

Elephant Robotics User Manual

myCobot Pro 600



Language: English
Compiled in 2022/3/28

Copyright Declaration

No unit or individual may extract, compile, translate or reproduce any contents of this manual (eg: technical documentation, software, etc.), nor disseminate in any form (including materials and publications) without the written permission of Shenzhen Elephant Robotics Technology Co., Ltd. (hereinafter referred to as “Elephant Robotics”).

In addition, the product information and related resources mentioned in this manual are for reference only and the contents are subject to change without notice.

Except as expressly stated in this manual, nothing in this manual should be construed as any warranty or guarantee by the Elephant Robotics of personal loss, damage to property, or fitness for a particular purpose.

All rights reserved!

Overview to the manual

About the manual

Welcome to use myCobot robot and thank you for your purchase.

This manual describes the precautions for proper installation and use of the myCobot robot.

Please read this manual and other related manuals carefully before installing this robot system. After reading, please keep it in a safe place so that you can access it at any time.

Reading objects of the manual

This manual is targeted to:

- installer.
- Debugger.
- Maintenance staff.

How to use

This manual should be used when doing the following works:

- Installation work: Move the robot to the working position and fix it to the base according to the installation instructions.
- Debugging: Debugging the robot to work status.
- Maintenance work: regular maintenance robot system to ensure its normal functioning. When the robot malfunctions due to environmental influences or improper operation of the user, or a certain component of the robot system exceeds the normal service life, the robot needs to be repaired.

Main contents of the manual

- Precautions for safe use of the robot.
- Mechanical, electrical installation and commissioning of the robot.
- Maintenance and repair of the robot.

Remarks:

This manual is updated from time to time, and the update date is the version number. Users can download the latest version from the official website of Elephant Robot.

目录

1 Security	6
1.1 Introduction.....	6
1.2 Safety alert symbol description.....	6
1.3 Hazard identification.....	7
1.4 Safety Precautions.....	9
1.5 Label, nameplate introduction	11
1.6 Avoid misuse.....	11
1.7 Emergency stop.....	12
1.7.1 Emergency button	12
1.7.2 Collision checking	12
1.8 Urgent handling	13
2 QuickStart	14
2.1 Installation Instructions for the Robot Arm	14
2.2 Display Module Connection	20
2.2.1 Display Screen Connection.....	20
2.2.2 Remote Connection.....	21
2.3 Quickly Building a Runnable Project	22
2.3.1 Preparation Work	22
2.3.2 Flow Chart	23
2.3.3 Specific Steps.....	24
3 Product Introduction	36
3.1 Overview.....	36
3.2 Product Appearance and Composition.....	37
3.3 Working Principles and Specifications	38
3.3.1 Working Space	38
3.3.2 Coordinate System.....	39
3.3.3 Moving Functions	42
3.4 Technical Specifications	43
3.4.1 Technical Parameters	43
3.4.2Size Parameters	44
4 Interface Specifications.....	46
4.1 Base Electrical Interfaces.....	46
4.1.1 Introduction to Base Electrical Interfaces.....	46
4.1.2 Description of Base Electrical Interfaces.....	48
4.2Electrical Interfaces of Robot Arm End.....	51
4.2.1 Introduction to Robot Arm End	51
4.2.2 End Electrical Specification.....	51

5 Operation Guide.....	53
5.1 RoboFlow Software Instructions	53
5.1.1 Overview	53
5.1.2 Main interface introduction.....	54
5.1.3 Introduction to common tools	79
5.1.4 Function instruction	96
5.2 API Interface Description	126
5.2.1 Overview	126
5.2.2 Socket String format rules	126
5.2.3 Socket API Usage Example	133

1 Security

1.1 Introduction

1 Introduction to this chapter

This chapter details general safety information for people who perform installation, maintenance, and repair work on the robots. Please read and understand the contents and precautions of this chapter before handling, installation and use.

As described in GB 11291.1-2011, whether it is a robot manufacturer, system integrator, or individual user, it is necessary to carry out hazard identification and risk assessment before using the robot. It is required to conduct a hazard analysis to identify any hazards that may arise; and for hazards identified in hazard identification, a risk assessment should be performed to maximize personal safety and property safety.

This chapter provides a basic guide to safe use by introducing different safety alert symbols and precautions.

2 Interpretation of related terms

1) Collaborative operation

A specially designed robot that works directly with people in a defined workspace.


2) Collaborative workspace





In the safety protection space of the robot work unit, the robot and the person can complete the task at the same time in the production activity.

1.2 Safety alert symbol description

As shown in Table 1-1, this section describes the safety alert symbols used in this manual. You can find the corresponding symbols described in this chapter in other chapters, please note the meaning of these symbols and their meanings.

Table1- 1 Safety Warning Symbol Table

 Danger	Danger: A dangerous situation that is likely to result in death or serious injury if not avoided.
---	---

 Warning	<p>WARNING: Conditions that may cause a hazard that, if not avoided, could result in personal injury or serious damage to the equipment.</p>
 Caution Electricity	<p>Be careful of electric shock: It may cause dangerous use of electricity. If it is not avoided, it may cause personal injury or serious damage to equipment.</p>
 Prohibited	<p>Prohibited: Things that are not allowed to do.</p>
 Attention	<p>Caution: Important things to be noted.</p>

1.3 Hazard identification

The safety of the collaborative robot is based on the premise of proper configuration and use of the robot, and even if all safety instructions are observed, the injury or damage caused by the operator may still occur. Therefore, it is very important to understand the safety hazards of robot use, which is beneficial to prevent problems before they occur.

Tables 1-2~4 below are common safety hazards that may exist in the context of using robots:

Table1- 2 Dangerous safety hazards


 Danger	
1	Personal injury or robot damage caused by incorrect operation during robot handling.
2	Personal injury or robot damage caused because the robot is not fixed as required, for example, the screw is not screwed or tightened, and the base is not enough to stably support the robot for high-speed movement, causing the robot to tip down.
3	Failure to perform proper safety function configuration of the robot, or installation of safety protection tools, etc., may cause the safety function of the robot to fail.

Table1- 3 Warning level security risks



 Warning	
1	Play around the robot, you may be hit by a running robot, or be tripped by an obstacle such as a cable to cause personal injury.
2	Unauthorized personnel change the security configuration parameters, causing the safety function to fail or danger.
3	Scratches and punctures caused by sharp surfaces such as other devices in the work environment or robot end effector.
4	The robot is a precision machine and pedaling may cause damage to the robot.
5	If the clamp is not in place or before the power supply of the robot is turned off or the gas source is turned off (it is not determined whether the end effector firmly holds the object without falling off due to loss of power). If the clamped object is not removed, it may cause danger, such as people being injured by crashing.
6	There is a risk of accidental movement of the robot. Under no circumstances should you stand under any axis of the robot!
7	The robot is a precision machine. If it is not placed smoothly during handling, it may cause vibration and may cause damage to the internal components of the robot.

