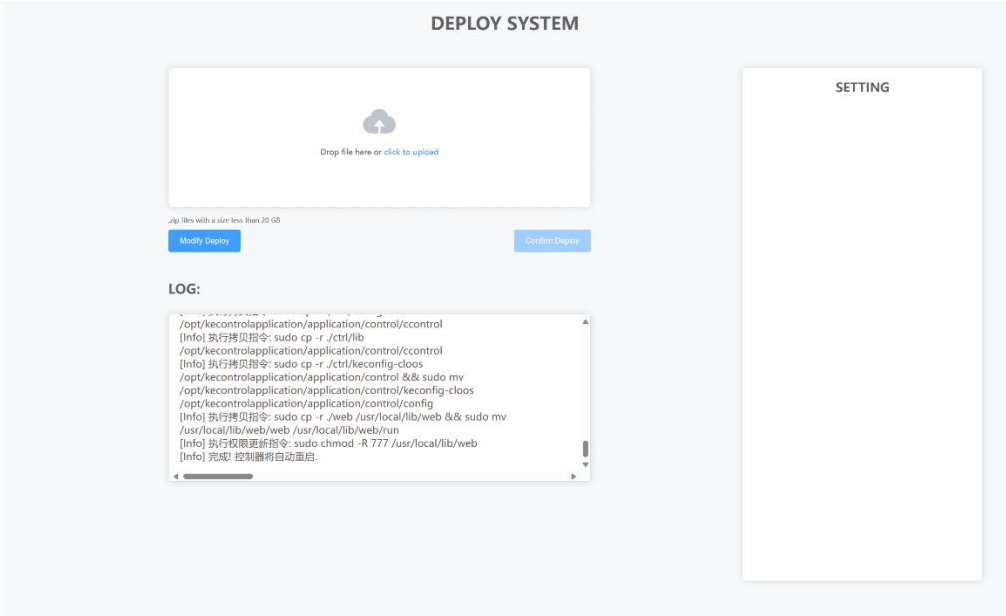
Before updating, please check the release notes of the version you are updating to. For detailed information, contact CoDroid technicians.

6.3.1 Update steps

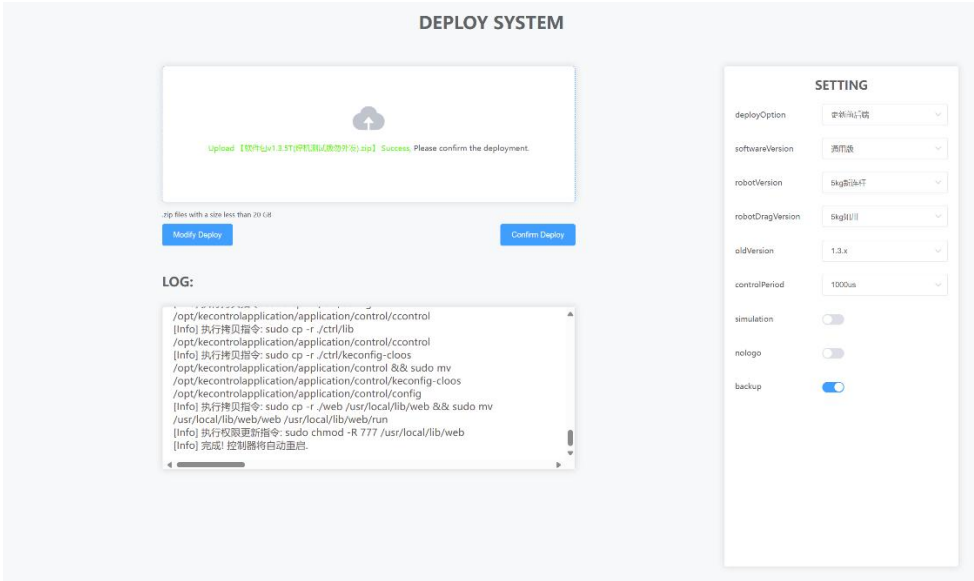
After starting up, enter the robot control platform, go to the project tab, click on the project management interface, and select the program to be backed up for downloading to perform the program backup.



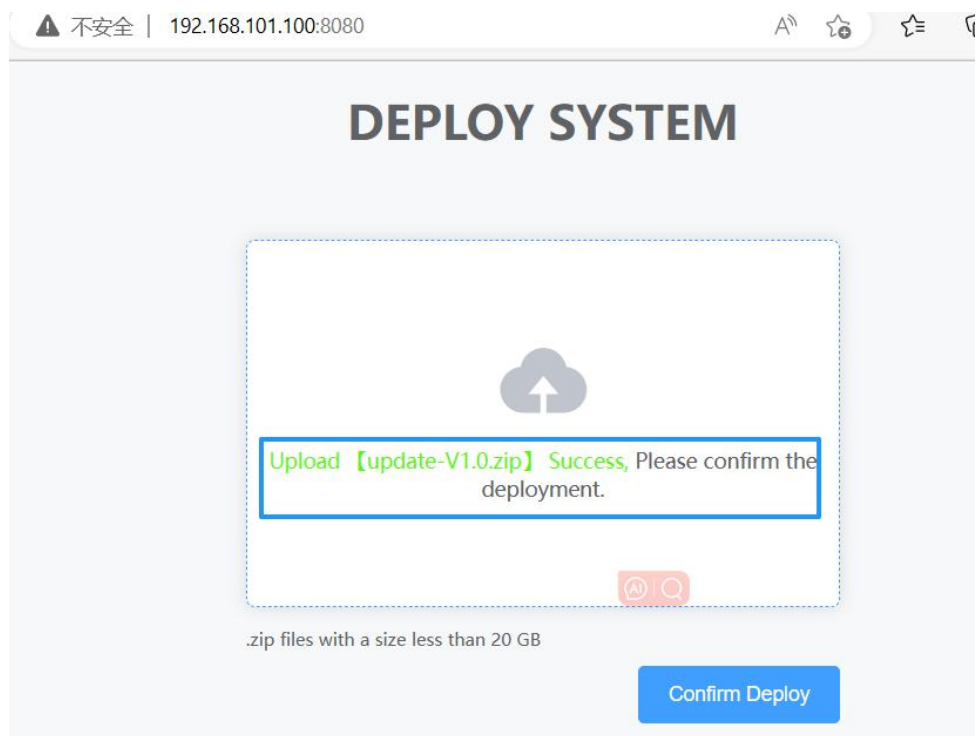
- 2. Switch the robot to the "power-off" state and press the emergency stop button.
- 3. Click on the system version number at the lower right corner of the page to enter the update interface.



- 4. Drag the update file into the file selection box, or click the 'click to upload' button to select the file you need to update and wait for the upload to complete.
- 5. Select the appropriate options based on the model requirements.



6. After confirming the update, wait for the robot software to restart automatically. The update is complete once the restart is finished.



6.4 Common Mistakes

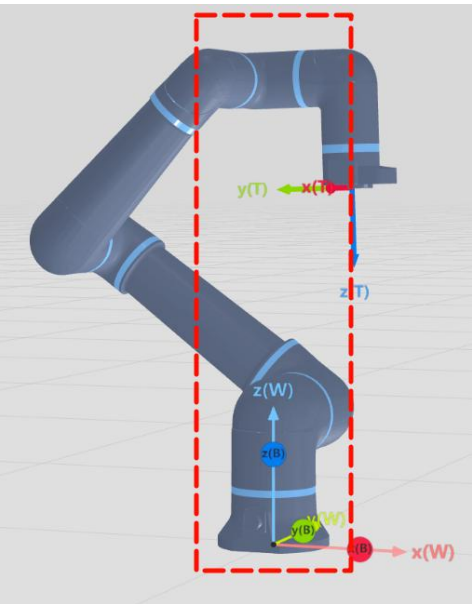
This section lists some common errors that may occur during the use of the robot. If you encounter other errors that cannot be resolved, you can download the robot log file in the log interface and send it to the after-sales personnel for analysis and processing.

6.4.1 Singularity/Inverse solution failure

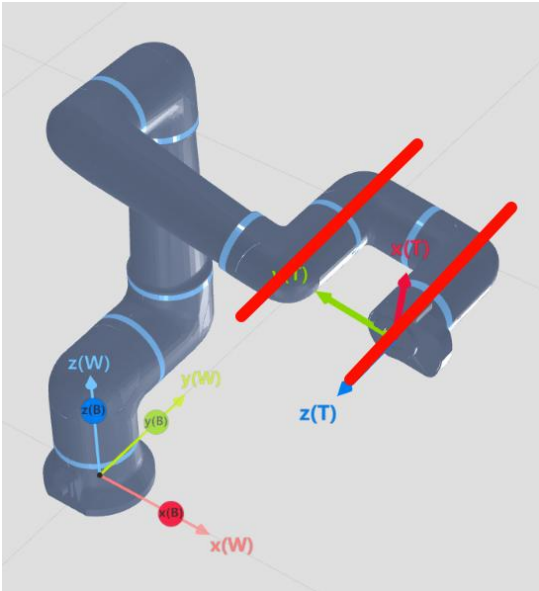
The working range of a robot is a spherical space with the arm's reach as the radius. However, there are some special positions and postures that are singular points for the robot, and these should be avoided during operation.

The following are three typical types of singularities:

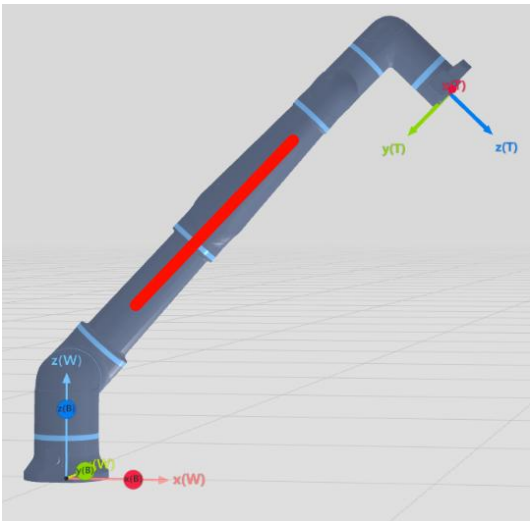
A cylindrical area with the base of the robot's pedestal as the bottom surface;



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



When the angle between the robot's upper arm and lower arm approaches 180°,



6.4.2 Trigger collision detection

The torque sensors in the robot joints will detect the force exerted on the robot in real time. When the force exceeds the expected value, a collision detection will be triggered. At this point, it is necessary to confirm whether the robot's movement trajectory is correct

and whether there is anything obstructing the robot's movement.

If the robot's motion trajectory is correct but the collision detection is still triggered, it is necessary to check whether the tool is set correctly, whether the load is set correctly, and whether the pipeline of the end tool is normal, etc.

6.4.3 Location/Speed Exceedance

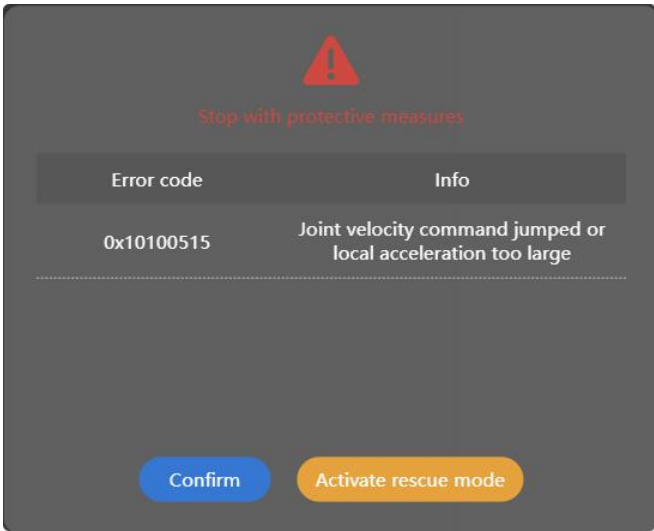
When the robot exceeds the position or speed limit during operation, check whether the program is correctly written. If it is correct, you can modify the corresponding parameter limit in the safety settings of the settings.

If a position limit error occurs and the robot remains in an over-limit state even after the error is cleared, and it still alarms upon re-powering, the rescue mode can be enabled to adjust the robot to an appropriate posture.




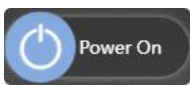
6.4.4 Joint tracking error is too large

When excessive joint tracking errors occur during the robot's movement, it is necessary to check whether the movement speed and acceleration are reasonable, and whether the robot's load is correct and within the robot's load capacity.

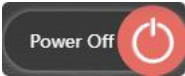
6.4.5 Alarm cleared



When an alarm pop-up window appears, you can directly activate the rescue mode or click "OK" and then manually reset it to enter the rescue mode. The steps to clear the alarm in the rescue mode are as follows:

- a) click , click  to clear the error report;
- b) Click , to turn on the rescue mode;
- c) Tap  to power up the robot. 4;
- d) In rescue mode, rotate the overrun joints to the correct position by tapping the

joints;

e) Click  to shut down the robot. 6;

f) Tap  to exit rescue mode;

g) Repeat step 3 to power up the robot.

6.5 Fault code description

Currently, there are a total of 6 information levels for the robot. The fourth digit of the error code indicates the error level.

No.	Error & Level
0	System occupancy
1	System prompt
2	Alert
3	General Error
4	Critical error
5	Fatal error

- When general errors or more serious issues occur, the robot will power off and stop operating.
- When a warning-level error occurs, the robot will slow down and stop.
- If multiple errors occur at the same time, the one with the highest severity level will be executed.
- There will only be one error code for the same type of error, but the content of the error will be specifically displayed on the demonstrator
- For specific error codes and details, please refer to the appendix.

6.6 Disclaimer

Estun Codroid is committed to creating a harmonious future where humans and machines coexist. While continuously enhancing the reliability and performance of our products, we reserve the right to upgrade them without prior notice. Estun Codroid strives to ensure the accuracy and reliability of the information in this manual, but assumes no responsibility for any errors or omissions.

The following situations resulting in malfunctions are not covered by this warranty:

- Installation, wiring, and connection to other control devices were not carried out in accordance with the requirements of the user manual;
- Use beyond the specifications or standards indicated in the user manual;
- Product damage caused by improper transportation or use;
- Damage caused by accidents or collisions;
- Natural disasters such as fire, earthquake, tsunami, lightning strike, strong wind and

flood;

- Modifications to system software or internal data;
- Use of this product in radioactive equipment, biological testing equipment or for hazardous purposes;
- The production date or the start date of the warranty cannot be identified.
- Faults not caused by Nanjing Estun Codroid Co., Ltd. other than the above situations.

6.7 Abandoned robots

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

Chapter 7 Overview of the Teaching Pendant

Interface

7.1 Login interface

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

Clicking the "Clear Cache" button can clear the browser cache. It is recommended to clear the cache when switching the connected robot.

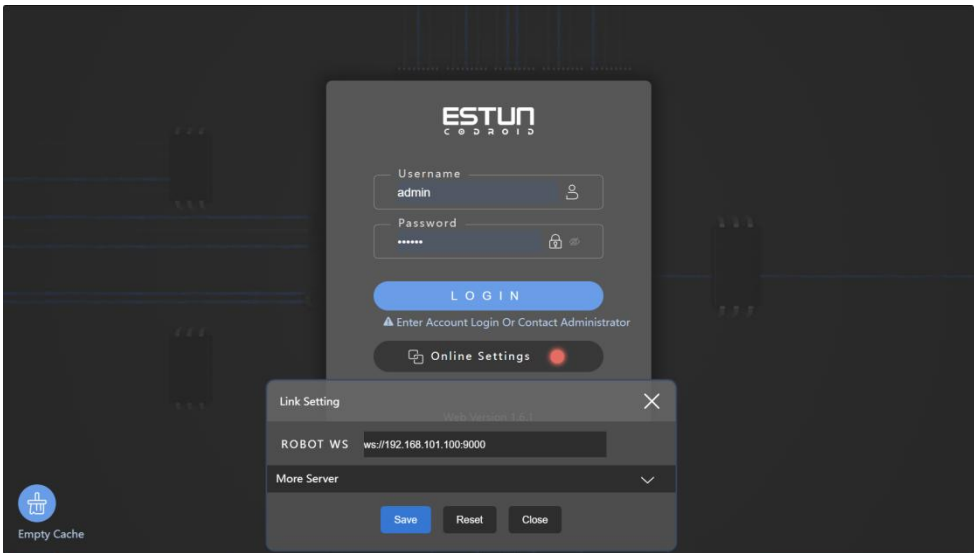


Figure 7-1 Login Interface

7.2 Home page

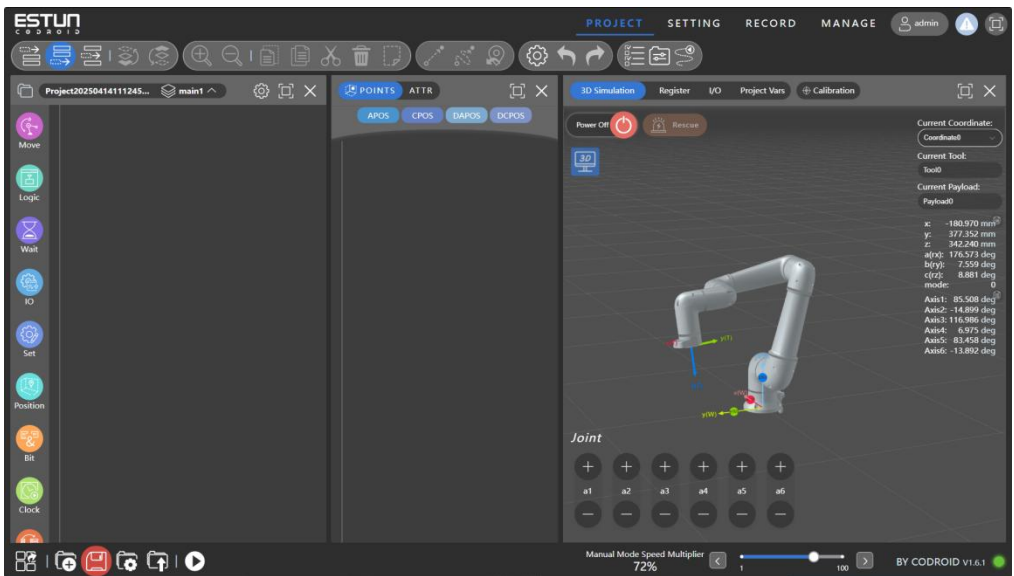
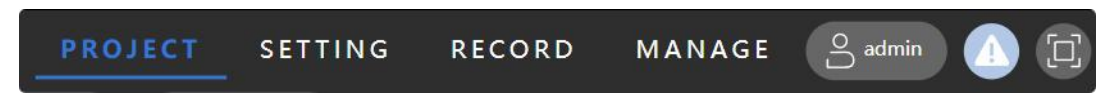


Figure 7-2 Main Interface

After successfully logging in, you will be redirected to the main interface, which displays the contents of "Project" tab by default and is divided into 4 operable areas:

7.2.1 Switch tab area



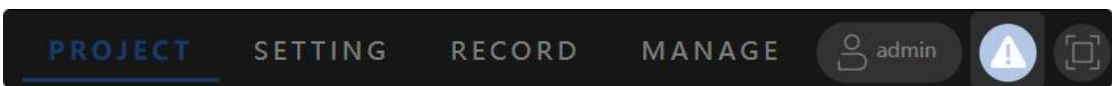
It includes four buttons: "Engineering", "Settings", "Logs", and "Management", which respectively switch to four different display interfaces.

7.2.2 Account Settings Button

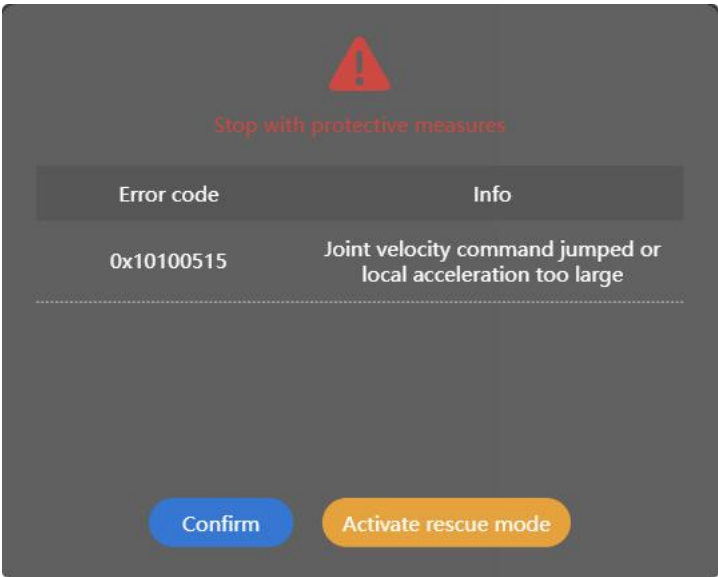


The button displays the currently logged-in account. Clicking it leads to a "Re-login" option that redirects to the password interface.

7.2.3 Error message and real-time log window button



Pop up the error message and real-time log window.



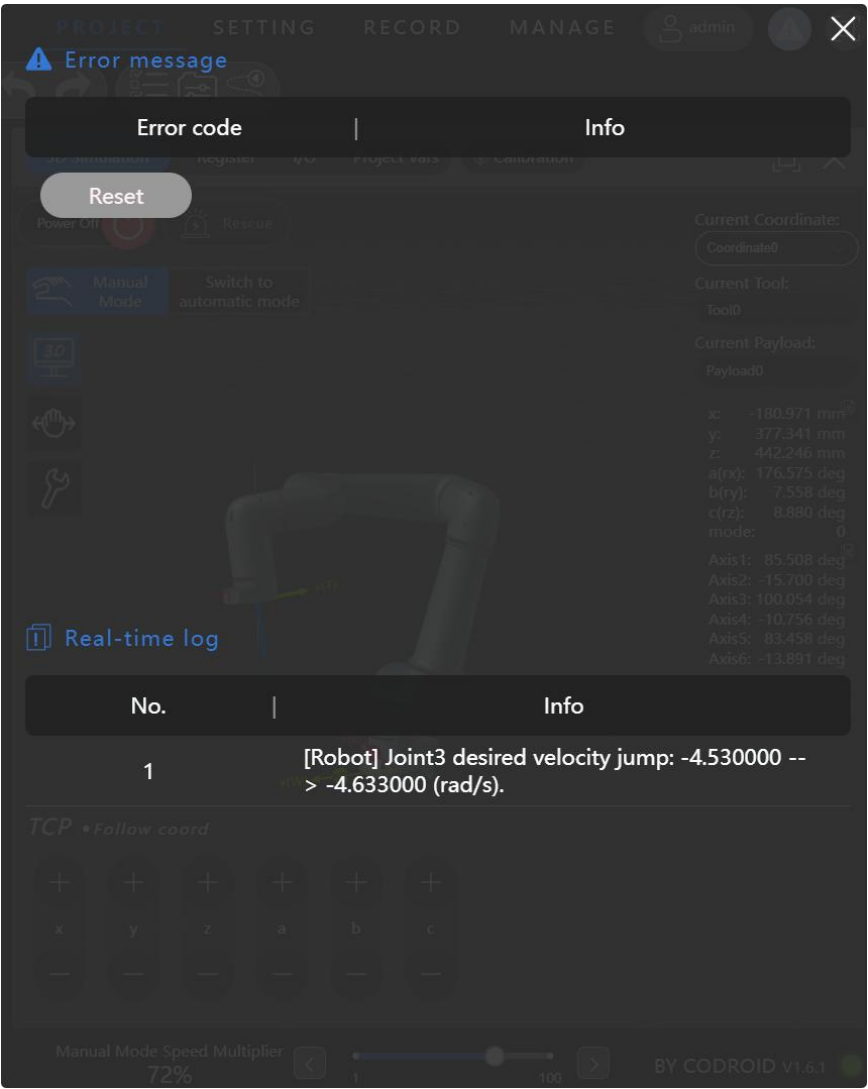


Figure 7-3 Error Messages and Real-time Logs

When the robot reports an error, an error message is displayed here, containing the time of the error, the error code, and a description of the error message. After confirming that the robot's fault status is clear, you can press the “Reset” button to clear the error message, and the robot can be powered up again after the error status is cleared.

7.2.4 Full-screen display button



Used to switch between full-screen and non-full-screen mode of the web page (full-screen display is recommended).

7.3 Project Tab

In the Project tab, the main areas include the menu area, graphic programming area, pose list, variable list, parameter area, 3D display area, IO area, project management area, and speed multiplier adjustment area.

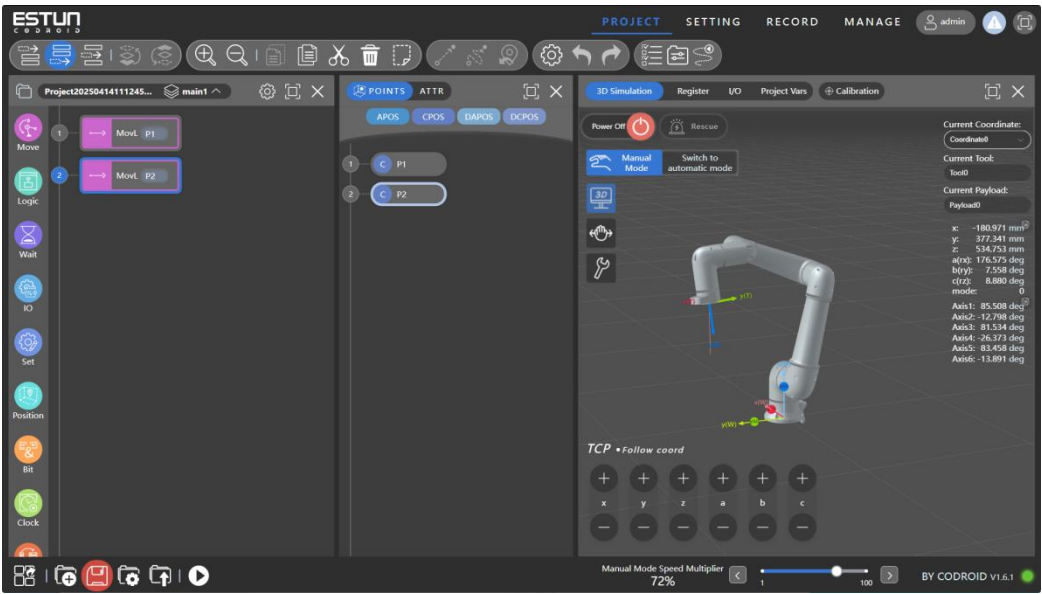















Figure 7-4 Layout of the Project Interface


7.3.1 Quick operation area


-  Insert instructions above;
-  Insert instructions below;
-  Insert instructions internally;
-  Move the command up one level;
-  Move the command down one level;
-  Zoom in on the program tree area;
-  Reduce the program tree area;
-  Paste command;
-  Copy the selected command;
-  Cut the selected command;
-  Delete the selected command;
-  Annotate the selected command;
-  Linear movement to the point

 Joint movement to the point

 Update Points;


 Project Settings

 Undo the current operation

 Redo the current operation

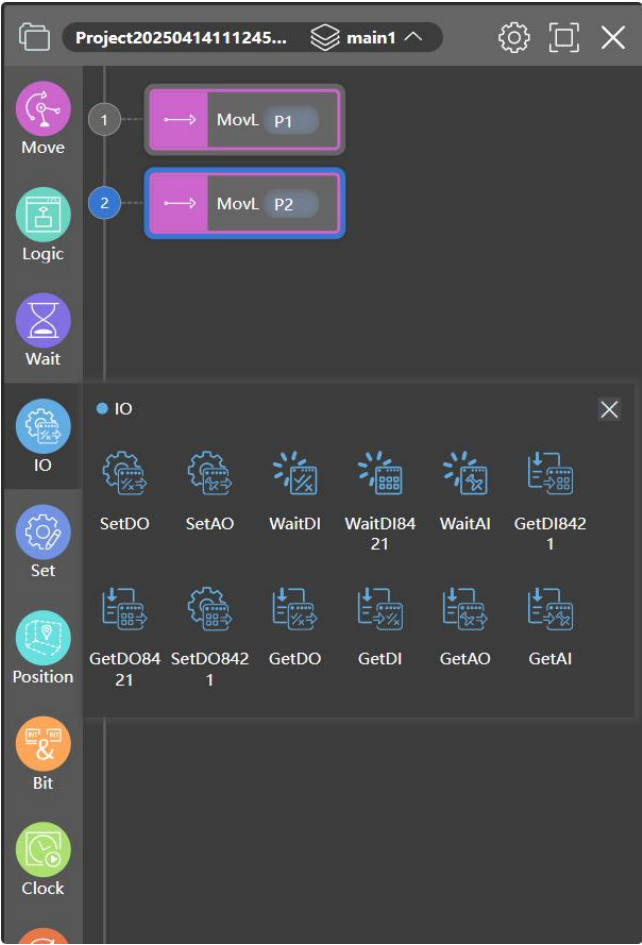


Open the batch management operation; after opening it, you can manage the project, program tree and point list in batch. You can select a command individually or select all or reverse selection, and then you can do copy, paste, delete and other operations on

it;  Open the Variable Management screen;

 Track record

7.3.2 Graphics Programming Area



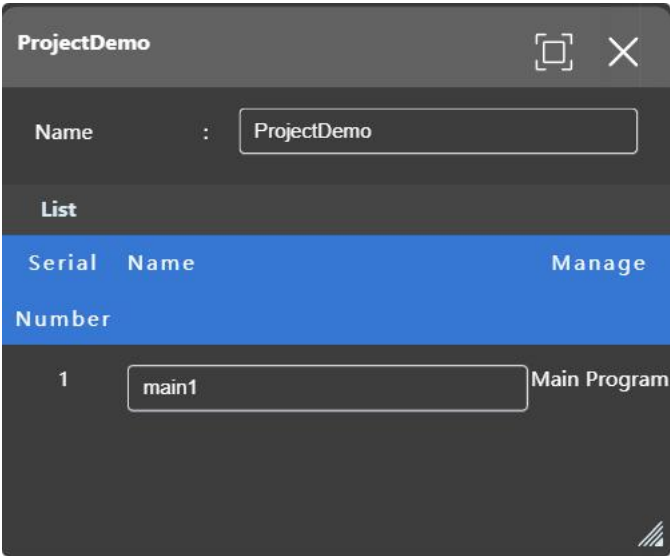
The graphic programming area can be divided into four parts, namely: the title area, the instruction classification area for programming, the instruction area for programming, and the program tree area.

7.3.2.1 Title Area

There are three buttons in the title area, namely:



Project property editing and task management:



- Full screen / Restore window, Close window
- Name the current project.
- Switch between multiple programs / (a single) program
- Program management in multiple programs (subroutine naming, adding new subroutines, deleting subroutines), with a maximum support of 30 multiple programs in one project.



Full Screen/Restore Window: Full screen/restore the “Graphics Programming Area” display.




Close the Graphics Programming Area window.



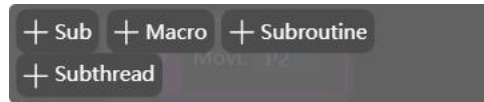
After closing the “Graphic Programming Area”, you can click on the “Visual Programming” button A to restore the display.

7.3.2.2 Multitasking

The robot supports multi-tasking programs. Different types of tasks can be added by

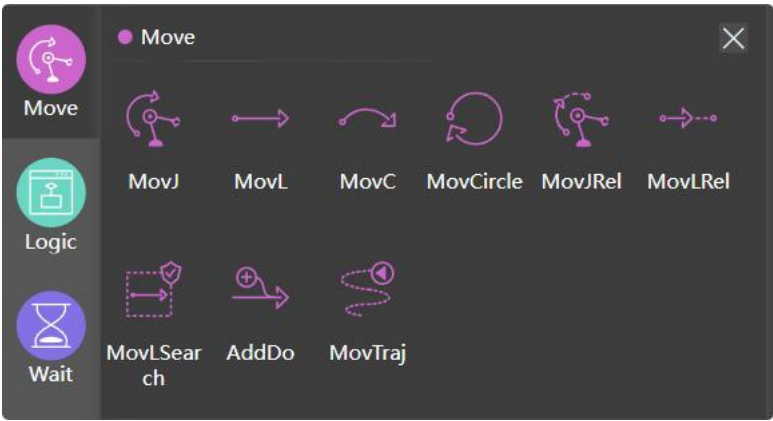
clicking on the  button, namely subtasks, interrupt tasks, programs with motion instructions and programs without motion instructions. Selecting a task or program switches the currently programmed program tree.

Interrupt tasks, programs containing motion instructions, and programs without motion instructions. Selecting a certain task or program can switch the current programming program tree.



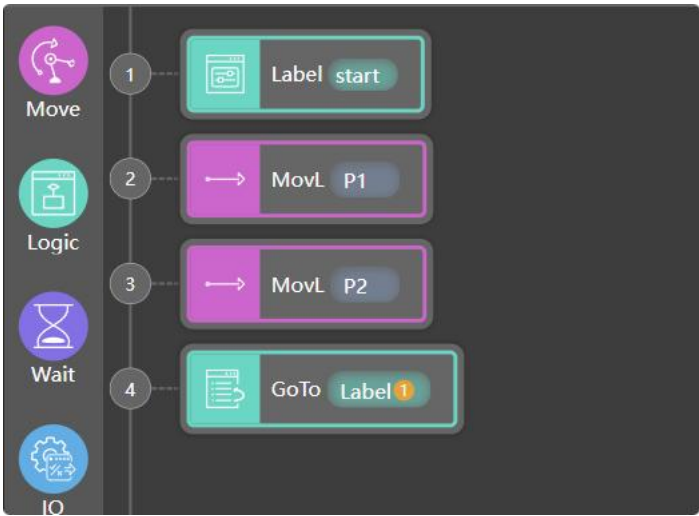
- A project only contains one main task, but can include multiple sub-tasks and multiple interrupt tasks. When the project runs, it starts from the main task.
- Sub tasks must be initiated and run within the main task through the RUN command and do not run automatically. Sub-tasks are not allowed to contain motion instructions. The main task can stop sub-tasks through the KILL command (sub-tasks are allowed to RUN and KILL other sub-tasks), but the main task cannot be KILLED.
- Macro tasks must be bound in the main task through interrupt-related instructions. When the interrupt condition is triggered, the main task enters a suspended state and then switches to the execution of the bound interrupt task. After the interrupt task finishes running, the main task resumes operation.
- The macro task cannot contain motion instructions and cannot run other tasks. Before the interrupt task exits, the robot must be moved to the position where the interrupt was triggered. Otherwise, after the interrupt task exits, the main task will remain paused. It is necessary to manually move the robot to the position where the interrupt was triggered and then click to resume operation.
- When multiple macro conditions are triggered simultaneously, the macro task that was bound first will be executed. After its completion, the triggering conditions will be re-evaluated. This logic will be repeated (i.e., only one interrupt task can run at a time and will not be interrupted by other interrupt tasks).
- A task cannot be run simultaneously by multiple other tasks.
- Programs can only be called by tasks through the Call instruction. Programs are distinguished based on whether they contain motion instructions.
- Both the main task and interrupt tasks can call all programs.
- Sub tasks can only call programs that do not contain motion instructions.
- A program can be called by multiple tasks simultaneously.

