

# MG400 Hardware

## User Guide

### (DT-MG400-4R075-01)

---

Issue: V1.4

Date: 2022-03-03

**Copyright © Shenzhen Yuejiang Technology Co., Ltd 2022. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without the prior written consent of Yuejiang Technology Co., Ltd.

### **Disclaimer**

To the maximum extent permitted by applicable law, the products described (including its hardware, software, and firmware, etc.) in this document are provided **AS IS**, which may have flaws, errors or faults. Yuejiang makes no warranties of any kind, express or implied, including but not limited to, merchantability, satisfaction of quality, fitness for a particular purpose and non-infringement of third party rights. In no event will Yuejiang be liable for any special, incidental, consequential or indirect damages resulting from the use of our products and documents.

Before using our product, please thoroughly read and understand the contents of this document and related technical documents that are published online, to ensure that the robot is used on the premise of fully understanding the robot and related knowledge. Please use this document with technical guidance from professionals. Even if follow this document or any other related instructions, damages or losses will be happening in the using process. Dobot shall not be considered as a guarantee regarding all security information contained in this document.

The user has the responsibility to make sure of following the relevant practical laws and regulations of the country, in order that there is no significant danger in the use of the robot.

## **Shenzhen Yuejiang Technology Co., Ltd**

Address: Floor 9-10, Building 2, Chongwen Garden, Nanshan iPark, Liuxian Blvd, Nanshan District, Shenzhen, Guangdong Province, China

Website: [www.dobot.cc](http://www.dobot.cc)

## Preface

### Purpose

This Document describes the functions, technical specifications, installation guide of DOBOT MG400 robot, making it easy for users to fully understand and use it.

### Intended Audience

This document is intended for:





- Customer
- Sales Engineer
- Installation and Commissioning Engineer
- Technical Support Engineer

### Change History

Date	Change Description
2021/02/06	The first releases
2021/04/06	Add description of hand-teaching button and diameter of the air interface
2021/07/26	Add input and output circuit examples in different application scenarios
2021/08/23	Updated end-effector size, and add reserved mounting hole
2022/03/03	Updated the sequence of <b>3. Electrical Specifications</b> , and modified the motion range of J4 joint

### Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
 DANGER	Indicates a hazard with a high level of risk which, if not avoided, could result in death or serious injury
 WARNING	Indicates a hazard with a medium level or low level of risk which, if not avoided, could result in minor or moderate injury, robot damage
 NOTICE	Indicates a potentially hazardous situation which, if not avoided, can result in equipment damage, data loss, or unanticipated result
 NOTE	Provides additional information to emphasize or supplement important points in the main text

## Contents







<b>1. Security Precautions .....</b>	<b>1</b>
1.1 Security Warning Sign .....	1
1.2 General Security.....	1
1.3 Personal Security .....	4
<b>2. Overview .....</b>	<b>6</b>
2.1 Technical Specifications .....	6
2.2 Robot Dimension .....	7
2.3 Robot Workspace .....	8
2.4 End-effector Size .....	9
2.5 End-effector Load Description.....	10
2.6 Stop Time and Angle.....	11
2.7 Home Description .....	11
2.8 Factory point .....	11
2.9 Product Features .....	12
2.9.1 Motion Function .....	12
2.9.2 Coordinate System.....	14
2.9.3 Collision Detection.....	16
<b>3. Electrical Specifications.....</b>	<b>17</b>
3.1 Interface Description.....	17
3.1.1 Base Interface Board .....	17
3.1.2 Forearm Interface Description.....	18
3.2 I/O Interface Description .....	19
3.2.1 Base I/O Interface .....	19
3.2.2 End I/O Interface .....	19
3.3 Digital Circuit Description.....	19
3.3.1 Digital Input .....	19
3.3.2 Digital Output.....	21
<b>4. Installation .....</b>	<b>23</b>
4.1 Installation Environment.....	23
4.2 Installation Location .....	23
<b>5. Maintenance and Repair .....</b>	<b>24</b>
5.1 Safety Instructions .....	24
5.2 Body Maintenance .....	24
<b>Appendix A Servo Alarm Description .....</b>	<b>26</b>
<b>Appendix B Controller Alarm Description .....</b>	<b>31</b>

## 1. Security Precautions

This topic describes the security precautions that should be noticed when using this product. Please read this document carefully before using the robot for the first time. This product needs to be carried out in an environment meeting design specification. You cannot remold the product without authorization, otherwise, it could lead to product failure, and even personal injury, electric shock, fire, etc. People who use this product for system design and manufacture must be trained by our company, relevant institution, or must have the same professional skills. The installation personnel, operators, teaching personnel, programmers and system developers of the robot must read this document carefully and use the robot strictly according to the regulations of this document strictly.

### 1.1 Security Warning Sign

The following safety warning signs may appear in this manual, and their meanings are as follows.

Sign	Description
 DANGER	Indicates a high degree of potential danger, which, if unavoidable, will result in death or serious injury
 ELECTRICITY	Dangerous power consumption will soon be caused. If it cannot be avoided, it will cause personal injury or serious injury to the equipment.
 HOT	May cause dangerous hot surfaces, if touched, may cause personal injury
 WARNING	Indicates that there is a moderate or low potential hazard. If it cannot be avoided, it may cause minor injuries to the equipment and damage to the equipment.
 ATTENTION	Indicates a potential risk, and ignoring these texts may result in damage to the robotic arm, loss of data, or unpredictable results
 NOTICE	A situation that, if unavoidable, can cause personal injury or equipment damage  For items marked with such symbols, depending on the specific situation, there is sometimes the possibility of significant consequences

### 1.2 General Security

The following security rules should be followed when using the robot for industrial design and manufacture.



- Robot is electrical equipment. Non-professional technicians cannot modify the circuit, otherwise, it can injure the device or the person.
- You should comply with the local laws and regulations when operating the robot. The security precautions in this document are only supplemental to the local laws and regulations.
- Please use the robot in the specified environment scope. If not, exceeding the specifications or load conditions will shorten the service life of the robot, even damage it.
- Please ensure that the robot is operated under the security conditions and there is no harmful object around the robot.
- Turning on or off the power continually may result in that the performance of the main circuit components inside the robot is degraded. If turning on or off the power continually is required, please keep frequency less than once a minute.

 NOTICE

- The personnel responsible for installation, operation and maintenance of equipment must first undergo rigorous training, understand various safety precautions, and master the correct operation and maintenance methods before they can operate and maintain equipment.
- Personnel without professional training shall not disassemble and repair the equipment without authorization. If the device fails, please contact Shenzhen Yuejiang Technology Co., Ltd technical support engineer in time.
- Be sure to carry out daily inspections and regular maintenance, and replace faulty components in time to ensure the safe operation of the equipment.
- If the equipment is scrapped, please comply with relevant laws to properly handle industrial waste and protect the environment.
- In order to prevent personnel from accidentally entering the working space of the robotic arm, be sure to set up safety fence to prevent personnel from entering the hazardous area.
- Before operating the robot, make sure that no one is inside the safety fence. When operating the robot, be sure to operate outside the safety fence.
- Do not expose the robot to permanent magnetic fields all the time. Strong magnetic fields can cause damage to the robot.
- Shenzhen Yuejiang Technology Co., Ltd. assumes no responsibility for robot damage or personal injury caused by failure to follow product instructions or other improper operations.
- Shenzhen Yuejiang Technology Co., Ltd. is not responsible for the damage caused during the transportation and handling of equipment.
- Please make sure that the robot is in the packing posture before packaging, and the brakes on each axis are normal.

- When the robot is transported, the packaging needs to be fixed to ensure that the robot is stable.
- After removing the outer packaging, please make sure that the robot maintains the original packing posture and the brakes on each axis are normal.
- During the commissioning process, it is necessary to confirm that no relevant personnel and equipment (include computer used for debugging) stay in the dangerous area of the machine.
- If necessary, wear corresponding safety protective equipment, such as safety helmets, safety shoes (with non-slip soles), face shields, protective glasses and gloves. Inappropriate clothing may cause personal injury.
- In order to prevent personnel from entering the working space of the robot arm by mistake, please set up safety barriers to prevent personnel from entering the hazardous area.
- Do not enter the working space of the manipulator at will during operating the robot, otherwise cause injury to the robot or yourself.
- When an abnormality occurs in the mechanical arm, it is necessary to ensure that the machine is stopped and then checked.
- If the controller needs to be restarted due to power failure, when restarting, the robot must be manually returned to the initial position of the automatic operation program before restarting the automatic operation.
- Before maintenance and wiring work, the power supply must be cut off, and the sign **No power supply** must be put on. Otherwise, electric shock and personal injury may result.
- Please contact our technical support staff for the disassembly and repair of the robot.
- Maintenance and repair work must be carried out by designated personnel, otherwise electric shock and personal injury may result.
- If the brake is manually released, the robot may move because of the action of gravity. So, when manually releasing the brake, please ensure that the robot body and the tools or workpieces installed on the robot are effectively supported.
- In order to prevent electric shock, when replacing parts, please turn off the circuit breaker in advance and cut off the main power before proceeding.
- Turn off the main power supply for 5 minutes before replacing parts.
- The replacement operation must be performed by the specified operator.
- The robot is designed and tested according to the group I class A engineering medical robot standard. In order to reduce the radio interference in light industry or family environment, please take protective measures.
- It is prohibited to operate the robot in strong radiation environment, for example,

RF source without shielding, otherwise, it could lead to robot abnormally.

 **WARNING**

- In order to protect the equipment and personal safety, when turning off the power, please press the ship type switch, then unplug the AC power cable.
- Before the operation, please wear protective clothing, such as antistatic uniform, protective gloves, and protective shoes.
- It is prohibited to modify or remove the nameplates, instructions, icons, and marks on the robot and the related equipment.
- Before operating and maintaining the robot, the personnel responsible for the installation, operation and maintenance must be trained to understand the various security precautions and to master the correct methods of operation and maintenance.
- Be careful during the robot carrying or installing. Please follow the instructions on the packing box to put down the robot gently and place it correctly in direction of the arrow.
- Please use the matched cables when connecting a robot to internal or external equipment for personal security and equipment protection.
- Please ensure that robot and tools are installed correctly.
- Please ensure that the robot has enough space to move freely.
- If the robot is damaged, please do not continue to use it.
- Any impact will release a lot of kinetic energy, having a much more significant effect than that under high speed and high load.

### 1.3 Personal Security

When operating the robot system, it is necessary to ensure the personal safety of the operator. The general precautions are listed below, please strictly follow.

 **WARNING**

- To reduce the risk of personal injury, please comply with local regulations with regard to the maximum weight one person is permitted to carry.
- Do not touch the terminal blocks or disassemble the equipment with the power **ON**. Otherwise, it may result in an electric shock
- Please confirm that the equipment is well grounded, otherwise it will endanger personal safety.
- Do not touch the terminal blocks or remove the interval circuit components in 10 minutes after the power is shut off, to avoid an electric shock since there is residual capacitance inside the robot.



- Even if the power switch of the robot is already in the **OFF** status, touching the terminal blocks or removing the interval circuit components is not allowed, to avoid an electric shock since there is residual capacitance inside the robot.
- When working with robots, please do not wear loose clothing or jewelry. When operating the robot, make sure that the long hair bundle is behind your head.
- If the robot appears to have stopped during the operation of the equipment, it may be because the robot is waiting for the start signal and is in the state of being about to move. In this case, the robot should also be considered to be in motion, please do not approach the robot.
- Please ensure that the robot establishes safety measures near the operation area, such as guardrails, to protect the operator and surrounding people.

## 2. Overview

The collaborative robot work system is composed of the collaborative robot body, robot control software, and robot operation software. DOBOT MG400 supports direct connection with computer, which is really simple and easy to use. With the self-developed dynamic algorithm, one-handed teach-in and sensor less collision detection are realized to ensure the safety of human and machine working together. DOBOT MG400 has a repeat positioning accuracy of  $\pm 0.05\text{mm}$ , a max load of 500g. It is a product with the advantages of both industrial robots and collaborative robots.



Figure 2.1 DOBOT MG400

### 2.1 Technical Specifications

Table 2.1 MG400 technical parameters

Product	DOBOT MG400
Model	DT-MG400-4R075-01
Weight	8kg
Max load	500g
Reach	440mm
Power adapter	100V~240V AC, 50/60Hz, Max. 240W
Rated voltage	DC48V
Installation	Table installation, indoor
Rated power	150W
Repeatability	$\pm 0.05\text{mm}$
Base size	190mm * 190mm

Operation software	DobotStudio2020、SCStudio	
Motion range	J1	$\pm 160^{\circ}$
	J2	$-25^{\circ} \sim 85^{\circ}$
	J3	$-25^{\circ} \sim 105^{\circ}$
	J4	$-360^{\circ} \sim 360^{\circ}$
Joint maximum speed	J1	300 %s
	J2	300 %s
	J3	300 %s
	J4	300 %s
End-effector I/O interface	DI	2
	DO	2
Base interface	DI	16
	DO	16
	ABZ incremental encoder (differential)	1
	Ethernet	2
	USB 2.0	2
Communication mode	TCP/IP, Modbus, TCP	
Temperature range	Storage temperature: $-25^{\circ}\text{C} \sim 55^{\circ}\text{C}$ Working temperature: $0^{\circ}\text{C} \sim 40^{\circ}\text{C}$	
Operating altitude range	$\leq 1000$ m	
Safety Standard	EN ISO 10218-1:2011 Steel wire and wire products. General. Test methods EN 60204-1:2018 Safety of machinery. Electrical equipment of machines. General requirements EN ISO 12100:2010 Safety of machinery. General principles for design. Risk assessment and risk reduction	
EMC Standard	EN 61000-6-2:2019 Electromagnetic compatibility (EMC). Generic standards. Immunity standard for industrial environments EN 61000-6-4:2019 Electromagnetic compatibility (EMC). Generic standards. Emission standard for industrial environments	

## 2.2 Robot Dimension

Figure 2.2 shows the dimension of MG400 robot.

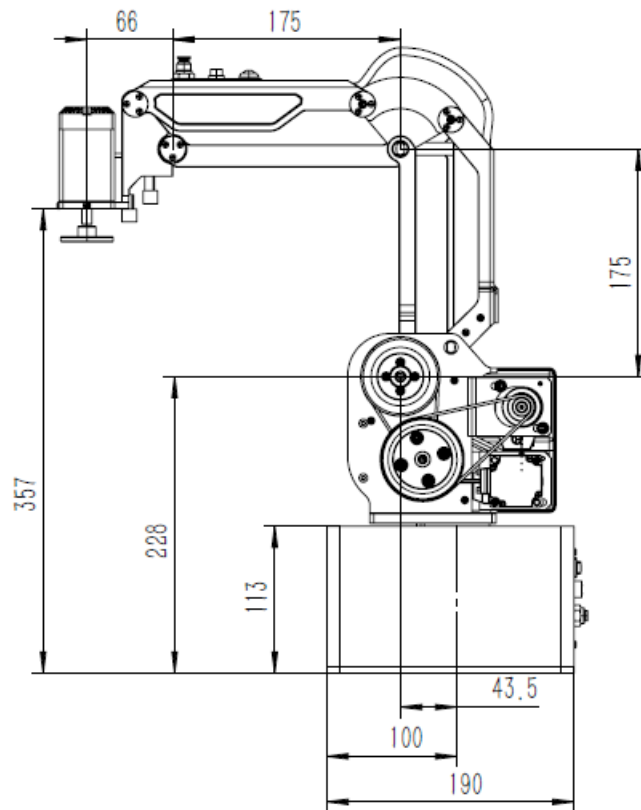


Figure 2.2 MG400 robot dimension

### 2.3 Robot Workspace

Figure 2.3 shows the workspace of MG400 robot.

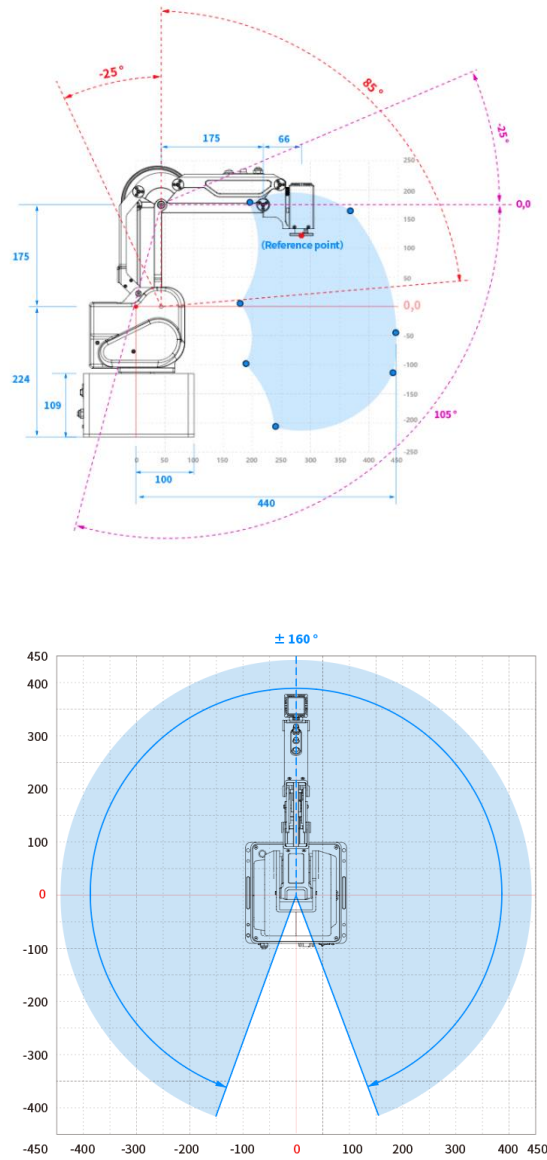


Figure 2.3 MG400 robot workspace

**⚠ NOTICE**

When operating the robot, be sure to operate inside the workspace.

**2.4 End-effector Size**

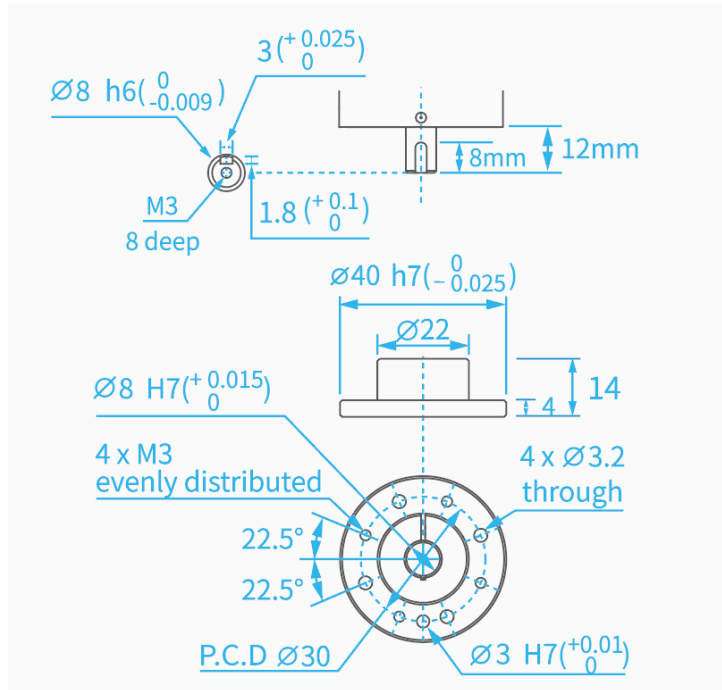


Figure 2.4 End-effector Size

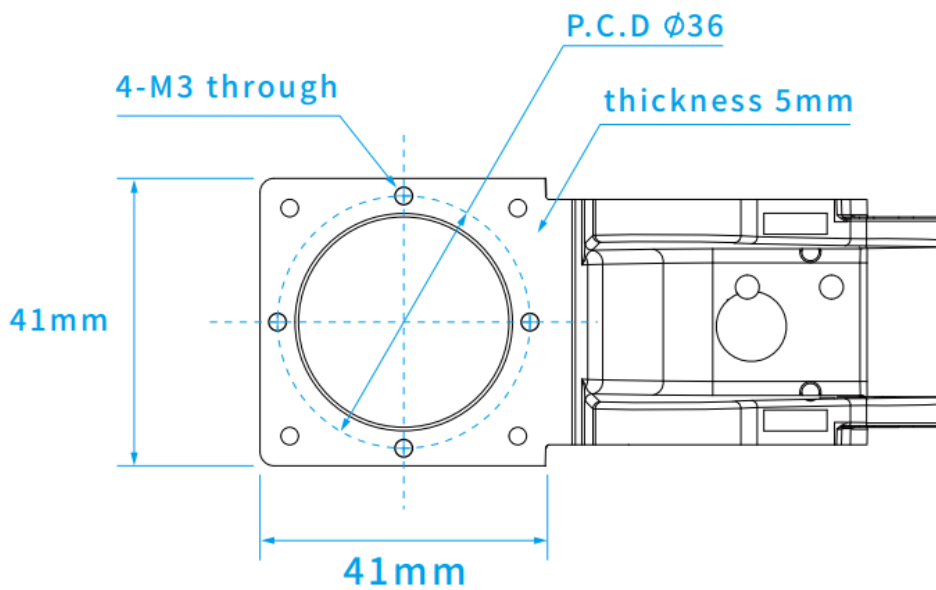


Figure 2.5 Reserved mounting hole

## 2.5 End-effector Load Description

When the rated load is 500g, the maximum eccentricity of the center of mass shall not exceed 40mm.

## 2.6 Stop Time and Angle

The Max. stop time and angle of axis J1, J2, J3 and J4 at the max speed, load and arm stretch are shown below.

Table 2.2 Stop time and angle

Axis	Max. stop angle ( ° )	Max. stop time (ms)
J1	63.391	427
J2	16.13	114
J3	17.951	123
J4	22.027	131

## 2.7 Home Description

After some parts (motors, reduction gear units) of the robot have been replaced or the robot has been hit, the origin of the robot will be changed. You need to reset the origin. For details, please see *DobotSCStudio User Guide (MG400 Robot)*.

## 2.8 Factory point

When the robot leaves the factory, moving robot to the factory point can reduce the robot space, easy to pack and transport. Figure 2.6 shows the factory point. The robot has 4 joints, respectively J1, J2, J3 and J4, please see 2.9.2.1 Joint Coordinate System for explanation of joints. The joint angles corresponding to the factory point are: J1= 0 °, J2= 0 °, J3= 60 °, and J4= 0 °. Adjust joint Angles by jog or programming. For details, please see *DobotSCStudio User Guide (MG400 Robot)*.

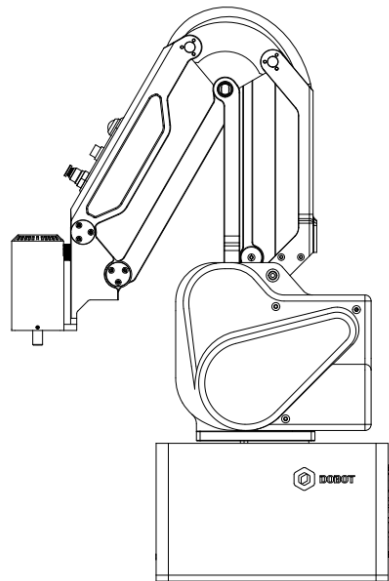


Figure 2.6 Factory point

## 2.9 Product Features

### 2.9.1 Motion Function

The motion trajectory consists of a series of interpolated motions since the interpolated motion is the basic motion type. According to the different trajectories, motion functions are classified as joint interpolated motion, linearly interpolated motion, circular interpolated motion and continuous path. The joint interpolated motion is in the joint space. And the other interpolated motions are in the Cartesian space.

#### 2.9.1.1 Joint Interpolated Motion

Joint interpolated motion includes **Go**, **MoveJ** modes.

- Go/MoveJ: From point A to point B, each joint will run from an initial angle to its target angle, regardless of the trajectory, as shown in Figure 2.7.

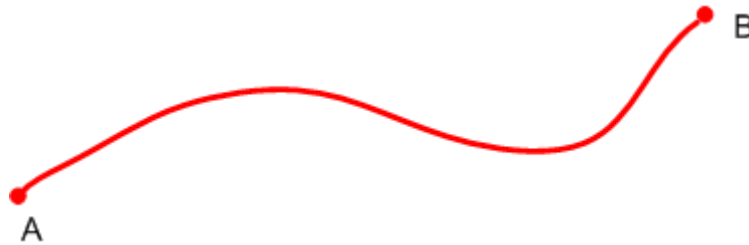


Figure 2.7 Go/MoveJ modes

#### 2.9.1.2 Linearly Interpolated Motion

The joints will perform a straight line trajectory from point A to point B, as shown in Figure 2.8.



Figure 2.8 Move mode

- Jump: The trajectory looks like a door. From point A to point B, the robot will move in the **Move** mode
  1. Move up to the lifting height (**StartHeight** is a relative height).
  2. Move up to the maximum lifting height (**zLimit**).
  3. Move horizontally to a point that is above point **B**.
  4. Move down to a point where the height is point **B** plus the dropping height (**EndHeight** is a relative height).



5. Move down to Point **B**.

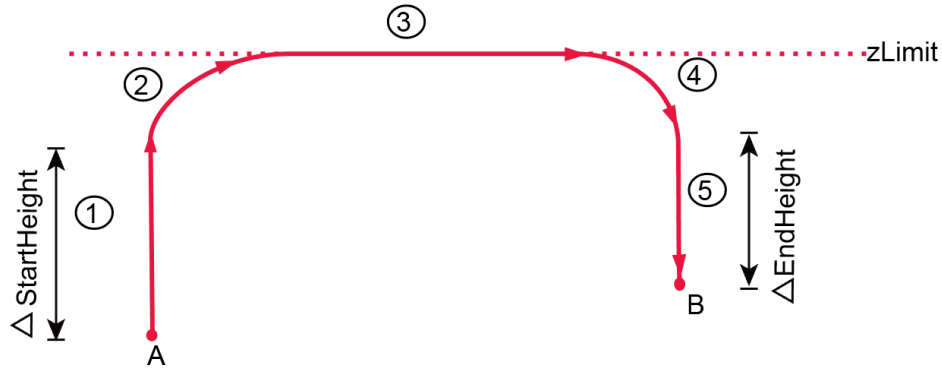


Figure 2.9 Jump mode

NOTICE

- Point **A** and point **B** cannot be higher than **zLimit**. Otherwise, an alarm will be triggered.
- If point **A** plus **StartHeight** or point **B** plus **EndHeight** is higher than **zLimit**, the robot moves up from point **A** to **zLimit** or moves down from **zLimit** to point **B** directly, the trajectory looks like a door without transition, as shown in Figure 2.10.

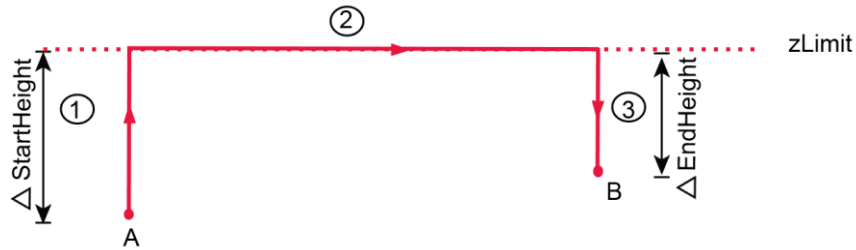


Figure 2.10 Jump mode (1)

- If the heights of point **A** and point **B** are the same with **zLimit**, the trajectory is shown in Figure 2.11.



Figure 2.11 Jump mode (2)

### 2.9.1.3 ARC (Circular Interpolated Motion)

The trajectory is an arc, which is determined by three points (the current point, any point and the end point on the arc), as shown in Figure 2.12.

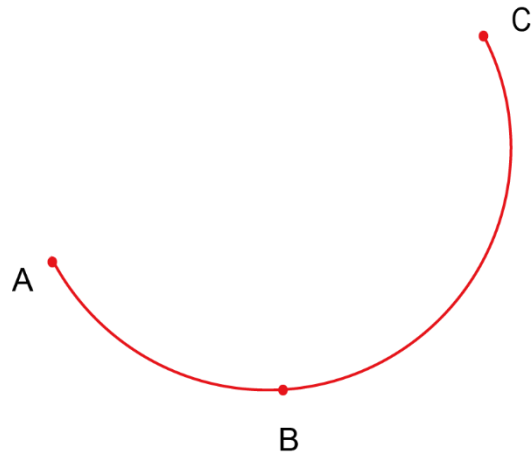


Figure 2.12 Arc trajectory

#### 2.9.1.4 Circle (Circular Interpolated Motion)

The trajectory is a circle, which is determined by three points (the current point, any point and the end point on the arc) as well, as shown in Figure 2.13.

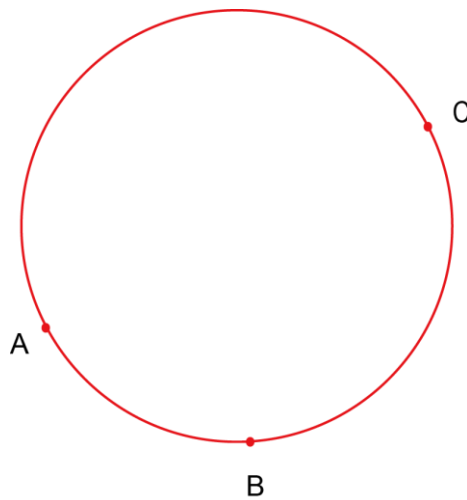


Figure 2.13 Circle trajectory

### 2.9.2 Coordinate System

#### 2.9.2.1 Joint Coordinate System

The Joint coordinate system is determined by the motion joints.

Figure 2.14 shows the Joint coordinate system of a MG400 robot. All the joints are rotating.

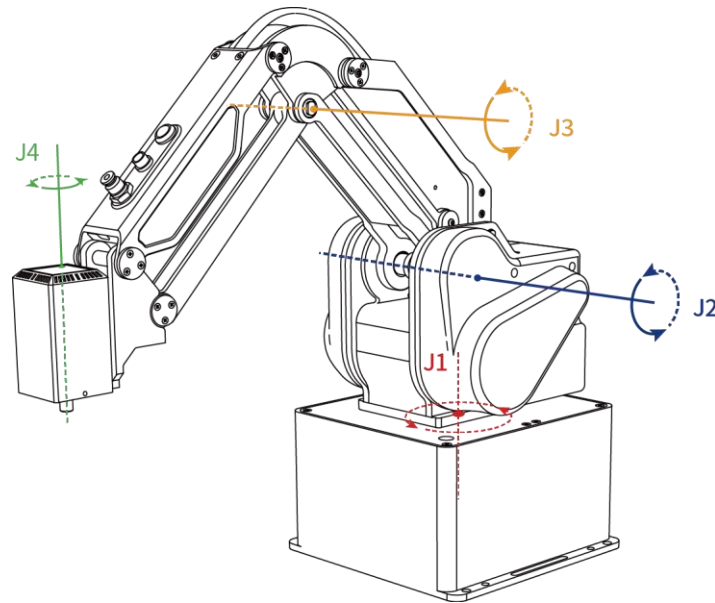


Figure 2.14 Joint coordinate of a MG400 robot

### 2.9.2.2 Base Coordinate System

The Base coordinate system is determined by the base. Figure 2.15 shows the Base coordinate system of MG400 robot.

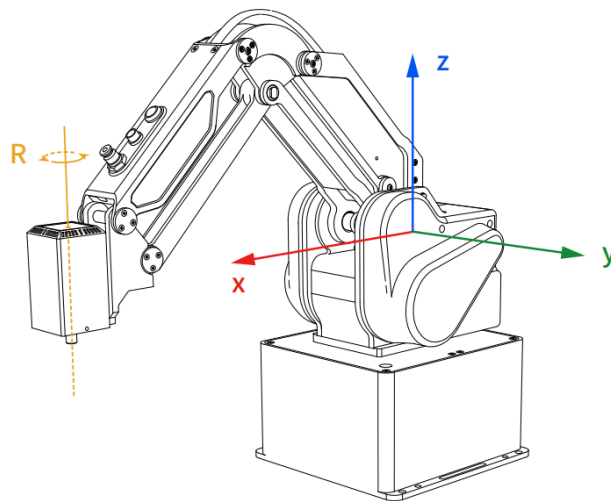


Figure 2.15 Base coordinate system of MG400 robot

### 2.9.2.3 Tool Coordinate System

Tool coordinate system is the coordinate system that defines the distance and rotation angle of the offset, of which the origin and orientations vary with the position and attitude of the workpiece located at the robot flange. The 10 types of tool coordinate systems can be defined. Tool 0 coordinate system is the predefined Tool coordinate system which is located at the robot flange without end effector and cannot be changed. And the others can be customized by users. Figure 2.16 shows the

default Tool coordinate system of a MG400 robot.

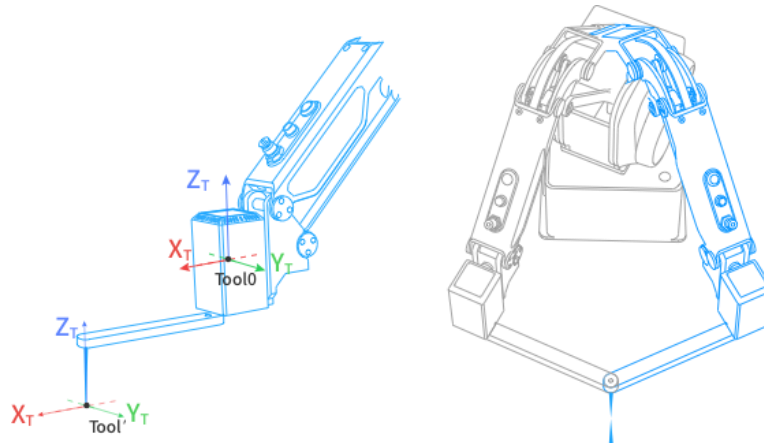


Figure 2.16 The default Tool coordinate system of MG400 robot

### 2.9.2.4 User Coordinate System

The User coordinate system is a movable coordinate system which is used for representing equipment like fixtures, workbenches. The origin and the orientations of axes can be defined based on site requirements, to measure point data within the workspace and arrange tasks conveniently.

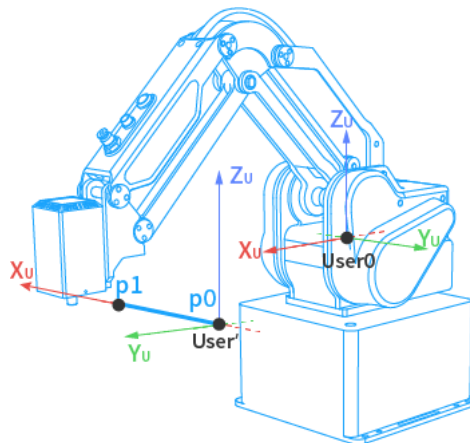


Figure 2.17 The default User coordinate system of MG400 robot

### 2.9.3 Collision Detection

Collision detection is mainly used for reducing the impact on the robot, to avoid damage to the robot or external equipment. If the collision detection is activated, the robot arm will stop running automatically when the robot arm hits an obstacle.

### 3. Electrical Specifications

#### 3.1 Interface Description

##### 3.1.1 Base Interface Board

Figure 3.1 shows the interface board of the Base. Table 3.1 lists the description.

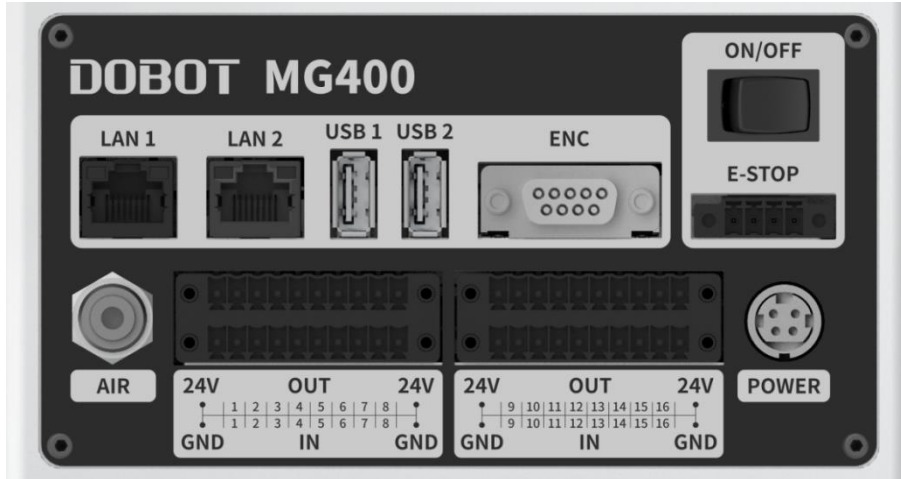


Figure 3.1 Interface board of the base

Table 3.1 Interface description

screen printing	Description
LAN1	LAN interface The default IP address is 192.168.1.6, which cannot be modified. It can be used for software debugging on the upper computer
LAN2	LAN interface For connecting to external equipment. The default IP address is 192.168.2.6, which can be modified.
USB1	USB interface For connecting WiFi module, updating firmware, etc
USB2	USB interface For connecting WiFi module, updating firmware, etc
ENC	Encoder interface For connecting to the conveyor belt for dynamic tracking
ON/OFF	Power switch For controlling the robot power on and off
E-Stop	Emergency stop interface
Power	Power interface

screen printing	Description
	For connecting to DC 48V power supply
I/O	I/O interface
AIR	Air interface, the corresponding trachea diameter was 4mm

The ENC interface of the MG400 is shown in Figure 3.2, Table 3.2 lists the description of ENC interface.

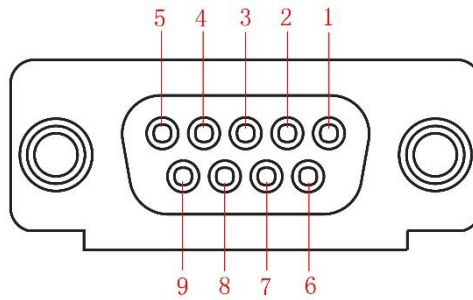


Figure 3.2 ENC interface

Table 3.2 ENC Interface description

No.	1	2	3	4	5	6	7	8	9
Description	ABZ_A+	ABZ_A-	ABZ_B+	ABZ_B-	ABZ_Z+	ABZ_Z-	5V	0V	unused

### 3.1.2 Forearm Interface Description

The forearm interface of the MG400 includes a hand-teach button, an end I/O Interface, and an air interface, as shown in Figure 3.3. The diameter of the trachea corresponding to the air interface is 4mm.

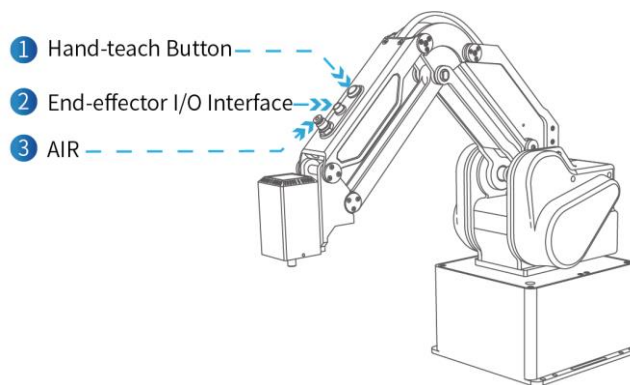


Figure 3.3 Forearm interface

Press the hand-teaching button on the forearm and drag the forearm to a point and then press the button again. In addition, when the power supply is normal, this button can also release the brake to rescue the trapped personnel in an emergency.

**NOTE**

In the process of teaching by pressing the hand-teaching button, it is necessary to support the forearm of MG400 with your hand, otherwise the forearm may lift slowly or fall off.

### 3.2 I/O Interface Description

#### 3.2.1 Base I/O Interface

A robot controller contains I/O interfaces, for connecting to external equipment, such as air pump, PLC, etc. These I/O interfaces provide 16 digital inputs, 16 digital outputs, as shown in Figure 3.4.

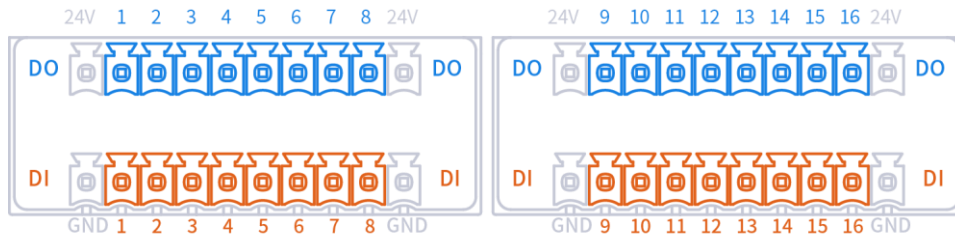


Figure 3.4 I/O interface

**NOTE**

- The output current of every I/O can't exceed 500mA.
- The total current can't exceed 2A.

#### 3.2.2 End I/O Interface

The cable used for the end pins is the designated cable, the model is SF810/P6.

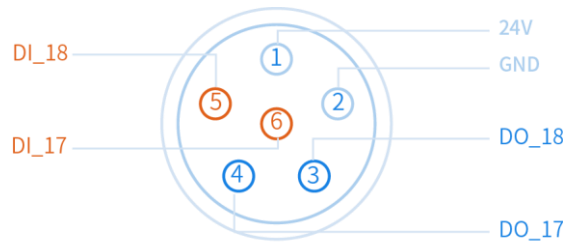


Figure 3.5 end I/O interface

### 3.3 Digital Circuit Description

#### 3.3.1 Digital Input

Figure 3.6 shows the simple digital input circuit and Table 3.3 lists the technical specifications.

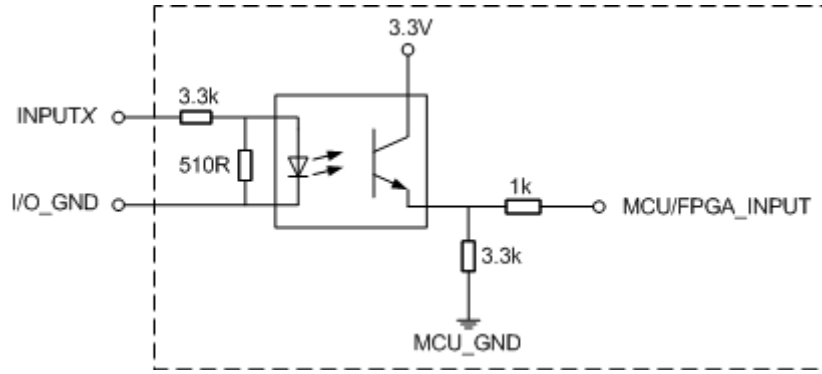


Figure 3.6 Simple digital input circuit

Table 3.3 Technical specifications

Item	Specification
Input channel	16 channels
Connection method	Crimping terminal
Input type	PNP
Input voltage (DC)	24V ±10%
Isolation method	Optical coupling isolation

Figure 3.7 is the circuit diagram of DI external mechanical contact switch (such as relay contact, button, switch, etc.).

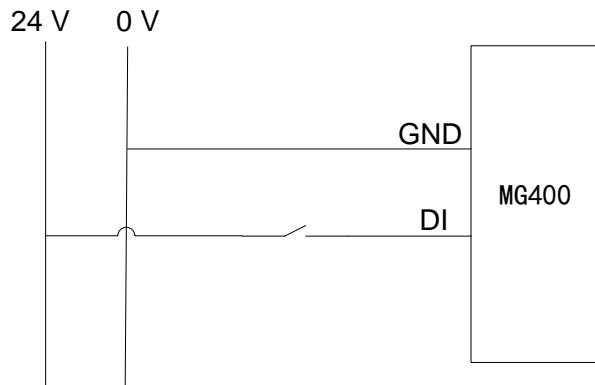


Figure 3.7 DI external mechanical contact switch

Figure 3.8 shows the schematic diagram of DI external PNP three-wire switch.



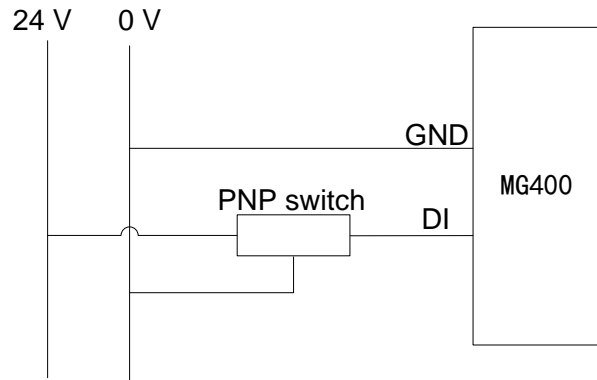


Figure 3.8 DI external PNP three-wire switch

### 3.3.2 Digital Output

Figure 3.9 shows the simple digital output circuit and Table 3.4 lists the technical specifications.

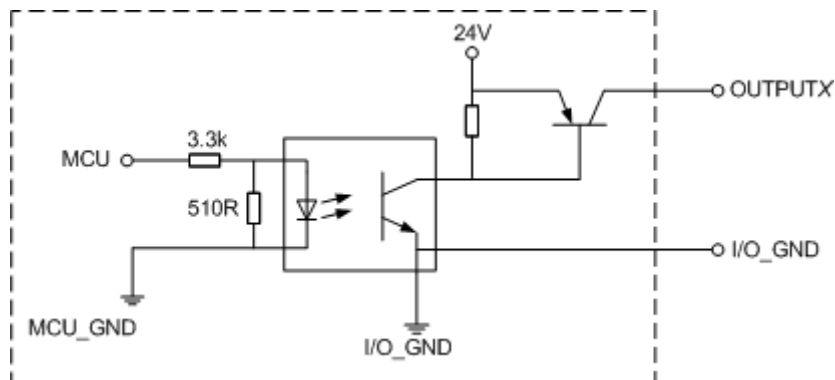


Figure 3.9 Simple digital output circuit

Table 3.4 Technical specifications

Item	Specification
Output channel	16 channels
Connection method	Crimping terminal
Output type	PNP
Power supply (DC)	24V ±10%
Load current of single channel	500mA
Output current	2A
Isolation method	Magnetic isolation

Figure 3.10 is the circuit diagram of DO external load without external power supply. At this time, the load driving current is less than or equal to 500mA. Where, 0V is the grounding terminal corresponding to the external power supply.

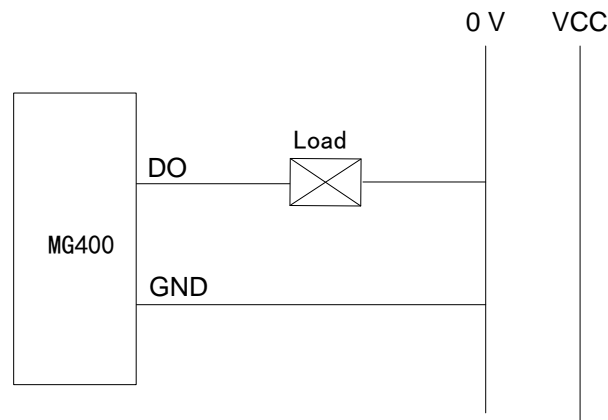


Figure 3.10 DO external load without external power supply

When the load connected through the DO interface requires a large driving capability (that is, the load driving current is greater than 500mA), the default driving capability of the MG400 cannot meet the requirements. In this case, an external driving circuit needs to be connected to increase the driving capability.

Figure 3.11 is the circuit diagram of DO external load with external power supply, where VCC is the external voltage and 0V is the grounding terminal corresponding to the power supply.

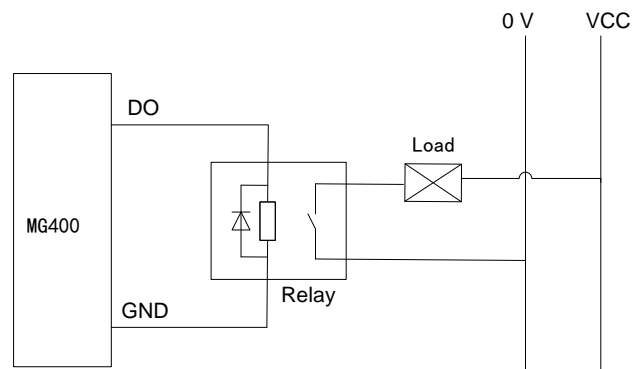


Figure 3.11 DO external load with external power supply

## 4. Installation

### 4.1 Installation Environment

To maintain the controller and robot performance and to ensure the safety, please place them in an environment with the following conditions.

- Install indoors with good ventilation.
- Keep away from excessive and shock.
- Keep away from direct sunlight.
- Keep away from dust, oily smoke, salinity, metal powder, corrosive gases, and other contaminants.
- Keep away from flammable.
- Keep away from cutting and grinding fluids
- Keep away from sources of electromagnetic interference.
- When the robot is installed, corresponding measures should be taken for positioning. You must use four hexagon socket bolts M5 (GB/T 3098.1-2010) and tighten the base of the robot with  $9 \text{ N} \cdot \text{m}$  torque.
- When the robot is installed, the robot must be fixed on a sufficiently strong base. The base must be able to withstand the reaction force of the robot during acceleration and deceleration and the static weight of the robot and the workpiece.

### 4.2 Installation Location

The stability of a robot depends on the installation. You can design the platform according to the size of the hole of the base and the real environment for mounting a robot. And the installation height of the robot should be above 0.6 meters. The platform must not only bear the robot but also bear the dynamic force by the maximum acceleration. Note the following before mounting the robot.

- Design the platform according to the robot's workspace, and ensure that the robot moves without interference.
- Keep the platform level which is used to mount a robot.

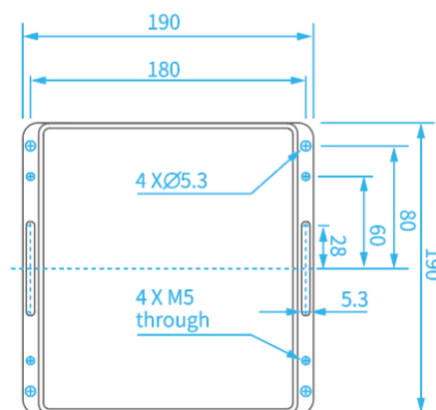


Figure 4.1 Robot base size

## 5. Maintenance and Repair

Maintenance and repairing must be performed in compliance with all safety instructions in this manual.

The purpose of maintenance and repairing is to ensure that the system is kept operational, or to return the system to an operational state in the event of a fault. Repairing includes troubleshooting in addition to the actual repair itself.

Repairing must be performed by an authorized system integrator or Dobot staff.

Robots or parts returned to Dobot should be as the following instructions.

- Remove all parts that do not belong to Dobot.
- Before returning to Dobot, please make a backup copy of the files. Dobot will not be responsible for the loss of programs, data or files stored in robot.
- The robot should move to the package point before returning to Dobot. For details, please see 2.8 Factory point.

### 5.1 Safety Instructions

The following safety procedures and warnings must be observed during the operation of the robot or controller:

- Replace faulty components using new components with the same article number or equivalent components approved by Dobot.
- Reactivate any deactivated safety measures immediately after the repairing is completed.
- Record all repairs and save them in the technical document with the robot system.
- Remove the main input cables from the back of the robot to ensure that it is completely unpowered. Take necessary precautions to prevent other persons from powering on the system during the repair period.
- Observe ESD regulations during the disassembly of the parts of the robot.
- Prevent water and dust from entering the robot.

### 5.2 Body Maintenance

In order for the robot to maintain high performance for a long time, a maintenance check must be carried out. The person in charge of overhaul must prepare an overhaul plan and carry out an inspection. The overhaul items are shown below.

Table 5.1 Overhaul item

Cycle			Overhaul Item	Overhaul essential
Daily	3 month	6 month		
√			Robot clean	Wipe off dirt, dust, cutting residue on the body

				with water or 10% alcohol
√			Cable, cable protective cover and air pipe	Observe the moving part of the cable, check whether the cable is damaged, whether there is local bending or distortion; Check whether the cable protective cover is damaged Check whether the air pipe is locally bent, twisted, damaged, etc
	√		Joint bolts	Check the torque based on the specified tightening torque table (Push aside the rubber to check)
√			Tool mounting bolts	Check the torque based on the specified tightening torque table
√			Motor	Abnormal heating or sound confirmation
√			Brake	Check whether the robot arm or end-effector will fall when the servo is powered off
	√		Synchronous belt	Check whether the synchronous belt is worn out, elongated, broken, etc

Table 5.2 lists the bolted tightening torque table.

Table 5.2 Bolted tightening torque table

Bolt size	Hexagon socket head cap screw	Hexagon socket countersunk flat head screw	Cross recessed countersunk flat head screw	Hexagon screw (rubber part)
2.5	-	0.8	0.6	-
3 mm	2.2 Nm	-	-	1.2 Nm
4 mm	4 Nm	-	-	-
5 mm	7.5 Nm	-	-	-

The tightening torques will vary depending on the type of base metal or bolt. When not specified, please contact Dobot technical engineer.

In addition, overhauls are required every 10,000 hours of operation time or every 3 years. If you are not clear about the maintenance processes, please contact Dobot technical engineer.