Table1- 4 Potential safety hazards that may result in electric shock

 Caution Electricity	
1	Using a non-original cable may pose an unknown hazard.
2	Contact with liquids by electrical equipment may result in a risk of electric leakage.
3	There may be an electric shock hazard when the electrical connection is incorrect.
4	Be sure to handle replacement work after turning off the power to the controller and related equipment and unplugging the power cord. If the work is performed while the power is on, it may cause electric shock or malfunction.

1.4 Safety Precautions

In general, compared with ordinary machinery, robots have the characteristics of larger working range and faster speed, so they are accompanied by the dangers that ordinary machinery does not have. When installing, using, and maintaining the robot, please pay attention to the following items shown in Table 1-5 and Table 1-6 (the followings are some of the common precautions listed):

Table1- 5 Safety precautions on acts that need to be banned



 Prohibited	
1	It is forbidden to modify the robot or use non-original accessories.
2	Untrained non-professionals are prohibited from entering the robot work area at will, pressing any button or doing other operations at will.
3	The relevant personnel shall not maintain, repair or use the robot after being affected by drinking, taking drugs or stimulating drugs.

Table1- 6 General safety precautions

 Attention	
1	Anyone responsible for installing and maintaining the robot must read and follow these safety instructions.
2	Ensure that safety measures and robot safety configuration parameters are defined as required in the risk assessment to protect programmers, operators, and bystanders.
3	Production operators should not loosen long hair (long hair must be picked up) and wear a work cap, not wearing jewelry.
4	Operators operating in conjunction with the robot must be familiar with and understand the content and exact location of the various warning signs and warning symbols on the equipment and ensure that all warning signs and warning symbols are complete and clear, and that all safety devices are secured before opening and starting the equipment. And make sure the relevant accessories are normal and no one is in a dangerous location where the equipment is activated. When the robot runs abnormally, it should be stopped immediately and the situation must be reported it in time.
5	The operator must clarify the scope of operation, commissioning, maintenance and

	repair. The operator is not allowed to change the operating procedures and trials at will, and other personnel are not allowed to enter the collaborative operation space and danger zone.
6	When repairing work, operators must hang the warning sign to enter the collaborative operation space.
7	When the operator enters the safety zone of the equipment protected by the safety guard door, it shall be absolutely guaranteed that the safety guard door will always open when working in the area, and the door must be in an unlocked position.
8	When the operator is in production, it should be ensured that each starting device is normal and cannot be started at will.
9	When the maintenance and operation personnel perform maintenance on the equipment, the main power switch must be turned off to perform maintenance work.
10	No items should be stacked in the robot working area, and no debris should be placed in the control box.
11	After the operation is completed, the safety protection door should be closed immediately, and the various switches of gas and electricity should be closed according to the procedure, and the work site should be cleaned up.
12	Do not shake the robot and hang heavy objects on the robot.
13	No dangerous behaviors or games around the robot.
14	After installing the robot, make sure the robot is fixed on a stable surface for subsequent operations.
15	Make sure that the robot does not collide with itself or other objects during running.
16	If the robot is damaged, do not continue to use it.
17	Please use the robot within the robot's parameter range and service life, otherwise it will cause serious safety problems.
18	After the emergency stop state is canceled, and before the servo power is turned on, it is necessary to remove the obstacles and faults that cause the emergency stop, and then turn on the servo power.
19	Please pay attention to the rotating shaft of the robot to prevent the cable and the air tube from being entangled. Keep a distance from the shaft to prevent hair or clothing from getting entangled.

1.5 Label, nameplate introduction

Robots are high-precision equipment, and they are more dangerous than ordinary machines when they are unfamiliar or not in accordance with the manual. As shown in Figure 1-1, the labels are attached to the power box to remind the operator to read the relevant operating manual before use.

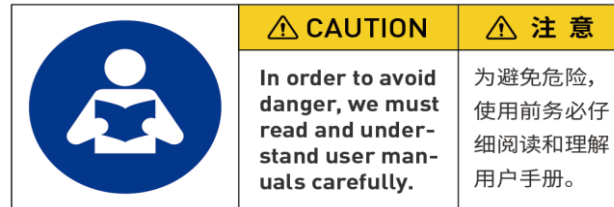


Figure 1- 1 Reading manual before the operation

The power box provides power to the entire robot system and must be operated correctly to prevent electric shock. As shown in Figure 1-2, A power-proof warning label is attached to the power box to remind the operator that there is a potential danger of electric shock to the power box, and it is required to be used correctly to prevent electric shock.

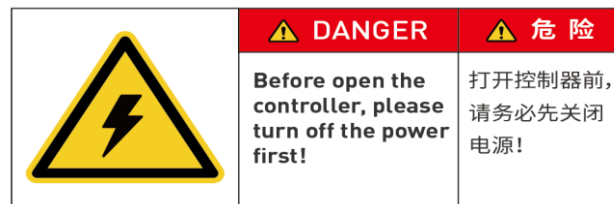


Figure 1- 2 Label to caution preventing electric shock

1.6 Avoid misuse

Please do not use the Catbot collaborative robot for the following purposes.

- Medical and life-critical applications.
- In environment that may cause an explosion.
- Used directly without risk assessment.
- Insufficient use of safety function levels.
- Inconsistent use of robot performance parameters.

1.7 Emergency stop

This section describes two types of emergency stop for robots:

- If you feel abnormal during the robot's motion, immediately press the emergency stop switch.
- When the force generated by the collision of the robot with the person or object is greater than the threshold, the robot detects the force generated by the collision, thereby stopping or moving to a certain position (collision return).

1.7.1 Emergency button

When the emergency stop button on the teach pendant is pressed, the drive will be stopped, the brake will start, the motor power will be turned off, and the electromagnetic brake will stop the robot's inertial motion, the robot will stop all motion, the program running in OS will also be stopped.

However, during normal operation, do not press the emergency stop switch at will. If the emergency stop switch is pressed during the operation, the robot movement trajectory before stopping will be different from the trajectory during normal operation and may hit a peripheral device or the like.

When it is in an emergency stop state (normal), if the robot system is to be placed in an emergency stop state, press the emergency stop switch when the robot does not operate.

Before using the emergency stop switch, you need to know the followings:

- The emergency stop (E-STOP) switch can only be used to stop the robot in an emergency.
- To stop the robot running the program in a non-emergency situation, use the Pause or STOP command. The Pause and STOP commands will not turn off the motor. Therefore, the brake will not work.
- If user need to control the emergency stop of the robot and other equipment at the same time, you can use the external E-STOP double loop circuit (user need to short it when not in use).

1.7.2 Collision checking

During the operation of the robot, it is possible to touch people or objects. It can be protected by setting a protection threshold. The specific operation mode is as follows: When the force generated by the collision of the robot with the person or the object is greater than the threshold, the robot detects the force generated by the collision, thereby stopping or moving to a certain position (collision return).

Please note that when the protection threshold is set too high, a large force

is required to stop the robot, which will reduce the sensitivity of the collision detection to a certain extent. When the protection threshold is set too low, the robot may stop when it is holding the load due to the excessive torque generated by its own motion. Please set the threshold of protection under guidance.

In addition, you can set the protection threshold for each movement and each movement of the robot, and set the two protection threshold directions including the X-Y plane (horizontal direction) and the Z plane (vertical direction).