7.3.2.3 Programming instructions



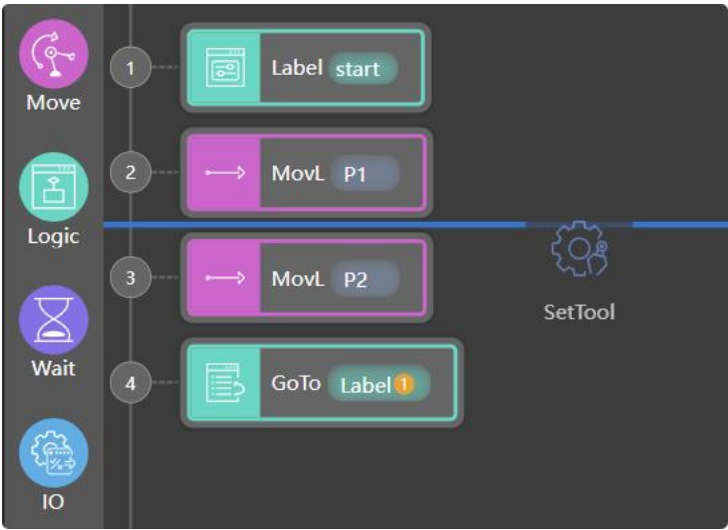
Programming instructions refer to graphical programming instructions. After selecting a category, click on the required graphical programming instruction to add it to the program tree on the right, or you can directly drag the instruction to the program tree on the right.

7.3.2.4 Program Tree



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

Add instructions

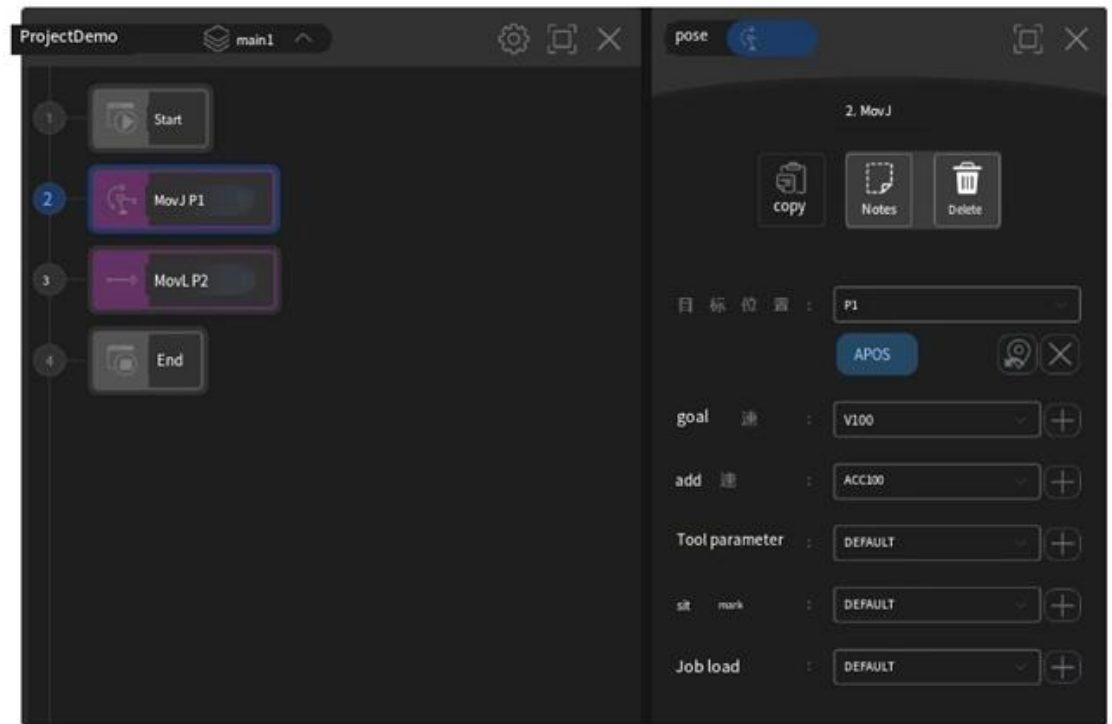



After selecting a category, click on the desired graphical programming instruction to add it to the program tree on the right. You can also directly drag and drop the instruction into the program tree on the right. Depending on the way the instruction is


inserted in the quick operation area, it can be added above, below, or as a sub-level of the currently selected instruction in the program tree.



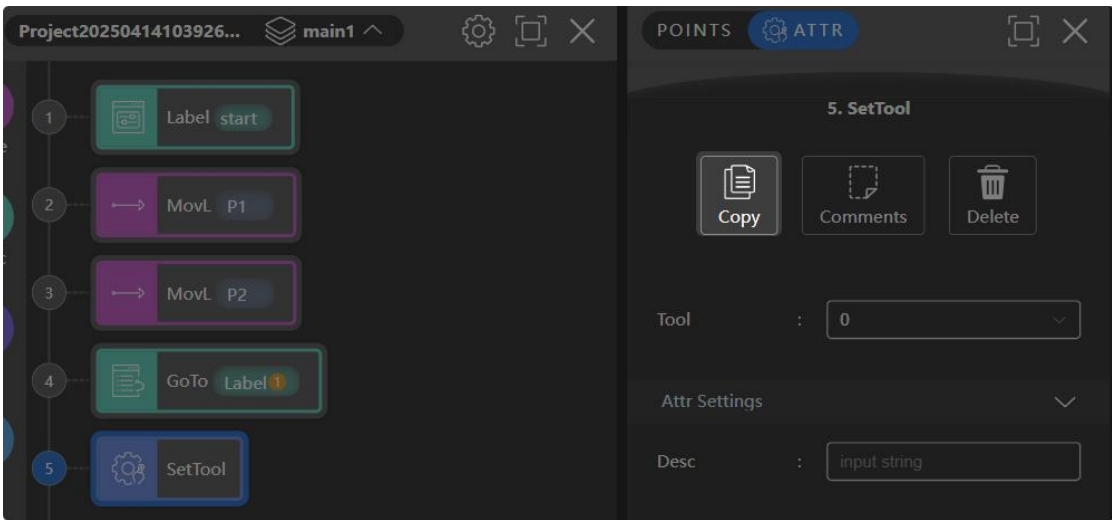
Delete/Comment Instruction




Double-click the program node to be deleted, or select the parameter list, the corresponding node edit window, and click the Delete  button.

Click the program node that needs to be commented, the corresponding node editing window pops up, click the Comment  button, the commented instruction will be kept in the program but the instruction will not be executed at runtime.

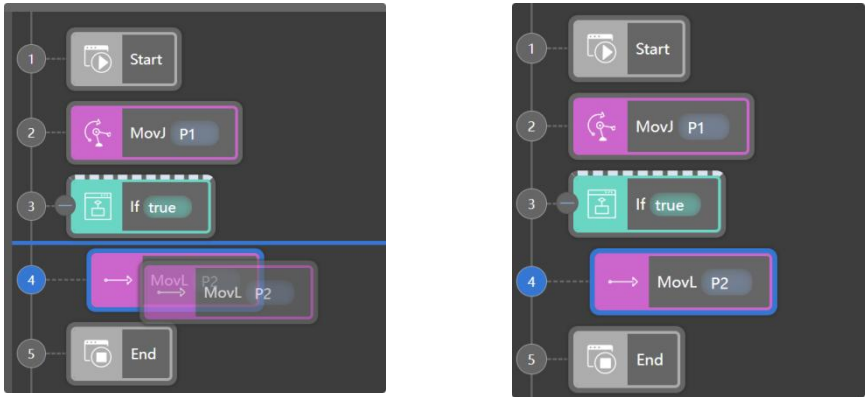
Copy instruction



Click the program node to be copied, the corresponding node editing window will pop up, click the Copy  button. The new node will be automatically pasted in the next

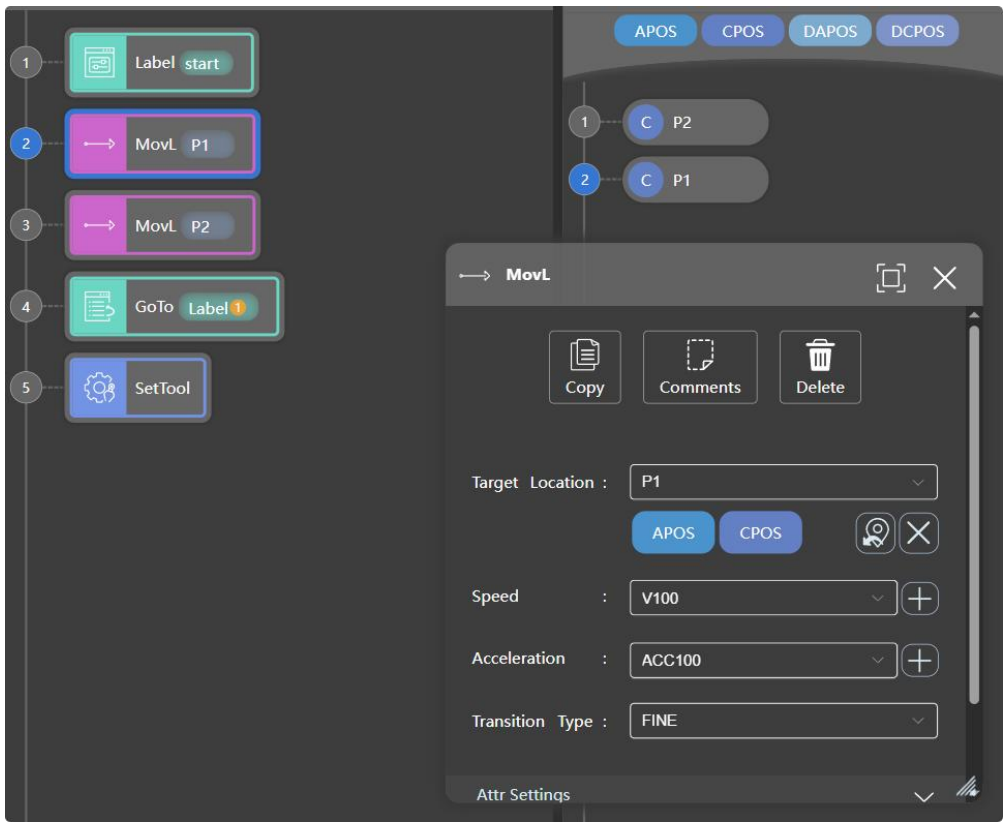
line of the copied one.

Sorting instructions



Select and drag the program node whose sequence needs to be changed and place it at the desired position. Depending on where it is released, instructions can be added above, below, or at a sub-level of a certain instruction.

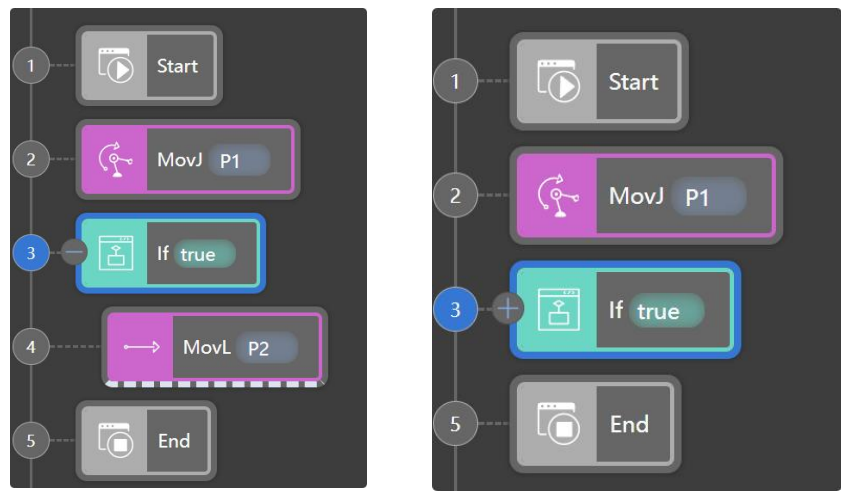
Editing instructions




Double-click the instruction that needs to be edited or select the instruction and then click the parameter list to edit the detailed parameters of the instruction.

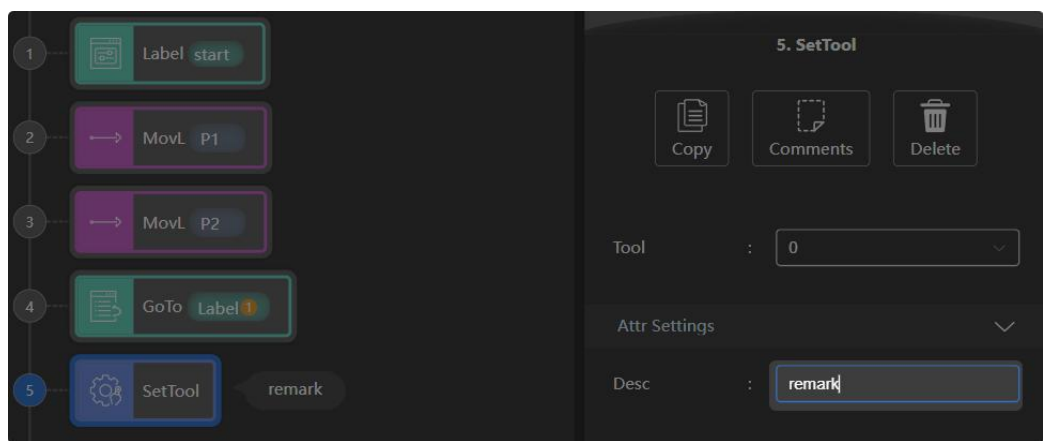
Pose List Area

Folding instructions



Some commands can be collapsed; clicking on the  in front of a command collapses the command's secondary, and vice versa expands it.

Instruction Notes



Notes can be added to commands, and the notes will be displayed on the right side of the commands.

7.3.3 Pose Zone

Add a new pose



In the pose tag button, you can double-click to add a new pose. Selecting different pose types will add the selected pose. There are four types of poses:

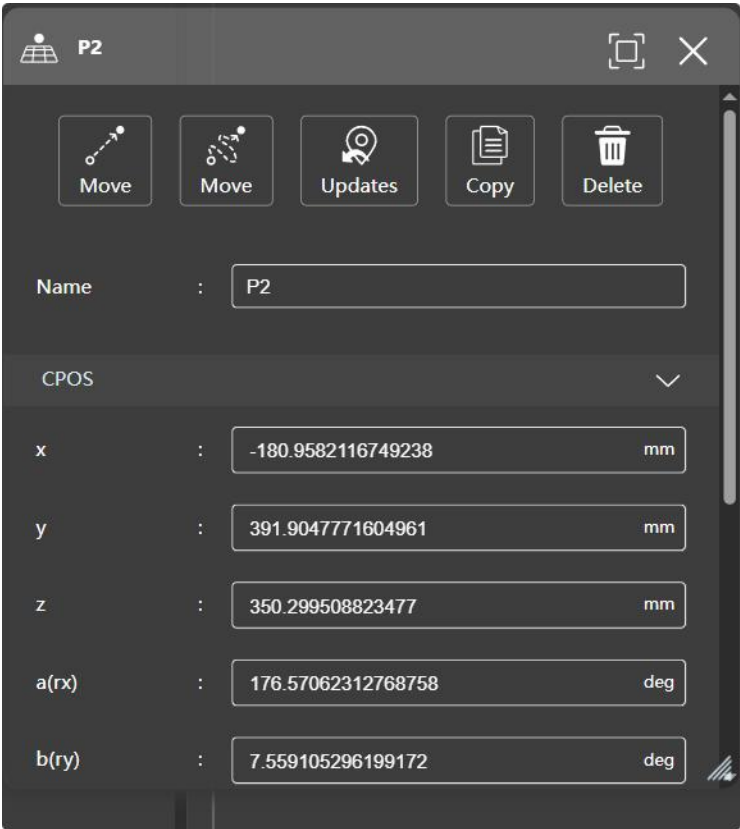
- CPOS: Cartesian position
- APOS: axis position (Joint position)

- DCPOS: delta cartesian position (increment of the Cartesian pose);
- DAPOS: delta axis position (Increment of joint position);

When CPOS and APOS are added, they represent the Cartesian pose and joint position of the current robot respectively. When DCPOS and DAPOS are added, all their values are 0.

For detailed information on various points, please refer to the Variables section.

Edit pose



Click to open the pose editor window. In this window, you can operate on the points.

- Move in a straight line to the point.
- The joint moves to the point.
- Update
- Copy
- Delete
- Edit the name of the location.
- Edit the position values of CPOS, APOS, DCPOS, and DAPOS.
- POSCFG configuration

Move to pose

In non-automatic mode, the "Move to" function has two buttons:



Move to the current position in the MovL mode.




Move to the current position in the MovJ mode.

Updated pose

Update the current Cartesian position/joint position to the selected point position via the button.

Copy pose



By copying the selected points with the  button and pasting them afterwards, the point name will be the serial number of the last added point plus 1.

Delete pose



Delete the selected points with the  button.

Edit pose name

Rename the point by using the "Name" text box.

Edit pose values

The text boxes under CPOS, APOS, DCPOS, and DAPOS can be edited. Entering a value will change the value or increment of the selected point's Cartesian pose/joint position.

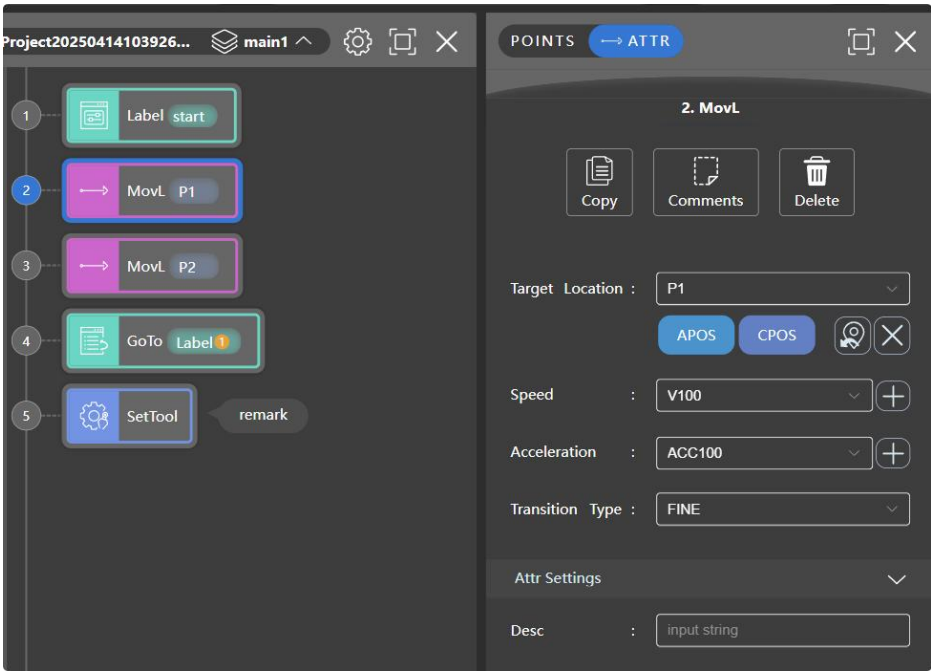
POSCG configuration

At the same Cartesian space position, a robot can have multiple combinations of joint positions (corresponding to the multiple solutions of the robot's inverse kinematics). This attribute is used to define the morphological configuration data corresponding to the spatial target point.

When mode = -1, it indicates that the current configuration is to be maintained. The kinematics of the general six-joint robot has eight sets of solutions. The mode values are defined as 0 to 7, with the meanings as shown in the following table:

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

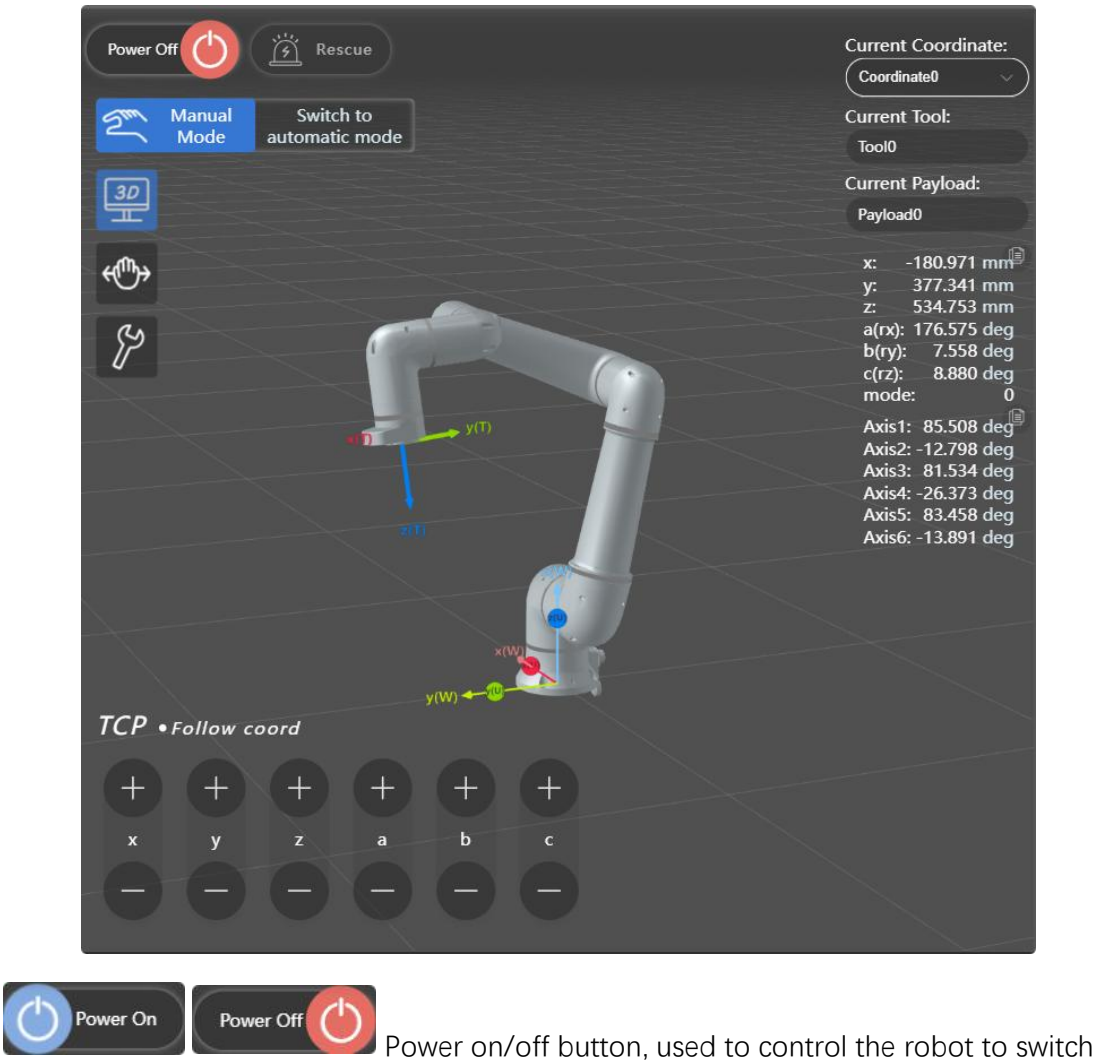
7.3.4 Parameter Area



View, edit, and delete the details of the selected program tree instructions. The parameters of each instruction are slightly different. For specific details, please refer to Chapter 10.

7.3.5 3D Simulation

Display real-time robot simulation animations as well as the Cartesian coordinate system pose and joint positions.



Power on/off button, used to control the robot to switch

on or off power.



Rescue mode, in which the joints can be jogged without motion range restrictions (enter this mode when the robot is in the "power-off state").

"Teach mode", after enabling it, power on the robot to move the joints in point mode.



Rescue Mode, to tap the joints without range of motion restrictions (enter Rescue Mode when the robot is "powered down", turn it on, and then power up the robot to tap the joints);



Current coordinate system. Switch the "user coordinate system" variable currently in use;



Current tool. Switch the "Default Tool" variable in the Settings tab - Basic;



Current load, switch the "Default Load" variable in the Settings tab - Basic;



Manual/Auto mode switching;



Simulation/live mode switching, switching between "simulation mode" and "live mode" of the robot in the power-down state, in the simulation mode, the live robot will not move;



Drag Sensitivity, which adjusts drag teach sensitivity and whether or not attitude lock is turned on;



Toolbox with tools for switching viewpoints, clearing motion trajectories, zero calibration, return to zero position, return to packed position, and more;



Stop simulation rendering and stop rendering the 3D simulation model, which saves the oscillator hardware resources.



Switch viewpoints to quickly switch between the viewpoints of the 3D simulation;



Clear Trajectory Line, clears the trajectory line of the end TCP in the 3D simulation space;



To return to the zero position, click and then long press the lower right button

back to the robot home point position;



To return to the safe point position, click and then long press the bottom right button to return to the robot's safe point position stance. This point position can be set in Setup Heavy Safety;



The robot returns to the vertical attitude position;



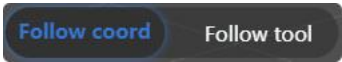
To return to the packing position, click and long press the lower right button to return to the robot's crating stance;



Target position display, whether or not to display the robot's target position for the next command in automatic mode;



Tap mode switching, switching between “joint jogging (articulation)”/“end jogging (Cartesian motion)”, and different speeds can be adjusted by the speed multiplier.



The End Tap coordinate system allows you to choose to move the robot along the current coordinate system or the tool coordinate system.


7.3.6 Register


RobotBase					
Address	Type	Name	Value	Permis...	Monitoring
0x0000	UInt8	majorVersion	-	ro	Off
0x0001	UInt8	minorVersion	-	ro	Off
0x0002	UInt8	reserved	-	ro	Off
0x0003	UInt8	reserved	-	ro	Off
0x0004	UInt16	milliSeconds	-	ro	Off
0x0005	UInt8	seconds	-	ro	Off
0x0006	UInt8	minutes	-	ro	Off
0x0007	UInt8	hours	-	ro	Off
0x0008	UInt8	moveRate	-	ro	Off
0x0009	UInt16	days	-	ro	Off
0x000A	UInt8	reserved	-	ro	Off
0x000B	UInt8	reserved	-	ro	Off
0x000C	UInt8	reserved	-	ro	Off

Save Change


The register interface displays the status of all registers. There are communication registers inside the robot. For the addresses and meanings of these registers, please refer to the register table.

In the permissions, "ro" stands for read-only access to the outside, and "rw" stands for read and write access to the outside.

When monitoring is turned on , the pendant refreshes the register values in real time. When debugging, you can communicate with the external device debugging by

modifying and issuing the value. .



7.3.7 I/O

Lock 


^ Digital Input

∨ Digital Output

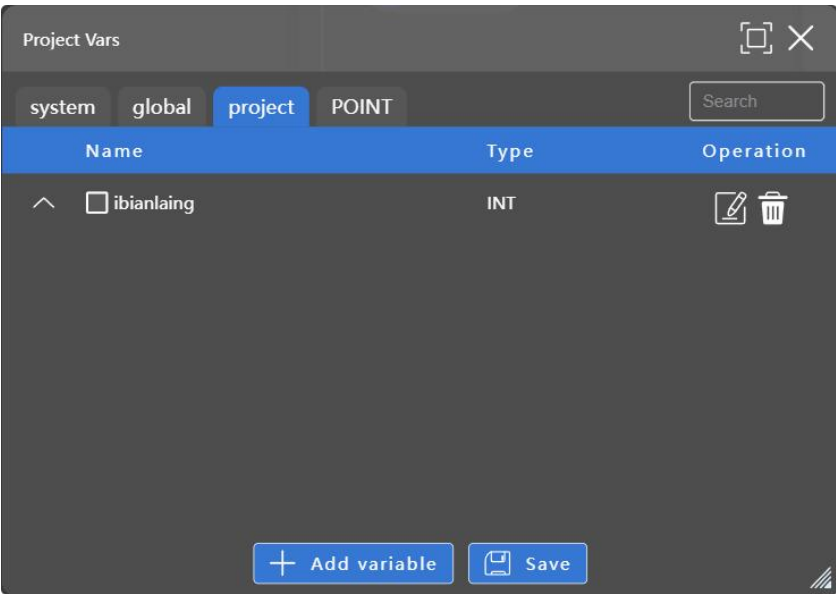
Port	Name	Value
16	DO0	<input type="checkbox"/>
17	DO1	<input type="checkbox"/>
18	DO2	<input type="checkbox"/>
19	DO3	<input type="checkbox"/>
20	DO4	<input type="checkbox"/>
21	DO5	<input type="checkbox"/>
22	DO6	<input type="checkbox"/>
23	DO7	<input type="checkbox"/>
24	DO8	<input type="checkbox"/>
25	DO9	<input type="checkbox"/>
26	DO10	<input type="checkbox"/>
27	DO11	<input type="checkbox"/>

The I/O interface shows the status of all digital IOs and analog IOs, and you can manually operate the IOs in this interface in the “unlocked” state , while the IOs in the “locked” state  cannot be manually operated.

After unlocking, you can rename the IOs to make them easier to program.

The mandatory option  forces the corresponding input to be changed to a manually selected state.

7.3.8 Variable Management



Under the variable label, defined variables can be stored. For specific details of each type of variable, please refer to Chapter 8.

Classification of variables

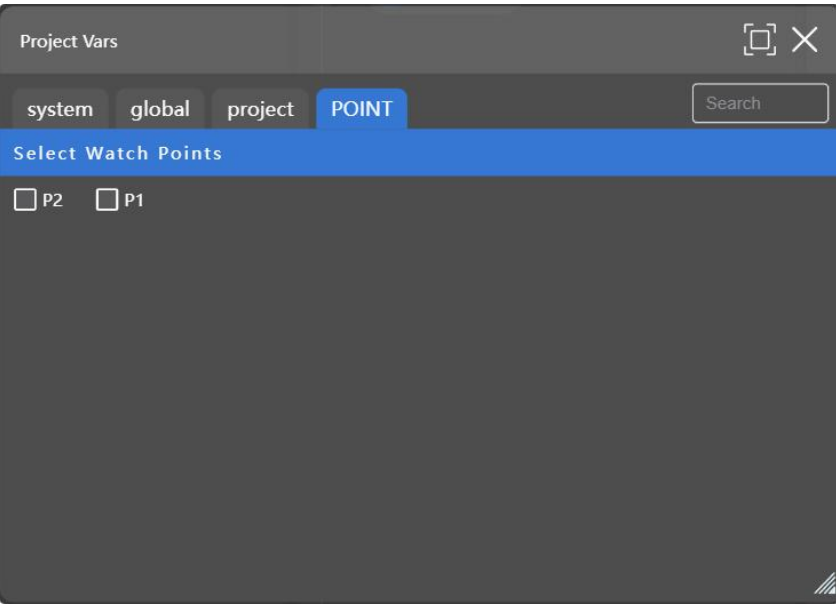
System: Stores DI/DO, AI/AO and other variables. Users are not allowed to create, edit or delete variables.

Global: Variables with a scope of "global", allowing users to create, edit or delete variables.

Project: Variables with a scope of "project", allowing users to create, edit or delete variables.

POINT variable

During the program's operation, by selecting the corresponding pose, the interface will refresh in real time to display the current point position variables.



Variable search

Enter the variable name in the search box to search for the corresponding variable.

Create a new variable

Add variable

Variable Class:

Project

Type

:

INT

Name

:

Search

Variable Value

INT

value(int)

:


0

Save


In the Variable tab, you can click “Add variable” button to add a new variable, select a different variable classification, type then add the selected variable classification, type, specific definitions refer to the introduction of variables.

The type should add the selected variable classification and type. For specific definitions, please refer to the variable introduction.

Variable monitoring

 Select the variable you want to monitor and expand it to view the running value. Currently, up to 10 variables can be viewed at the same time.

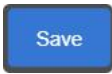
Edit variable

Click button  to edit the variable name, hold line variable and value for the current variable. The variable category and variable type cannot be changed.

Delete the variable

Click button  to delete the variable.

Save variables

Click button  (save variable) to save the variable to the controller.