## Appendix A Servo Alarm Description

ID	Level	Description	Solution
25376	0	Abnormalities in internal servo parameters	System error, please contact technical support engineer
21120	0	Programmable logic configuration faults	System error, please contact technical support engineer
29953	5	FPGA software version too low	Please contact technical support engineer
29954	5	Programmable logic interrupt fault	If connecting the power for many times, the alarm is still reported, please replace the drive
25377	5	Internal program exceptions	System error, please contact technical support engineer
21808	0	Parameter storage failure	Reset the parameter and power on again, or please contact technical support engineer
28962	0	Product matching faults	1. Check whether the motor parameter matches the motor model in nameplate; 2. Check whether the motor and driver match, otherwise, select the right motor and driver
21574	0	Invalid servo ON command fault	System error, please contact technical support engineer
28964	0	Absolute position mode product matching fault	System error, please contact technical support engineer
25378	0	Repeated assignment of DI functions	1. Check whether the same function is assigned to different DI's 2. Confirm whether the corresponding MCU supports the assigned functionality
25379	0	DO function allocation overrun	Check whether the motor and circuit are working properly, or contact technical support engineer
29488	0	Data in the motor encoder ROM is incorrectly checked or parameters are not stored	System error, please contact technical support engineer
8752	0	Hardware overcurrent	System error, please contact technical support engineer
8977	0	DQ axis current overflow fault	System error, please contact technical support engineer

ID	Level	Description	Solution
65288	0	FPGA system sampling operation timeout	System error, please contact technical support engineer
9024	0	Output shorted to ground	Please contact technical support engineer
13184	0	UVW phase sequence error	System error, please contact technical support engineer
33922	0	Flying Cars	Please contact technical support engineer
12816	0	Electrical over-voltage in the main circuit	System error, please contact technical support engineer
12832	0	Main circuit voltage undervoltage	System error, please contact technical support engineer
12592	0	Main circuit electrical shortage	Check the cable connection of power, otherwise, replace the driver
12576	0	Control of electrical undervoltage	System error, please contact technical support engineer
33920	0	Overspeed	System error, please contact technical support engineer
65296	0	Pulse output overspeed	System error, please contact technical support engineer
65282	0	Failure to identify angles	System error, please contact technical support engineer
9040	0	Drive overload	Replace the driver
29056	0	Motor overload	System error, please contact technical support engineer
28961	0	Overheating protection for blocked motors	Check whether the hardware is working properly, or contact technical support engineer
17168	0	Radiator overheating	Drop the environment temperature, or contact technical support engineer
29571	0	Encoder battery failure	Connect battery, or contact technical support engineer
29490	0	Encoder multi-turn count error	Replace the motor
29491	0	Encoder multi-turn count overflow	System error, please contact technical support engineer
29492	0	Encoder interference	System error, please contact technical

ID	Level	Description	Solution
			support engineer
29493	0	External encoder scale failure	System error, please contact technical support engineer
29494	0	Encoder data abnormalities	System error, please contact technical support engineer
29495	0	Encoder return checksum exception	System error, please contact technical support engineer
29496	0	Loss of encoder Z signal	System error, please contact technical support engineer
34321	0	Excessive position deviation	Check whether the motor is working properly, or contact technical support engineer
34322	0	Position command too large	System error, please contact technical support engineer
34323	0	Excessive deviation from fully closed-loop position	System error, please contact technical support engineer
25380	0	Electronic gear setting overrun	System error, please contact technical support engineer
25381	0	Wrong parameter setting for fully closed loop function	System error, please contact technical support engineer
25382	0	Software position upper and lower limits set incorrectly	System error, please contact technical support engineer
25383	0	Wrong home position offset setting	System error, please contact technical support engineer
30083	0	Loss of synchronization	System error, please contact technical support engineer
30081	0	Unburned XML configuration file	Burn the XML configuration file
65298	0	Network initialization failure	System error, please contact technical support engineer
30082	0	Sync cycle configuration error	System error, please contact technical support engineer
30084	0	Excessive synchronisation period error	System error, please contact technical support engineer
25384	0	Fault in crossover pulse output setting	System error, please contact technical support engineer



ID	Level	Description	Solution
65521	0	Zero return timeout fault	System error, please contact technical support engineer
29570	0	Encoder battery warning	Replace battery
21570	0	DI emergency brake	System error, please contact technical support engineer
12851	0	Motor overload warning	System error, please contact technical support engineer
12817	0	Brake resistor overload alarm	System error, please contact technical support engineer
25385	0	External braking resistor too small	System error, please contact technical support engineer
13105	0	Motor power cable disconnection	System error, please contact technical support engineer
25386	0	Change of parameters requires re-powering to take effect	Clear the alarm and power on again
30208	0	Frequent parameter storage	Check whether the upper computer is working normal, or contact technical support engineer
21571	0	Forward overtravel warning	System error, please contact technical support engineer
21572	0	Reverse overtravel warning	System error, please contact technical support engineer
29569	0	Internal failure of the encoder	System error, please contact technical support engineer
12597	0	Input phase failure warning	System error, please contact technical support engineer
65432	0	Zero return mode setting error	System error, please contact technical support engineer
65344	0	Parameter recognition failure	System error, please contact technical support engineer
21121	0	internal error	System error, please contact technical support engineer
29956	0	FPGA configuration error	System error, please contact technical

ID	Level	Description	Solution
			support engineer
51020	0	Driver board identification error	System error, please contact technical support engineer
29568	0	Encoder connection error	Check the cable connection of encoder, or contact technical support engineer
8992	0	Software overcurrent	System error, please contact technical support engineer
9088	0	Current zero point too large	System error, please contact technical support engineer
30080	0	EtherCAT communication failure	System error, please contact technical support engineer
33921	0	Excessive speed tracking error	System error, please contact technical support engineer
21120	0	STO Warning	System error, please contact technical support engineer
21569	0	Upper and lower board connection failure	System error, please contact technical support engineer
8980	0	Busbar overcurrent	System error, please contact technical support engineer
17169	0	Damaged or uninstalled temperature measuring resistors	System error, please contact technical support engineer
29572	0	Encoder Eeprom reading CRC fault	System error, please contact technical support engineer
12928	0	Servo and motor power matching faults	System error, please contact technical support engineer

## Appendix B Controller Alarm Description

ID	Level	Description	Solution
17	5	Inverse kinematics error with no solution	Reselect movement points
18	5	Inverse kinematics error with result out of working area	Reselect movement points
19	5	Duplicated data in JUMP or ARC or Circles instruction	Reselect movement points
20	5	Wrong input parameters for arc	Enter the correct parameters
21	5	The Start and the End is negative or the zLimit is below the start and end points	Enter the correct parameters
22	5	Wrong arm orientation switch	Reselect movement points
23	5	Plan point during linear motion out of working area	Reselect movement points
24	5	Plan point during circular arc motion out of working area	Reselect movement points
25	5	Wrong mode for motion instruction	Internal software error, restart or contact manufacturer
26	5	Wrong input parameters for speed	Input correct parameter
27	5	Wrong trajectory motion plan of continuous path	Input correct parameter
28	0	Wrong input parameters for circle	Input correct parameter
29	5	Plan point during circular circle motion out of working circle	Reselect movement points
30	5	Inching target position inaccessible	Reverse inch out of limit
32	5	Inverse kinematics singularity during moving	Reselect movement points
33	5	Inverse kinematics with no solution during moving	Reselect movement points
34	5	Inverse kinematcis with result out of working area	Reselect movement points
48	5	Joint1 overspeed	Reset the speed or re-select the movement point away from the singularity
49	5	Joint2 overspeed	Reset the speed or re-select the movement point away from the singularity

ID	Level	Description	Solution
50	5	Joint3 overspeed	Reset the speed or re-select the movement point away from the singularity
51	5	Joint4 overspeed	Reset the speed or re-select the movement point away from the singularity
52	0	Joint1 position out of range	Internal error, restart or contact manufacturer
53	0	Joint2 position lag error	Internal error, restart or contact manufacturer
54	0	Joint3 position lag error	Internal error, restart or contact manufacturer
55	0	Joint4 position lag error	Internal error, restart or contact manufacturer
64	5	Joint1 exceeds positive limit	Reverse jog out of limit
65	5	Joint1 exceeds negative limit	Reverse jog out of limit
66	5	Joint2 exceeds positive limit	Reverse jog out of limit
67	5	Joint2 exceeds negative limit	Reverse jog out of limit
68	5	Joint3 exceeds positive limit	Reverse jog out of limit
69	5	Joint3 exceeds negative limit	Reverse jog out of limit
70	5	Joint4 exceeds positive limit	Reverse jog out of limit
71	5	Joint4 exceeds negative limit	Reverse jog out of limit
72	5	Parallelogram positive limit	Reverse jog out of limit
73	5	Parallelogram negative limit	Reverse jog out of limit
74	5	Joint6 exceeds positive limit	Reverse jog out of limit
75	5	Joint6 exceeds negative limit	Reverse jog out of limit
80	0	Joint1 lose step	Internal error, restart or contact manufacturer
81	0	Joint2 lose step	Internal error, restart or contact manufacturer
82	0	Joint3 lose step	Internal error, restart or contact manufacturer
83	0	Joint4 lose step	Internal error, restart or contact manufacturer
84	0	Algorithm timeout	Internal error, restart or contact manufacturer
85	0	Emergency button pressed	Release the emergency stop button
96	0	Joint1 drive alarm	Check if the communication of joint 1 is normal and then clear the error
97	0	Joint1 Servo power off	Re-enable joint 1
98	0	Joint2 drive alarm	Check if the communication of joint 2 is normal and then clear the error

ID	Level	Description	Solution
99	0	Joint2 Servo power off	Re-enable joint 2
100	0	Joint3 drive alarm	Re-enable joint 3
101	0	Joint3 Servo power off	Re-enable joint 3
102	0	Joint4 drive alarm	Re-enable joint 4
103	0	Joint4 drive power off	Re-enable joint 4
104	0	Robot homing failed	Home again
105	0	Robot Servo on failed	Check whether the hardware is normal and re-enable
106	0	Abnormal conveyor data	Please contact technical support engineer
107	0	Abnormal conveyor synchronization	Please contact technical support engineer
108	0	Conveyor conveyor encoder 1 is disconnected	Please contact technical support engineer
109	0	Conveyor conveyor encoder 2 is disconnected	Please contact technical support engineer
110	0	Encoder position error	Internal error, restart or contact manufacturer
112	0	Collision Detection	Keep away from the work area and continue to run
161	0	Error switching drag and drop mode	Internal error, restart or contact manufacturer
4096	5	Failed to open mechanical file	Check if the file location is correct and restart
8192	5	Failed to open project file	Check if the file location is correct and restart
8193	5	Failed to open program file	Check if the file location is correct and restart
8194	5	Failed to open global variable file	Check if the file location is correct and restart
8195	5	Failed to open teaching point file	Check if the file location is correct and restart
8196	5	Failed to start debugger process	Rerun debugger process
12288	5	Emergency stop detected	Power on again
12289	5	External emergency stop detected	Power on again
12290	0	The servo power board temperature is too high	Turn off the machine and let it cool for a period of time
33024	5	No input parameters for CP instruction	Enter the correct parameters
33025	5	Input parameters of CP instruction out of range	Enter the correct parameters

ID	Level	Description	Solution
33280	5	No input parameters for Arch instruction	Please enter parameters
33281	5	Index parameter of Arch instruction out of range	Enter the correct parameters
33282	5	Index parameter of Arch instruction not configured yet	Please set index parameters
33536	5	No input parameters for LimZ instruction	Please enter parameters
33537	5	Input parameters of LimZ instruction out of range	Enter the correct parameters
33792	5	No input parameters for Speed instruction	Please enter parameters
33793	5	Ratio parameter of Speed instruction out of range [1, 100]	Enter the correct parameters
34048	5	No input parameters for Accel instruction	Please enter parameters
34049	5	Ratio parameter of Accel instruction out of range [1, 100]	Enter the correct parameters
34304	5	No input parameters for Jerk instruction	Please enter parameters
34305	5	Ratio parameter of Jerk instruction out of range [1, 100]	Enter the correct parameters
34560	5	No input parameters for SpeedS instruction	Please enter parameters
34561	5	Ratio parameter of SpeedS instruction out of range [1, 100]	Enter the correct parameters
34816	5	No input parameters for SpeedR instruction	Please enter parameters
34817	5	Ratio parameter of SpeedR instruction out of range [1, 100]	Enter the correct parameters
35072	5	No input parameters for AccelS instruction	Please enter parameters
35073	5	Ratio parameter of AccelS instruction out of range [1, 100]	Please enter parameters
35328	5	No input parameters for AccelR instruction	Enter the correct parameters
35329	5	Ratio parameter of AccelR instruction out of range [1, 100]	Enter the correct parameters
35584	5	No input parameters for JerkS instruction	Please enter parameters
35585	5	Ratio parameter of JerkS instruction out of range [1, 100]	Enter the correct parameters

ID	Level	Description	Solution
35840	5	No input parameters for JerkR instruction	Please enter parameters
35841	5	Ratio parameter of JerkR instruction out of range [1, 100]	Enter the correct parameters
36096	5	No input parameters for Go instruction	Please enter parameters
36097	5	No motion point parameter for Go instruction	Please enter parameters
36098	5	Incorrect motion point for Go instruction	Enter the correct parameters
36099	5	Incorrect control parameter for Go instruction	Enter the correct parameters
36352	5	No input parameters for Move instruction	Please enter parameters
36353	5	No motion point parameter for Move instruction	Please enter parameters
36354	5	Incorrect motion point for Move instruction	Enter the correct parameters
36355	5	Incorrect control parameter for Move instruction	Enter the correct parameters
36608	5	No input parameters for Arch3 instruction	Please enter parameters
36609	5	No motion point parameter for Arch3 instruction	Please enter parameters
36610	5	Incorrect motion point for Arch3 instruction	Enter the correct parameters
36611	5	Incorrect control parameter for Arch3 instruction	Enter the correct parameters
36864	5	No input parameters for Jump instruction	Please enter parameters
36865	5	No motion point parameter for Jump instruction	Please enter parameters
36866	5	Incorrect motion point for Jump instruction	Enter the correct parameters
36867	5	Incorrect control parameter for Jump instruction	Enter the correct parameters
40960	5	No input parameters for Circle3 instruction	Please enter parameters
40961	5	No motion point parameter for Circle3 instruction	Please enter parameters

ID	Level	Description	Solution
40962	5	Incorrect motion point for Circle3 instruction	Enter the correct parameters
40963	5	Incorrect control parameter for Circle3 instruction	Enter the correct parameters
45056	5	Circle3 Option Error	Enter the correct parameters
45057	5	Jump Option Error	Enter the correct parameters
45058	5	Arch Option Error	Enter the correct parameters
45059	5	Arch3 Option Error	Enter the correct parameters
45060	5	Jerk Option Error	Enter the correct parameters
45061	5	JerkR Option Error	Enter the correct parameters
45062	5	JerkS Option Error	Enter the correct parameters
45063	5	Accel Option Error	Enter the correct parameters
45064	5	AccelR Option Error	Enter the correct parameters
45065	5	AccelS Option Error	Enter the correct parameters
45066	5	SpeedFactor Option Error	Enter the correct parameters
45067	5	Speed Option Error	Enter the correct parameters
45068	5	SpeedR Option Error	Enter the correct parameters
45069	5	Limz Option Error	Enter the correct parameters
45070	5	CP Option Error	Enter the correct parameters
45071	5	DO Option Error	Enter the correct parameters
45072	5	Go Option Error	Enter the correct parameters
45073	5	Move Option Error	Enter the correct parameters
45074	5	MoveJ Option Error	Enter the correct parameters
45075	5	Ecp Option Error	Enter the correct parameters
45076	5	EcpSet Option Error	Enter the correct parameters
45077	5	SetExitMode Option Error	Enter the correct parameters
32768	5	No input parameters for speedFactor instruction	Enter the correct parameters
32769	5	Input parameters of speedFactor instruction out of range	Enter the correct parameters
32770	5	DO input parameters Error	Enter the correct parameters



ID	Level	Description	Solution
32771	5	DI input parameters Error	Enter the correct parameters
36100	5	No input parameters for movej instruction	Enter the correct parameters
36101	5	No motion point parameter for movej instruction	Enter the correct parameters
36102	5	No motion point parameter for movej instruction	Enter the correct parameters
36103	5	Incorrect motion point for RP instruction	Enter the correct parameters
36104	5	Incorrect offset for RP instruction	Enter the correct parameters
36105	5	Incorrect motion point for RJ instruction	Enter the correct parameters
36106	5	Incorrect offset for RJ instruction	Enter the correct parameters
36107	5	No input parameters for GoR instruction	Enter the correct parameters
36108	5	Incorrect motion point for GoR instruction	Enter the correct parameters
36109	5	No input parameters for MoveJR instruction	Enter the correct parameters
36110	5	Incorrect motion point for MoveJR instruction	Enter the correct parameters
45079	5	loadSwitch Option Error	Enter the correct parameters
45080	5	loadSet Options Error	Enter the correct parameters
45081	5	CPPParamErrorOption	Enter the correct parameters
45082	5	TOOLParamErrorOption	Enter the correct parameters
45083	5	USERParamErrorOption	Enter the correct parameters
45084	5	SPEEDParamErrorOption	Enter the correct parameters
45085	5	SPEEDSPParamErrorOption	Enter the correct parameters
45086	5	ACCELParamErrorOption	Enter the correct parameters
45087	5	ACCELSParamErrorOption	Enter the correct parameters
45088	5	ARCHParamErrorOption	Enter the correct parameters
45089	5	STARTParamErrorOption	Enter the correct parameters
45090	5	ZLIMITParamErrorOption	Enter the correct parameters
45091	5	ENDParamErrorOption	Enter the correct parameters
45092	5	SYNCaramErrorOption	Enter the correct parameters

ID	Level	Description	Solution
45093	5	ARMParmErrorOption	Enter the correct parameters
45312	5	loadSwitch Option Error	Enter the correct parameters
45313	5	loadSet Options Error	Enter the correct parameters
49152	5	Enable remote control when enabled	Enter the correct parameters
36111	5	No input parameters for GoIO instruction	Enter the correct parameters
36112	5	Incorrect motion point for GoIO instruction	Enter the correct parameters
36113	5	Incorrect parameters for GoIO instruction	Enter the correct parameters
36114	5	No input parameters for MoveIO instruction	Enter the correct parameters
36115	5	Incorrect motion point for MoveIO instruction	Enter the correct parameters
36116	5	Incorrect parameters for MoveIO instruction	Enter the correct parameters
36117	5	No input parameters for MoveJIO instruction	Enter the correct parameters
36118	5	Incorrect motion point for MoveJIO instruction	Enter the correct parameters
36119	5	No input parameters for MoveJIO instruction	Enter the correct parameters



**DOBOT**

User Guide

---

# **Dobot MG400**

# **Maintenance Guide**

---

Issue: V1.0

Date: 2022-1-06

Shenzhen Yuejiang Technology Co., Ltd.

**Copyright © Shenzhen Yuejiang Technology Co., Ltd. 2022. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without the prior written consent of Yuejiang Technology Co., Ltd.

### **Disclaimer**

To the maximum extent permitted by applicable law, the products described (including its hardware, software, and firmware, etc.) in this document are provided **AS IS**, which may have flaws, errors or faults. Yuejiang makes no warranties of any kind, express or implied, including but not limited to, merchantability, satisfaction of quality, fitness for a particular purpose and non-infringement of third party rights. In no event will Yuejiang be liable for any special, incidental, consequential or indirect damages resulting from the use of our products and documents.

Before using our product, please thoroughly read and understand the contents of this document and related technical documents that are published online, to ensure that the robot is used on the premise of fully understanding the robot and related knowledge. Please use this document with technical guidance from professionals. Even if follow this document or any other related instructions, damages or losses will be happening in the using process. Dobot shall not be considered as a guarantee regarding all security information contained in this document.

The user has the responsibility to make sure of following the relevant practical laws and regulations of the country, in order that there is no significant danger in the use of the robot.

## **Shenzhen Yuejiang Technology Co., Ltd.**

Address: Address: Floor 9-10, Building 2, Chongwen Garden, Nanshan iPark, Liuxian Blvd,  
Nanshan District, Shenzhen, Guangdong Province, China

Website: [www.dobot.cc](http://www.dobot.cc)

## Preface

### Purpose

This manual aims to help users with execution of operations related to service and troubleshooting.

### Intended Audience

This document is intended for:





- Customer
- Sales Engineer
- Installation and Commissioning Engineer
- Technical Support Engineer

### Change History

Date	Change Description
2022/1/06	The first release

### Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
 DANGER	Indicates a hazard with a high level of risk which, if not avoided, could result in death or serious injury
 WARNING	Indicates a hazard with a medium level or low level of risk which, if not avoided, could result in minor or moderate injury, robot damage
 NOTICE	Indicates a potentially hazardous situation which, if not avoided, can result in equipment damage, data loss, or unanticipated result
 NOTE	Provides additional information to emphasize or supplement important points in the main text

## Contents






<b>1. Safety Precautions.....</b>	<b>1</b>
1.1 Security warning sign .....	1
1.2 General security .....	1
1.3 Personal security .....	3
<b>2. Robot Arm Inspections.....</b>	<b>5</b>
2.1 General cleaning .....	5
2.2 Control cabinet.....	5
2.3 Torque value.....	6
2.4 Functional test.....	6
<b>3. Maintenance .....</b>	<b>7</b>
3.1 Vulnerable part list.....	7
3.2 Recommended tools.....	8
3.3 Preparation for parts return .....	9
3.4 Maintenance of robot arm.....	9
3.4.1 Replacement of J1 primary synchronous belt.....	9
3.4.2 Replacement of J1 secondary synchronous belt .....	11
3.4.3 Replacement of J2 primary synchronous belt.....	16
3.4.4 Replacement of J2 secondary synchronous belt .....	18
3.4.5 Replacement of J3 primary synchronous belt.....	20
3.4.6 Replacement of J3 secondary synchronous belt .....	22
3.4.7 Replacement of battery component .....	25
3.4.8 Replacement of button cells .....	26
<b>4. Calibration of Home Point .....</b>	<b>30</b>

## 1. Safety Precautions

This section describes the security precautions that should be noticed when you use this product. Please read this document carefully before using the robot arm for the first time. This product needs to be used in an environment meeting design specification. You cannot remold the product without authorization, otherwise, it could lead to product failure, and even personal injury, electric shock, fire, etc. People who use this product for system design and manufacture must be trained by our company, relevant institutions, or must have the same professional skills. The installation personnel, operators, teaching personnel, programmers and system developers of the robot arm must read this document carefully and use the robot arm strictly according to the regulations of this document.

### 1.1 Security warning sign

The following safety warning signs may appear in this manual, and their meanings are as follows.

Symbol	Description
 DANGER	Indicates a high degree of potential danger, which, if unavoidable, will result in death or serious injury
 ELECTRICITY	Dangerous power consumption will soon be caused. If it cannot be avoided, it will cause personal injury or serious injury to the equipment.
 HOT	May cause dangerous hot surfaces, if touched, may cause personal injury
 WARNING	Indicates that there is a moderate or low potential hazard. If it cannot be avoided, it may cause minor injuries to the personnel and damage to the equipment.
 NOTICE	Indicates a potential risk, and ignoring these texts may result in damage to the robotic arm, loss of data, or unpredictable results

### 1.2 General security

You need to follow the security rules below when starting or using the robot for the first time.

#### DANGER

- Please be sure to install the robot arm and all electrical equipment according to the requirements and specifications in this guide.
- Preliminary test and inspection of the robot arm and its protection system are required before it is used for the first time and put into production.
- Before starting the system and device for the first time, check whether the device and system are intact, safe to operate, and damaged. You should check whether it complies with national or regional safety regulations, and all safety functions must

be tested.

- You must check and ensure that all security parameters and user programs are correct and that all security features works properly. A person qualified to operate the robot arm is required to check each safety function. Start the machine only after it has passed a thorough and careful safety test and has reached the safety level.
- Professional personnel are required to install and debug the robot arm according to the installation standards.
- Once the robot arm is installed and constructed, you need to carry out a full risk assessment again and maintain the documentation.
- Set and change security parameters by authorized personnel. Use passwords or isolation measures to prevent unauthorized personnel from changing or setting security parameters. After changing security parameters, you need to analyze related security functions.
- In case of accidents or abnormal operation of robot arm, you can press the emergency stop switch to stop the action of the robot arm.
- MG400 joint modules are partly equipped with brakes. Maintain the posture of the robot arm when the power is off. Do not manually switch on or off the power supply system frequently. It is recommended that the interval between switching on and off the power supply system be longer than 10 seconds.
- MG400 robot arm has the function of collision detection. When the external force on the robot arm exceeds the normal force range set by the user, the robot arm will stop automatically to prevent it or the operator from being injured in collision. This function is specially set for the safety of man-machine cooperation, but the robot arm system must be in the normal operation range.

### HOT

- The robot and the control cabinet generate heat during operation. Please do not operate or touch the robot when the robot is working or has just stopped working.
- Turn off the power and wait an hour for the robot to cool down.
- Do not put your fingers where the control cabinet gets hot.

### WARNING

- Please ensure that the robot arm and tools are installed correctly and safely.
- Please ensure that the robot has enough space to move freely.
- If the robot is damaged, please do not use it.
- Do not connect security devices to normal I/O interfaces. Only secure interfaces can be used.
- Ensure proper installation settings (such as arm mounting angle, weight in TCP, TCP offset, security configuration). Save the installation file and load it into the program.



- Ensure that there are not sharp corners or torsion points in tools and obstacles, and that there are not sharp corners or torsion points in tools and obstacles, and that all personnel are out of reach of the robot arm.
- When using the teaching device, pay attention to the motion range of the robot arm.
- Any impact will release a lot of kinetic energy, which is much higher than that under high speed and high load.
- Connecting different devices may aggravate or cause new hazards. You should always conduct a comprehensive risk assessment of the entire installation. When you need to set safety and emergency performance levels, you are advised to select the highest performance level. Be sure to read and understand the manuals for all equipment used in installation.
- Do not alter the robot arm. Changes to the robot arm could cause hazards that the integrator could not predict. The authorized reorganization of the robot arm shall be in accordance with the latest version of the relevant service guide. Shenzhen Yuejiang Technology Co., Ltd. rejects all responsibility if the robot arm is changed or altered in any way.
- Before transporting the robot arm, you need to check the insulation and protection measures.
- Comply with transportation requirements when carrying the robot arm. Handle them carefully to avoid collisions.


 NOTICE

- It is strongly recommended that a separate inspection of all the functions of the robot arm and the arm procedures be performed when the robot arm is connected to or operated with machinery that can cause damage to the robot arm. It is recommended to use temporary waypoints outside other mechanical workspaces to test robot programs
- Do not expose the robot arm to permanent magnetic field all the time. Strong magnetic fields can damage robots.
- Shenzhen Yuejiang Technology Co., Ltd. shall not be liable for any damage or personal injury due to program error or improper operation of the robot arm

### 1.3 Personal security

It is of vital significance to ensure the safety of operators when using the robot arm system. Please strictly follow the general precautions listed below.

 NOTICE

- All operators using the robot arm system shall pass the training course sponsored by Shenzhen Yuejiang Technology Co., LTD., and ensure that they fully master the safe and standard operation process and have the qualification of robot arm operation. For more information, please contact us at [yuejiang@dobot.cc](mailto:yuejiang@dobot.cc). 

[箱为yuejiang@dobot.cc](mailto:yuejiang@dobot.cc)

- When working with robots, please do not wear loose clothing or jewelry. When operating the robot, make sure that you have bundled your long hair behind your head.
- If the robot appears to have stopped during the operation of the equipment, it may be because the robot is waiting for the start signal and is in the state of being about to move. In this case, the robot should also be considered to be in motion. Please do not approach the robot.
- Lines should be drawn on the floor to mark the motion range of the robot arm, so that the operator can understand the motion range of the robot arm with tools (manipulator, tools, etc.).
- Please ensure that safety measures (e.g. guardrails, ropes, or protective screens) have been established near the operation area of robot arm to protect the operator and surrounding people. Locks should be installed as required so that no one other than the operator responsible for the operation can access the power supply of the robot arm.

## 2. Robot Arm Inspections

### 2.1 General cleaning

If you notice dust/dirt/oil on the control cabinet or robot arm, clean the surface of the machine with a dust-free cloth/alcohol and check whether there are defects on the surface.

### 2.2 Control cabinet

In order to maintain high performance of the robot for a long time, maintenance inspection must be carried out. The personnel in charge of maintenance must draw up maintenance plans and carry out maintenance conscientiously. Table 2.1 and Table 2.2 list the recommended inspection periods and items. If the relevant parts are found to be unqualified, please repair or replace them immediately.

Table 2.1 Power-off (without movement) inspection

Item	Content	Period		
		Daily	Once three months	Once six months
Clean the robot body	You can use water and 10% ethanol to wipe any visible dust, dirt and oil on the robot body.	√		
Whether joint screws are loose	Confirm the screw torque at each joint of the robot body according to the screw securing torque table.			√
Whether connector is loose	Robot cable unit and external connectors on the robot.		√	
Brake	Confirm that the robot's 2/3 joint brake functions normally.	√		
Synchronous belt	Remove the shell and check whether the synchronous belt is loose. If it is, you need to re-tension.		√	
Button cell	—	Replace once one year and a half		
Battery component	The upper computer SCS alarms when the encoder battery fails.	Replace once five years		

Table 2.2 Power-on (with movement) inspection

Item	Content	Period		
		Daily	Once three months	Once six months
Working area	Movement area of each joint			√
Movement, sound and vibration of the robot	Check whether there are abnormal movement, sound and vibration in the movement of the robot	√		
Whether connector is loose	Robot cable unit and external connectors on the robot		√	
Brake	Powered on and off to check the sound of the motor brake. If there is no sound, it needs to be replaced	√		

## 2.3 Torque value

Table 2.3 Torque for securing screws

Nominal diameter of screw thread	Countersunk head hexagon socket screw	Hexagon socket button head screws	Hexagon socket cap screws
2.5 mm	0.3 Nm	0.3 Nm	0.5 Nm
3 mm	0.5 Nm	4.5 Nm	2 Nm
4 mm	-	2 Nm	-
5 mm	-	-	7.5 Nm

The torque for securing screws varies according to screw types or base materials. If it is not recorded in the table, please contact the after-sales service department of Yuejiang Technology Co., Ltd.

In addition, equipment must be overhauled after every 10,000 hours of operation. If you have any questions about the maintenance and adjustment method, please contact the after-sale service department of Yuejiang Technology Co., Ltd.

## 2.4 Functional test

The purpose of the functional test is to ensure that screws, bolts, tools and robot arm are not loose. The screws/bolts mentioned in the inspection plan shall be checked with a torque wrench and the torque shall comply with the specification.

### 3. Maintenance

The purpose of maintenance is to ensure the normal operation of the system or restore the system to normal operation in case of a fault. Maintenance includes fault diagnosis and actual repairs.

Maintenance must be performed by a system integrator authorized by Yuejiang Technology Co., Ltd. or after sales staff of yuejiang technology Co., Ltd.

#### 3.1 Vulnerable part list

Table 3.1 Vulnerable part list

No.	Material	Specification	Number	Material code
1	synchronous belt	Ceptor-VI 80S3M303	2	2299059300
2	synchronous belt	Ceptor-VI 80S3M267	1	2299059400
3	synchronous belt	Ceptor-VI 80S3M291	1	2299059500
4	synchronous belt	Ceptor-VI 80S3M249	1	2299059600
5	synchronous belt	Ceptor-VI 80S3M243	1	2299059700
6	button cell	Panasonic, CR2032 button cell, 3.0V	1	2108000800
7	battery components	two LS14500 lithium batteries in parallel, Jianya terminal: A1250H-2P, L=80mm, V3.0	1	2108003100
8	ribbon	2.5mm*150mm, black	6	2604023200
9	aviation plug	ELECTRIC, aviation plug, Chrome plated copper, silver, six core, male pin, 3A/30V	1	2109025500
10	wiring terminals	CONNECTOR, wiring terminals, 3.5mm interval, 10Pin, KF12EKNM-3.50-10P-1K, single interval, black	4	2106058800
11	power adapter	Module, power adapter, input: 100-240VAC 50/60Hz, output: 48VDC, 240W	1	2004031200
12	power adapter	Module, power adapter, input: 85~264VAC/47Hz~63Hz, output: 48V, 240W	1	2004031000
13	M.Pro emergency stop components	wire, M.Pro emergency stop components, LAS1-AY-22TSB/R+S10+HB-BX1/16 + four-core flexible cable + four-pin connector, 1500mm, V3.0	1	2005059501
14	M.Pro servo drive board PCBA	PCBA, M.Pro, servo drive board PCBA, four-layer board, V2.0, using 2002021103	1	2001018403
15	M.Pro servo drive board PCBA	PCBA, M.Pro, servo drive board PCBA, six-layer board, V1.0, using 2002021002	1	2001018502
16	M.Pro main control panel PCBA	PCBA, M.Pro, main control panel PCBA, V1.0, using 2002021802	1	2001018102
17	M.Pro indicator board PCBA	PCBA, M.Pro, indicator board PCBA, two-layer board, V1.0, using PCB material code 2002022002	1	2001018302

18	M.Pro wire adapter plate PCBA	PCBA, M.Pro wire adapter plate PCBA, four-layer board, V1.0, screen 2002023201	1	2001019901
----	----------------------------------	--	---	------------

### 3.2 Recommended tools

When installing and maintaining a robot arm, you need to prepare the following tools (anti-static tools):

- Cross head screwdriver -PH0, PH1



- Hex socket screwdriver and screw bit: M2.5, M3, M4, M5



- Socket screwdriver: 5.5mm (diagonal size), 6mm (diagonal size)



- ESD wrist strap



- Sound wave tensiometer



### 3.3 Preparation for parts return

- Remove all non-Dobot equipment such as clamps, hoses, cables etc. from the robot arm. Yuejiang technology Co., Ltd. will not be responsible for any damage to non-Dobot equipment installed on the robot arm.
- Back up all relevant documents before sending the arm/part to Yuejiang Technology Co., Ltd. Yuejiang Technology Co., Ltd. is not responsible for the loss of programs, data or files stored in the robot arm.
- The robot arm should restore to the factory posture before returning. Factory posture can reduce the space occupied by the robot arm, which is convenient for packaging and transportation.

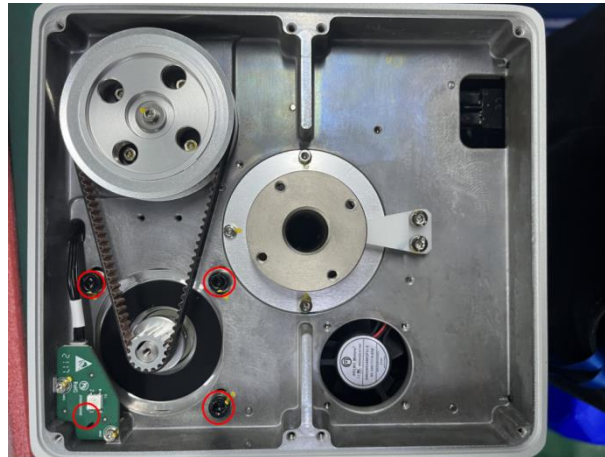
### 3.4 Maintenance of robot arm

#### 3.4.1 Replacement of J1 primary synchronous belt

- Step 1** Remove the top cover of the base: Remove four M3\*8 socket countersunk screws from the base cover using a hex wrench.

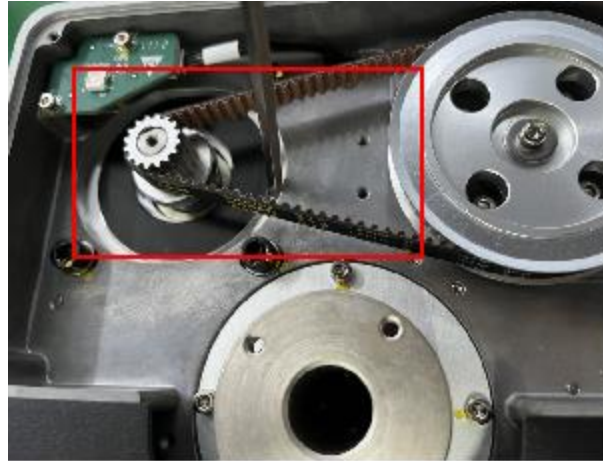


- Step 2** Loosen screws of the J1 motor: Loosen four M5\*16 hexagon socket cap screws of the J1 motor using a hex wrench to move the motor back and forth in the screw slot.



- Step 3** Install J1 primary synchronous belt: Install the primary synchronous belt (80S3M303) to the two synchronous pulleys, and push the motor to tension the synchronous belt. Lock the four screws on the motor using a torque wrench (torque: 7.5 Nm), Place the test probe of sound wave tensiometer (parameter: M:2.3, W:8, S: 89) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results, requiring that the test results are within the range of 50~60Hz.





- Step 4** Install the top cover of the base. Secure four M3\*8 hex socket countersunk screws on the top cover of the base using a torque wrench with a torque of 0.5 Nm.



- Step 5** Calibrate the home point.

### 3.4.2 Replacement of J1 secondary synchronous belt

- Step 1** Remove the top cover of the base: Remove four M3\*8 hex socket countersunk screws from the base top cover-1 using a hex wrench.



- Step 2** Loosen the screws of the J1 motor: Loosen four M5\*16 hexagon socket cap screws for the J1 motor using a hex wrench to move the motor back and forth in the screw slot.



- Step 3** Loosen the secondary reduction mechanism. Remove two M2.5\*5 button head screws that secure the LED plate. Remove the LED plate, and loosen four M3 hexagon socket cap screws.



- Step 4** Remove the base cover of the robot: Put the robot upside down, and remove the anti-removal label of the base. Remove four M3\*8 hex socket countersunk screws from

the base cover, and remove the rear cover.



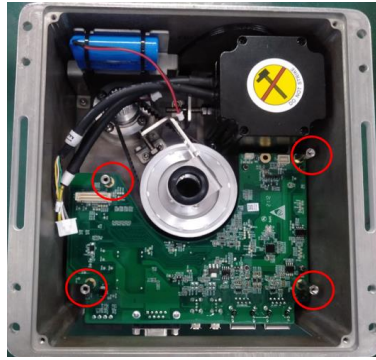
- Step 5** Remove the servo drive board PCBA. Remove the plug terminals on the PCBA. Remove four M3\*6 hexagon socket cap screws using a hex wrench, and place the PCBA in an ESD bag.



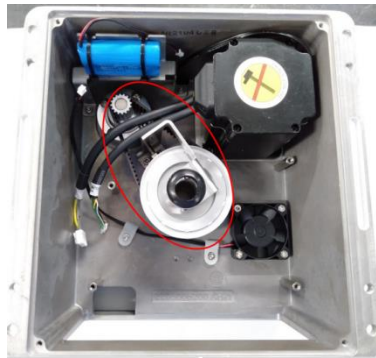
- Step 6** Remove the PCBA on the servo control board, the plug terminals on the PCBA, and four M3\*16 single-head hexagonal studs using a socket wrench. Remove the rocker switch connectors, and place the PCBA in an ESD bag.



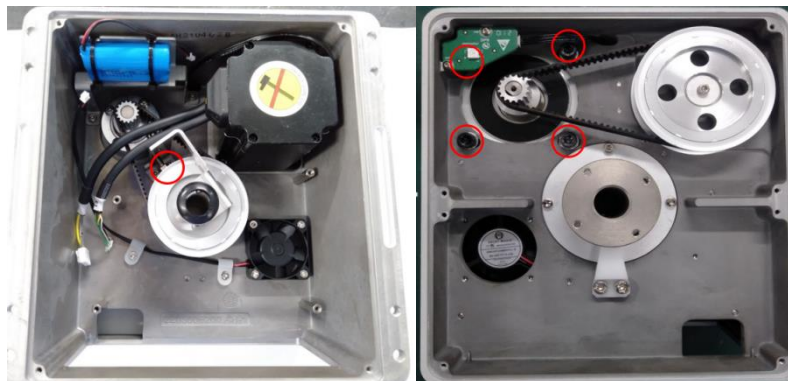
- Step 7** Remove the PCBA main control board, remove the plug terminals on the PCBA, remove the four M3\*20 single-head hexagonal studs using a socket wrench, remove the rocker switch connectors, and place the PCBA in an ESD bag.



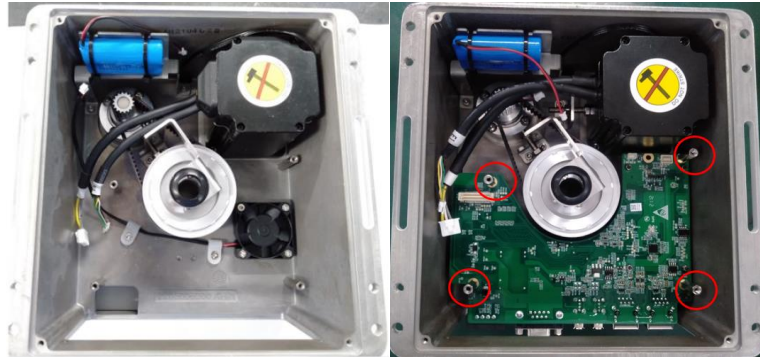
- Step 8** Remove the old synchronous belt, loosen the hex screws, and remove the synchronous pulley.



- Step 9** Install a new synchronous belt: Install the 120S3M267 synchronous belt on the two synchronous pulleys. Tighten the outer hexagonal bolts with an outer hexagonal wrench, and tighten the synchronous belt. Secure four M3\*10 hexagon socket cap screws at the secondary deceleration place using a torque wrench (torque: 2.0N.m). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 12, S: 74) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 85~96Hz.



- Step 10** Install the main control panel PCBA: Secure the main control panel PCBA using four M3\*20 single-head hexagonal studs with a torque of 0.6 Nm.



- Step 11** Install the servo control panel PCBA: Connect the rocker switch wire to the servo control panel PCBA, and then fix the PCBA with four M3\*16 single-head hexagonal studs with a torque value of 0.6Nm. Connect each terminal according to the position in the figure below.



- Step 12** Install the servo drive board PCBA: Connect the rocker switch wire to the servo drive board PCBA, and then fix the PCBA using four M3\*6 inner hexagon button head screws with a torque of 0.6 Nm. Connect each terminal according to the position in the figure. Fix the wire harness with a cable tie.



- Step 13** Install the base cover of the robot: Fix the base cover with four M3\*8 hex socket countersunk screws.

- Step 14** Install the LED adapter plate. Fix the plate using two M2.5\*5 hexagon socket cap screws, with a torque of 0.3Nm.

- Step 15** Install J1 primary synchronous belt: Mount the primary synchronous belt which was removed before to the two synchronous pulleys, and push the motor to tension the synchronous belt. Secure the four screws on the motor using a torque wrench (torque: 7.5 Nm). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 8, S: 89) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.
- Step 16** Install the top cover of the base: Secure four M3\*8 hex socket countersunk screws on the base top cover-1 using a torque wrench with a torque of 0.5 Nm.



- Step 17** Calibrate the home point.

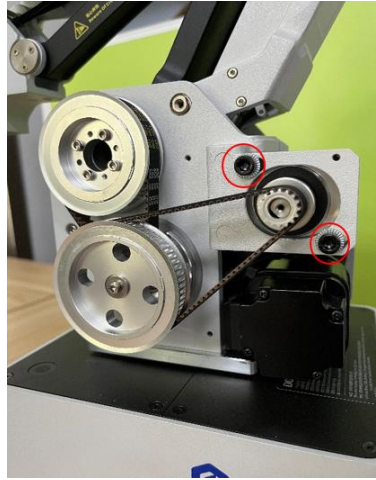
### 3.4.3 Replacement of J2 primary synchronous belt

- Step 1** Disassemble the motor casing (left): Remove four M3\*8 hexagon socket cap screws from the base top cover-1 using a hex wrench.



- Step 2** Loosen the screws of the J2 motor: Loosen two M5\*16 hexagon socket cap screws

of the J2 motor using a hex wrench to move the motor back and forth in the screw slot.



**Step 3** Disassemble J2 primary synchronous belt.



**Step 4** Install a new J2 primary synchronous belt: Mount the primary synchronous belt (80S3M291) to the two synchronous pulleys, and push the motor to tension the synchronous belt. Lock the two screws on the motor using a torque wrench (torque: 7.5 Nm). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 8, S: 84) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.



- Step 5** Disassemble the motor casing (left): Fix the motor casing (left) with three M3\*8 hexagon socket cap screws (torque: 1.0 Nm).



- Step 6** Calibrate the home point.

#### 3.4.4 Replacement of J2 secondary synchronous belt

- Step 1** Disassemble the motor casing (left): Remove four M3\*8 hexagon socket cap screws from the base top cover-1 using a hex wrench.



- Step 2** Loosen the screws of the J2 motor: Loosen two M5\*16 hexagon socket cap screws of the J2 motor using a hex wrench to move the motor back and forth in the screw slot.





**Step 3** the secondary deceleration mechanism Loosen four M3\*10 screws using a hex wrench.

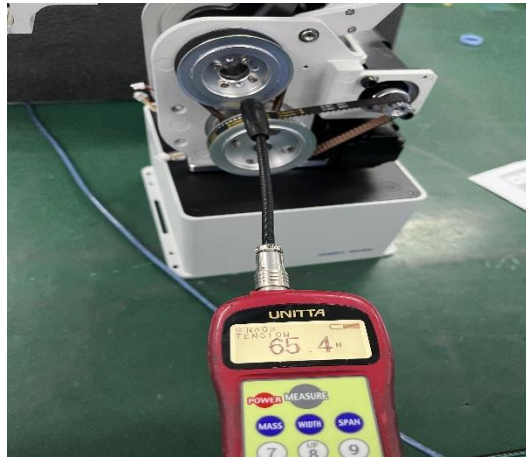


**Step 4** Disassemble the J2 secondary synchronous belt.



**Step 5** Install a new synchronous belt: Install the 100S3M249 synchronous belt on the two synchronous pulleys. Tighten the synchronous belt, and secure four M3\*10 hexagon socket cap screws at the secondary deceleration place using a torque wrench (torque: 2.0N.m). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 10, S: 65) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.

- Step 6** Install a new J2 primary synchronous belt: Mount the primary synchronous belt (80S3M291) to the two synchronous pulleys, and push the motor to tension the synchronous belt. Lock the two screws on the motor using a torque wrench (torque: 7.5 Nm). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 8, S: 84) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.



- Step 7** Install the motor casing (left): Fix the motor casing (left) with three M3\*8 hexagon socket cap screws (torque: 1.0 Nm).



- Step 8** Calibrate the home point.

### 3.4.5 Replacement of J3 primary synchronous belt

- Step 1** Disassemble the motor casing (right): Remove four M3\*8 hexagon socket cap screws from the motor casing (right) using a hex wrench.



- Step 2** Loosen the screws of the J3 motor: Loosen two M5\*16 hexagon socket cap screws of the J3 motor using a hex wrench to move the motor back and forth in the screw slot.

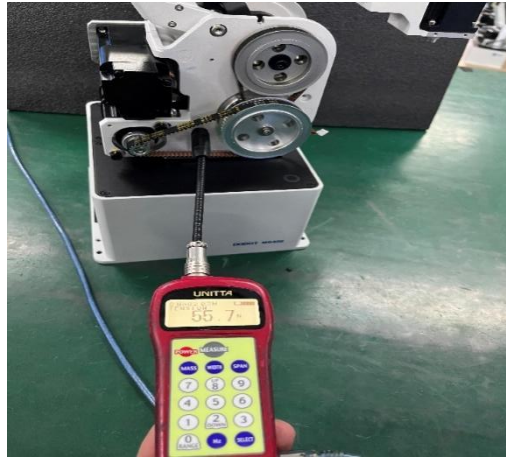


- Step 3** Disassemble J3 primary synchronous belt.



- Step 4** Install a new J3 primary synchronous belt: Install the primary synchronous belt (80S3M303) to the two synchronous pulleys, and push the motor to tension the

synchronous belt. Lock the two screws on the motor using a torque wrench (torque: 7.5 Nm). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 8, S: 88) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.



**Step 5** Install the motor casing (left): Fix the motor casing (left) with three M3\*8 hexagon socket cap screws (torque: 1.0 Nm).



**Step 6** Calibrate the home point.

### 3.4.6 Replacement of J3 secondary synchronous belt

**Step 1** Disassemble the motor casing (left): Remove three M3\*8 hexagon socket cap screws from the motor casing (left) using a hex wrench.



**Step 2** Loosen the screws of the J3 motor: Loosen two M5\*16 hexagon socket cap screws of the J1 motor using a hex wrench to move the motor back and forth in the screw slot.



**Step 3** Loosen the secondary deceleration mechanism. Loosen four M3\*10 screws using a hex wrench.



**Step 4** Disassemble the J3 secondary synchronous belt.



- Step 5** Install a new synchronous belt: Install the 80S3M243 synchronous belt on the two synchronous pulleys. Tighten the synchronous belt, and secure four M3\*10 hexagon socket cap screws at the secondary deceleration place using a torque wrench (torque: 2.0N.m). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 8, S: 60) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.



- Step 6** Install J3 primary synchronous belt: Install the primary synchronous belt (80S3M303) to the two synchronous pulleys, and push the motor to tension the synchronous belt. Lock the two screws on the motor using a torque wrench (torque: 7.5 Nm). Place the test probe of sound wave tensiometer (parameter: M: 2.3, W: 8, S: 88) in the middle part of the synchronous belt, about 10mm away from the synchronous belt. Move the middle part of the synchronous belt by hand to read the test results. The test results should be within the range of 50~60Hz.



**Step 7** Install the motor casing (left): Fix the motor casing (left) with three M3\*8 hexagon socket cap screws (torque: 1.0 Nm).



**Step 8** Calibrate the home point.

### 3.4.7 Replacement of battery component

**Step 1** Remove the base cover of the robot: Put the robot upside down, and remove the anti-removal label of the base. Remove four M3\*8 hex socket countersunk screws from the base cover, and remove the rear cover.



- Step 2** Disassemble the battery components: Unplug the battery cable terminal. Remove two M3\*8 hexagon socket cap screws using a hex wrench. Cut off the cable ties using scissors.



- Step 3** Replace the new battery components: Bind the new battery component to the sheet metal with cable ties, and secure the two M3\*8 hexagon socket head cap screws with a torque wrench.
- Step 4** Calibrate the home point.

### 3.4.8 Replacement of button cells

- Step 1** Remove the base cover of the robot: Put the robot upside down, and remove the anti-removal label of the base. Remove four M3\*8 hex socket countersunk screws from the base cover, and remove the rear cover.



- Step 2** Disassemble the servo drive board PCBA: Remove the plug terminals on the PCBA. Remove four M3\*6 hexagon socket button head screws using a hex wrench, and place the PCBA in an ESD bag.





- Step 3** Install the servo control panel PCBA: Remove the plug terminals on the PCBA. Remove four M3\*16 single-head hexagonal studs using a socket wrench. Remove the rocker switch connectors, and place the PCBA in an ESD bag.



- Step 4** Disassemble the main control panel PCBA: Remove the plug terminals on the PCBA, and four M3\*20 single-head hexagonal studs using a socket wrench. Remove the rocker switch connectors, and place the PCBA in an ESD bag.

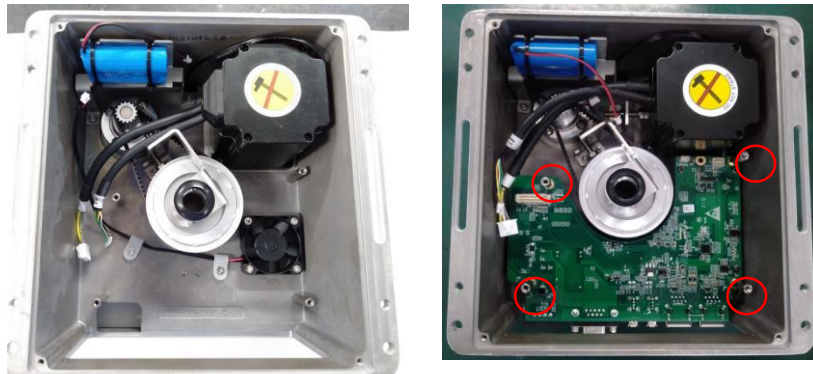


- Step 5** Replace the button cells: Replace the button cells on the main control panel with new

cells.



- Step 6** Install the main control panel PCBA: Fix the main control panel PCBA using four M3\*20 single-head hexagonal studs with a torque of 0.6 Nm.



- Step 7** Install the servo control panel PCBA: Connect the rocker switch wire to the servo control panel PCBA, and then fix the PCBA with four M3\*16 single-head hexagonal studs with a torque value of 0.6Nm. Connect each terminal according to the position in the figure below.



- Step 8** Install the servo drive board PCBA: Connect the rocker switch wire to the servo drive board PCBA, and then fix the PCBA using four M3\*6 hexagon socket button head screws with a torque of 0.6 Nm. Connect each terminal according to the position in

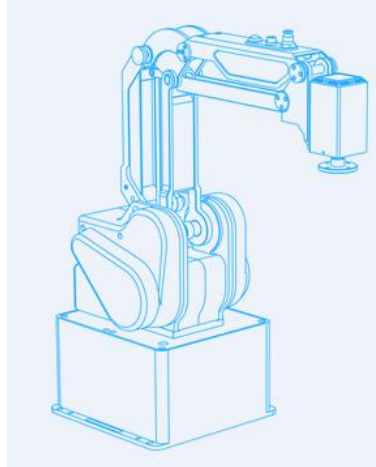
the figure. Fix the wire harness with a ribbon.



**Step 9** Install the base cover of the robot: Fix the base cover with four M3\*8 hex socket countersunk screws.

## 4. Calibration of Home Point

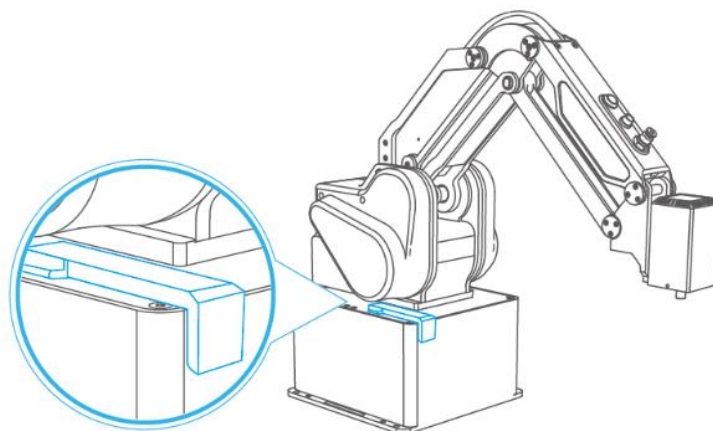
When the synchronous belt, button cells and battery components are replaced, the home point of the robot arm will change. You need to reset the robot arm. The home point of MG400 is shown in the figure below.



- Step 1** Use the calibration block to move MG400 to the vertical state of upper arm and forearm, that is, adjust each axis to the mechanical zero point.

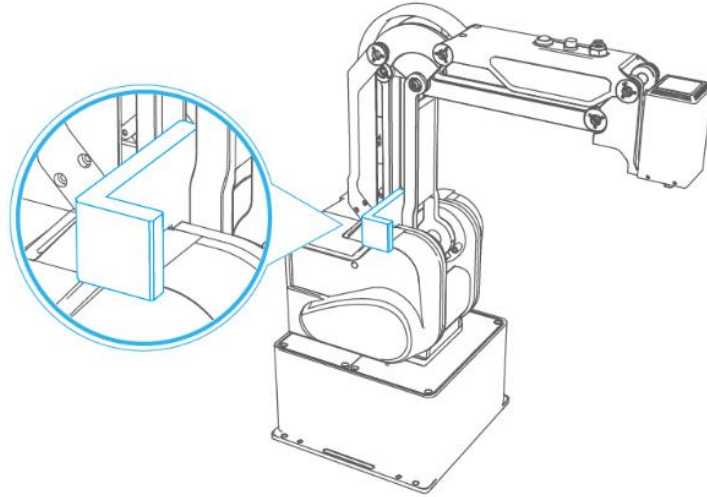


- 1) Place the calibration block in the position shown below and close to the rotating disc. Rotate J1 axis to make the rotating disc parallel and close to the calibration block.

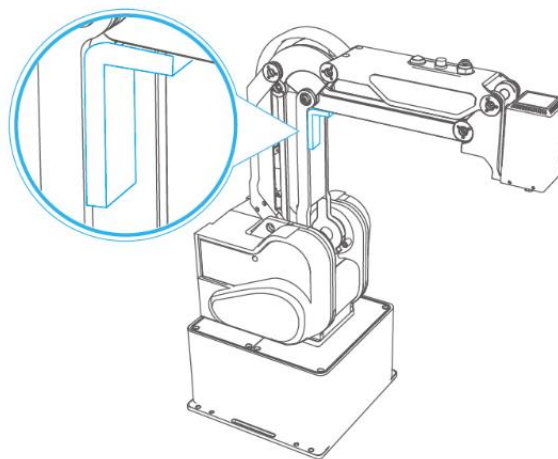


- 2) Clamp the convex groove at the bottom of the calibration block in the gap shown

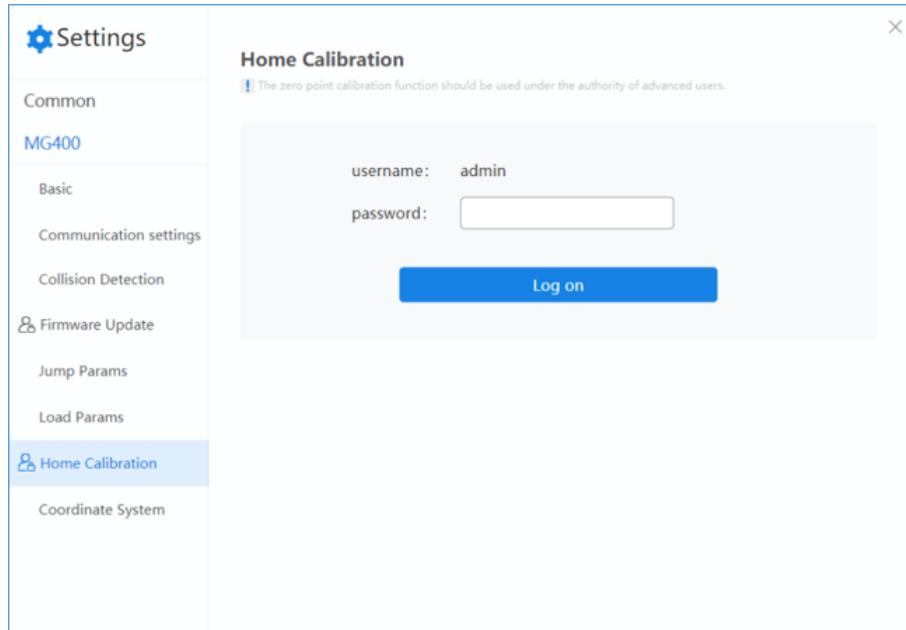
in the figure below, and make the short side of the calibration block face the upper arm. Press the teaching button, and drag J2 axis and J3 axis to make the upper arm parallel and close to the calibration block. At the same time make the angle between the upper arm and the forearm greater than 90°.