1.8 Urgent handling



Attention

If the software pops up with a fatal error message, please activate the emergency stop quickly, write down the condition that caused the error, and contact your supplier.

In the event of a fire, use a carbon dioxide (CO₂) fire extinguisher!

2 QuickStart

2.1 Installation Instructions for the Robot Arm

1 Full unpacked items of robot arm



Robot arm
host machine



48V power
adapter



Wrenches and
mounting screws



G-type clamp



Mounting
plate



Emergency stop switch



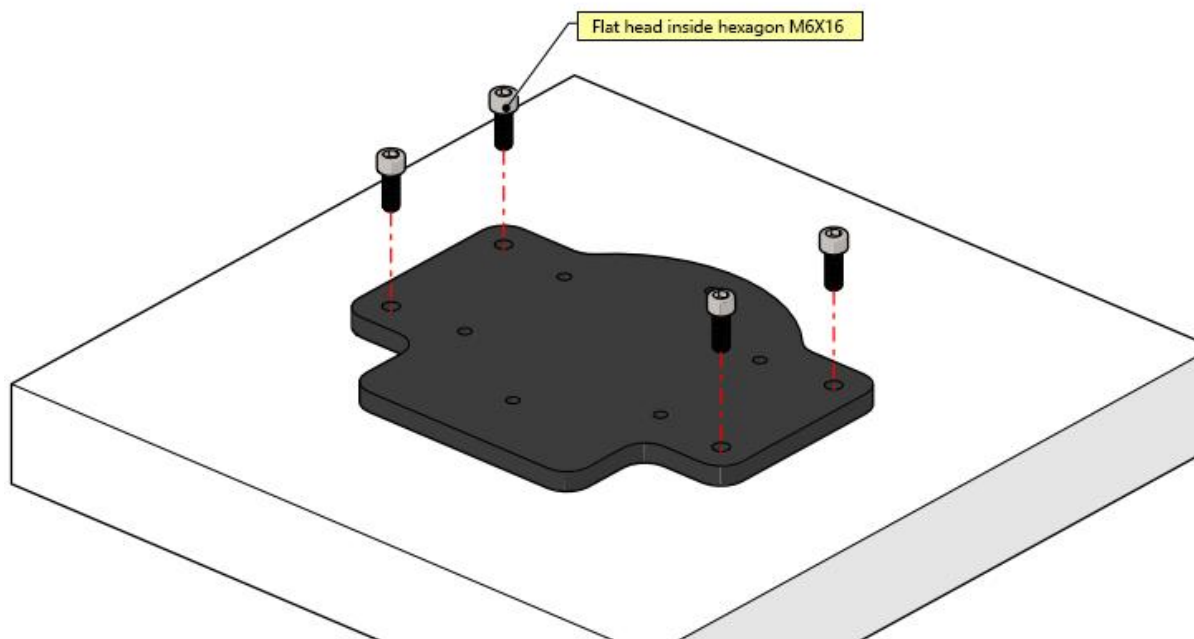
Product
brochure



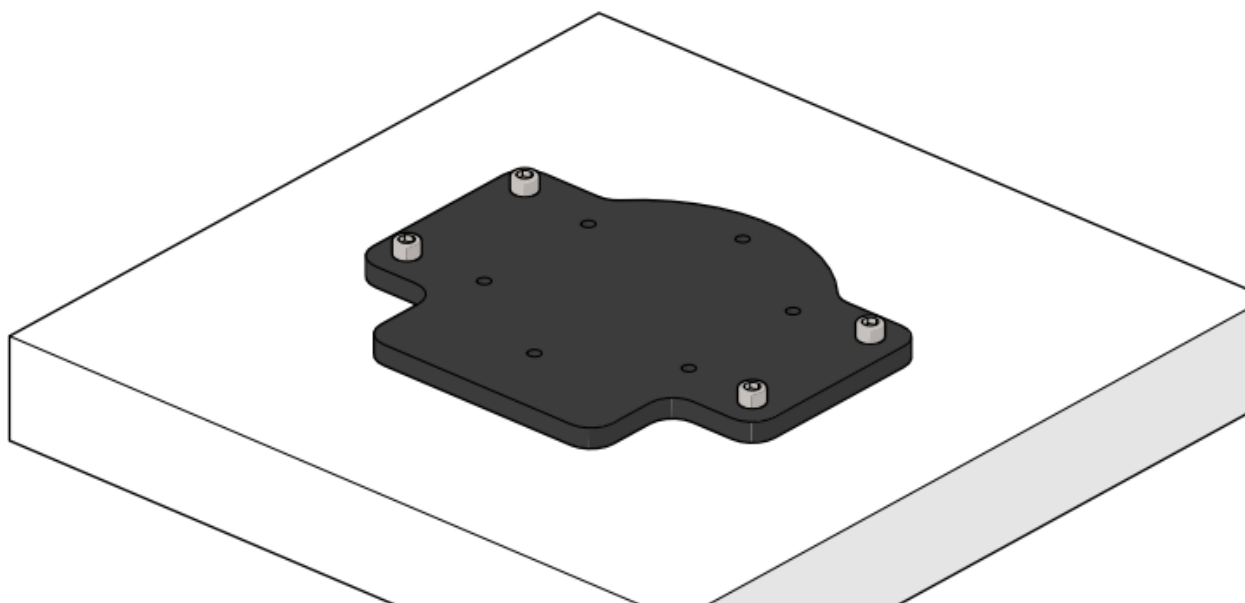
HDMI and M8
aviation plug wire

2 Pedestal mounting

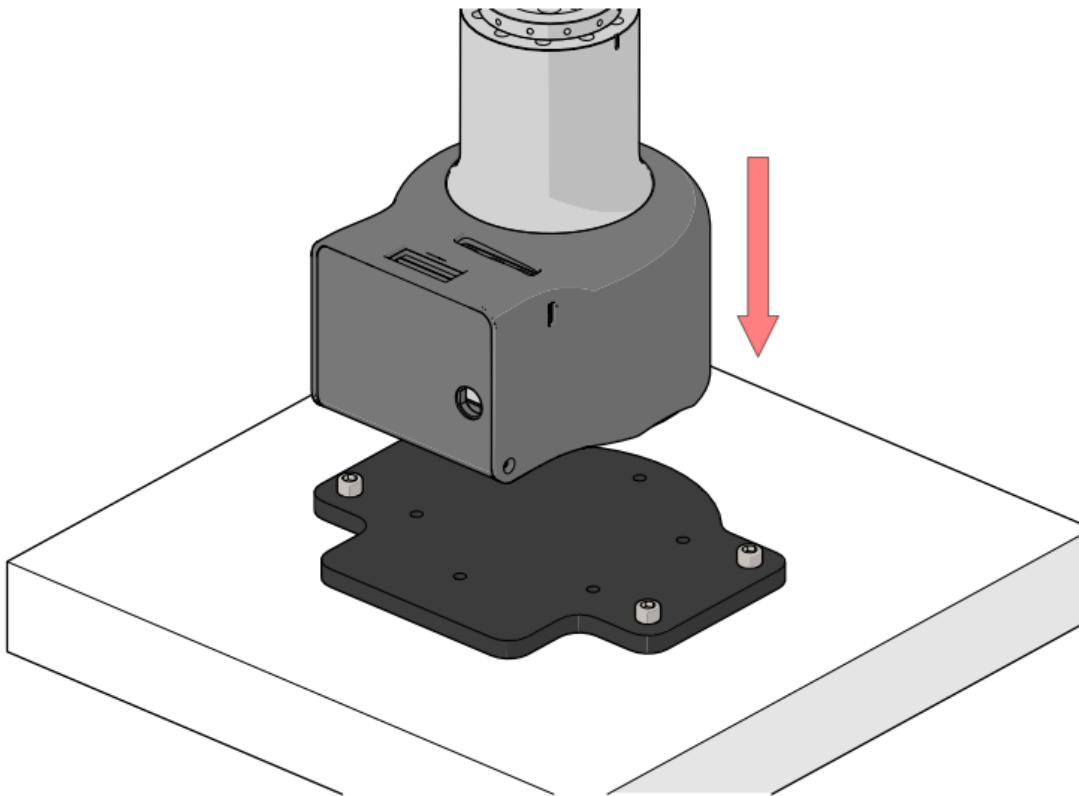
1. Install M6X16 screws to fix the baseplate on the table