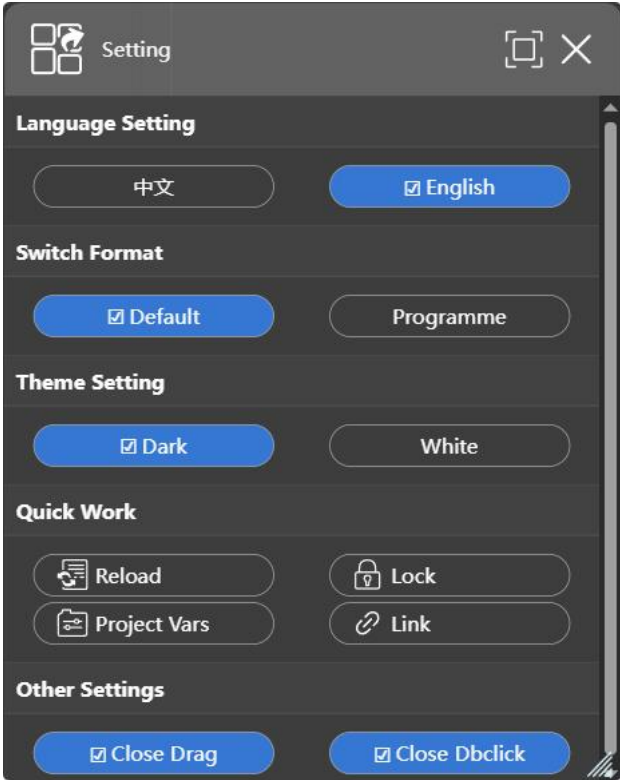
7.3.9 Project Management Area

In the Project Management menu, you can manage the projects.





Project settings, switching language, switching layout, switching theme, refreshing page, locking window, variable management, and setting online options; you can also set whether the program tree can be dragged and dropped for commands, and whether double-click is enabled in the tutorial interface.



New Project, create a new robot project;



Save project, when the button of save project is “red”, the changes of the current project have not been saved, when the button of save project is “blue”, the changes of the current project have been saved;



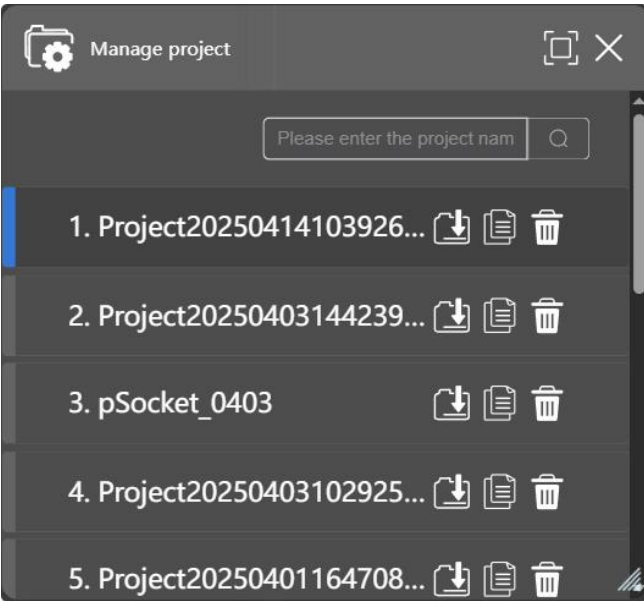
Project management, you can download, copy and delete the saved project;






Import project, import the project saved locally;



Run, run the current project (single-step execution, automatic execution), running the project must be in “automatic mode”.



In the Project Management dialog box,  is to download the project,  is to copy the project,  is to delete the project.

7.3.10 Speed ratio adjustment area



The speed rate bar for movement can adjust the speed rate. The values for manual jog mode and automatic operation mode are independent. The value range is 1% to 100%.

The actual running speed of the robot in automatic mode = the speed of motion instructions × speed ratio.

In manual mode, the joint jogging speed is 100% of the maximum joint jogging speed, the Cartesian jogging linear speed is 100% of the maximum Cartesian jogging linear speed, and the Cartesian jogging rotational speed is 100% of the maximum Cartesian jogging rotational speed. These values can be modified in the relevant options of the settings tab.

7.4 Settings tab

7.4.1 Basic

When saving parameters, the robot will automatically power off. When powered on again, the new parameters will be applied.

7.4.1.1 IP address

Double-click the IP address to change the robot's IP address. The change will take effect after the control cabinet is powered off and restarted.

7.4.1.2 Serial number

The serial numbers of the entire machine, control cabinet, robotic arm, and each joint are the unique identifiers for each component of the robot. The serial number of the entire machine will also be marked on the labels of the robotic arm and control cabinet.

7.4.1.3 Default tools

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

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

The default tool is the tool parameter loaded at startup. Incorrect selection of the default tool may cause the robot to shut down, and in severe cases, it may damage the robot's joints.

7.4.1.4 Default load

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

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

The default load is the load parameter that is loaded when the machine starts up. Selecting an incorrect default load may cause the robot to shut down, and in severe cases, it may damage the robot's joints.

7.4.1.5 DH Parameters

Users can view the DH parameters of this robot here.

7.4.1.6 Installation

You can choose a preset installation method or customize the installation offset and rotation relative to the world coordinate system. Once the robot is installed and fixed, the installation rotation and offset will not change.

7.4.1.7 xyz offset

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

7.4.1.8 abc rotation

The parameters for installation-rotation are related to the installation posture of the robot. When installing at other angles, parameter settings need to be made in the installation-rotation. After setting the parameters, the robot model on the right will rotate in real time according to the input parameters. When the simulated robot posture is consistent with the actual one, click the save button. After re-powering on, the parameters will take effect.

7.4.2 Tools, load, coordinate system

7.4.2.1 Tools

The robot can store up to 16 tool parameters, among which parameter No. 0 cannot be modified. Tool parameters can be generated through user calibration or freely inputted with numerical values. The meanings of the tool parameters are as follows:

Parameter	Parameter	Data type	Parameter Meaning
TOOL Used to record tool parameters that define tool end displacement and rotation relative to the robot flange.	x	real	The displacement offset of the TCP in the x-direction with respect to the flange coordinate system in mm.
	y	real	The displacement offset of the TCP in the y-direction with respect to the flange coordinate system in mm.
	z	real	The displacement offset of the TCP in the z-direction with respect to the flange coordinate system in mm.
	a	real	The Euler angle of rotation of the TCP with respect to the z-axis of the flange coordinate system, in deg.
	b	real	The Euler angle of rotation of the TCP with respect to the y' axis of the flange coordinate system, in deg.
	c	real	The Euler angle of rotation of the TCP with respect to the x'' axis of the flange coordinate system, in deg.
dyn(LoadDyn) is used to store the robot end tool and load quality information parameters.	M	real	The wight of the tool in kg.
Pos The position of the installed tool or load on the coordinate system	Mx	real	The offset of the center of gravity C of the mounted tool or clamped load in the X direction of the coordinate system OTool-XYZ in mm.
	My	real	The offset of the center of gravity C of the mounted tool or clamped load in the Y direction of the coordinate system OTool-XYZ in mm.

OTool-XYZ.	Mz	real	The offset in mm of the center of gravity C of the mounted tool or clamped load in the Z direction of the coordinate system OTool-XYZ.
------------	----	------	--

7.4.2.2 Load

The robot can store up to 16 load parameters, among which parameter No. 0 cannot be modified. The load parameters can be generated by user calibration.

Or enter values freely. The meanings of the load parameters are as follows:

Parameter	Parameter	Data type	Parameter Meaning
CenterPos The position of the installed tool or load on the coordinate system OTool-XYZ.	M	real	The weight of payload
	Mx	real	The offset of the center of gravity C of the clamped load in the X direction of the coordinate system OTool-XYZ in mm.
	My	real	The offset of the center of gravity C of the clamped load in the Y direction of the coordinate system OTool-XYZ in mm.
	Mz	real	The offset of the center of gravity C of the clamped load in the Z direction of the coordinate system OTool-XYZ in mm.

7.4.2.3 Coordinate system

The robot can store up to 16 coordinate system parameters, among which parameter 0 cannot be modified. The coordinate system parameters can be generated through user calibration or freely inputted with numerical values. The meanings of the coordinate system parameters are as follows:

Parameter	Data type	Parameter Meaning
x	real	The displacement offset of the user coordinate system origin relative to the world coordinate system in the x-direction in mm.
y	real	The displacement offset of the user coordinate system origin relative to the world coordinate system in the x-direction in mm.
z	real	The displacement offset of the user coordinate system origin relative to the world coordinate system in the x-direction in mm.
a	real	The Euler angle of rotation of the user coordinate system relative to the z-axis of the world coordinate system, in deg.
b	real	Euler angle of rotation of the user coordinate system relative to the y' axis of the world coordinate system, in deg.
c	real	Euler angle of rotation of the user coordinate system with respect to the x'' axis of the world coordinate system, in deg.

7.4.3 Others

The master switch can be used to determine whether to enable the security rules. When

the master switch is turned off, no rule will take effect.

7.4.3.1 Joint overspeed protection

Whether to enable the safety overspeed protection. After disabling it, the system will not detect whether the joint speed exceeds the joint overspeed threshold.

7.4.3.2 Joint hypermobility threshold

The overspeed threshold for each joint.

7.4.3.3 End-of-travel overspeed protection

Whether to enable the safety overspeed protection. If it is turned off, the system will not detect whether the terminal speed exceeds the threshold.

7.4.3.4 Joint collision detection sensitivity

Users do not need to be concerned about the specific threshold parameters for each axis. The threshold is dynamically changing. 100% represents the highest sensitivity, and 0% means it is off. The more accurate the load configuration is, the higher this setting can be.

7.4.3.5 Joint collision detection threshold

Each joint of the Codroid S series robot is equipped with a torque sensor to detect the torque applied to the joint. When the robot is powered on and the detected torque value exceeds the output torque limit threshold of the joint, the robot will report an error and power off. At this point, it is necessary to check the cause of the situation. After resolving the issue, power on the robot again.

The possible reasons for the robot exceeding the torque limit are:

- a) The actual load at the end does not match the setting.
- b) 2. The robot collided.
- c) 3. The settings for speed and acceleration are unreasonable.
- d) 4. Other circumstances.

Users can adjust the threshold as needed for their specific applications, but it is not recommended to disable the protection, as this may lead to potential security risks.

7.4.3.6 Joint Limiting

Joint limit is used to restrict the movement of each robot joint in the joint space, defining the position range of each joint. Customers can modify the threshold according to the actual application. If the threshold is set too small, it will affect the movement range of the robot.

7.4.3.7 End stop limit

The end limit is used to restrict the movement position of the robot's TCP, defining the position range in the x, y, and z axes directions and rotation. Customers can modify the threshold according to the actual application. If the threshold is set too small, it will affect the robot's movement range.

7.4.3.8 Safe positions

The robot's posture at the set safety position can be configured to output a signal in the set IO when the robot is at that position.

7.4.3.9 Manual mode terminal speed limit

In manual mode, the maximum Cartesian speed of the robot can be jogged. Under any circumstances in manual mode, the speed will not exceed this value.

7.4.3.10 Load verification sensitivity

When the drag robot function is enabled, the robot will verify whether the current load is correct before the function is turned on. If the actual load deviates too much from the theoretical load, the robot will not start dragging to protect itself and the operator. Adjusting the sensitivity level can limit the deviation threshold.

7.4.3.11 Drag enable sensitivity check

At the moment the drag button is pressed, the robot will recheck whether the load configuration is correct to prevent sudden movement of the robot caused by the user disabling collision detection while the load configuration is incorrect.

7.4.4 Sports

Motion parameters define the maximum values of the robot's speed, acceleration and jerk in both automatic and manual modes.

Optimization of sports performance

When enabled, it will optimize the jittering situation during low-speed movement.

7.4.4.1 Point movement

Joint speed

In manual mode, the maximum speed of joint jogging is 30°/s. You can limit the maximum speed of joint jogging here.

End linear velocity

In manual mode, the maximum linear speed of Cartesian point movement is 250 mm/s. You can limit the maximum linear speed of Cartesian manual point movement here.

Angular velocity at the end

In manual mode, the maximum angular velocity of the end rotation for Cartesian point movement is 30°/s. You can set the maximum angular velocity of the end rotation for Cartesian manual point movement here.

7.4.4.2 It's time for exercise.

Joint speed

In manual mode, the default joint angular velocity for moving to a point in joint mode is 30°/s, with a maximum of 90°/s.

End linear velocity

In manual mode, the Cartesian linear velocity for moving to a point in Cartesian mode is 250 mm/s by default, with a maximum of 1000 mm/s.

Angular velocity at the end

In manual mode, the default Cartesian angular velocity for moving to a point in Cartesian mode is 30°/s, with a maximum of 90°/s.

7.4.4.3 Automatic

Maximum joint speed

The upper limit of the speed that the robot can reach when moving in automatic mode. After setting the maximum speed, when creating a new variable of the SPEED type, the speed setting of the TCP will be restricted to be less than this maximum value.

Maximum joint acceleration

In automatic mode, the joint acceleration limits the maximum value of acceleration. Users can adjust the maximum acceleration value appropriately according to the application, which can increase the movement tempo. However, if the acceleration is set too high, it may cause the robot to shake when starting and stopping. Long-term use of unreasonable acceleration may cause damage to the joint reducer.

Joint acceleration increment

The upper limit of the jerk that the robot can achieve during movement in automatic mode. The smaller the value, the smoother the movement process, but the longer the time consumed.

Maximum terminal velocity

The maximum linear speed of the robot end effector in automatic mode. After setting the maximum speed, when creating a new variable of the SPEED type, the speed setting of the TCP will be restricted to be less than this maximum value.

Maximum acceleration at the end

In automatic mode, the maximum value of the linear acceleration at the robot's end is limited. Users can adjust the maximum acceleration value appropriately according to the application, which can increase the motion cycle. However, if the acceleration is set too high, it may cause the robot to shake when starting or stopping. Long-term use of an unreasonable acceleration setting may cause damage to the joint reducer.

Add acceleration at the end.

The upper limit of the linear acceleration that the robot can achieve when moving in automatic mode. The smaller the value, the smoother the movement process, but the longer the time consumed.

Pause time

The deceleration time when the robot program is paused in automatic mode.

7.4.5 Register communication

Since ModbusTCP, ProfiNet, and EtherNet/IP all operate on the same register address, only one communication protocol can be selected for use. If none of the communication protocols are to be used, the enable function should be turned off. To use ProfiNet or EtherNet/IP, a communication module for the entity must be added. After modifying the settings, the changes will take effect only after saving and restarting the robot.

7.4.5.1 ModbusTCP

- Protocol Version

The current software version of ModbusTCP used by the robot.

- Port

The port used by the ModbusTCP protocol when the robot is in the standby state.

- Slaves address

The address where the robot stands when it is in the ready position.

7.4.5.2 ProfiNet

- Protocol Version

The current ProfiNet software version used by the robot.

- Slaves name

The name of the robot ProfiNet slave station can be modified by double-clicking. The modification will take effect after the robot is restarted.

- IP

The IP address of the robot ProfiNet module can be modified by double-clicking to change the name. The changes will take effect after the robot is restarted.

- Data mode

The ProfiNet protocol data mode supported by different brand devices varies slightly. You can choose either big-endian mode or little-endian mode.

7.4.5.3 EtherNetIP

- Protocol Version

The current software version of EtherNet/IP used by the robot.

- IP

The IP address of the robot's EtherNetIP module. Double-click to change the name and reboot the robot before it takes effect.

- Data model

The EtherNet/IP protocol data modes supported by different brand devices are slightly different, and you can choose either big-endian mode or little-endian mode.

7.4.6 IO

7.4.6.1 DI Function Configuration

When the system detects that the corresponding digital input variable meets the trigger conditions, it executes the corresponding usage function. Click to create a new function. You can add multiple actions with the same variable and condition to accomplish the effect of multiple actions.

Stop dragging: Disable manual dragging of the robot in the mode.

Power on: Enable the robot by powering it on.

Power on in rescue mode: Enable the robot in rescue mode, which temporarily disables safety checks; Power off: Disable the robot by cutting off power.

Switch to automatic mode: The robot switches to the automatic operation program mode.

Switch to manual mode: The robot switches to manual teaching mode.

Run the last saved program: Run the last saved program in the automatic operation mode; Run the specified program: Run the program specified in the drop-down box in the automatic operation mode.

Stop operation: The robot stops running the program.

Pause operation: The robot pauses the running program.

Continue running: Resume the paused program;

Error reset: Clear the robot's error report;

Protective stop: Protective emergency stop of the robot;

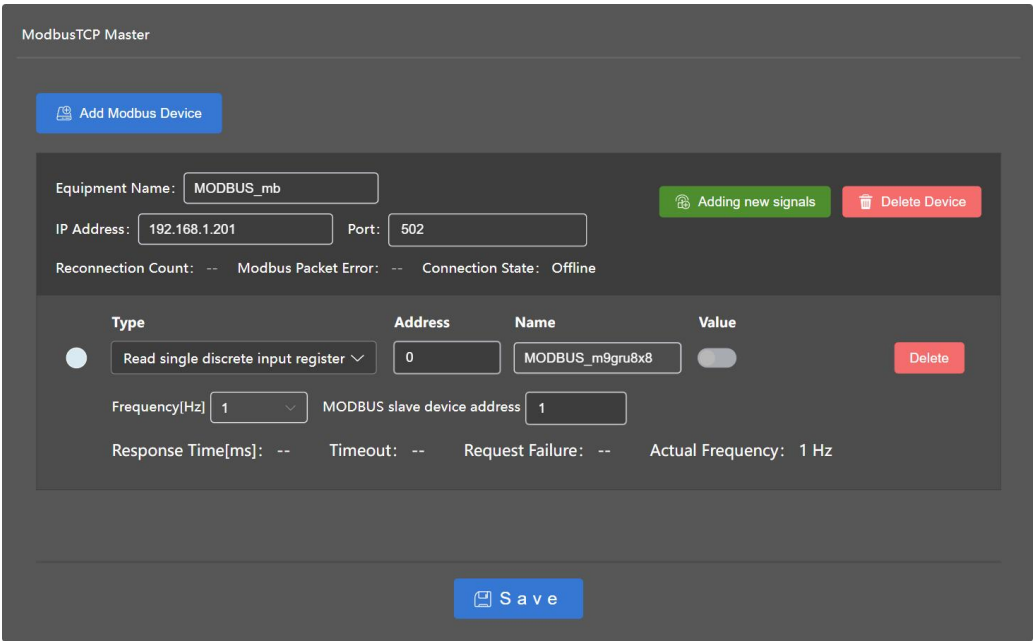
7.4.6.2 DO Function Configuration

When the system detects that the trigger conditions are met, the corresponding digital output function is executed. Click to create a new function configuration. Only one operation can be added for the same variable.

- Exception: When an abnormality occurs in the robot, the corresponding electrical level is output.
- Program running: The corresponding level is output when the robot program is running;
- Program pause: The robot outputs the corresponding level when the program is paused;
- At safety position: The robot outputs the corresponding level when it is at the safety position.

7.4.7 MODBUS Master

Here, you can set the parameters of the MODBUS master (client). A connection can be established between the local machine and the MODBUS slave (server) at the target IP address. Each signal has a unique name and can thus be used in the program.



- Add device**
This button can be used to add a new MODBUS slave device.
- Delete device**
This button can delete the MODBUS slave device and all its signals.
- Device name**

The device name can be set to distinguish each device.

- **IP address**

The IP address of the MODBUS slave device can be changed here.

- **Port**

The port address of the MODBUS slave device can be changed here

- **Reconnection count**

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

- **Modbus data pack error**

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

- **Connection status**

TCP connection status.

- **Add signal**

This button can add signals to the corresponding MODBUS slave device.

- **Delete signal**

This button can delete the signal from the corresponding MODBUS slave device.

- **Type**

Select the signal type. Available types include:

Read a single coil register (read output coil), read discrete input register (read input coil), read a single holding register (read output register), read input register (read input register), write a single coil register (write output coil), write a single holding register (write output register).

- **Address**

To display the address of the remote MODBUS slave device, you can select a different address. The valid address depends on the manufacturer and the configuration of the remote MODBUS slave device.

- **Name**

Names can be assigned to signals. Signal names are used when the signals are utilized in the program.

- **Frequency**

It can be used to change the update frequency of the signal. The update frequency refers to the frequency at which requests are sent to the remote MODBUS slave device to read or write signal values. When the frequency is set to 0, the MODBUS request will be initiated on demand using instructions in the program.

- **From device address**

This text field can be used to set the specific slave device address for requests

corresponding to a particular signal. The value must be within the range of 0 to 255, with the default value being 255. If you need to change this value, it is recommended to first consult the manual of the remote MODBUS device to verify that the function is normal after the slave device address is changed.

- **Response time [ms]**

The time between sending a MODBUS request and receiving a response is updated only when the communication is active.

Overtime

The number of unresponded MODBUS requests.

Request Failed

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

Actual frequency

The average frequency of signal status updates for the main station device (client). This value is recalculated each time a response is received from the slave station device (server).

Save

Save the settings and refresh all MODBUS connections. All MODBUS slave devices will be disconnected and reconnected. All statistical information will be cleared.

7.4.8 Panel IO

The IO settings can enable the analog output on the control cabinet panel and set the analog input mode.

When using the analog output port, a load must be connected; otherwise, the robot will report an error. If it is not in use, the corresponding port must be turned off.

When using the analog input port, it is necessary to specify the usage mode, either current mode or voltage mode; otherwise, the robot will report an error.

7.5 Log tab

The log module records some operation anomalies of users, provides relevant prompts, and offers assistance in using the software. Meanwhile, when encountering problems, it can display the corresponding window prompts for viewing, providing information to professionals for obtaining help and resolving issues.

ESTUN
C O D R O I D

PROJECT SETTING **RECORD** MANAGE

admin

Number	File Name	Create Time	File Size	Operation
1	OutputCtrl.txt	2025-4-14 16:02:40	407.89 KB	Download Log
2	OutputCtrl.1.txt	2025-4-14 10:45:46	96.42 KB	Download Log
3	OutputCtrl.2.txt	2025-4-9 13:17:13	89.41 KB	Download Log
4	OutputCtrl.3.txt	2025-4-8 09:03:36	90.22 KB	Download Log
5	OutputCtrl.4.txt	2025-4-7 09:19:29	367.67 KB	Download Log
6	OutputCtrl.5.txt	2025-4-3 12:53:26	132.22 KB	Download Log
7	OutputCtrl.6.txt	2025-4-3 10:32:28	145.94 KB	Download Log
8	OutputCtrl.7.txt	2025-4-1 17:21:11	1 MB	Download Log
9	OutputCtrl.8.txt	2025-4-1 17:00:56	111.12 KB	Download Log
10	OutputCtrl.9.txt	2025-3-31 13:34:48	88.66 KB	Download Log

Click the exclamation mark button at the top right corner of the program to view the error message. If the button is flashing, it indicates that there is an error in the program and the program will stop running.

Only the latest 10 entries are retained in the system log. Click the button [Download Log](#) (log download) in the Logs tab to download this log message locally.

7.6 Management tab

The management interface can import or export some settings or engineering parameters of the controller, as well as manage users.

ESTUN
C O D R O I D

PROJECT SETTING RECORD **MANAGE**

admin

Manage Update User List Manage user

Export user config Import user config

Select the config to export

☐ 参数设置(导入完成后必须立即重启)

☐ 全局变量

☐ ModbusTCP主站配置(导入完成后必须立即重启)

☐ 工具、负载、坐标系参数(导入完成后必须立即重启)

☐ 寄存器配置(导入完成后必须立即重启)

☐ IO配置(导入完成后必须立即重启)

☐ 机械参数(零点、DH、序列号、拖动参数,谨慎导入,导入完成后必须立即重启)

Export compression file name

.zip

Confirm Export

The meanings of the import and export configuration are as follows:

- Parameter settings (take effect after restart)

Parameter settings in the Robot Settings tab.

- Global variable

Global variables in robot engineering are not imported or exported along with the project, so all users can manage them uniformly here.

- ModbusTCP Master Configuration (Effective after restart)

The ModbusTCP master configuration set by the user in the settings tab.

Manage Update

User List

Manage user

Export user config

Import user config

Select the config to export

☐

参数设置(导入完成后必须立即重启)

☐

全局变量

☐

ModbusTCP主站配置(导入完成后必须立即重启)

☐

工具、负载、坐标系参数(导入完成后必须立即重启!)

☐

寄存器配置(导入完成后必须立即重启!)

☐

IO配置(导入完成后必须立即重启!)

☐

机械参数(零点、DH、序列号、拖动参数,谨慎导入,导入完成后必须立即重启!)

Export compression file name

.zip

Confirm Export

In the user list, the admin user can create and delete users here.

The initial accounts and passwords available are as follows. Different accounts have different permissions. For details, please refer to the appendix.

Account	Password	Level
user	123456	User
admin	123456	Administrator

Among the registered users, new users can be created and assigned usernames, passwords, and permission levels.

Chapter 8 Introduction to Variables

8.1 Variable Overview

Different domains support different variable types, as described specifically below:

- System domain: System-defined variables that cannot be edited.
- Global domain: IO data types, PLC data types, socket data types, position data types, area data types, basic data types, clock data types, palletizing data types, system data types.
- Engineering domain: IO data types, socket data types, position data types, basic data types, palletizing data types, system data types.

The following "names" are reserved by the system. Names are not case-sensitive, and users cannot create variables with the same name:

abs, acos, and, asin, assert, atan, break, ceil, collectgarbage, coroutine, cos, debug, deg, do, dofile, else, elseif, end, error, exp, false, findEnd, floor, fmod, for, format, function, getAt, getmetatable, goto, huge, if, in, io, ipairs, left, load, loadfile, local, log, math, max, min, modf, next, nil, not, or, os, package, pairs, pcall, pi, print, rad, random, randomseed, rawequal, rawget, rawset, real, repeat, require, return, reverse, right, select, strcmp, setmetatable, sin, sqrt, string, table, tan, then, tonumber, tostring, true, type, until, while, xpcall, AI, AO, APOS, APosToCPos, APosToStr, AREA, AreaActivate, AreaDeactivate, ARRAYS, BitAnd, BitNeg, BitOr, BitXOr, BitLSH, BitRSH, BOOL, CalcTool, CalcCoord, CALL, CenterPos, CLKRead, CLKReset, CLKStart, CLKStop, CLOCK, CompareAI, CompareSimAI, CPOS, CPosToAPos, CPosToCPos, CPosToStr, DAPOS, DCPOS, DI, DO, ELSE, ELSIF, ENDIF, ENDWHILE, EXT CPC, GetCamPos, GetCurAPos, GetCurCPos, GetCurOverride, GetDI8421, GetMatrix, GetSimAIToVar, GetSimDI8421, GetSimDIToVar, GetTrackId, GOTO, Hand, InertiaTensor, INT, IToStr, LABEL, LoadDyn, MovArch, MovC, MovCW, MovCircle, MovCircleW, MovE, MovH, MovJ, MovJRel, MovJSearch, MovL, MovLRel, MovLSearch, MovLSync, MovJSyncQuit, MovLSyncQuit, MovLW, OnDistance, OnParameter, PalletFromGet, PalletFromPut, PalletReset, PalletToGet, PalletToPut, PAYLOAD, PLCBOOL, PLCDINT, PLCINT, PLCREAL, POLYHEDRON, PolyhedronAreaActivate, PolyhedronAreaDeactivate, POSCFG, POSITIONER, PulseOut, PulseSimOut, ReadModbusReg, REAL, RefRobotAxis, RET, RETURN, RToStr, RUN, SendMessage, SetAxisVibraBLevel, SetAO, SetCartDyn, SetCoord, SetDIEdge, SetDO, SetDO8421, SetExternalTCP, SetJointDyn, SetMotionMode, SetOverride, SetPayload, SetPositioner, SetRestorePC, SetRtInfo, SetRtToErr, SetRtWarning, SetMatrix, SetSimAO, SetSimAOByVar, SetSimDIEdge, SetSimDO, SetSimDO8421, SetSimDOByVar, SetTargetPos, SetTool, SetSyncoord, SimAI, SimAO, SimDI, SimDO, SocketClose, SocketCreate, SocketReadInt, SocketReadReal, SocketReadStr, SocketSendStr, SoftFloatStart, SoftFloatStop, SPEED, Stop, STRING, StrToI, StrToR, SYNCOORD, SynCToUserC, TOOL, Tracking, TranStrToApos, TranStrToCpos, TranStrToInt, TranStrToReal, TrigCam, trimLeft, trimRight, USERCOOR, Wait, WaitAI, WaitCondition, WaitConvDis, WaitDI, WaitDI8421, WaitFinish, WaitFinishCAM, WaitSimAI, WaitSimDI, WaitSimDI8421, WaitWObj, WEAVE, WHILE, WriteModbusReg,

8.2 Variable

8.2.1 POSE

Store the absolute coordinate values and offset values of each axis in the Cartesian space of the robot joint.

Parameter	Parameter	Data type	Parameter Meaning
APOS Stores the joint angle values for each axis under the joint space.	jntpos1	real	Angle of joint 1 axis.
	jntpos2	real	Angle of joint 2 axis.
	jntpos3	real	Angle of joint 3 axis.
	jntpos4	real	Angle of joint 4 axis.
	jntpos5	real	Angle of joint 5 shaft.
	jntpos6	real	Angle of joint 6 axis.
CPOS Stores the position of the TCP point under the Cartesian coordinate system.	x	real	The coordinate of the TCP point in the x-direction on the reference coordinate system.
	y	real	The coordinate of the TCP point in the y-direction on the reference coordinate system.
	z	real	The coordinate of the TCP point in the z-direction on the reference coordinate system.
	a(rx)	real	The Euler angle of rotation of the TCP point with respect to the x-axis of the fixed reference coordinate system.
	b(ry)	real	The Euler angle of the rotation of the TCP point with respect to the y-axis of the fixed reference coordinate system.
	c(rz)	real	Euler angle of rotation of the TCP point with respect to the z-axis of the fixed reference coordinate system.
DAPOS Stores the relative joint angle offsets for each axis under the joint space.	djntpos1	real	Angular offset of joint 1 axis.
	djntpos2	real	Angular offset of joint 2 axis.
	djntpos3	real	Angular offset of joint 3 axis.
	djntpos4	real	Angular offset of joint 4 axis.
	djntpos5	real	Angular offset of joint 5 axis.
	djntpos6	real	Angular offset of joint 6.
DCPOS Stores the position of the TCP point under the Cartesian coordinate system.	dx	real	The coordinate offset of the TCP point in the x-direction on the reference coordinate system.
	dy	real	The coordinate offset of the TCP point in the y-direction on the reference coordinate system.
	dz	real	The coordinate offset of the TCP point in the z direction on the reference coordinate system.
	da	real	The offset of the Euler angle of rotation of the TCP point with respect to the x-

			axis of the reference coordinate system.
	db	real	The Euler angle offset of the TCP point rotated with respect to the y-axis of the reference coordinate system.
	dc	real	The offset of the Euler angle for the rotation of the TCP point with respect to the z-axis of the reference coordinate system.

8.2.2 Basic Data Types

Parameter	Data type	Scope	Remark
STRING	String	Global, Engineering, Tasks	String
BOOL	Boolean	Global, Engineering, Tasks	Numeric range: true, false
INT	Plastic	Global, Engineering, Tasks	Numeric range: -9999999999999~99999999999999
REAL	Real	Global, Engineering, Tasks	Numeric range: -9999999999~99999999999999
BoolOneArray	Boolean Arrays	Global, Engineering, Tasks	Data length: 1~255
IntOneArray	Plastic Arrays	Global, Engineering, Tasks	Data length: 1~255
RealOneArray	Real Arrays	Global, Engineering, Tasks	Data length: 1~255

8.2.3 SPEED

It is used to define the movement speed of the robot and external axes. For the convenience of users, the system presets commonly used speed variables (system variables that are not allowed to be modified by users), and at the same time, it supports users to create, delete, modify and other operations on this variable in the three variable scopes of global, project and program.

Parameter	Data type	Parameter Meaning
per	real	Joint Speed Percentage. Used to specify the speed of movement during joint movement commands, applicable to MovJ and other commands, value range 1%~100%.
tcp	real	TCP Linear Velocity. Define the linear velocity of the robot end point, used for MovL, MovC and other linear arc motion instructions.
ori	real	Spatial Rotation Velocity. Define the rotation speed of robot end point attitude, used for MovL, MovC and other linear circular motion instructions.
exj_l	real	External Axis Speed. Defines the speed of the external linear axis motion.
exj_r	real	External Axis Angular Velocity. Defines the motion speed of the external rotary axis.

8.2.4 ACC

This parameter is used to define the motion acceleration of the robot and external axes. To achieve a sufficiently fast motion speed, this parameter is usually adjusted, but it is not recommended to set the value too high, as it may cause vibrations and even damage to the joints after long-term operation.

Parameter	Data type	Parameter Meaning
joint	real	Joint Acceleration Hundred. Used to specify the motion acceleration during joint motion commands, for commands such as MovJ.
tcp	real	TCP Line Acceleration. Defines the line acceleration at the robot's end point, used for MovL, MovC, and other linear circular motion commands.
ori	real	Spatial Rotation Acceleration. Defines the rotational acceleration of the robot's end point attitude, used for MovL, MovC and other linear circular motion instructions.

8.2.5 ZONE

It is used to define how a certain motion ends or the size of the turning area between two motion trajectories. For the convenience of users, the system presets commonly used transition variables (system variables that users are not allowed to modify), and at the same time, it supports users to create, delete, modify and perform other operations on this variable within the three variable scopes of global, project and program.

Parameter	Data type	Parameter Meaning
per	real	Turning percentage. Applies to motion commands such as MovJ,MovL,MovC, etc. Indicates how far away from the target point the turn will start.
dis	real	Cartesian space turn area size. Used for MovL,MovC and other linear arc motion instructions, defines the size of the turn zone of the Cartesian space trajectory, i.e., when the robot moves to dis millimeters away from the target point, it starts to turn to move toward the next target point, the unit is mm.

8.2.6 CLOCK

The value of CLOCK stores clock information.

Parameter	Data type	Parameter Meaning
state	bool	Enable state of the clock variable.
value	int	The count value of the clock variable.

8.2.7 Socket

The network connection Socket variable.

Parameter	Data type	Parameter Meaning
Socket 名称	String	Non

8.2.8 INTERRUPT

Interrupt variable.

Parameter	Data type	Parameter Meaning
value	String	Non

8.2.9 LsScale

The LsScale type variable is used to record the gain ratio threshold parameters of each joint axis, which helps to improve the low-speed jitter phenomenon of the robot within a certain speed range. It is used in conjunction with the speed range threshold parameters. The setting range is [100, 1000], with the unit being %. This variable can only be created and modified in the global domain.