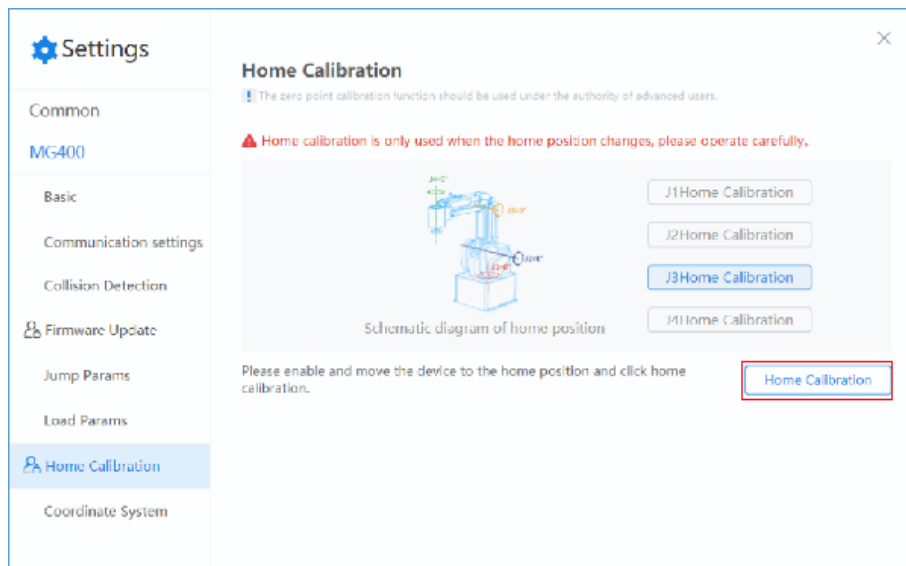
- 3) Put the calibration block in the position shown in the figure below, that is, the angle between the upper arm and the forearm, and make the long side of the calibration block parallel and close to the upper arm. By moving the J3 axis on the control panel, make the forearm parallel and close to the short edge of the calibration block.



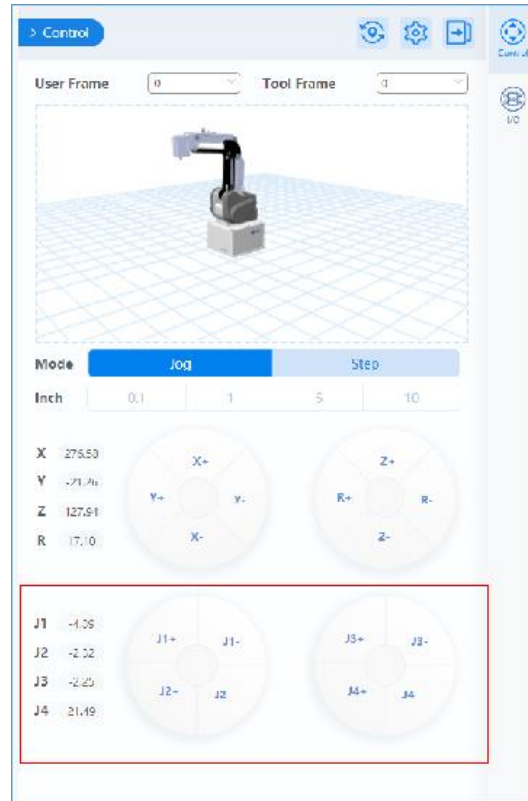
**Step 2** Click **Settings > Home Calibration**. Enter the password (default password: 888888), and click **Log on**.



**Step 3** Click **Home Calibration** when MG400 is enabled.



If the calibration is successful, you can view the joint coordinates in control panel.  
Now the values of J1~J4 are zero.

 NOTICE

Home calibration is used only when the home position changes. Please operate cautiously.



**DOBOT**

User Guide

---

# **DobotSCStudio**

## **User Guide**

### **(MG400 & M1 Pro)**

---

Issue: V2.1.8

Date: 2021-10-19

Shenzhen Yuejiang Technology Co., Ltd



**Copyright © Shenzhen Yuejiang Technology Co., Ltd 2021. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without the prior written consent of Yuejiang Technology Co., Ltd

### **Disclaimer**

To the maximum extent permitted by applicable law, the products described (including its hardware, software, and firmware, etc.) in this document are provided **AS IS**, which may have flaws, errors or faults. Yuejiang makes no warranties of any kind, express or implied, including but not limited to, merchantability, satisfaction of quality, fitness for a particular purpose and non-infringement of third party rights. In no event will Yuejiang be liable for any special, incidental, consequential or indirect damages resulting from the use of our products and documents.

Before using our product, please thoroughly read and understand the contents of this document and related technical documents that are published online, to ensure that the robot is used on the premise of fully understanding the robot and related knowledge. Please use this document with technical guidance from professionals. Even if follow this document or any other related instructions, damages or losses will be happening in the using process. Dobot shall not be considered as a guarantee regarding all security information contained in this document.

The user has the responsibility to make sure of following the relevant practical laws and regulations of the country, in order that there is no significant danger in the use of the robot.

## **Shenzhen Yuejiang Technology Co., Ltd.**

Address: Address: Floor 9-10, Building 2, Chongwen Garden, Nanshan iPark, Liuxian Blvd,  
Nanshan District, Shenzhen, Guangdong Province, China

Website: [www.dobot.cc](http://www.dobot.cc)

## Preface

### Purpose

This manual introduces the functions and usage of the robot control software DobotSCStudio, which is convenient for users to understand and use MG400.

### Intended Audience

This document is intended for:





- Customer
- Sales Engineer
- Installation and Commissioning Engineer
- Technical Support Engineer

### Change History

Date	Change Description
2021/10/19	Update the UI interface; add Blockly programming and safesetting functions; update programming commands; delete calibration, drag and brake function
2021/04/29	The first release

### Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
 DANGER	Indicates a hazard with a high level of risk which, if not avoided, could result in death or serious injury
 WARNING	Indicates a hazard with a medium level or low level of risk which, if not avoided, could result in minor or moderate injury, robot damage
 NOTICE	Indicates a potentially hazardous situation which, if not avoided, can result in equipment damage, data loss, or unanticipated result
 NOTE	Provides additional information to emphasize or supplement important points in the main text

## Contents

<b>1. Overview.....</b>	<b>5</b>
1.1 Description on Main Interface.....	5
<b>2. Fast Connection.....</b>	<b>7</b>
<b>3. Function Description.....</b>	<b>10</b>
3.1 Enabling.....	10
3.2 Setting Global Velocity Rate.....	10
3.3 Alarm Description.....	11
3.4 Jogging.....	11
3.5 Blockly.....	13
3.6 Programming.....	16
3.6.1 Project Description.....	16
3.6.2 Programming Interface Description.....	16
3.6.3 Programming Description.....	18
3.7 Parameter.....	28
3.7.1 Setting User Coordinate System.....	28
3.7.2 Setting Tool Coordinate System.....	33
3.7.3 I/O Monitor.....	36
3.7.4 Controller Setting.....	37
3.7.5 Remote Control.....	39
3.7.6 RobotParams.....	42
3.7.7 RobotSetting.....	47
3.7.8 SafeSetting.....	48
3.8 ToolConfig.....	50
3.8.1 BasicConfig.....	50
3.8.2 PluginsInfo.....	50
3.8.3 Log.....	50
3.8.4 Network Service.....	51
3.8.5 Tools.....	52
3.8.6 VirtualRobot.....	52
3.8.7 WiFi Setting.....	52
<b>4. Program Language.....</b>	<b>54</b>
4.1 Arithmetic Operators.....	54
4.2 Relational Operator.....	54
4.3 Logical Operators.....	54
4.4 General Keywords.....	55
4.5 General Symbol.....	55
4.6 Processing Control Commands.....	55
4.7 Global Variable.....	55
4.8 Motion Commands.....	56
4.9 Motion Parameter Commands.....	62
4.10 Input/output Commands.....	64
4.11 Program Managing Commands.....	65

4.12 Pose Getting Command.....	67
4.13 TCP.....	69
4.14 UDP.....	71
4.15 Modbus.....	72
4.15.1 Description on Modbus Register.....	72
4.15.2 Command Description.....	74
4.16 Conveyor Tracking.....	76
4.17 Pallet.....	78
<b>Appendix A Servo Alarm Description.....</b>	<b>83</b>
<b>Appendix B Controller Alarm Description.....</b>	<b>88</b>

# 1. Overview

DobotSCStudio is an industrial robot programming platform launched by Yuejiang, which is suitable for the whole series of industrial robots (MG400/SA/SR/CR/M1 Pro). With friendly interface, it supports secondary development by users. It also provides kinematics algorithm of various mechanical structures and integrated virtual simulation environment to realize rapid deployment of various process applications on site.

## 1.1 Description on Main Interface

Figure 1.1 shows the main interface of DobotSCStudio. Table 1.1 lists the interface description.

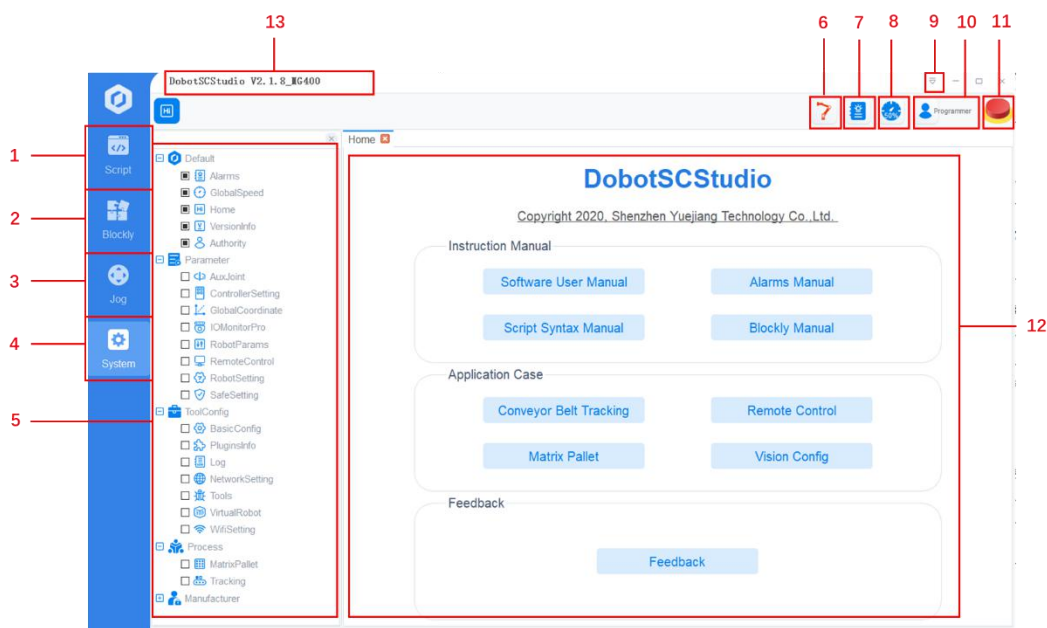


Figure 1.1 Main interface

Table 1.1 Interface description

No.	Description
1	Script You can build or import a project, and debug or run it
2	Blockly You can write programs by graphical language to quickly and conveniently control the robot
3	Jog Jog the robot in different coordinate systems. Jog the robot in the Joint coordinate system: From top to bottom, jog J1, J2, J3 and J4

No.	Description
	Jog the robot in the Cartesian coordinate system: From top to bottom, jog the X, Y, Z, R
4	<p>System</p> <p>You can set system configurations, such as NetworkSetting, RobotParams, Coordinate, Process, etc.</p>
5	System bar
6	Click this button to enable or disable the motor
7	<p>Check robot alarms</p> <p>When an alarm is triggered, this icon will flash red</p> <p>You can check the alarm details on the operation panel and clear it</p>
8	Set global velocity rate
9	<p>IP setting or check update</p> <p>After connecting robot and PC with network cable, you need to select <b>Real</b> on the IP Settings page and select robot's IP address for connecting to DobotSCStudio</p>
10	<p>Select user mode</p> <ul style="list-style-type: none"> <li>• Watcher: Check the system status, I/O status, robot pose, and alarms</li> <li>• Operator: Operate a robot based on the existing scripts without programming</li> <li>• Programmer: On the basis of operator authority, you can program and teach</li> <li>• Manager: On the basis of programmer authority, you can set or modify parameters</li> </ul> <p>Please select user mode based on site requirements</p> <p>Default password: admin. You can modify the password on the <b>ToolConfig &gt; BasicConfig &gt; UserMode</b> page in the Manager mode</p>
11	<p>Emergency stop switch</p> <p>Press and hold it in an emergency, and the drive power supply of MG400 will be powered off for emergency braking</p>
12	Interactive window
12	<p>Show the current running mode</p> <p>Running mode: I/O, Modbus, SCStudio (SCStudio mode is not displayed)</p>

## 2. Fast Connection

DobotSCStudio can communicate with MG400 directly through Ethernet1. At this point, the IP address of MG400 should be in the same network segment as that of the PC. The default IP address of the robot is 192.168.1.6 and cannot be modified. Please modify the IP address of PC to make them in the same network segment.

### NOTE

- Minimum computer configuration for installing DobotSCStudio:  
 System: Windows7 64-bit/Windows10 32/64 bit  
 Memory: 4GB or above  
 CPU: Intel i3 or above
- This section uses Windows7 OS as an example to describe how to change the IP address. Please change it based on site requirements.

### Procedure

**Step 1** Connect power adapter to robot **Power Switch** interface.

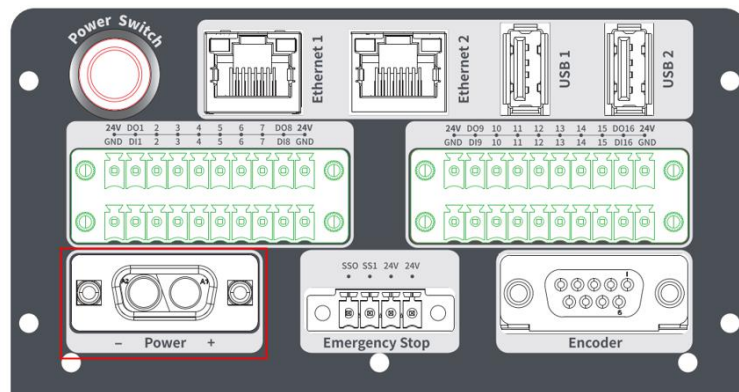


Figure 2.1 Connect to robot Power Switch interface

**Step 2** Connect emergency stop switch to **Emergency Stop** interface.

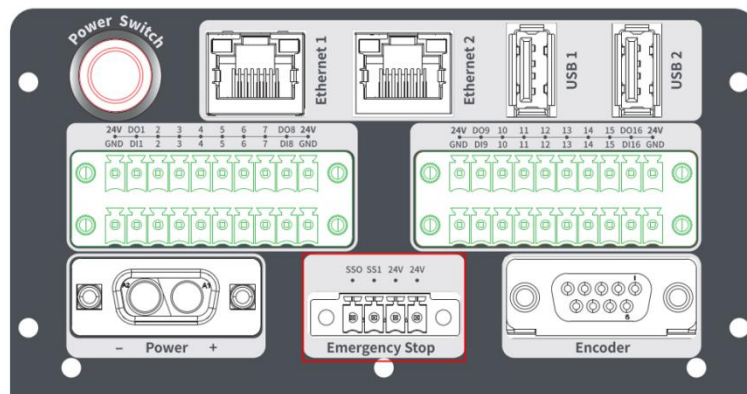


Figure 2.2 Connect to Emergency Stop interface

- Step 3** Connect one end of the network cable to the **Ethernet1** interface on the robot and the other end to the PC.
- Step 4** Click **Start > Control Panel** on the PC and select **Network and Sharing Centre**. The **Network and Sharing Centre** page is displayed.
- Step 5** Click **Local Area Connection** on the **Network and Sharing Center** page.
- Step 6** Click **Properties**.
- Step 7** Double-click Internet Protocol Version 4(TCP/IPv4).
- Step 8** Select **Use the following IP address**, and change the IP address, subnet mask, and gateway of the PC.

You can change the IP address of the PC to make it on the same network segment as that of the robot without conflict. The subnet mask and gateway of the PC must be the same as that of robot. For example, the computer IP address is 192.168.1.40, and the default gateway is 255.255.255.0.

### NOTICE

If the PC is connected to robot over a network cable directly, you only need to set the IP address and subnet mask of the PC.

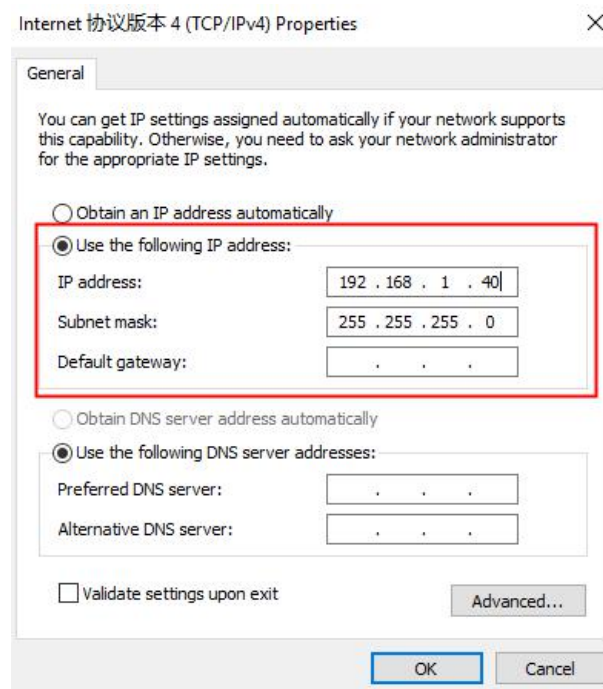



Figure 2.3 IP address modification

- Step 9** Click **OK**.
- Step 10** Click  **> IP settings...** on the upper right pane of the DobotSCStudio page and select the robot's IP address, then click **OK**.



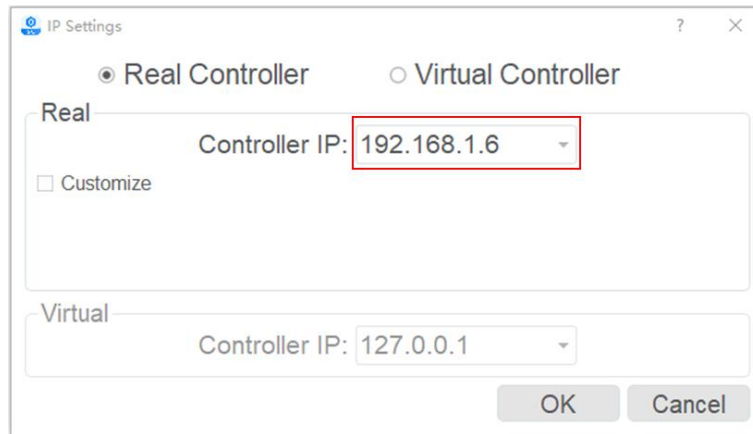


Figure 2.4 IP setting

After the connection is successful, the DobotSCStudio will be shown as below.

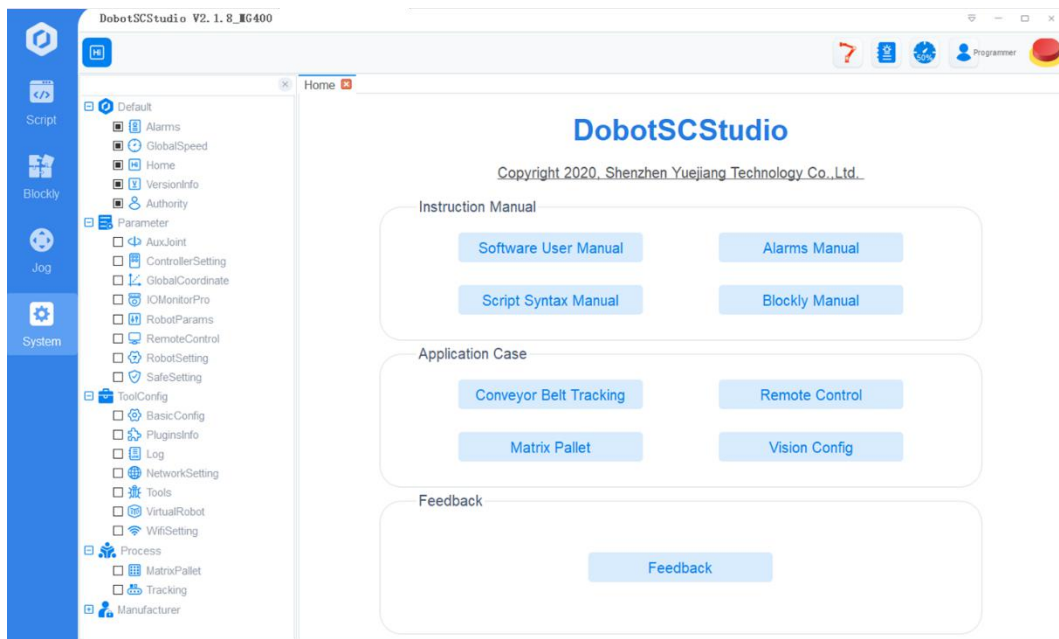







Figure 2.5 Connected successfully

### 3. Function Description

#### 3.1 Enabling

Click  to enable MG400. When the icon  turns into , MG400 can be controlled by running the program or Jogging.

#### 3.2 Setting Global Velocity Rate

Please click  and then click buttons on the operation panel to increase or decrease the global velocity ratio by 1%, 5%, 10%, 25% and 50%, as shown in Figure 3.1. You can also change the velocity in  > **Default** > **GlobalSpeed**. The Global Velocity Rate is not modified when the program is running. It can only be done when the program is not running or is suspended.

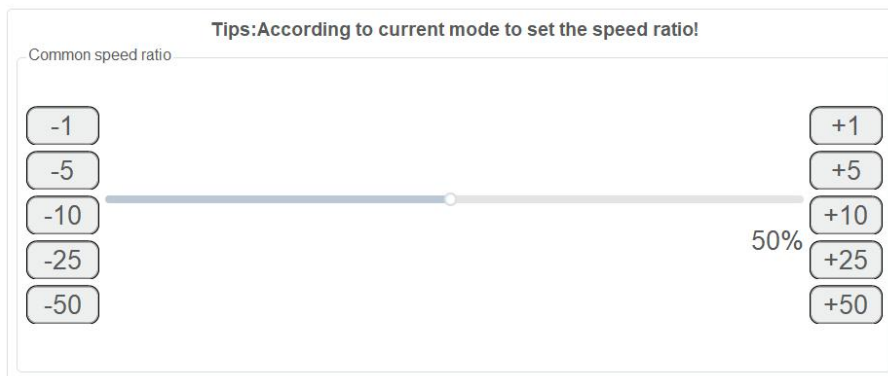


Figure 3.1 Modify the global velocity rate

When doing jogging or playback, the method calculating the velocity and acceleration for each axis (in Joint or Cartesian coordinate system) is shown as follows.

- Actual jogging velocity = the maximum jogging velocity \* global velocity rate
- Actual jogging acceleration = the maximum jogging acceleration\* global velocity rate
- Actual playback velocity = the maximum playback velocity \* global velocity rate \* the set velocity rate in the velocity function
- Actual playback acceleration = the maximum playback acceleration\* global velocity rate \* the set acceleration rate in the acceleration function
- Actual playback jerk = the maximum playback jerk \* global velocity rate \* the set acceleration rate in the jerk function



#### NOTE

- The maximum velocity, acceleration, or jerk can be set on the **Settings** page. For details, please see *3.7.6 RobotParams*.
- The rates (velocity rate, acceleration rate, or jerk rate) can be set in the related


speed functions.

### 3.3 Alarm Description

If teaching point is incorrect, for example, a robot moves to where a point is at a limited position or a singular point, an alarm will be triggered.

If an alarm is triggered when running MG400, the alarm icon  on the DobotSCStudio turns into . You can check the alarm information on the **Alarm** page, as shown in Figure 3.2.

Please clear the alarm as follows:

- If a limitation alarm is triggered, please jog the limited joint axis towards the opposite direction to clear the alarm.
- If other alarms are triggered, please click  on the alarm page to clear the alarm. If the alarm cannot be cleared, please reboot MG400.

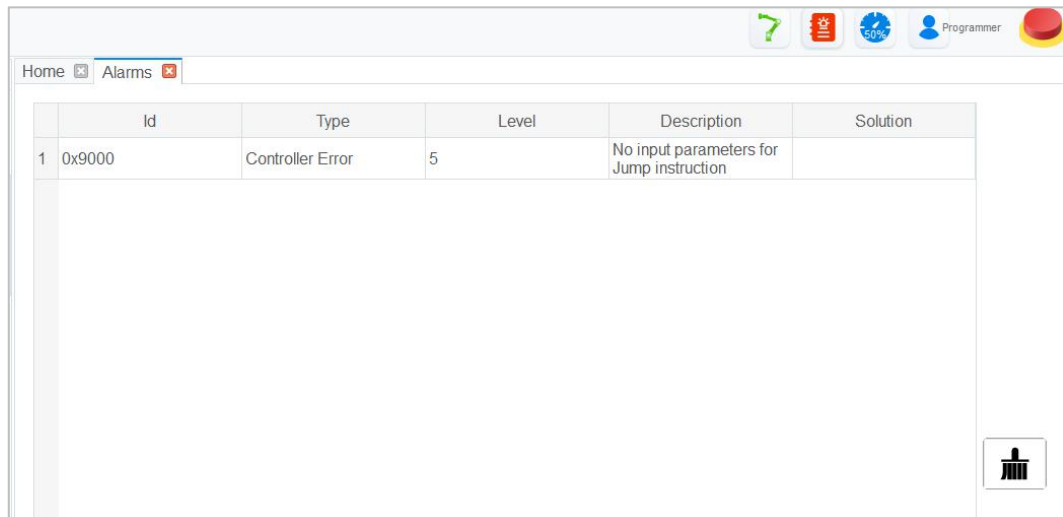


Figure 3.2 Alarm page

### 3.4 Jogging

You can jog the robot in different coordinate systems, Figure 3.3 shows the jogging panel, and Table 3.1 lists the description on jogging panel.

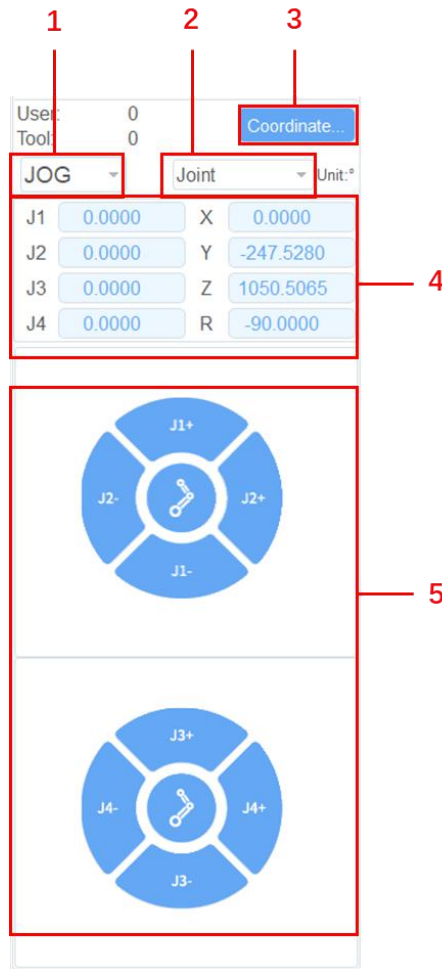


Figure 3.3 Jogging panel

Table 3.1 Description of jogging panel

No.	Description
1	<p>Step mode</p> <p>You can select the right step in the Step mode. The step supports JOG, 0.1, 0.5, 1 and 5.</p> <ul style="list-style-type: none"> <li>JOG: indicates that in continuous jog movement, the speed is the maximum speed * global velocity rate</li> <li>0.1, represents the displacement of 0.1° (joint coordinate system) or 0.1mm (Cartesian coordinate system) in a single jog movement</li> <li>0.5, represents the displacement of 0.5° (joint coordinate system) or 0.5mm (Cartesian coordinate system) in a single jog movement.</li> <li>1.0, represents the displacement of 1° (joint coordinate system) or 1mm (Cartesian coordinate system) in a single jog movement.</li> <li>5.0, represents the displacement of 5° (joint coordinate system) or 5mm (Cartesian coordinate system) in a single jog movement</li> </ul>

No.	Description
2	Jogging type It supports two types: Joint coordinate system, Cartesian coordinate system
3	Coordinate system According to the actual needs, you can select one of the preset user coordinate systems as the current user coordinate system
4	Location data Display the current joint position and tool center position
5	Jogging button Jog the robot in the Joint coordinate system: From top to bottom, jog J1, J2, J3 and J4 Jog the robot in the Cartesian coordinate system: From top to bottom, jog X, Y, Z, and R

### 3.5 Blockly

Blockly is a kind of block programming. You can write programs by graphical language to quickly and conveniently control the robot. Figure 3.4 shows the blockly panel, and Table 3.2 lists the description on Blockly panel.

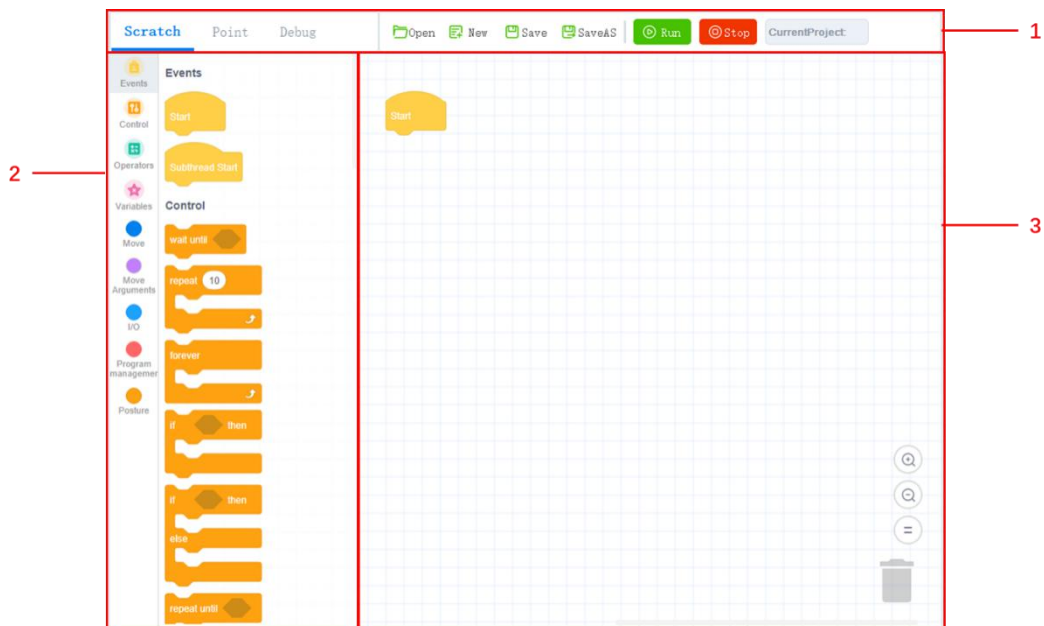













Figure 3.4 Blockly page

Table 3.2 Description on Blockly page


No.	Function	Description
-----	----------	-------------

1	Menu bar	<p> <b>Open</b> : Open a project that you have created</p> <p> <b>New</b> : Create a new project</p> <p> <b>Save</b> : Save the project</p> <p> <b>SaveAS</b> : Save the current project with a new name</p> <p> <b>Run</b> : Start running the program in the current code area</p> <p> <b>Stop</b> : Stop the running program</p> <p>Point: Save teaching point that can be called when writing a program</p> <p>Debug: Convert blockly into corresponding script. You can copy the script to  to see its running status. Please see <i>3.6 Programming</i> for details</p>
2	Block area	Provide all blocks
3	Code area	<p>Drag block to this page and edit it. Click the icon    in the code area to zoom in, zoom out and restore the blocks,  can be used to delete the selected block</p>

### Prerequisites

The robot has been powered on.

### Procedure

**Step 1** Click  to enter the Blockly page.

The system creates a new project by default and supports multi-thread movement.

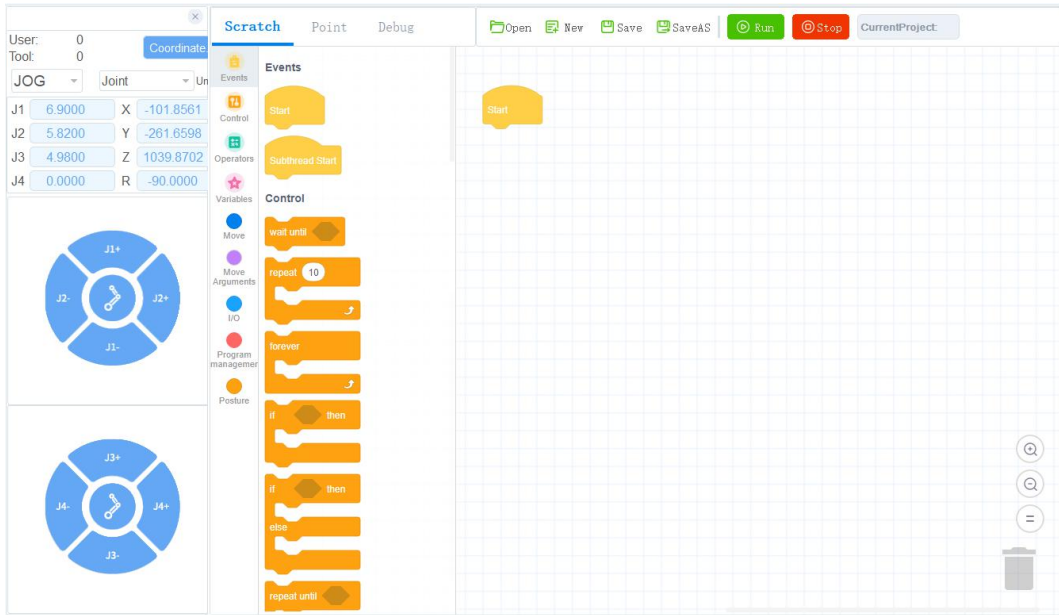


Figure 3.5 Blockly page

**Step 2** Drag the blocks to the code area to start programming, as shown in Figure 3.6.

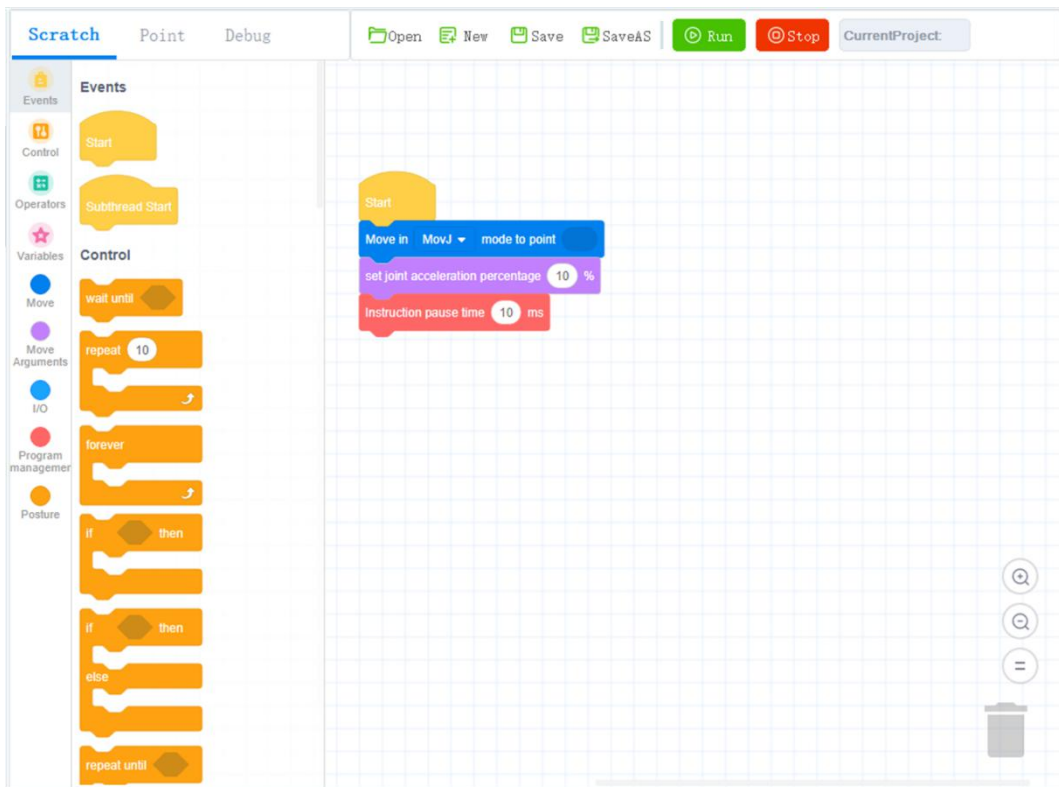



Figure 3.6 Writing a program

**Step 3** Click Save to save the current project

If it is the first time to save, you need to enter the project name.

**Step 4** Click  to enable the robotic arm.

**Step 5** Click  to start and run the program in the current code area.

## 3.6 Programming

### 3.6.1 Project Description

The robot program is managed in project form, including teaching points list, global variables, and program files. Figure 3.7 shows the project structure.

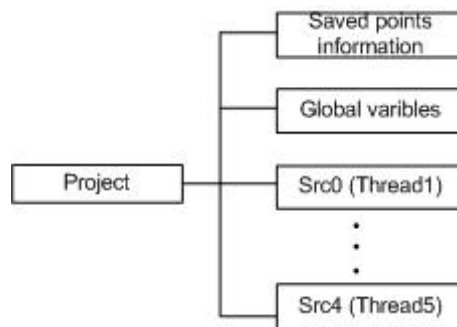


Figure 3.7 Project structure

- In script, no more than five threads can be executed simultaneously. Src0.lua is the main thread, and other threads are sub threads, which run parallel to the main thread.
- In the sub threads, the motion commands cannot be called. Only the main thread supports motion commands.
- Global variable module is only used to define global variables and module functions. The motion commands cannot be called here.

### 3.6.2 Programming Interface Description

When you write a program, you need to switch to programmer mode or above. Figure 3.8 shows the programming panel and Table 3.3 lists its description.



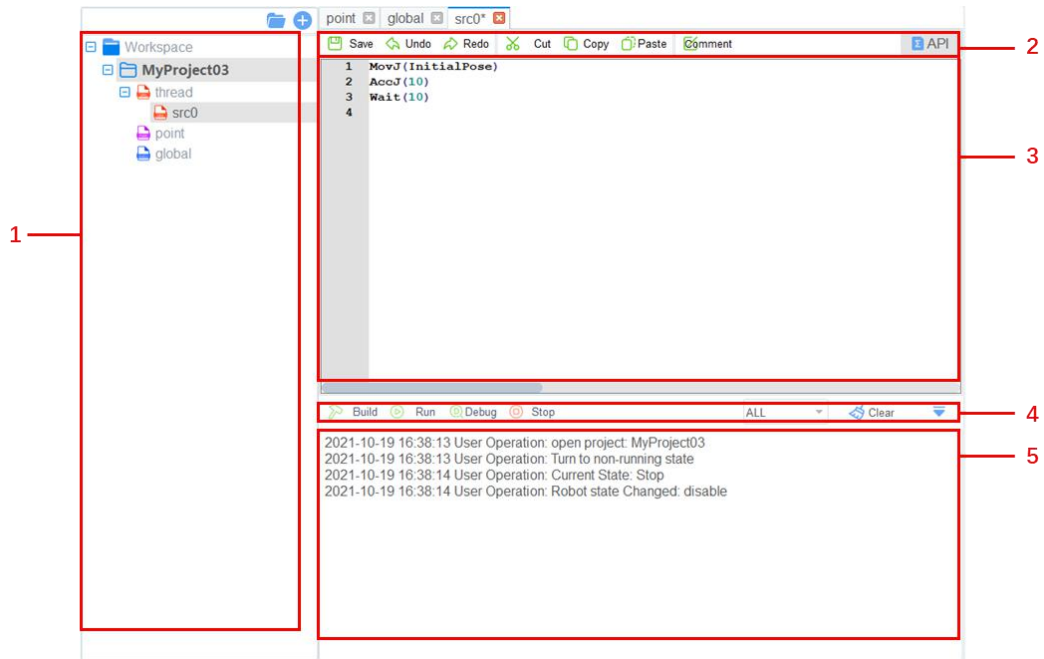


Figure 3.8 Programming panel







Table 3.3 Description on programming panel

No.	Description
1	Project files <ul style="list-style-type: none"> <li>Point: Teach points. For details, please see 3.6.3.3 <i>Teaching points</i>.</li> <li>Global: Define and initialize global variables, points or functions</li> <li>Src0~Src4: Multithreaded files. The number of tasks is related to threads that you set when creating a project. Up to five threads can be executed simultaneously</li> </ul>
2	Common buttons. For details, please see Table 3.4
3	Programming area
4	Running button, for details, please see Table 3.6
5	Debug result

Table 3.4 describes the common button.

Table 3.4 Description on common buttons

Icon	Description
Save	Save the project
Undo	Cancel

 Redo	Redo
 Copy	Copy the selected codes
 Cut	Cut the selected codes
 Paste	Paste the selected codes
 Comment	Code comment
 API	API libraries, for details, please see <i>4 Program Language</i>

### 3.6.3 Programming Description

Figure 3.9 shows the programming process.

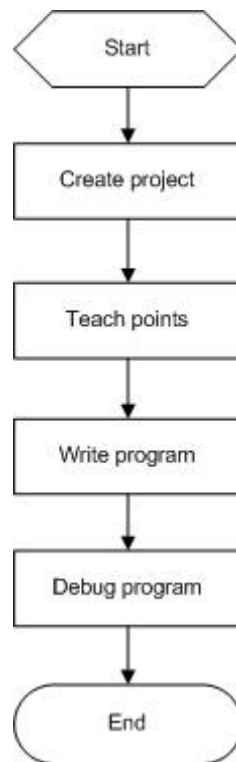


Figure 3.9 Programming process

#### 3.6.3.1 Creating Project

##### Prerequisites

The robot has been powered on.

## Procedure

**Step 1** Click  .

The programming page is displayed, as shown in Figure 3.10.

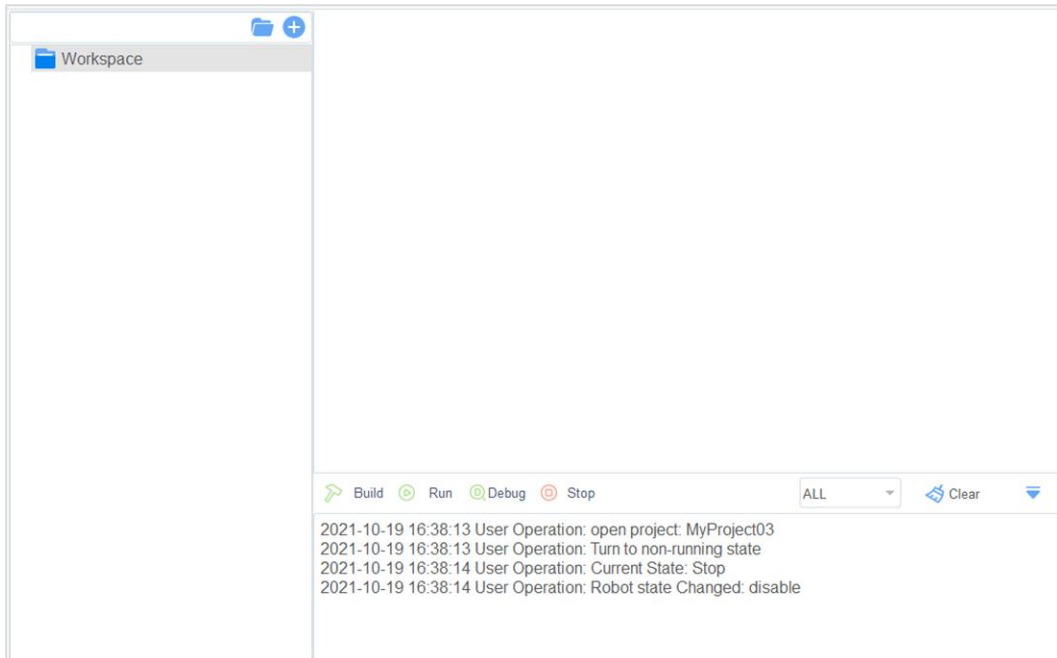



Figure 3.10 Programming page

**Step 2** Click  to enter the project creating page. Input the project name, and you can also select a template. Click **OK**.

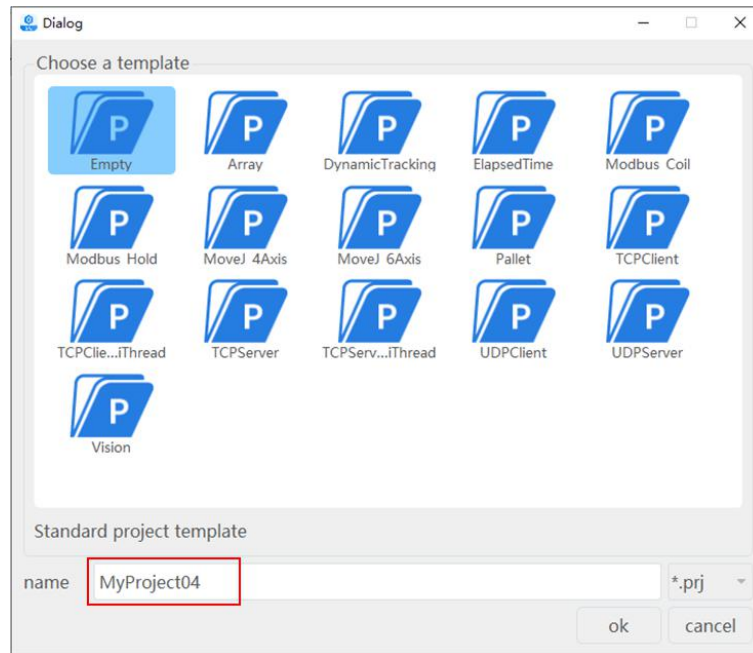


Figure 3.11 Create a project

**Step 3** Set the number of threads based on site requirements, as shown in Figure 3.12. Click **thread** and right-click **New thread file**.

The maximum number of threads is 5.

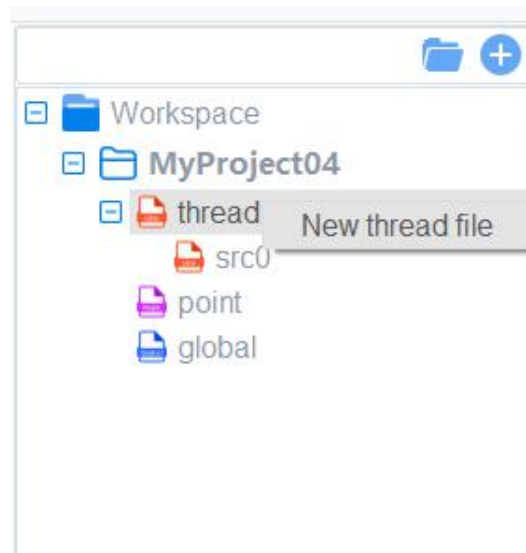


Figure 3.12 Create a project

**Step 4** (Optional) Import the existing taught positions list.

If you want to reuse a taught position list from an existing project, please right-click **point** and click **Import points File**, as shown in Figure 3.13.

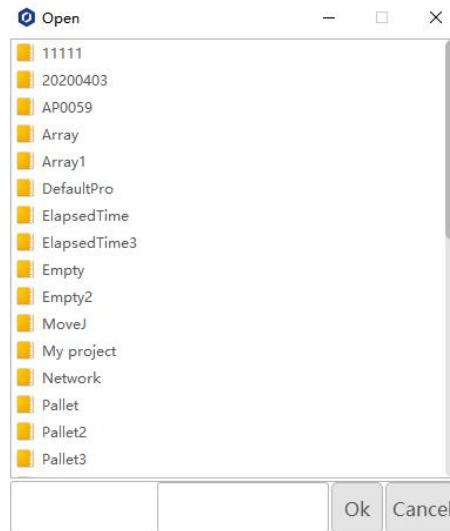


Figure 3.13 Import the existing teaching points list

### 3.6.3.2 Import Project

If you need to reuse project files of other MG400, you can export project files of other MG400 to local computer and then import them into the current MG400 from local computer.

#### Prerequisites

The robot has been powered on.

#### Procedure

**Step 1** Click **Workspace** and right-click **Import Project**.

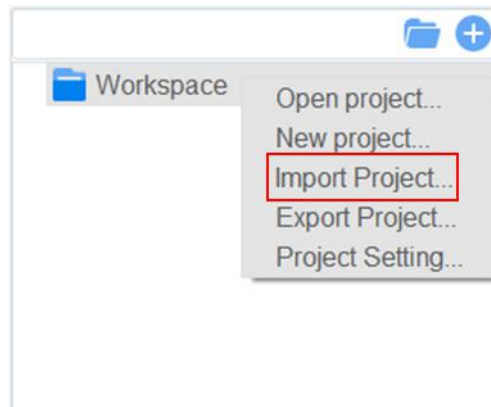


Figure 3.14 Import Project

**Step 2** Select a project to be imported.

In the **Import Project** page, there are two files: **prj.json** and **point.json**. Please select the project file **prj.json**.

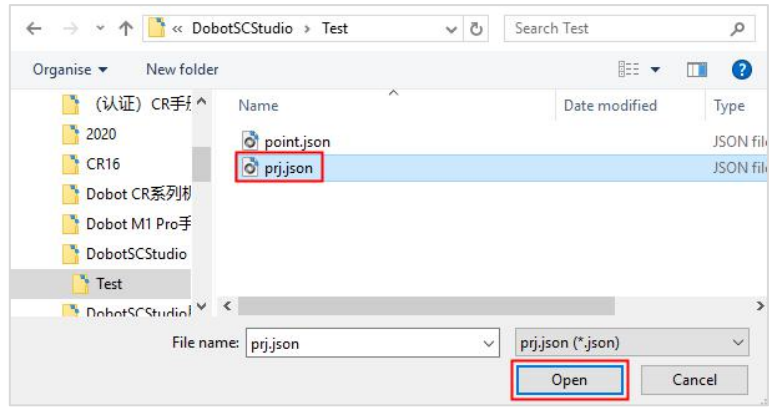


Figure 3.15 Select a project

**Step 3** Click **Open**.

The imported project files are displayed.

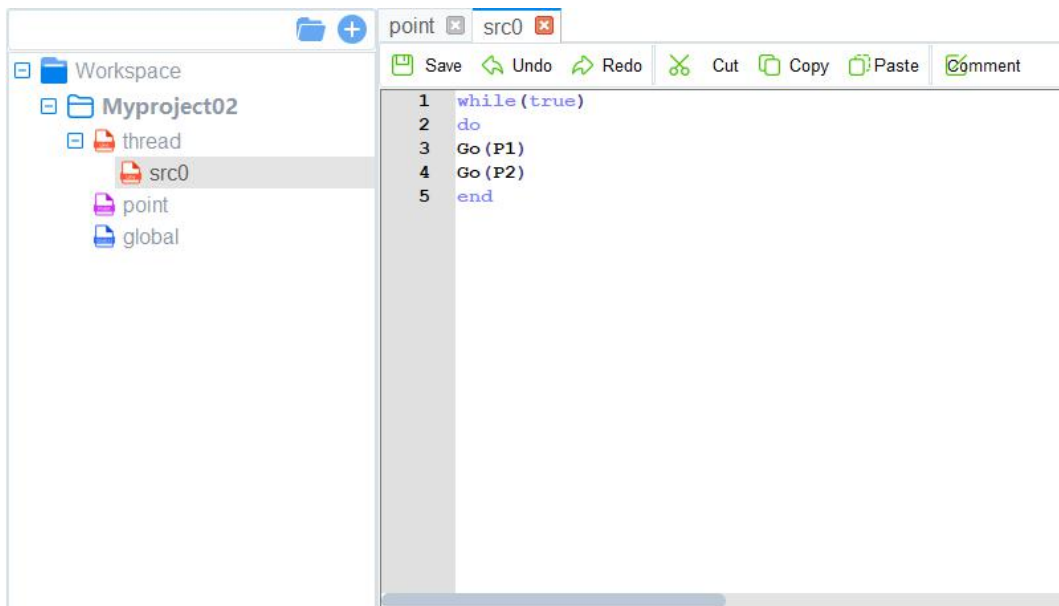


Figure 3.16 Display the project

**NOTE**

If the project has already been saved or imported into MG400, it is not allowed to import the same project, and a warning window will pop up indicating that you need to re-select a new project file to be imported.

### 3.6.3.3 Teaching points

**Prerequisites**


The project has been created or imported.


**Procedure**

After creating a project, please teach positions on the **point** page for calling commands when programming a robot. If the existing taught positions list has been imported, this operation can be

skipped.

**Step 1** Enable MG400.

**Step 2** Click  to move the robot to a point.

**Step 3** Double click **point** to enter point page and click  **Add** to add a teaching point.

The information of teaching point is displayed on the **point** page, as shown in Figure 3.17.

**Tool** is the Tool coordinate system, **User** is the User coordinate system.


Save                      RunTo                      Cover                      Add                      Delete                      Undo                      Redo									
No.	Alias	X	Y	Z	R	Arm	Tool	User	
1	P1	-17.0139	-348.4705	986.5601	-90.0000	Right	No.0	No.0	
2	P2	-74.4932	-400.2468	979.8826	-90.0000	Right	No.0	No.0	

Figure 3.17 Teaching points list


Table 3.5 Button description

Button	Description
Add	Add a point
Delete	Delete a point
Cover	Cover a point. Select a teaching point, after jogging the robot to a point, click the icon to cover the selected teaching point
RunTo	Run to a point, select a point, click the button to run the robot to this point
Save	Save teaching point
Undo	Cancel
Redo	Recover

- You can select a teaching position and double-click the parameters to modify the relevant information.

- Also, you can select a teaching position and click  **Cover** to cover the current teaching position.