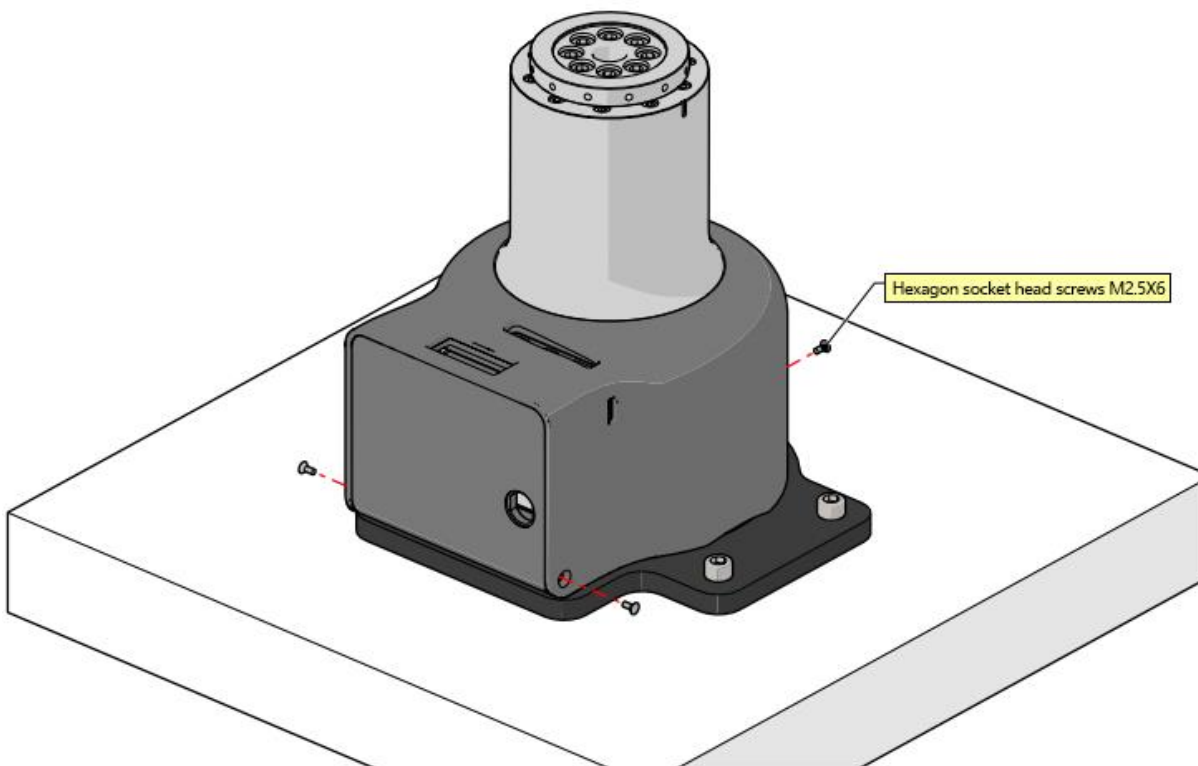
2. An effect picture of the completed installation



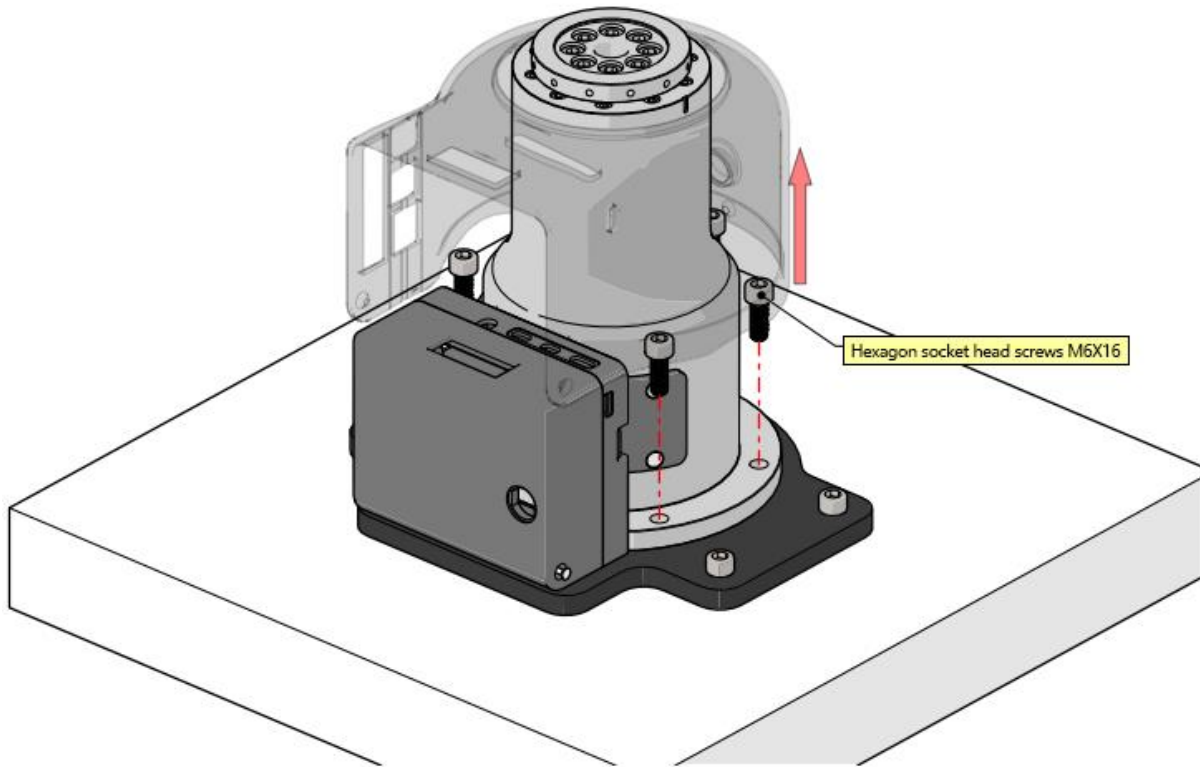
3. Place the pedestal of the robot arm on the fixed baseplate