Parameter	Data type	Parameter Meaning
J1	int	Gain ratio threshold for J1 axis in %.
J2	int	Gain scale threshold value for J2 axis in %.
J3	int	Gain proportional threshold value for J3 axis in %.
J4	int	Gain ratio threshold value for J4 axis in %.
J5	int	Gain ratio threshold value for J5 axis in %.
J6	int	Gain scale threshold for J6 axis in %.

8.2.10 LsThresh

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

Parameter	Data type	Parameter Meaning
J1	int	Speed interval threshold for J1 axis, in r/min.
J2	int	Speed interval threshold for J2 axis, in r/min
J3	int	J3 axis speed interval threshold in r/min.
J4	int	Speed interval threshold for J4 axis in r/min.
J5	int	Threshold value of the speed interval in r/min for the J5 axis.
J6	int	Speed interval threshold in r/min for J6 axis

8.2.11 VibrationSuppression

Vibration Suppression Parameters: Vibration suppression parameters.

Parameter	Data type	Parameter Meaning
Frequency X	real	Intrinsic frequency of vibration in the X direction
Frequency Y	real	Intrinsic frequency of vibration in Y direction
Frequency Z	real	Intrinsic frequency of vibration in the Z-direction

Damping Ratio X	real	Damping ratio in X direction
Damping Ratio Y	real	Damping ratio in Y direction
Damping Ratio Z	real	Damping ratio in Z direction

8.2.12 Matrix2

The Matrix2 type variable is used to record a two-dimensional array.

Parameter	Data type	Parameter Meaning
Matrix2 Name	string	Array name

8.2.13 Matrix3

The Matrix2 type variable is used to record a two-dimensional array.

Parameter	Data type	Parameter Meaning
Matrix3 Name	string	Array name

8.2.14 Matrix4

The Matrix2 type variable is used to record a two-dimensional array.

Parameter	Data type	Parameter Meaning
Matrix4 Name	string	Array name

8.2.15 Matrix9

The Matrix2 type variable is used to record a two-dimensional array.

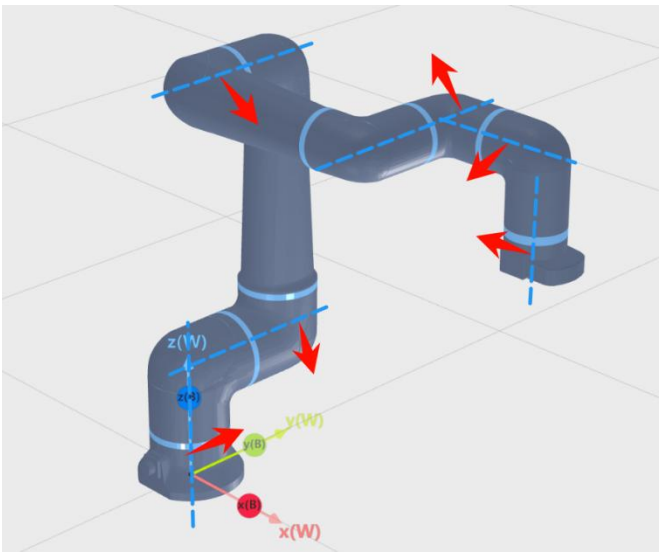
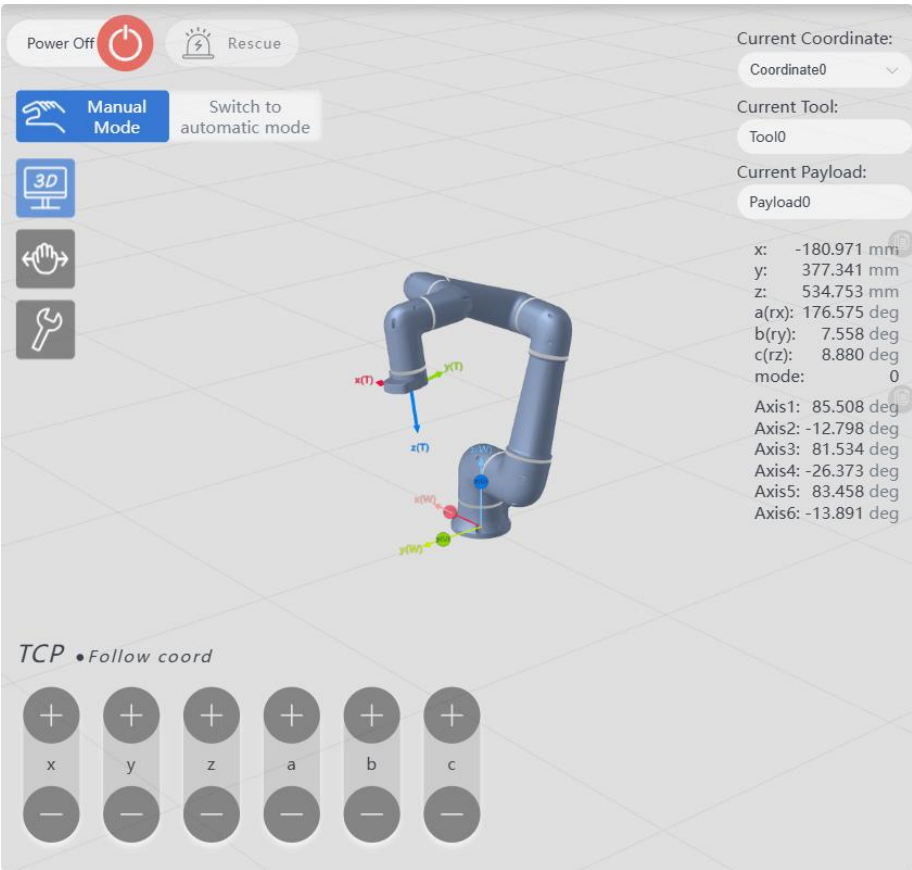
Parameter	Data type	Parameter Meaning
Matrix9 Name	string	Array name

Chapter 9 Calibration

This chapter will describe the joint coordinate system, world coordinate system, user coordinate system, tool coordinate system and their usage.

9.1 Joint coordinate system

The joint coordinate system or joint space refers to the independent movement of robot joints, which is called joint motion.

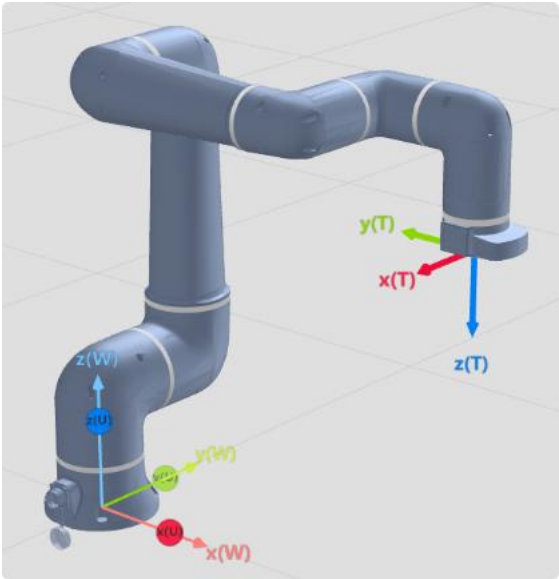


9.2 World coordinate system

The Cartesian coordinate system of the Codroid robot is a right-handed coordinate system, and its Euler angle format is X-Y-Z fixed angles. For example, the pose [900mm, 200mm, 1200mm, 20°, 30°, 45°] means that it first moves to the position of x=900mm, y=200mm, z=1200mm in the reference coordinate system, and then rotates the end TCP point as the center of rotation. First, it rotates the end along the X-axis of the reference coordinate system by 20°, then rotates the end along the Y-axis of the world coordinate system by 30°, and finally rotates the end along the Z-axis of the world coordinate system by 45°.

When leaving the factory, the robot is by default located at the position [0, 0, 0, 0, 0, 0] in the world coordinate system, meaning that the pose of the robot's base coordinate system coincides with that of the world coordinate system. The base of the robot is oriented such that the Y-axis points in the negative direction and the Z-axis points

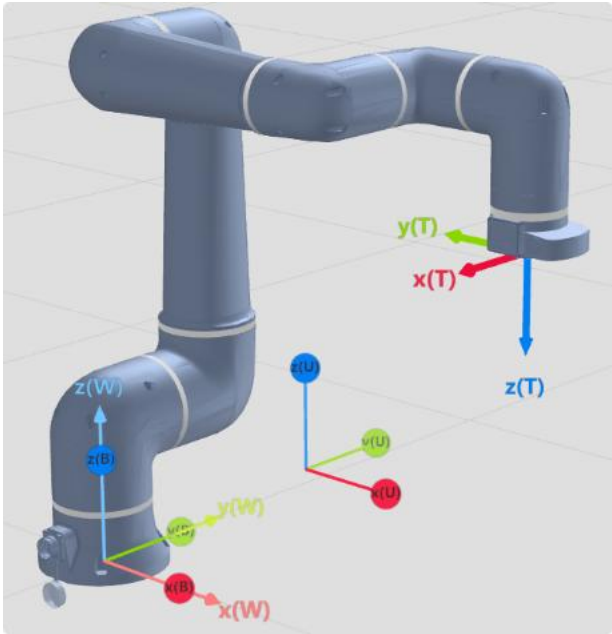
towards the interior of the base in the robot's base coordinate system.



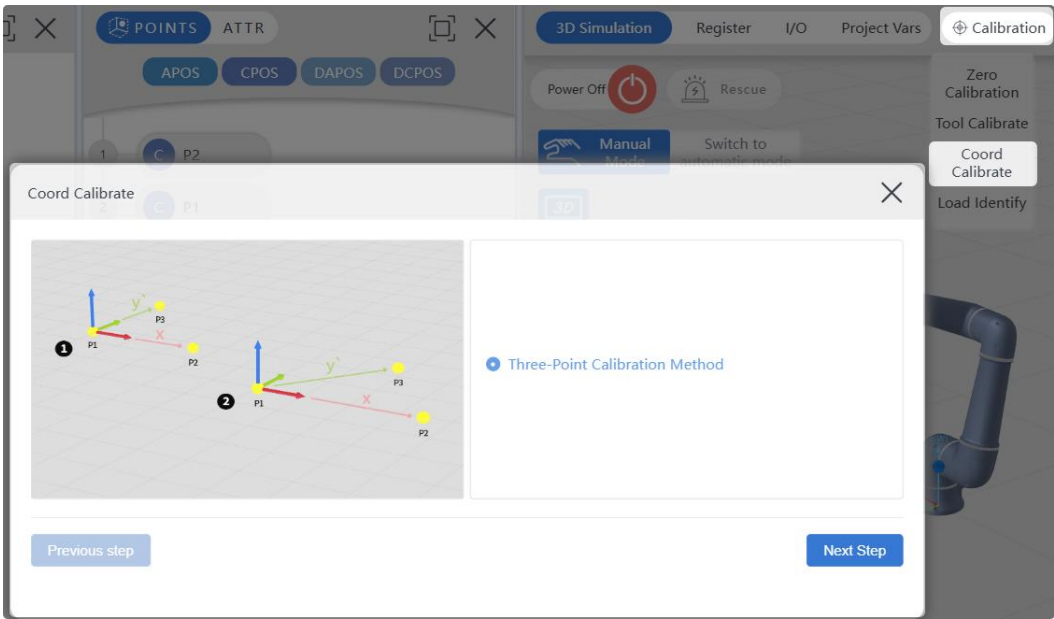
The installation of the robot can be selected from preset installation methods or customized with its installation offset and installation rotation relative to the world coordinate system.

9.3 Coordinate System and Calibration

Users can define the coordinate system. The user-defined coordinate system is offset based on the world coordinate system. The offset values can be directly input by users on the settings page or determined through the calibration function with assistance.

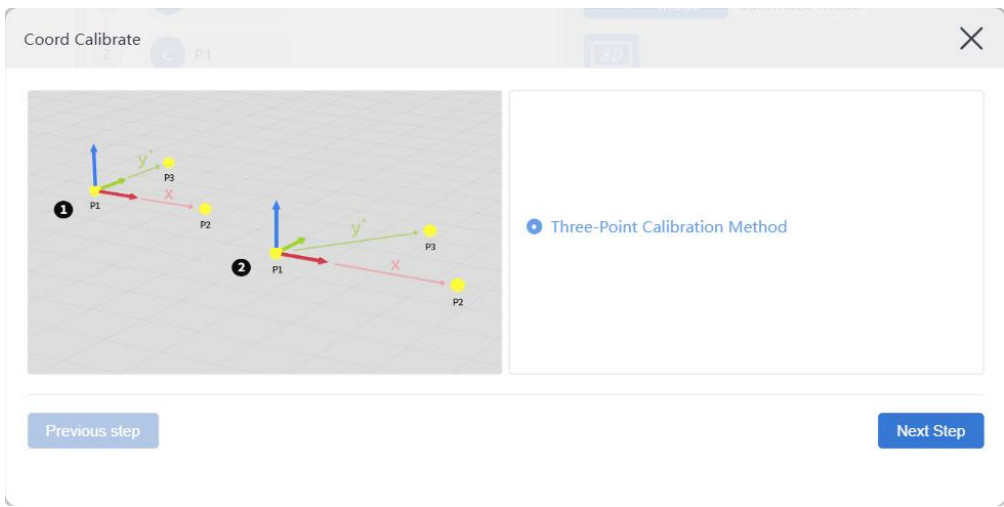


When it is necessary to calibrate the user coordinate system, the "Coordinate System Calibration" assistant can be used to create it.



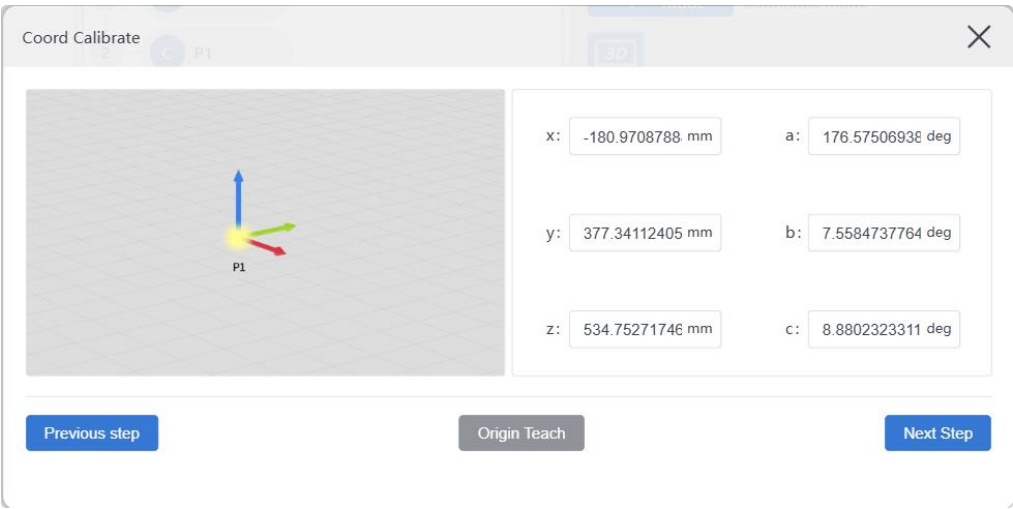
9.3.1 Three-point calibration method

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

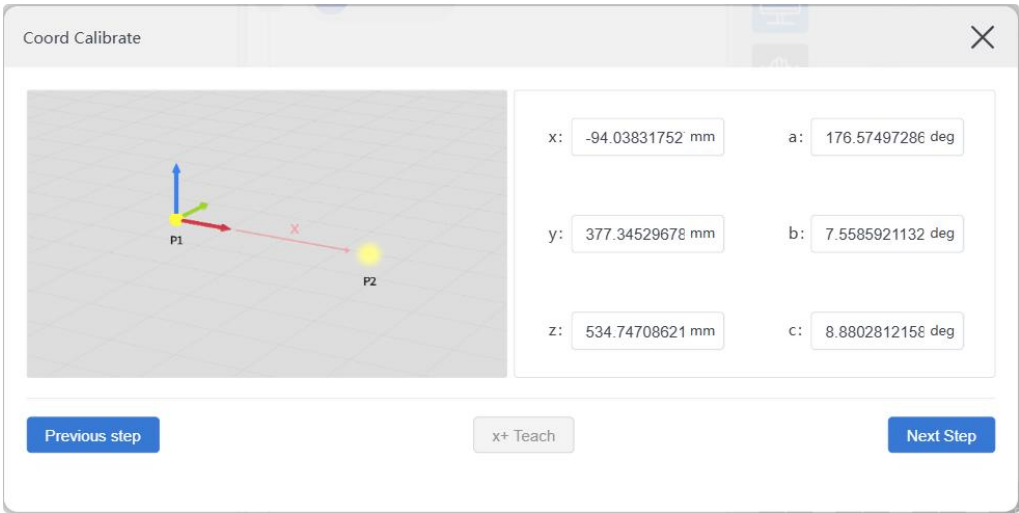


9.3.1.1 Start calibration

Define the origin of the user coordinate system. Move the robot's TCP to the origin of the coordinate system to be defined by point-to-point motion, click Origin Teach and then proceed to the next step.



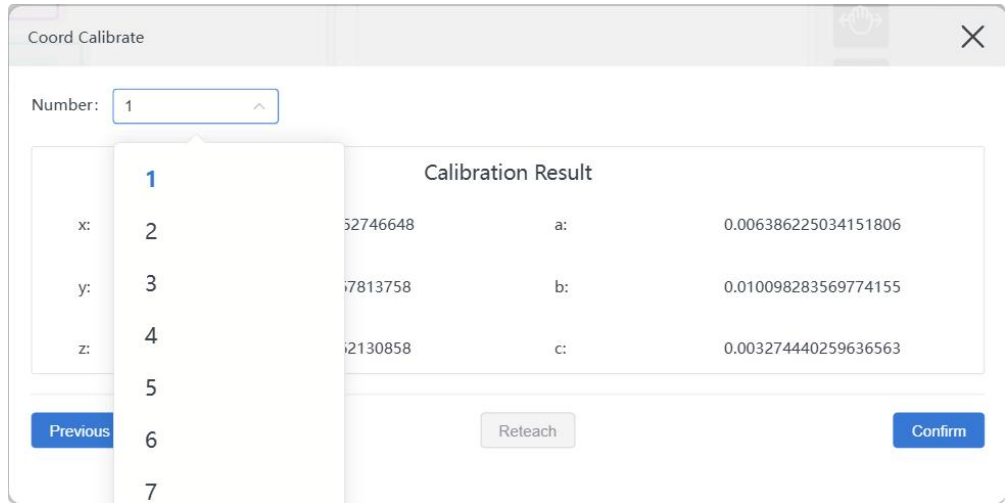
2. Define the positive x-direction of the user coordinate system. Move the robot's TCP to the positive x-direction of the coordinate system to be defined and click x+ Teach then proceed to the next step.



3. Define the y+ direction of the user coordinate system. Move the robot's TCP to the y positive direction of the coordinate system to be defined and click y+ Teach Then next step.

9.3.1.2 Calibration successful

After clicking the confirmation button, the specific values of the successfully calibrated coordinate system will be automatically filled in the selected coordinate system number.



9.3.1.3 Calibration failed

If the calibration result does not show any values and indicates calibration failure, please recalibrate and note that among the three points defined for calibration, namely the origin and x+ and y+, avoid having two or more identical points.

Coord Calibrate

2. Manual Mode

Switch to automatic mode

X

Number: 1

Calibration Result

x:

a:

y:

b:

z:

c:

Previous step

Retach

Confirm

i

User coordinate system calibration failed.

X

9.3.1.4 List of coordinate systems

In the Tools, Load, and Coordinate System options in the settings interface, all coordinate systems are recorded. Here, you can view or edit the values.

ESTUN
C O D R O I D

PROJECT

SETTING

RECORD

MANAGE

admin

⚠

🔍

Base

Tool, Payload, Coord...

Safety

Move

Register Communi...

IO

ModbusTCP Master

IO on Board

Tool, Payload, Coordinate

> Tool (Unit: kg,mm)

> Payload (Unit: kg,mm)

Coordinate (Unit: mm,deg)

Number	x	y	z	a	b	c
0	0	0	0	0	0	0
1	-94.03831752746	377.3452967813	534.7470862130	0.006386225034	0.010098283569	0.003274440259
2	0	0	0	10	20	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0

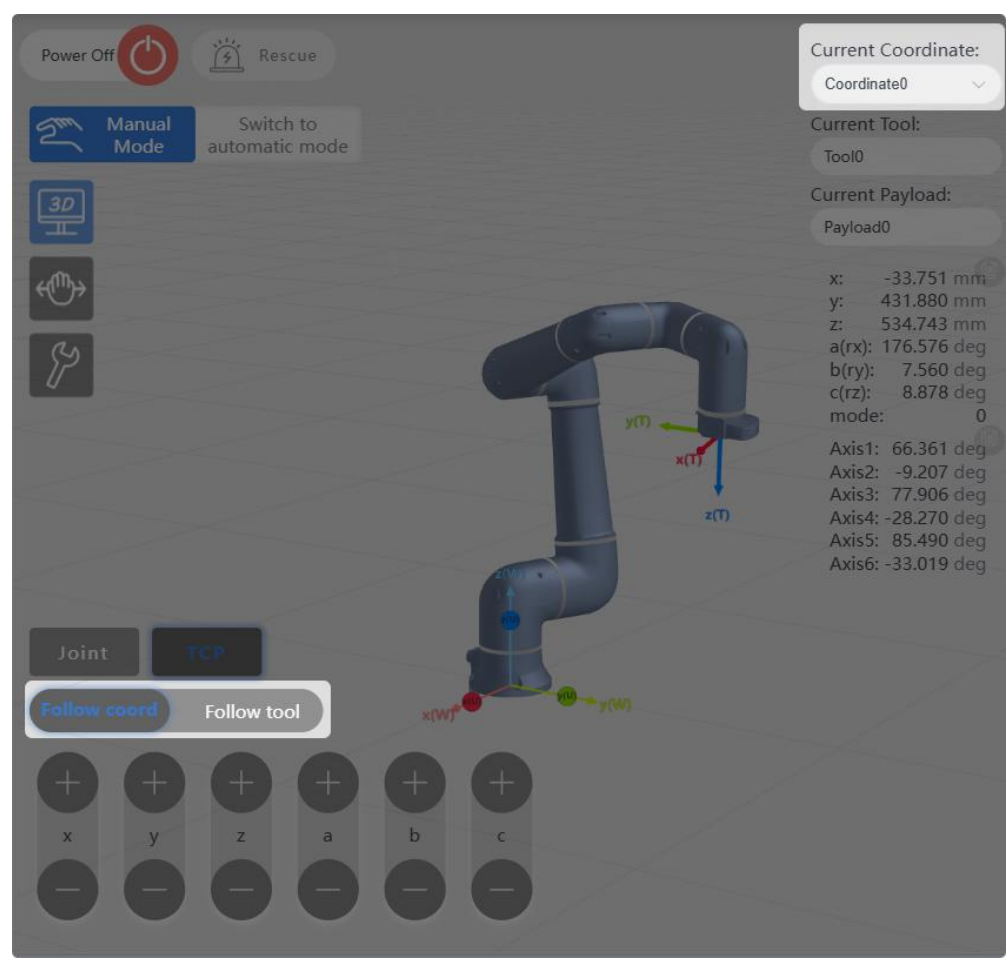
Save

9.3.2 Use the user coordinate system

9.3.2.1 Use the user coordinate system when jogging.

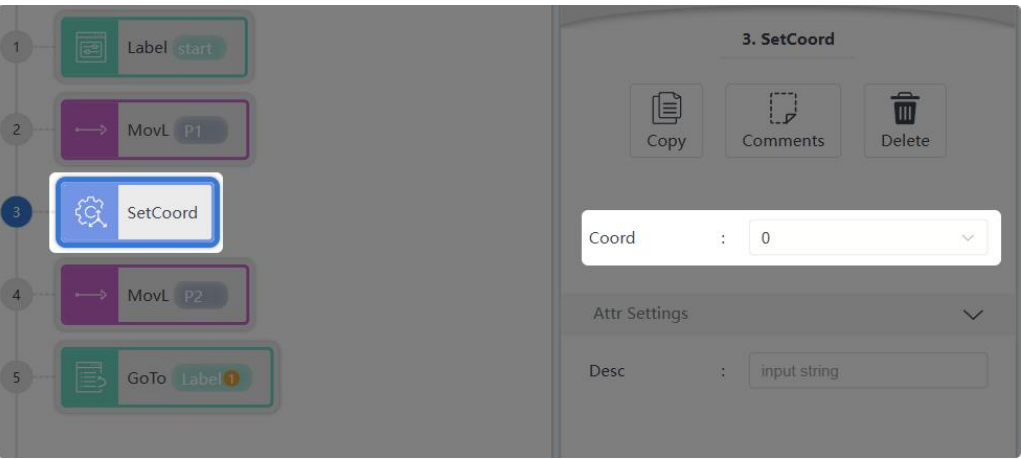
When performing point-by-point movement of the robot at the end, you can choose to move along the current coordinate system. If the current coordinate system is selected as the user coordinate system, the point-by-point movement can be carried out along the

user coordinate system.



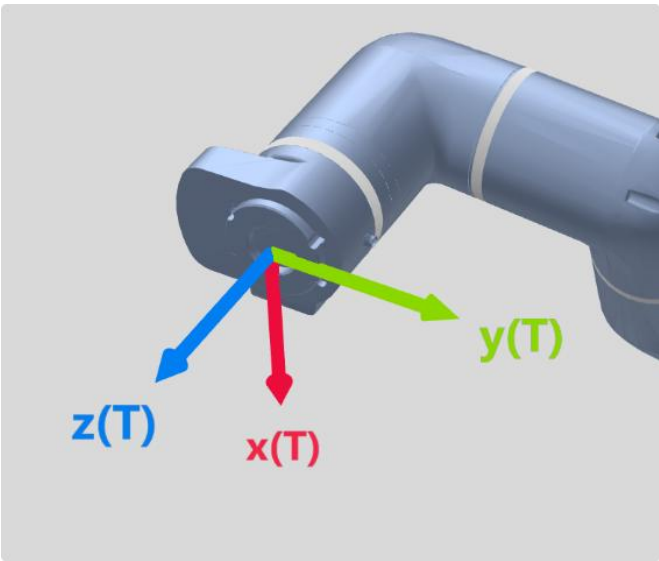
9.3.2.2 Switching coordinate systems in the program

Add the SetCoord command in the program tree and select the defined user coordinate system from the drop-down menu of "Coordinate System".

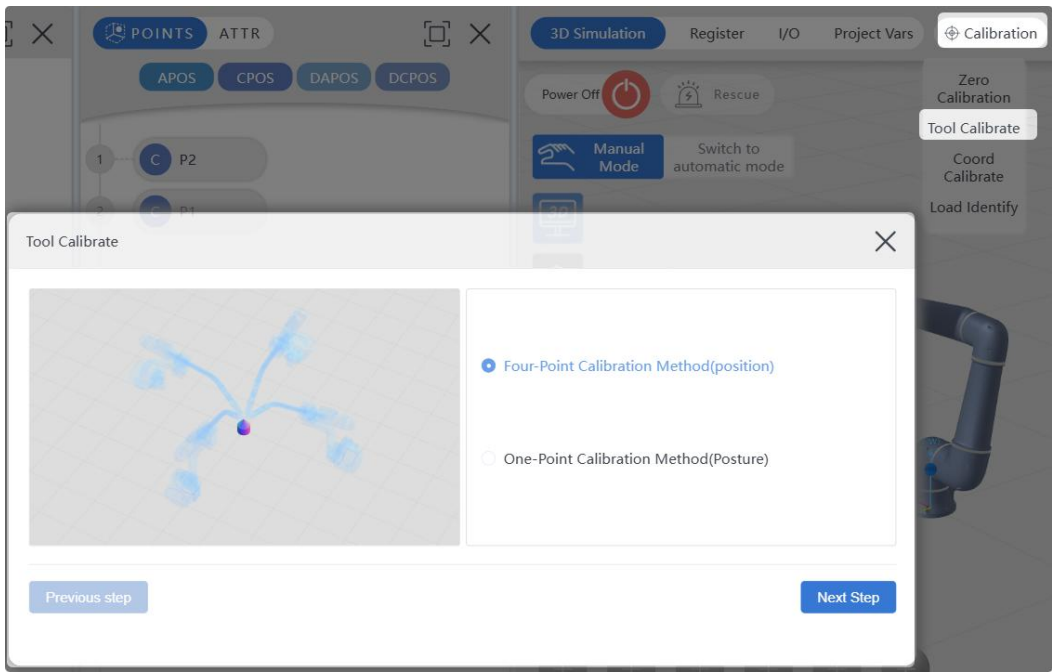


9.4 Tools and Calibration

Users can create new tool variables. The tool coordinate system is offset based on the default tool coordinate system (NOTOOL) at the end of the flange. The offset values can be directly input by the user or determined through auxiliary calibration. The origin of the default tool coordinate system is located at the center of the flange end, with the Z-axis pointing outward from the flange and the Y-axis pointing towards the installation locating pin hole.

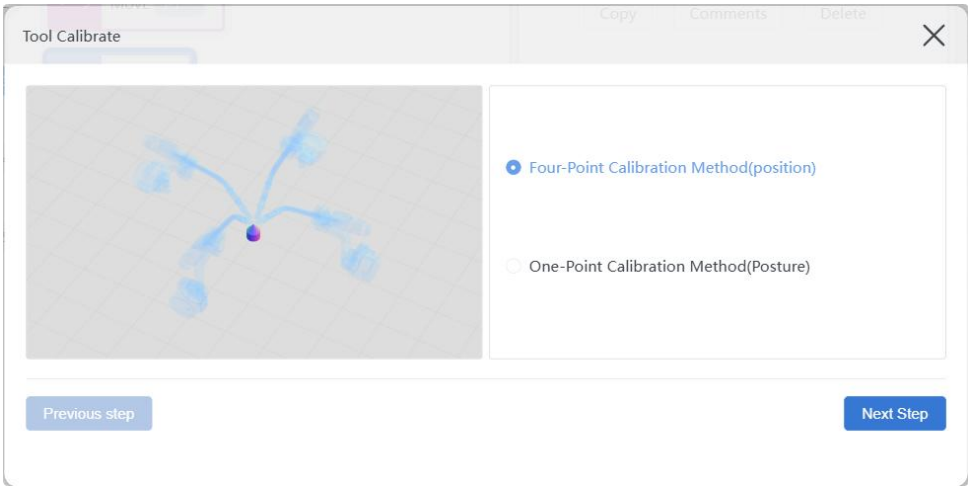


When it is necessary to calibrate the tool coordinate system, the four-direction calibration method of "Tool Calibration" can be used to assist in calculating the position offset, or the one-point calibration method can be used to assist in calculating the rotation angle.



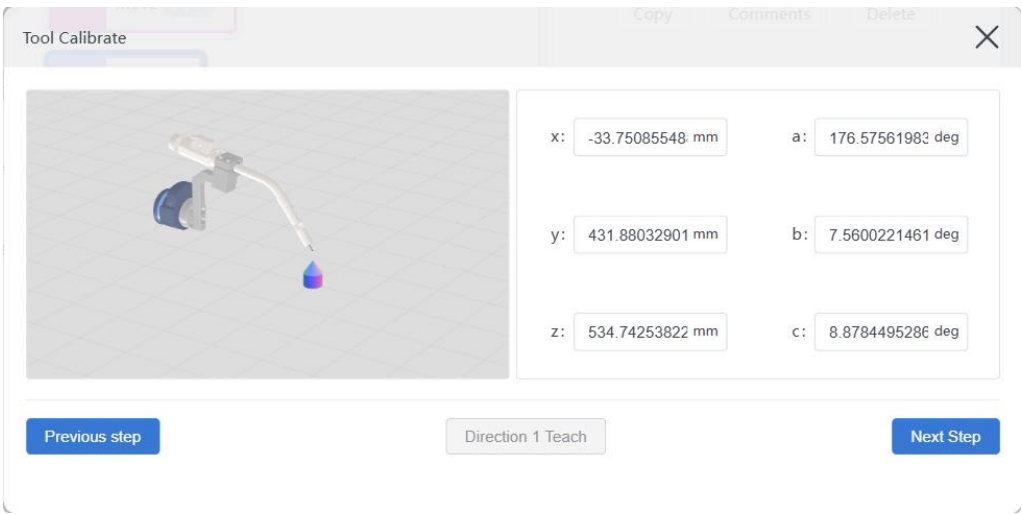
9.4.1 Four-direction calibration method

Manually move the robot (by point-to-point or dragging) to four different poses. Each time, make the tool tip touch the same needle tip placed in space and click the "Direction Teaching" button. After completing the four poses, the offset value of TCP relative to the center of the tool output flange can be obtained.



9.4.1.1 Start calibration

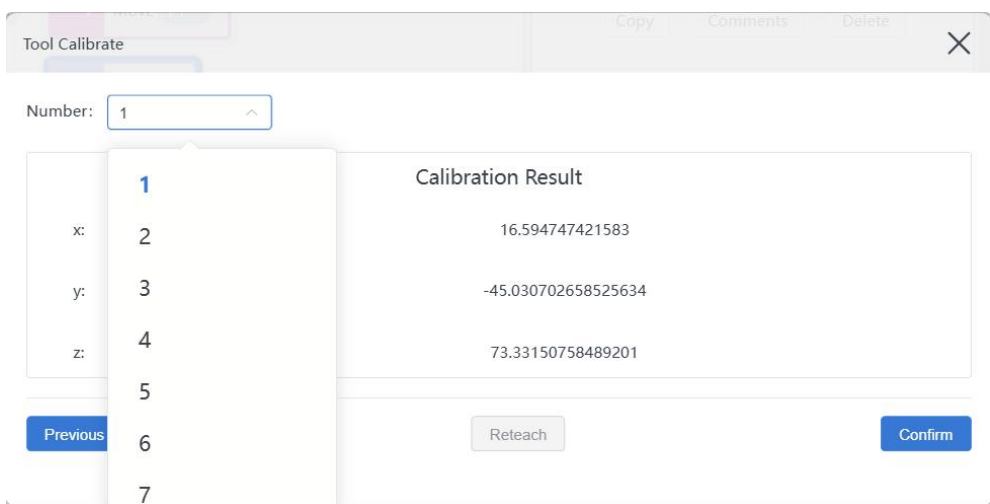
- a) The mobile robot brings the TCP (Tool Center Point) into contact with the tip of the needle placed in space.
- b) Click the **Direction 1 Teach** (orientation 1 teaching) button to record the current actual robot position.