**Step 4** Add points by referring to Step 2 and Step 3.

**Step 5** Click  **Save** to save the teaching points.

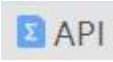
### 3.6.3.4 Writing a Program

#### Prerequisites

- The project has been created or imported.
- The points have been taught.

#### Procedure

In MG400, common commands for programming with Lua language have been encapsulated. Please see 4Program Language.

**Step 1** Click , and the command list will display on the right side of the page, as shown in Figure 3.18.

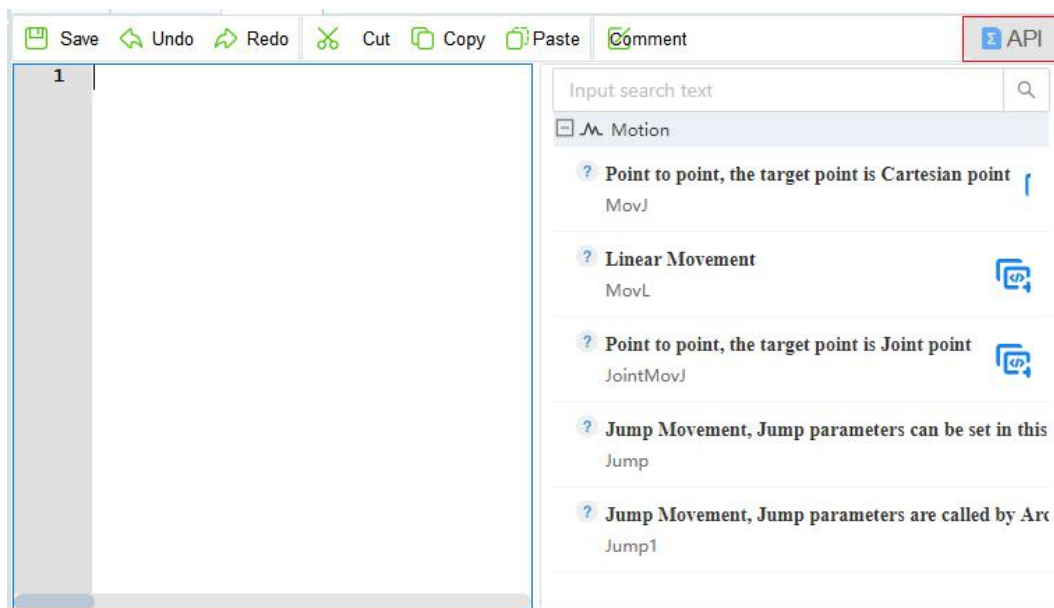



Figure 3.18 Command list

**Step 2** Select the commands from the list. Double-click the command and it will display on the script area.

**Step 3** Click  **Save** after you finish the programming.



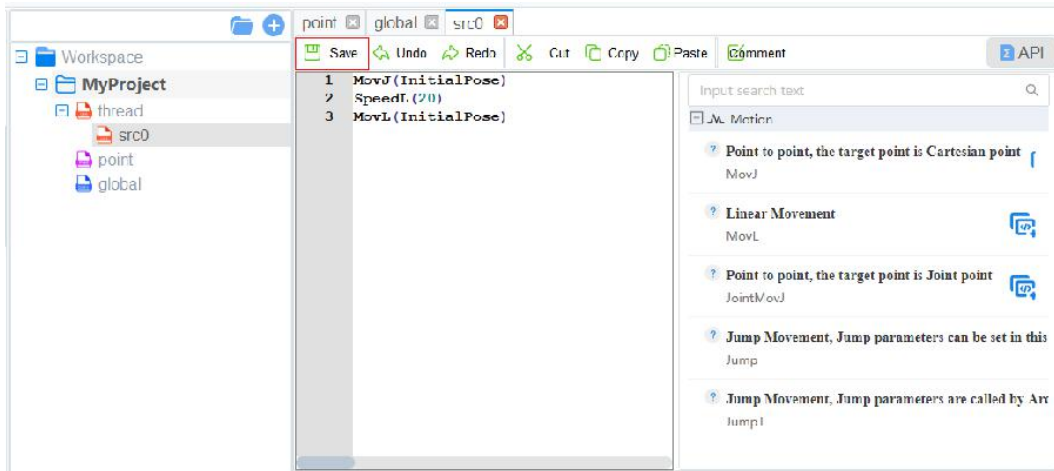


Figure 3.19 Save the project

### 3.6.3.5 Debugging Program

**Step 1** Click  to enable the motor.

Now, the programming page is as shown in Figure 3.20.

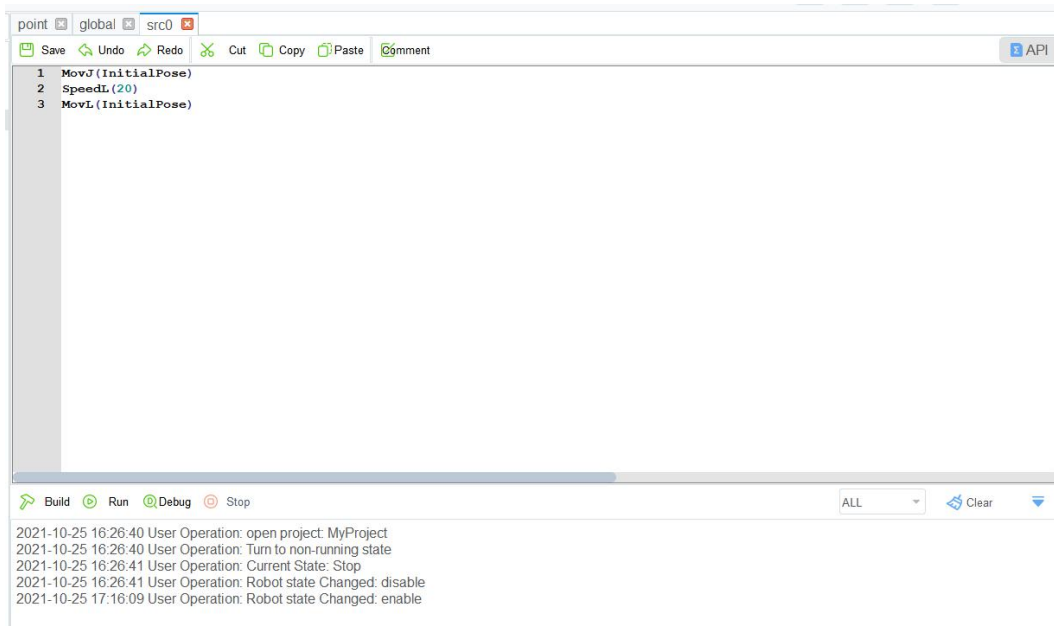



















Figure 3.20 Programming page

Table 3.6 lists the description on the program-running buttons which are shown in Figure 3.20

Table 3.6 Program-running button description

Icon	Description
 Build	Build program Check if the code is correct
 Run	Once-click run After clicking this button,  Run turns into  Pause and the program starts running If you need to pause the running program, please click 
 Debug	Start to run a program Click once: Start to debug a program,  Step appears on the screen. Click  Step: Start to run a program
 Stop	Stop the running program
 Monitor	After you click  Step in debugging the program,  Monitor appears on the screen to monitor the running of program
 Resume	After you click  Pause,  Resume appears on the screen. You can click it to continue running the program.

**Step 2** Click  Run to start debugging the program.

- If you want to run a program step by step, click  Debug. Click  Step after it appears on the screen.

### 3.6.3.6 Export Project

Project is saved in MG400 by default, and you can import the project to a local directory.

**Step 1** Click **Workspace** and right-click **Export Project**.

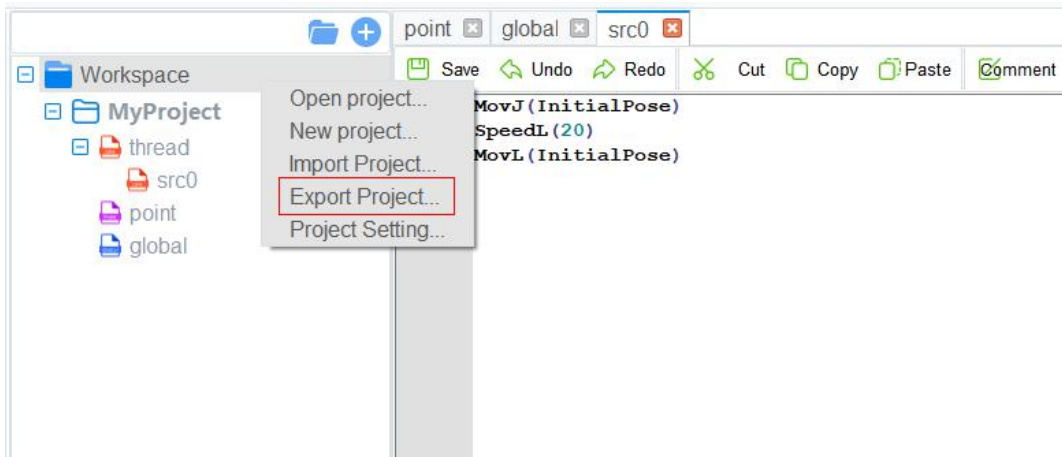


Figure 3.21 Export Project

- Step 2** In the project list, click the project to be exported.  
For example, click “Factory Test 1”.

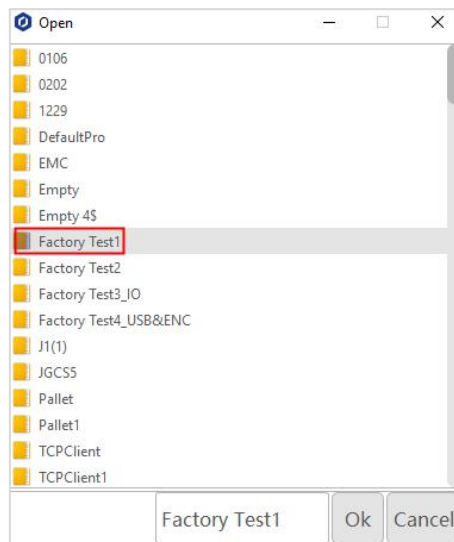


Figure 3.22 Select Project Files

- Step 3** Select a right path, then click **Select Folder**.

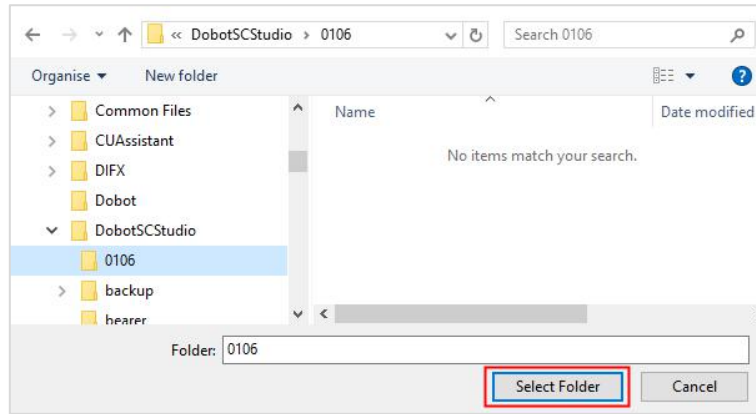


Figure 3.23 Save path

## 3.7 Parameter

Before teaching or running robot programs, a series of settings are required, including motion parameter setting, language selecting, user mode selecting and process setting, etc.

### 3.7.1 Setting User Coordinate System

When the position of workpiece is changed or a robot program needs to be reused in multiple processing systems of the same type, you can create coordinate systems on the workpiece to simplify programming. There are totally 10 groups of User coordinate systems, of which the first one is defined as the Base coordinate system by default and cannot be changed. And the others can be customized by users.

#### NOTICE

When creating a User coordinate system, please make sure that the reference coordinate system is the Base coordinate system. Namely, the User coordinate system icon should be `User: 0` when creating a User coordinate system.

User coordinate system is created by two-point calibration method. Move the robot to three points **P0(x0, y0, z0)**, **P1(x1, y1, z1)**. Point P0 is defined as the origin and the line from point P0 to Point P1 is defined as the positive direction of X-axis. And then the Y-axis and Z-axis can be defined based on the right-handed rule, as shown in Figure 3.24.

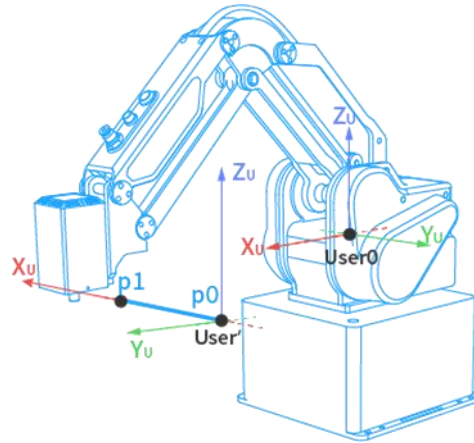


Figure 3.24 Two point calibration

Take the establishment of User 2 coordinate system as an example based on two-point calibration method.

#### Prerequisites

- The robot has been powered on.
- MG400 has been enabled.
- The robot is in the Cartesian coordinate system.

#### Procedure

**Step 1** Click  > **Parameter** > **GlobalCoordinate** > **Coordinate User**.

The **Coordinate User** page is displayed, as shown in Figure 3.25.

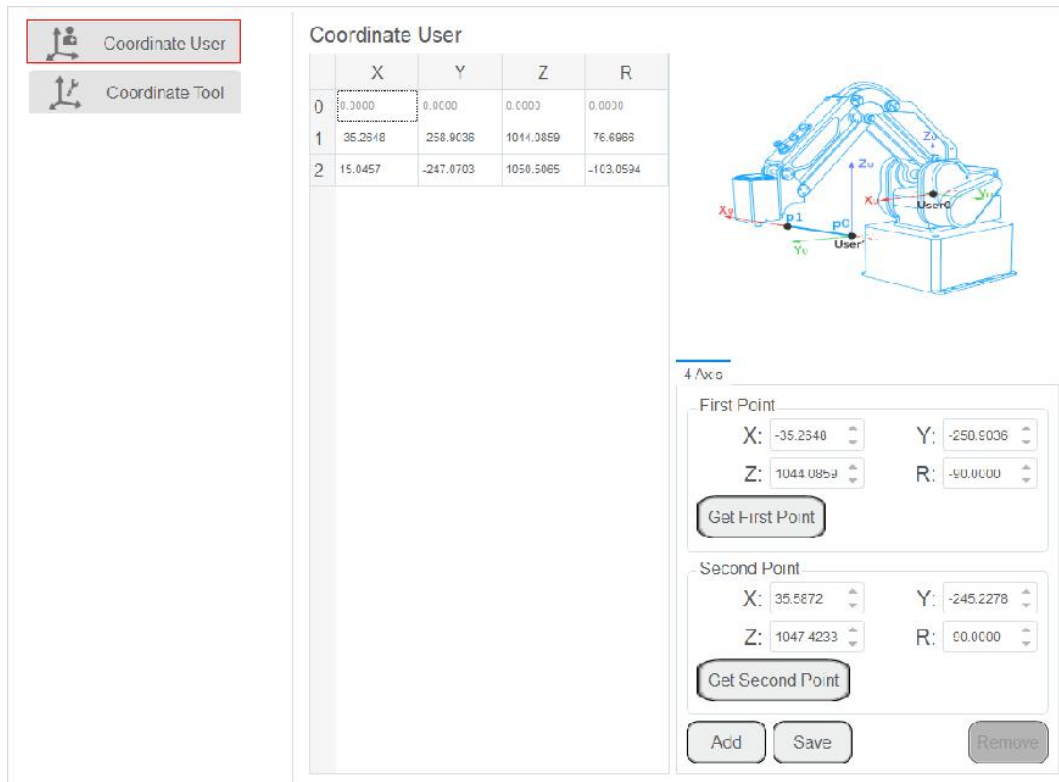


Figure 3.25 User coordinate system page

- Step 2** Jog the robot to the first point, then click **Get First Point** on the **P1** tab to obtain the coordinates of the first point.
- Step 3** Jog the robot to the second point, then click **Get Second Point** on the **P2** tab to obtain the coordinates of the second point.
- Step 4** Click **Cover** and **Save** to generate the User 2 coordinate system, as shown in Figure 3.26. If you do not select an existing coordinate system, you can repeat Step1~Step3, and then click **Add** and **Save**, as shown in Figure 3.27.

### Coordinate User

	X	Y	Z	R
0	0.0000	0.0000	0.0000	0.0000
1	0.0000	0.0000	0.0000	0.0000
2	15.0457	-247.0703	1050.5065	-103.0594

tips X

Save User Coordinate Success

4 Axis

First Point

X:  Y:

Z:  R:

Second Point

X:  Y:

Z:  R:

Figure 3.26 Click Cover to generate a coordinate system

### Coordinate User

	X	Y	Z	R
0	0.0000	0.0000	0.0000	0.0000
1	-35.2648	-258.9036	1044.0859	-76.6996
2	0.0000	0.0000	0.0000	0.0000
3	13.8233	-250.6402	1048.6381	-0.0256
4	400.4405	69.7307	947.8878	-166.6274
5	-3.5011	-248.0172	1050.2390	179.2129

4 Axis

**First Point**

X:  Y:

Z:  R:

**Second Point**

X:  Y:

Z:  R:

Figure 3.27 Click Add to generate a coordinate system

**Step 5** Select **User 2** on Jog interface.

You can use the **User 2** coordinate system for teaching and programming.



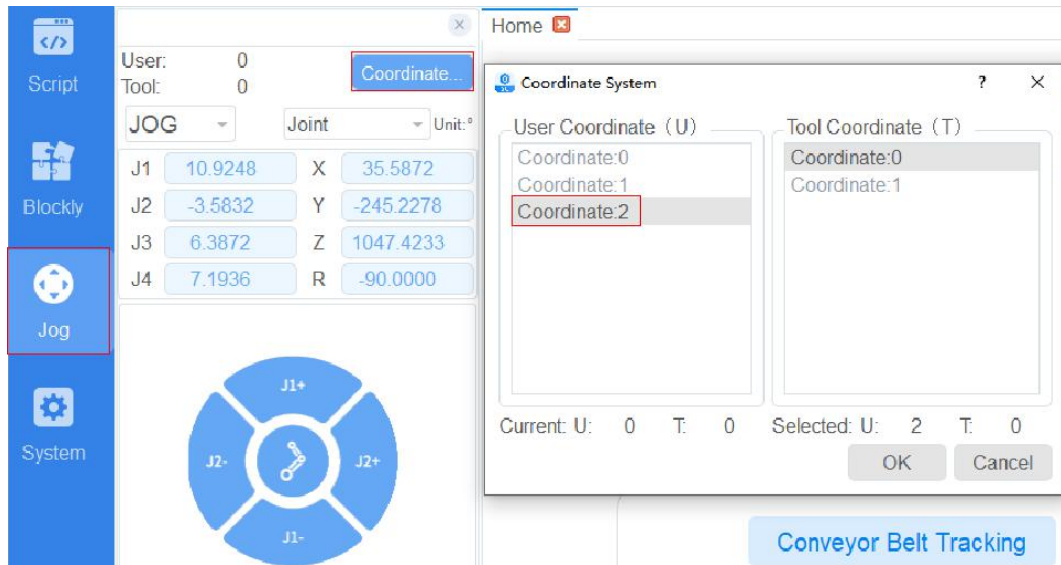


Figure 3.28 Select user coordinate system

### 3.7.2 Setting Tool Coordinate System

When an end effector such as welding gun, gripper is mounted on the robot, the Tool coordinate system is required for programming and operating a robot. For example, you can use multiple grippers to carry multiple workpieces simultaneously to improve the efficiency by setting each gripper to a Tool coordinate system.

There are totally 10 groups of Tool coordinate systems. Tool 0 coordinate system is the predefined Tool coordinate system which is located at the robot flange and cannot be changed.

#### NOTICE

When creating a Tool coordinate system, please make sure that the reference coordinate system is the predefined Tool coordinate system. Namely, the Tool coordinate system icon should be `Tool: 0` when creating a Tool coordinate system.

Tool coordinate system of robot is created by two-point calibration method: After an end effector is mounted, please adjust the direction of this end effector to make the TCP (Tool Center Point) align with the same point (reference point) in two different directions, for obtaining the position offset to generate a Tool coordinate system, as shown in Figure 3.29.

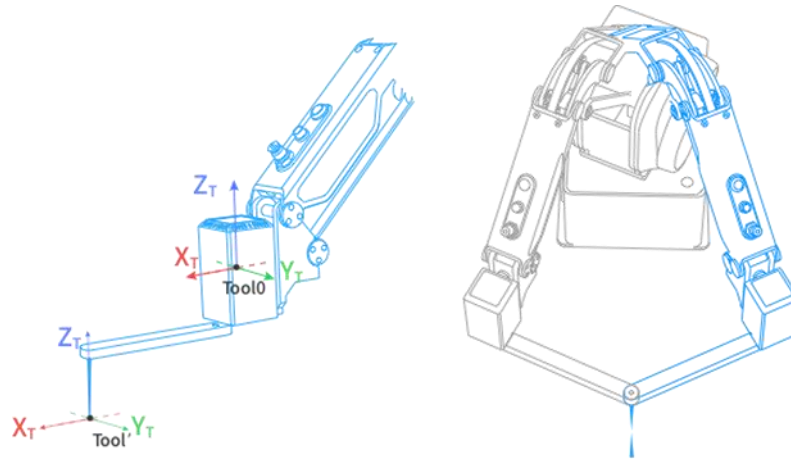


Figure 3.29 Two points calibration method (TCP+ZX)


Take the establishment of Tool 1 coordinate system as an example.

#### Prerequisites

- The robot has been powered on.
- MG400 has been enabled.
- The robot is in the Cartesian coordinate system.

#### Procedure

**Step 1** Mount an eccentric end effector on the robot. The detailed instructions are not described in this topic.

**Step 2** Click  > **Parameter** > **GlobalCoordinate** > **Coordinate Tool**.

The Coordinate Tool page is displayed in Figure 3.30.

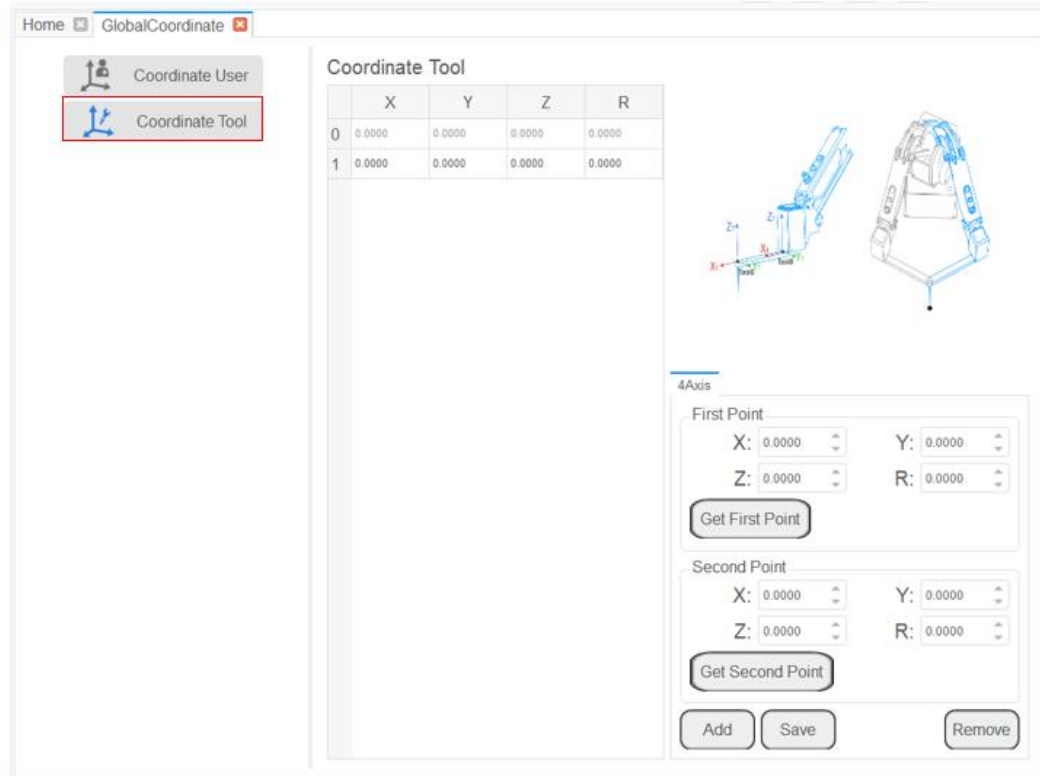


Figure 3.30 Tool Coordinate page

- Step 3** Jog the robot to the reference point in the first direction, then click **Get First Point** to obtain the coordinates of the first point.
- Step 4** Jog the robot to the reference point in the second direction, then click **Get Second Point** to obtain the coordinates of the second point.
- Step 5** Click **Save** and **Save** to generate the Tool 1 coordinate system.

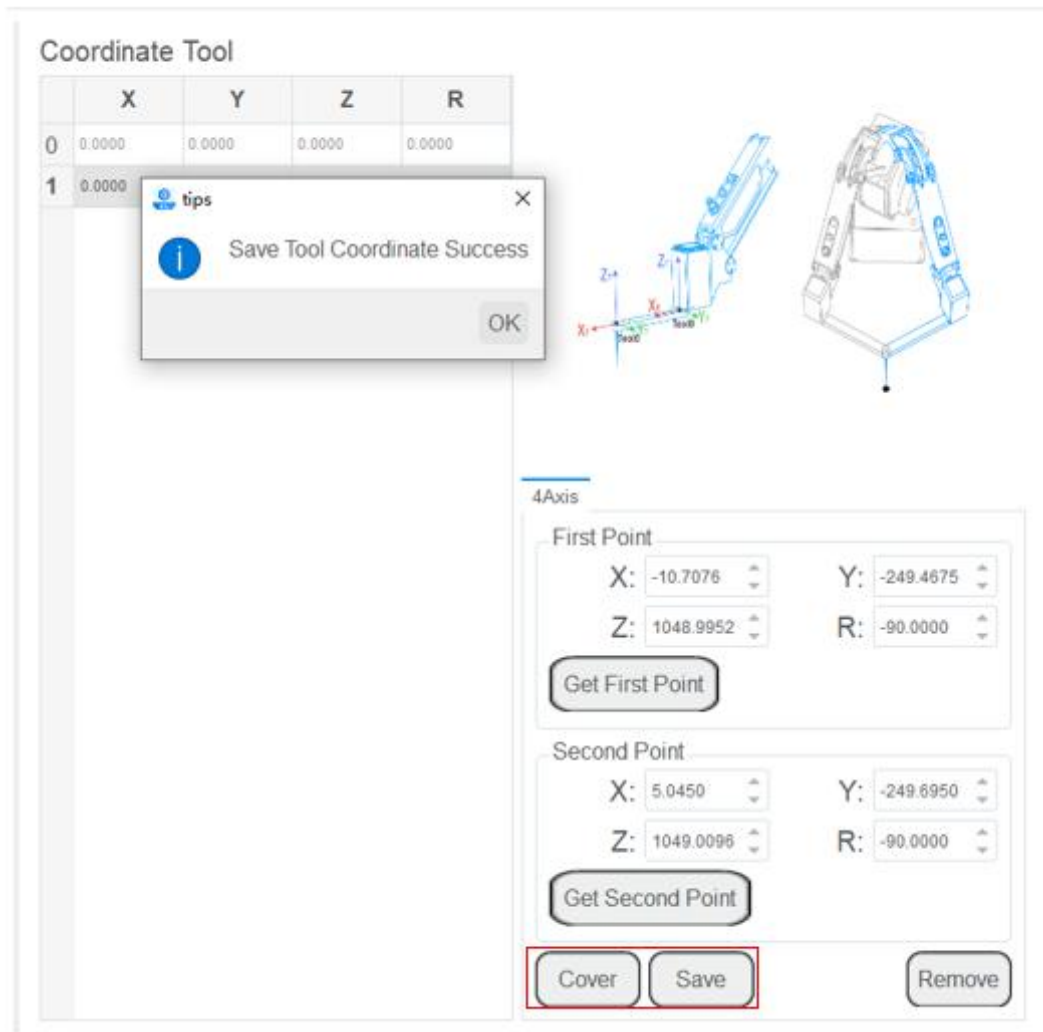


Figure 3.31 Tool 2 coordinate system

### 3.7.3 I/O Monitor


You can set or monitor the I/O status of the robot and robot end on this page. Click  > **Parameter > IOMonitorPro** to enter the I/O monitor page, as shown in Figure 3.32.



Figure 3.32 I/O monitor page

There are three features: Output, monitor and simulation.

- Output: Set the digital output.
- Monitor: Check the status of the input and output.
- Simulation: Simulate the digital input for debugging and running program, as shown in Figure 3.33.

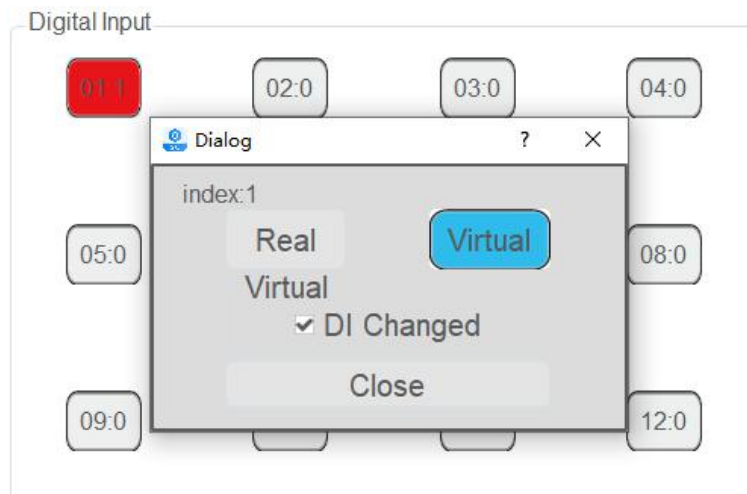



Figure 3.33 Simulation

### 3.7.4 Controller Setting

#### 3.7.4.1 Reboot

When the controller firmware has been updated or the robot is abnormal, you need to reboot

the robot. Now you can click  to reboot it on the **Parameter > ControllerSetting > Reboot** page.

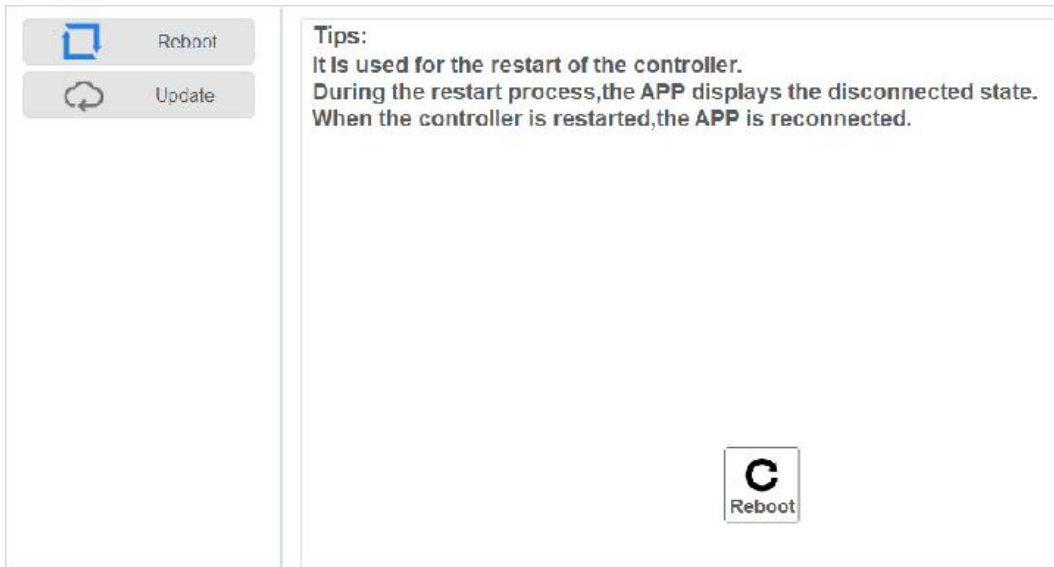


Figure 3.34 Reboot

### 3.7.4.2 Update

When the controller firmware needs to be updated, you can import the latest firmware on this page. After importing the firmware, please reboot the robot.

Please contact the Dobot support engineer to obtain the latest firmware.

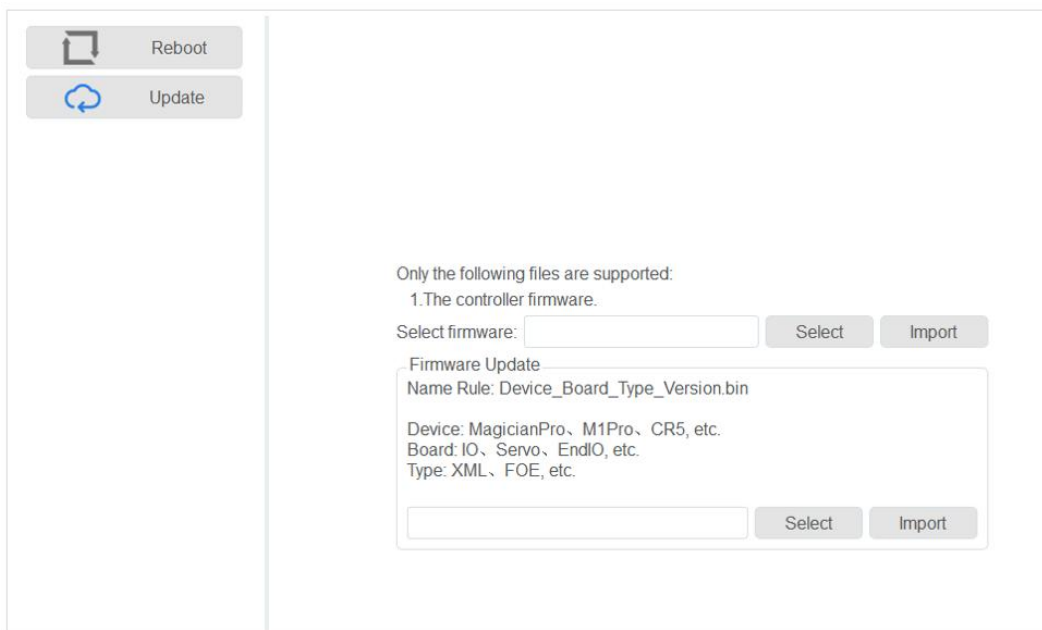


Figure 3.35 Update firmware

### 3.7.5 Remote Control

External equipment can send commands to a robot by different remote control modes, such as remote I/O mode and remote Modbus mode. The default mode is Teaching mode when the robot is shipped out. When you need to set the remote mode, please set it on the DobotSCStudio with MG400 in the disabled state.

#### NOTICE

- Robot rebooting is not required when switching the remote mode.
- The emergency stop switch on the hardware is always available no matter what mode MG400 is in.
- When MG400 is running in remote control mode, if the start signal is triggered on the external device, MG400 will be automatically enabled and run according to the selected project file.

#### 3.7.5.1 Remote I/O

When the remote mode is I/O mode, external equipment can control a robot in this mode. The specific description on I/O interface is shown in Table 3.7.

Table 3.7 Description on I/O interface

I/O interface	Description
Input (For external control)	
DI 11	Clear alarm
DI 12	Continue to run
DI 13	Pause running in the I/O mode
DI 14	Stop running and exit the I/O mode
DI 15	Start to run in the I/O mode
DI 16	Emergency stop and exit the I/O mode
Output (For displaying the status)	
DO 13	Ready status
DO 14	Pause status
DO 15	Alarm status
DO 16	Running status

#### NOTICE

All input signals are rising-edge triggered. A change from low level to high level is effective.

## Prerequisites

- The project run in the remote mode has been prepared.
- The external equipment has been connected to MG400 by the I/O interface. The specific I/O interface description is shown in Table 3.7.
- The robot has been powered on.

### NOTE

The details on how to connect external equipment and use it are not described in this topic.

## Procedure

- Step 1** Click  > **Parameter** > **RemoteControl**.

The remote control page is displayed, as shown in Figure 3.36.

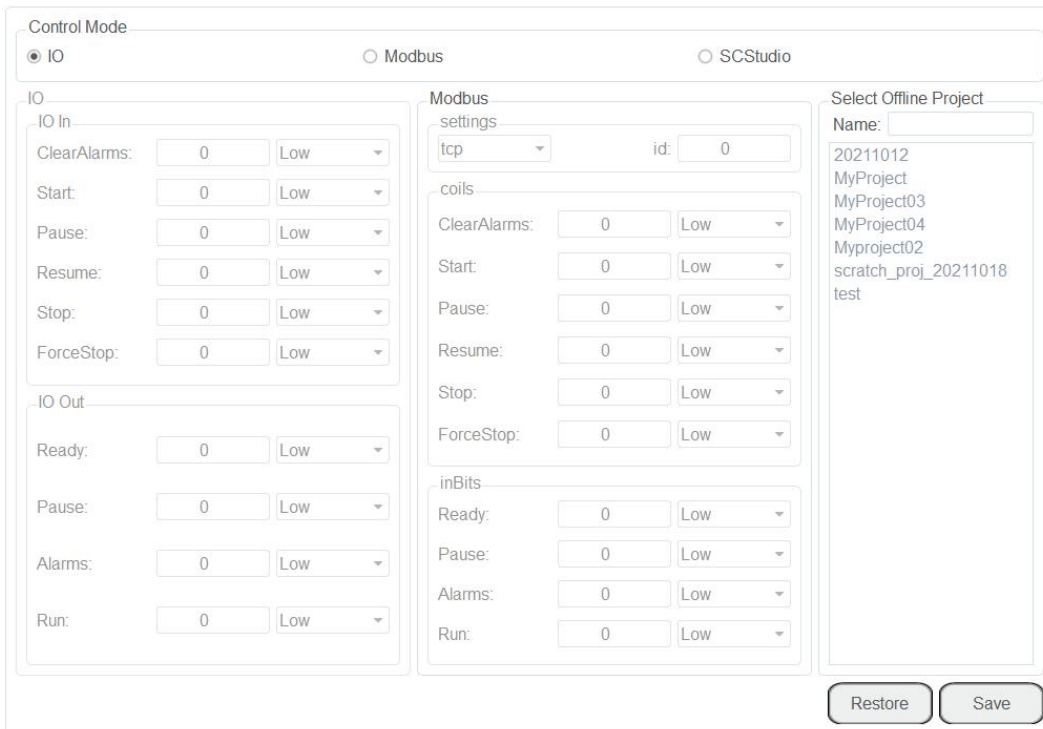


Figure 3.36 Remote control page

- Step 2** Select **IO** on the **Control Mode** section and select the offline project on the **Select Offline Project** section, click **Save**.

The **Save success, now remote control mode is IO** page is displayed.

Right now, only the emergency stop button is available.

- Step 3** Trigger the starting signal on the external equipment.

The robot will move as the selected offline project. If the stop signal is triggered, the robot will exit remote IO mode.



### 3.7.5.2 Remote Modbus

When the remote mode is Modbus mode, external equipment can control a robot in this mode. For details about Modbus registers, please see *4.15.1 Description on Modbus Register*. The specific description on Modbus register is shown in Table 3.8.

Table 3.8 Specific Modbus register description

Register address (Take a PLC as an example)	Register address (MG400)	Description
Coil register		
00001	0	Start running in the remote Modbus mode
00002	1	Pause running in the remote Modbus mode
00003	2	Continue to run
00004	3	Stop to run and exit the remote Modbus mode
00005	4	Emergency stop and exit the remote Modbus mode
00006	5	Clear alarm
Discrete input register		
10001	0	Auto-exit
10002	1	Ready status
10003	2	Pause status
10004	3	Running status
10005	4	Alarm status

#### Prerequisites

- The project to be running in the remote mode has been prepared.
- The robot has been connected to the external equipment with the Ethernet2 interface. The default IP address is **192.168.2.6**. You can connect them directly or via a router, please select based on site requirements.

The IP address of MG400 must be in the same network segment of the external equipment without conflict. You can modify the IP address on the **ToolConfig > NetworkSetting** page; the default port is **502** and cannot be modified.

- The robot has been powered on.

## NOTE

The details on how to connect external equipment and use it are not described in this topic.

## Procedure

**Step 1** Click  > **Parameter** > **RemoteControl**.

The remote control page is displayed, as shown in Figure 3.37.

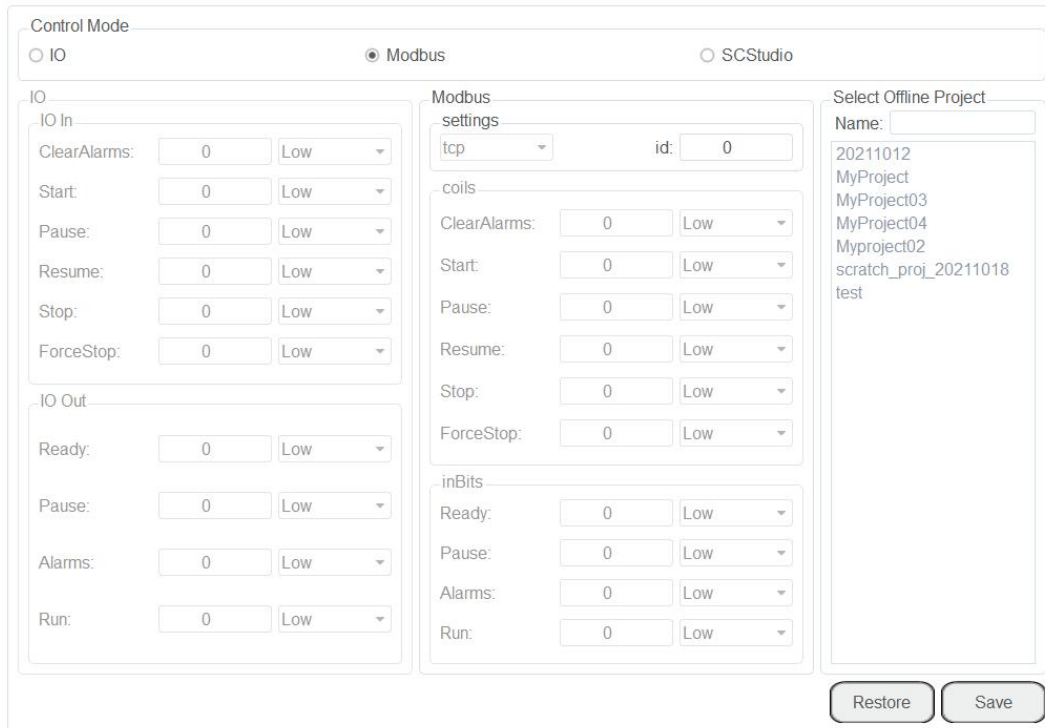


Figure 3.37 Remote control page

**Step 2** Select **Modbus** on the **Control Mode** section and select the offline project on the **Select Offline Project** section, and click **Save**.

The **Save success, now remote control mode is Modbus** page is displayed.

**Step 3** Trigger the starting signal on the external equipment.

The robot will move as the selected offline project. If the stop signal is triggered, the remote Modbus mode will be invalid.

### 3.7.6 RobotParams

You can set the velocity, acceleration or other parameters in different coordinate systems when jogging a robot or running robot programs. After setting the parameters, please click **Save**.

Click  > **Parameter** > **RobotParams** to enter **RobotParams** interface.

- Teach Joint Parameter: Set the maximum velocity and acceleration in the Joint coordinate system when jogging a robot. The jogging parameters of a robot in the Joint coordinate system are as shown in Figure 3.38.

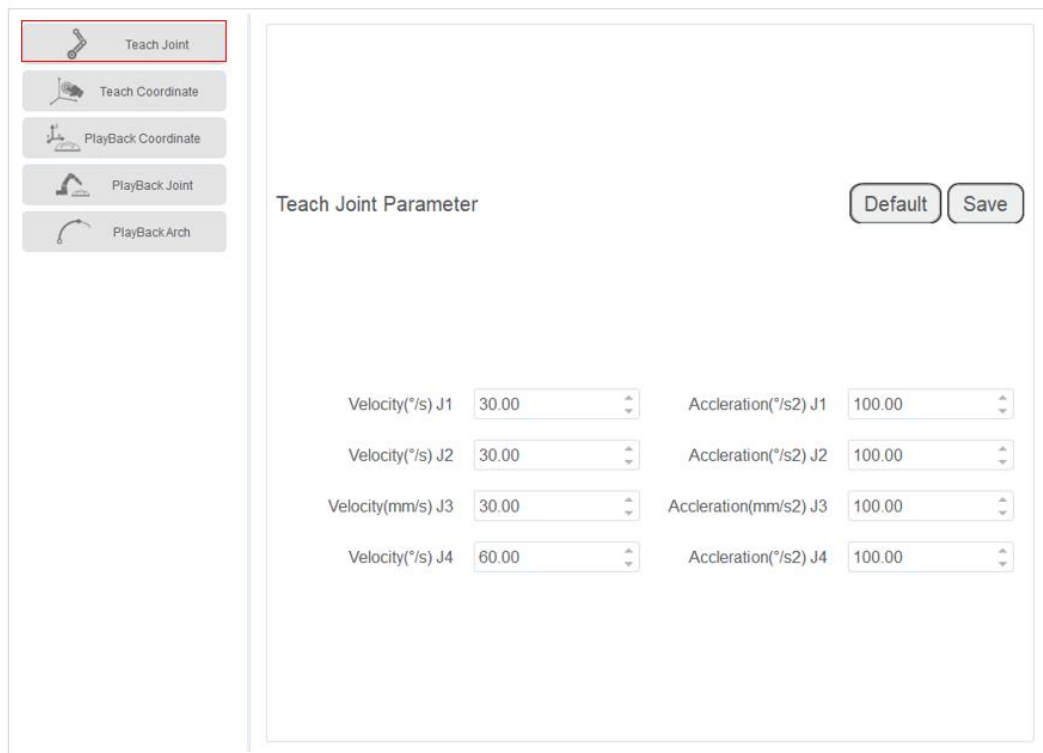


Figure 3.38 Jogging parameters in the Joint coordinate system

The relations between the actual velocity of each joint and the maximum velocity are shown as follows:

- Jog velocity of each joint = maximum velocity of each joint \* global velocity rate
- Jog acceleration of each joint = maximum acceleration of each joint \* global velocity rate

#### NOTE

- You can set the global velocity rate on the main page. For details, see *3.2 Setting Global Velocity Rate*.
  - You can set the percentage of commands by calling speed commands when programming.
- Teach Coordinate Parameter: Set the maximum velocity and acceleration in the Cartesian coordinate system when jogging a robot. The jogging parameters of a robot in the Cartesian coordinate system are as shown in Figure 3.39.

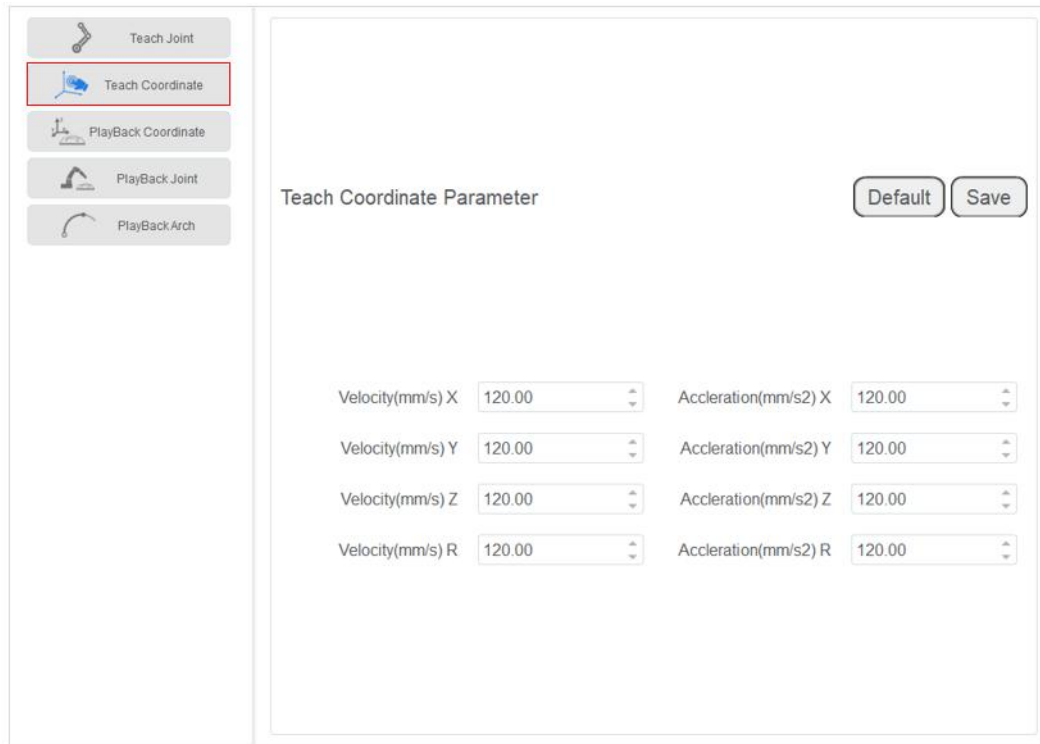


Figure 3.39 Jogging parameters in the Cartesian coordinate system

The relations between the actual velocity of each Cartesian axis and the maximum velocity are shown as follows:

- Jog velocity of each Cartesian axis = maximum velocity of each Cartesian axis \* global velocity rate
- Jog acceleration of each Cartesian axis = maximum acceleration of each Cartesian axis \* global velocity rate
- Playback Coordinate Parameter: Set the maximum velocity, acceleration and jerk in the Cartesian coordinate system when running robot programs. The playback parameters of a robot in the Cartesian coordinate system are as shown in Figure 3.40.

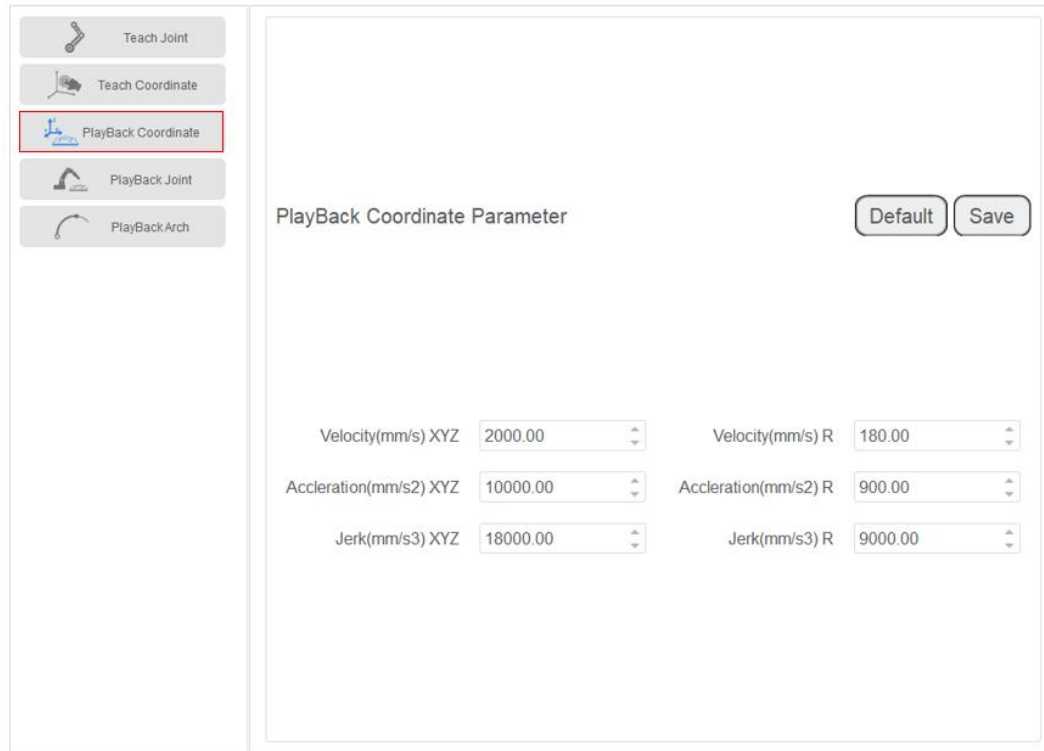


Figure 3.40 Playback parameters in the Cartesian coordinate system

The relations between the actual velocity of each Cartesian axis and the maximum velocity are shown as follows:

- Playback velocity of each Cartesian axis = maximum velocity of each Cartesian axis \* global velocity rate \* percentage set in commands
- Playback acceleration of each Cartesian axis = maximum acceleration of each Cartesian axis \* global velocity rate \* percentage set in commands
- Playback jerk of each Cartesian axis = maximum jerk of each Cartesian axis \* global velocity rate \* percentage set in commands

- Playback Joint Parameter: Set the maximum velocity, acceleration, and jerk in the Joint coordinate system when running robot programs. The playback parameters of a robot in the Joint coordinate system are as shown in Figure 3.41.

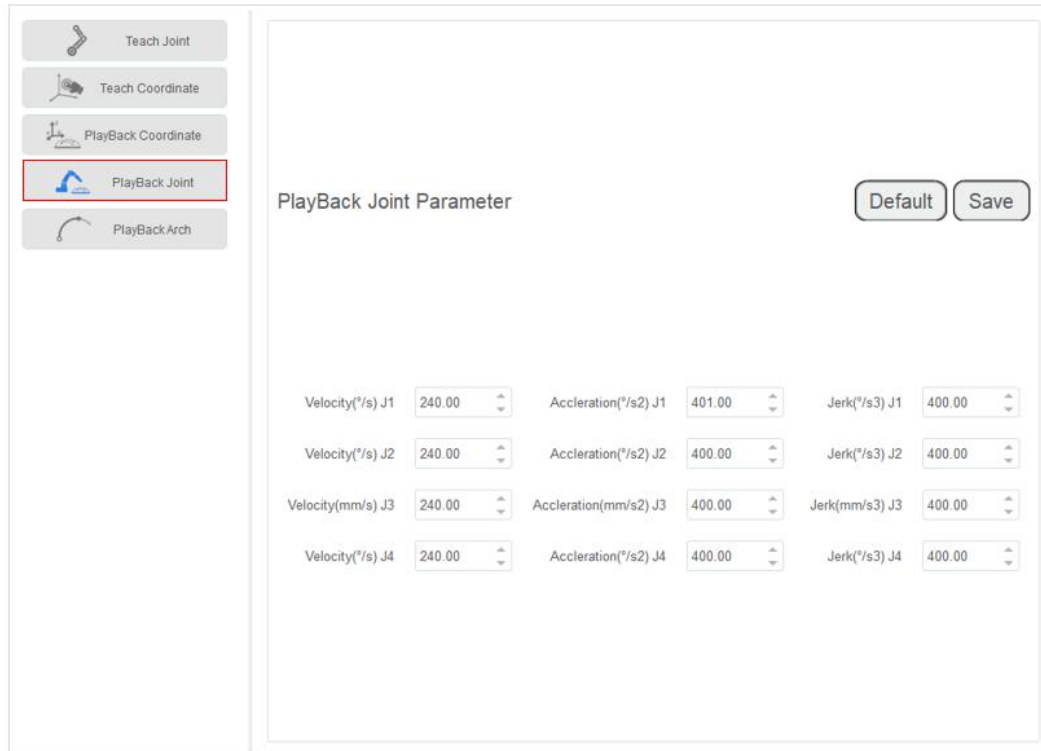


Figure 3.41 Playback parameters in the Joint coordinate system

The relations between the actual velocity of each joint and the maximum velocity are shown as follows:

- Playback velocity of each joint = maximum velocity of each joint \* global velocity rate \* percentage set in commands
- Playback acceleration of each joint = maximum acceleration of each joint \* global velocity rate \* percentage set in commands
- Playback jerk of each joint = maximum jerk of each joint \* global velocity rate \* percentage set in commands

- Playback Arch Parameter: If the motion mode is **Jump** when running robot programs, you need to set **StartHeight**, **EndHeight**, and **zLimit**.