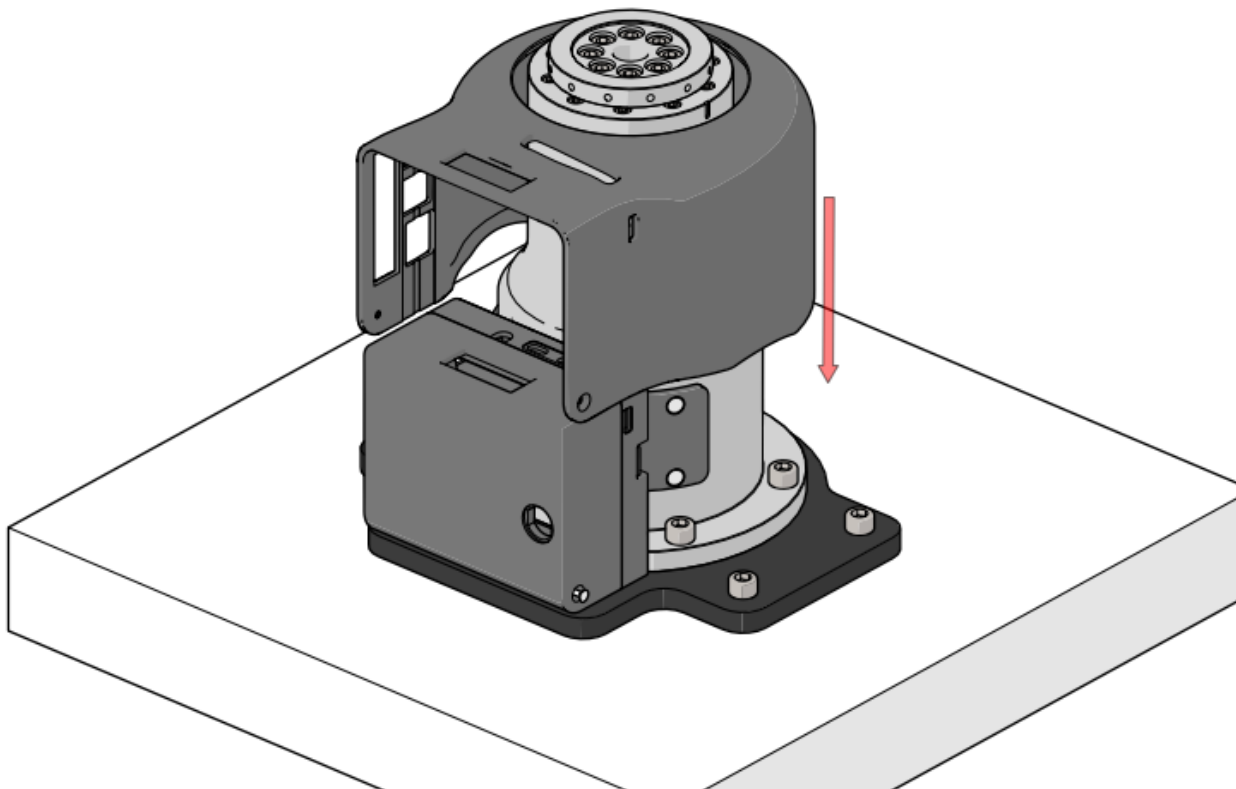
4. Remove the set screws of housing from the pedestal of robot arm



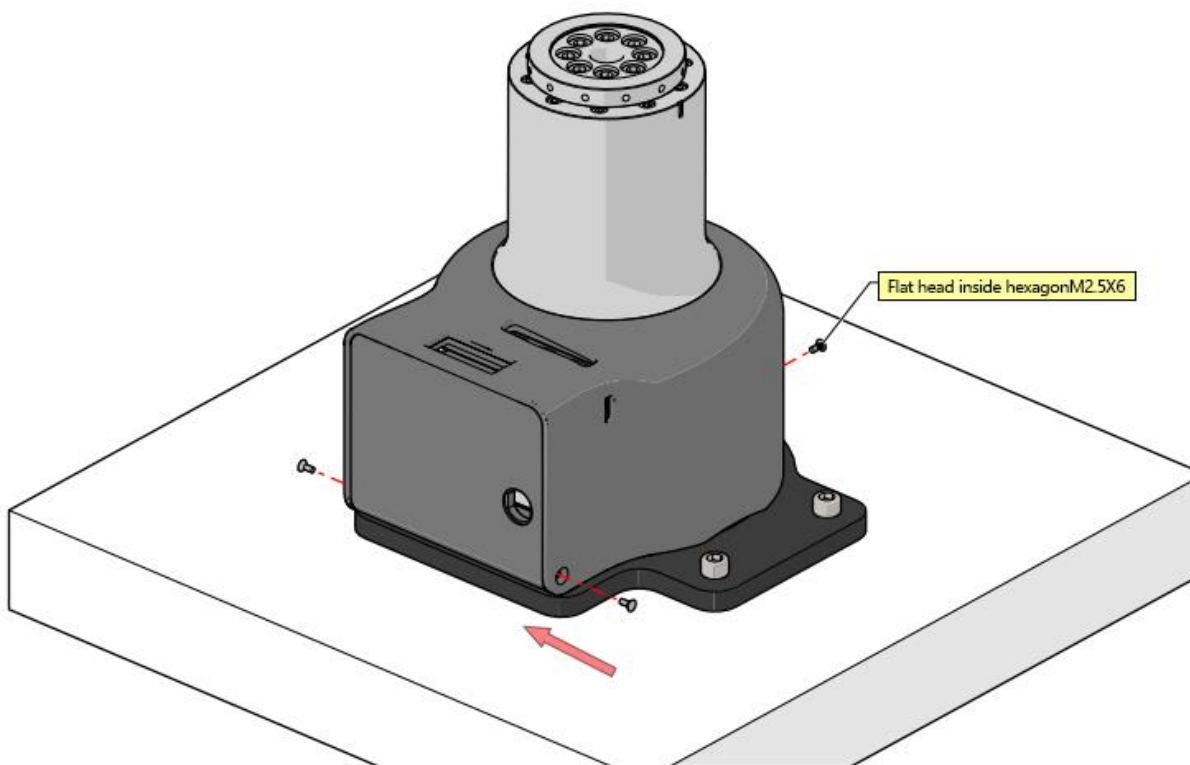
5. Open the housing of the pedestal of robot arm, and fix the the pedestal and baseplate of robot arm with M6X16 screws