- c) Click button **Next Step** (next) to repeat steps 1 and 2 until the fourth point, then click button **Confirm** (OK) to complete the direction instruction.

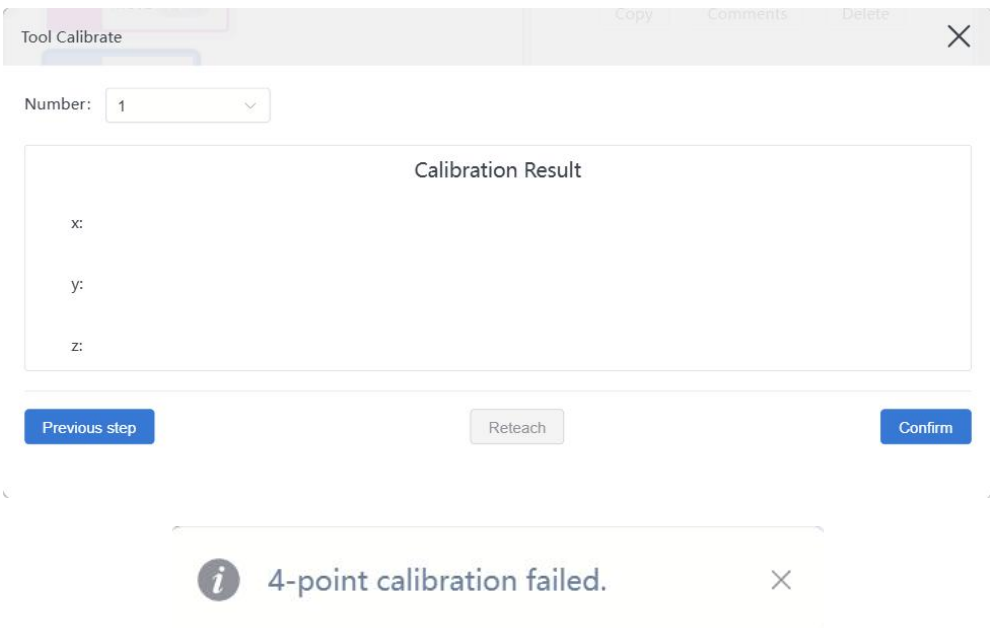
9.4.1.2 Calibration succussed

After clicking the "Confirm" button, the x, y, and z values of the successfully calibrated tool will be automatically filled in the selected tool number.



9.4.1.3 Calibration failed

If the "Calibration TOOL" window shows no result and prompts "4-point calibration failed", it indicates that the calibration has failed.

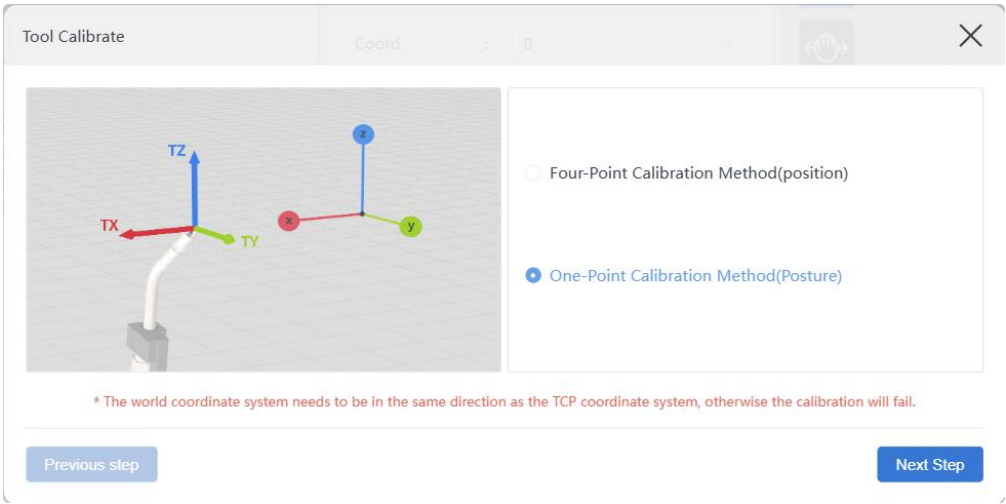


Please start the calibration again and make sure that:

- the four pose changes are large enough
- the needle tips are aligned (the tool center point is in sufficient contact with the needle tips in space).

9.4.2 One-point calibration method (attitude)

After completing the four-direction calibration method (to obtain the translational relationship of the TCP relative to the center of the tool output flange), the certain calibration method (attitude) can be started to obtain the rotational relationship of the TCP relative to the center of the tool output flange.



9.4.2.1 Start calibration

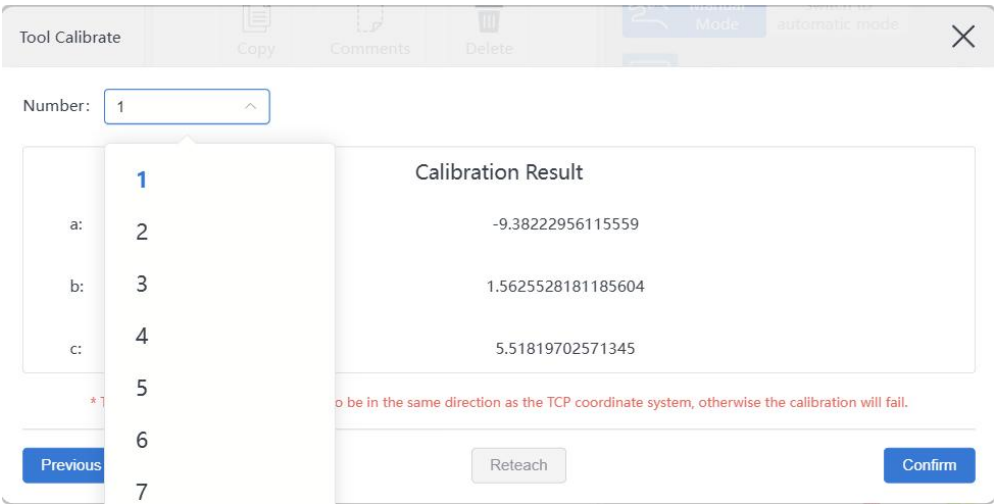
Move the robot so that the desired tool coordinate system orientation coincides with the robot's world coordinate system orientation, and click the 方向 示教 (orientation teaching) button to complete the Orientation Teaching.



9.4.2.2 Calibration results

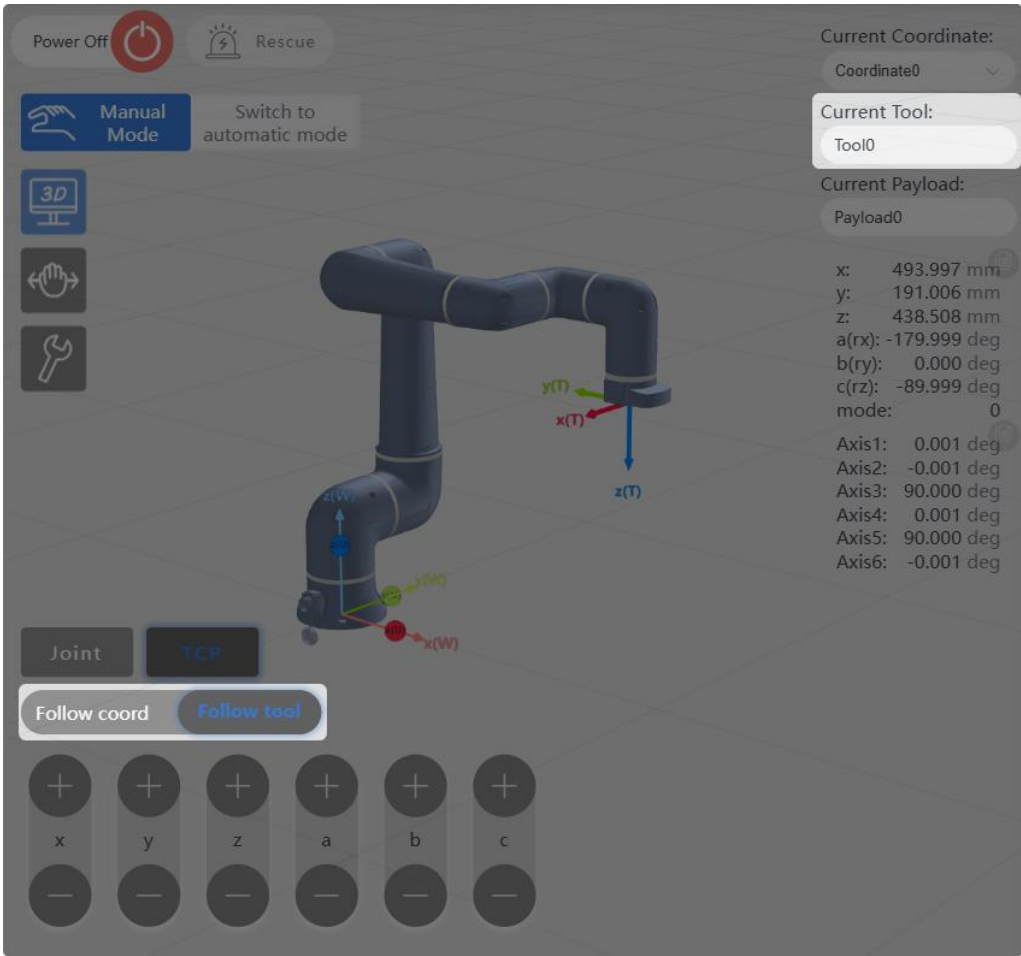
When calibrating the posture, the robot cannot verify the accuracy. The user can visually check by manually moving the tool coordinate system.

After obtaining the translation and rotation of the TCP (Tool Center Point) relative to the center of the tool output flange through the execution of the "Four-Point Calibration Method" and the "One-Point Calibration Method (Attitude)", a complete tool coordinate system is established, and the calibration is completed.



9.4.3 Use the tool coordinate system

9.4.3.1 Use the tool coordinate system when jogging.



When performing point-to-point movement of the robot at the end, you can choose to move along the tool coordinate system. Select the current tool as the target tool coordinate system to move along the tool coordinate system. The current tool can be switched in the Settings tab.

9.4.3.2 Tools used in the program

Add the "SetTool" instruction in the program tree and select the defined tool parameters from the drop-down menu.

1

Label start

2

3

SetTool

4

5

GoTo Label 1

3. SetTool

Copy

Comments

Delete

Tool : 1

Attr Settings

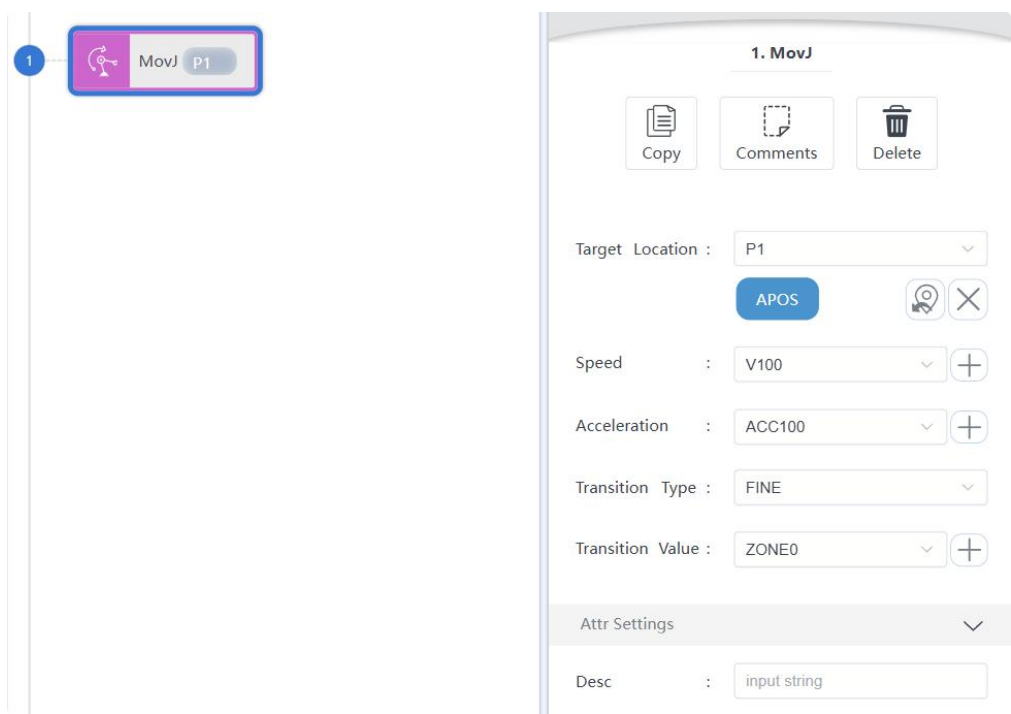
Desc : input string

Chapter 10 Instruction Introduction

10.1 Displacement Instructions

10.1.1 MovJ

This is joint movement. This instruction indicates that the robot's joints perform point-to-point motion, and the end trajectory of the robot is an irregular curve. Double-click the added MovJ instruction or select the parameters in the programming instruction details area and click MovJ to configure the instruction parameters.

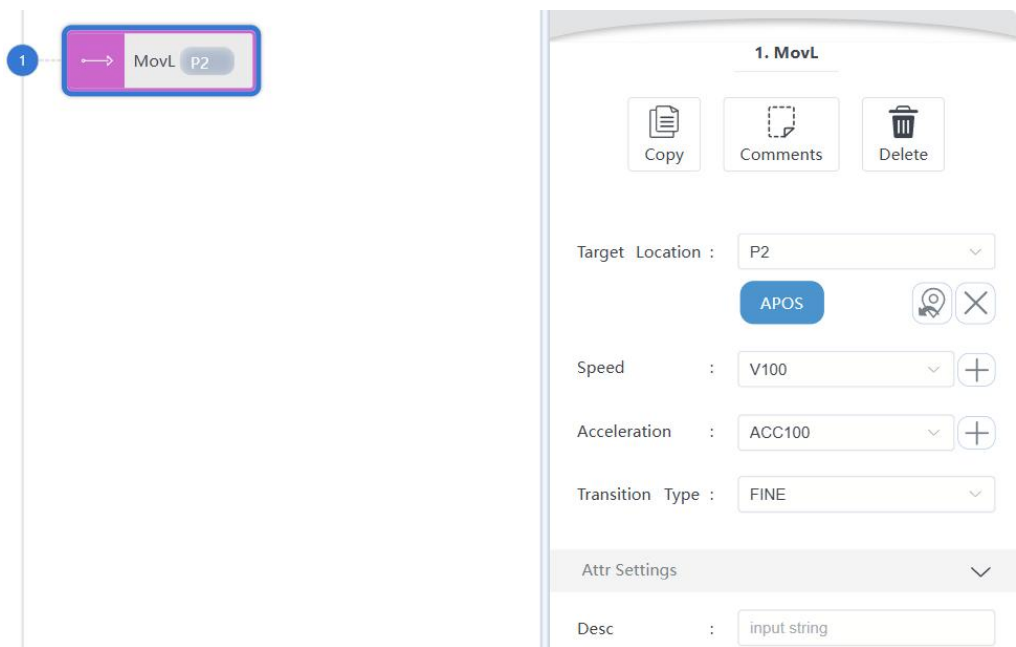


Parameter	Description
Target position	Points that have been shown and taught can be selected in the Target Position option; only APOS and CPOS can be added.
Target speed	Set as a SPEED type variable, you can choose the system predefined value or create your own; where the target speed is a percentage.
Acceleration	See Variable Management for details on creating and setting variables of type SPEED.
Transition Type	Set the variable as ACC, you can choose the predefined value or create it by yourself;
Transition Value	See Variable Management for details on creating and setting variables of type ACC.

10.1.2 MovL

The MovL instruction is a linear motion command. By using this command, the TCP point

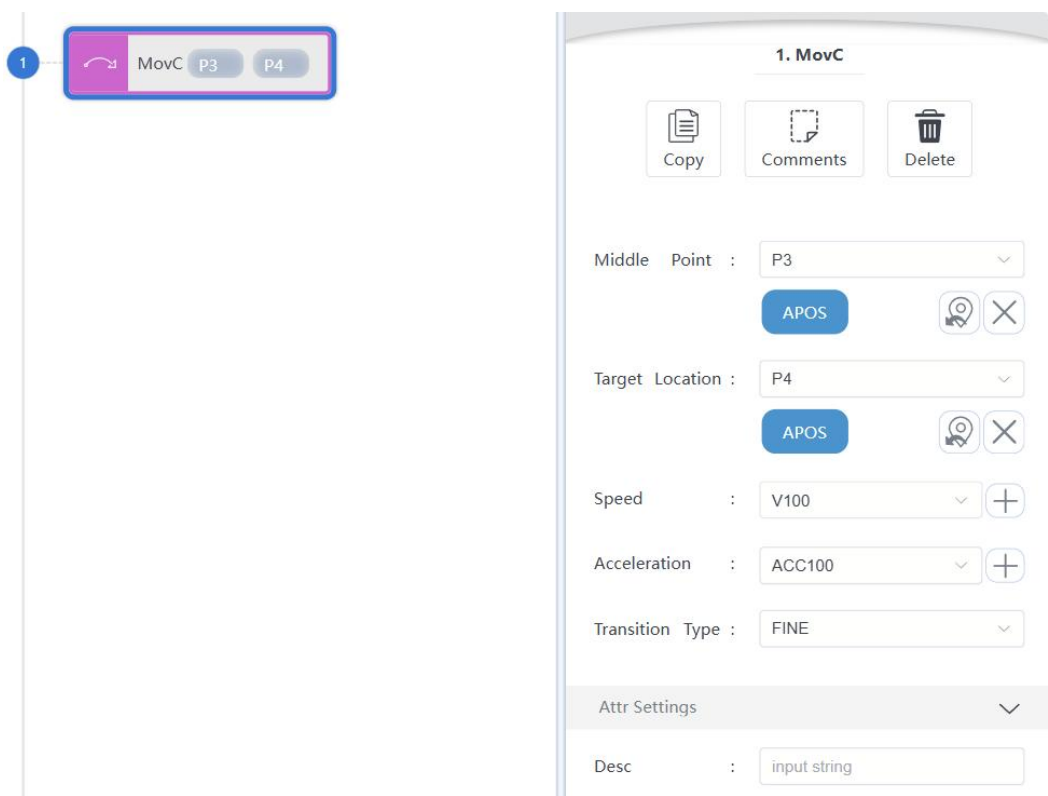
of the robot can move linearly to the target position at the set speed. If the starting and ending postures of the movement are different, the posture will rotate synchronously with the position to the ending posture during the movement. Compared with joint motion, linear movement may pass through singular points. Double-click the added MovL instruction or select the parameters in the programming instruction details area and click MovL to configure the instruction parameters.



Parameter	Description
Target position	Points that have been shown and taught can be selected in the Target Position option; only APOS and CPOS can be added.
Target speed	Set as SPEED type variable, you can choose the system predefined value or create it by yourself; in which, the target speed is an absolute value, unit mm/s.
Acceleration	The creation and setting of SPEED type variables are described in the Variables section.
Transition Type	The ACC variable can be predefined by the system or created by yourself;
Transition Value	See the Variable Management section for details on creating and setting variables of type ACC.

10.1.3 MovC

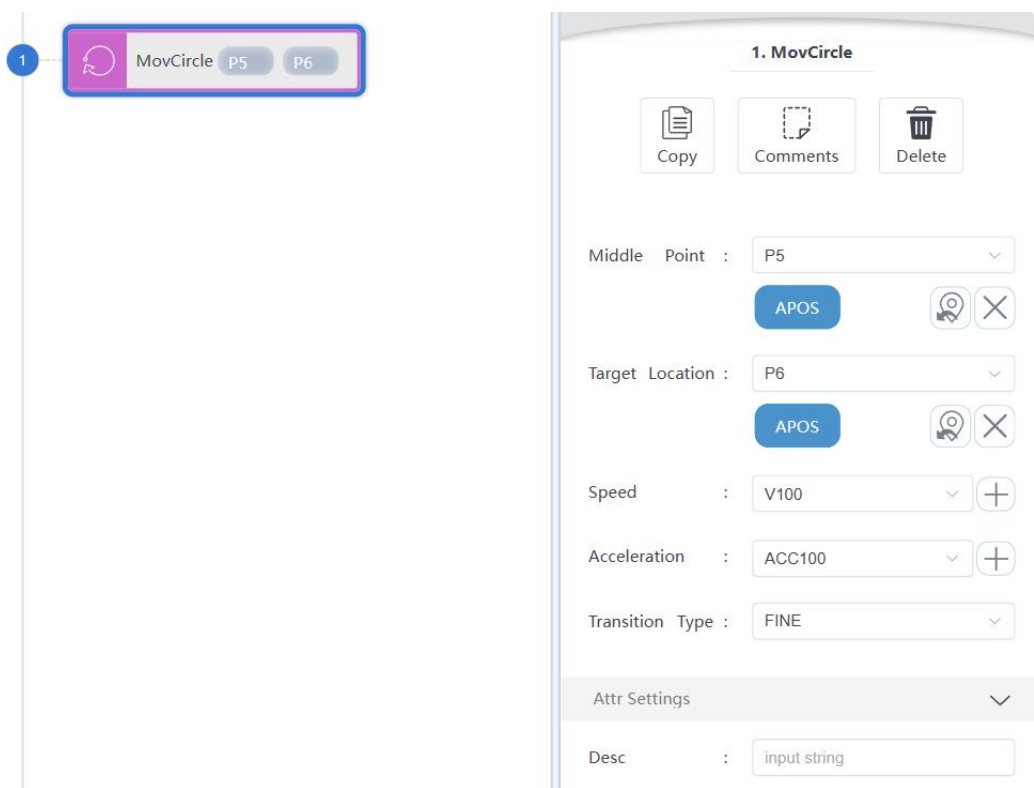
The completion of an arc instruction must involve three poses, and the positions of these three poses in space must not be on the same straight line. When using this instruction, the robot's TCP point moves in an arc from the starting position through the intermediate position to the target position. The starting position is the end point of the previous movement instruction. When using the MoveC instruction, if the starting and ending poses are different, the pose will rotate synchronously with the position during the movement to reach the ending pose, but it may not pass through the intermediate pose. Compared with joint movement, arc movement may pass through singular points. Double-click the added MoveC instruction or select the parameters in the programming instruction details area and click MoveC to configure the instruction parameters.



Parameter	Description
Intermediate position	The position of the middle auxiliary point of the arc, the type can only be APOS or CPOS.
Target position	The position of the end point of the arc, the type can only be APOS or CPOS.
Target speed	Set as SPEED variable, you can choose the system predefined value or create it by yourself; in which, the target speed is an absolute value, the unit is mm/s.
Acceleration	The creation and setting of SPEED type variables are described in the Variables section.
Transition Type	The ACC variable can be predefined by the system or created by yourself;
Transition Value	See the Variable Management section for details on creating and setting variables of type ACC.

10.1.4 MovCircle

The full circle instruction refers to the movement of the robot's TCP point from the starting position to the target position via an intermediate position, with the three positions in the pose space not being collinear. When using this instruction, the robot's TCP point performs a full circle movement from the starting position to the target position via the intermediate position, and the posture remains unchanged during the full circle movement. Compared with joint movement, the full circle movement may pass through singular points. Double-click the added MovCircle instruction or select the parameters in the programming instruction details area and click MoveCircle to configure the instruction parameters.



Parameter	Description
Intermediate position	The position of the middle auxiliary point of the arc, the type can be APOS or CPOS.
Target position	The position of the end point of the arc, the type can be APOS or CPOS.
Target speed	Set as SPEED type variable, you can choose the system predefined value or create it by yourself; among them, the target speed is an absolute value, unit mm/s.
Acceleration	The creation and setting of SPEED type variables are described in the Variables section.
Transition Type	The ACC variable can be predefined by the system or created by yourself;
Transition Value	For details on the creation and setting of ACC type variables, see Variable Management.

10.1.5 MovJRel

MovJRel is an interpolation relative offset instruction. This instruction always takes the current robot position or the target position of the previous motion instruction as the starting position, and then the robot moves relatively by the specified offset.



1. MovJRel

Copy

Comments

Delete

Target Location :

P7

DAPOS

×

Speed :

V100

+

Acceleration :

ACC100

+

Transition Type :

FINE

▼

Transition Value :

ZONE0

+

Attr Settings

▼

Desc :

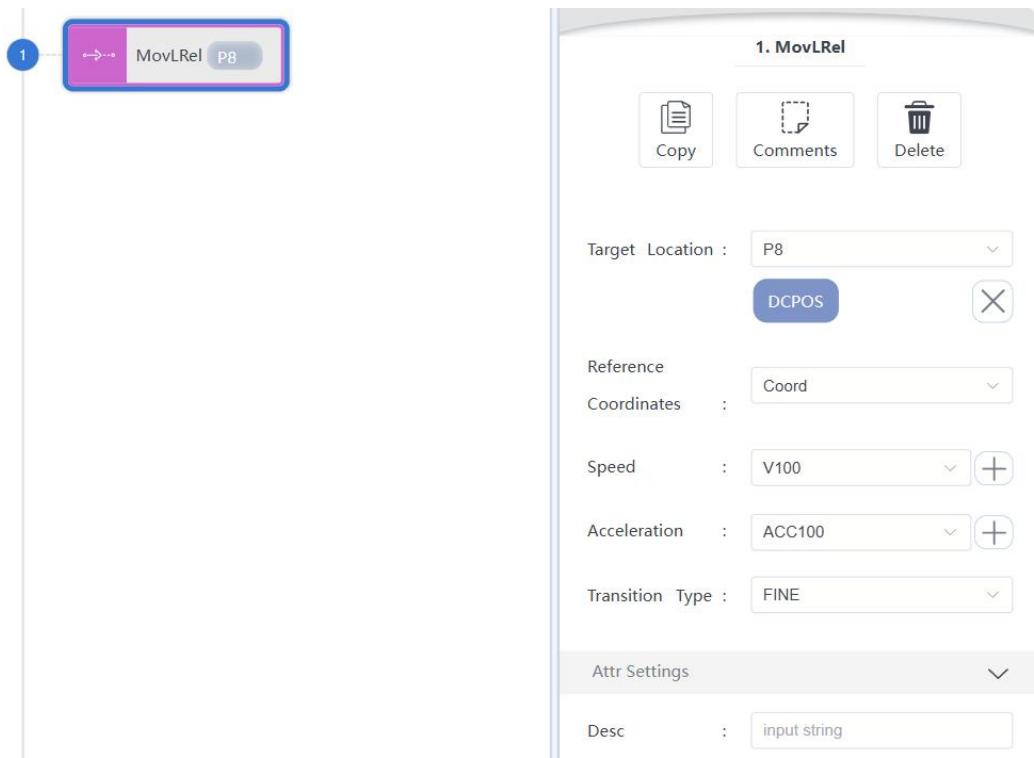
input string

Parameter	Description
Target Relative Position	The position increment that the robot is to move when executing this command can only be added to DAPOS.
Target speed	The setting is a variable of type SPEED, which can be either a system predefined value or created by yourself; where the target speed is a percentage.
Acceleration	See the Variables section for details on creating and setting variables of type SPEED.
Transition Type	Set the variable to ACC, which can be predefined by the system or created by yourself;
Transition Value	See the Variable Management section for details on creating and setting variables of type ACC.

10.1.6 MovLRel

MovLRel interpolation relative offset instruction. This instruction always takes the current robot position or the target of the previous motion instruction as the reference.

The position is the starting position, and then the robot performs an offset movement relative to the coordinate system or the tool.



Parameter	Description
Target position	The position increment that the robot is to move when this command is executed can only be added to the DCPOS.
Reference coordinates	Coordinate system offset or tool offset selection;
Target speed	- Coord: offset relative to the current user coordinate system;
Acceleration	- Tool: offset relative to the tool coordinate system, i.e. reference Tx, Ty, Tz translation or rotation.
Transition type	Set as SPEED type variable, you can choose the system predefined value or create it by yourself; among them, the target speed is the absolute value, the unit is mm/s.
Transition value	The creation and setting of SPEED type variables are described in the Variables section.

10.1.7 MovLSearch

The positioning command refers to performing IO detection or torque detection when executing this MovL command.

1

MovLSearch

P9

1. MovLSearch

Copy

Comments

Delete

Target Location :

P9

APOS

📍

✕

Speed :

V100

+

Acceleration :

ACC100

+

Test Type :

DITrig

Search Index :

0

Search Value :

0

Stop Time :

0

ms

Result Value :

Please select

+

Search Pos :

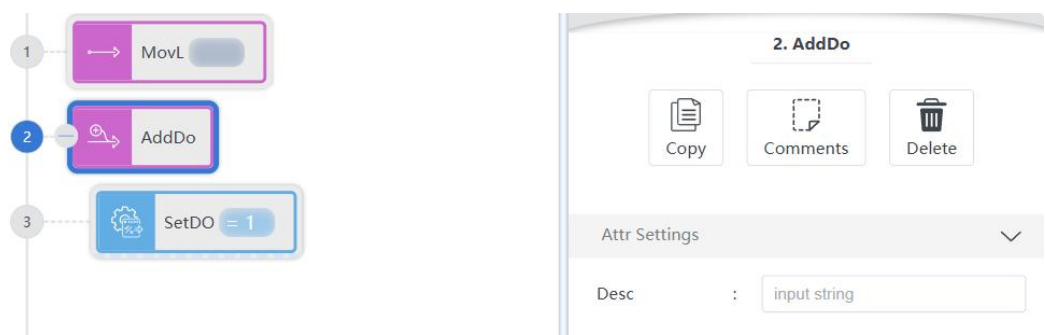
Please select

Parameter	Description
Target position	You can select the points that have been shown and taught in the target position option; only APOS and CPOS can be added, DAPOS and DCPOS are not selectable.
Target speed	Set as SPEED type variable, you can choose the system predefined value or create it by yourself; among them, the target speed is the absolute value, unit mm/s.
Acceleration	The creation and setting of SPEED type variables are described in the Variables section.
Detection Type	The ACC variable can be predefined by the system or created by yourself;
Trigger Index	For details on the creation and setting of ACC type variables, see the Variable Management.
Trigger Value	DITrig: Physical digital IO input detection.
Deceleration time	AITrig: Physical analog IO input detection.
Return Value	For InputTrig, this parameter indicates the IO port number to be detected.
Successful position	For TorqTirg, this parameter indicates the axis number to be detected.
Jump node	For InputTrig, the threshold for IO detection. For TorqTirg, the threshold for torque detection in thousandths of the rated torque.

10.1.8 AddDo

The AddDO instruction must be placed after motion instructions, including MovJ, MovL, MovC, MovCircle, MovJRel, and MovLRel. This instruction is mainly used to ensure that the transition between two motion instructions is not interrupted. If AddDO is added between two motion instructions, the IO operations in its sub-controls will not interrupt the transition; otherwise, the transition value of the previous motion instruction will not take effect.

After the robot has executed this instruction, it can perform IO operations. The AddDO instruction must have sub-controls added, and the sub-controls can only be: SetDO, SetAO, SetSimDO, SetSimAO, SetDO8421, SetSimDO8421.




10.1.9 MovTraj

Run the specified drag trajectory. Before running the trajectory, the robot must be at the starting point of the trajectory. You can use the GetTrajStartPoint instruction to obtain the starting point position and use the MovJ instruction to move to that point.

Parameter	Description
Trajectory	The track to be run.
Speed multiplier	The running speed multiplier to run the trajectory, based on the speed at the time of dragging and dropping.



To run a trajectory you can use the  button on the shortcut menu bar to open the trajectory editor. After clicking “Enable Trajectory Recording”, press the button of Drag & Drop in manual mode to start recording the robot’s trajectory, release the button to stop recording, and click the Save button to save the most recent drag & drop trajectory. After selecting a trajectory, long press “Run to Start” to run to the starting point of the trajectory, and long press “Run Trajectory” to run the trajectory.

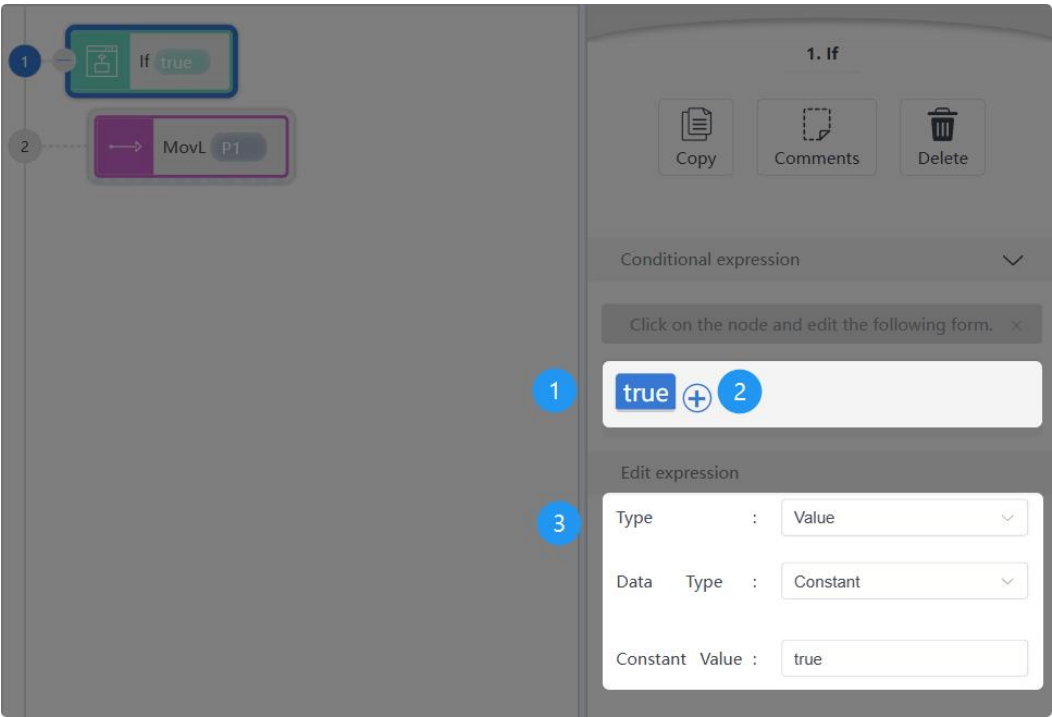
10.2 Logical Instructions

10.2.1 GoTo

The GOTO instruction is used to jump to different parts of a program.