You can set 10 sets of Jump parameters. Please set and select any set of parameters for calling Jump command during programming, as shown in Figure 3.42.

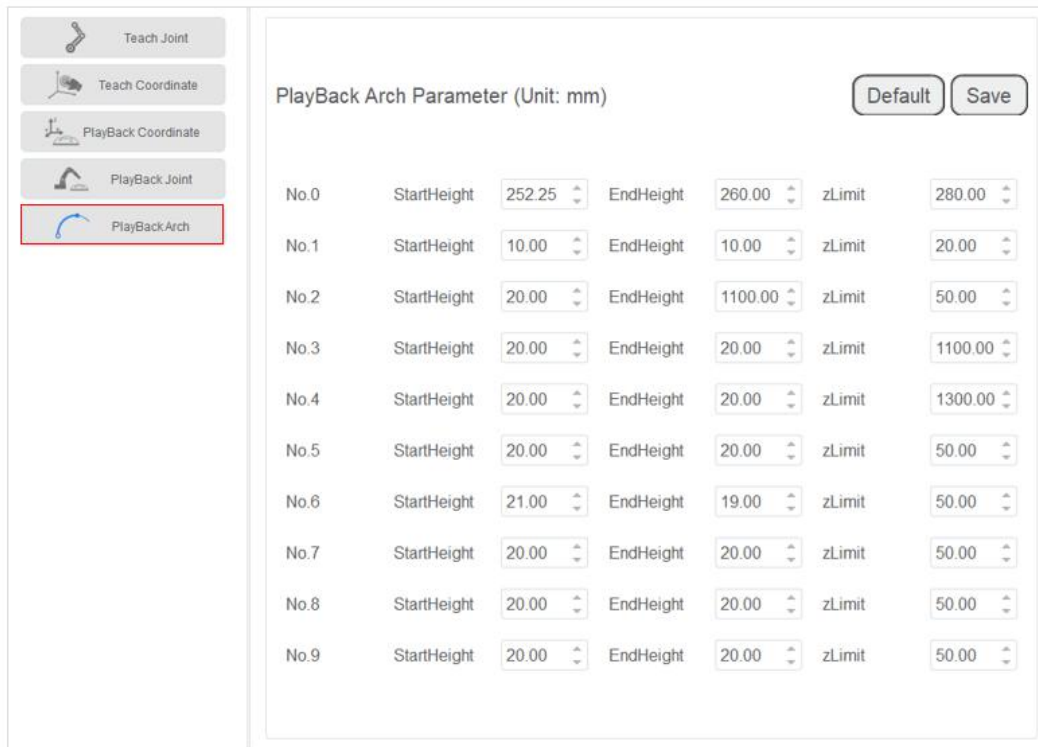



Figure 3.42 Jump parameters

### 3.7.7 RobotSetting

#### 3.7.7.1 Zero Calibration

After some parts (motors, reduction gear units) of the robot have been replaced or the robot has been hit, the origin of the robot will be changed. You need to reset the origin.

**Step 1** Click  > **Parameter** > **RobotSetting** > **Zero Calibration** to enter Zero Calibration interface, as shown in Figure 3.43.

Adjust each axis of M1 to the mechanical homing point according to the prompts.

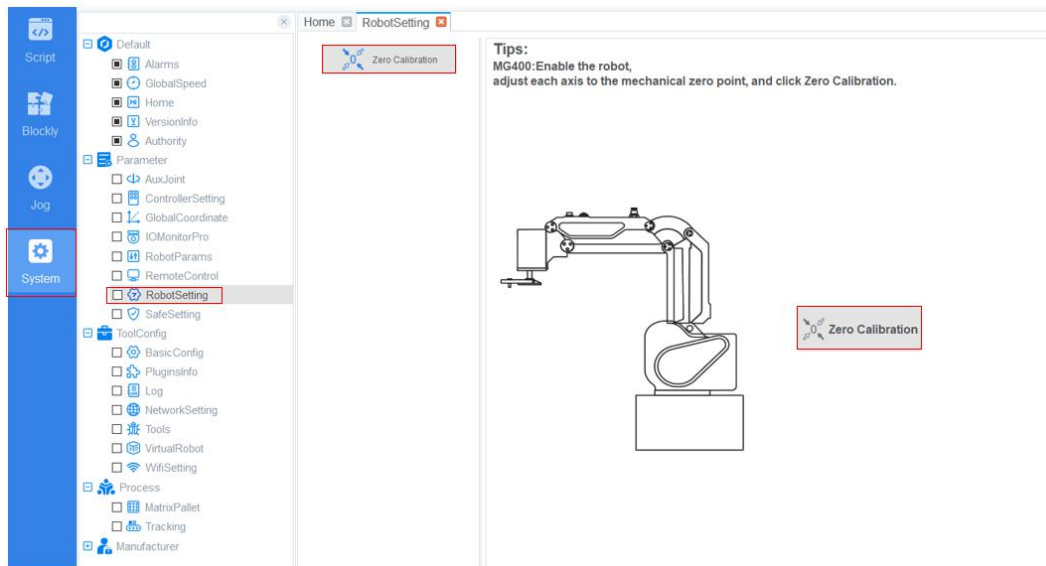


Figure 3.43 Zero Calibration

- Step 2** Click the **Zero Calibration**.
- Step 3** Click **Yes** in the current prompt window.



Figure 3.44 Confirm the zero calibration

### 3.7.8 SafeSetting

#### 3.7.8.1 Collision Detection

Collision detection is mainly used for reducing the impact on the robot arm, to avoid damage to the robot arm or external equipment. If the collision detection is activated, the robot arm will stop running automatically when the robot arm hits an obstacle.

You can enable collision detection function on the **Parameter > SafeSetting > Collision Detection** page and set the collision level. Meanwhile, you can select **Automatic restart 5 seconds after collision**, namely, when the robot arm stops for five seconds after hitting an obstacle, you can drag the robot to a safe position.

There are five collision levels to select. The higher the level is, the less force the robot arm needs to stop after collision detection.



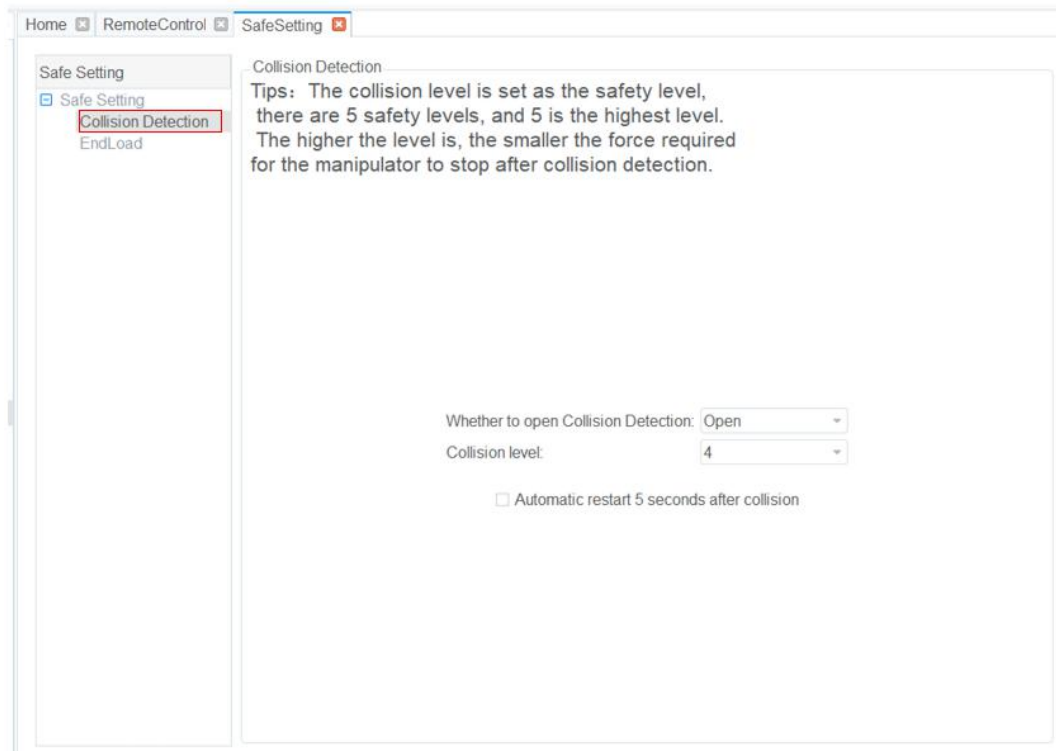


Figure 3.45 Collision detection

### 3.7.8.2 End load

To ensure optimum robot performance, it is important to make sure the load and inertia of the end effector are within the maximum range for the robot.

The weight of load includes weight of the end effector and work piece. You can set the load in **Parameter > SafeSetting > End load**. Or you can also set it when enabling the robot motor, as shown in Figure 3.46.

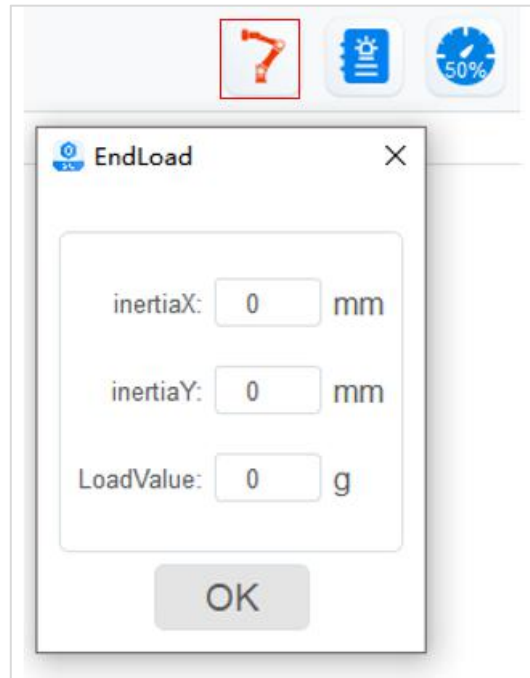


Figure 3.46 End load

## 3.8 ToolConfig

### 3.8.1 BasicConfig

You can select languages on the **ToolConfig > BasicConfig > Language** page. Also, you can modify the password on the **ToolConfig > BasicConfig > UserMode** page.

### 3.8.2 PluginsInfo

User can check the plug information on this page, including author, version, etc. The details will not be described in this topic.

### 3.8.3 Log

You can understand the historical operation of the robot by viewing the log. The log can be screened according to three types of logs: user operation, control error and servo error. Click **Reset** to clear the log.

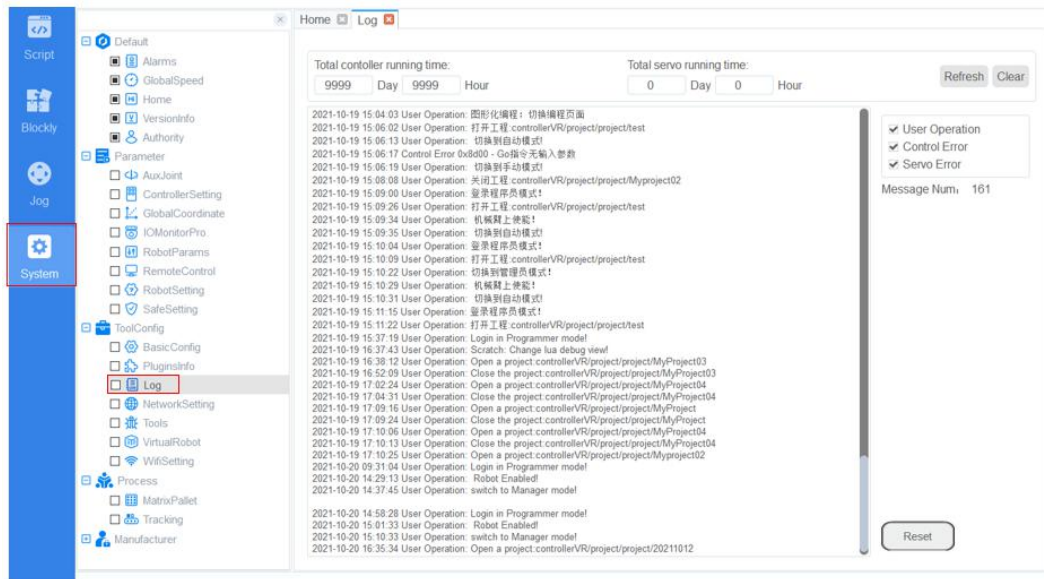



Figure 3.47 Log

### 3.8.4 Network Service

MG400 can be communicated with external equipment by the **Ethernet2** interface which supports TCP, UDP and Modbus protocols. The default IP address is **192.168.2.6**. In real applications, if the TCP or UDP protocol is used, MG400 can be a client or a server based on site requirements; if the Modbus protocol is used, MG400 only can be the Modbus slave, and the external equipment is the master.

You can modify the IP address on the  > **ToolConfig** > **NetworkSetting** page, as shown in Figure 3.48. The IP address of MG400 must be in the same network segment of the external equipment without conflict.

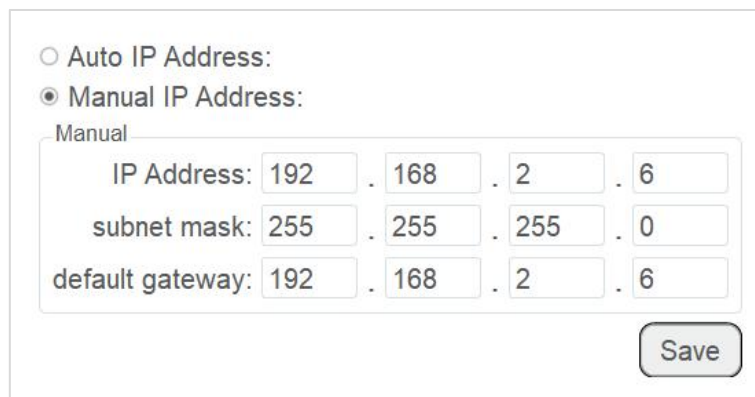


Figure 3.48 IP address setting

If MG400 connects to the external equipment directly, with a router or with a

switchboard, please select **Manual IP Address** and modify **IP Address**, **subnet mask**, **default gateway**, and then click **Save**.

### NOTICE

Please **DO NOT** insert the network cable into the WAN interface when using a router for the connection.

### 3.8.5 Tools

DobotSCStudio supports serial port debugging, TCP/UDP debugging and Modbus debugging for user. The details on how to use it will not be described in this topic.

### 3.8.6 VirtualRobot

When user jogs or runs a robot, the virtual simulation interface can be used to view the robot movement in real time.

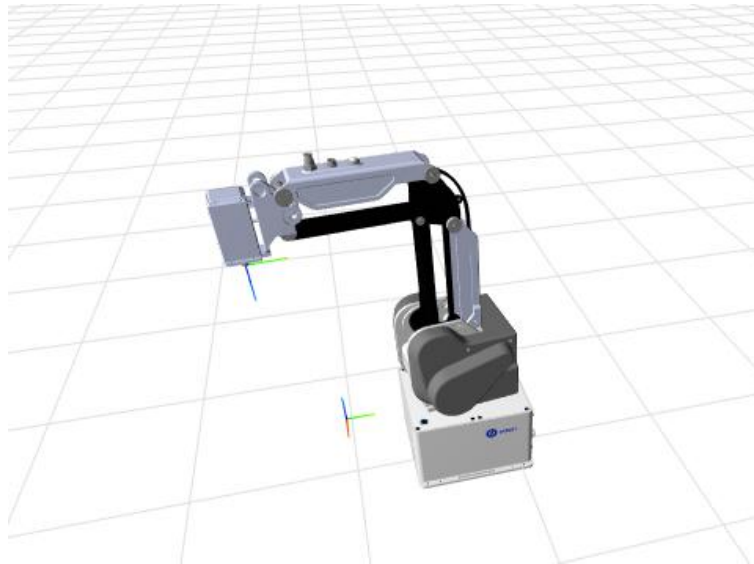



Figure 3.49 Virtual simulation

### 3.8.7 WiFi Setting

The robot system can be communicated with external equipment by the WiFi module. You can modify the WiFi name and password on the  > **ToolConfig** > **WiFiSetting** page and then restart the controller to make it effective. The default password is **1234567890**.

Wifi

SSID:

Password:

Figure 3.50 WiFi setting

## 4. Program Language

SC series controller encapsulates the robot dedicated API commands for programming with Lua language. This section describes commonly used commands for reference.

### 4.1 Arithmetic Operators

Table 4.1 Arithmetic operator

Command	Description
+	Addition
-	Subtraction
*	Multiplication
/	Floating point division
//	Floor division
%	Remainder
^	Exponentiation
&	And operator
	OR operator
~	XOR operator
<<	Left shift operator
>>	Right shift operator

### 4.2 Relational Operator

Table 4.2 Relational Operator

Command	Description
==	Equal
~=	Not equal
<=	Equal or less than
>=	Equal or greater than
<	Less than
>	Greater than

### 4.3 Logical Operators

Table 4.3 Logical operator

Command	Description
or	Logical OR operator
not	Logical NOT operator
and	Logical AND operator

## 4.4 General Keywords

Table 4.4 General keyword

Command	Description
break	Break out of a loop
local	Define a local variable, which is available in the current script
nil	Null
return	Return a value
enter	Line feed

## 4.5 General Symbol

Table 4.5 General symbol

Command	Description
#	Get the length of the array table

## 4.6 Processing Control Commands

Table 4.6 Processing control command

Command	Description
if...then...elseif...then...else...end	Conditional instruction (if)
while...do...end	Loop instruction (while)
for...do...end	Loop instruction (for)
repeat... until()	Loop instruction (repeat)

## 4.7 Global Variable

The robot global variables can be defined in the **global.lua** file, including global functions, global points, and global variables.

- Global function:

```
function exam()
    print("This is an example")
end
```

- Global point:

Define a Cartesian coordinate point, the User and Tool coordinate systems are both default coordinate systems.

```
P = { armOrientation = "right", coordinate = {10,10,10,0}, tool = 0, user = 0}
```

- Define a joint coordinate point

```
P = {joint = {20,10,22,85}}
```

- Global variable

```
flag = 0
```

## 4.8 Motion Commands

Table 4.7 Motion commands

Command	Description
MovJ	Point to Point, the target point is Cartesian point
JointMovJ	Point to point, the target point is Joint point
MovL	Linear Movement, the target point is Cartesian point
Jump	Jump Movement. The jump parameters can be set in this command
	Jump Movement. The jump parameters are called by Arch index
RelMovL	Move to the Cartesian offset position in a straight line
RelMovJ	Move to the Cartesian offset position in a point-to-point mode
MovLIO	Linear movement in parallel with output
MovJIO	Point to point movement in parallel with output
Arc	Arc movement
Circle	Circle movement

### NOTICE

Optional parameters for each motion command can be set individually



Table 4.8 MovJ command


Function	MovJ(P)
	local Option={CP=1, SpeedJ=50, AccJ=20} MovJ(P, Option)
Description	Point to Point, the target point is Cartesian point
Parameter	Required parameter: P, Indicate target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
	Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedJ: Velocity rate. Value range: 1~100</li> <li>• AccJ: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.9 JointMovJ command


Function	JointMovJ(P)
	local Option={CP=1, SpeedJ=50, AccJ=20} local P={joint={J1,J2,J3,J4}} JointMovJ(P, Option)
Description	Point to point, the target point is Joint point
Parameter	Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only joint point is supported.
	Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedJ: Velocity rate. Value range: 1~100</li> <li>• AccJ: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.10 MovL command

Function	MovL(P)
	local Option={CP=1, SpeedL=50, AccL=20} MovL(P, Option)
Description	Linear Movement, the target point is Cartesian point
Parameter	Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.


	Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> </ul>
--	---

Table 4.11 Arc command


<b>Function</b>	Arc(P1, P2) local Option={CP=1, SpeedL=50, AccL=20} Arc(P1, P2, Option)
<b>Description</b>	Arc movement. This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory
<b>Parameter</b>	Required parameter : <ul style="list-style-type: none"> <li>• P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> <li>• P2, End point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> </ul> Optional parameter : {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.12 Jump command

<b>Function</b>	local Option={SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20} Jump(P, Option)
<b>Description</b>	Jump Movement. The jump parameters can be set in this command.
<b>Parameter</b>	Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported. Optional parameter: {SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20} <ul style="list-style-type: none"> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> <li>• Start: Lifting height(h1).</li> <li>• ZLimit: Maximum lifting height(z_limit). The height of the starting point and target point</li> </ul>

	cannot exceed ZLimit; otherwise, an error alarm is triggered. <ul style="list-style-type: none"> <li>• End: Dropping height(h2).</li> </ul>
--	---

Table 4.13 Jump command

<b>Function</b>	local Option={SpeedL=50, AccL=20, Arch=1} Jump(P, Option)
<b>Description</b>	Jump Movement. The jump parameters are called by Arch index
<b>Parameter</b>	Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported. Optional parameter: {SpeedL=50, AccL=20, Start=10, Arch=1} <ul style="list-style-type: none"> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> <li>• Arch: Arch index. Value range: 0~9. Please set Jump parameters on the System &gt; Parameters &gt; RobotParams &gt; PlayBackArch page.</li> </ul>

Table 4.14 Circle command


<b>Function</b>	Circle(P1, P2, Count) local Option={CP=1, SpeedL=50, AccL=20} Circle(P1, P2, Count, Option)
<b>Description</b>	Move from the current position to a target position in a circular interpolated mode under the Cartesian coordinate system  This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory
<b>Parameter</b>	Required parameter: <ul style="list-style-type: none"> <li>• P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> <li>• P2, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> <li>• Count, Number of circles.</li> </ul> Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.15 RelMovJ command


Function	local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR} RelMovJ(Offset)
	local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR} local Option={CP=1, SpeedJ=50, AccJ=20} RelMovJ(Offset, Option)
Description	Move to the Cartesian offset position in a point-to-point mode
Parameter	<p>Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR}, X, Y, Z, R axes offset in the Cartesian coordinate system.</p> <p>Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to inset the command with optional parameters.</p> <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedJ: Velocity rate. Value range: 1~100</li> <li>• AccJ: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.16 RelMovL command


Function	local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR} RelMovL(Offset)
	local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR} local Option={CP=1, SpeedL=50, AccL=20} RelMovL(Offset, Option)
Description	Move to the Cartesian offset position in a straight line
Parameter	<p>Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR}, X, Y, Z, R axes offset in the Cartesian coordinate system.</p> <p>Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to inset the command with optional parameters.</p> <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.17 MovLIO command

Function	local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...} MovLIO(P, IO)
	local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}



	local Option={CP=1, SpeedL=50, AccL=20} MovLIO(P, IO, Option)
Description	Linear movement in parallel with output . Multiple digital output ports can be set
Parameter	Required parameter: <ul style="list-style-type: none"> <li>• P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> <li>• {Mode, Distance, Index, Status}, {Mode, Distance, Index, Status}... Multiple digital output ports can be set.                         <ul style="list-style-type: none"> <li>■ Mode: Set Distance mode. 0: Distance is a percentage; 1: Distance from the starting point, or from the target point.</li> <li>■ Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.</li> <li>■ Index: Digital output port. Value range: 1~18</li> <li>■ Status: Status of the digital output port. Value range: 0 or 1</li> </ul> </li> </ul> Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedL: Velocity rate. Value range: 1~100</li> <li>• AccL: Acceleration rate. Value range: 1~100</li> </ul>

Table 4.18 MovJIO

Function	local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...} MovJIO(P, IO)
	local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...} local Option={CP=1, SpeedJ=50, AccJ=20} MovJIO(P, IO, Option)
Description	Point to point movement in parallel with output. Multiple digital output ports can be set
Parameter	Required parameter: <ul style="list-style-type: none"> <li>• P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> <li>• {Mode, Distance, Index, Status}, {Mode, Distance, Index, Status}... Multiple digital output ports can be set.                         <ul style="list-style-type: none"> <li>■ Mode: Set Distance mode. 0: Distance is a percentage; 1: Distance from the starting point, or from the target point.</li> </ul> </li> </ul>

	<ul style="list-style-type: none"> <li>■ Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.</li> <li>■ Index: Digital output port. Value range: 1~18</li> <li>■ Status: Status of the digital output port. Value range: 0 or 1</li> </ul> <p>Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters.</p> <ul style="list-style-type: none"> <li>• CP: Continuous path rate. Value range: 0~100</li> <li>• SpeedJ: Velocity rate. Value range: 1~100</li> <li>• AccJ: Acceleration rate. Value range: 1~100</li> </ul>
--	---

## 4.9 Motion Parameter Commands

Table 4.19 Motion parameter commands

Command	Description
AccJ	Set the joint acceleration rate. This command is valid only when the motion mode is MovJ, MovJIO, or JointMovJ
AccL	Set the Cartesian acceleration rate. This command is valid only when the motion mode is MovL, MovLIO, Jump, Arc, Circle
SpeedJ	Set the joint velocity rate. This command is valid only when the motion mode is MovJ, MovJIO, or JointMovJ
SpeedL	Set the Cartesian velocity rate. This command is valid only when the motion mode is MovL, MovLIO, Jump, Arc, Circle
CP	Set the continuous path rate
Sync	Whether to stop at this point
SetPayload	Set payload, X-axis offset, Y-axis offset and servo index

Table 4.20 AccJ command

Function	AccJ(R)
Description	Set the joint acceleration rate. This command is valid only when the motion mode is MovJ, MovJIO, or JointMovJ
Parameter	Required parameter: Acceleration rate. Value range: 1~100

Table 4.21 AccL command

Function	AccL(R)
Description	Set the Cartesian acceleration rate. This command is valid only when the motion mode is MovL, MovLIO, Jump, Arc, Circle
Parameter	Acceleration rate. Value range: 1~100

Table 4.22 SpeedJ command

Function	SpeedJ(R)
Description	Set the joint velocity rate. This command is valid only when the motion mode is MovJ, MovJIO, or JointMovJ
Parameter	Velocity rate. Value range: 1~100

Table 4.23 SpeedL command

Function	SpeedL(R)
Description	Set the Cartesian velocity rate. This command is valid only when the motion mode is MovL, MovLIO, Jump, Arc, Circle
Parameter	Velocity rate. Value range: 1~100

Table 4.24 CP command

Function	CP(R)
Description	Set the continuous path rate. CP means when the robot arm passes through the middle point from the starting point to the end point, whether it transitions through the middle point in a rectangular way or in a curve way. This command is invalid when the motion mode is Jump
Parameter	Continuous path rate. Value range: 0~100

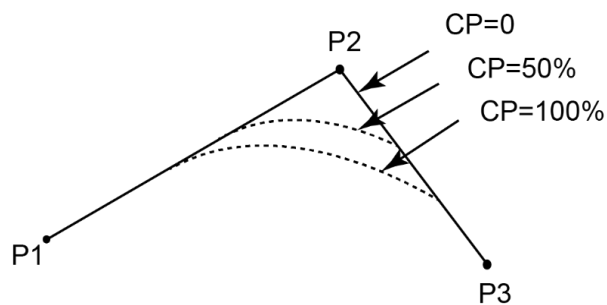


Figure 4.1 Continuous path

Table 4.25 Sync command

Function	Sync()
Description	Whether to stop at this point
Parameter	Null

Table 4.26 SetPayload command

Function	SetPayload(payload, {x, y}, index)
Description	Set payload, X-axis offset, Y-axis offset and servo index
Parameter	Required parameter: <ul style="list-style-type: none"> <li>• payload: Payload. Value range: 0~750. Unit: g</li> <li>• {x,y}: Offset in X-axis and Y-axis</li> </ul> Optional parameter: index, servo parameter index. The default value range is 1~10.

## 4.10 Input/output Commands

Table 4.27 Input/output command

Command	Description
DI	Get the status of the digital input port
DO	Set the status of the digital output port (Queue command)
DOInstant	Set the status of digital output port (Immediate command)

### NOTE

Dobot MG400 supports two kinds of commands: immediate command and queue command:

- Immediate command: MG400 will process the command once the command is received regardless of whether the controller is processing other commands.
- Queue command: When MG400 receives a command, this command will be pressed into the internal command queue. MG400 will execute commands in the order in which the commands were pressed into the queue.

Table 4.28 DI command

Function	DI(index)
Description	Get the status of the digital input port



Parameter	Index: Digital input index. Value range: 1~18
Return	<ul style="list-style-type: none"> <li>When an index is set in the DI function, DI(index) returns the status (ON/OFF) of this specified input port</li> <li>When there is no index in the DI function, DI() returns the status of all the input ports, which are saved in a table</li> </ul> <p>For example, local di= DI(), the saving format is {num = 24 value = {0x55, 0xAA, 0x52}}, you can obtain the status of the specified input port with di.num and di.value[n]</p>

Table 4.29 DO command (Queue command)

Function	<i>DO(index, ON   OFF)</i>
Description	Set the status of digital output port (Queue command)
Parameter	Index: Digital output port. Value range: 1~18 <ul style="list-style-type: none"> <li>ON/OFF: Status of the digital output port.</li> </ul>

Table 4.30 DOInstant command

Function	DOInstant(Index,ON/OFF)
Description	Set the status of digital output port (Immediate command)
Parameter	<ul style="list-style-type: none"> <li>Index: Digital output port. Value range: 1~18</li> <li>ON/OFF: Status of the digital output port</li> </ul>

## 4.11 Program Managing Commands

Table 4.31 Program managing commands

Command	Description
Wait	Set the delay time for robot motion commands
Sleep	Set the delay time for all commands
Pause	Pause the running program
ResetElapsedTime	Start timing
ElapsedTime	Stop timing
System	Get the current time

Table 4.32 Wait command

Function	Wait( <i>time</i> )
Description	Set the delay time for robot motion commands
Parameter	time: Delay time. Unit: ms

Table 4.33 Sleep command

Function	Sleep( <i>time</i> )
Description	Set the delay time for all commands
Parameter	time: Delay time. Unit: ms

Table 4.34 Pause command

Function	Pause()
Description	Pause the running program. When the program runs to this command, robot pauses running and you need to click <b>Resume</b> on the Software to recover the running.
Parameter	Null

Table 4.35 Start timing command

Function	ResetElapsedTime()
Description	Start timing after all commands before this command are executed completely. Use in conjunction with ElapsedTime() command
Parameter	Null

Table 4.36 Stop timing command

Function	ElapsedTime()
Description	Stop timing and return the time difference. Use in conjunction with ResetElapsedTime() command
Parameter	Null
Return	Time difference. Unit: ms

Table 4.37 Get current time command

Function	Systemtime()
Description	Get the current time
Parameter	Null

## 4.12 Pose Getting Command

Table 4.38 Pose commands

Command	Description
GetPose	Get Cartesian coordinates
GetAngle	Get Joint coordinates
RelPoint	Cartesian point offset
RelJoint	Joint point offset
Cartesian Point	Define a Cartesian point
Joint Point	Define a joint point

Table 4.39 Pose command (1)

Function	GetPose()
Description	Get the current pose of the robot under the Cartesian coordinate system If you have set the User or Tool coordinate system, the current pose is under the current User or Tool coordinate system
Parameter	Null
Return	Cartesian coordinate of the current pose

Table 4.40 Pose command (2)

Function	GetAngle()
Description	Get the current pose of the robot under the Joint coordinate system
Parameter	Null
Return	Joint coordinate of the current pose

Table 4.41 RelPoint command

Function	local Offset={OffsetX, OffsetY, OffsetZ, OffsetR} RelPoint(P, Offset)
Description	Set the X, Y, Z,R axes offset under the Cartesian coordinate system to return a new Cartesian coordinate point. The robot can move to this point in all motion commands except JointMovJ
Parameter	<ul style="list-style-type: none"> <li>P, Indicate the current Cartesian point, which is user-defined or obtained from the points list. Only Cartesian point is supported.</li> <li>{OffsetX, OffsetY, OffsetZ, OffsetR}: X, Y, Z, R axes offset in the Cartesian coordinate system.</li> </ul>
Return	Cartesian point

Table 4.42 RelJoint command

Function	local Offset={Offset1, Offset2, Offset3, Offset4} RelJoint(P, Offset)
Description	Set the joint offset in the Joint coordinate system to return a new joint point. The robot can move to this point only in JointMovJ command
Parameter	P, Indicate the current joint point, which is user-defined or obtained from the points list. Only joint point is supported.  {Offset1, Offset2, Offset3, Offset4}: J1 - J4 axes offset.
Return	Joint point

Table 4.43 local P command

Function	local P={coordinate = {x,y,z,r}, tool = 0, user = 0}
Description	Define a Cartesian point
Parameter	{x,y,z,r}: X, Y, Z, R axes coordinates.  tool: Tool coordinate system index. Value range: 0~9  user: User coordinate system index. Value range: 0~9

Table 4.44 local P command

Function	local P={joint= {j1,j2,j3,j4}}
Description	Define a joint point
Parameter	{j1,j2,j3,j4}, J1-J4 axes coordinates

## 4.13 TCP

Table 4.45 TCP commands

Command	Description
TCPCreate	Create a TCP network
TCPStart	Establish TCP connection
TCPRead	Receive data from a client
TCPWrite	Sends data to a client
TCPDestroy	Release a TCP network

Table 4.46 Create TCP command

Function	Err, Socket = TCPCreate(IsServer, IP, Port)
Description	Create a TCP network Only support a single connection
Parameter	<ul style="list-style-type: none"> <li>IsServer: Whether to create a server. false: Create a client; true: Create a server</li> <li>IP: IP address of the server, which is in the same network segment of the client without conflict</li> <li>Port: Server port. When the robot is set as a server, port cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail</li> </ul>
Return	Err: <ul style="list-style-type: none"> <li>0: TCP network is created successfully</li> <li>1: TCP network fails to be created</li> </ul> Socket: Socket object

Table 4.47 TCP connection command

Function	TCPStart(Socket, Timeout)
Description	Establish TCP connection
Parameter	Required parameter: <ul style="list-style-type: none"> <li>Socket: Socket object.</li> <li>Timeout: Wait timeout. Unit: s. If Timeout is 0, the connection is still waiting. If not, after exceeding the timeout, the connection is exited.</li> </ul>

Return	<ul style="list-style-type: none"> <li>• 0: TCP connection is successful</li> <li>• 1: Input parameters are incorrect</li> <li>• 2: Socket object is not found</li> <li>• 3: Timeout setting is incorrect</li> <li>• 4: If the robot is set as a client, it indicates that the connection is wrong. If the robot is set as a server, it indicates that receiving data is wrong</li> </ul>
--------	---

Table 4.48 Receive data command

Function	Err, RecBuf = TCPRead(Socket, Timeout, Type)
Description	Robot as a client receives data from a server or as a server receives data from a client
Parameter	<ul style="list-style-type: none"> <li>• Socket: socket object</li> <li>• Timeout: Receiving timeout. Unit: s. If timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete</li> <li>• Type: Buffer type. If type is not set, the buffer format of RecBuf is a table. If type is set to string, the buffer format is a string</li> </ul>
Return	Err: <ul style="list-style-type: none"> <li>• 0: Receiving data is successful</li> <li>• 1: Data fails to be received</li> </ul> Recbuf: Data buffer

Table 4.49 Send data command

Function	TCPWrite(Socket, Buf, Timeout)
Description	Robot as a client sends data to a server or as a server sends data to a client
Parameter	<ul style="list-style-type: none"> <li>• Socket: Socket object.</li> <li>• Buf: Data sent by the robot.</li> <li>• Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.</li> </ul>
Return	0: Sending data is successful 1: Data fails to be sent

Table 4.50 Release TCP network command

Function	TCPDestroy(Socket)
Description	Release a TCP network
Parameter	Socket: Socket object
Return	0: Releasing TCP is successful 1: TCP fails to be released

## 4.14 UDP

Table 4.51 UDP commands

Command	Description
TCPCreate	Create a UDP network
UDPRead	Receive data from a client
UDPWrite	Send data to a client

Table 4.52 Create UDP network command

Function	Err, Socket = UDPCreate(IsServer, IP, Port)
Description	Create a UDP network Only a single connection is supported
Parameter	<ul style="list-style-type: none"> <li>IsServer: Whether to create a server. false: Create a client; true: Create a server.</li> <li>IP: IP address of the server, which is in the same network segment of the client without conflict.</li> <li>Port: Server port. When the robot is set as a server, port cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.</li> </ul>
Return	Err: <ul style="list-style-type: none"> <li>0: The UDP network is created successfully</li> <li>1: The UDP network fails to be created</li> </ul> Socket: Socket object

Table 4.53 Receive data command

Function	Err, RecBuf = UDPRead(Socket, Timeout, Type)
----------	--

Description	Robot as a client receives data from a server or as a server receives data from a client
Parameter	<ul style="list-style-type: none"> <li>• Socket: Socket object.</li> <li>• Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading.</li> <li>• Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.</li> <li>• Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string</li> </ul>
Return	Err: <ul style="list-style-type: none"> <li>• 0: Receiving data is successful</li> <li>• 1: Data fails to be received</li> </ul> Recbuf: Data buffer

Table 4.54 Send data command

Function	UDPWrite(Socket, Buf, Timeout)
Description	Robot as a client sends data to a server or as a server sends data to a client
Parameter	Socket: Socket object Buf: Data sent by the robot Timeout: Timeout. Unit: s. If timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete
Return	0: Sending data is successful 1: Data fails to be sent

## 4.15 Modbus

### 4.15.1 Description on Modbus Register

Modbus protocol is a serial communication protocol. MG400 can communicate with external equipment by this protocol. Here, External equipment such as a PLC is set as the Modbus master, and MG400 is set as the slave.

Modbus data is most often read and written as registers. Based on our robot memory space, we also define four types of registers: coil, discrete input, input, and holding registers for data interaction between the external equipment and MG400. Each register has 4096 addresses. For details, please see as follows.

- Coil register



Table 4.55 Coil register description

Coil register address (e.g.: PLC)	Coil register address (MG400)	Data type	Description
00001	0	Bit	Start
00002	1	Bit	Pause
00003	2	Bit	Continue
00004	3	Bit	Stop
00005	4	Bit	Emergency stop
00006	5	Bit	Clear alarm
00007~0999	6~998	Bit	Reserved
01001~04096	999~4095	Bit	User-defined

- Discrete input register

Table 4.56 Description on discrete input register

Discrete input register address (e.g. PLC)	Discrete input register address (MG400)	Data type	Description
10001	0	Bit	Automated exit
10002	1	Bit	Ready state
10003	2	Bit	Paused state
10004	3	Bit	Running state
10005	4	Bit	Alarm state
10006~10999	5~998	Bit	Reserved
11000~14096	999~4095	Bit	User-defined

- Input register

Table 4.57 Description on input register

Input register address (e.g. PLC)	Input register address (MG400)	Data type	Description
30001~34096	0-4095	Byte	Reserved

- Holding register

Table 4.58 Description on holding register

Holding register address (e.g.: PLC)	Holding register address (MG400)	Data type	Description
40001~41000	0~999	Byte	Reserved
41001~44096	1000~4095	Byte	User-defined

#### 4.15.2 Command Description

Table 4.59 Modbus commands

Command	Description
GetCoils	Read the value from Modbus slave coil register address
SetCoils	Set the coil register in the Modbus slave
GetInBits	Read the value from the Modbus slave discrete register address
GetInRegs	Read the input register value with the specified data type from the Modbus slave
GetHoldRegs	Read the holding register value from the Modbus slave
SetHoldRegs	Set the holding register in the Modbus slave

Table 4.60 Read coil register command

Function	GetCoils(Addr, Count)
Description	Read the value from Modbus slave coil register address
Parameter	Addr: Starting address of the coils. Value range: 0-4095 Count: Number of the coils to read. Value range: 0 to 4096- Addr
Return	Return a table, each with the value 1 or 0, where the first value in the table corresponds to the coil value at the starting address

Table 4.61 Set coil register command

Function	SetCoils(Addr, Count, Table)
Description	Set the coil register in the Modbus slave This command is not supported when the coil register address is from 0 to 5
Parameter	Addr: Starting address of the coils to set. Value range: 6 - 4095 Count: Number of the coils to set. Value range: 0 to 4096- Addr Table: The values written into the coil register: Data type: bit

Table 4.62 Read discrete input register command

Function	GetInBits(Addr, Count)
Description	Read the value from the Modbus slave discrete register address
Parameter	Addr: Starting address of the discrete inputs to read. Value range: 0-4095 Count: Number of the discrete inputs to read. Value range: 0 to 4096- Addr
Return	Return a table, each with the value 1 or 0, where the first value in the table corresponds to the discrete value at the starting address. Data type: bit

Table 4.63 Read input register command

Function	GetInRegs(Addr, Count, Type)
Description	Read the input register value with the specified data type from the Modbus slave
Parameter	Addr: Starting address of the input registers. Value range: 0 - 4095 Count: Number of the input registers to read. Value range: 0 ~ 4096-addr Type: Data type <ul style="list-style-type: none"> <li>• Empty: Read 16-bit unsigned integer ( two bytes, occupy one register)</li> <li>• “U16”: Read 16-bit unsigned integer ( two bytes, occupy one register)</li> <li>• “U32”: Read 32-bit unsigned integer (four bytes, occupy two registers)</li> <li>• “F32”: Read 32-bit single-precision floating-point number (four bytes, occupy two registers)</li> <li>• “F64”: Read 64-bit double-precision floating-point number (eight bytes, occupy four registers)</li> </ul>
Return	Return a table, the first value in the table corresponds to the input register value at the starting address

Table 4.64 Read holding register command

Function	GetHoldRegs(Addr, Count, Type)
Description	Read the holding register value from the Modbus slave according to the specified data type

<b>Parameter</b>	Addr: Starting address of the holding registers. Value range: 0 - 4095 Count: Number of the holding registers to read. Value range: 0 to 4096-addr Type: Datatype <ul style="list-style-type: none"> <li>• Empty: Read 16-bit unsigned integer ( two bytes, occupy one register)</li> <li>• “U16”: Read 16-bit unsigned integer ( two bytes, occupy one register)</li> <li>• “U32”: Read 32-bit unsigned integer (four bytes, occupy two registers)</li> <li>• “F32”: Read 32-bit single-precision floating-point number (four bytes, occupy two registers)</li> <li>• “F64”: Read 64-bit double-precision floating-point number (eight bytes, occupy four registers)</li> </ul>
<b>Return</b>	Return a table, the first value in the table corresponds to the input register value at the starting address

Table 4.65 Set holding register command

<b>Function</b>	SetHoldRegs(Addr, Count, Table, Type)
<b>Description</b>	Set the holding register in the Modbus slave
<b>Parameter</b>	Addr: Starting address of the holding registers to set. Value range: 0 - 4095 Count: Number of the holding registers to set. Value range: 0 to 4096-addr Table: Holding register value, stored in a table Type: Datatype <ul style="list-style-type: none"> <li>• Empty: Read 16-bit unsigned integer ( two bytes, occupy one register)</li> <li>• “U16”: Set 16-bit unsigned integer ( two bytes, occupy one register)</li> <li>• “U32”: Set 32-bit unsigned integer (four bytes, occupy two registers)</li> <li>• “F32”: Set 32-bit single-precision floating-point number (four bytes, occupy two registers)</li> <li>• “F64”: Set 64-bit double-precision floating-point number (eight bytes, occupy four registers)</li> </ul>

## 4.16 Conveyor Tracking

Table 4.66 Conveyor commands

Command	Description
CnvVison	Set conveyor number to create a tracing queue
GetCnvObject	Obtain status of the object
SetCnvPointOffset	Set X,Y axes offset under the set User coordinate system

Command	Description
SetCnvTimeCompensation	Set time compensation
SyncCnv	Synchronize the specified conveyor
StopSyncCnv	Stop synchronizing the conveyor

Table 4.67 CnvVison command

Function	CnvVison(CnvID)
Description	Set conveyor number to create a tracing queue
Parameter	CnvID, Conveyor number. Only support single conveyor
Return	0: No error 1: Error

Table 4.68 GetCnvObject command

Function	GetCnvObject(CnvID, ObjID)
Description	Obtain the information of the part on the conveyor to check whether the part is in the pickup area
Parameter	CnvID: Conveyor index. ObjID: Part index.
Return	Part status: Whether there is a part. Value range: true or false Part type Part coordinate (x,y,r)

Table 4.69 SetCnvPointOffset command

Function	SetCnvPointOffset(OffsetX,OffsetY)
Description	Set X,Y axes offset under the set User coordinate system
Parameter	OffsetX: X-axis offset. OffsetY: Y-axis offset.
Return	0: No error 1: Error

Table 4.70 SetCnvTimeCompensation command

Function	SetCnvTimeCompensation(Time)
Description	Set time compensation. This command is used for compensating the pick-up position offset in the moving direction of the conveyor which is caused by taking photos with a time delay
Parameter	Time, time-offset. Unit: ms
Return	0: No error 1: Error

Table 4.71 SyncCnv command

Function	SyncCnv(CnvID)
Description	Synchronize the specified conveyor. The motion commands used between SyncCnv( <i>CnvID</i> ) and StopSyncCnv( <i>CnvID</i> ) only support MovL command
Parameter	CnvID, Conveyor index
Return	0: No error 1: Error

Table 4.72 StopSyncCnv command

Function	StopSyncCnv(CnvID)
Description	Stop synchronizing the conveyor. The other commands following this command will not be executed until this command running is completed
Parameter	CnvID, Conveyor index
Return	0: No error 1: Error

## 4.17 Pallet

Table 4.73 Pallet commands

Command	Description
MatrixPallet	Instantiate matrix pallet
TeachPallet	Instantiate teaching pallet
SetPartIndex	Set the next stack index which is to be operated
GetPartIndex	Get the current operated stack index
SetLayerIndex	Set the next pallet layer index which is to be operated

Command	Description
GetLayerIndex	Get the current pallet layer index
Reset(Pallet)	Reset pallet
IsDone(Pallet)	Check whether the stack assembly or dismantling is complete
Release(Pallet)	Release palletizing instance
PalletMoveIn	The robot moves from the current position to the first stack position as the configured stack assembly path
PalletMoveOut	The robot moves from the current position to the transition point as the configured stack dismantling path

Table 4.74 MatrixPallet command


Function	Pallet = MatrixPallet (Index)
	local Option={IsUnstack= true, User= 1} Pallet = MatrixPallet (Index,ID, Option)
Description	Instantiate matrix pallet
Parameter	Required parameter: <ul style="list-style-type: none"> <li>Index: Matrix pallet index.</li> </ul> Optional parameter: {IsUnstack= true, User= 1}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode</li> <li>User: User coordinate system index. If not set, the default is User 0 coordinate system.</li> </ul>
Return	Matrix pallet object

Table 4.75 TeachPallet command

Function	Pallet = TeachPallet (Index, Option)
	local Option={IsUnstack= true, User= 1} Pallet = TeachPallet (Index,ID, Option)
Description	Instantiate teaching pallet


Parameter	Required parameter: <ul style="list-style-type: none"> <li>Index: Teaching pallet index.</li> </ul> Optional parameter: {IsUnstack= true, User= 1}. You can double-click  to insert the command with optional parameters. <ul style="list-style-type: none"> <li>IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode</li> <li>User: User coordinate system index. If not set, the default is User 0 coordinate system</li> </ul>
Return	Teaching pallet object

Table 4.76 SetPartIndex command

Function	SetPartIndex(Pallet, Index)
Description	Set the next stack index which is to be operated
Parameter	Pallet: Pallet object. Index: The next stack index. Initial value: 0

Table 4.77 GetPartIndex command

Function	GetPartIndex(Pallet)
Description	Get the current operated stack index
Parameter	Pallet, Pallet object
Return	The current operated stack index

Table 4.78 SetLayerIndex command

Function	SetLayerIndex(Pallet, Index)
Description	Set the next pallet layer index which is to be operated
Parameter	Pallet: Pallet object. Index: The next pallet layer index. Initial value: 0

Table 4.79 GetLayerIndex command

Function	GetLayerIndex(Pallet)
Description	Get the current pallet layer index
Parameter	Pallet, Pallet object



Return	The current pallet layer index
--------	--------------------------------

Table 4.80 Reset command

Function	Reset(Pallet)
Description	Reset pallet
Parameter	Pallet, Pallet object

Table 4.81 IsDone command

Function	IsDone(Pallet)
Description	Check whether the stack assembly or dismantling is complete
Parameter	Pallet, Pallet object
Return	true: Finished. false: Un-finished.