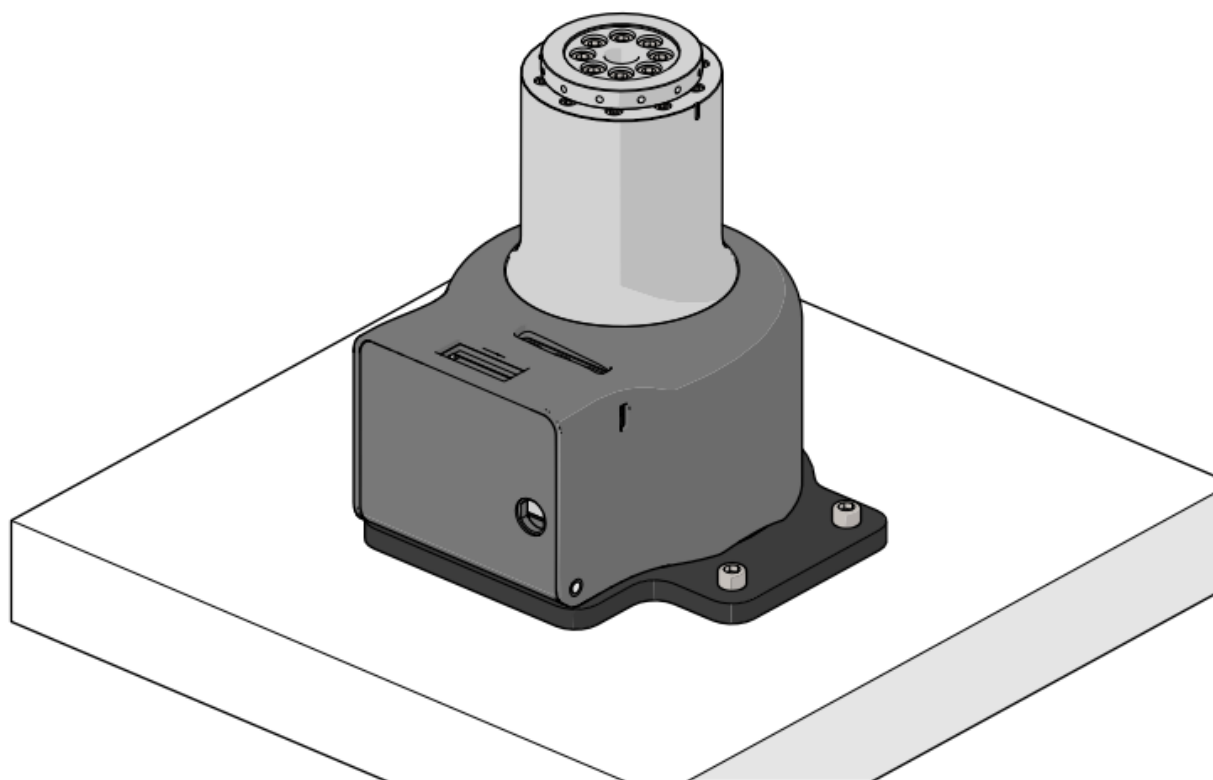
6. After installation, buckle the housing of the pedestal of robot arm



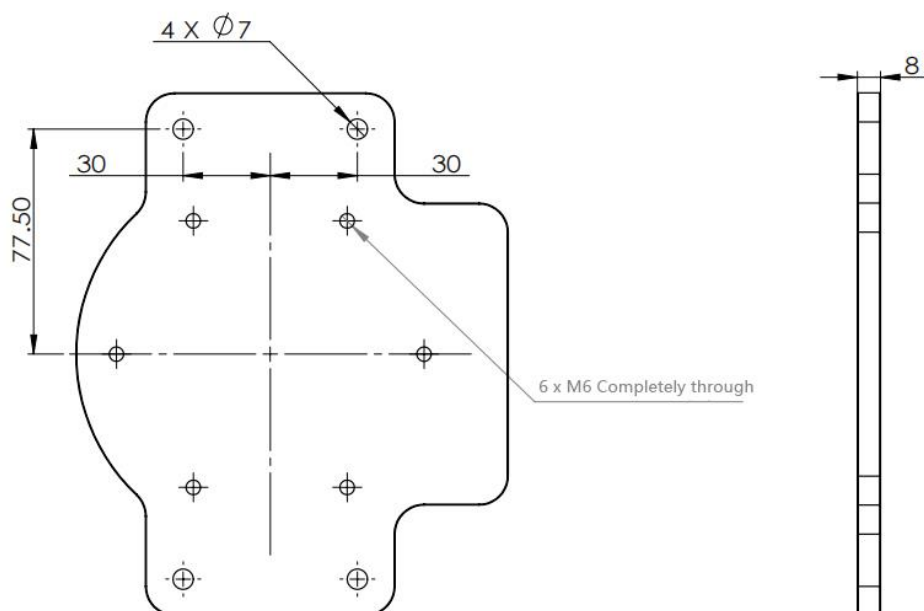
7. Install the set screws of housing for the pedestal of robot arm



8. Install the set screws of housing for the pedestal of robot arm



9. Baseplate dimension figure



2.2 Display Module Connection

2.2.1 Display Screen Connection

1 Prepare the displayer connecting line in accessories, as shown in Figure 2-1, and one port is an HDMI connector, and the other port is a micro HDMI connector.

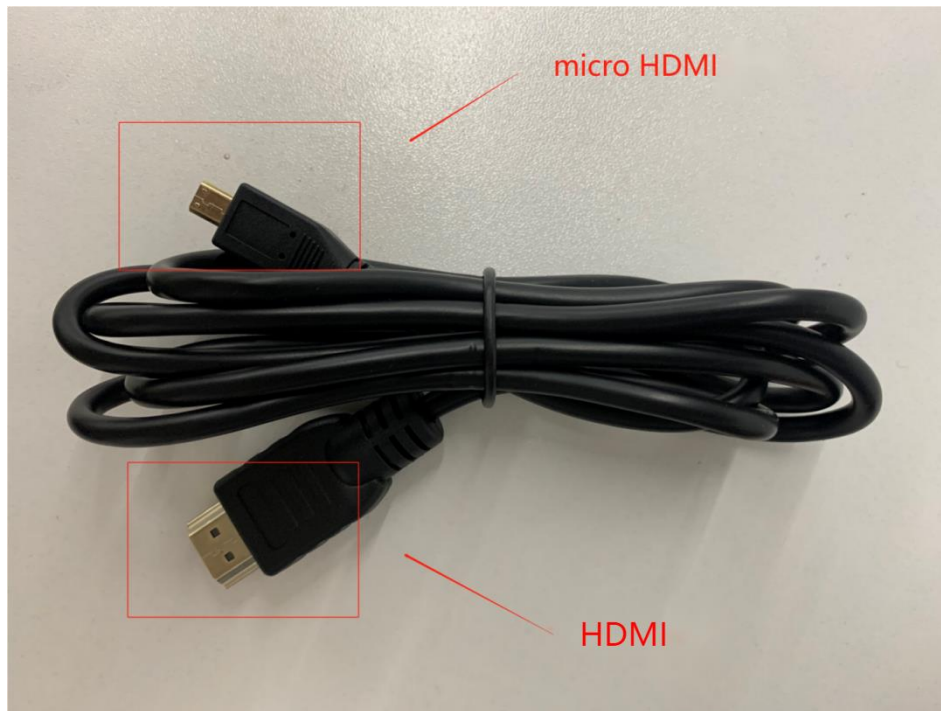


Fig. 2-1 Display Screen Connection

2 The HDMI connector should be connected to the computer, and the micro HDMI connector should be connected to the robot, and the connector position of the robot is shown in Figure 2-2. After the connection, it can be displayed by powering on the robot.