10.2.2 If

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

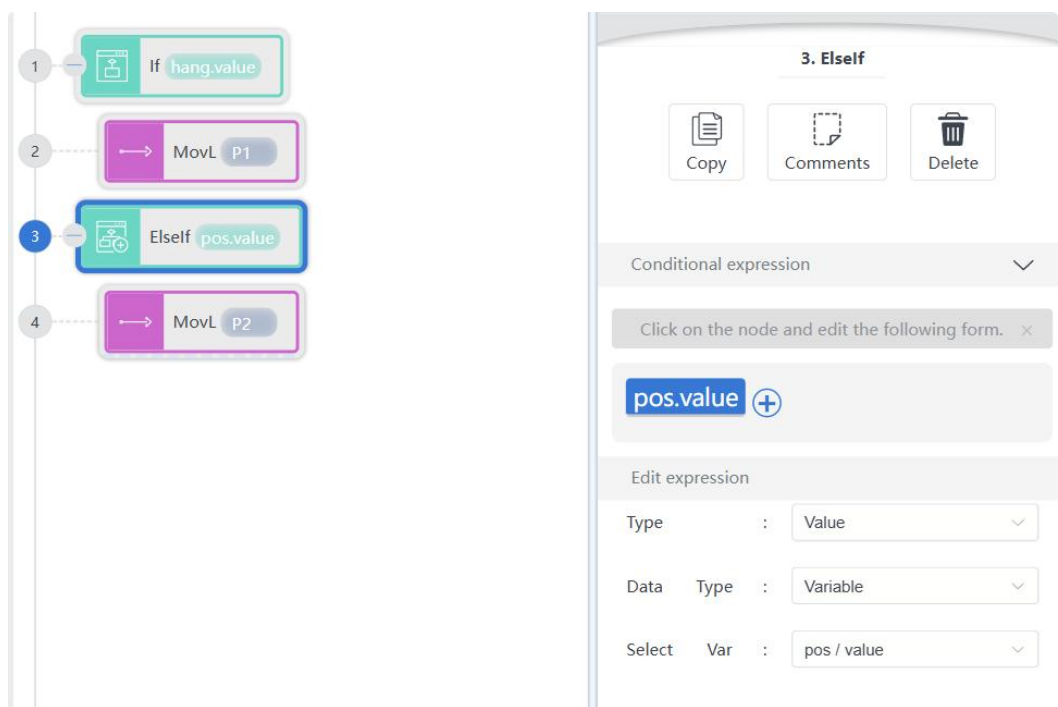


In the figure, 1 represents the overall expression, and you can add expressions by selecting the + sign in the 3 boxes within the frame. 2 is the expression currently being edited. The parameter editing of the expression is as follows:

Expression type	Description
value	Includes constant values and variable values. Constant values currently only support numeric quantities with true and false, while variables can choose from the basic variables provided, and currently include all IO signals.
operator	Operators include logical operators with or without and various math operators such as addition, subtraction, multiplication and division.
function	Provide commonly used math functions including sine, cosine, integer, remainder and other functions.

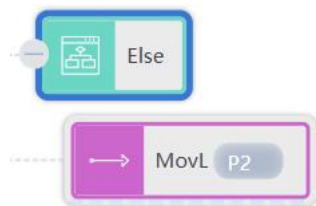
10.2.3 Elself

The ELSIF instruction depends on the IF instruction and follows immediately after the IF control. When the IF logic is not satisfied, the ELSIF logic is evaluated. The method of setting the ELSIF expression is the same as that of setting the IF expression.



10.2.4 Otherwise

The ELSE instruction depends on the IF or ELSIF instruction and follows immediately after the IF or ELSIF control. It is executed when the conditions of the IF or ELSIF instructions are not met. The ELSE instruction has no parameter configuration.



10.2.5 While

The WHILE instruction repeatedly executes the sub-statement when the condition is met. The loop control expression must be of the BOOL type.

The parameters of the expression are edited as follows:

1

While hang.value

2

MovC P4 P3

3

SetTool

4

MovL

1. While

Copy

Comments

Delete

Conditional expression

Click on the node and edit the following form. x

hang.value +

Edit expression

Type

:

Value

Data

Type

:

Variable

Select

Var

:

hang / value

Expression type	Description
value	Includes constant values and variable values. Constant values currently only support numeric quantities with true and false, while variables can choose from the basic variables provided, and currently include all IO signals.
operator	Operators include logical operators with or without and various math operators such as addition, subtraction, multiplication and division.
function	Provide commonly used math functions including sine, cosine, integer, remainder and other functions.

10.2.6... =...

Create an expression to assign a value to a certain variable. Currently, the assignment instruction supports assigning values to all IO and variables of INT, BOOL, and REAL types. Its configuration interface is as follows:

1

... = ... INT0.value=5

1. ... = ...

Copy

Comments

Delete

Conditional expression

Click on the node and edit the following form. x

INT0.value = 5 +

Edit expression

Type

:

Value

Data

Type

:

Constant

Constant

Value

:

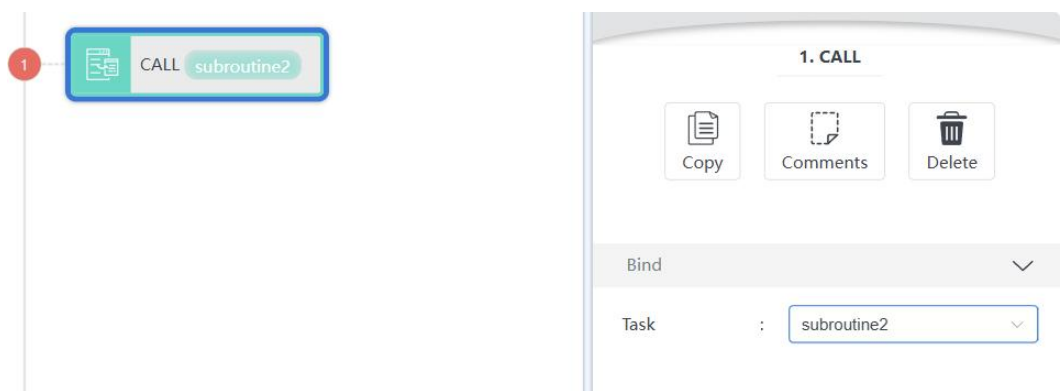
5

10.2.7 RETURN

Return instruction. Generally, after executing this instruction, the program jumps to the end of the program. If the RETURN instruction is used in a subroutine called by the CALL instruction, it returns to the program one level above the CALL instruction. For example, if the RETURN instruction is used in a subroutine called by the main program, it will return to the main program.

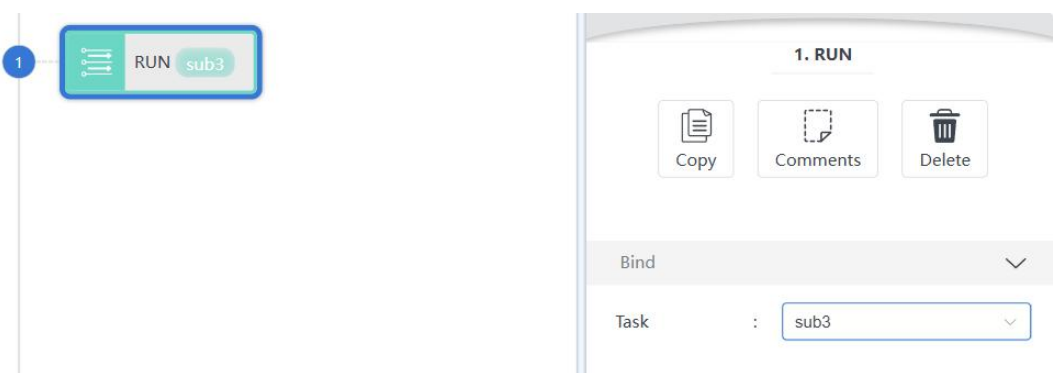
10.2.8 CALL

The call instruction makes the current program jump to another subroutine within the same project. After the subroutine is executed, it jumps back to the current program. Clicking on the name of the current program will bring up a drop-down menu where you can switch the currently edited program.



10.2.9 RUN

The sub-task parallel running instruction enables the robot to run sub-tasks in parallel while performing the main task. The tasks to be run must be in the same project.



10.2.10 KILL

The instruction to stop concurrently running programs (tasks) enables the robot to halt other programs while running the current one, and the programs to be stopped must be in the same project and in a running state.

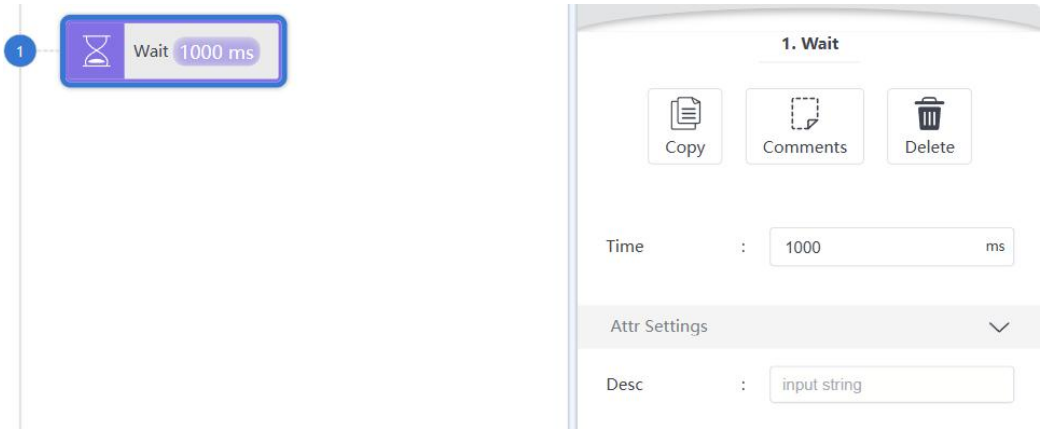
10.2.11 Labeling

The Label instruction is used to define the target for GOTO jumps.

10.3 Flow Control Instructions

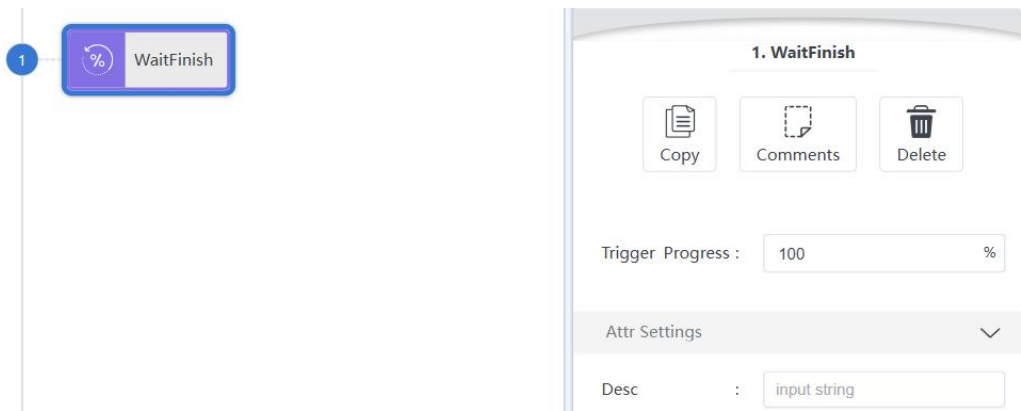
10.3.1 Wait

It is used to set the waiting time for the robot, with the time unit being milliseconds and it can be of the int constant type.



10.3.2 WaitFinish

It is used to synchronize the robot's movement and program execution. The robot will directly transition to execute the next instruction when the previous instruction reaches the trigger progress. When adding the WaitFinish instruction, sub-controls must be added. Sub-controls can be SetDO or SetAO and their similar instructions. As shown in the example program, the SetDO instruction will be triggered when the first MovL instruction reaches 20% of its execution. After the first MovL instruction is completed, it will directly transition to execute the second MovL instruction. If the WaitFinish instruction is deleted, the SetDO will be executed after the first MovL instruction is completed, and then the second MovL instruction will be executed. In this case, there is no transition between the two MovL instructions, and there will be a pause.



Parameter	Description
Trigger progress	Percentage of runtime when the previous move instruction triggers a child control in WaitFinish.

10.3.3 WaitCondition

Set the conditions for the robot to wait. If the conditions are not met within the set time, a timeout status will be returned. The next instruction will be executed only when the "discrimination condition" is true; otherwise, the program will continue to wait until the expression is true.

1

MovL P1

2

WaitCondition true 1000 ms Label 5

3

Wait 1000 ms

4

MovL P2

5

Label continue

2. WaitCondition

Copy

Comments

Delete

Time

:

1000

ms

Interrupt

:

Reset Timer

Timeout

:

hang

Bind

jump

node

:

Label

Conditional expression

Click on the node and edit the following form.

true

Edit expression

Type

:

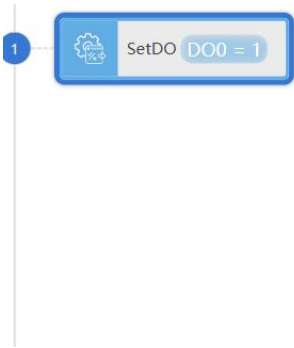
Value

Parameter	Description
Time	The amount of time, in ms, required to execute the wait.
Timeout value	If the value of this parameter is 0, it will force the system to wait until the discriminant condition is true before continuing to execute the next instruction.
Jump Node	If the value of this parameter is non-zero, the system will skip the instruction and continue to execute the next instruction after waiting for the given amount of time, even if the discriminating condition is still not true.
Conditional expression	Select a variable and assign a value to it under the following two conditions.

10.4 IO Instructions

10.4.1 SetDO

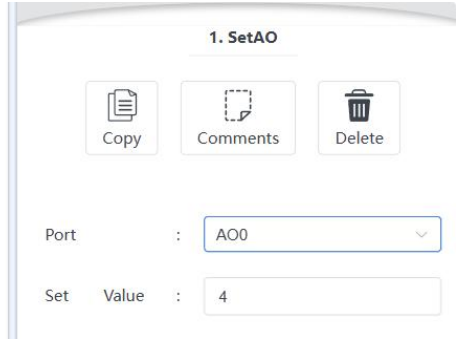
Set the digital output ports to the TRUE (1) or FALSE (0) state. Here, DO0-DO15 represent the 16 digital output ports of the control cabinet, DI0-DI15 represent the 16 input ports of the control cabinet, and switch0-switch3 indicate the status of the buttons at the robot's end effector.



Parameter	Description
Port	Sets the port number of the digital output DO.
Setting value	Sets the port value, with 0 indicating a high level and 1 indicating a low level.

10.4.2 SetAO

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



Parameter	Description
Port	Sets the port number of the digital output AO.
Setting value	Set port value, only current is supported, range 4mA-20mA

10.4.3 WaitDI

This instruction is used to wait for the status of a digital input (DI) port for a specified duration. If the waiting condition is met within the set duration, the program continues to execute downward; if the condition is not met within the set duration, the timeout judgment value is set to 1 and the program jumps to the jump node.

1

WaitDI 1000 ms Label 2

2

Label wait

1. WaitDI

Copy

Comments

Delete

Port : DI0

Port Value : High Level

Time : 1000 ms

Interrupt : Reset Timer

Timeout : hang

Bind

jump node : Label

Parameter	Description
Port Variables	Input port number to wait for.
Port Value	Digital input port level to wait for.
Duration (ms)	The unit of time to wait for the signal to change is ms.
Timeout value	Return the result of instruction execution to the variable set in the timeout value, the timeout value variable can only be INT type variable.
Jump node	When a signal is successfully waited for within the waiting time, the running value of the timeout value will be set to 0;

10.4.4 WaitDI8421

This instruction is used to wait for a combination of states of a group of consecutive digital input (DI) ports within a specified duration. If the waiting condition is met within the set duration, the program continues to execute downward; if the condition is not met within the set duration, the timeout judgment value becomes 1, and the program jumps to the jump node.

1

WaitDI8421 1000 ms Label 2

2

Label wait

1. WaitDI8421

Copy

Comments

Delete

Start Port : Please select

End Port : Please select

Time : 1000 ms

Interrupt : Reset Timer

8421 Value : 0

Timeout : hang

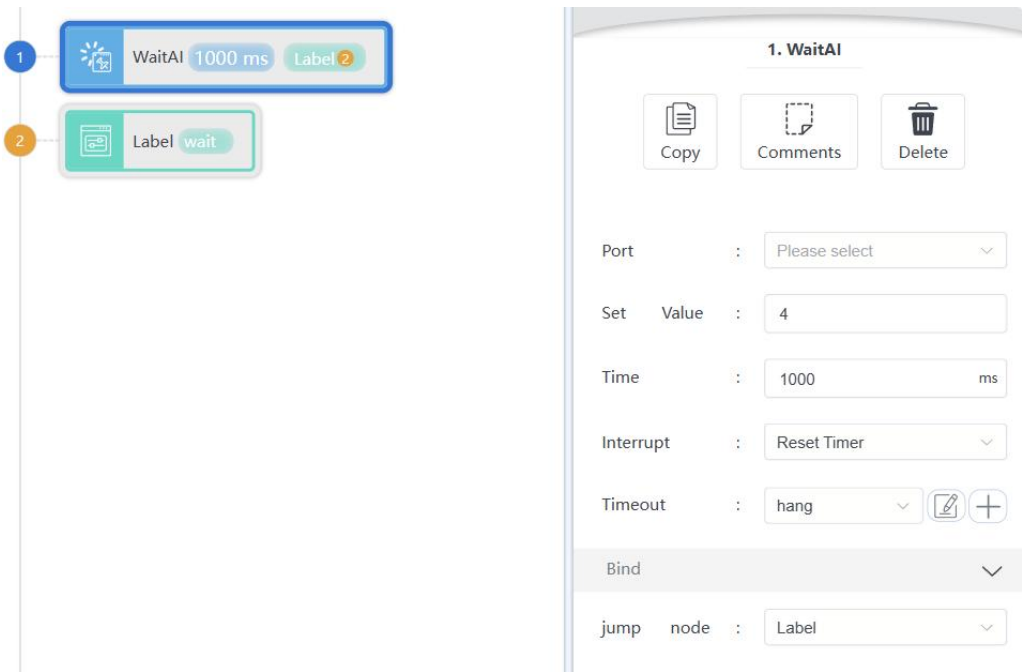
Bind

jump node : Label

Parameter	Description
Starting Port	Starting port number of the continuous DI port for the segment, indicating the low bit of the 8421 conversion value
End Port	End port number of the continuous DI port for the segment. Indicates the high bit of the 8421 conversion value
Time(ms)	The length of time to wait for the DI signal of this group of digital inputs, which can be of int constant type in ms.
8421 Value	Convert the value of this segment of consecutive DI ports to a decimal number according to the 8421 rule, and the condition is considered satisfied if it is equal to the VALUE value.
Timeout value	For example, if the start port is 0 and the end port is 2, and the 8421 value is set to 4, the condition is said to be satisfied when DI0 is 0, DI1 is 0, and DI2 is 1.
Jump Node	The result of the instruction execution is returned to the variable set in the timeout value, and the timeout value variable can only be an INT type variable.

10.4.5 WaitAI

This instruction is used to wait for the status of an analog input (AI) port for a specified duration. If the waiting condition is met within the set duration, the program continues to execute downward; if the condition is not met within the set duration, the timeout judgment value is set to 1 and the program jumps to the jump node.



Parameter	Description
Port Variables	The analog input port number to be waited for.
Port Value	The analog input port current value (4mA-20mA) to wait for.
Duration (ms)	The unit of time to wait for signal conversion is ms.
Timeout value	Returns the result of command execution to the variable set in the timeout value, which can be an INT type variable only.
Jump node	When a signal is successfully waited for within the waiting time, the running value of the timeout value will be set to 0;

10.4.6 GetDI8421

This command is used to obtain the status of a continuous section of DI ports (treated as binary data) and return it as a decimal number.

1

GetDI8421

1. GetDI8421

Copy

Comments

Delete

Start

Port

:

DI0

▼

End

Port

:

DI2

▼

Return

Value

:

int1

▼

+

Parameter	Description
Starting Port	Start DI port number you want to acquire, the lowest bit of the 8421 value.
End Port	The end DI port number you want to get, the highest bit of the 8421 value.
Return Value	INT type variable, the program runtime will get the port status in binary to decimal conversion and then pass into the int variable.

10.4.7 GetDO8421

This command is used to obtain the status of a continuous section of DI ports (treated as binary data) and return it as a decimal number.

1

GetDO8421

1. GetDO8421

Copy

Comments

Delete

Start

Port

:

DO0

▼

End

Port

:

DO3

▼

Return

Value

:

int2

▼

+

Parameter	Description
Starting Port	Start DO port number you want to acquire, the lowest bit of the 8421 value.
End Port	The end DO port number you want to get, the highest bit of the 8421 value.
Return Value	INT type variable, the program runtime will get the port status in binary to decimal conversion and then pass into the int variable.

10.4.8 SetDO8421

Set a continuous DO port status (regard it as a binary data segment), and convert the input decimal number into a binary number to set it on the specified DO port.

1

SetDO8421 ~ 1

1. SetDO8421

Copy

Comments

Delete

Start Port : DO0

End Port : DO2

Set Value : 1

Parameter	Description
Starting Port	The low bit when the desired value is transmitted in binary.
End Port	The high bit when the desired setting value is transmitted in binary.
Return Value	The decimal setting value of the desired port output.

10.4.9 GetDO

This command is used to obtain the status of the DO port and return it as a binary number.

1

GetDO

1. GetDO

Copy

Comments

Delete

Port : DO1

Variable : sensor

Parameter	Description
Port	The DO port from which you want to get the value.
Variable	The binary value that you want the port to output.

10.4.10 GetDI

This command is used to obtain the status of the DI port and return it as a binary number.

1

GetDI

1. GetDI

Copy

Comments

Delete

Port : DI2

Variable : sensor

Parameter	Description
Port	The DI port from which you want to get the value.
Variable	The binary value that you want the port to output.

10.4.11 GetAO

This command is used to obtain the status of the AO port and return it as a decimal fraction.

1

GetAO

1. GetAO

Copy

Comments

Delete

Port : AO1

Variable : anglo

Parameter	Description
Port	The AO port that needs to get the value.
Variable	Decimal decimal value of the desired port output.

10.4.12 GetAI

This command is used to obtain the status of the AI port and return it as a decimal fraction.

1

GetAI

1. GetAI

Copy

Comments

Delete

Port : AI1

Variable : anglo

Parameter	Description
Port	The AI port that needs to get the value.
Variable	The binary value that you want the port to output.

10.5 Set instructions

10.5.1 SetTool

Set the tool parameter command. Switch to this tool parameter.

1

SetTool

1. SetTool

Copy

Comments

Delete

Tool : 0

Parameter	Description
Tool param	Change to the selected tool number.

10.5.2 SetCoord

Set the user coordinate system command. Switch to this user coordinate system.

1

SetCoord

1. SetCoord

Copy

Comments

Delete

Coord : 1

Parameter	Description
Coordinate system	Changed to select the variable number of the coordinate system that has been created for the selection.

10.5.3 SetPayload

Select the workpiece load parameter instruction. Switch to this workpiece load parameter.

1

SetPayload

1. SetPayload

Copy

Comments

Delete

Payload : 3

Parameter	Description
Workpiece Load	Change to Select to select the load variable number that has been created.

10.5.4 Stop

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

10.5.5 EnaVibraSuppr

This command is used to enable vibration suppression.

10.5.6 DisVibraSuppr

This command is used to disable the servo vibration suppression.

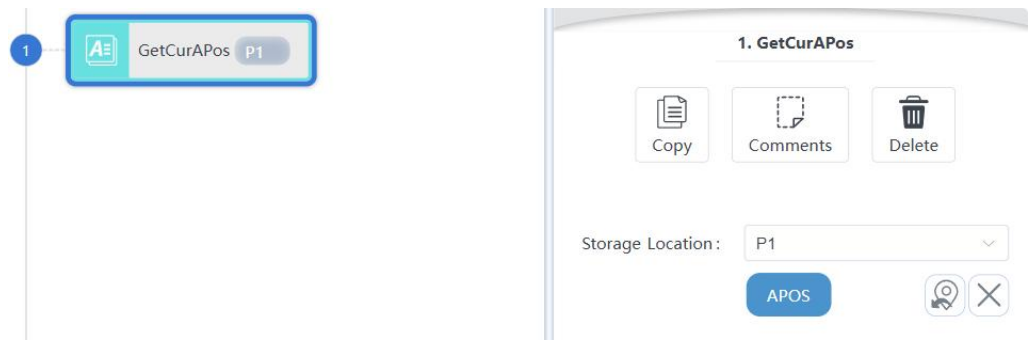
10.5.7 ClsDectLevel

This command is used to set the sensitivity of collision detection.

10.6 Position Operation Instructions

10.6.1 GetCurAPos

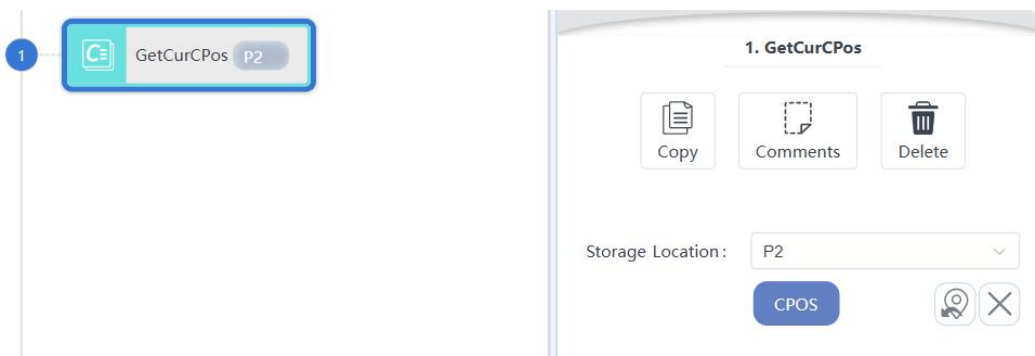
This instruction is used to obtain the current position in the joint coordinate system and assign it to an Apos type variable. You can click on +APOS to add an Apos type variable.



Parameter	Description
Storage location	Current Apos value variable.

10.6.2 GetCurCPos

This command is used to obtain the Cartesian space position in the current reference coordinate system and assign it to a variable of type Cpos. You can click +CPOS to add a variable of type APos.



Parameter	Description
Storage location	Current Cpos value variable.

10.6.3 APosToCPos

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

1

APosToCPosP1P3

1. APosToCPos

Copy

Comments

Delete

Transition

start

P1

point

:

APOS

Tool

:

0

Coord

:

Default

Transition

end

P3

point

:

CPOS

Parameter	Description
Pre-conversion point	Apos variable before conversion
Points after conversion	Cpos variable after conversion
Tool parameters	Tool number involved in the conversion
Coordinate system	Coordinate system number involved in the conversion

10.6.4 CPosToAPos

The robot position point conversion instruction, given the CPos point and the reference coordinate system and tool parameters it belongs to, can obtain the value of the target APos point.

1

CPosToAPosP2P1

1. CPosToAPos

Copy

Comments

Delete

Transition

start

P2

point

:

CPOS

Tool

:

Default

Coord

:

Default

Transition

end

P1

point

:

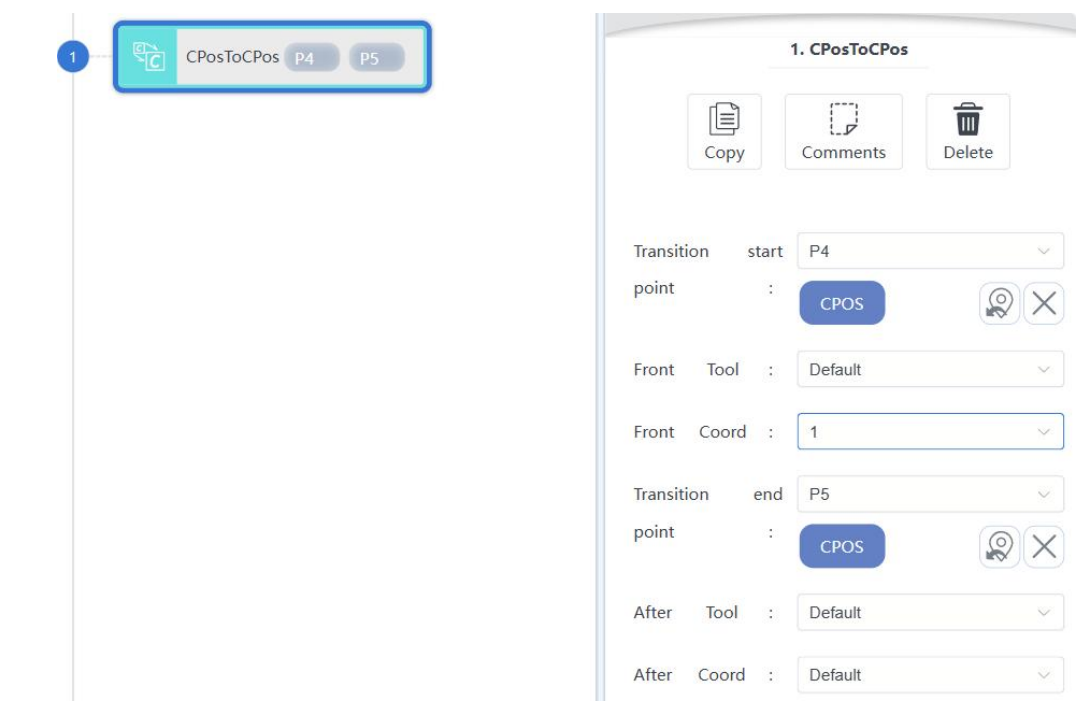
APOS

Parameter	Description
Conversion front point	Cpos variables before conversion
Tool parameters	Tool number involved in the conversion
Coordinate system	Coordinate system number involved in the conversion

Converted point	Apos variable after conversion
-----------------	--------------------------------

10.6.5 CPosToCPos

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



Parameter	Description
Convert Pre-Points	Cpos variables before conversion
Target tool parameters	Tool number involved in Cpos before conversion
Target user coordinate system	Coordinate system number of Cpos before conversion
Post-transformation point	Apos variable after conversion
Baseline tool parameters	Tool number of Cpos after conversion
User coordinate system	Coordinate system number of Cpos after conversion.

10.6.6 ToolOffset

The robot tool coordinate system offset instruction allows for the generation of a new tool coordinate system by rotating or offsetting the reference tool coordinate system. Given the reference tool coordinate system TOOL and the offset or rotation amount required, the value of the target tool coordinate system can be obtained.

1

ToolOffset

P6

1. ToolOffset

Copy

Comments

Delete

Tool

:

1

Offset

Quantity

:

P6

DCPOS

Tool

Param

:

1

Parameter	Description
Tool Parameters	Tool TCP parameter before offset
Offset	Offset DCpos parameters
Tool Parameters	Tool TCP parameters after offset

10.6.7 UserOffset

The robot user coordinate system offset instruction allows for the generation of a new user coordinate system by rotating or offsetting the reference user coordinate system. Given the reference user coordinate system USERCOORD and the offset or rotation amount required, the value of the target user coordinate system can be obtained.

1

UserOffset

P6

1. UserOffset

Copy

Comments

Delete

Coord

:

2

Offset

Quantity

:

P6

DCPOS

Target Usercoord:

1

Parameter	Description
Coordinate system	Coordinate system parameters before offset
Offset	Offset DCpos parameter
Coordinate system	Coordinate system parameter after offset

10.6.8 CposOffset

The robot Cpos offset command allows for the generation of a target Cpos by rotating or offsetting the original Cpos.

1

CposOffset

1. CposOffset

Copy

Comments

Delete

Original Point :

P2

CPOS

Offset Quantity :

P6

DCPOS

TarPos :

P4

CPOS

Parameter	Description
Coordinate system	Cpos parameter before offset
Offset	Offset DCpos parameter
Coordinate system	Cpos parameter after offset

10.6.9 GetAxis

This command is used to obtain the angle of the specified axis.

1

GetAxis

1. GetAxis

Copy

Comments

Delete

Original Point :

P1

APOS

Axis :

Axis1

Angle Value :

anglo

+

Parameter	Description
Origin	Selected Apos
Axis	Selected axis number
Angular value	Angular value

10.6.10 GetCartesian

This command is used to obtain the specified Cartesian pose values.

1

GetCartesian

1. GetCartesian

Copy

Comments

Delete

Point : P2

CPOS

Coordinate Or Direction : x

Store Value : anglo

+

Parameter	Description
Origin	Selected Cpos
Axis	Selected direction or angle
Stored value	Value

10.6.11 Position Inverse

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

1

PositionInverse

1. PositionInverse

Copy

Comments

Delete

Original Point : P2

CPOS

Storage Point : P3

CPOS

Parameter	Description
Origin	Position inversion to original position
Stored point	Position inversion to resultant position

10.6.12 PointsDistance

This instruction is used to calculate the distance between two pose points.