Table 4.82 Release command

Function	Release(Pallet)
Description	Release palletizing instance
Parameter	Pallet, Pallet object

Table 4.83 PalletMoveIn command

Function	PalletMoveIn(Pallet)
	local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20} PalletMoveIn(Pallet, Option)
Description	The robot moves from the current position to the first stack position as the configured stack assembly path



Parameter	<p>Required parameter: Pallet, Pallet object.</p> <p>Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click  to insert the command with optional parameters.</p> <ul style="list-style-type: none"> <li>• SpeedAB: Velocity rate when the robot moves from the transition point to the preparation point. Value range: 1~100</li> <li>• SpeedBC: Velocity rate when the robot moves from the preparation point to the first stack point. Value range: 1~100</li> <li>• AccAB: Acceleration rate when the robot moves from the transition point to the preparation point. Value range: 1~100</li> <li>• AccBC: Acceleration rate when the robot moves from the preparation point to the first stack point. Value range: 1~100</li> <li>• CP: Continuous path rate. Value range: 0~100</li> </ul>
-----------	---

Table 4.84 PalletMoveOut command

Function	PalletMoveOut(Pallet) local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20} PalletMoveOut(Pallet, Option)
Description	The robot moves from the current position to the transition point as the configured stack dismantling path
Parameter	<p>Required parameter: Pallet, Pallet object.</p> <p>Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click  to insert the command with optional parameters.</p> <ul style="list-style-type: none"> <li>• SpeedAB: Velocity rate when the robot moves from the preparation point to the transition point. Value range: 1~100</li> <li>• SpeedBC: Velocity rate when the robot moves from the first stack point to the preparation point. Value range: 1~100</li> <li>• AccAB: Acceleration rate when the robot moves from the preparation point to the transition point. Value range: 1~100</li> <li>• AccBC: Acceleration rate when the robot moves from the first stack point to the preparation point. Value range: 1~100</li> <li>• CP: Continuous path rate. Value range: 0~100</li> </ul>

## Appendix A Servo Alarm Description

ID	Level	Description	Solution
25376	0	Abnormalities in internal servo parameters	System error, please contact technical support engineer
21120	0	Programmable logic configuration faults	System error, please contact technical support engineer
29953	5	FPGA software version too low	Please contact technical support engineer
29954	5	Programmable logic interrupt fault	If connecting the power for many times, the alarm is still reported, please replace the drive
25377	5	Internal program exceptions	System error, please contact technical support engineer
21808	0	Parameter storage failure	Reset the parameter and power on again, or please contact technical support engineer
28962	0	Product matching faults	<ol style="list-style-type: none"> <li>1. Check whether the motor parameter matches the motor model in nameplate;</li> <li>2. Check whether the motor and driver match, otherwise, select the right motor and driver</li> </ol>
21574	0	Invalid servo ON command fault	System error, please contact technical support engineer
28964	0	Absolute position mode product matching fault	System error, please contact technical support engineer
25378	0	Repeated assignment of DI functions	<ol style="list-style-type: none"> <li>1. Check whether the same function is assigned to different DI's</li> <li>2. Confirm whether the corresponding MCU supports the assigned functionality</li> </ol>
25379	0	DO function allocation overrun	Check whether the motor and circuit are working properly, or contact technical support engineer
29488	0	Data in the motor encoder ROM is incorrectly checked or parameters are not stored	System error, please contact technical support engineer
8752	0	Hardware overcurrent	System error, please contact technical support engineer
8977	0	DQ axis current overflow fault	System error, please contact technical support engineer

ID	Level	Description	Solution
65288	0	FPGA system sampling operation timeout	System error, please contact technical support engineer
9024	0	Output shorted to ground	Please contact technical support engineer
13184	0	UVW phase sequence error	System error, please contact technical support engineer
33922	0	Flying Cars	Please contact technical support engineer
12816	0	Electrical over-voltage in the main circuit	System error, please contact technical support engineer
12832	0	Main circuit voltage undervoltage	System error, please contact technical support engineer
12592	0	Main circuit electrical shortage	Check the cable connection of power, otherwise, replace the driver
12576	0	Control of electrical undervoltage	System error, please contact technical support engineer
33920	0	Overspeed	System error, please contact technical support engineer
65296	0	Pulse output overspeed	System error, please contact technical support engineer
65282	0	Failure to identify angles	System error, please contact technical support engineer
9040	0	Drive overload	Replace the driver
29056	0	Motor overload	System error, please contact technical support engineer
28961	0	Overheating protection for blocked motors	Check whether the hardware is working properly, or contact technical support engineer
17168	0	Radiator overheating	Drop the environment temperature, or contact technical support engineer
29571	0	Encoder battery failure	Connect battery, or contact technical support engineer
29490	0	Encoder multi-turn count error	Replace the motor
29491	0	Encoder multi-turn count overflow	System error, please contact technical support engineer
29492	0	Encoder interference	System error, please contact technical support

ID	Level	Description	Solution
			engineer
29493	0	External encoder scale failure	System error, please contact technical support engineer
29494	0	Encoder data abnormalities	System error, please contact technical support engineer
29495	0	Encoder return checksum exception	System error, please contact technical support engineer
29496	0	Loss of encoder Z signal	System error, please contact technical support engineer
34321	0	Excessive position deviation	Check whether the motor is working properly, or contact technical support engineer
34322	0	Position command too large	System error, please contact technical support engineer
34323	0	Excessive deviation from fully closed-loop position	System error, please contact technical support engineer
25380	0	Electronic gear setting overrun	System error, please contact technical support engineer
25381	0	Wrong parameter setting for fully closed loop function	System error, please contact technical support engineer
25382	0	Software position upper and lower limits set incorrectly	System error, please contact technical support engineer
25383	0	Wrong home position offset setting	System error, please contact technical support engineer
30083	0	Loss of synchronization	System error, please contact technical support engineer
30081	0	Unburned XML configuration file	Burn the XML configuration file
65298	0	Network initialization failure	System error, please contact technical support engineer
30082	0	Sync cycle configuration error	System error, please contact technical support engineer
30084	0	Excessive synchronisation period error	System error, please contact technical support engineer
25384	0	Fault in crossover pulse output setting	System error, please contact technical support engineer

ID	Level	Description	Solution
65521	0	Zero return timeout fault	System error, please contact technical support engineer
29570	0	Encoder battery warning	Replace battery
21570	0	DI emergency brake	System error, please contact technical support engineer
12851	0	Motor overload warning	System error, please contact technical support engineer
12817	0	Brake resistor overload alarm	System error, please contact technical support engineer
25385	0	External braking resistor too small	System error, please contact technical support engineer
13105	0	Motor power cable disconnection	System error, please contact technical support engineer
25386	0	Change of parameters requires re-powering to take effect	Clear the alarm and power on again
30208	0	Frequent parameter storage	Check whether the upper computer is working normal, or contact technical support engineer
21571	0	Forward overtravel warning	System error, please contact technical support engineer
21572	0	Reverse overtravel warning	System error, please contact technical support engineer
29569	0	Internal failure of the encoder	System error, please contact technical support engineer
12597	0	Input phase failure warning	System error, please contact technical support engineer
65432	0	Zero return mode setting error	System error, please contact technical support engineer
65344	0	Parameter recognition failure	System error, please contact technical support engineer
21121	0	internal error	System error, please contact technical support engineer
29956	0	FPGA configuration error	System error, please contact technical support

ID	Level	Description	Solution
			engineer
51020	0	Driver board identification error	System error, please contact technical support engineer
29568	0	Encoder connection error	Check the cable connection of encoder, or contact technical support engineer
8992	0	Software overcurrent	System error, please contact technical support engineer
9088	0	Current zero point too large	System error, please contact technical support engineer
30080	0	EtherCAT communication failure	System error, please contact technical support engineer
33921	0	Excessive speed tracking error	System error, please contact technical support engineer
21120	0	STO Warning	System error, please contact technical support engineer
21569	0	Upper and lower board connection failure	System error, please contact technical support engineer
8980	0	Busbar overcurrent	System error, please contact technical support engineer
17169	0	Damaged or uninstalled temperature measuring resistors	System error, please contact technical support engineer
29572	0	Encoder Eeprom reading CRC fault	System error, please contact technical support engineer
12928	0	Servo and motor power matching faults	System error, please contact technical support engineer

## Appendix B Controller Alarm Description

ID	Level	Description	Solution
17	5	Inverse kinematics error with no solution	Reselect movement points
18	5	Inverse kinematics error with result out of working area	Reselect movement points
19	5	Duplicated data in JUMP or ARC or Circles instruction	Reselect movement points
20	5	Wrong input parameters for arc	Enter the correct parameters
21	5	The Start and the End is negative or the zLimit is below the start and end points	Enter the correct parameters
22	5	Wrong arm orientation switch	Reselect movement points
23	5	Plan point during linear motion out of working area	Reselect movement points
24	5	Plan point during circular arc motion out of working area	Reselect movement points
25	5	Wrong mode for motion instruction	Internal software error, restart or contact manufacturer
26	5	Wrong input parameters for speed	Input correct parameter
27	5	Wrong trajectory motion plan of continuous path	Input correct parameter
28	0	Wrong input parameters for circle	Input correct parameter
29	5	Plan point during circular circle motion out of working circle	Reselect movement points
30	5	Inching target position inaccessible	Reverse inch out of limit
32	5	Inverse kinematics singularity during moving	Reselect movement points
33	5	Inverse kinematics with no solution during moving	Reselect movement points
34	5	Inverse kinematics with result out of working area	Reselect movement points
48	5	Joint1 overspeed	Reset the speed or re-select the movement point away from the singularity
49	5	Joint2 overspeed	Reset the speed or re-select the movement point away from the singularity



ID	Level	Description	Solution
50	5	Joint3 overspeed	Reset the speed or re-select the movement point away from the singularity
51	5	Joint4 overspeed	Reset the speed or re-select the movement point away from the singularity
52	0	Joint1 position out of range	Internal error, restart or contact manufacturer
53	0	Joint2 position lag error	Internal error, restart or contact manufacturer
54	0	Joint3 position lag error	Internal error, restart or contact manufacturer
55	0	Joint4 position lag error	Internal error, restart or contact manufacturer
64	5	Joint1 exceeds positive limit	Reverse jog out of limit
65	5	Joint1 exceeds negative limit	Reverse jog out of limit
66	5	Joint2 exceeds positive limit	Reverse jog out of limit
67	5	Joint2 exceeds negative limit	Reverse jog out of limit
68	5	Joint3 exceeds positive limit	Reverse jog out of limit
69	5	Joint3 exceeds negative limit	Reverse jog out of limit
70	5	Joint4 exceeds positive limit	Reverse jog out of limit
71	5	Joint4 exceeds negative limit	Reverse jog out of limit
72	5	Parallelogram positive limit	Reverse jog out of limit
73	5	Parallelogram negative limit	Reverse jog out of limit
80	0	Joint1 lose step	Internal error, restart or contact manufacturer
81	0	Joint2 lose step	Internal error, restart or contact manufacturer
82	0	Joint3 lose step	Internal error, restart or contact manufacturer
83	0	Joint4 lose step	Internal error, restart or contact manufacturer
84	0	Algorithm timeout	Internal error, restart or contact manufacturer
85	0	Emergency button pressed	Release the emergency stop button
96	0	Joint1 drive alarm	Check if the communication of joint 1 is normal and then clear the error
97	0	Joint1 Servo power off	Re-enable joint 1
98	0	Joint2 drive alarm	Check if the communication of joint 2 is normal and then clear the error
99	0	Joint2 Servo power off	Re-enable joint 2
100	0	Joint3 drive alarm	Re-enable joint 3

ID	Level	Description	Solution
101	0	Joint3 Servo power off	Re-enable joint 3
102	0	Joint4 drive alarm	Re-enable joint 4
103	0	Joint4 drive power off	Re-enable joint 4
104	0	Robot homing failed	Home again
105	0	Robot Servo on failed	Check whether the hardware is normal and re-enable
106	0	Abnormal conveyor data	Please contact technical support engineer
107	0	Abnormal conveyor synchronization	Please contact technical support engineer
108	0	Conveyor conveyor encoder 1 is disconnected	Please contact technical support engineer
109	0	Conveyor conveyor encoder 2 is disconnected	Please contact technical support engineer
110	0	Encoder position error	Internal error, restart or contact manufacturer
112	0	Collision Detection	Keep away from the work area and continue to run
161	0	Error switching drag and drop mode	Internal error, restart or contact manufacturer
4096	5	Failed to open mechanical file	Check if the file location is correct and restart
8192	5	Failed to open project file	Check if the file location is correct and restart
8193	5	Failed to open program file	Check if the file location is correct and restart
8194	5	Failed to open global variable file	Check if the file location is correct and restart
8195	5	Failed to open teaching point file	Check if the file location is correct and restart
8196	5	Failed to start debugger process	Rerun debugger process
12288	5	Emergency stop detected	Power on again
12289	5	External emergency stop detected	Power on again
12290	0	The servo power board temperature is too high	Turn off the machine and let it cool for a period of time
33024	5	No input parameters for CP instruction	Enter the correct parameters
33025	5	Input parameters of CP instruction out of range	Enter the correct parameters
33280	5	No input parameters for Arch instruction	Please enter parameters
33281	5	Index parameter of Arch instruction out of range	Enter the correct parameters

ID	Level	Description	Solution
33282	5	Index parameter of Arch instruction not configured yet	Please set index parameters
33536	5	No input parameters for LimZ instruction	Please enter parameters
33537	5	Input parameters of LimZ instruction out of range	Enter the correct parameters
33792	5	No input parameters for Speed instruction	Please enter parameters
33793	5	Ratio parameter of Speed instruction out of range [1, 100]	Enter the correct parameters
34048	5	No input parameters for Accel instruction	Please enter parameters
34049	5	Ratio parameter of Accel instruction out of range [1, 100]	Enter the correct parameters
34304	5	No input parameters for Jerk instruction	Please enter parameters
34305	5	Ratio parameter of Jerk instruction out of range [1, 100]	Enter the correct parameters
34560	5	No input parameters for SpeedS instruction	Please enter parameters
34561	5	Ratio parameter of SpeedS instruction out of range [1, 100]	Enter the correct parameters
34816	5	No input parameters for SpeedR instruction	Please enter parameters
34817	5	Ratio parameter of SpeedR instruction out of range [1, 100]	Enter the correct parameters
35072	5	No input parameters for AccelS instruction	Please enter parameters
35073	5	Ratio parameter of AccelS instruction out of range [1, 100]	Please enter parameters
35328	5	No input parameters for AccelR instruction	Enter the correct parameters
35329	5	Ratio parameter of AccelR instruction out of range [1, 100]	Enter the correct parameters
35584	5	No input parameters for JerkS instruction	Please enter parameters
35585	5	Ratio parameter of JerkS instruction out of range [1, 100]	Enter the correct parameters
35840	5	No input parameters for JerkR instruction	Please enter parameters
35841	5	Ratio parameter of JerkR instruction out of range [1, 100]	Enter the correct parameters
36096	5	No input parameters for Go instruction	Please enter parameters

ID	Level	Description	Solution
36097	5	No motion point parameter for Go instruction	Please enter parameters
36098	5	Incorrect motion point for Go instruction	Enter the correct parameters
36099	5	Incorrect control parameter for Go instruction	Enter the correct parameters
36352	5	No input parameters for Move instruction	Please enter parameters
36353	5	No motion point parameter for Move instruction	Please enter parameters
36354	5	Incorrect motion point for Move instruction	Enter the correct parameters
36355	5	Incorrect control parameter for Move instruction	Enter the correct parameters
36608	5	No input parameters for Arch3 instruction	Please enter parameters
36609	5	No motion point parameter for Arch3 instruction	Please enter parameters
36610	5	Incorrect motion point for Arch3 instruction	Enter the correct parameters
36611	5	Incorrect control parameter for Arch3 instruction	Enter the correct parameters
36864	5	No input parameters for Jump instruction	Please enter parameters
36865	5	No motion point parameter for Jump instruction	Please enter parameters
36866	5	Incorrect motion point for Jump instruction	Enter the correct parameters
36867	5	Incorrect control parameter for Jump instruction	Enter the correct parameters
40960	5	No input parameters for Circle3 instruction	Please enter parameters
40961	5	No motion point parameter for Circle3 instruction	Please enter parameters
40962	5	Incorrect motion point for Circle3 instruction	Enter the correct parameters
40963	5	Incorrect control parameter for Circle3 instruction	Enter the correct parameters
45056	5	Circle3 Option Error	Enter the correct parameters
45057	5	Jump Option Error	Enter the correct parameters

ID	Level	Description	Solution
45058	5	Arch Option Error	Enter the correct parameters
45059	5	Arch3 Option Error	Enter the correct parameters
45060	5	Jerk Option Error	Enter the correct parameters
45061	5	JerkR Option Error	Enter the correct parameters
45062	5	JerkS Option Error	Enter the correct parameters
45063	5	Accel Option Error	Enter the correct parameters
45064	5	AccelR Option Error	Enter the correct parameters
45065	5	AccelS Option Error	Enter the correct parameters
45066	5	SpeedFactor Option Error	Enter the correct parameters
45067	5	Speed Option Error	Enter the correct parameters
45068	5	SpeedR Option Error	Enter the correct parameters
45069	5	Limz Option Error	Enter the correct parameters
45070	5	CP Option Error	Enter the correct parameters
45071	5	DO Option Error	Enter the correct parameters
45072	5	Go Option Error	Enter the correct parameters
45073	5	Move Option Error	Enter the correct parameters
45074	5	MoveJ Option Error	Enter the correct parameters
45075	5	Ecp Option Error	Enter the correct parameters
45076	5	EcpSet Option Error	Enter the correct parameters
45077	5	SetExitMode Option Error	Enter the correct parameters
32768	5	No input parameters for speedFactor instruction	Enter the correct parameters
32769	5	Input parameters of speedFactor instruction out of range	Enter the correct parameters
32770	5	DO input parameters Error	Enter the correct parameters
32771	5	DI input parameters Error	Enter the correct parameters
36100	5	No input parameters for movej instruction	Enter the correct parameters
36101	5	No motion point parameter for movej instruction	Enter the correct parameters
36102	5	No motion point parameter for movej instruction	Enter the correct parameters

ID	Level	Description	Solution
36103	5	Incorrect motion point for RP instruction	Enter the correct parameters
36104	5	Incorrect offset for RP instruction	Enter the correct parameters
36105	5	Incorrect motion point for RJ instruction	Enter the correct parameters
36106	5	Incorrect offset for RJ instruction	Enter the correct parameters
36107	5	No input parameters for GoR instruction	Enter the correct parameters
36108	5	Incorrect motion point for GoR instruction	Enter the correct parameters
36109	5	No input parameters for MoveJR instruction	Enter the correct parameters
36110	5	Incorrect motion point for MoveJR instruction	Enter the correct parameters
45079	5	loadSwitch Option Error	Enter the correct parameters
45080	5	loadSet Options Error	Enter the correct parameters
45081	5	CPPParamErrorOption	Enter the correct parameters
45082	5	TOOLParamErrorOption	Enter the correct parameters
45083	5	USERParamErrorOption	Enter the correct parameters
45084	5	SPEEDParamErrorOption	Enter the correct parameters
45085	5	SPEEDSPParamErrorOption	Enter the correct parameters
45086	5	ACCELParamErrorOption	Enter the correct parameters
45087	5	ACCELSParamErrorOption	Enter the correct parameters
45088	5	ARCHParamErrorOption	Enter the correct parameters
45089	5	STARTParamErrorOption	Enter the correct parameters
45090	5	ZLIMITParamErrorOption	Enter the correct parameters
45091	5	ENDParamErrorOption	Enter the correct parameters
45092	5	SYNCaramErrorOption	Enter the correct parameters
45093	5	ARMPParamErrorOption	Enter the correct parameters
45312	5	loadSwitch Option Error	Enter the correct parameters
45313	5	loadSet Options Error	Enter the correct parameters
49152	5	Enable remote control when enabled	Enter the correct parameters
36111	5	No input parameters for GoIO instruction	Enter the correct parameters

ID	Level	Description	Solution
36112	5	Incorrect motion point for GoIO instruction	Enter the correct parameters
36113	5	Incorrect parameters for GoIO instruction	Enter the correct parameters
36114	5	No input parameters for MoveIO instruction	Enter the correct parameters
36115	5	Incorrect motion point for MoveIO instruction	Enter the correct parameters
36116	5	Incorrect parameters for MoveIO instruction	Enter the correct parameters
36117	5	No input parameters for MoveJIO instruction	Enter the correct parameters
36118	5	Incorrect motion point for MoveJIO instruction	Enter the correct parameters
36119	5	No input parameters for MoveJIO instruction	Enter the correct parameters

# Table of Contents

Introduction	1.1
1 Motion Commands	1.2
1.1 Point to point, the target point is Cartesian point	1.2.1
1.2 Linear Movement	1.2.2
1.3 Point to point, the target point is Joint point	1.2.3
1.4 Jump Movement, Jump parameters can be set in this command	1.2.4
1.5 Jump Movement, Jump parameters are called by Arch index	1.2.5
1.6 Move to the Cartesian offset position in a point to point mode	1.2.6
1.7 Move to the Cartesian offset position in a straight line	1.2.7
1.8 Linear movement in parallel with output	1.2.8
1.9 Point to point movement in parallel with output	1.2.9
1.10 Arc Movement	1.2.10
1.11 Circle Movement	1.2.11
2 Motion Parameters	1.3
2.1 Joint Acceleration	1.3.1
2.2 Cartesian Acceleration	1.3.2
2.3 Joint Speed	1.3.3
2.4 Cartesian Speed	1.3.4
2.5 CP	1.3.5
2.6 Synchronization	1.3.6
2.7 Set Load Parameters	1.3.7
3 IO	1.4
3.1 DI	1.4.1
3.2 DO	1.4.2
3.3 DOInstant	1.4.3
4 Program Managing Commands	1.5
4.1 Motion command waiting	1.5.1
4.2 Blocking instruction issuance	1.5.2
4.3 Pause program operation	1.5.3
4.4 Start timing	1.5.4
4.5 Stop timing	1.5.5
4.6 Get current time	1.5.6
5 Pose	1.6
5.1 Get Cartesian coordinates	1.6.1
5.2 Get Joint coordinates	1.6.2



5.3 Cartesian point offset	1.6.3
5.4 Joint point offset	1.6.4
5.5 Cartesian point	1.6.5
5.6 Joint point	1.6.6
6 TCP	1.7
6.1 Create TCP	1.7.1
6.2 Establish TCP connection	1.7.2
6.3 Receive TCP data	1.7.3
6.4 Send TCP data	1.7.4
6.5 Close TCP	1.7.5
7 UDP	1.8
7.1 Create UDP	1.8.1
7.2 Receive UDP data	1.8.2
7.3 Send UDP data	1.8.3
8 Modbus	1.9
8.1 Read the value from Modbus slave coil register address	1.9.1
8.2 Set the coil register in the Modbus slave	1.9.2
8.3 Read the value from the Modbus slave discrete register address	1.9.3
8.4 Read the value from the Modbus slave input register address	1.9.4
8.5 Read the value from the Modbus slave holding register address	1.9.5
8.6 Set the holding register in the Modbus slave	1.9.6
9 Conveyor Tracking	1.10
9.1 Set conveyor number to create a tracing queue	1.10.1
9.2 Obtain status of the object	1.10.2
9.3 Set X,Y axes offset under the set User coordinate system	1.10.3
9.4 Set time compensation	1.10.4
9.5 Synchronize the specified conveyor	1.10.5
9.6 Stop synchronous conveyor	1.10.6
10 Pallet	1.11
10.1 Instantiate matrix pallet	1.11.1
10.2 Instantiate teaching pallet	1.11.2
10.3 Set the next stack index which is to be operated	1.11.3
10.4 Get the current operated stack index	1.11.4
10.5 Set the next pallet layer index which is to be operated	1.11.5
10.6 Get the current pallet layer index	1.11.6
10.7 Reset pallet	1.11.7
10.8 Check whether the stack assembly or dismantling is complete	1.11.8
10.9 Release palletizing instance	1.11.9
10.10 The robot moves from the current position to the first stack position as the configured stack	

assembly path	1.11.10
10.11 The robot moves from the current position to the transition point as the configured stack dismantling path	1.11.11

# Program Guide

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 15:03:14

# 1 Motion Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Point to Point, the target point is Cartesian point

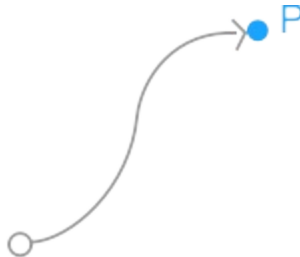
- Function:

```
MovJ(P)
```

Or:

```
local Option={CP=1, SpeedJ=50, AccJ=20}  
MovJ(P, Option)
```

- Description: Point to Point, the target point is Cartesian point.
- Required parameter: P, Indicate target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Linear Movement

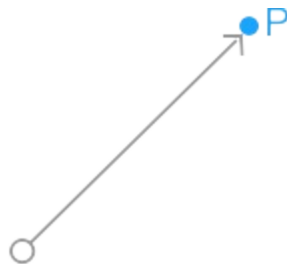
- Function:

```
MovL(P)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}  
MovL(P, Option)
```

- Description: Linear Movement, the target point is Cartesian point.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Point to point, the target point is Joint point

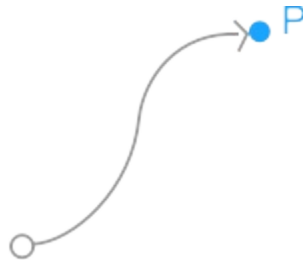
- Function:

```
JointMovJ(P)
```

Or:

```
local Option={CP=1, SpeedJ=50, AccJ=20}  
local P={joint={J1,J2,J3,J4}}  
JointMovJ(P, Option)
```

- Description: Point to point, the target point is Joint point.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only joint point is supported.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100



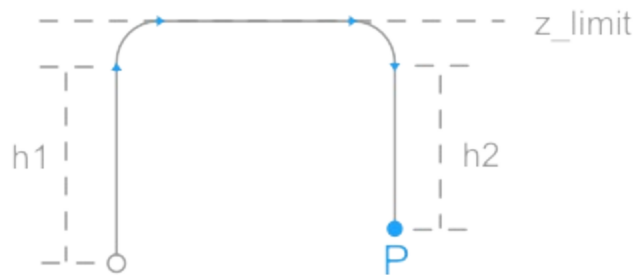
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-03-03 17:41:10

# Jump Movement, Jump parameters can be set in this command

- Function:

```
local Option={SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20}  
Jump(P, Option)
```

- Description: Jump Movement. The jump parameters can be set in this command.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20}
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100
  - Start: Lifting height(h1).
  - ZLimit: Maximum lifting height(z\_limit).
  - End: Dropping height(h2).



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:21:39



# Jump Movement, Jump parameters are called by Arch index

- Function:

```
local Option={SpeedL=50, AccL=20, Arch=1}
Jump(P, Option)
```

- Description: Jump Movement. The jump parameters are called by Arch index.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {SpeedL=50, AccL=20, Start=10, Arch=1}
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100
  - Arch: Arch index. Value range: 0 - 9. Please set Jump parameters on the **Settings > Jump Params** page.

Settings
×

Jump parameter setting

Number	h1(mm)	h2(mm)	zLimit(mm)
0	5	50	50
1	0	0	135
2	6	24	50
3	7	50	17
4	7	50	50
5	7	31	49
6	7	50	14
7	7	50	50
8	7	50	50
9	7	50	21

modify




# Move to the Cartesian offset position in a point to point mode

- Function:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}  
RelMovJ(Offset)
```

Or:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}  
local Option={CP=1, SpeedJ=50, AccJ=20}  
RelMovJ(Offset, Option)
```

- Description: Move to the Cartesian offset position in a point to point mode.
- Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR}, X, Y, Z, R axes offset in the Cartesian coordinate system.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to inset the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Move to the Cartesian offset position in a straight line

- Function:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}  
RelMovL(Offset)
```

Or:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}  
local Option={CP=1, SpeedL=50, Accl=20}  
RelMovL(Offset, Option)
```

- Description: Move to the Cartesian offset position in a straight line.
- Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR} , X, Y, Z ,R axes offset in the Cartesian coordinate system.
- Optional parameter: {CP=1, SpeedL=50, Accl=20}. You can double-click  to inset the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - Accl: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Linear movement in parallel with output

- Function:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
MovLIO(P, IO)
```

Or:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
local Option={CP=1, SpeedL=50, AccL=20}  
MovLIO(P, IO, Option)
```

- Description: Linear movement in parallel with output . Multiple digital output ports can be set.
- Required parameter:
  - P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...: Multiple digital output ports can be set.
    - Mode: Set Distance mode. 0: Distance is a percentage; 1: Distance from the starting point, or from the target point.
    - Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.
    - Index: Digital output port. Value range: 1- 18
    - Status: Status of the digital output port.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Point to point movement in parallel with output

- Function:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
MovJIO(P, IO)
```

Or:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
local Option={CP=1, SpeedL=50, AccL=20}  
MovJIO(P, IO, Option)
```

- Description: Point to point movement in parallel with output . Multiple digital output ports can be set.
- Required parameter:
  - P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...: Multiple digital output ports can be set.
    - Mode: Set Distance mode. 0: Distance is a percentage; 1: Distance from the starting point, or from the target point.
    - Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.
    - Index: Digital output port. Value range: 1- 18
    - Status: Status of the digital output port.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Arc Movement

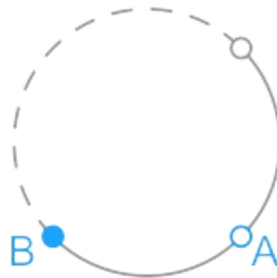
- Function:

```
Arc(P1, P2)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}  
Arc(P1, P2, Option)
```

- Description: Arc movement. This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory.
- Required parameter:
  - P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - P2, End point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Circle Movement

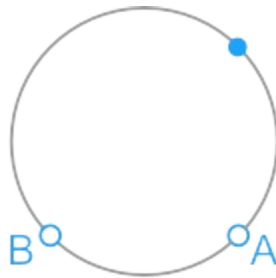
Function:

```
Circle(P1, P2, Count)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}  
Circle(P1, P2, Count, Option)
```

- Description: Circle movement. This command needs to combine with other motion commands, to obtain the starting point of a circle trajectory
- Required parameter:
  - P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - P2, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - Count, Number of circles.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



## 2 Motion Parameters

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Joint Acceleration

- Function:

AccJ(R)

- Description: Set the joint acceleration rate . This command is valid only when the motion mode is MovJ, MovJIO, MovJR, or JointMovJ .
- Required parameter: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:28:42

# Cartesian Acceleration

- Function:

AccL(R)

- Description: Set the Cartesian acceleration rate. This command is valid only when the motion mode is MovL, MovLIO, MovLR, Jump, Arc, Circle.
- Required parameter: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:28:53

# Joint Speed

- Function:

SpeedJ(R)

- Description: Set the joint velocity rate . This command is valid only when the motion mode is MovJ, MovJIO, MovJR, or JointMovJ .
- Required parameter: Velocity rate. Value range: 1 - 100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:20:30

# Cartesian Speed

- Function:

SpeedL(R)

- Description: Set the Cartesian velocity rate. This command is valid only when the motion mode is MovL, MovLIO, MovLR, Jump, Arc, Circle.
- Required parameter: Velocity rate. Value range: 1 -100

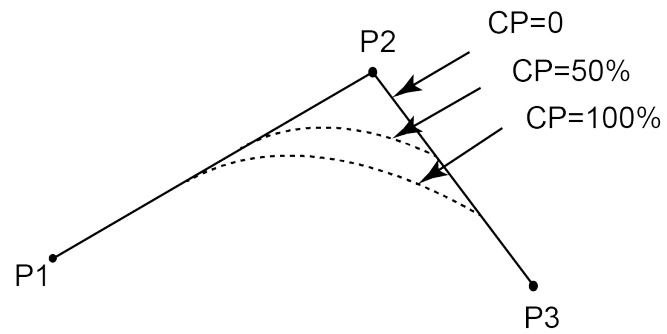
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:19:55

# CP

- Function:

CP(R)

- Description: Set the continuous path rate. This command is invalid when the motion mode is Jump.
- Required parameter: Continuous path rate. Value range: 0-100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 11:01:34

# Synchronization

- Function:

```
Sync()
```

- Description: Whether to stop at this point.
- Required parameter: None.

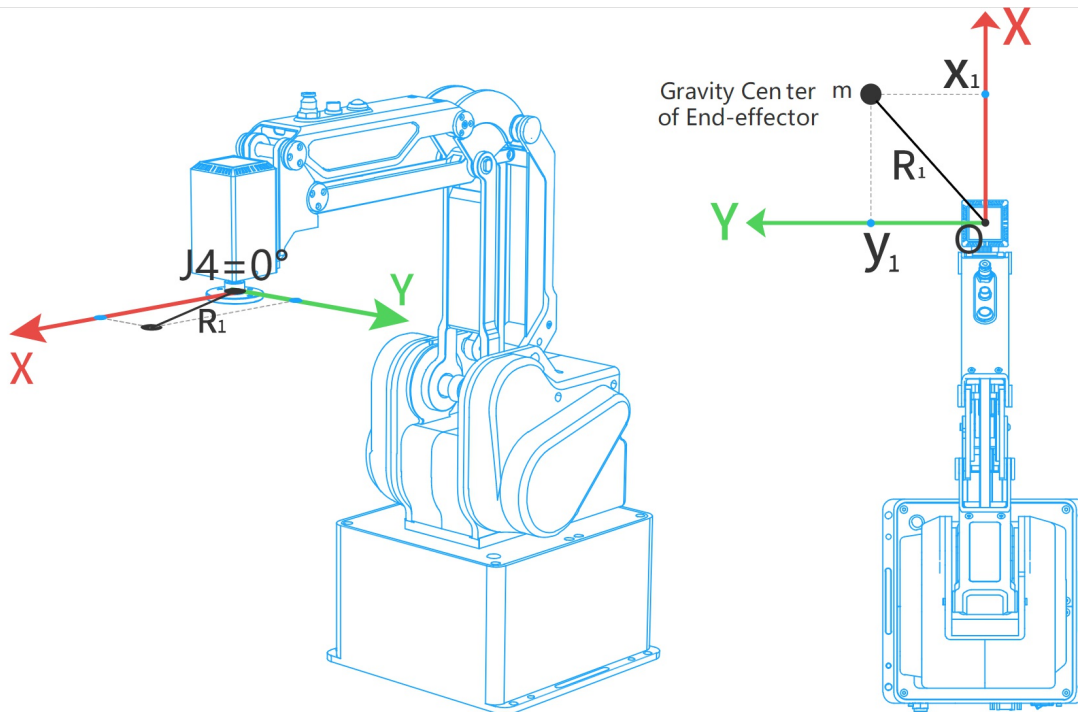
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Set Load Parameters

- Function:

```
SetPayload(payload, {x, y}, index)
```

- Description: Set payload, X-axis offset, Y-axis offset and servo index.
- Required parameter:
  - payload: Payload. Value range: 0- 750. Unit: g
  - {x,y}: Offset in X-axis and Y-axis
- Optional parameter: index, servo parameter index. The default value range is 1 - 10.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# 3 IO

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# DI

- Function:

```
DI(Index)
```

- Description: Get the status of the digital input port.
- Required parameter: Index, Digital input port. Value range: 1-18
- Return:
  - When an port is set in the DI function, **DI(index)** returns the status (ON/OFF) of this specified input port.
  - When there is no port in the DI function, **DI()** returns the status of all the input ports, which are saved in a table. For example, local di=(), the saving format is **{num = 24 value = {0x55, 0xAA, 0x52}}**, you can obtain the status of the specified input port with **di.num** and **di.value[n]**.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 11:01:33

# DO

- Function:

```
DO(Index,ON/OFF)
```

- Description: Set the status of digital output port (Queue command).
- Required parameter:
  - Index: Digital output port. Value range: 1 - 18
  - ON/OFF: Status of the digital output port.

Queue command: When the robot system receives a command, this command will be pressed into the internal command queue. The robot system will execute commands in the order in which the commands were pressed into the queue.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:24:07

# DOInstant

- Function:

```
DOInstant(Index,ON/OFF)
```

- Description: Set the status of digital output port (Immediate command).
- Required parameter:
  - Index: Digital output port. Value range: 1 - 18
  - ON/OFF: Status of the digital output port.

Immediate command: The robot system will process the command once received regardless of whether there is the rest commands processing or not in the current controller;

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-03-03 17:41:10

## 4 Program Managing Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Motion command waiting

- Function:

```
Wait(time)
```

- Description: Set the delay time for robot motion commands.
- Required parameter: time, Delay time. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Blocking instruction issuance

- Function:

```
Sleep(time)
```

- Description: Set the delay time for all commands.
- Required parameter: time, Delay time. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Pause program operation

- Function:

```
Pause()
```

- Description: Pause the running program. When the program runs to this command, robot pauses running and you need to click **Resume** on the DobotStudio2020 to recover the running.
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54



# Start timing

- Function:

```
ResetElapsedTime()
```

- Description: Start timing after all commands before this command are executed completely. Use in conjunction with ElapsedTime() command.
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Stop timing

- Function:

```
ElapsedTime()
```

- Description: Stop timing and return the time difference. Use in conjunction with ResetElapsedTime() command
- Required parameter: None
- Return: Time difference. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Get current time

- Function:

```
Systeme()
```

- Description: Get the current time
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

## 5 Pose

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Get Cartesian coordinates

- Function:

```
GetPose()
```

- Description: Get the current pose of the robot under the Cartesian coordinate system. If you have set the User or Tool coordinate system, the current pose is under the current User or Tool coordinate system.
- Required parameter: None
- Return: Cartesian coordinate of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Get Joint coordinates

- Function:

```
GetAngle()
```

- Description: Get the current pose of the robot under the Joint coordinate system.
- Required parameter: None
- Return: Joint coordinates of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Cartesian point offset

- Function:

```
local Offset={OffsetX, OffsetY, OffsetZ, OffsetR}  
RelPoint(P, Offset)
```

- Description: Set the X, Y, Z,R axes offset under the Cartesian coordinate system to return a new Cartesian coordinate point. he robot can move to this point in all motion commands except JointMovJ.
- Required parameter:
  - P, Indicate the current Cartesian point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {OffsetX, OffsetY, OffsetZ, OffsetR}: X, Y, Z, R axes offset in the Cartesian coordinate system.
- Return: Cartesian point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Joint point offset

- Function:

```
local Offset={Offset1, Offset2, Offset3, Offset4}  
RelJoint(P, Offset)
```

- Description: Set the joint offset in the Joint coordinate system to return a new joint point. The robot can move to this point only in JointMovJ command .
- Required parameter:
  - P, Indicate the current joint point, which is user-defined or obtained from the points list. Only joint point is supported.
  - {Offset1, Offset2, Offset3, Offset4}: J1 - J4 axes offset.
- Return: Joint point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# Cartesian point

- Function:

```
local P={coordinate = {x,y,z,r}, tool = 0, user = 0}
```

- Description: User-define a Cartesian point.
- Required parameter:
  - {x,y,z,r}: X, Y, Z, R axes coordinates.
  - tool: Tool coordinate system index. Value range: 0-9
  - user: User coordinate system index. Value range: 0-9

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Joint point

- Function:

```
local P={joint= {j1,j2,j3,j4}}
```

- Description: User-define a joint point.
- Required parameter: {j1,j2,j3,j4}, J1-J4 axes coordinates.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

## 6 TCP

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# Create TCP

- Function:

```
Err, Socket = TCPCreate(IsServer, IP, Port)
```

- Description: Create a TCP network. Only support a single connection.
- Required parameter:
  - IsServer: Whether to create a server. false: Create a client; true: Create a server.
  - IP: IP address of the server, which is in the same network segment of the client without conflict.
  - Port: Server port. When the robot is set as a server, **port** cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.
- Return:
  - Err:
    - 0: TCP network is created successfully.
    - 1: TCP network is created failed.
  - Socket: Socket object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Establish TCP connection

- Function:

```
TCPStart(Socket, Timeout)
```

- Description: Establish TCP connection.
- Required parameter:
  - Socket: Socket object.
  - Timeout: Wait timeout. Unit: s. If Timeout is 0, the connection is still waiting. If not, after exceeding the timeout, the connection is exited.
- Return:
  - 0: TCP connection is successful.
  - 1: Input parameters are incorrect.
  - 2: Socket object is not found.
  - 3: Timeout setting is incorrect.
  - 4: If the robot is set as a client, it indicates that the connection is wrong. If the robot is set as a server, it indicates that receiving data is wrong.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Receive TCP data

- Function:

```
Err, RecBuf = TCPRead(Socket, Timeout, Type)
```

- Description: Robot as a client receives data from a server or as a server receives data from a client .
- Required parameter:
  - Socket: Socket object.
  - Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.
  - Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string.
- Return:
  - Err:
    - 0: Receiving data is successful.
    - 1: Receiving data is failed.
  - Recbuf: Data buffer.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Send TCP data

- Function:

```
TCPWrite(Socket, Buf, Timeout)
```

- Description: Robot as a client sends data to a server or as a server sends data to a client.
- Required parameter:
  - Socket: Socket object.
  - Buf: Data sent by the robot.
  - Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Close TCP

- Function:

```
TCPDestroy(Socket)
```

- Description: Release a TCP network.
- Required parameter: Socket, Socket object.
- Return:
  - 0: Releasing TCP is successful.
  - 1: Releasing TCP is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



## 7 UDP

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# Create UDP

- Function:

```
Err, Socket = UDPCreate(IsServer, IP, Port)
```

- Description: Create a UDP network. Only support a single connection.
- Required parameter:
  - IsServer: Whether to create a server. false: Create a client; true: Create a server.
  - IP: IP address of the server, which is in the same network segment of the client without conflict.
  - Port: Server port. When the robot is set as a server, **port** cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.
- Return:
  - Err:
    - 0: UDP network is created successfully.
    - 1: UDP network is created failed.
    - Socket: Socket object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Receive UDP data

- Function:

```
Err, RecBuf = UDPRead(Socket, Timeout, Type)
```

- Description: Robot as a client receives data from a server or as a server receives data from a client .
- Required parameter:
  - Socket: Socket object.
  - Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.
  - Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string.
- Return:
  - Err:
    - 0: Receiving data is successful.
    - 1: Receiving data is failed.
  - Recbuf: Data buffer.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Send UDP data

- Function:

```
UDPWrite(Socket, Buf, Timeout)
```

- Description: Robot as a client sends data to a server or as a server sends data to a client.
- Required parameter:
  - Socket: Socket object.
  - Buf: Data sent by the robot.
  - Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# 8 Modbus

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# Read the value from Modbus slave coil register address

- Function:

```
GetCoils(Addr, Count)
```

- Description: Read the value from Modbus slave coil register address.
- Required parameter:
  - Addr: Starting address of the coils. Value range: 0-4095
  - Count: Number of the coils to read. Value range: 0 to 4096- Addr
- Return: Return a table, each with the value 1 or 0, where the first value in the table corresponds to the coil value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

## Set the coil register in the Modbus slave

- Function:

```
SetCoils(Addr, Count, Table)
```

- Description: Set the coil register in the Modbus slave. This command is not supported when the coil register address is from 0 to 5.
- Required parameter:
  - Addr: Starting address of the coils to set. Value range: 6 - 4095
  - Count: Number of the coils to set. Value range: 0 to 4096- Addr

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Read the value from the Modbus slave discrete register address

- Function:

```
GetInBits(Addr, Count)
```

- Description: Read the value from the Modbus slave discrete register address.
- Required parameter:
  - Addr: Starting address of the discrete inputs to read. Value range: 0-4095
  - Count: Number of the discrete inputs to read. Value range: 0 to 4096- Addr
- Return: Return a table, each with the value 1 or 0, where the first value in the table corresponds to the discrete value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# Read the value from the Modbus slave input register address

- Function:

```
GetInRegs(Addr, Count, Type)
```

- Description: Read the input register value with the specified data type from the Modbus slave.
- Required parameter:
  - Addr: Starting address of the input registers. Value range: 0 - 4095
  - Count: Number of the input registers to read. Value range: 0 to 4096- Addr
  - Type: Data type.
    - Empty: Read 16-bit unsigned integer ( two bytes, occupy one register) .
    - "U16": Read 16-bit unsigned integer ( two bytes, occupy one register).
    - "U32": Read 32-bit unsigned integer (four bytes, occupy two registers).
    - "F32": Read 32-bit single-precision floating-point number (four bytes, occupy two registers).
    - "F64": Read 64-bit double-precision floating-point number (eight bytes, occupy four registers).
- Return: Return a table, the first value in the table corresponds to the input register value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-03-03 17:41:10

# Read the value from the Modbus slave holding register address

- Function:

```
GetHoldRegs(Addr, Count, Type)
```

- Description: Read the holding register value from the Modbus slave according to the specified data type
- Required parameter:
  - Addr: Starting address of the holding registers. Value range: 0 - 4095
  - Count: Number of the holding registers to read. Value range: 0 to 4096- Addr
  - Type: Data type.
    - Empty: Read 16-bit unsigned integer ( two bytes, occupy one register).
    - "U16": Read 16-bit unsigned integer ( two bytes, occupy one register).
    - "U32": Read 32-bit unsigned integer (four bytes, occupy two registers).
    - "F32": Read 32-bit single-precision floating-point number (four bytes, occupy two registers).
    - "F64": Read 64-bit double-precision floating-point number (eight bytes, occupy four registers).
- Return: Return a table, the first value in the table corresponds to the input register value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Set the holding register in the Modbus slave

- Function:

```
SetHoldRegs(Addr, Count, Table, Type)
```

- Description: Set the holding register in the Modbus slave
- Required parameter:
  - Addr: Starting address of the holding registers to set. Value range: 0 - 4095
  - Count: Number of the holding registers to set. Value range: 0 to 4096- Addr
  - Type: Data type.
    - Empty: Read 16-bit unsigned integer ( two bytes, occupy one register) .
    - "U16": Set 16-bit unsigned integer ( two bytes, occupy one register).
    - "U32": Set 32-bit unsigned integer (four bytes, occupy two registers).
    - "F32": Set 32-bit single-precision floating-point number (four bytes, occupy two registers).
    - "F64": Set 64-bit double-precision floating-point number (eight bytes, occupy four registers).

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# 9 Conveyor Tracking

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Set conveyor number to create a tracing queue

- Function:

```
CnvVison(CnvID)
```

- Description: Set conveyor number to create a tracing queue.
- Required parameter: CnvID, Conveyor number. Only support single conveyor.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

## Obtain status of the object

- Function:

```
GetCnvObject(CnvID, ObjID)
```

- Description: Obtain the information of the part on the conveyor to check whether the part is in the pickup area .
- Required parameter:
  - CnvID: Conveyor index.
  - ObjID: Part index.
- Return:
  - Part status: Whether there is a part. Value range: true or false
  - Part type
  - Part coordinate (x,y,r)

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Set X,Y axes offset under the set User coordinate system

- Function:

```
SetCnvPointOffset(OffsetX,OffsetY)
```

- Description: Set X,Y axes offset under the set User coordinate system.
- Required parameter:
  - OffsetX: X-axis offset.
  - OffsetY: Y-axis offset.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:27:19

# Set time compensation

- Function:

```
SetCnvTimeCompensation(Time)
```

- Description: Set time compensation. This command is used for compensating the pick-up position offset in the moving direction of the conveyor which is caused by taking photos with a time delay.
- Required parameter: Time, time-offset. Unit: ms
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# Synchronize the specified conveyor

- Function:

```
SyncCnv(CnvID)
```

- Description: Synchronize the specified conveyor. The motion commands used between `SyncCnv(CnvID)` and `StopSyncCnv(CnvID)` only support `MovL` command.
- Required parameter: `CnvID`, Conveyor index.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:14:42

# Stop synchronous conveyor

- Function:

```
StopSyncCnv(CnvID)
```

- Description: Stop synchronizing the conveyor. The other commands following this command will not be executed until this command running is completed.
- Required parameter: CnvID, Conveyor index.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# 10 Pallet

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Instantiate matrix pallet

- Function:

```
Pallet = MatrixPallet (Index,ID)
```

Or:

```
local Option={IsUnstack= true, User= 1}  
Pallet = MatrixPallet (Index,ID, Option)
```

- Description: Instantiate matrix pallet.
- Required parameter:
  - Index: Matrix pallet index.
  - ID: Unique identification of pallet
- Optional parameter: {IsUnstack= true, User= 1}. You can double-click  to insert the command with optional parameters.
  - IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode
  - User: User coordinate system index. If not set, the default is User 0 coordinate system.
- Return: Matrix pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Instantiate teaching pallet

- Function:

```
Pallet = TeachPallet (Index, ID, Option)
```

Or:

```
local Option={IsUnstack= true, User= 1}  
Pallet = TeachPallet (Index, ID, Option)
```

- Description: Instantiate teaching pallet.
- Required parameter:
  - Index: Teaching pallet index.
  - ID: Unique identification of pallet
- Optional parameter: {IsUnstack= true, User= 1}. You can double-click  to insert the command with optional parameters.
  - IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode
  - User: User coordinate system index. If not set, the default is User 0 coordinate system.
- Return: Teaching pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

## Set the next stack index which is to be operated

- Function:

```
SetPartIndex(Pallet, Index)
```

- Description: Set the next stack index which is to be operated.
- Required parameter:
  - Pallet: Pallet object.
  - Index: The next stack index. Initial value: 0

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Get the current operated stack index

- Function

```
GetPartIndex(Pallet)
```

- Description: Get the current operated stack index.
- Required parameter: Pallet, Pallet object.
- Return: The current operated stack index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Set the next pallet layer index which is to be operated

- Function:

```
SetLayerIndex(Pallet, Index)
```

- Description: Set the next pallet layer index which is to be operated.
- Required parameter:
  - Pallet: Pallet object.
  - Index: The next pallet layer index. Initial value: 0

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



## Get the current pallet layer index

- Function:

```
GetLayerIndex(Pallet)
```

- Description: Get the current pallet layer index.
- Required parameter: Pallet, Pallet object.
- Return: The current pallet layer index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Reset pallet

- Function:

```
Restet(Pallet)
```

- Description: Reset pallet.
- Required parameter: Pallet, Pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Check whether the stack assembly or dismantling is complete

- Function:

```
IsDone(Pallet)
```

- Description: Check whether the stack assembly or dismantling is complete.
- Required parameter: Pallet, Pallet object.
- Return:
  - true: Finished.
  - false: Un-finished.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Release palletizing instance

- Function:

```
Release(Pallet)
```

- Description: Release palletizing instance.
- Required parameter: Pallet, Pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# The robot moves from the current position to the first stack position as the configured stack assembly path

- Function:

```
PalletMoveIn(Pallet)
```

Or:

```
local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}  
PalletMoveIn(Pallet, Option)
```

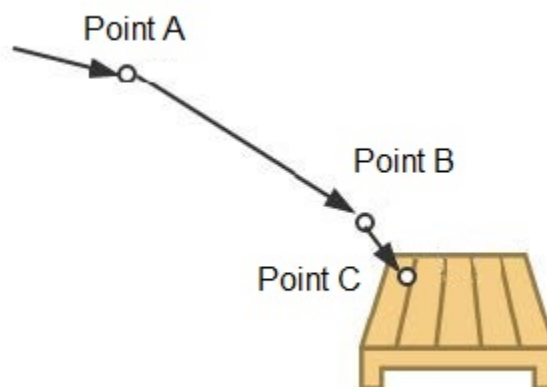
- Description: The robot moves from the current position to the first stack position as the configured stack assembly path.
- Required parameter: Pallet, Pallet object.
- Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click

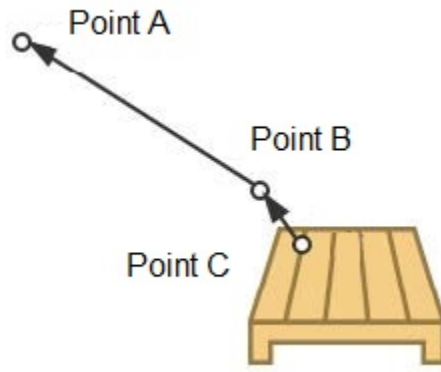


to insert the command with optional parameters.

- SpeedAB: Velocity rate when the robot moves from the transition point to the preparation point. Value range: 1-100
- SpeedBC: Velocity rate when the robot moves from the preparation point to the first stack point. Value range: 1-100
- AccAB: Acceleration rate when the robot moves from the transition point to the preparation point. Value range: 1-100
- AccBC: Acceleration rate when the robot moves from the preparation point to the first stack point. Value range: 1-100
- CP: Continuous path rate. Value range: 0-100

The stack assembly path and dismantling path are shown as follows. Point A is the transition point, which is fixed or varies with the pallet layer. Point B is the preparation point which is calculated by the target point and the set offset. Point C is the first stack point.





Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# The robot moves from the current position to the transition point as the configured stack dismantling path

- Function:

```
PalletMoveOut(Pallet)
```

Or:

```
local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}  
PalletMoveOut(Pallet, Option)
```

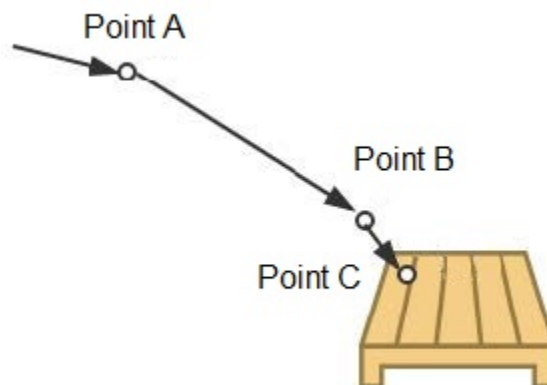
- Description: The robot moves from the current position to the transition point as the configured stack dismantling path.
- Required parameter: Pallet, Pallet object.
- Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click

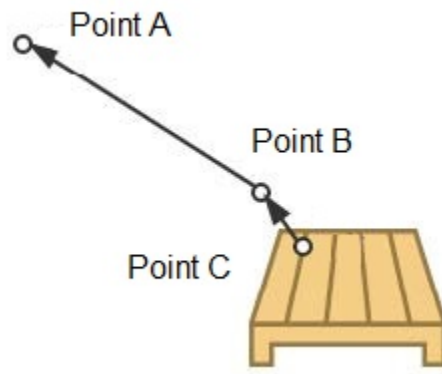


to insert the command with optional parameters.

- SpeedAB: Velocity rate when the robot moves from the preparation point to the transition point. Value range: 1-100
- SpeedBC: Velocity rate when the robot moves from the first stack point to the preparation point. Value range: 1-100
- AccAB: Acceleration rate when the robot moves from the preparation point to the transition point. Value range: 1-100
- AccBC: Acceleration rate when the robot moves from the first stack point to the preparation point. Value range: 1-100
- CP: Continuous path rate. Value range: 0-100

The stack assembly path and dismantling path are shown as follows. Point A is the transition point, which is fixed or varies with the pallet layer. Point B is the preparation point which is calculated by the target point and the set offset. Point C is the first stack point.





Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54



# Table of Contents

Introduction	1.1
1 Motion Commands	1.2
1.1 Point to point, the target point is Cartesian point	1.2.1
1.2 Linear Movement	1.2.2
1.3 Point to point, the target point is Joint point	1.2.3
1.4 Jump Movement, Jump parameters can be set in this command	1.2.4
1.5 Jump Movement, Jump parameters are called by Arch index	1.2.5
1.6 Move to the Cartesian offset position in a point to point mode	1.2.6
1.7 Move to the Cartesian offset position in a straight line	1.2.7
1.8 Linear movement in parallel with output	1.2.8
1.9 Point to point movement in parallel with output	1.2.9
1.10 Arc Movement	1.2.10
1.11 Circle Movement	1.2.11
2 Motion Parameters	1.3
2.1 Joint Acceleration	1.3.1
2.2 Cartesian Acceleration	1.3.2
2.3 Joint Speed	1.3.3
2.4 Cartesian Speed	1.3.4
2.5 CP	1.3.5
2.6 Synchronization	1.3.6
2.7 Set Load Parameters	1.3.7
3 IO	1.4
3.1 DI	1.4.1
3.2 DO	1.4.2
3.3 DOInstant	1.4.3
4 Program Managing Commands	1.5
4.1 Motion command waiting	1.5.1
4.2 Blocking instruction issuance	1.5.2
4.3 Pause program operation	1.5.3
4.4 Start timing	1.5.4
4.5 Stop timing	1.5.5
4.6 Get current time	1.5.6
5 Pose	1.6
5.1 Get Cartesian coordinates	1.6.1
5.2 Get Joint coordinates	1.6.2

5.3 Cartesian point offset	1.6.3
5.4 Joint point offset	1.6.4
5.5 Cartesian point	1.6.5
5.6 Joint point	1.6.6
6 TCP	1.7
6.1 Create TCP	1.7.1
6.2 Establish TCP connection	1.7.2
6.3 Receive TCP data	1.7.3
6.4 Send TCP data	1.7.4
6.5 Close TCP	1.7.5
7 UDP	1.8
7.1 Create UDP	1.8.1
7.2 Receive UDP data	1.8.2
7.3 Send UDP data	1.8.3
8 Modbus	1.9
8.1 Create Modbus master station	1.9.1
8.2 Disconnect with Modbus slave	1.9.2
8.3 Read the value from the Modbus slave coil register address	1.9.3
8.4 Set the coil register in the Modbus slave	1.9.4
8.5 Read the value from the Modbus slave discrete register address	1.9.5
8.6 Read the value from the Modbus slave input register address	1.9.6
8.7 Read the value from the Modbus slave holding register address	1.9.7
8.8 Set the holding register in the Modbus slave	1.9.8
9 Conveyor Tracking	1.10
9.1 Set conveyor number to create a tracing queue	1.10.1
9.2 Obtain status of the object	1.10.2
9.3 Set X,Y axes offset under the set User coordinate system	1.10.3
9.4 Set time compensation	1.10.4
9.5 Synchronize the specified conveyor	1.10.5
9.6 Stop synchronous conveyor	1.10.6
10 Pallet	1.11
10.1 Instantiate matrix pallet	1.11.1
10.2 Instantiate teaching pallet	1.11.2
10.3 Set the next stack index which is to be operated	1.11.3
10.4 Get the current operated stack index	1.11.4
10.5 Set the next pallet layer index which is to be operated	1.11.5
10.6 Get the current pallet layer index	1.11.6
10.7 Reset pallet	1.11.7
10.8 Check whether the stack assembly or dismantling is complete	1.11.8

10.9 Release palletizing instance	1.11.9
10.10 The robot moves from the current position to the first stack position as the configured stack assembly path	1.11.10
10.11 The robot moves from the current position to the transition point as the configured stack dismantling path	1.11.11
11 Vision	1.12
11.1 Initialize the connection to the camera	1.12.1
11.2 Trigger the camera to take a picture	1.12.2
11.3 Send data	1.12.3
11.4 Receive data	1.12.4
11.5 Close the camera	1.12.5

# Program Guide

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 15:03:14

# 1 Motion Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Point to Point, the target point is Cartesian point

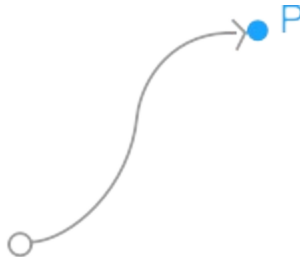
- Function:

```
MovJ(P)
```

Or:

```
local Option={CP=1, SpeedJ=50, AccJ=20}  
MovJ(P, Option)
```

- Description: Point to Point, the target point is Cartesian point.
- Required parameter: P, Indicate target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Linear Movement

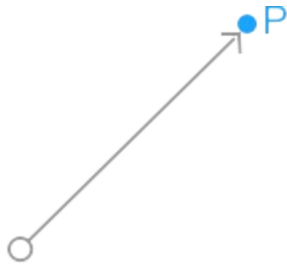
- Function:

```
MovL(P)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}  
MovL(P, Option)
```

- Description: Linear Movement, the target point is Cartesian point.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Point to point, the target point is Joint point

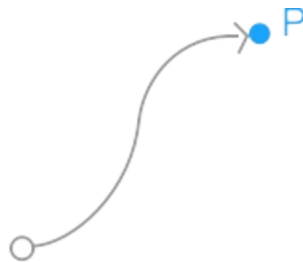
- Function:

```
JointMovJ(P)
```

Or:

```
local Option={CP=1, SpeedJ=50, AccJ=20}  
local P={joint={J1,J2,J3,J4}}  
JointMovJ(P, Option)
```

- Description: Point to point, the target point is Joint point.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only joint point is supported.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-03-03 17:41:10

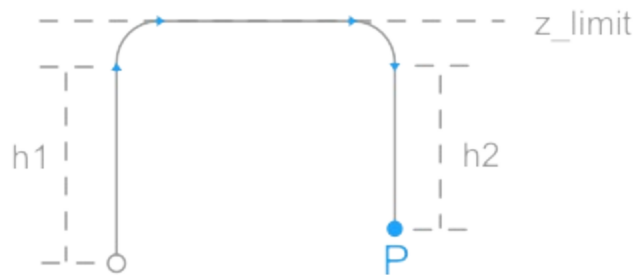


# Jump Movement, Jump parameters can be set in this command

- Function:

```
local Option={SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20}  
Jump(P, Option)
```

- Description: Jump Movement. The jump parameters can be set in this command.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20}
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100
  - Start: Lifting height(h1).
  - ZLimit: Maximum lifting height(z\_limit).
  - End: Dropping height(h2).