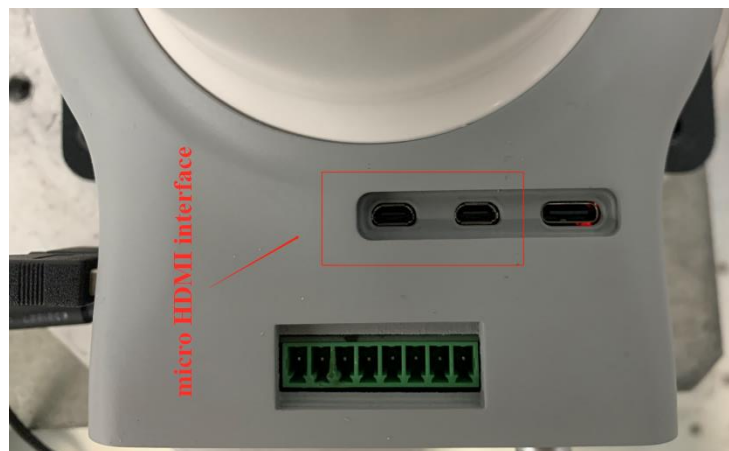


Fig. 2-2 Robot micro HDMI Connector

2.2.2 Remote Connection

1. Power on the robot and connect the robot arm to the PC by using the network cable.
2. Open the software "VNC Viewer" on the remote port, as shown in Figure 2-3, then enter the IP address of the robot in the input box, and press Enter to connect.

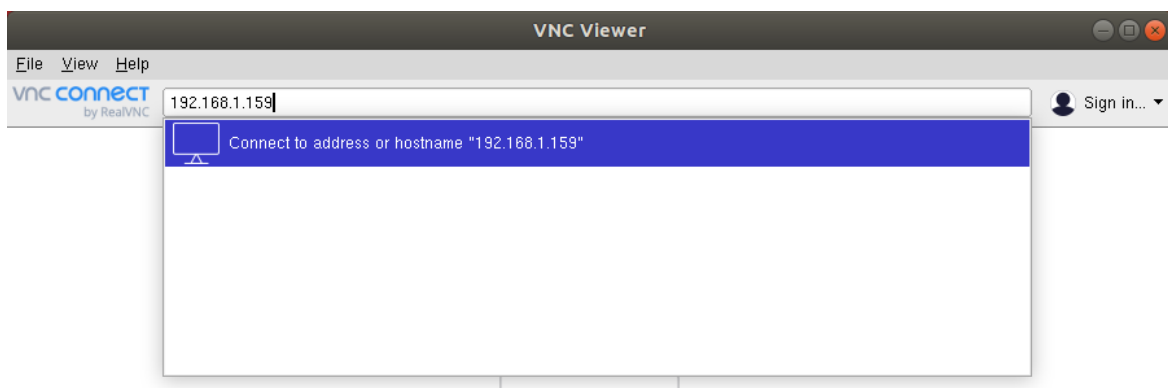


Fig. 2-3 Opening VNC Software

- 1, As shown in Figure 2-4, in the authentication window that pops up, enter the account and password of the robot port, and the default account name is "PI" and the password is "elephant". Click "OK" button to connect when you have finished typing.

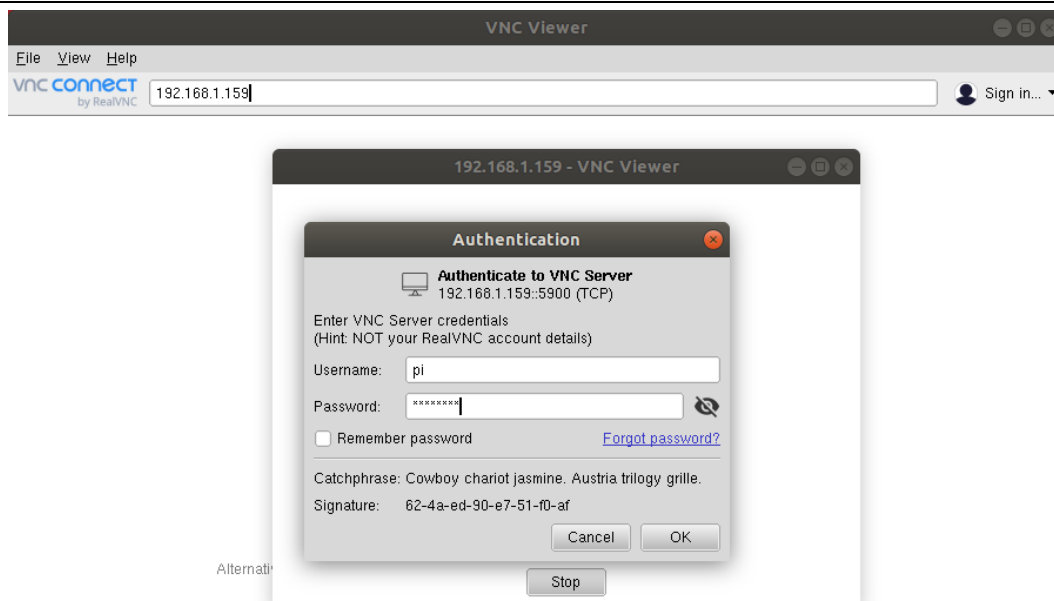


Figure 2-4 Remotely Connecting to Robot

2.3 Quickly Building a Runnable Project

2.3.1 Preparation Work

1, Preconditions

- 1) Check that the robot arm is in perfect condition and undamaged
- 2) Install the fixed robot arm
- 3) Connect the power adapter and provide the appropriate voltage
- 4) Connect the visual devices (displayer /PC remote connection)
- 5) Connect the keyboard and mouse (when connected to displayer)
- 6) Keep the emergency stop switch on

2.3.2 Flow Chart

The following is the program editing flow chart, as shown in Fig. 2-5.