1

PointsDistance

1. PointsDistance

Copy

Comments

Delete

Start Point : P2

CPOS

End Point : P3

CPOS

Distance(mm) : result

Parameter	Description
Origin	Starting Position
Stored point	End Position
Distance	Distance between two attitude points

10.6.13 InterpolationCpos

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

1

InterpolationCpos

1. InterpolationCpos

Copy

Comments

Delete

Start Point : P2

CPOS

End Point : P2

CPOS

Coefficient : 0.5

Storage Point : P5

Parameter	Description
Beginning point	Starting Position
End point	End position
Coefficient	Interpolation factor, interval 0~1, 0 is the starting point, 1 is the end point
Stored point	Position interpolation result position

10.6.14 TransformPlane

This instruction performs a transformation in the XY(YZZX) plane. At the base point, it first

rotates around the Z(XY) axis, then translates along the X(YIZ) axis, and finally along the Y(ZIX) axis. Position variables, array variables, or taught points can be dragged into the base point, and the storage point is the pose after the transformation.

1

TransformPlane

1. TransformPlane

Copy

Comments

Delete

Original Point : P2

CPOS

Plane : yz

Rotation Angle : 0

Y direction : 0 mm

Z direction : 0 mm

Storage Point : P2

CPOS

Parameter	Description
Primitive point	primitive point
Plane	The plane in the point's coordinate system
Angle of rotation	Rotation angle along the selected plane
Translation in the ____ direction	Translation distance in both directions of the selected plane
Stored Points	Resulting point variable

10.6.15 GetTrajStartPoint

This instruction is used to obtain the starting point of the drag trajectory.

1

GetTrajStartPoint

1. GetTrajStartPoint

Copy

Comments

Delete

Trajectory : trj

Start Point : P1


APOS

Parameter	Description
Trajectory	Selected trajectory
starting point	Point storage

10.6.16 GetTrajEndPoint

This instruction is used to obtain the end point of the drag trajectory.

1

GetTrajEndPoint

1. GetTrajEndPoint

Copy

Comments

Delete

Trajectory : trj

End Point : P7

APOS


Parameter	Description
Trajectory	Selected trajectory
End point	Point storage

10.7 Bitwise Operation Instructions

10.7.1 BitAnd

This instruction performs a bitwise AND operation on two operands and assigns the result to the first operand.

1

BitAnd op1 , op2

1. BitAnd

Copy

Comments

Delete

Operand1 : op1

Operand2 : op2

Parameter	Description
Operand 1	INT variable; the result of the operation is also assigned to this operand
Operand 2	INT variable

10.7.2 BitNeg

Implement the bitwise NOT operation. This instruction performs a bitwise NOT operation on the operand and assigns the result back to the operand.

1

BitNeg op1

1. BitNeg

Copy

Comments

Delete

Operand : op1

Parameter	Description
Operand 1	INT variable; the result of the operation is also assigned to this operand

10.7.3 BitOr

Implements the operation of bitwise or. This instruction performs a bitwise or operation on two operands and assigns the result to the first operand.

1

BitOr op1 , op2

1. BitOr

Copy

Comments

Delete

Operand1 : op1

Operand2 : op2

Parameter	Description
Operand 1	INT variable; the result of the operation is also assigned to this operand
Operand 2	INT variable

10.7.4 BitLSH

This instruction performs a bitwise left shift operation. The first operand is shifted left by the number of bits specified by the second operand, and the result is assigned back to the first operand.

1

BitLSH op1 , op2

1. BitLSH

Copy

Comments

Delete

Operand1 : op1

Operand2 : op2

Parameter	Description
Operand 1	INT variable; the result of the operation is also assigned to this operand
Operand 2	INT variable

10.7.5 BitRSH

Implement the bitwise right shift operation. This instruction performs a bitwise right shift on the first operand by the number of bits specified by the second operand and assigns the result back to the first operand.

1

BitRSH op1 , op2

1. BitRSH

Copy

Comments

Delete

Operand1 : op1

Operand2 : op2

Parameter	Description
Operand 1	INT variable; the result of the operation is also assigned to this operand
Operand 2	INT variable

10.8 Clock Instruction

When using clock instructions, a variable of type CLOCK needs to be created.

10.8.1 CLKStart

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

1

CLKStart cTime

2

CLKStop cTime

3

CLKReset cTime

1. CLKStart

Copy

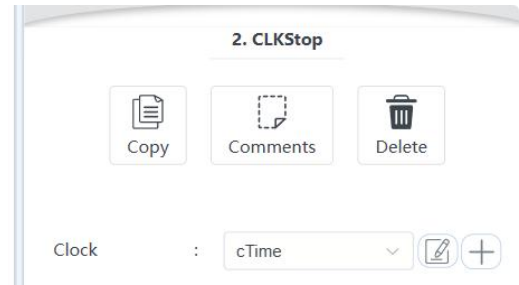
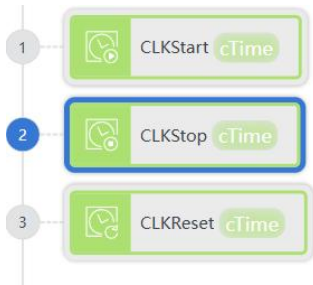
Comments

Delete

Clock : cTime

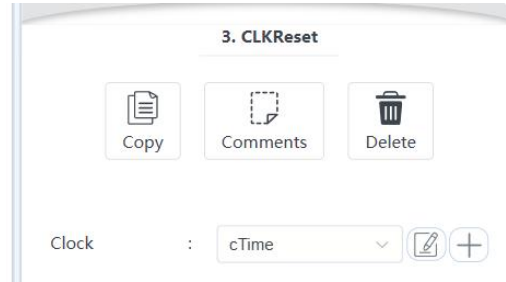
10.8.2 CLKStop

Stop the specified clock (its state is false, but it will not be reset).



10.8.3 CLKReset

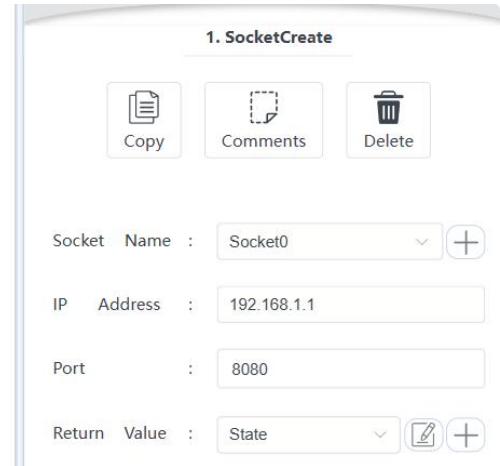
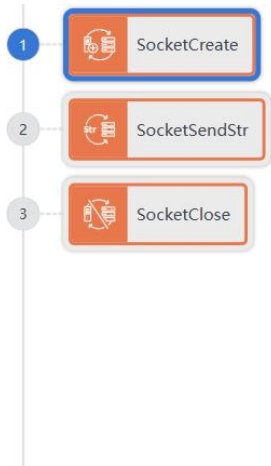
Reset the state value of the specified clock.



10.9 Socket Command

10.9.1 SocketCreate

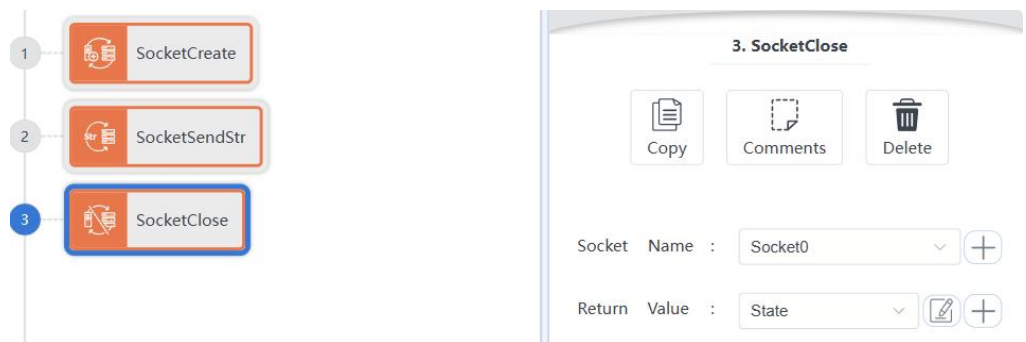
Create a socket client to facilitate data interaction with the server. Based on the parameters passed from the server side, create a client locally and establish a connection with the server.



Parameter	Description
Socket Name	The name of the socket to be created, this value is passed as a socket variable.
IP address	The ip address of the server to connect to
Port number	The port number of the server to connect to.
Return Value	The return value variable for whether the operation was successful, a value of 0 indicates success, a value of 1 indicates failure. If the socket has already been created, it returns 1. This operation does not indicate whether communication is established.

10.9.2 SocketClose

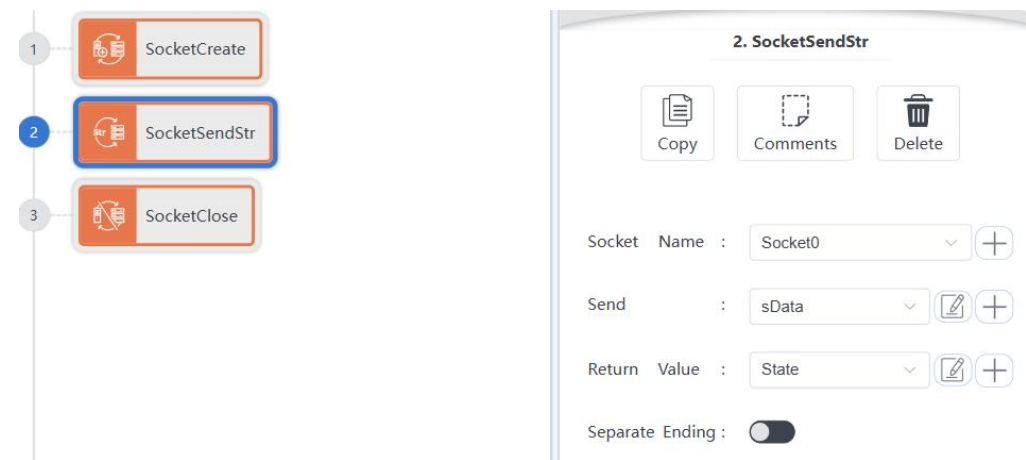
Close the previously created socket client. Based on the passed-in socket name, close the created client and return the success or failure of the operation.



Parameter	Description
Socket Name	The name of the socket to be closed, this value is passed in as a socket variable.
Return Value	A value of 0 indicates success, while a value of 1 indicates failure.

10.9.3 SocketSendStr

Send a string to the server side for command interaction. Send a string to the already established server connection and return whether it is successful or not.



Parameter	Description
Socket Name	The name of the socket on which the send operation is to be performed, passed as a socket variable.
Send	The string of data to be sent to the server.
Return Value	The return value of whether the operation is successful or not, a return value of 0 means success, a return value of 1 means failure.
End of newline	Whether to add “\n” line breaks.

10.9.4 SocketSendReal

Send the real array to the server side for command interaction. Send the real array to the already established server connection and return whether it is successful or not. The starting and ending characters of the sent string can be customized, and the data is

separated by a delimiter.

1

SocketCreate

2

SocketSendReal

3

SocketClose

2. SocketSendReal

Copy

Comments

Delete

Socket

Name

:

Socket0

+

Send

:

sreal

+

Return

Value

:

State

+

Start

String

:

[

End

String

:

]

Separate

:

,

Separate Ending

:

Parameter	Description
Socket Name	The name of the socket on which the send operation should be performed, this value is passed as a socket variable.
Send	The real array of data to be sent to the server.
Return Value	The return value of whether the operation is successful or not, a value of 0 means success, a value of 1 means failure.
Start String	Start String
End String	End string
Separator	Separator between data
End of line break	Whether to add “\n” line breaks.

10.9.5 SocketSendInt

Send an int array to the server side for command interaction. Send an int array to the already established server connection and return whether it is successful or not.

1

SocketCreate

2

SocketSendInt

3

SocketClose

2. SocketSendInt

Copy

Comments

Delete

Socket

Name

:

Socket0

+

Send

:

arr1

+

Return

Value

:

State

+

Start

String

:

[

End

String

:

]

Separate

:

,

Separate Ending

:

Parameter	Description
-----------	-------------

Socket Name	The name of the socket on which the send operation should be performed, this value is passed as a socket variable.
Send	The int array of data to be sent to the server.
Return Value	The return value of whether the operation is successful or not, a value of 0 means success, a value of 1 means failure.
Start String	Start String
End String	End string
Separator	Separator between data
End of line break	Whether to add “\n” line breaks.

10.9.6 SocketReadReal

Read the string sent from the server and store it in the form of a real array. Wait and receive the string sent from the server, which is in the format of start and end strings, with data separated by delimiters. After receiving the string, the robot system will split and parse it and store it in the array in sequence.

1

SocketCreate

2

SocketReadReal

3

SocketClose

2. SocketReadReal

Copy

Comments

Delete

Socket Name :

Socket0

+

Data Number :

0

Data Valueie :

sreal

+

Test Time :

2000

ms

Return Value :

State

+

Start String :

[

End String :

]

Separate :

,

Parameter	Description
Socket name	The name of the socket on which the read operation is to be performed, passed as a socket variable.
Number of data	The number of sockets to read into the array.
Data return value	Stores the read and converted values into the array variable and returns the array variable.
Detection time	Waiting time for the server to send the data. Timeout alarm.
Return Value	The return value of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.
Start String	Start String
End String	End string
Separator	Separator between data.

10.9.7 SocketReadInt

Read the string sent from the server and store it in the form of an int array. Wait and receive the string sent from the server, which is in the format of start and end strings, with data separated by delimiters. After receiving the string, the robot system will split and parse it and store it in the array in sequence.

1

SocketCreate

2

SocketReadInt

3

SocketClose

2. SocketReadInt

Copy

Comments

Delete

Socket

Name

:

Socket0

+

Data

Number

:

0

Data

Value

:

arr1

+

Test

Time

:

2000

ms

Return

Value

:

State

+

Start

String

:

[

End

String

:

]

Separate

:

|

Parameter	Description
Socket name	The name of the socket on which the read operation is to be performed, passed as a socket variable.
Number of data	The number of sockets to read into the array.
Data return value	Stores the read and converted values into the array variable and returns the array variable.
Detection time	Waiting time for the server to send the data. Timeout alarm.
Return Value	The return value of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.
Start String	Start String
End String	End string
Separator	Separator between data.

10.9.8 SocketReadStr

Read the string sent from the server and store it in the form of a string. Wait and receive the string sent from the server, which is in the format of start and end strings and string data. After receiving the string, the robot system will split and parse it and store it in character variables.

1

SocketCreate

2

SocketReadStr

3

SocketClose

2. SocketReadStr

Copy

Comments

Delete

Socket

Name

:

Socket0

+

Data

Value

:

sData

+

Test

Time

:

2000

ms

Return

Value

:

State

+

Start

String

:

[

End

String

:

]

Parameter	Description
Socket Name	The name of the socket on which the read operation should be performed, passed as a socket variable.
Data return value	The string data sent from the server, the value is returned as a string variable.
Detection time	Waiting time for the server to send the data. Timeout alarm.
Return Value	The return value of whether the operation is successful or not, a value of 0 means success, a value of 1 means failure.
Start String	Start String
End String	End String

10.10 Interrupt Instruction

10.10.1 IConnect

This instruction is used to create an interrupt identifier and connect an interrupt task.

1

IConnect macro3

2

ITimer

3

ICondition

4

IDelete

1. IConnect

Copy

Comments

Delete

Interrupt

:

ipt

+

Bind

▼

Task

:

macro3

▼

Parameter	Description
Interrupt ID	ID name
Tasks	Task called

10.10.2 IDelete

This command is used to disconnect the interrupt name from the interrupt task.

1

IConnect macro3

2

ITimer

3

ICondition

4

IDelete

4. IDelete

Copy

Comments

Delete

Interrupt : ipt

Attr Settings

Desc : input string

Parameter	Description
Interrupt ID	ID name
Tasks	Task called

10.10.3 ITimer

This instruction is used to trigger an interrupt at a specified time.

1

IConnect macro3

2

ITimer

3

ICondition

4

IDelete

2. ITimer

Copy

Comments

Delete

Interrupt : ipt

Trigger Time : 300 ms

Parameter	Description
Interrupt ID	ID name
Tasks	Interval trigger time

10.10.4 ICondition

This instruction is used to execute an interrupt when the conditions are met.

1

IConnect macro3

2

ITimer

3

ICondition

4

IDelete

3. ICondition

Copy

Comments

Delete

Interrupt : ipt

Conditional expression

Click on the node and edit the following form.

State.value

Edit expression

Type : Value

Data Type : Variable

Select Var : State / value

Parameter	Description
Interrupt ID	ID name
conditional expression	When the expression satisfies the condition, an interrupt will be executed

10.11 Modbus Commands

10.11.1 GetModConState

This command is used to obtain the connection status of the robot's communication with the outside world via ModbusTCP.

1

GetModConState

1. GetModConState

Copy

Comments

Delete

Device Name : MODBUS_mb

Is Connected : status

Parameter	Description
Device name	Name of the Modbus device to be operated
Connection Status	Returns the current connection status, type BOOL

10.11.2 ReadSingleCoilReg

This command is used to read a single coil register of the specified Modbus.

1

ReadSingleCoilReg

1. ReadSingleCoilReg

Copy

Comments

Delete

Device Name : MODBUS_mb

Register ID : 0

Register Value : regb

Device Address : 1

Timeout time : 3000 ms

Return Value : State

Parameter	Description
Device name	Modbus device name to be operated
Address	Address of the register to be read
Destination register value	Variable holding the value of the register to be read, type BOOL
Slave Device Address	Address of slave device
Timeout	Waiting time for reading, timeout alarm
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.11.3 ReadDiscretelInputReg

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

1

ReadDiscretelInputReg

1. ReadDiscretelInputReg

Copy

Comments

Delete

Device Name : MODBUS_mb

Register ID : 0

Register Value : regb

Device Address : 1

Timeout time : 3000 ms

Return Value : State

Parameter	Description
Device name	Modbus device name to be operated
Address	Address of the register to be read
Destination register value	Variable holding the value of the register to be read, type Int
Slave Device	Address of slave device

Address	
Timeout	Waiting time for reading, timeout alarm
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.11.4 ReadSingleHoldReg

This command is used to read a single holding register of the specified Modbus.

1

ReadSingleHoldReg

Copy

Comments

Delete

Device Name :

MODBUS_mb

Register ID :

0

Register Value :

int1

+

Device Address :

1

Timeout time :

3000

ms

Return Value :

State

+

Parameter	Description
Device name	Modbus device name to be operated
Address	Address of the register to be read
Destination register value	Variable holding the value of the register to be read, type Int
Slave Device Address	Address of slave device
Timeout	Waiting time for reading, timeout alarm
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.11.5 ReadInputReg

This command is used to read the specified Modbus input registers.

1

ReadInputReg

1. ReadInputReg

Copy

Comments

Delete

Device Name : MODBUS_mb

Register ID : 0

Register Value : int1

Device Address : 1

Timeout time : 3000 ms

Return Value : State

Parameter	Description
Device name	Modbus device name to be operated
Address	Address of the register to be read
Destination register value	Variable holding the value of the register to be read, type Int
Slave Device Address	Address of slave device
Timeout	Waiting time for reading, timeout alarm
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.11.6 WriteSingleCoilReg

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

1

WriteSingleCoilReg

1. WriteSingleCoilReg

Copy

Comments

Delete

Device Name : MODBUS_mb

Register ID : 0

Register Value : regb

Device Address : 1

Timeout time : 3000 ms

Return Value : State

Parameter	Description
Device name	Modbus device name to be operated
Address	Address of the register to be written
Destination register value	Variable holding the value of the register to be written, type BOOL

Slave Device Address	Address of slave device
Timeout	Waiting time for reading, timeout alarm
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.11.7 WriteSingleHoldReg

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

1

WriteSingleHoldReg

Copy

Comments

Delete

Device Name :

MODBUS_mb

Register ID :

0

Register Value :

int2

+

Device Address :

1

Timeout time :

3000

ms

Return Value :

State

+

Parameter	Description
Device name	Modbus device name to be operated
Address	Address of the register to be written
Destination register value	Variable holding the value of the register to be written, type Int
Slave Device Address	Address of slave device
Timeout	Waiting time for reading, timeout alarm
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.12 Array Instructions

10.12.1 SetMatrix2

To form a linear array in space by specifying two points, and then evenly divide this linear array according to the set number of rows to obtain a matrix point group.

1

SetMatrix2

2

GetMatrix2

1. SetMatrix2

Copy

Comments

Delete

Matrix Name : MM2

p1 : P1

CPOS

p2 : P2

CPOS

Number : 3

P1-1

P1-2

Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
p1	Specifies the first point of the linear array, type CPOS.
p2	Specifies the last point of the linear array, type CPOS.
Number of Pieces	Number of rows of the generated array, type INT

10.12.2 SetMatrix3

To form a parallelogram array in space by specifying three points, and then divide this parallelogram into equal parts according to the set number of rows and columns to obtain a matrix of points.

1

SetMatrix3

2

GetMatrix3

1. SetMatrix3

Copy

Comments

Delete

Matrix Name : MM3

p1-1 : P1

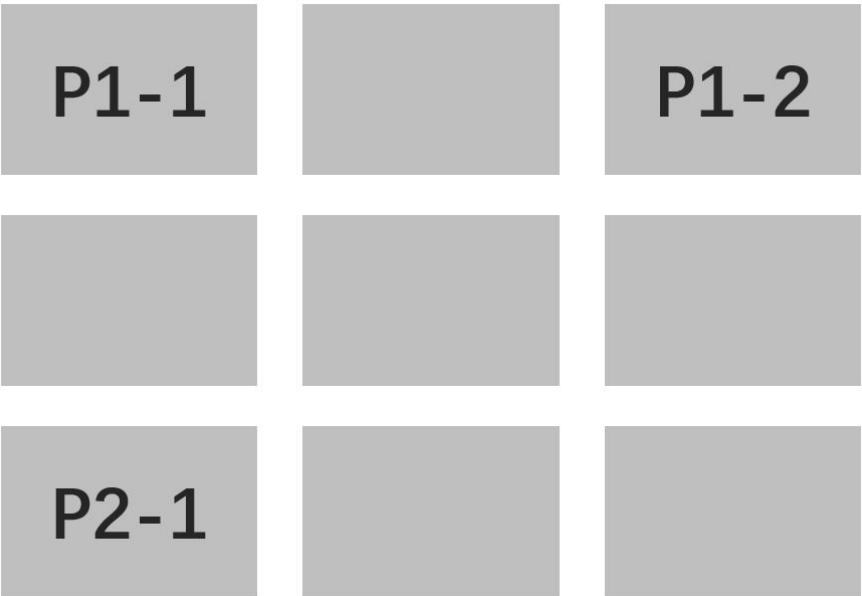
CPOS

p1-2 : P2

CPOS

p2-1 : P3

CPOS



Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
p1-1	Specifies the first point in the first row of the parallelogram, also called the origin, of type CPOS.
p1-2	Specify the last point of the first row of the parallelogram, type CPOS
p2-1	Specifies the first point of the last row of the parallelogram, of type CPOS
Number of rows	Generates the number of rows of the array, type INT
Columns	Generates the number of columns of the array, type INT

10.12.3 SetMatrix4

To form a parallelogram array in space by specifying four points, and then equally divide this parallelogram into a matrix point group according to the set number of rows and columns. Compared with the Matrix3 command, this function can obtain more accurate point positions. When calculating the target point position, the array is divided into four regions, and then the three points closest to the target point are automatically selected in each region for Matrix3 operation.

1

SetMatrix4

2

GetMatrix4

1. SetMatrix4

Copy

Comments

Delete

Matrix Name

:

MM4

+

p1-1

:

P1

CPOS

p1-2

:

P2

CPOS

p2-1

:

P3

CPOS

p2-2

:

P4

CPOS

Row

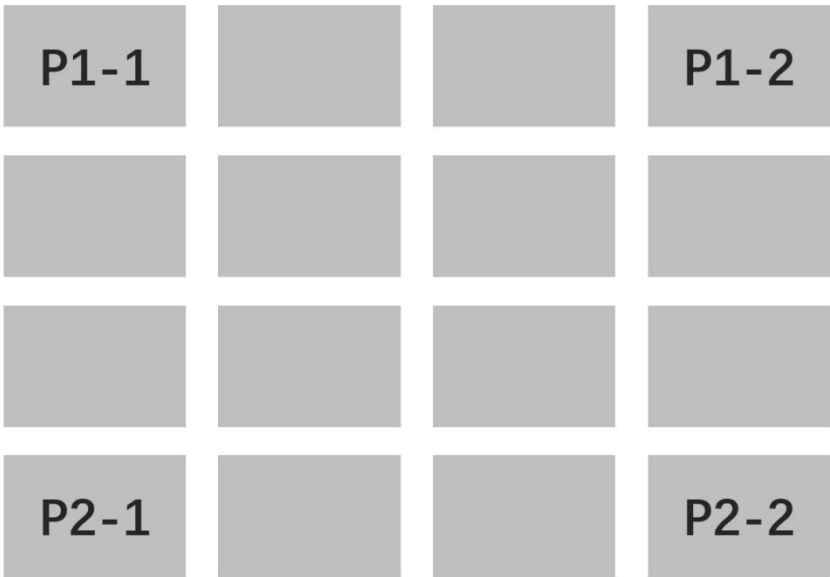
:

4

Column

:

3



Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
p1-1	Specifies the first point in the first row of the parallelogram, also called the origin, of type CPOS.
p1-2	Specify the last point of the first row of the parallelogram, type CPOS
p2-1	Specifies the first point of the last row of the parallelogram, of type CPOS
p2-2	Specify the point near the first point in the middle of the parallelogram, type CPOS
Number of rows	Specifies the last point of the last row of a parallelogram, of type CPOS
Columns	Generates the number of rows of the array, type INT

10.12.4 SetMatrix9

To form a parallelogram array in space by specifying four points, and then evenly divide this parallelogram into a matrix point group according to the set number of rows and columns. This function can obtain more accurate point positions compared to the Matrix3 command. When calculating the target point position, the array is divided into 9 regions, and then the three points closest to the target point are automatically selected in each region for Matrix3 operation. When the number of rows or columns is even, the middle point should be selected as the point closest to the first point of the row or column at the middle position.

1

SetMatrix9

2

GetMatrix9

1. SetMatrix9

Copy

Comments

Delete

Matrix Name

:

MM9

p1-1

:

P1

CPOS

p1-2

:

P2

CPOS

p1-3

:

P3

CPOS

p2-1

:

P4

CPOS

p2-2

:

P5

CPOS

P1-1

P1-2

P1-3

P2-1

P2-2

P2-3

P3-1

P3-2

P3-3

Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
p1-1	Specifies the first point in the first row of the parallelogram, also called the origin, of type CPOS.
p1-2	Specifies the point near the first point at the middle of the first row of the parallelogram, type CPOS
p1-3	Specifies the last point of the first row of the parallelogram, of type

Version V1.0

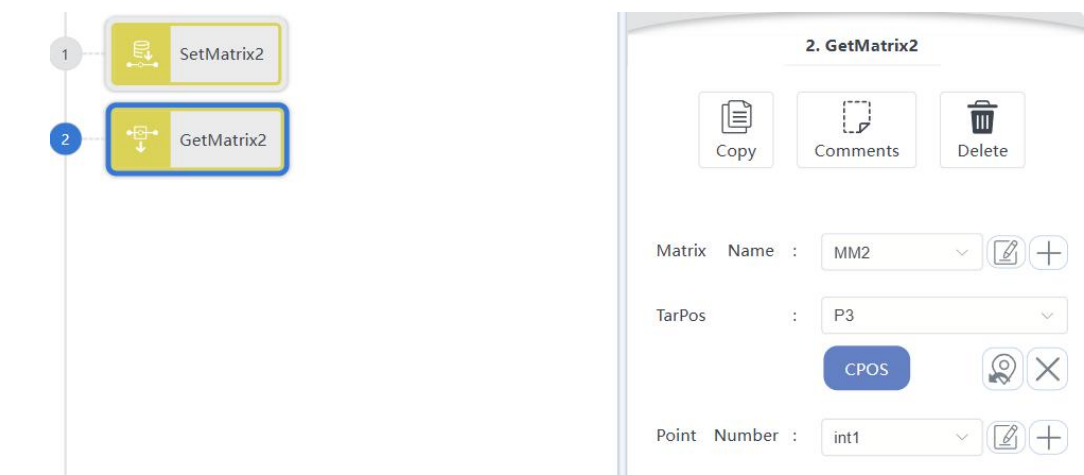
Copyright @ Estun Codroid

172

	CPOS
p2-1	Specifies the first point in the last row of a parallelogram, of type CPOS
p2-2	Specify the point near the first point in the middle of the parallelogram, type CPOS
p2-3	Specify the last point of the last row of the parallelogram, type CPOS
p3-1	Specify the first point of the last row of the parallelogram, type CPOS
p3-2	Specify the point near the first point in the middle of the last row of the parallelogram, type CPOS
p3-3	Specify the last point in the last row of the parallelogram, type CPOS
Number of rows	Number of rows of the generated array, type INT
Number of columns	Generates the number of columns of the array, type INT

10.12.5 GetMatrix2

Take the value of the corresponding point in the row and column after the execution of the SetMatrix command and assign it to the target point. The orientation and additional axis angle value of the target point remain consistent with those of the p1 point in the SetMatrix command.



Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
Target Point	The value of the point to be fetched, type CPOS.
Point number	The serial number of the point to be fetched in the Matrix, type INT, the serial number counts from 0.

10.12.6 GetMatrix3

Take the value of the corresponding point in the row and column after the execution of the SetMatrix command and assign it to the target point. The orientation and additional axis angle value of the target point remain consistent with those of the p1-1 point in the SetMatrix command.

1

SetMatrix3

2

GetMatrix3

2. GetMatrix3

Copy

Comments

Delete

Matrix

Name

:

MM3

+

Row

:

hang

+

Column

:

lie

+

TarPos

:

P4

CPOS

Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
Rows	Row number of the point to be fetched in the matrix, type INT, number counting from 0
Columns	Row number of the point to be fetched in the matrix, type INT, counting from 0
Target Points	The value of the point to be fetched, type CPOS.

10.12.7 GetMatrix4

Take the value of the corresponding row and column of the point after the execution of the SetMatrix command and assign it to the target point. The orientation and additional axis angle value of the target point remain consistent with those of the p1-1 point of the SetMatrix command. Compared with the Matrix3 command, this function can achieve more accurate point positions. When calculating the target point position, the array is divided into four regions, and then three points closest to the target point are automatically selected in each region for Matrix3 operation.

1

SetMatrix4

2

GetMatrix4

2. GetMatrix4

Copy

Comments

Delete

Matrix

Name

:

MM4

+

Row

:

lie

+

Column

:

hang

+

TarPos

:

P5

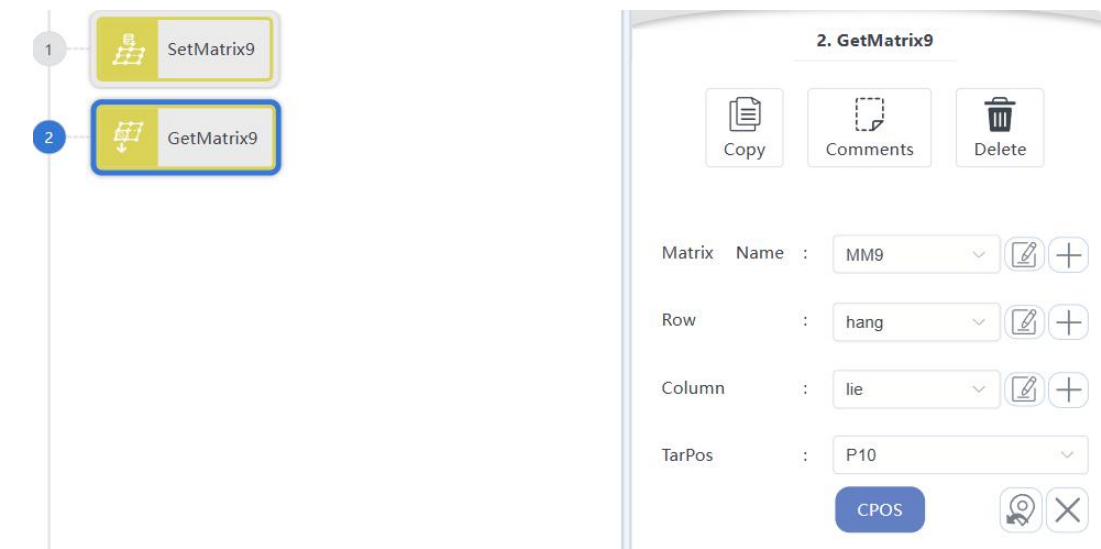
CPOS

Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
Rows	Row number of the point to be fetched in the matrix, type INT, number counting from 0
Columns	Row number of the point to be fetched in the matrix, type INT, counting from 0

Target Points	The value of the point to be fetched, type CPOS.
---------------	--

10.12.8 GetMatrix9

Take the value of the corresponding row and column of the point after the execution of the SetMatrix command and assign it to the target point. The orientation and additional axis angle value of the target point remain consistent with those of point p1-1 in the SetMatrix command. Compared with the Matrix3 command, this function can achieve more accurate point positions. When calculating the target point position, the array is divided into 9 regions, and then three points closest to the target point are automatically selected in each region for Matrix3 operation.



Parameter	Description
Matrix Name	Name of the Matrix to be manipulated
Rows	Row number of the point to be fetched in the matrix, type INT, number counting from 0
Columns	Row number of the point to be fetched in the matrix, type INT, counting from 0
Target Points	The value of the point to be fetched, type CPOS.