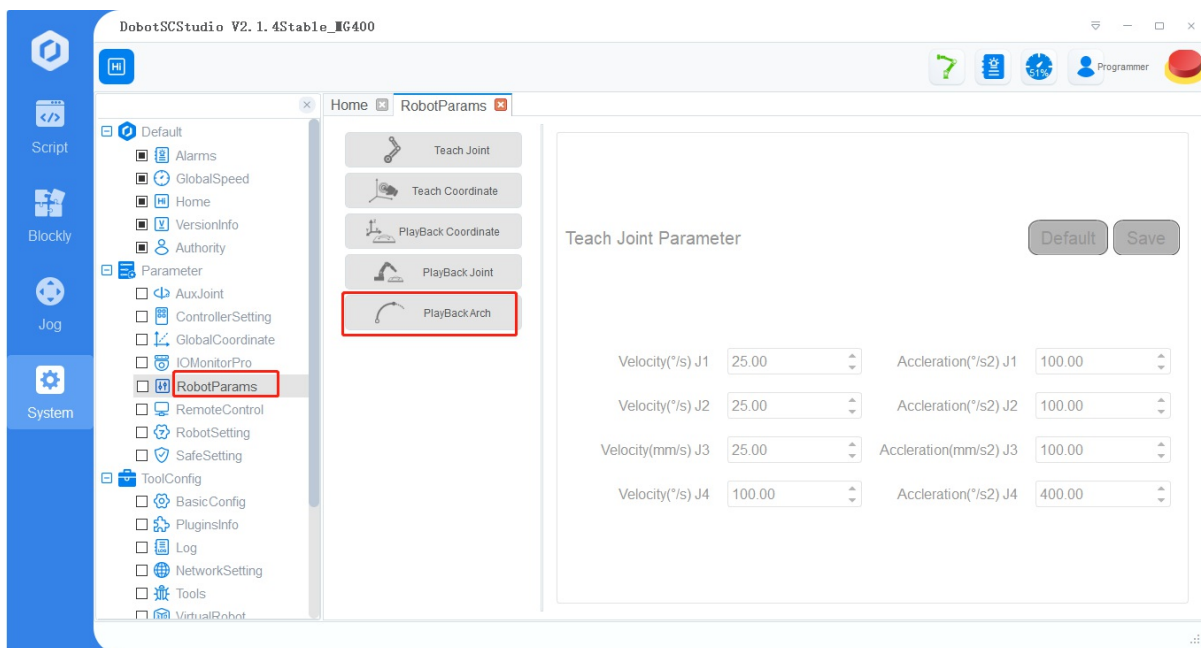
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:21:39

## 1.5 Jump Movement, Jump parameters are called by Arch index

- Function:

```
local Option={SpeedL=50, AccL=20, Arch=1}
Jump(P, Option)
```

- Description: Jump Movement. The jump parameters are called by Arch index.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {SpeedL=50, AccL=20, Start=10, Arch=1}
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100
  - Arch: Arch index. Value range: 0 - 9. Please set Jump parameters on the **System > Parameters > RobotParams > PlayBackArch** page.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 15:27:01


# Move to the Cartesian offset position in a point to point mode

- Function:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}
RelMovJ(Offset)
```

Or:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}
local Option={CP=1, SpeedJ=50, AccJ=20}
RelMovJ(Offset, Option)
```

- Description: Move to the Cartesian offset position in a point to point mode.
- Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR}, X, Y, Z, R axes offset in the Cartesian coordinate system.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to inset the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Move to the Cartesian offset position in a straight line

- Function:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}  
RelMovL(Offset)
```

Or:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}  
local Option={CP=1, SpeedL=50, Accl=20}  
RelMovL(Offset, Option)
```

- Description: Move to the Cartesian offset position in a straight line.
- Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR} , X, Y, Z ,R axes offset in the Cartesian coordinate system.
- Optional parameter: {CP=1, SpeedL=50, Accl=20}. You can double-click  to inset the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - Accl: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Linear movement in parallel with output

- Function:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
MovLIO(P, IO)
```

Or:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
local Option={CP=1, SpeedL=50, AccL=20}  
MovLIO(P, IO, Option)
```

- Description: Linear movement in parallel with output . Multiple digital output ports can be set.
- Required parameter:
  - P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...: Multiple digital output ports can be set.
    - Mode: Set Distance mode. 0: Distance is a percentage; 1: Distance from the starting point, or from the target point.
    - Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.
    - Index: Digital output port. Value range: 1- 18
    - Status: Status of the digital output port.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Point to point movement in parallel with output

- Function:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
MovJIO(P, IO)
```

Or:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}  
local Option={CP=1, SpeedJ=50, AccJ=20}  
MovJIO(P, IO, Option)
```

- Description: Point to point movement in parallel with output . Multiple digital output ports can be set.
- Required parameter:
  - P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...: Multiple digital output ports can be set.
    - Mode: Set Distance mode. 0: Distance is a percentage; 1: Distance from the starting point, or from the target point.
    - Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.
    - Index: Digital output port. Value range: 1- 18
    - Status: Status of the digital output port.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 15:11:45


# Arc Movement

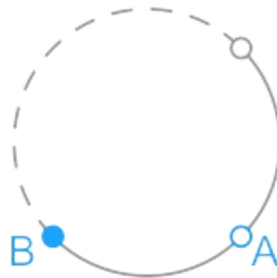
- Function:

```
Arc(P1, P2)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}  
Arc(P1, P2, Option)
```

- Description: Arc movement. This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory.
- Required parameter:
  - P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - P2, End point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53


# Circle Movement

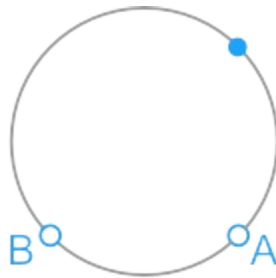
Function:

```
Circle(P1, P2, Count)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}  
Circle(P1, P2, Count, Option)
```

- Description: Circle movement. This command needs to combine with other motion commands, to obtain the starting point of a circle trajectory
- Required parameter:
  - P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - P2, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - Count, Number of circles.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click  to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



## 2 Motion Parameters

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Joint Acceleration

- Function:

AccJ(R)

- Description: Set the joint acceleration rate . This command is valid only when the motion mode is MovJ, MovJIO, MovJR, or JointMovJ .
- Required parameter: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:28:42

# Cartesian Acceleration

- Function:

AccL(R)

- Description: Set the Cartesian acceleration rate. This command is valid only when the motion mode is MovL, MovLIO, MovLR, Jump, Arc, Circle.
- Required parameter: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:28:53

# Joint Speed

- Function:

SpeedJ(R)

- Description: Set the joint velocity rate . This command is valid only when the motion mode is MovJ, MovJIO, MovJR, or JointMovJ .
- Required parameter: Velocity rate. Value range: 1 - 100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:20:30

# Cartesian Speed

- Function:

SpeedL(R)

- Description: Set the Cartesian velocity rate. This command is valid only when the motion mode is MovL, MovLIO, MovLR, Jump, Arc, Circle.
- Required parameter: Velocity rate. Value range: 1 -100

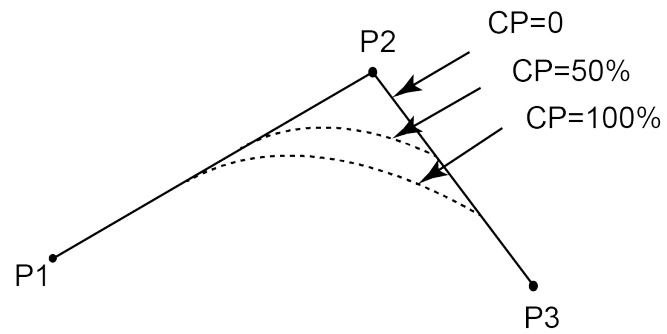
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:19:55

# CP

- Function:

CP(R)

- Description: Set the continuous path rate. This command is invalid when the motion mode is Jump.
- Required parameter: Continuous path rate. Value range: 0-100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 11:01:34

# Synchronization

- Function:

```
Sync()
```

- Description: Whether to stop at this point.
- Required parameter: None.

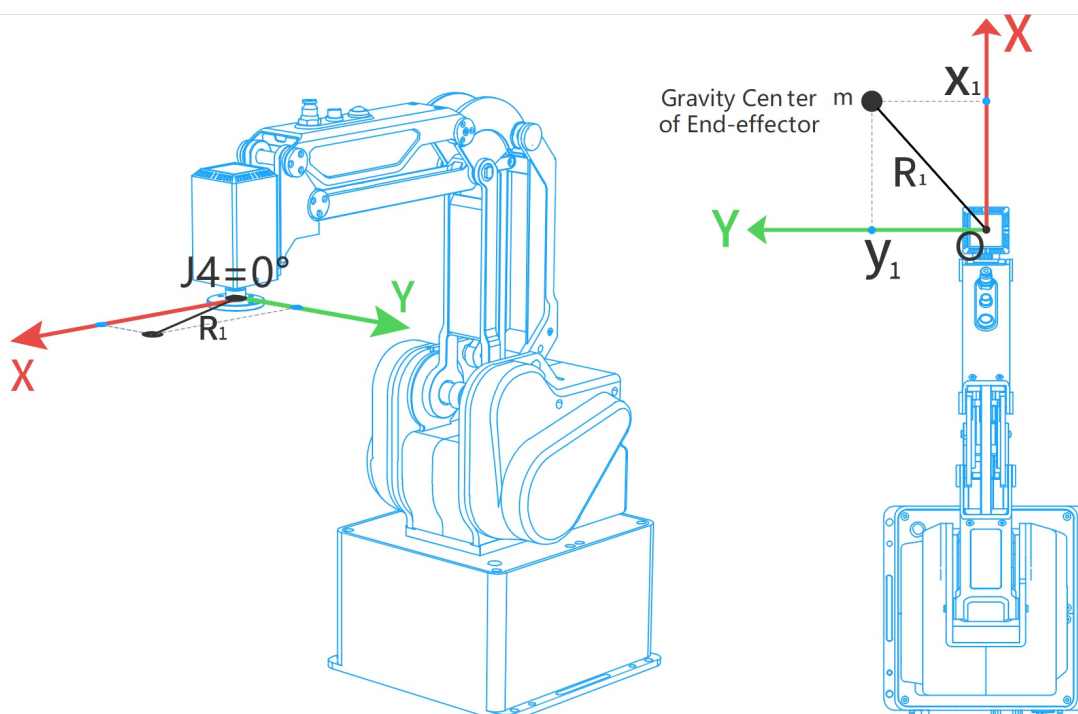
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Set Load Parameters

- Function:

```
SetPayload(payload, {x, y}, index)
```

- Description: Set payload, X-axis offset, Y-axis offset and servo index.
- Required parameter:
  - payload: Payload. Value range: 0- 750. Unit: g
  - {x,y}: Offset in X-axis and Y-axis
- Optional parameter: index, servo parameter index. The default value range is 1 - 10.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# 3 IO

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# DI

- Function:

```
DI(Index)
```

- Description: Get the status of the digital input port.
- Required parameter: Index, Digital input port. Value range: 1-18
- Return:
  - When an port is set in the DI function, **DI(index)** returns the status (ON/OFF) of this specified input port.
  - When there is no port in the DI function, **DI()** returns the status of all the input ports, which are saved in a table. For example, local di=(), the saving format is **{num = 24 value = {0x55, 0xAA, 0x52}}**, you can obtain the status of the specified input port with **di.num** and **di.value[n]**.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 11:01:33

# DO

- Function:

```
DO(Index,ON/OFF)
```

- Description: Set the status of digital output port (Queue command).
- Required parameter:
  - Index: Digital output port. Value range: 1 - 18
  - ON/OFF: Status of the digital output port.

Queue command: When the robot system receives a command, this command will be pressed into the internal command queue. The robot system will execute commands in the order in which the commands were pressed into the queue.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:24:07

# DOInstant

- Function:

```
DOInstant(Index,ON/OFF)
```

- Description: Set the status of digital output port (Immediate command).
- Required parameter:
  - Index: Digital output port. Value range: 1 - 18
  - ON/OFF: Status of the digital output port.

Immediate command: The robot system will process the command once received regardless of whether there is the rest commands processing or not in the current controller;

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-03-03 17:41:10

## 4 Program Managing Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Motion command waiting

- Function:

```
wait(time)
```

- Description: Set the delay time for robot motion commands.
- Required parameter: time, Delay time. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Blocking instruction issuance

- Function:

```
Sleep(time)
```

- Description: Set the delay time for all commands.
- Required parameter: time, Delay time. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Pause program operation

- Function:

```
Pause()
```

- Description: Pause the running program. When the program runs to this command, robot pauses running and you need to click **Resume** on the Software to recover the running.
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 15:14:12



# Start timing

- Function:

```
ResetElapsedTime()
```

- Description: Start timing after all commands before this command are executed completely. Use in conjunction with ElapsedTime() command.
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Stop timing

- Function:

```
ElapsedTime()
```

- Description: Stop timing and return the time difference. Use in conjunction with ResetElapsedTime() command
- Required parameter: None
- Return: Time difference. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Get current time

- Function:

```
Systeme()
```

- Description: Get the current time
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

## 5 Pose

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

## Get Cartesian coordinates

- Function:

```
GetPose()
```

- Description: Get the current pose of the robot under the Cartesian coordinate system. If you have set the User or Tool coordinate system, the current pose is under the current User or Tool coordinate system.
- Required parameter: None
- Return: Cartesian coordinate of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Get Joint coordinates

- Function:

```
GetAngle()
```

- Description: Get the current pose of the robot under the Joint coordinate system.
- Required parameter: None
- Return: Joint coordinates of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Cartesian point offset

- Function:

```
local Offset={OffsetX, OffsetY, OffsetZ, OffsetR}  
RelPoint(P, Offset)
```

- Description: Set the X, Y, Z,R axes offset under the Cartesian coordinate system to return a new Cartesian coordinate point. he robot can move to this point in all motion commands except JointMovJ.
- Required parameter:
  - P, Indicate the current Cartesian point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {OffsetX, OffsetY, OffsetZ, OffsetR}: X, Y, Z, R axes offset in the Cartesian coordinate system.
- Return: Cartesian point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Joint point offset

- Function:

```
local Offset={Offset1, Offset2, Offset3, Offset4}  
RelJoint(P, Offset)
```

- Description: Set the joint offset in the Joint coordinate system to return a new joint point. The robot can move to this point only in JointMovJ command .
- Required parameter:
  - P, Indicate the current joint point, which is user-defined or obtained from the points list. Only joint point is supported.
  - {Offset1, Offset2, Offset3, Offset4}: J1 - J4 axes offset.
- Return: Joint point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# Cartesian point

- Function:

```
local P={coordinate = {x,y,z,r}, tool = 0, user = 0}
```

- Description: User-define a Cartesian point.
- Required parameter:
  - {x,y,z,r}: X, Y, Z, R axes coordinates.
  - tool: Tool coordinate system index. Value range: 0-9
  - user: User coordinate system index. Value range: 0-9

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Joint point

- Function:

```
local P={joint= {j1,j2,j3,j4}}
```

- Description: User-define a joint point.
- Required parameter: {j1,j2,j3,j4}, J1-J4 axes coordinates.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

## 6 TCP

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# Create TCP

- Function:

```
Err, Socket = TCPCreate(IsServer, IP, Port)
```

- Description: Create a TCP network. Only support a single connection.
- Required parameter:
  - IsServer: Whether to create a server. false: Create a client; true: Create a server.
  - IP: IP address of the server, which is in the same network segment of the client without conflict.
  - Port: Server port. When the robot is set as a server, **port** cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.
- Return:
  - Err:
    - 0: TCP network is created successfully.
    - 1: TCP network is created failed.
  - Socket: Socket object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Establish TCP connection

- Function:

```
TCPStart(Socket, Timeout)
```

- Description: Establish TCP connection.
- Required parameter:
  - Socket: Socket object.
  - Timeout: Wait timeout. Unit: s. If Timeout is 0, the connection is still waiting. If not, after exceeding the timeout, the connection is exited.
- Return:
  - 0: TCP connection is successful.
  - 1: Input parameters are incorrect.
  - 2: Socket object is not found.
  - 3: Timeout setting is incorrect.
  - 4: If the robot is set as a client, it indicates that the connection is wrong. If the robot is set as a server, it indicates that receiving data is wrong.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Receive TCP data

- Function:

```
Err, RecBuf = TCPRead(Socket, Timeout, Type)
```

- Description: Robot as a client receives data from a server or as a server receives data from a client .
- Required parameter:
  - Socket: Socket object.
  - Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.
  - Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string.
- Return:
  - Err:
    - 0: Receiving data is successful.
    - 1: Receiving data is failed.
  - Recbuf: Data buffer.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Send TCP data

- Function:

```
TCPWrite(Socket, Buf, Timeout)
```

- Description: Robot as a client sends data to a server or as a server sends data to a client.
- Required parameter:
  - Socket: Socket object.
  - Buf: Data sent by the robot.
  - Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Close TCP

- Function:

```
TCPDestroy(Socket)
```

- Description: Release a TCP network.
- Required parameter: Socket, Socket object.
- Return:
  - 0: Releasing TCP is successful.
  - 1: Releasing TCP is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# 7 UDP

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# Create UDP

- Function:

```
Err, Socket = UDPCreate(IsServer, IP, Port)
```

- Description: Create a UDP network. Only support a single connection.
- Required parameter:
  - IsServer: Whether to create a server. false: Create a client; true: Create a server.
  - IP: IP address of the server, which is in the same network segment of the client without conflict.
  - Port: Server port. When the robot is set as a server, **port** cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.
- Return:
  - Err:
    - 0: UDP network is created successfully.
    - 1: UDP network is created failed.
    - Socket: Socket object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Receive UDP data

- Function:

```
Err, RecBuf = UDPRead(Socket, Timeout, Type)
```

- Description: Robot as a client receives data from a server or as a server receives data from a client .
- Required parameter:
  - Socket: Socket object.
  - Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.
  - Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string.
- Return:
  - Err:
    - 0: Receiving data is successful.
    - 1: Receiving data is failed.
  - Recbuf: Data buffer.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Send UDP data

- Function:

```
UDPWrite(Socket, Buf, Timeout)
```

- Description: Robot as a client sends data to a server or as a server sends data to a client.
- Required parameter:
  - Socket: Socket object.
  - Buf: Data sent by the robot.
  - Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# 8 Modbus

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-25 09:35:43

# Create Modbus master station

- Function:

```
ModbusCreate()
```

- Description: create Modbus master station, and establish connection with the slave station

- Parameter:

- IP: IP address of slave station
- port: slave station port
- slave\_id: ID of slave station

- Return:

- err:
  - 0: Modbus master station is created successfully
  - 1: Modbus master station fails to be created
- id: device ID of slave station, supporting at most five devices, range: 0~4

Note: When ip, port, slave\_id is void, or ip is 127.0.0.1 or 0.0.0.1, connect the Modbus slave station. For example, if you input any one of the following commands, it indicates connecting Modbus slave station.

- ModbusCreate()
- ModbusCreate("127.0.0.1")
- ModbusCreate("0.0.0.1")
- ModbusCreate("127.0.0.1",xxx,xxx) //xxx arbitrary value
- ModbusCreate("0.0.0.1",xxx,xxx) //xxx arbitrary value

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-17 12:41:48

## Disconnect with Modbus slave

- Function:

```
ModbusClose()
```

- Description: disconnect with Modbus slave station
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
- Return:
  - 0: Modbus master station is closed successfully
  - 1: Modbus master station fails to be closed
- Example

```
err, id = ModbusCreate(ip, port, slave_id)
if err == 0 then
  coils = {0, 1, 1, 1, 0}
  SetCoils(id, 1024, #coils, coils)
  ModbusClose(id)
else
  print("Create failed:", err)
end
```

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-12 19:04:27

# Read the value from the Modbus slave coil register address

- Function:

```
GetCoils(id, addr, count)
```

- Description: read the coil value from the Modbus slave
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
  - addr: starting address of the coils to read, range: 0~4095
  - count: number of the coils to read, range: 0 to 4096-addr
- Return: coil value stored in a table, where the first value in the table corresponds to the coil value at the starting address; data type: bit
- Example

```
Read 5 coils starting at address 0
Coils = GetCoils(id,0,5)
Return:
Coils={1,0,0,0,0}
As shown in Table 16.3, it indicates that the robot is in the starting state
```

Coil register address (e.g.: PLC)	Coil register address (Robot system)	Data type	Description
00001	0	Bit	Start
00002	1	Bit	Pause
00003	2	Bit	Continue
00004	3	Bit	Stop
00005	4	Bit	Emergency stop
00006	5	Bit	Clear alarm
00007~0999	6~998	Bit	Reserved
01001~04096	999~4095	Bit	User-defined

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 16:37:48



# Set the coil register in the Modbus slave

- Function:

```
SetCoils(id, addr, count, table)
```

- Description: set the address value of coil register in the Modbus slave This command is not supported when the coil register address is from 0 to 5
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
  - Addr: starting address of the coils to set, range: 6 - 4095
  - count: number of the coils to set, range: 0 to 4096-addr
  - table: coil value, stored in a table, data type: bit
- Return: null
- Example

```
local Coils = {0,1,1,1,0}  
SetCoils(id, 1024, #coils, Coils)
```

Set 5 coils starting at address 1024.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 16:37:48

# Read the value from the Modbus slave discrete register address

- Function:

```
GetInBits(id, addr, count)
```

- Description: read the discrete input value from Modbus slave
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
  - addr: starting address of the discrete inputs to read, range: 0~4095
  - count: number of the discrete inputs to read, range: 0 to 4096-addr
- Return: coil value stored in a table, where the first value in the table corresponds to the input register value at the starting address; data type: bit
- Example

```
Read 5 discrete inputs starting at address 0
inBits = GetInBits(id,0,5)
Return:
inBits = {0,0,0,1,0}
As shown in Table 17.1, it indicates the robot is in running state
```

Coil register address (e.g.: PLC)	Coil register address (Robot system)	Data type	Description
00001	0	Bit	Start
00002	1	Bit	Pause
00003	2	Bit	Continue
00004	3	Bit	Stop
00005	4	Bit	Emergency stop
00006	5	Bit	Clear alarm
00007~0999	6~998	Bit	Reserved
01001~04096	999~4095	Bit	User-defined

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-08-11 16:37:48

# Read the value from the Modbus slave input register address

- Function:

```
GetInRegs(id, addr, count, type)
```

- Description: read the input register value with the specified data type from the Modbus slave
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
  - addr: starting address of the input registers, range: 0 - 4095
  - count: number of the input registers to read, range: 0 ~ 4096-addr
  - type: data type
    - Empty: read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U16": read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U32": read 32-bit unsigned integer (four bytes, occupy two registers)
    - "F32": read 32-bit single-precision floating-point number (four bytes, occupy two registers)
    - "F64": read 64-bit double-precision floating-point number (eight bytes, occupy four registers)
- Return: value of input register stored in a table, where the first value in the table corresponds to the input register value at the starting address
- Example 1

```
data = GetInRegs(id,2048,1)
```

Read a 16-bit unsigned integer starting at address 2048.

- Example 2

```
data = GetInRegs(id, 2048, 1, "U32")
```

Read a 32-bit unsigned integer starting at address 2048.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 16:37:48

# Read the value from the Modbus slave holding register address

- Function:

```
GetHoldRegs(id, addr, count, type)
```

- Description: Read the holding register value from the Modbus slave according to the specified data type
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
  - addr: starting address of the holding registers. Value range: 0 - 4095
  - count: number of the holding registers to read. Value range: 0 to 4096-addr
  - type: data type
    - Empty: read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U16": read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U32": read 32-bit unsigned integer (four bytes, occupy two registers)
    - "F32": read 32-bit single-precision floating-point number (four bytes, occupy two registers)
    - "F64": read 64-bit double-precision floating-point number (eight bytes, occupy four registers)
- Return: coil value stored in a table, where the first value in the table corresponds to the input register value at the starting address
- Example 1:

```
data = GetHoldRegs(id,2048,1)
```

Read a 16-bit unsigned integer starting at address 2048.

- Example 2:

```
data = GetHoldRegs(id, 2048, 1, "U32")
```

Read a 32-bit unsigned integer starting at address 2048.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-08-11 16:37:48

# Set the holding register in the Modbus slave

- Function:

```
SetHoldRegs(id, addr, count, table, type)
```

- Description: set the holding register in the Modbus slave
- Parameter:
  - id: device ID of slave station, supporting at most five devices, range: 0~4
  - addr: starting address of the holding registers to set, range: 0 - 4095
  - count: number of the holding registers to set, range: 0 to 4096-addr
  - table: holding register value, stored in a table
  - type: datatype
    - Empty: read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U16": read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U32": read 32-bit unsigned integer (four bytes, occupy two registers)
    - "F32": read 32-bit single-precision floating-point number (four bytes, occupy two registers)
    - "F64": read 64-bit double-precision floating-point number (eight bytes, occupy four registers)
- Return: null
- Example 1

```
local data = {6000}  
SetHoldRegs(id, 2048, #data, data, "U16")
```

Set a 16-bit unsigned integer starting at address 2048.

- Example 2

```
local data = {95.32105}  
SetHoldRegs(id, 2048, #data, data, "F64")
```

Set a 64-bit double-precision floating-point number starting at address 2048.

# 9 Conveyor Tracking

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

## Set conveyor number to create a tracing queue

- Function:

```
CnvVison(CnvID)
```

- Description: Set conveyor number to create a tracing queue.
- Required parameter: CnvID, Conveyor number. Only support single conveyor.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

## Obtain status of the object

- Function:

```
GetCnvObject(CnvID, ObjID)
```

- Description: Obtain the information of the part on the conveyor to check whether the part is in the pickup area .
- Required parameter:
  - CnvID: Conveyor index.
  - ObjID: Part index.
- Return:
  - Part status: Whether there is a part. Value range: true or false
  - Part type
  - Part coordinate (x,y,r)

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# Set X,Y axes offset under the set User coordinate system

- Function:

```
SetCnvPointOffset(OffsetX,OffsetY)
```

- Description: Set X,Y axes offset under the set User coordinate system.
- Required parameter:
  - OffsetX: X-axis offset.
  - OffsetY: Y-axis offset.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:27:19

# Set time compensation

- Function:

```
SetCnvTimeCompensation(Time)
```

- Description: Set time compensation. This command is used for compensating the pick-up position offset in the moving direction of the conveyor which is caused by taking photos with a time delay.
- Required parameter: Time, time-offset. Unit: ms
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Synchronize the specified conveyor

- Function:

```
SyncCnv(CnvID)
```

- Description: Synchronize the specified conveyor. The motion commands used between `SyncCnv(CnvID)` and `StopSyncCnv(CnvID)` only support MovL command.
- Required parameter: CnvID, Conveyor index.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 17:14:42

# Stop synchronous conveyor

- Function:

```
StopSyncCnv(CnvID)
```

- Description: Stop synchronizing the conveyor. The other commands following this command will not be executed until this command running is completed.
- Required parameter: CnvID, Conveyor index.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# 10 Pallet

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54


# Instantiate matrix pallet

- Function:

```
Pallet = MatrixPallet (Index)
```

Or:

```
local Option={IsUnstack= true, User= 1}  
Pallet = MatrixPallet (Index, Option)
```

- Description: Instantiate matrix pallet.
- Required parameter: Index: Matrix pallet index.
- Optional parameter: {IsUnstack= true, User= 1}. You can double-click  to insert the command with optional parameters.
  - IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode
  - User: User coordinate system index. If not set, the default is User 0 coordinate system.
- Return: Matrix pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-10-28 17:01:02


# Instantiate teaching pallet

- Function:

```
Pallet = TeachPallet (Index)
```

Or:

```
local Option={IsUnstack= true, User= 1}  
Pallet = TeachPallet (Index,Option)
```

- Description: Instantiate teaching pallet.
- Required parameter: Index: Teaching pallet index.
- Optional parameter: {IsUnstack= true, User= 1}. You can double-click  to insert the command with optional parameters.
  - IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode
  - User: User coordinate system index. If not set, the default is User 0 coordinate system.
- Return: Teaching pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-10-28 17:15:51

## Set the next stack index which is to be operated

- Function:

```
SetPartIndex(Pallet, Index)
```

- Description: Set the next stack index which is to be operated.
- Required parameter:
  - Pallet: Pallet object.
  - Index: The next stack index. Initial value: 0

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# Get the current operated stack index

- Function

```
GetPartIndex(Pallet)
```

- Description: Get the current operated stack index.
- Required parameter: Pallet, Pallet object.
- Return: The current operated stack index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Set the next pallet layer index which is to be operated

- Function:

```
SetLayerIndex(Pallet, Index)
```

- Description: Set the next pallet layer index which is to be operated.
- Required parameter:
  - Pallet: Pallet object.
  - Index: The next pallet layer index. Initial value: 0

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

## Get the current pallet layer index

- Function:

```
GetLayerIndex(Pallet)
```

- Description: Get the current pallet layer index.
- Required parameter: Pallet, Pallet object.
- Return: The current pallet layer index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

# Reset pallet

- Function:

```
Reset(Pallet)
```

- Description: Reset pallet.
- Required parameter: Pallet, Pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-06-04 15:51:09

# Check whether the stack assembly or dismantling is complete

- Function:

```
IsDone(Pallet)
```

- Description: Check whether the stack assembly or dismantling is complete.
- Required parameter: Pallet, Pallet object.
- Return:
  - true: Finished.
  - false: Un-finished.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53

## Release palletizing instance

- Function:

```
Release(Pallet)
```

- Description: Release palletizing instance.
- Required parameter: Pallet, Pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# The robot moves from the current position to the first stack position as the configured stack assembly path

- Function:

```
PalletMoveIn(Pallet)
```

Or:

```
local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}  
PalletMoveIn(Pallet, Option)
```

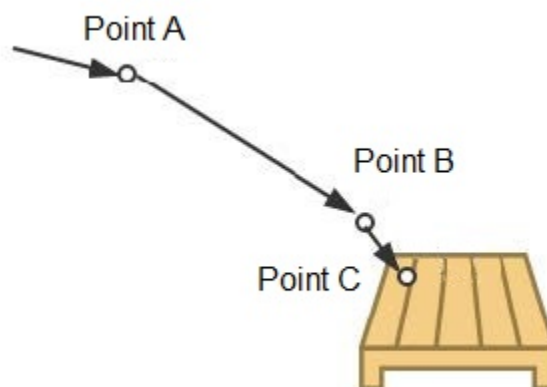
- Description: The robot moves from the current position to the first stack position as the configured stack assembly path.
- Required parameter: Pallet, Pallet object.
- Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click

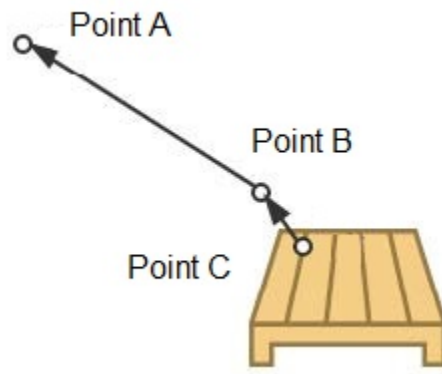


to insert the command with optional parameters.

- SpeedAB: Velocity rate when the robot moves from the transition point to the preparation point. Value range: 1-100
- SpeedBC: Velocity rate when the robot moves from the preparation point to the first stack point. Value range: 1-100
- AccAB: Acceleration rate when the robot moves from the transition point to the preparation point. Value range: 1-100
- AccBC: Acceleration rate when the robot moves from the preparation point to the first stack point. Value range: 1-100
- CP: Continuous path rate. Value range: 0-100

The stack assembly path and dismantling path are shown as follows. Point A is the transition point, which is fixed or varies with the pallet layer. Point B is the preparation point which is calculated by the target point and the set offset. Point C is the first stack point.





Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:53



# The robot moves from the current position to the transition point as the configured stack dismantling path

- Function:

```
PalletMoveOut(Pallet)
```

Or:

```
local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}  
PalletMoveOut(Pallet, Option)
```

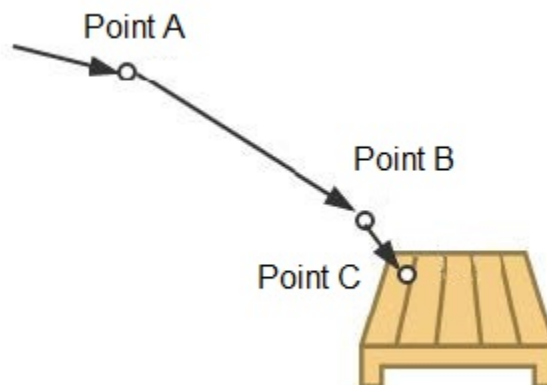
- Description: The robot moves from the current position to the transition point as the configured stack dismantling path.
- Required parameter: Pallet, Pallet object.
- Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click

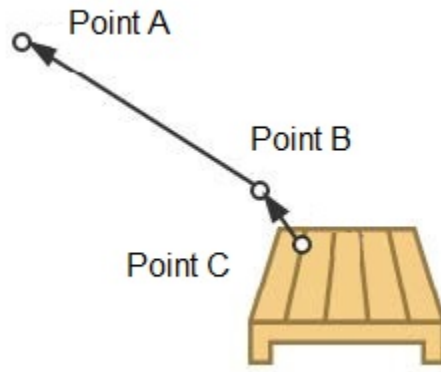


to insert the command with optional parameters.

- SpeedAB: Velocity rate when the robot moves from the preparation point to the transition point. Value range: 1-100
- SpeedBC: Velocity rate when the robot moves from the first stack point to the preparation point. Value range: 1-100
- AccAB: Acceleration rate when the robot moves from the preparation point to the transition point. Value range: 1-100
- AccBC: Acceleration rate when the robot moves from the first stack point to the preparation point. Value range: 1-100
- CP: Continuous path rate. Value range: 0-100

The stack assembly path and dismantling path are shown as follows. Point A is the transition point, which is fixed or varies with the pallet layer. Point B is the preparation point which is calculated by the target point and the set offset. Point C is the first stack point.





Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-01-26 16:58:54

# Vision

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-09-16 12:23:42

# Initialize the connection to the camera

- Function:

```
InitCam ("CAM0")
```

- Description: Initialize the connection to the camera.
- Parameter:
  - CAM0: Name of the camera
- Return:
  - 0: Initialize successfully
  - 1: Failed to initialize

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-09-16 16:35:18

# Trigger the camera to take a picture

- Function:

```
TriggerCam ("CAM0")
```

- Description: Trigger the camera to take a picture.
- Parameter:
  - CAM0: Name of the camera
- Return:
  - 0: Trigger successfully
  - 1: Fail to trigger

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-09-16 15:30:25

## Send data

- Function:

```
SendCam ("CAM0", "0,0,0,0")
```

- Description: Send data to the camera.
- Parameter:
  - CAM0: Name of the camera
  - "0, 0, 0, 0": Data
- Return:
  - 0: Send successfully
  - 1: Failed to send

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-09-16 15:40:07

# Receive data

- Function:

```
RecvCam ("CAM0", "number")
```

- Description: Receive data from the camera.
- Parameter:
  - CAM0: Name of the camera
  - number: Data type, value range: number or string
- Return:
  - err:
    - 0: Receive data correctly
    - 1: Time out
    - 2: The data format is incorrectly and cannot be parsed.
    - 3: Network disconnection
  - n: The number of data groups sent by the camera.
  - data: The data sent by the camera is stored in a two-dimensional array.
- Example:

```
InitCam("CAM0")
SendCam("CAM0", "0,0,0,0;")
while true do
  local err,n,data = RecvCam("CAM0")
  if err == 0 then
    for i = 1, n do
      pos.x=data[i][1] --data[i][1]assign to pos.x
      pos.y=data[i][2] --data[i][2]assign to pos.y
      pos.c=data[i][3] --data[i][3]assign to pos.r
      pos.z = -20 --Set the height according to the actual situation
      Move(pos)
    end
  elseif err == 1 then
    print("Data receive timeout")
    break
  elseif err == 2 then
    print("The data format is incorrectly and cannot be parsed")
    break
  elseif err == 3 then
    print("Network disconnection")
    break
  end
end
end
DestroyCam("CAM0")
```





## Close the camera

- Function:

```
DestroyCam ("CAM0")
```

- Description: Release the connection with the camera
- Parameter:
  - CAM0: Name of the camera
- Return:
  - 0: Send successfully
  - 1: Failed to send

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision:  
2021-09-16 16:15:56

# Table of Contents

Introduction	1.1
1. Overview	1.2
2. Introduction of Commands	1.3
Control Commands	1.3.1
2.1 Pause program command	1.3.1.1
2.2 Set the delay time command	1.3.1.2
2.3 Wait command	1.3.1.3
2.4 Custom Script	1.3.1.4
2.5 Get the current time	1.3.1.5
Variables Commands	1.3.2
2.6 Make a variable	1.3.2.1
2.7 Variable assignment	1.3.2.2
2.8 Modify variables	1.3.2.3
2.9 Make a list	1.3.2.4
2.10 Add a variable to the list	1.3.2.5
2.11 Deletes an item	1.3.2.6
2.12 Delete all items	1.3.2.7
2.13 Insert an item	1.3.2.8
2.14 Replace an item	1.3.2.9
2.15 Get an item	1.3.2.10
2.16 Get the total number of items	1.3.2.11
Motion Commands	1.3.3
2.17 Advanced configuration	1.3.3.1
2.18 Move command	1.3.3.2
2.19 Offset move command	1.3.3.3
2.20 Jump Movement, Jump parameters can be set in this command	1.3.3.4
2.21 Jump Movement, Jump parameters are called by Arch index	1.3.3.5
2.22 Circle command	1.3.3.6
2.23 Arc command	1.3.3.7
Move Arguments Commands	1.3.4
2.24 Set the joint acceleration rate	1.3.4.1
2.25 Set the joint velocity rate	1.3.4.2
2.26 Set the Cartesian acceleration rate	1.3.4.3
2.27 Set the Cartesian velocity rate	1.3.4.4
2.28 Set the continuous path rate	1.3.4.5

2.29 Stop at this point	1.3.4.6
2.30 Set Load Parameters	1.3.4.7
Posture Commands	1.3.5
2.31 Gets the value of the current Cartesian position	1.3.5.1
2.32 Gets a value of the current Cartesian position	1.3.5.2
2.33 Gets the value of the current Joint position	1.3.5.3
2.34 Gets a value of the current Joint position	1.3.5.4
2.35 Cartesian position offset	1.3.5.5
2.36 Angle position offset	1.3.5.6
2.37 Custom Cartesian point	1.3.5.7
2.38 Custom joint point	1.3.5.8
2.39 Get coordinates	1.3.5.9
2.40 Gets a coordinate value of a point	1.3.5.10
I/O Commands	1.3.6
2.41 Get digital input	1.3.6.1
2.42 Set digital output	1.3.6.2
2.43 Set digital output immediately command	1.3.6.3
Modbus Commands	1.3.7
2.44 Creates a Modbus master command	1.3.7.1
2.45 Get the connection result command	1.3.7.2
2.46 Get input register command	1.3.7.3
2.47 Get holding register command	1.3.7.4
2.48 Get discrete input register command	1.3.7.5
2.49 Get coil register command	1.3.7.6
2.50 Get multiple coil register command	1.3.7.7
2.51 Set coil register command	1.3.7.8
2.52 Set multiple coil register values command	1.3.7.9
2.53 Set holding register command	1.3.7.10
2.54 Close Modbus master command	1.3.7.11
TCP Commands	1.3.8
2.55 Open socket command	1.3.8.1
2.56 Get open socket command	1.3.8.2
2.57 Create socket command	1.3.8.3
2.58 Get create socket command	1.3.8.4
2.59 Close socket command	1.3.8.5
2.60 Get variable command	1.3.8.6
2.61 Socket send variable command	1.3.8.7
2.62 Get socket send result command	1.3.8.8
Vision Command	1.3.9

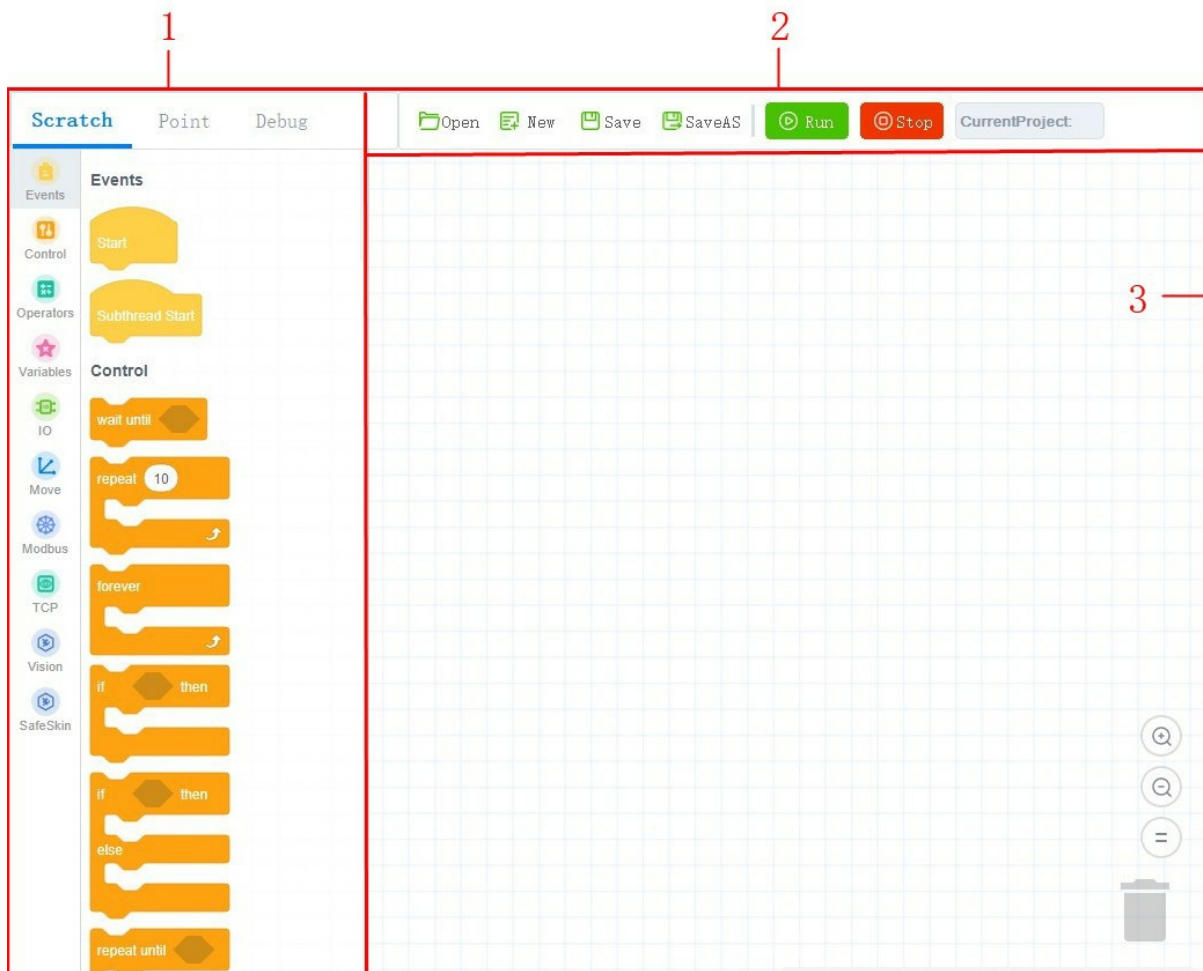
2.63 Connect camera command	1.3.9.1
2.64 Get connect result command	1.3.9.2
2.65 Trigger the camera command	1.3.9.3
2.66 Send data to the camera command	1.3.9.4
2.67 Receive data from camera command	1.3.9.5
2.68 Get the number of data groups command	1.3.9.6
2.69 Gets a data command	1.3.9.7
2.70 Close camera command	1.3.9.8
Pallet Command	1.3.10
2.71 Instantiate matrix pallet	1.3.10.1
2.72 Set the next stack index which is to be operated	1.3.10.2
2.73 Get the current operated stack index	1.3.10.3
2.74 Set the next pallet layer index which is to be operated	1.3.10.4
2.75 Get the current pallet layer index	1.3.10.5
2.76 Reset pallet	1.3.10.6
2.77 Check whether the stack assembly or dismantling is complete	1.3.10.7
2.78 Release palletizing instance	1.3.10.8
3. Description of Programming	1.4
3.1 Basic operation	1.4.1
3.2 Teaching points	1.4.2
3.3 Quick Start	1.4.3
3.3.1 Robot movement	1.4.3.1
3.3.2 I/O Setting	1.4.3.2
3.3.3 Register setting and reading	1.4.3.3
3.3.4 Create TCP Client	1.4.3.4
3.3.5 Create TCP Server	1.4.3.5
3.3.6 Vision Interaction	1.4.3.6





# Introduction






Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-24 18:21:51

# 1. Overview

Blockly is a kind of building block programming. You can write programs by block to quickly and conveniently control the robot. The blockly panel is shown in the following figure, and the description of blockly panel is listed in the following table.



No.	Description
1	<p>Block area</p> <p><b>Scratch</b> : Provide all blocks</p> <p><b>Point</b> : Save teaching point that can be called when writing a program</p> <p><b>Debug</b> : View the Lua instruction corresponding to the blocks</p>
2	<p>Menu bar</p> <p> <b>Open</b> : Open a project</p> <p> <b>New</b> : Create a new project</p> <p> <b>Save</b> : Save the project</p> <p> <b>SaveAS</b> : Save the current project with a new name</p>

	 : Start running the program in the current code area.  : Stop the running program
3	<p>Code area</p> <p>Drag block to this page and edit it. Click the icon   in the code area to zoom in, zoom out and restore the blocks,  can be used to delete the selected block.</p>

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-14 18:07:02

## 2. Introduction of Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

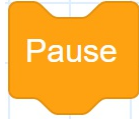


# Control Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.1 Pause program command

- Function:



- Description: When the program runs to this command, pause the running program.
- Parameter: None
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:12:17

## 2.2 Set the delay time command

- Function:

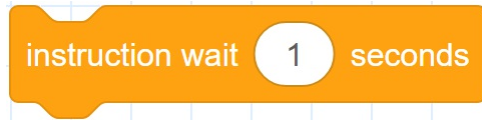


- Description: Set the delay time for all commands.
- Parameter: Time: Delay time; Unit: second
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:12:36

## 2.3 Wait command

- Function:

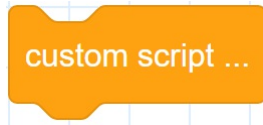


- Description: Set the delay time for robot motion commands.
- Parameter: time: Delay time; Unit: second
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:12:42

## 2.4 Custom Script

- Function:



- Description: Double-click the block, a script editing window will pop up. You can write Lua programs as required. After writing, click "Save".
- Parameter: None
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:12:50

## 2.5 Get the current time

- Function:



- Description: Gets the current time.
- Parameter: None
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:52:16

# Variables Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.6 Make a variable

- Function:



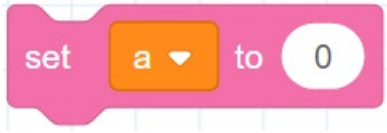
- Description: Make a variable.
- Parameter: Set variable name. The variable name must start with a letter and cannot contain special characters such as Spaces.
- Return: a variable

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:56:11



## 2.7 Set the value of a variable

- Function:

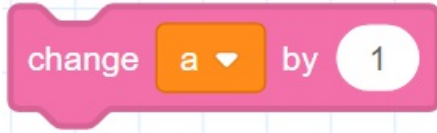


- Description: Set the value of a variable.
- Parameter:
  - Name: name of a variable
  - parameter: value of a variable
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:48:09

## 2.8 Modify the value of a variable

- Function:

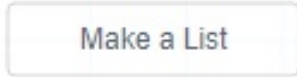


- Description: Modify the value of a variable.
- Parameter:
  - Name: name of a variable
  - parameter: The value of an increase or decrease
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 17:56:35

## 2.9 Make a list

- Function:

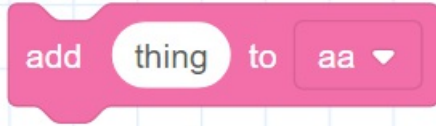


- Description: Make a list. When there are many variables, you can use lists to manage variables.
- Parameter: Set list name. The list name must start with a letter and cannot contain special characters such as Spaces.
- Return: a list

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:00:38

## 2.10 Add a variable to the list

- Function:



- Description: Add a variable to the list.
- Parameter:
  - variable: Set a variable
  - list: Select a list
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:10:01

## 2.11 Deletes an item

- Function:



- Description: Delete an item from the list.
- Parameter:
  - Index: Set specified item
  - list: Select a list
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:14:16

## 2.12 Delete all items

- Function:



- Description: Delete all items.
- Parameter: Select a list.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:15:18

## 2.13 Insert an item

- Function:



- Description: Inserts an item before an item in the list.
- Parameter:
  - Content: Set the content to be inserted.
  - Index: Set the specified item.
  - List: Select a list.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:22:43

## 2.14 Replace an item

- Function:



- Description: Replace the contents of an item in the list.
- Parameter:
  - Index: Set the specified item.
  - List: Select a list.
  - Content: Set the content to be inserted.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:22:35



## 2.15 Get an item

- Function:



- Description: Get a variable in the list.
- Parameter:
  - Index: Set the specified item.
  - List: Select a list.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:53:04

## 2.16 Get the total number of items

- Function:



- Description: Get the total number of items.
- Parameter: Select a list.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:53:17

# Motion Commands

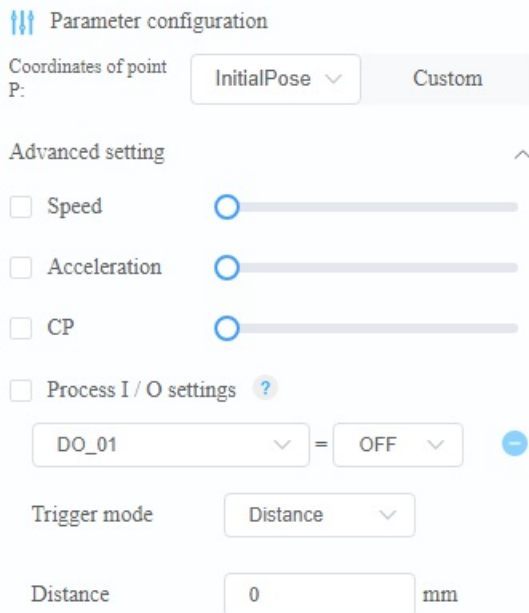
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.17 Advanced configuration

- Function:



- Description: Click Advanced configuration, on the displayed **Settings panel**, select a motion type and set basic and advanced parameters (optional).
- Type 1: MovJ: Move from the current position to a target position in a point-to-point mode under the Cartesian coordinate system.



- Parameter configuration
  - Coordinate of point P: Indicate target point, which is user-defined or obtained from the **Point** page. Only Cartesian coordinate points are supported.
- Advanced setting
  - Speed: Velocity rate. Value range: 1 - 100
  - Acceleration: Acceleration rate. Value range: 1 - 100
  - CP: Whether to set continuous path function. Value range: 0 - 100
  - Process I/O settings
    - DO: Output port of CR
    - Trigger mode: Distance or Percentage. Distance refers to the distance away from the starting point or target point. Percentage refers to the distance percentage between the starting point and the target point. range: 0~100
    - Distance: If the Distance value is positive, it refers to the distance away from the starting point; If the Distance value is negative, it refers to the distance away from the target point.
- Type 2: MovL: Move from the current position to a target position in a straight line under the Cartesian coordinate system.

Parameter configuration

Coordinates of point P: InitialPose Custom

Advanced setting ^

Speed ○ —————

Acceleration ○ —————

CP ○ —————

Process I / O settings ?

DO\_01 = OFF —

Trigger mode Distance

Distance 0 mm

- Parameter configuration
  - Coordinate of point P: Indicate target point, which is user-defined or obtained from the **Point** page. Only Cartesian coordinate points are supported.
- Advanced setting
  - Speed: Velocity rate. Value range: 1 - 100
  - Acceleration: Acceleration rate. Value range: 1 - 100
  - CP: Whether to set continuous path function. Value range: 0 - 100
  - Process I/O settings
    - DO: Output port of CR
    - Trigger mode: Distance or Percentage. Distance refers to the distance away from the starting point or target point. Percentage refers to the distance percentage. between the starting point and the target point. range: 0~100.
    - Distance: If the Distance value is positive, it refers to the distance away from the starting point; If the Distance value is negative, it refers to the distance away from the target point.
- Type 3: Jump: The robot moves from the current position to a target position in the **Move** mode. The trajectory looks like a door.

Parameter configuration

Coordinates of point P: InitialPose Custom

Raise height h1 10 mm

Descent height h2 20 mm

Max height z\_limit 100 mm

Advanced setting ^

Speed ○ —————

Acceleration ○ —————

- Parameter configuration
  - Coordinate of point P: Indicate target point, which is user-defined or obtained from the **Point** page. Only Cartesian coordinate points are supported.
  - Raise height h1: Lifting height
  - Descent height h2: Dropping height
  - Max height z\_limit: Maximum lifting height
- Advanced setting
  - Speed: Velocity rate. Value range: 1 - 100
  - Acceleration: Acceleration rate. Value range: 1 - 100
- Type 4: JointMovJ: Move from the current position to a target position in a point-to-point motion under the Joint coordinate system.

Parameter configuration

Coordinates of point P: InitialPose Custom

j1	0.000	j3	0.000
j2	0.000	j4	0.000

Get coordinates

Advanced setting

Speed ○

Acceleration ○

CP ○

- Parameter configuration
  - J1~J4: Indicate the joint angle of the target point
- Advanced setting
  - Speed: Velocity rate. Value range: 1 - 100
  - Acceleration: Acceleration rate. Value range: 1 - 100
  - CP: Whether to set continuous path function. Value range: 0 - 100
- Type 5: ReMovJ: Move from the current position to the offset position in a point-to-point mode under the Cartesian coordinate system.

Parameter configuration

Offset

$\Delta X$   mm  $\Delta Z$   mm

$\Delta Y$   mm  $\Delta R$   mm

Advanced setting

Speed ○

Acceleration ○

CP ○

- Parameter configuration
  - Offset: X, Y, Z, R axes offset in the Cartesian coordinate system. Unit: mm
- Advanced setting
  - Speed: Velocity rate. Value range: 1 - 100
  - Acceleration: Acceleration rate. Value range: 1 - 100
  - CP: Whether to set continuous path function. Value range: 0 - 100

- Type 6: RelMovL: Move from the current position to the offset position in a straight line under the Cartesian coordinate system.

Parameter configuration

Offset

ΔX 0 mm ΔZ 0 mm

ΔY 0 mm ΔR 0 mm

Advanced setting

Speed

Acceleration

CP

- Parameter configuration
    - Offset: X, Y, Z, R axes offset in the Cartesian coordinate system. Unit: mm
  - Advanced setting
    - Speed: Velocity rate. Value range: 1 - 100
    - Acceleration: Acceleration rate. Value range: 1 - 100
    - CP: Whether to set continuous path function. Value range: 0- 100
- Type 7: Arc: Move from the current position to a target position in an arc interpolated mode under the Cartesian coordinate system. This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory.

Parameter configuration

Intermediate point A coordinate: InitialPose Custom

End point B coordinate: InitialPose Custom

Advanced setting

Speed

Acceleration

CP

- Parameter configuration
    - Intermediate point A coordinate: Middle point, which is user-defined or obtained from the **Point** page.
    - End point B coordinate: End point, which is user-defined or obtained from the **Point** page.
  - Advanced setting
    - Speed: Velocity rate. Value range: 1 - 100
    - Acceleration: Acceleration rate. Value range: 1 - 100
    - CP: Whether to set continuous path function. Value range: 0 - 100
- Type 8: Circle: Move from the current position to a target position in a circular interpolated mode under the Cartesian coordinate system. This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory.

Parameter configuration

Intermediate point A coordinate: InitialPose Custom

End point B coordinate: InitialPose Custom

Number of cycles: 1

Advanced setting

Speed

Acceleration

CP

- Parameter configuration
    - Intermediate point A coordinate: Middle point, which is user-defined or obtained from the **Point** page.