10.13 String instructions

10.13.1 APosToStr

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

1

AP

Str

APosToStr

2

CP

Str

CPosToStr

3

DAP

Str

DAPosToStr

4

DCP

Str

DCPosToStr

1. APosToStr

Copy

Comments

Delete

Unconverted

P1

APOS :

APOS

Memory String :

sData

+

Start String :

[

End String :

]

Separate :

,

Angle Unit :

Angle

Parameter	Description
APOS to be converted	APOS values to be converted
Stored String	Converted string variable
Start String	Add start string
End String	Adding the ending string
Split character	Spacing between values
Angle unit	Angular units in APOS

10.13.2 CPosToStr

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

1

AP

Str

APosToStr

2

CP

Str

CPosToStr

3

DAP

Str

DAPosToStr

4

DCP

Str

DCPosToStr

2. CPosToStr

Copy

Comments

Delete

Unconverted

P2

CPOS :

CPOS

Memory String :

sData

+

Start String :

[

End String :

]

Separate :

,

Angle Unit :

Angle

Length Unit :

Millimetre

Parameter	Description
CPOS to be converted	CPOS value to be converted
Stored String	Converted string variable
Start String	Add start string

End String	Adding the ending string
Split character	Interval symbols between values
Angle unit	Angle numeric unit in CPOS
Length unit	Numerical units of length in CPOS

10.13.3 DAPosToStr

This instruction is used to convert the Dapos variable into a string variable.

1

AP

Str

APosToStr

2

CP

Str

CPosToStr

3

DA

Str

DAPosToStr

4

DCP

Str

DCPosToStr

3. DAPosToStr

Copy

Comments

Delete

Unconverted

P3

DAPOS

:

DAPOS

🔍

✕

Memory String

:

sData

📝

+

Start String

:

[

End String

:

]

Separate

:

,

Angle Unit

:

Angle

▼

Parameter	Description
DAPOS to be converted	DAPOS value to be converted
Stored Strings	Converted string variable
Start String	Add start string
End String	Adding the ending string
Split character	Spacing between values
Angle unit	DAPOS Angle Unit Format

10.13.4 DCPosToStr

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

1

APosToStr

2

CPosToStr

3

DAPosToStr

4

DCPosToStr

4. DCPosToStr

Copy

Comments

Delete

Unconverted

P4

DCPOS
:

DCPOS

Memory String
:

sData

Start String
:

[

End String
:

]

Separate
:

,

Angle Unit
:

Angle

Length Unit
:

Millimetre

Parameter	Description
DCPOS to be converted	DCPOS values to be converted
Stored Strings	Converted string variable
Start String	Add start string
End String	Adding the ending string
Split character	Spacing symbols between values
Angle unit	DCPOS Angle Numeric Units
Length unit	Numerical units of length in DCPOS

10.13.5 TranStrToIntArray

This instruction is used to convert a string variable into an int array variable.

1

TranStrToIntArray

2

TranStrToRealArray

1. TranStrToIntArray

Copy

Comments

Delete

Current String
:

sData

Separate
:

,

Memory Var
:

arr1

Return Value
:

State

Start String
:

[

End String
:

]

Parameter	Description
Current String	String to be converted
Splitter	Spacing symbols between values
Stored Variables	The int array variable after conversion.

Return Value	Return value variable for success, 0 means success, 1 means failure.
Starting String	Add start string
End String	Add the end string.

10.13.6 TranStrToRealArray

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

1

TranStrToIntArray

2

TranStrToRealArray

2. TranStrToRealArray

Copy

Comments

Delete

Current String :

sData

+

Separate :

,

Memory Var :

sreal

+

Return Value :

State

+

Start String :

[

End String :

]

Parameter	Description
Current String	String to be converted
Splitter	Spacing symbols between values
Stored Variables	The real array variable after conversion.
Return Value	Return value variable for success, 0 means success, 1 means failure.
Starting String	Add start string
End String	Add the end string.

10.13.7 TranStrToApos

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

1

TranStrToApos

2

TranStrToCpos

3

TranStrToDApos

4

TranStrToDCpos

1. TranStrToApos

Copy

Comments

Delete

Current String : sData

Separate : ,

Memory Var : P1

APOS

Return Value : State

Start String : [

End String :]

Angle Unit : Angle

Parameter	Description
Current String	String to be converted
Splitter	Spacing symbols between values
Stored Variables	Apos variable after conversion
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, a return value of 1 means failure.
Starting String	Add start string
End String	Add end string
Angle unit	Angle numerical units in APOS

10.13.8 TranStrToCpos

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

1

TranStrToApos

2

TranStrToCpos

3

TranStrToDApos

4

TranStrToDCpos

2. TranStrToCpos

Copy

Comments

Delete

Current String

:

sData

Separate

:

,

Memory Var

:

P2

CPOS

Return Value

:

State

Start String

:

[

End String

:

]

Angle Unit

:

Angle

Length Unit

:

Millimetre

Parameter	Description
Current String	String to be converted
Splitter	Spacing symbols between values
Stored Variables	Cpos variable after conversion
Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, a return value of 1 means failure.
Starting String	Add start string
End String	Add end string
Angle unit	Angle unit of Cpos
Length unit	Cpos unit of length

10.13.9 TranStrToDApos

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

1

TranStrToApos

2

TranStrToCpos

3

TranStrToDApos

4

TranStrToDCpos

3. TranStrToDApos

Copy

Comments

Delete

Current String

:

sData

+

Separate

:

,

Memory Var

:

P3

DAPOS

Return Value

:

State

+

Start String

:

[

End String

:

]

Angle Unit

:

Angle

Parameter	Description
Current String	String to be converted
Splitter	Spacing symbols between values
Stored Variables	DApos variable after conversion
Return Value	The return value of the operation, a value of 0 indicates success, a value of 1 indicates failure.
Starting String	Add start string
End String	Add end string
Angle unit	Angle unit form in DApos

10.13.10 TranStrToDCpos

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

1

TranStrToApos

2

TranStrToCpos

3

TranStrToDApos

4

TranStrToDCpos

4. TranStrToDCpos

Copy

Comments

Delete

Current String

:

sData

+

Separate

:

,

Memory Var

:

P4

DCPOS

Return Value

:

State

+

Start String

:

[

End String

:

]

Angle Unit

:

Angle

Length Unit

:

Millimetre

Parameter	Description
Current String	String to be converted
Splitter	Spacing symbols between values
Stored Variables	DCpos variable after conversion
Return Value	Return value variable for successful operation, 0 means success, 1 means failure.
Starting String	Add start string
End String	Add end string
Angle unit	Angle unit of DCpos
Length unit	The unit of length in DCpos

10.13.11 IntArrayToString

This instruction is used to convert an Int array into a string.

1

Int

Str

IntArrayToString

2

Real

Str

RealArrayToString

3

Bool

Str

BoolArrayToString

1. IntArrayToString

Copy

Comments

Delete

Int

Array

:

arr1

▼

📝

+

String

:

sData

▼

📝

+

Start

String

:

[

End

String

:

]

Separate

:

,

Parameter	Description
Current String	String to be converted
Int Array	Array to be converted
String	Output string result
Starting String	Add start string
End String	Add end string
Separator	Spacing between values

10.13.12 RealArrayToString

This instruction is used to convert a Real array into a string.

1

Int

Str

IntArrayToString

2

Real

Str

RealArrayToString

3

Bool

Str

BoolArrayToString

2. RealArrayToString

Copy

Comments

Delete

Real

Array

:

sreal

+

String

:

sData

+

Start

String

:

[

End

String

:

]

Separate

:

,

Parameter	Description
Real Array	Array to be converted
String	Output string result
Start String	Add start string
End String	Add end string
Separator	Spacing between values

10.13.13 BoolArrayToString

This instruction is used to convert a Boolean array into a string.

1

Int

Str

IntArrayToString

2

Real

Str

RealArrayToString

3

Bool

Str

BoolArrayToString

3. BoolArrayToString

Copy

Comments

Delete

Bool

Array

:

barr

+

String

:

Please select

+

Start

String

:

[

End

String

:

]

Separate

:

,

Parameter	Description
Bool Array	Array to be converted
String	Output string result
Start String	Add start string
End String	Add end string
Separator	Spacing between values

10.14 RS485 Instructions

10.14.1 RS485Init

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

1

RS485Init

2

RS485Read

3

RS485Write

4

RS485FlushReadBuffer

1. RS485Init

Copy

Comments

Delete



Number : 2.(Robot terminal interface' ▾

Baud Rate : 115200 ▾

Data Bit : 8 ▾

Parity : None ▾

Stop Bit : 1 ▾

Return Value : State ▾  

Parameter	Description
No.	Robot control cabinet RS485 port or robot end RS485 port
Baud rate	RS485 communication baud rate
Data Bit	RS485 communication data bits
Check Bit	RS485 communication parity bit
Stop Bit	RS485 communication stop bit
Operation Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.14.2 RS485Read

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

1

RS485Init

2

RS485Read

3

RS485Write

4

RS485FlushReadBuffer



2. RS485Read

Copy

Comments



Delete

Number : 2.(Robot terminal interface' ▾

Int Array : arr1 ▾  

Timeout time : 3000 ms

Byte Number : 5

Return Value : State ▾  

Parameter	Description
No.	Robot control cabinet RS485 port or robot end RS485 port
Int array	Variable in which the data to be read is stored
Timeout Time	Timeout for reading, if no data is read after this time, an error will be reported.
Bytes	Length of data bytes to be read
Operation	Return value variable of whether the operation is successful or not,

Return Value	a return value of 0 means success, and a return value of 1 means failure.
--------------	---

10.14.3 RS485Write

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

1

RS485Init

2

RS485Read

3

RS485Write

4

RS485FlushReadBuffer

3. RS485Write

Copy

Comments

Delete

Number

:

2.(Robot terminal interface'

▼

Int

Array

:

arr1

▼

+

Byte

Number

:

5

Return

Value

:

State

▼

+

Parameter	Description
No.	Robot control cabinet RS485 port or robot end RS485 port
Int arrays	Variable where data to be sent is stored
Bytes	Length of data bytes to be sent
Operation Return Value	Return value variable of whether the operation is successful or not, a return value of 0 means success, and a return value of 1 means failure.

10.14.4 RS485FlushReadBuffer

This command is used to clear the cache data read from the RS485 port on the control cabinet. It is generally cleared after reading to ensure normal reading in the next operation. Alternatively, it can be left uncleared, and the data can be processed together after multiple receptions.

1

RS485Init

2

RS485Read

3

RS485Write

4

RS485FlushReadBuffer

4. RS485FlushReadBuffer

Copy

Comments

Delete

Number

:

2.(Robot terminal interface'

▼

Attr Settings

▼

Desc

:

input string

Parameter	Description
No.	Robot control cabinet RS485 port or robot end RS485 port

10.15 Mathematical operation functions

In the "IF" instruction and "... =..." In the instructions, mathematical operation functions or string operation functions may be used. This section explains the "mathematical functions" that can be used.

10.15.1 sin

Sine trigonometric function.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.2 cos

Cosine trigonometric function.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.3 tan

Tangent trigonometric function.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.4 asin

Inverse sine trigonometric function.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.5 acos

Inverse cosine trigonometric function.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.6 atan

Inverse tangent trigonometric function.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.7 atan2

The X/Y inverse tangent function returns the radian value from the X-axis to the point (x, y).

Parameter 1: An integer or real variable or constant.

Parameter 2: An integer or real variable or constant.

The function return value: a real constant.

10.15.8 sinh

Hyperbolic sine function.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.9 cosh

Hyperbolic cosine function.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.10 tanh

Hyperbolic tangent function.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.11 log

Natural logarithmic function.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.12 log10

The logarithmic function with base 10.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.13 sqrt

Square root function.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.14 exp

The exponential function with base e.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.15 pow

Exponential function.

Parameter 1: An integer or real variable or constant, representing the base.

Parameter 2: An integer or real variable or constant, representing the exponent.

The function return value: a real constant.

10.15.16 deg

Radian-to-degree conversion function.

Parameter 1: An integer or real type variable or constant.

The function return value: a real constant.

10.15.17 rad

Function for converting degrees to radians.

Parameter 1: An integer or real variable or constant.

The function return value: a real constant.

10.15.18 fmod

Modulo function.

Parameter 1: An integer or real variable or constant, the dividend.

Parameter 2: An integer or real variable or constant, the divisor.

The function return value: a real constant.

10.15.19 floor

Floor function.

Parameter 1: An integer or real type variable or constant.

The function return value is an int type constant.

10.15.20 random

Get a random integer between two parameters.

Parameter 1: An integer variable or constant.

Parameter 2: An integer variable or constant.

The function return value is an int type constant.

10.16 String Functions

10.16.1 byte

Get the ASCII code of the character at the nth position in the string.

Parameter 1: A string type variable or constant.

Parameter 2: An integer variable or constant.

The function return value is an int type constant.

10.16.2 char

Return the character corresponding to the ASCII code.

Parameter 1: An integer variable or constant.

The function returns a string type constant.

10.16.3 find2

Return the position of a substring in a string.

Parameter 1: A string type variable or constant.

Parameter 2: string type variable or constant

The function return value is an int constant. (When the corresponding character or string is not found, the return value is -1.)

10.16.4 findEnd

The string reverse search instruction finds the last occurrence of a specified string within a string and returns the index number.

Parameter 1: The source string to be searched, a string type variable or constant.

Parameter 2: The specified string to be searched for, a string type variable or constant.

The function return value: the index number after the search, an int type variable.

10.16.5 format

The formatted string instruction returns the formatted data by passing a reasonable format control character in parameter 1, followed by any number of parameters to fill this format control character.

Parameter 1: String format, a string type variable or constant.

Parameter 2: The parameter to be filled with the format specifier, a string/real/int type variable or constant.

Parameter 3: The parameter to be filled with the format specifier, a string/real/int type variable or constant.

... There are no restrictions on the parameters. The total length just needs to be within the limit of a single instruction string.

The return value of the function: the number of successfully split and saved elements in the array, an int type variable.

Format strings start with % and support the following usages: %c - accepts a number and converts it to the corresponding character in the ASCII table; %d, %i - accepts a number and converts it to a signed integer format; %o - accepts a number and converts it to an octal format; %u - accepts a number and converts it to an unsigned integer format; %x - accepts a number and converts it to a hexadecimal format using lowercase letters x; %X - accepts a number and converts it to a hexadecimal format using uppercase letters X; %f - accepts a number and converts it to a floating-point format; %s - accepts a string and formats it according to the given parameters. Examples: format("%c: %c", 83) outputs S; format("%+d", 17.0) outputs +17; format("%05d", 17) outputs 00017; format("%o", 17) outputs 21; format("%u", 3.14) outputs 3; format("%x", 13) outputs d; format("%X", 13) outputs D; format("%6.3f", 13) outputs 13.000; format("%s", "monkey") outputs monkey; format("%10s", "monkey") outputs monkey.

10.16.6 getAt

Single string acquisition instruction: Acquire the string data of a specific position and return the acquired data.

Parameter 1: The string to be extracted, a string type variable or constant.

Parameter 2: The position to be obtained, an int type variable or constant.

Function return value: The obtained string, a string-type variable.

10.16.7 gsub

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

Parameter 1: A string type variable or constant.

Parameter 2: A string type variable or constant.

Parameter 3: A string type variable or constant.

The function returns a string type constant.

10.16.8 len

Calculate the length of a string.

Parameter 1: A string type variable or constant.

The function return value is an int type constant.

10.16.9 left

The string left extraction instruction starts from the left side of the string, extracts a specified number of characters, and returns the extracted data.

Parameter 1: The string to be extracted, a string type variable or constant.

Parameter 2: The quantity to be extracted, an int type variable or constant.

The function return value: the truncated string, a string type variable.

10.16.10 lower

Return the lowercase format of the string.

Parameter 1: A string type variable or constant.

The function returns a string type constant.

10.16.11 right

The string right extraction instruction starts from the right side of the string, extracts a specified number of characters, and returns the extracted data.

Parameter 1: The string to be extracted, a string type variable or constant.

Parameter 2: The quantity to be extracted, an int type variable or constant.

The function return value: the truncated string, a string type variable.

10.16.12 reverse

The string reversal instruction reverses the string and returns it.

Parameter 1: The string to be reversed, a string type variable or constant.

The function return value: the reversed string, a string type variable.

10.16.13 strcmp

String comparison instruction, returning the ASCII code difference between the first different characters.

Parameter 1: The string data to be compared, a string type variable or constant.

Parameter 2: The string data to be compared, a string type variable or constant.

The function return value: The returned ASCII code value, an int type variable.

10.16.14 trimLeft

The left trim instruction for strings removes the spaces on the left side of the string and returns the modified string data.

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

The function return value: the trimmed string, a string type variable.

10.16.15 trimRight

The string right trim instruction removes the spaces on the right side of the string and returns the modified string data.

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

The function return value: the trimmed string, a string type variable.

10.16.16 upper

Return the uppercase format of the string.

Parameter 1: A string type variable or constant.

The function returns a string type constant.

10.16.17 IToStr

The integer-to-string conversion instruction converts integer data into string type data and returns the converted string.

Parameter 1: The integer data to be converted, an int type variable or constant.

The function return value: the converted string data, a string type variable.

10.16.18 RToStr

The real number to string conversion instruction converts real number data into string type data and returns the converted string.

Parameter 1: The real number data to be converted, a REAL type variable or constant.

The function return value: the converted string data, a string type variable.

10.16.19 StrToI

The string-to-integer conversion instruction converts string data into integer type data and returns the converted integer data.

Parameter 1: The string data to be converted, a string type variable or constant.

The function return value: the converted integer data, an int type variable.

10.16.20 StrToR

The string-to-real data conversion instruction converts string data into real number type data and returns the converted real number.

Parameter 1: The string data to be converted, a string type variable or constant.

The function return value: the converted real number data, a REAL type variable.

10.16.21 Append

The Append instruction is used to append strings.

Parameter 1: The string to be appended 1.

Parameter 2: String 1 to be appended.

The function return value: a string type variable of string 1 + string 2.

Appendix to Chapter 1

11.1 Error Codes

Currently, there are a total of 6 information levels for the robot. The fourth digit of the error code indicates the error level.

No.	Error & Level
0	System occupancy
1	System prompt
2	Alert
3	General Error
4	Critical error
5	Fatal error

When general errors or more serious issues occur, the robot will power off and stop operating.

When a warning-level error occurs, the robot will slow down and stop.

If multiple errors occur at the same time, the one with the highest severity level will be executed.

There will only be one error code for the same type of error, but the specific error content will be displayed on the teaching pendant.

Error Code	Description
FFF10000	Undefined Hints
FFF20000	Undefined warning
FFF30000	Undefined error
FFF40000	Undefined Critical Error
50010000	Robot power-up prompt
50010001	Robot power down prompt
50010002	Robot encoder calibration prompt
50030003	Robot state switching timeout
50040004	Abnormal axis status
50030005	Unusual position at pointing
50010006	Reset
50030007	Reset timeout
50030008	Joint position overrun
50030009	End position overrun
5003000A	Joint desired position jump
5003000B	Joint output torque jump
5003000C	Joint tracking error too large
5003000D	Joint speed overrun
5003000E	Joint collision detection trigger
5003000F	Unable to effectively calculate joint collision detection
50030010	Unable to calculate end collision detection effectively
50030011	End collision detection trigger
50030012	End speed overrun
50030013	Error while dragging
50030014	Error when dragging stops

50030015	Cannot perform endpoint movement
50020016	Error when resetting motion planner
50020017	Error setting initial position of Motion Planner
50020018	Error in adding commands to the Motion Planner
50030019	Emergency stop
5002001A	Parameterization is in progress.
5003001B	Error during parameterization
5002001C	Emergency stop pressed during power-up
5002001D	Desired joint speed jump
5002001E	Drag and drop overspeed
5002001F	Configuration parameters changed during motion
58020000	Illegal IO configuration
58020001	Illegal bus configuration
59020000	Welder current setting error
59020001	Wrong welder voltage setting
60020000	Motion planner path calculation error
60020001	Motion planner operation error
60020003	Node data to json failed
60020004	Failed to get shared memory node
60030003	Failed to get inverse matrix of robot's velocity Jacobi matrix
60030004	Failed to get the inverse matrix of the force Jacobi matrix of the robot.
60030005	Failed to get the positive kinematic position of the robot
60030006	Failed to get the positive kinematic velocity of the robot
60030007	Cannot get the inverse kinematic position of the robot
60030008	Cannot get the inverse kinematic velocity of the robot
60030009	Wrong robot setup
6003000A	Joint overrun
6003000B	Unable to get robot inverse kinematics
6003000C	Cannot get robot joint equivalent moments of inertia
6003000D	Cannot get robot joint equivalent gravitational moments
6003000E	Unable to get the equivalent kinematic moments of the robot joints.
6003000F	It is not possible to obtain the inertia matrix of a robot dynamics model.
60030010	It is not possible to obtain the gravity matrix for a robot dynamics model.
60030011	It is not possible to obtain the scientific force matrix of the robot dynamics model.
60030012	It is not possible to obtain the rotation matrix from the base coordinate system to the flange coordinate system of the robot.
61010000	Unknown file
61010001	File parsing error
61010002	File loading error
61010003	Format-specific file conversion error
61010004	Format-specific file write error
70020000	Fitting matrix is not full of rank
70020001	Calibrated triple point covariance
71020000	Robot initial position unknown
71020001	Insufficient initial conditions, wait for additional conditions, no error reported.
71020002	Input reference coordinate system type does not exist during relative

	motion.
71020003	Transition type unknown
71020004	Point type unknown
71020005	Arc type unknown
71020006	Move command queue is full
71020007	Velocity is not normal
71020008	Unable to create path
71020009	Index out of range
7102000A	Failed to solve
7102000B	Trajectory planning failed
7102000C	Move type does not exist
7102000D	Move type mismatch
7102000E	Trigger type does not match
7102000F	The Move command Id for the trigger does not exist.
71020010	Path attribute does not exist
71020011	Trigger type does not match
71020012	The Move command Id of the trigger does not exist.
71020013	Position point does not exist
71020014	Motion magnification is out of range
71020015	The number of points exceeds the maximum value
71020016	Parameter error
71020017	Spline interpolation failure
71020018	Index update failure
71020019	Failed to get arm angle
76020000	Oscillation type not present
76020001	Oscillation amplitude is negative
76020002	Oscillation frequency is negative
76020003	Oscillation angle is negative
76020004	Operation angle is negative
76020005	Left dwell time is negative
76020006	Negative right dwell time
76020007	Frequency too low
76020008	Frequency too high
76020009	Dwell time too long
7602000A	Azimuth too large
7602000B	Path type does not exist
7602000C	Weld direction is the same as the Z direction of the current tcp, unable to determine the swing direction
7602000D	Compensation method does not exist
7602000E	Compensation value update failure
7602000F	Incorrect number of sampling periods
76020010	The number of sampling cycles used for reference value calculation is wrong
76020011	Attitude correction failure
76020012	Point position update failure
76020013	Surfacing error
78030000	Input parameter dimensions do not match the robot
78030001	External force estimator initialization failure
78030002	External force estimator did not set initial state
78030003	Kalman filter built into the external force estimator fails to update the output.

78030104	Unable to get the joint external force estimated by the external force estimator.
78030105	Cannot get the joint acceleration estimated by the external force estimator.
78030106	The collision detector was not initialized successfully.
78030107	Cannot get the status of collision detection.
78030108	Failure to initialize the conductivity controller.
78030109	The parameter setting of the guider controller is wrong.
7803010A	Unable to update the output of the joint guide program.
7803010B	The end space axis lock was not initialized successfully.
7803010C	Unable to set the direction of end space lock
7803010D	Cannot get the end impedance force of end space lock.
7803010E	Unable to get the impedance force of the end locking axis converted to the joint end.
7803010F	Unable to update the output of the teach-in program
78030110	The joint limit in drag mode is exceeded.
78030111	Unable to acquire external force from the six-dimensional force transducer
78030112	Constant force tracking & soft force control is turned on at the same time during force control, not allowed.
80030000	Joint tracking error overrun trigger
80030001	Joint collision detection trigger
80030002	Joint position limit trigger
80030003	Joint speed limit trigger
91010000	WHILE control expression is empty
91010001	IF control expression is empty
91010002	ELSEIF control expression is empty
91010003	ELSE control followed by ELSEIF
91010004	Unknown operator
91010005	The variable name of the data is not of string type
91010006	Wait time parameter is not an integer
91010007	Control parameter is not legal
91010008	Control type is not legal
91021007	Failed to open configuration file
91011008	Failed to save global variable
91011009	Failed to get global variable
9101100A	Failed to save project variables
9101100B	Failed to get project variables
9101100C	Failed to save project
9102100D	Failed to read project file
9102100E	Failed to read lua file
92020000	Array variable index out of range
92020001	Failed to find variable by variable name
92020002	Unknown Variable Type
92020003	Failed to find IO port
92020004	Request parameter error
93010000	Setting shared memory node failed
93010001	CPOS to APOS failed
93010002	APOS to CPOS failed
93010003	Point data calculation failure
93010004	Motion kernel state error

93010005	Calibration Failure
94010002	Subscribed topic does not exist
94010003	Failed to open topic configuration file
94010004	Failed to parse topic configuration file
94010005	Duplicate topic name
94010006	Memory node corresponding to topic not found
96010000	Unknown command parsed
96020001	Failed to load instruction
96020003	Kernel state does not support this directive
96020004	Project status does not support this instruction
96020005	Invalid project control command
96020006	Failed to load project data
96020007	Project load failed
96020008	Invalid control ID for project start run
91010009	Project started running
9101000A	Project stopped running
9101000B	Task status error
97020000	Too many addDo commands
97020001	Jump control does not exist
97020002	Illegal IO port number parameter
97020003	Lua execution expression fails
97020004	Invalid task control instruction
97020005	AddDo instruction execution failure
97020006	Failed to execute an instruction in the Waiting for Execution instruction queue
97020007	Execution of unknown instruction
97020008	Lua load instruction failed
97020009	Failed to execute instruction in lua
9702000A	Failed to write instruction to motion kernel
9702000B	Failed to update AddDo instruction status
9702000C	Registering variables to lua failed
9702000D	Failed to initialize lua
9702000E	Failed to load initialization for lua configuration scripts
9702000F	Unknown user variable type
97020010	Failed to create Path
97020011	Calculate Path failed
97020012	Failed to run Path
97020013	OnDistance cannot be associated with MovJ command
97020014	Invalid parameter

11.2 User Levels and Permissions

Category	Function	user	admin
Project	New	✓	✓
	Switch	✓	✓
	Save	✓	✓
	Copy	✓	✓

	Download	✓	✓
	Delete	✓	✓
	Import	✓	✓
	Autorun	✓	✓
	Stop	✓	✓
	Single-step run	✓	✓
	Run pointer	✓	✓
	Single-task and multi-task switching	✓	✓
Visual Programming	Control View	✓	✓
	Drag and Drop Commands	✓	✓
	Add Command	✓	✓
	Command Selection	✓	✓
	Command Properties Editing	✓	✓
	Command Copy	✓	✓
	Command Delete	✓	✓
	Expanding and Collapsing Tree Commands	✓	✓
	Command Attribute Editing	✓	✓
	Checksums		
	Conditional expression checking	✓	✓
	Target value checking for goto type instructions	✓	✓
	Check result message	✓	✓
Positioning	Add Position	✓	✓
	Deleting poses	✓	✓
	Duplicating poses	✓	✓
	Adding a pose from a mov control	✓	✓
	Updating a pose from a mov-like control	✓	✓
Variables	Adding Variables	✓	✓
	Deleting Variables	✓	✓
	Editing Variables	✓	✓
	Displaying Variables	✓	✓
	Running variables in real time	✓	✓
	Adding a Variable of a Specified Type from the Control Properties	✓	✓
Setup	Basic	✗(No authority)	✗(No authority)
	Mechanics - Installation	✗(No authority)	✓
	Mechanical - Relative to World Coordinate System	✗(No authority)	✓
	Mechanical - DH	✗(No access)	✗(No access)
	Safety - Joint/End Limits	✗(No authority)	✓
	Safety - Other	✗(No authority)	✓
	Motion - automatic mode	✗(No authority)	✓
	Motion - manual mode	✗(No authority)	✓

	Motion-Servo	X(No access)	X
	Debugging	X	X
3D Simulation	Simulation Show	✓	✓
	Switching Viewpoints	✓	✓
	Clearing the trajectory line	✓	✓
	Return to zero position	✓	✓
	Return to packing position	✓	✓
	Switch coordinate system	✓	✓
	Teach mode configuration	✓	✓
	Automatic mode	✓	✓
	Manual mode	✓	✓
	Nodal movement	✓	✓
	End Point Motion	✓	✓
	I/O Configuration	✓	✓
	Peripherals	✓	✓
Logging	View	✓	✓
	Download	✓	✓
Plug-ins	Welding Process Template List	✓	✓
	Add Template	✓	✓
	Edit Template	✓	✓
	JOB number selection	✓	✓
Monitoring	Monitoring System	X	X
	Specify monitoring data	X	X
Debugging	Send path data	X	X
	Debug Data Cache	X	X
Configuration	Changing configuration values	X	X
	Changing Configuration Structure	X	X
User	Registering a new user	X	✓
	Deleting a user	X	✓
Bus	Register Editing	X	✓
Error Messages	Clearing Errors	✓	✓
	Reset	✓	✓
	Real-time logging	✓	✓
Other Functions	Undo and Redo	✓	✓
	Reload Configuration	✓	✓
	Refresh Page	✓	✓
	Maximizing the Module Window	✓	✓
	Closing the Module Window	✓	✓
	Online Settings	✓	✓
	Online status	✓	✓
	Lock Window	✓	✓
	Switch between Chinese and English	✓	✓
	Trace ID related functions	X	X

11.3 Declaration

DECLARATION OF INCORPORATION

According to the following EU Directive(s)
Machinery Directive: 2006/42/EC
Electromagnetic Compatibility Directive: 2014/30/EU

Manufacturer: NANJING ESTUN CODROID TECHNOLOGY CO., LTD.
Address: 5/F, Building 1, Jiangning Double Innovation Base, National University Science Park, Southeast University, No.33, Southeast University Road, Jiangning District. Nanjing, 211102 Jiangsu, P. R. China

Declares that the machine described here after
Product name: Collaborative Robots
Model(s): S3-60 Eco,S5-90 Eco,S10-140 Eco,S20-180 Eco
S3-60 Pro,S5-90 Pro,S10-140 Pro,S20-180 Pro
Serial No.:

Provided that it is used and maintained in accordance with the general accepted codes of good practice and the recommendations of the instruction manual, meet the essential safety and health requirements of the Machinery Directive
Person authorized to compile the technical file:
Name: ESTUN Robotics Europe AG
Address: Graben Strasse 25,6340 Baar,Switzerland

We confirm that:
a) The specific technical documents which we provided pursuant to Appendix VII Part B
b) The assembly instructions which are provided pursuant Appendix VI
c) The declaration of incorporation which we provided pursuant to Appendix II Part 1 Section B of Directive 2006/42/EC
Upon justified request, we provide specific documents regarding the products listed above within an adequate period. The document will be made available via e-mail
The following essential health and safety requirements are executed and observed according to annex I of the directive specific above:
1.1.1, 1.1.2, 1.1.3, 1.1.5, 1.1.6, 1.3.1, 1.3.2, 1.3.3, 1.3.4, 1.3.7, 1.3.8, 1.3.9, 1.4.1, 1.4.2.1, 1.4.3, 1.5.2, 1.5.4, 1.5.5, 1.5.6, 1.5.8, 1.5.9, 1.5.11, 1.5.14, 1.5.15, 1.6.1, 1.6.2, 1.6.4, 1.6.5, 1.7.1, 1.7.2, 1.7.3, 1.7.4
For the most specific risks of this machine, safety and compliance with the essential requirements of the Directive has been based on elements of:
EN ISO 12100:2010 Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1: 2018 Safety of machinery - Electrical equipment of machines - Part 1: General requirements
EN IEC 61000-6-2:2019 Electromagnetic compatibility (EMC) - Part 6-2: Generic standards - Immunity for industrial environments

EN IEC 61000-6-4:2019 Electromagnetic compatibility (EMC) - Part 6-4: Generic standards - Emission standard for industrial environment

The relevant technical documentation is compiled in accordance with part B of Annex VII 2006/42/EC, and are available by reasoned requested by national authorities via proper methods.
It must not be put into service until the final machinery into which it is to be incorporated has been declared in conformity with the provisions of the above Directives.

Print Name: Ge QingQing Position: Chief Engineer of Robot Development
Signature: 葛庆卿 Date: 2024.9.6
Place: 5/F, Building 1, Jiangning Double Innovation Base, National University Science Park, Southeast University, No.33, Southeast University Road, Jiangning District. Nanjing, 211102 Jiangsu, P. R. China

Chapter 12 Spare Parts List

The spare parts and consumables system includes vulnerable and consumable components. The items listed in the table and their service lives are for reference only. The actual condition depends on the frequency of use and maintenance.

No.	Name	Model	Brand	Durability
1	Key Switch	NP6-22Y2	CHINT	100000 times
2	Enable Button (black)	HE6B-M200BPN10	IDEC	100000 times
3	Button	HBGO12SH-10W/J/S(N)	HBAN	50000 times
4	Button	MP16S/F11-EDY -24V/B	CMP	100000 times
5	Button	NP6-22ZS	CHINT	100000 times
6	Switch Power Supply	LMFXXXX-20B48	MORNSUN	25000h
7	Fan	JC6025B24UC2	JENCE	70000h
8	Thermal Protect	BW-BCM-95°C	SAFTTY	10000cycles

Chapter 13 Contact Information



China:

Nanjing Estun Codroid Technology Co., Ltd.

5F, Building 1, Southeast University National Science Park -
Jiangning Branch, No. 33 Southeast University Road, Jiangning District, Nanjing

Service: +86-400-025-3336

Europe:

ESTUN Robotics Europe AG

Graben Strasse 256340 Baar, Switzerland

To improve the product, the specifications, ratings and dimensions of this product may be changed without further notice.

For inquiries regarding the content of this document, please contact our sales department.