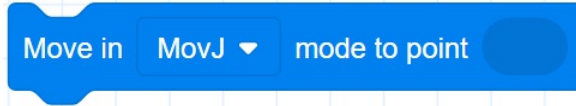
- End point B coordinate: End point, which is user-defined or obtained from the **Point** page.
- Number of cycles: Number of circles.
- Advanced setting
  - Speed: Velocity rate. Value range: 1 - 100
  - Acceleration: Acceleration rate. Value range: 1 - 100
  - CP: Whether to set continuous path function. Value range: 0 - 100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 15:11:14



## 2.18 Move command

- Function:

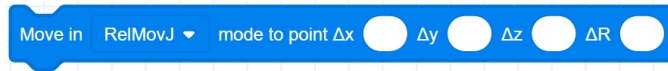


- Description: Move from the current position to a target position in straight line mode or point to point mode.
- Parameter:
  - Mode:
    - MovL: straight line mode, the target point is the Cartesian point.
    - MovJ: point to point mode, the target point is the Cartesian point.
    - JointMovJ: point to point mode, the target point is the Cartesian point.
  - point: Indicate target point, which is obtained from the TeachPoint page.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:41:23

## 2.19 Offset move command

- Function:



- Description: Move the corresponding offset in X, Y and Z directions from the current position in straight line mode or point to point mode.
- Parameter:
  - Mode:
    - RelMovL: straight line mode
    - RelMovJ: point to point mode
  - $\Delta x$ : Indicate offset of X axis
  - $\Delta y$ : Indicate offset of Y axis
  - $\Delta z$ : Indicate offset of Z axis
  - $\Delta r$ : Indicate offset of R axis
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 18:45:37

## 2.20 Jump Movement, Jump parameters can be set in this command

- Function:

Move in Jump mode to point  Raise height h1  mm Descent height h2  mm Max height z\_limit  mm

- Description: Jump Movement. The jump parameters can be set in this command.
- Parameter:
  - point: Set the target point. Only Cartesian point is supported.
  - h1: Lifting height form starting point.
  - h2: Dropping height.
  - z\_limit: Lifting height h1 and dropping height h2 cannot be greater than z\_limit, otherwise the alarm will be triggered.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 19:03:22

## 2.21 Jump Movement, Jump parameters are called by Arch index

- Function:

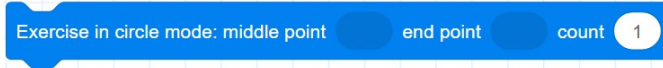


- Description: Jump Movement. The jump parameters are called by Arch index.
- Parameter:
  - point: Set the target point. Only Cartesian point is supported.
  - index: Arch index. Value range: 0 - 9. Please set Jump parameters on the System > Parameters > RobotParams > PlayBackArch page.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 19:08:39

## 2.22 Circle command

- Function:

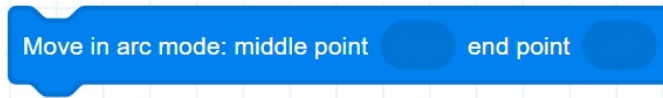


- Description: Move from the current position to a target position in a circular interpolated mode.
- Parameter:
  - middle point: Indicate middle point, which is obtained from the TeachPoint page.
  - end point: Indicate end point, which is obtained from the TeachPoint page.
  - count: number of whole circles, value range: 1 ~ 999.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:53:49

## 2.23 Arc command

- Function:



- Description: Move from the current position to a target position in an arc interpolated mode.
- Parameter:
  - middle point: Indicate middle point, which is obtained from the TeachPoint page.
  - end point: Indicate end point, which is obtained from the TeachPoint page.
- Return: None

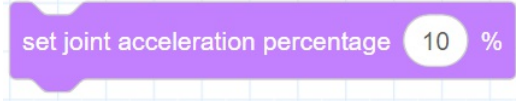
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:53:57

# Move Arguments Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.24 Set the joint acceleration rate

- Function:



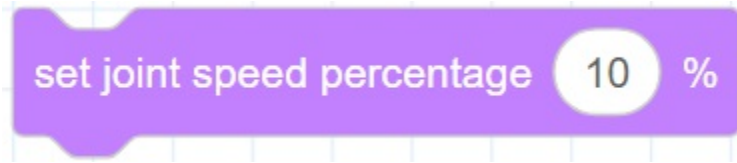
- Description: Set the joint acceleration rate.
- Parameter: Joint acceleration rate, range: 0~100.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:54:05



## 2.25 Set the joint velocity rate

- Function:

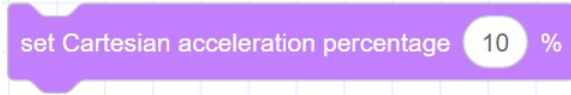


- Description: Set the joint velocity rate.
- Parameter: Joint speed ratio, range: 0~100.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 19:15:42

## 2.26 Set the Cartesian acceleration rate

- Function:

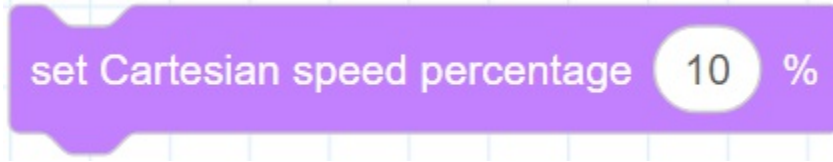


- Description: Set the Cartesian acceleration rate.
- Parameter: Acceleration rate, range: 0~100.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:54:38

## 2.27 Set the Cartesian velocity rate

- Function:



- Description: Set the Cartesian velocity rate.
- Parameter: Velocity ratio, range: 0~100.
- Return: None

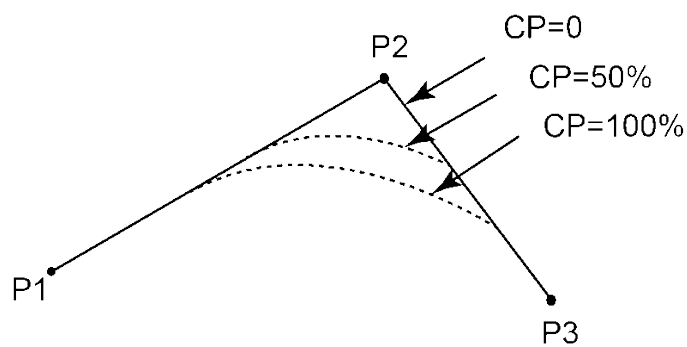
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 19:21:26

## 2.28 Set the smooth path rate

- Function:

set smooth transition percentage 10 %

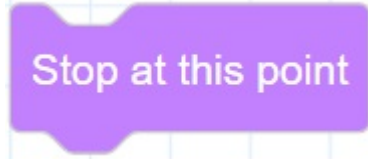
- Description: Set the smooth path rate. when reaching the end point from the starting point through the intermediate point, whether the transition through the intermediate point is in a right Angle mode or in a curve mode, as shown in the following figure.
- Parameter: Smooth path ratio, range: 0~100.
- Return: None



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:54:28

## 2.29 Stop at this point

- Function:



- Description: Block the program from executing queue instructions, and return after all the instructions are executed.
- Parameter: None
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 19:42:46

## 2.30 Set Load Parameters

- Function:

Set Payload Parametes: Payload  g X-offset  mm Y-Offset  mm Servo Index(Optional)

- Description: Set payload, X-axis offset, Y-axis offset and servo index.
- Required parameter:
  - payload: Payload. Value range: 0- 750. Unit: g.
  - X-Offset: Offset in X-axis.
  - Y-Offset: Offset in Y-axis.
  - Optional parameter: index, servo parameter index. The default value range is 1 - 10.
- Return: None

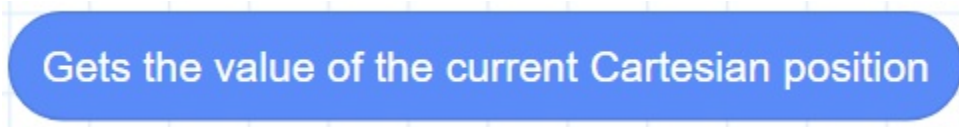
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 19:46:11

# Posture Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.31 Gets the value of the current Cartesian position

- Function:



- Description: Get the current pose of the robot under the Cartesian coordinate system.
- Parameter: None
- Return: The cartesian coordinate of the current position

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:08:50



## 2.32 Gets a value of the current Cartesian position

- Function:

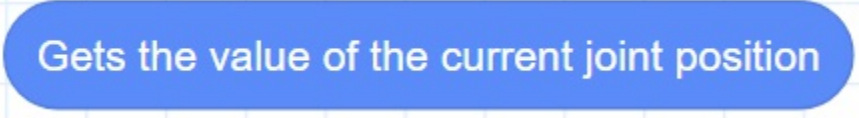
Gets the X value of the current Cartesian position

- Description: Get a coordinate value of the robot under the Cartesian coordinate system. If you have set the User or Tool coordinate system, get the index of User or Tool coordinate system.
- Parameter: Select an axis or coordinate system.
- Return: a coordinate value or index of coordinate system.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:55:06

## 2.33 Gets the value of the current Joint position

- Function:



Gets the value of the current joint position

- Description: Get the current pose of the robot under the Joint coordinate system.
- Parameter: None
- Return: Joint coordinates of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:10:49

## 2.34 Gets a value of the current Joint position

- Function:



- Description: Get a coordinate value of the robot under the Joint coordinate system.
- Parameter: Select J1~J4.
- Return: The Joint coordinate of the current position.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:14:04

## 2.35 Cartesian position offset

- Function:



- Description: Set the X, Y, Z, R axes offset under the Cartesian coordinate system to return a new Cartesian coordinate point.
- Parameter:
  - $\Delta x$ : Indicate offset of X axis
  - $\Delta y$ : Indicate offset of Y axis
  - $\Delta z$ : Indicate offset of Z axis
  - $\Delta r$ : Indicate offset of R axis
  - position: Indicate the current Cartesian point.
- Return: Cartesian point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:49:16

## 2.36 Angle position offset

- Function:

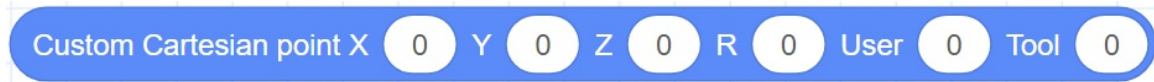


- Description: Set the joint offset in the Joint coordinate system to return a new joint point.
- Parameter:
  - $\Delta J1 \sim \Delta J4$ : Indicate J1 - J4 axes offset
  - position: Indicate the current joint point.
- Return: Joint point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:19:41

## 2.37 Custom Cartesian point

- Function:



- Description: User-define a Cartesian point.
- Parameter:
  - {X, Y, Z, R}: Indicate X, Y, Z, R axes coordinates.
  - User: User coordinate system index. Value range: 0-9.
  - Tool: Tool coordinate system index. Value range: 0-9.
- Return: Cartesian point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:55:28

## 2.38 Custom joint point

- Function:



- Description: User-define a joint point.
- Parameter: J1- J4 axes coordinates.
- Return: Joint point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:24:13

## 2.39 Get coordinates

- Function:



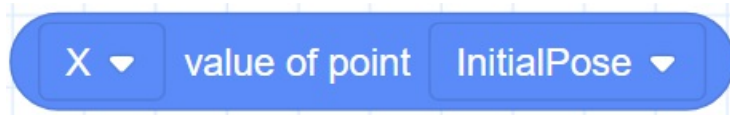
- Description: Gets the coordinates of the target point.
- Parameter: Select the target point from the **Point** page.
- Return: Coordinates of the target point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:27:08



## 2.40 Gets a coordinate value of a point

- Function:



- Description: Gets a coordinates of the target point. If you have set the User or Tool coordinate system, get the index of User or Tool coordinate system.
- Parameter:
  - axis: Select an axis or coordinate system
  - point: Select the target point from the **Point** page.
- Return: Coordinates of the target point or index of coordinate system.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:50:37

# I/O Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:17:56

## 2.41 Get digital input

- Function:



- Description: Get the value of digital input port.
- Parameter: Select digital input port.
- Return: The value of digital input port.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:30:45

## 2.42 Set digital output

- Function:



- Description: Set the status of digital output port.
- Parameter:
  - DO: Digital output index.
  - Status: set the DO to on or off.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:32:41

## 2.43 Set digital output immediately command

- Function:

set the status of digital output (immediate) DO\_1 ▼ to ON ▼

- Description: Without entering the queue command, set the status of digital output port immediately after pre-reading.
- Parameter:
  - Control end: controller or tool.
  - DO: Digital output index.
  - Status: Set the DO to on or off.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:35:42

# Modbus Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.44 Creates a Modbus master command

- Function:

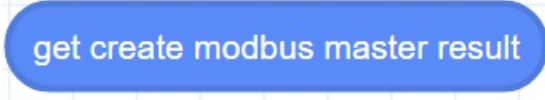


- Description: Creates a Modbus master to connect to a Modbus slave.
- Parameter:
  - IP: indicate IP address of the Modbus slave.
  - Port: indicate the port of the Modbus slave station.
  - ID: indicate ID of the Modbus slave, the value range is 1 to 4.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:56:08

## 2.45 Get the connection result command

- Function:



get create modbus master result

- Description: Get the connection result.
- Parameter: None.
- Return:
  - 0: Modbus master is created successfully.
  - 1: Modbus master is created failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:45:53



## 2.46 Get input register command

- Function:

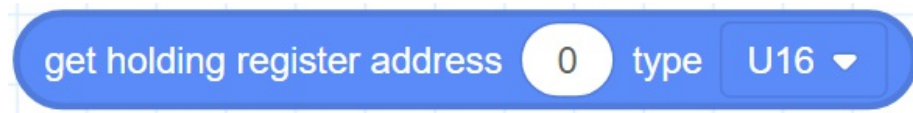


- Description: Read the input register value with the specified data type from the Modbus slave.
- Parameter:
  - Address: Starting address of the input registers. Value range: 0 – 4095.
  - type: Data type
    - “U16”: Read 16-bit unsigned integer ( two bytes, occupy one register)
    - “U32”: Read 32-bit unsigned integer (four bytes, occupy two registers)
    - “F32”: Read 32-bit single-precision floating-point number (four bytes, occupy two registers)
    - “F64”: Read 64-bit double-precision floating-point number (eight bytes, occupy four registers).
- Return: The input register value

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:46:01

## 2.47 Get holding register command

- Function:

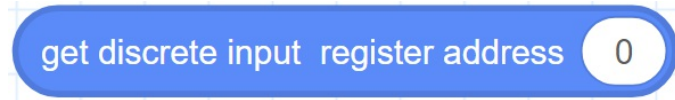


- Description: Read the holding register value from the Modbus slave according to the specified data type.
- Parameter:
  - Address: starting address of the holding registers. Value range: 0 - 4095.
  - type: Data type
    - "U16": Read 16-bit unsigned integer ( two bytes, occupy one register)
    - "U32": Read 32-bit unsigned integer (four bytes, occupy two registers)
    - "F32": Read 32-bit single-precision floating-point number (four bytes, occupy two registers)
    - "F64": Read 64-bit double-precision floating-point number (eight bytes, occupy four registers)
- Return: The holding register value

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:46:07

## 2.48 Get discrete input register command

- Function:

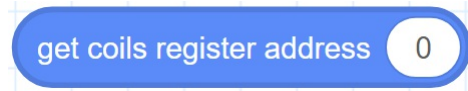


- Description: Read the discrete input register value from Modbus slave.
- Parameter: Address: starting address of the discrete inputs register. Value range: 0-4095.
- Return: The discrete input register value

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:46:15

## 2.49 Get coil register command

- Function:

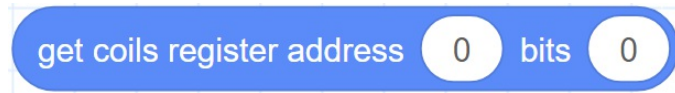


- Description: Read the coil register value from the Modbus slave.
- Parameter: Address: starting address of the coils register. Value range: 0 - 4095.
- Return: The coil register value

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:46:22

## 2.50 Get multiple coil register command

- Function:

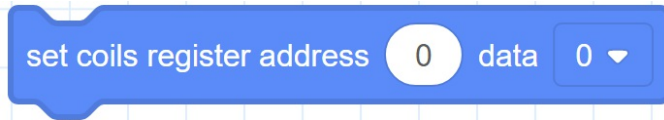


- Description: Read multiple coil register values from the Modbus slave.
- Parameter:
  - Address: starting address of the coils register. Value range: 0 – 4095.
  - Bits: Number of the coils to read. Value range: 0 to 4096- address.
- Return: The coil register values

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:51:57

## 2.51 Set coil register command

- Function:



- Description: Set the coil register in the Modbus slave. This command is not supported when the coil register address is from 0 to 5.
- Parameter:
  - Address: Starting address of the coils register. Value range: 6 – 4095.
  - Data: the values written into the coil register.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:52:15

## 2.52 Set multiple coil register values command

- Function:

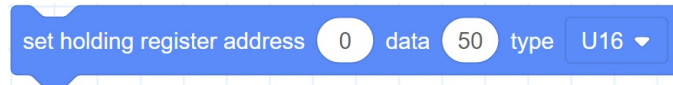
```
set coils register address 0 bits 0 data {0,0,0,0,0,0,0,0,0,0}
```

- Description: Set the multiple coil register values in the Modbus slave. This command is not supported when the coil register address is from 0 to 5.
- Parameter:
  - Address: Starting address of the coils register. Value range: 6 – 4095.
  - Bits: Number of the coils to read. Value range: 0 to 4096- address.
  - Data: the values written into the coil register.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:56:33

## 2.53 Set holding register command

- Function:



The image shows a blue rectangular interface for setting a Modbus command. It contains three input fields: 'set holding register address' with the value '0', 'data' with the value '50', and 'type' with a dropdown menu showing 'U16'.

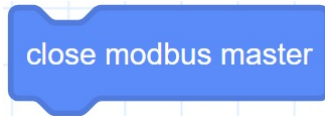
- Description: Set the holding register value in the Modbus slave.
- Parameter:
  - Address: Starting address of the holding registers to set. Value range: 0 – 4095.
  - Data: the values written into the holding register.
  - type: Data type
    - “U16”: Set 16-bit unsigned integer ( two bytes, occupy one register)
    - “U32”: Set 32-bit unsigned integer (four bytes, occupy two registers)
    - “F32”: Set 32-bit single-precision floating-point number (four bytes, occupy two registers)
    - “F64”: Set 64-bit double-precision floating-point number (eight bytes, occupy four registers).
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:56:40



## 2.54 Close Modbus master command

- Function:



- Description: Release a Modbus connection.
- Parameter: None
- Return: None

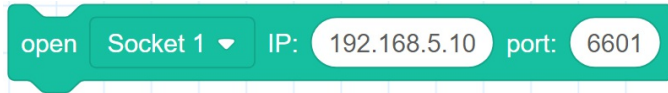
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:52:36

# TCP Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.55 Open socket command

- Function:



- Description: Create a TCP network, robot as a client.
- Parameter:
  - Socket: indicate Socket index, value range: Socket 1 to Socket 4.
  - IP: IP address of the server.
  - port: port of the server.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:56:51

## 2.56 Get open socket command

- Function:

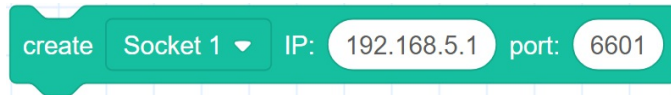


- Description: Get the connection result.
- Parameter: Socket: indicate Socket index, value range: Socket 1 to Socket 4.
- Return:
  - 0: TCP connection is successful
  - 1: Input parameters are incorrect

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:57:33

## 2.57 Create socket command

- Function:



- Description: Create a TCP network, robot as a server.
- Parameter:
  - Socket: indicate Socket index, value range: Socket 1 to Socket 4.
  - IP: address of the server
  - port: Server port

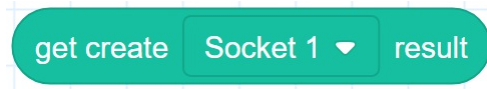
The port cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.

- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:57:03

## 2.58 Get create socket command

- Function:



- Description: Get the connection result.
- Parameter: Socket: indicate Socket index, value range: Socket 1 to Socket 4.
- Return:
  - 0: TCP network is created successfully
  - 1: TCP network is created failed
  - Socket: Socket object

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:57:47

## 2.59 Close socket command

- Function:

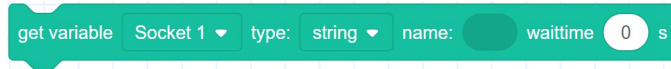


- Description: Release a TCP network.
- Parameter: Socket: indicate Socket index, value range: Socket 1 to Socket 4.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:57:54

## 2.60 Get variable command

- Function:



- Description: Obtain data through Socket communication.
- Parameter:
  - Socket: indicate Socket index, value range: Socket 1 to Socket 4.
  - Type: string or number
  - Name: Variable used to hold data
  - Waiting time: Set the waiting time, if the waiting time value is 0, it will wait until get data.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:57:21



## 2.61 Socket send variable command

- Function:

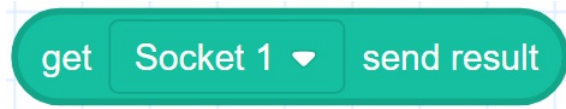


- Description: Send data through socket communication.
- Parameter:
  - Socket: indicate Socket index, value range: Socket 1 to Socket 4.
  - Variable: data to be sent.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:57:27

## 2.62 Get socket send result command

- Function:



- Description: Get the result of the data communication through the Socket.
- Parameter: Socket: indicate Socket index, value range: Socket 1 to Socket 4.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

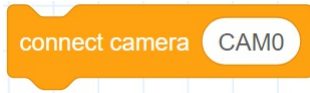
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 20:58:16

# Vision Command

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.63 Connect camera command

- Function:



- Description: Establish a communication connection with the camera.
- Parameter: Name of the camera.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:41:01

## 2.64 Get connect result command

- Function:

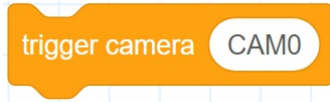
get connect camera result

- Description: Get the connection result.
- Parameter: None
- Return:
  - 0: Connect successfully
  - 1: Fail to Connect

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:40:35

## 2.65 Trigger the camera command

- Function:



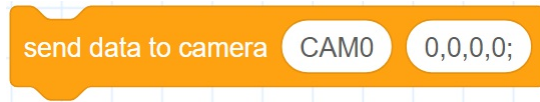
- Description: Trigger the camera to take a picture.
- Parameter: Name of the camera
- Return: None

s

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:43:25

## 2.66 Send data to the camera command

- Function:



- Description: Send data to the camera.
- Parameter:
  - Parameter 1: Camera name
  - Parameter 2: Data to be sent
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:44:44

## 2.67 Receive data from camera command

- Function:



- Description: Receive data from the camera.
- Parameter:
  - Parameter 1: Camera name
  - Parameter 2: Data type, the value can be number or string
- Return: The data sent by the camera

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:44:30



## 2.68 Get the number of data groups command

- Function:

camera data group

- Description: Gets the number of data groups sent by the camera.
- Parameter: None
- Return: The number of data groups

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:45:13

## 2.69 Gets a data command

- Function:

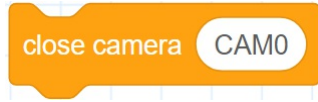
camera data at 1 group, at 1 item

- Description: Gets a data sent by the camera.
- Parameter:
  - Group: Specifies the group. Value range: 1 to the number of camera data groups.
  - Item: Select the item.
- Return: a data sent by the camera

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:45:46

## 2.70 Close camera command

- Function:



- Description: Release the connection.
- Parameter: None
- Return: None

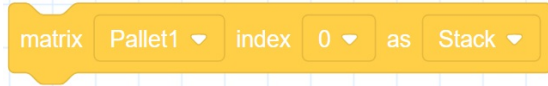
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:46:13

# Pallet Command

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:16:07

## 2.71 Instantiate matrix pallet

- Function:

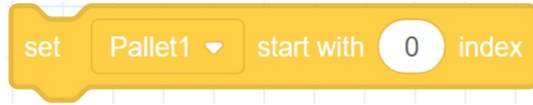


- Description: Instantiate matrix pallet.
- Parameter:
  - Pallet: Select a pallet from Pallet1 ~ Pallet4.
  - Index: Matrix pallet index.
  - IsUnstack: Stack mode. Value range: Stack, Assembly.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:57:50

## 2.72 Set the next stack index which is to be operated

- Function:

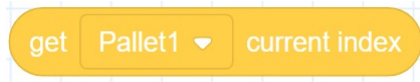


- Description: Set the next stack index which is to be operated.
- Parameter:
  - Pallet: Select a pallet from Pallet1 ~ Pallet4.
  - Index: The next stack index. Initial value: 0.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 05:59:23

## 2.73 Get the current operated stack index

- Function:

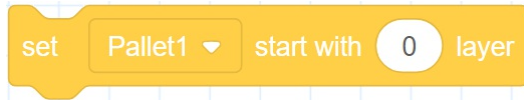


- Description: Get the current operated stack index.
- Parameter: Select a pallet from Pallet1 ~ Pallet4.
- Return: The current operated stack index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 14:59:31

## 2.74 Set the next pallet layer index which is to be operated

- Function:



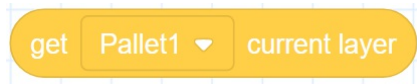
- Description: Set the next pallet layer index which is to be operated.
- Parameter:
  - Pallet: Select a pallet from Pallet1 ~ Pallet4.
  - Index: The next pallet layer index. Initial value: 0.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 06:00:58



## 2.75 Get the current pallet layer index

- Function:



- Description: Get the current pallet layer index.
- Parameter: Select a pallet from Pallet1 ~ Pallet4.
- Return: The current pallet layer index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 06:02:59

## 2.76 Reset pallet

- Function:

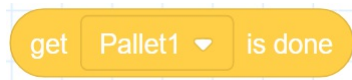


- Description: Reset pallet.
- Parameter: Select a pallet from Pallet1 ~ Pallet4.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 15:36:45

## 2.77 Check whether the stack assembly or dismantling is complete

- Function:

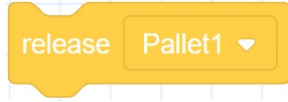


- Description: Check whether the stack assembly or dismantling is complete.
- Parameter: Select a pallet from Pallet1 ~ Pallet4.
- Return:
  - true: Finished.
  - false: Un-finished.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 06:08:19

## 2.78 Release palletizing instance

- Function:



- Description: Release palletizing instance.
- Parameter: Select a pallet from Pallet1 ~ Pallet4.
- Return: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 06:10:48

## 3. Description of Programming

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-28 11:31:18

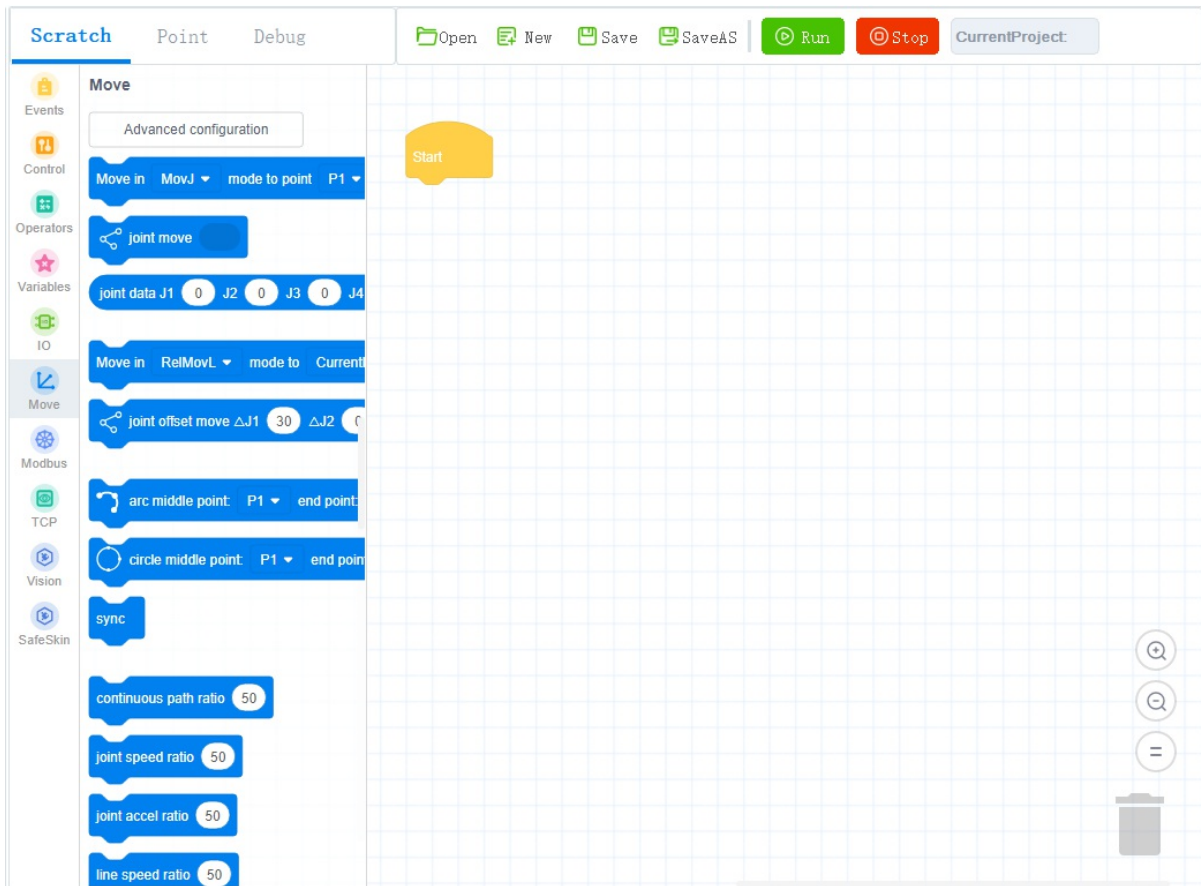
## 3.1 Basic operation

### Prerequisites

The robot has been powered on.

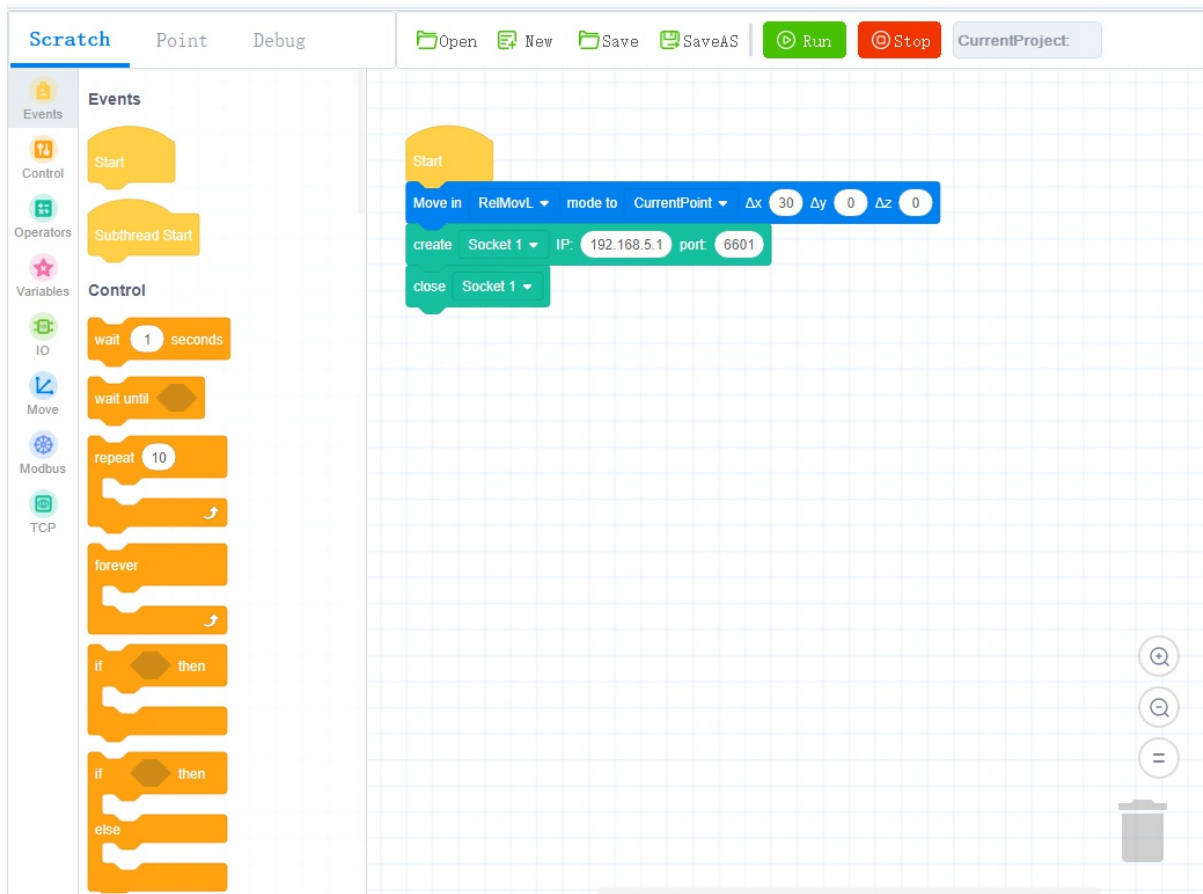
### Procedure


**Step 1** Enter the Blockly page. The system creates a new project by default.




**Step 2** Drag the blocks to the code area to start programming, as shown in following figure.

- Set the corresponding parameters of each block according to actual needs, for details see 2 Introduction.
- In the **point** page, you can save teaching point, when setting the parameters of the block, you can call the save point directly, for details see 3.2 Teaching points.



**Step 3** Click  Save to save the current project.

If it is the first time to save, you need to enter the project name.

**Step 4** Click  to enable the robotic arm.

**Step 5** Click  to run projects in the current code area.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-18 14:39:32

## 3.2 Teaching points


### Prerequisites

The project has been created or imported.


### Procedure

After creating a project, please teach positions on the **point** page for calling commands when programming a robot. If the existing taught positions list has been imported, this operation can be skipped.








**Step 1** Click  to enable the robotic arm.

**Step 2** Click **Jog** buttons to move the robot to a point.


**Step 3** Click **Point** to enter point page and click  **Add** to add a teaching point.

The teaching point information is displayed on the **point** page, as shown in the following figure.

No.	Alias	X	Y	Z	Rx	Ry	Rz	R	D	N	Cfg	Tool	User
1	P1	-0.0000	-247.5...	1050.5...	-90.0000	0.0000	180.0000	1	1	-1	-1	No.0	No.0
2	P2	-37.6501	-252.9...	1045.3...	-80.2021	31.1770	-151.9...	1	1	-1	1	No.0	No.0

Button	Description
 Add	Add a point
 Delete	Delete a point
 Cover	Cover a point. Select a teaching point, after jogging the robot to a point, click the icon to cover the selected teaching point
 RunTo	Run to a point, select a point, click the button to run the robot to this point
 Undo	Cancel



 Redo

Recover

- You can select a taught position and double-click the parameters on the line to modify the relevant information.

- Also, you can select a taught position and click  Cover to cover the current taught position.

**Step 4** Add points by referring to Step 2 and Step 3.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-17 14:38:07

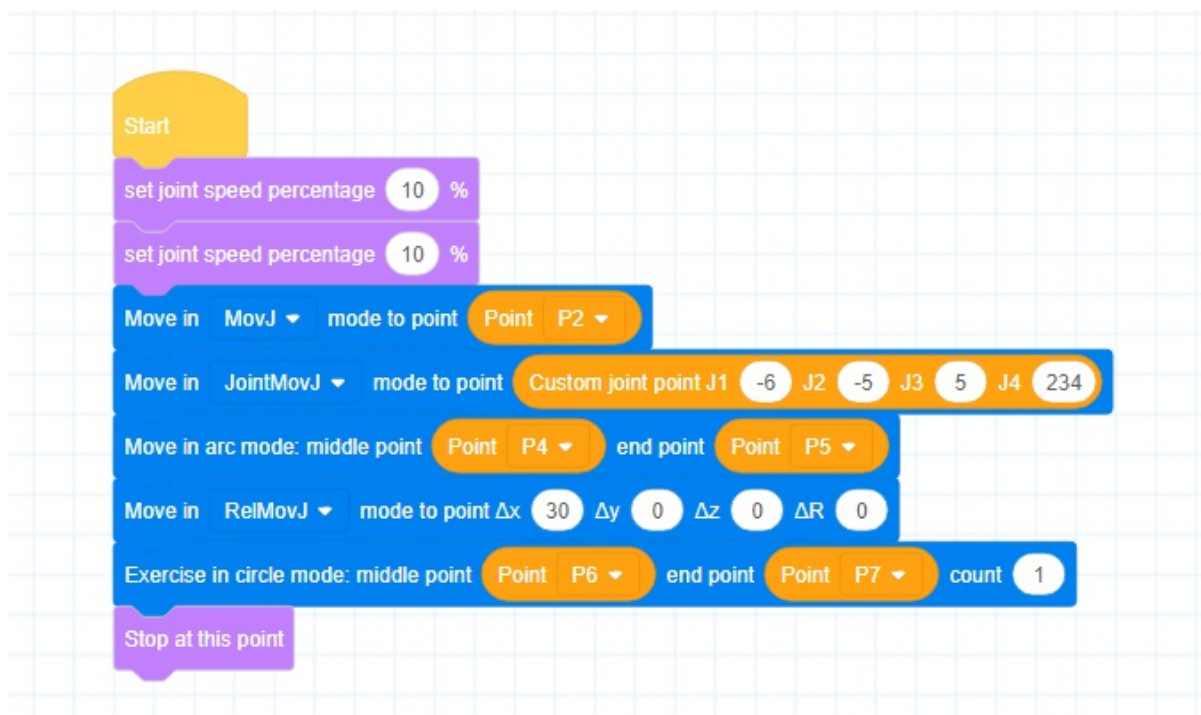
## 3.3 Quick Start

This section gives examples of Blockly for Motion commands, I/O commands, Modbus commands and TCP commands, for user reference only.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision: 2021-09-15 15:19:33

### 3.3.1 Robot movement

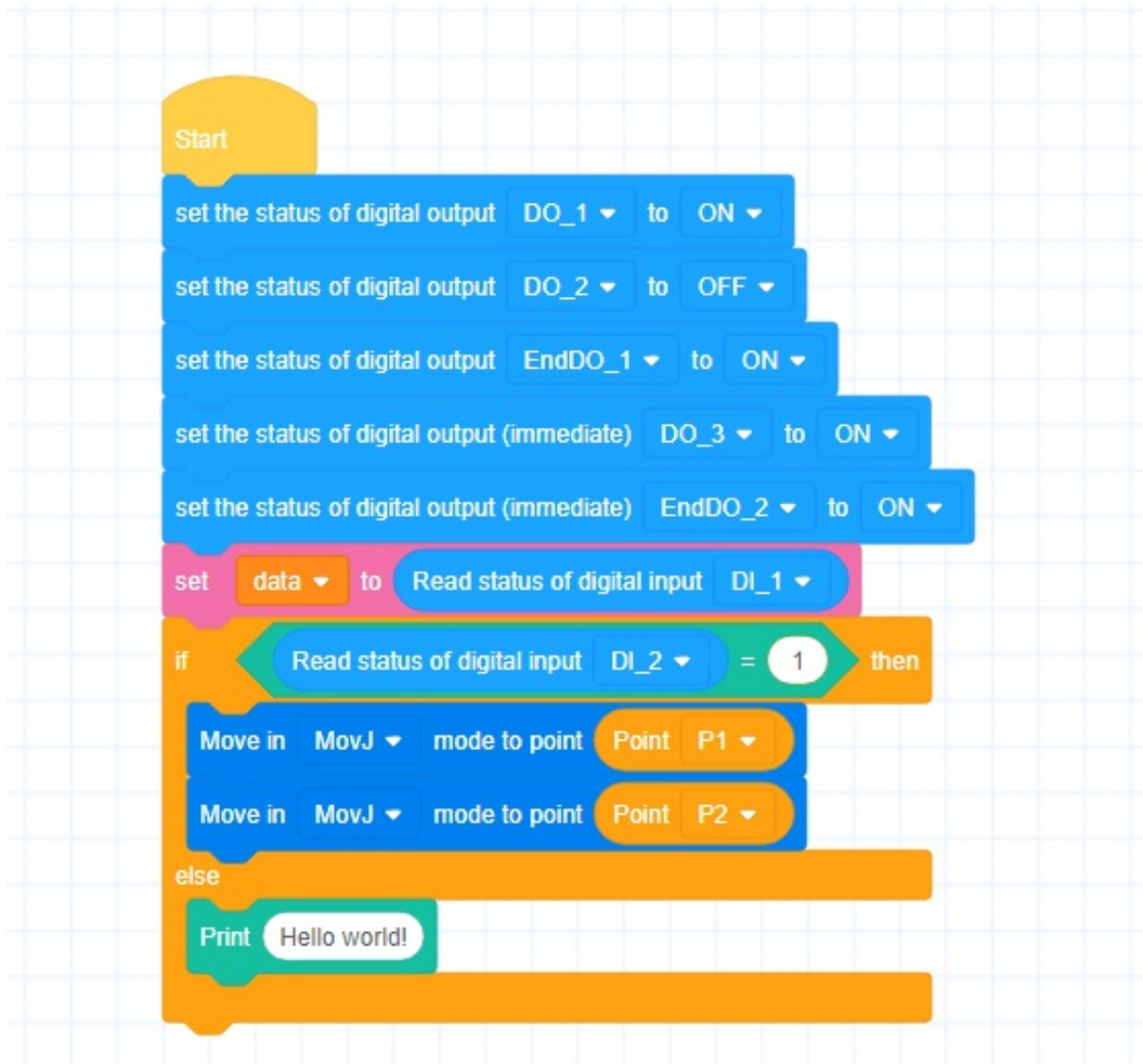
Running the Motion commands can control the movement of robot in the joint coordinate system and the Cartesian coordinate system. The detailed description of the Motion commands, please see **Motion Commands**. The following figure shows a programming program that includes Motion commands.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 15:16:37

### 3.3.2 I/O Setting

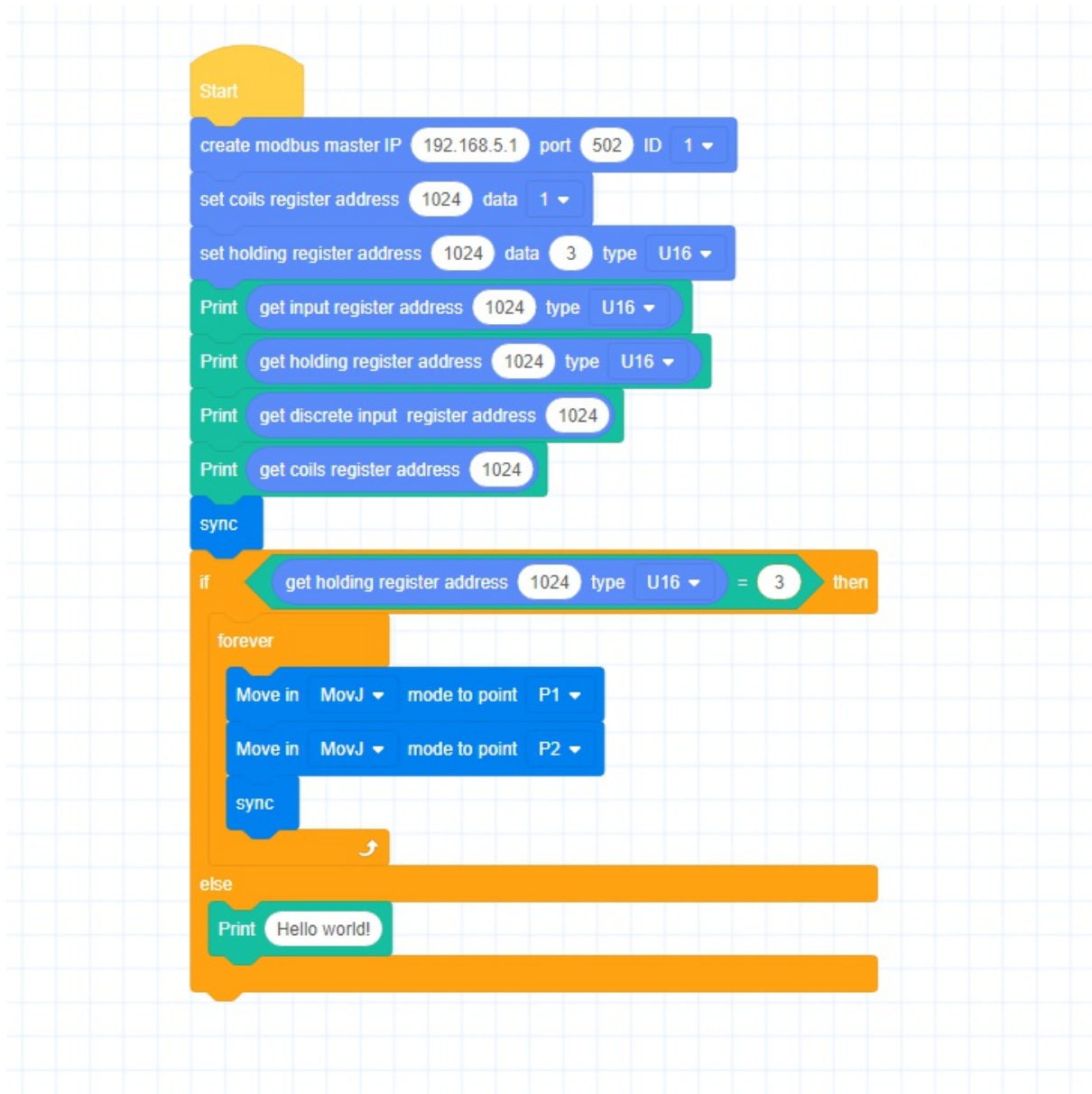
Running I/O commands to set or get each I/O statue. The detailed description of the I/O commands, please see **I/O Commands**. The following figure shows a programming program that includes I/O commands.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-17 15:03:40

### 3.3.3 Register setting and reading

By running the Modbus commands to set or read the value of each register address. The detailed description of the Modbus commands, please see **Modbus Commands**. The following figure shows a programming program that includes Modbus commands.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-15 15:29:10

### 3.3.4 Create TCP Client

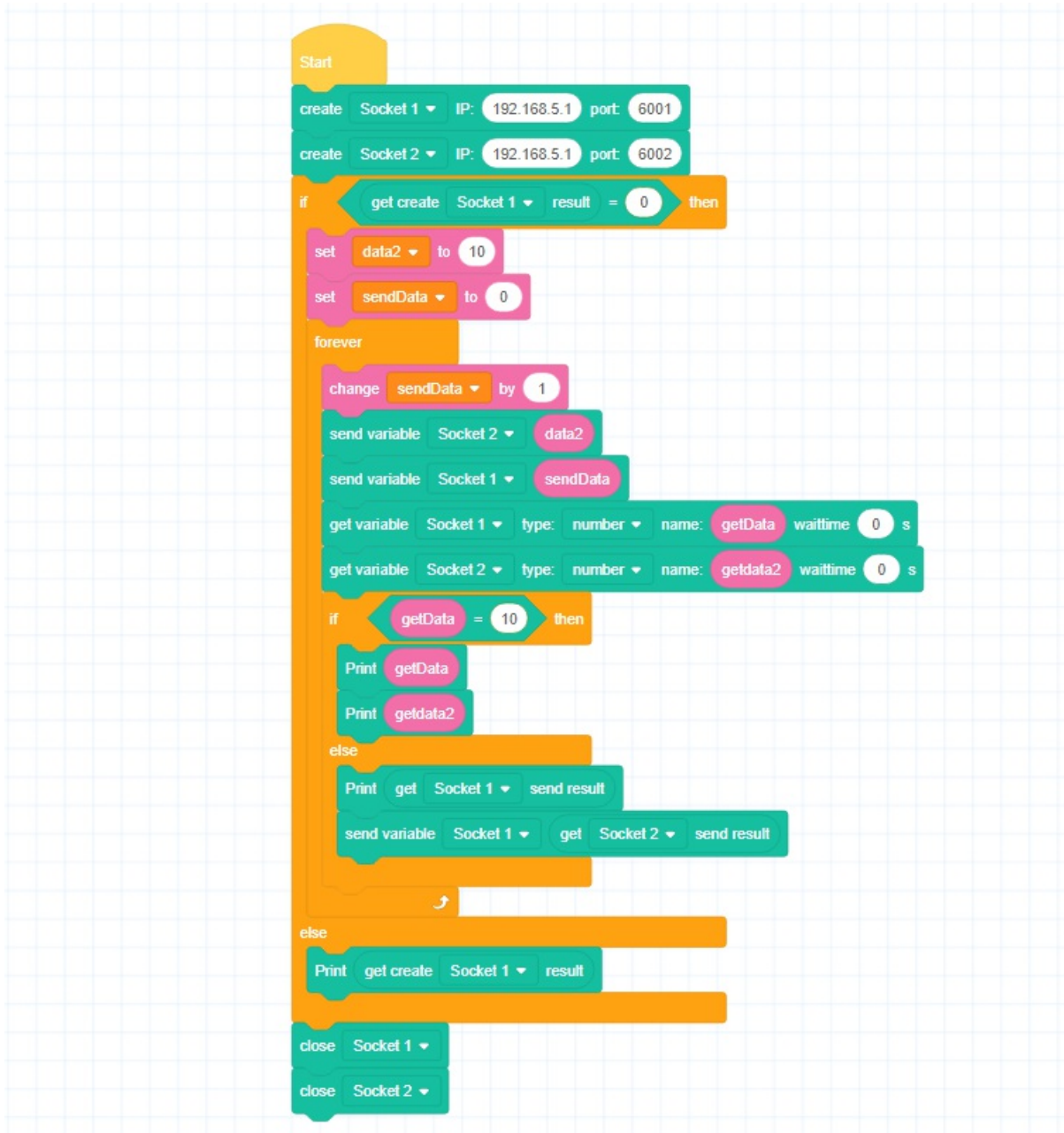
Run `open Socket 1 IP: 192.168.5.10 port: 6601` to establish communication with the TCP server, the robot as the TCP client. Running TCP commands can send and read communication data, the detailed description of TCP commands, please see **TCP Commands**. The following figure shows a programming program that includes TCP commands.

```
Start
open Socket 1 IP: 192.168.5.12 port: 6001
open Socket 2 IP: 192.168.5.12 port: 6002
if get open Socket 1 result = 0 then
  set data2 to 10
  set sendData to 0
  forever
    change sendData by 1
    send variable Socket 2 data2
    send variable Socket 1 sendData
    get variable Socket 1 type: string name: getData waittime 0 s
    get variable Socket 2 type: string name: getdata2 waittime 0 s
    if get Socket 1 send result = 0 then
      Print getData
      Print getdata2
    else
      Print get Socket 1 send result
      send variable Socket 1 get Socket 2 send result
  end
else
  Print get open Socket 1 result
close Socket 1
close Socket 2
```



### 3.3.5 Create TCP Server

Run `create Socket 1 IP: 192.168.5.1 port: 6601` to set the robot as the server, waiting for the TCP client to connect. Running TCP instruction can send and read communication data, the detailed description of TCP commands, please see **TCP Commands**. The following figure shows a programming program that includes TCP commands.

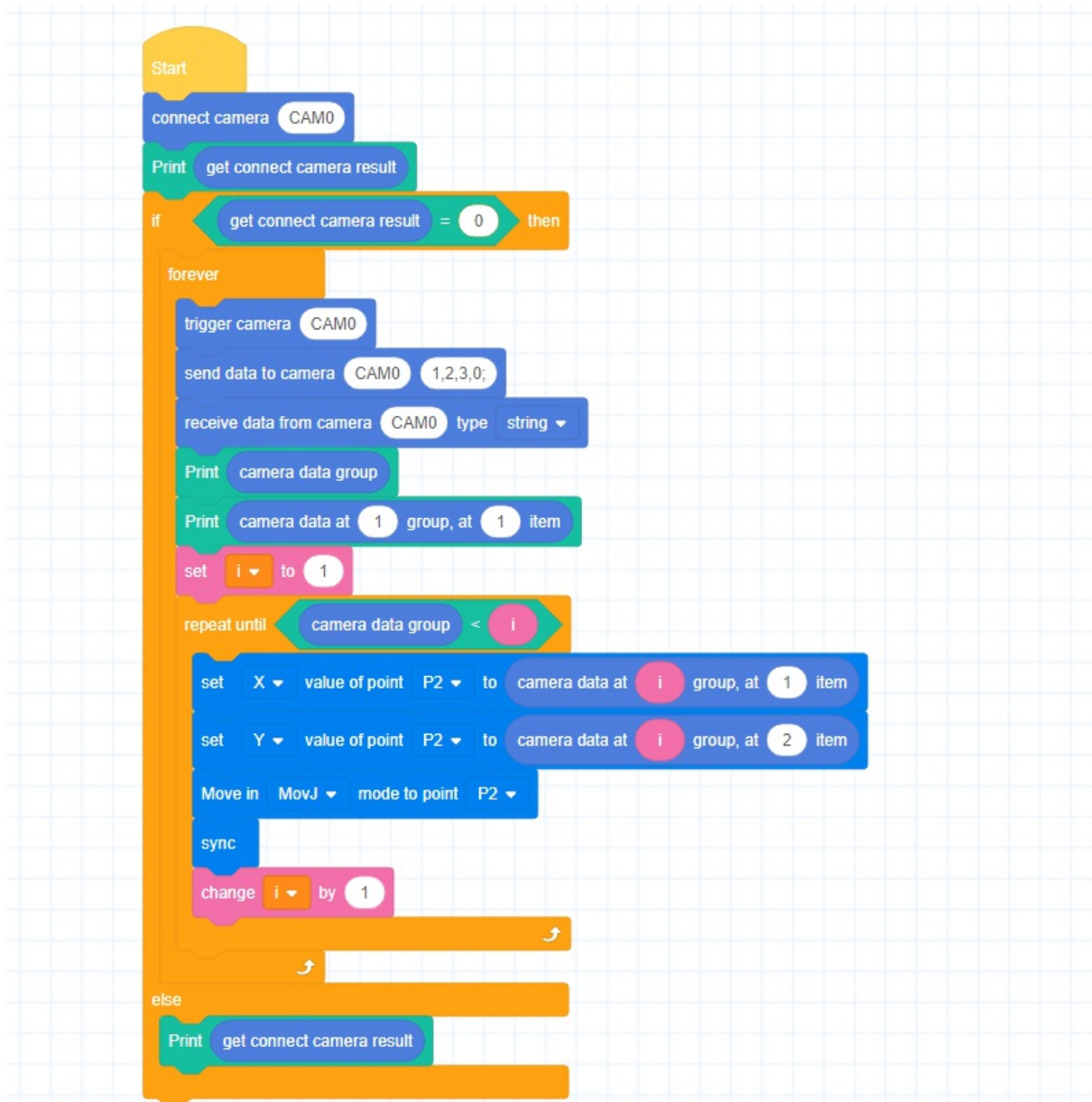


Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision: 2021-09-15 15:38:41



### 3.3.6 Vision Interaction

Running the vision instructions to establish communication with the camera, and to send and read the camera data. The detailed description of vision commands, please see **Vision Commands**. The following figure shows a programming program that includes vision commands.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2021 all right reserved, powered by GitbookRevision:  
2021-09-29 15:16:57