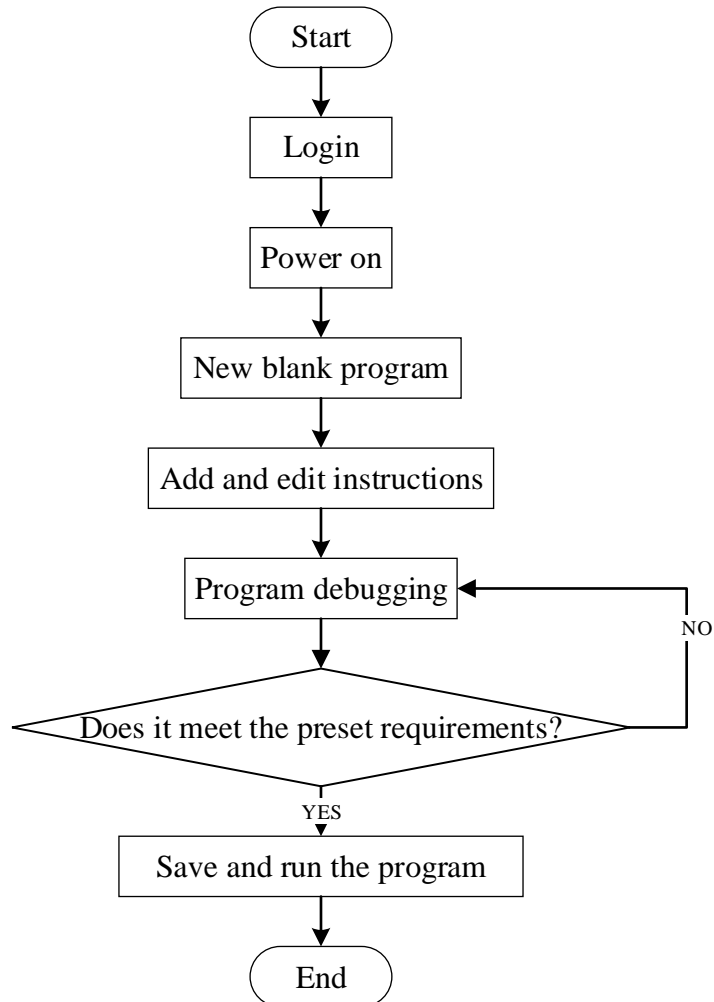


Fig. 2-5 Program Editing Flow Chart

2.3.3 Specific Steps

2.3.3.1 Login

After the system starts successfully, the login interface of RoboFlow operating system is displayed, as shown in Figure 2-6.

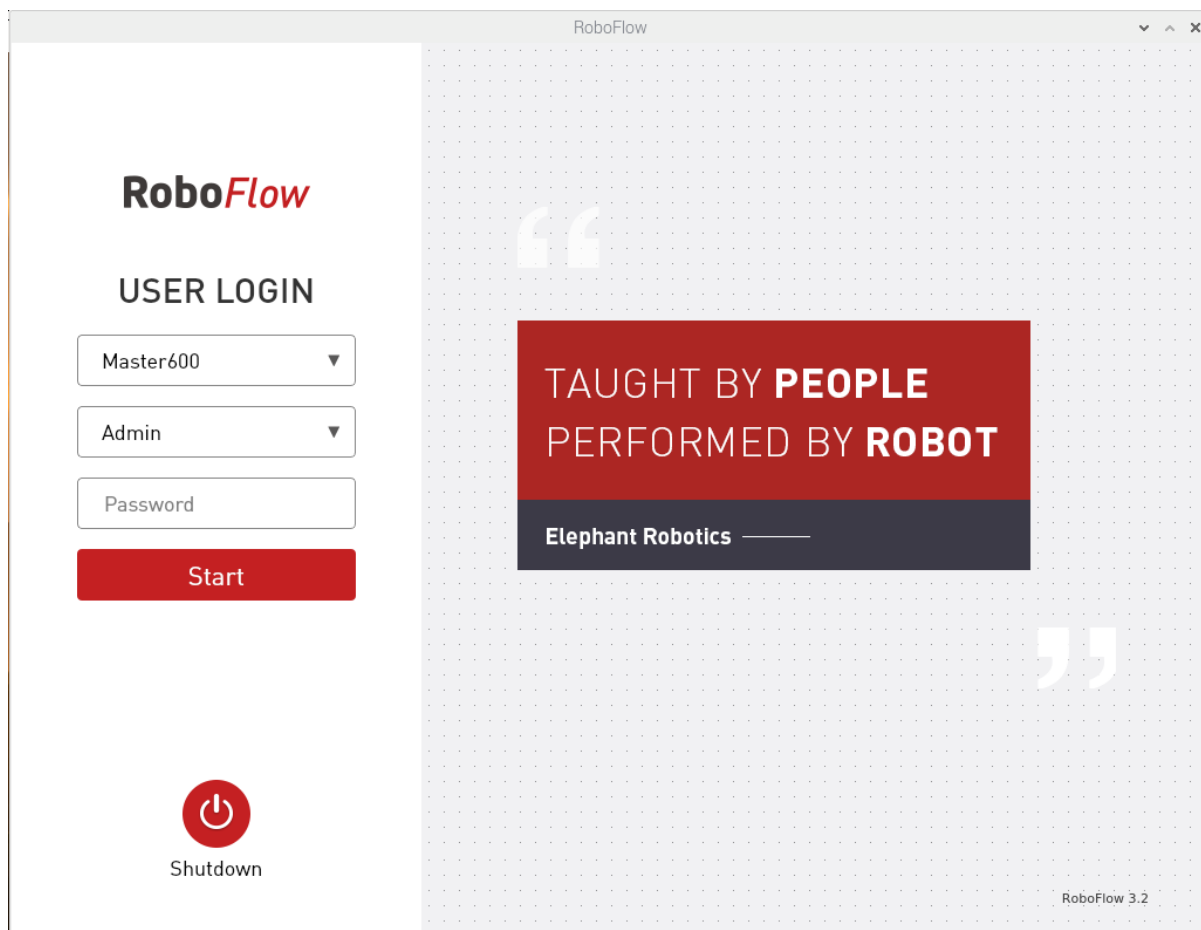


Fig. 2-6 Login Interface

Select the login user name "Admin" or other administrator user name (only administrators are allowed to edit and debug programs), click the password box, and a pop-up window will appear as shown in Figure 2-7.

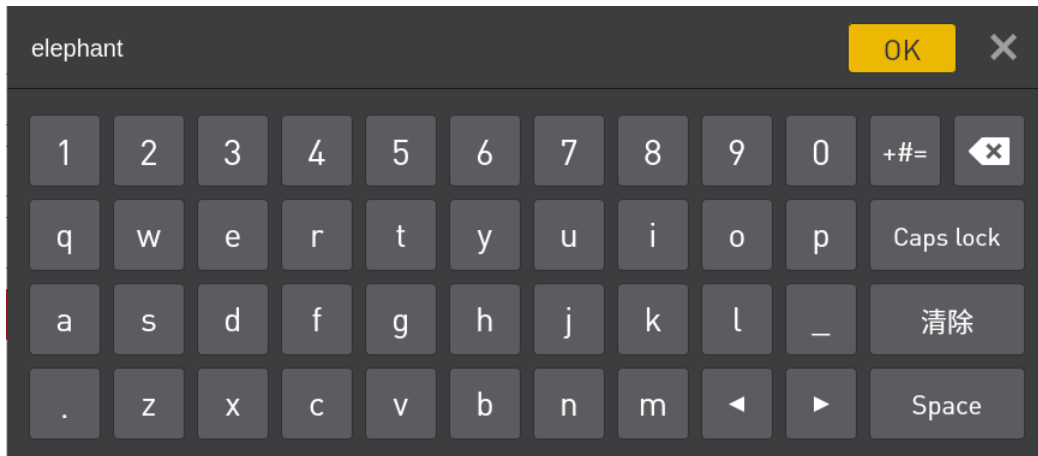


Fig. 2-7 Input Keyboard

The default login password of the user name "Admin" is "elephant" (please enter the corresponding login password if you select other administrator user name), and enter the password and click "OK" to return to the page shown in Figure 2-6. Then click "Login" again to log in successfully.

2.3.3.2 Power On

After successful login, the main menu is displayed, as shown in Figure 2-8.

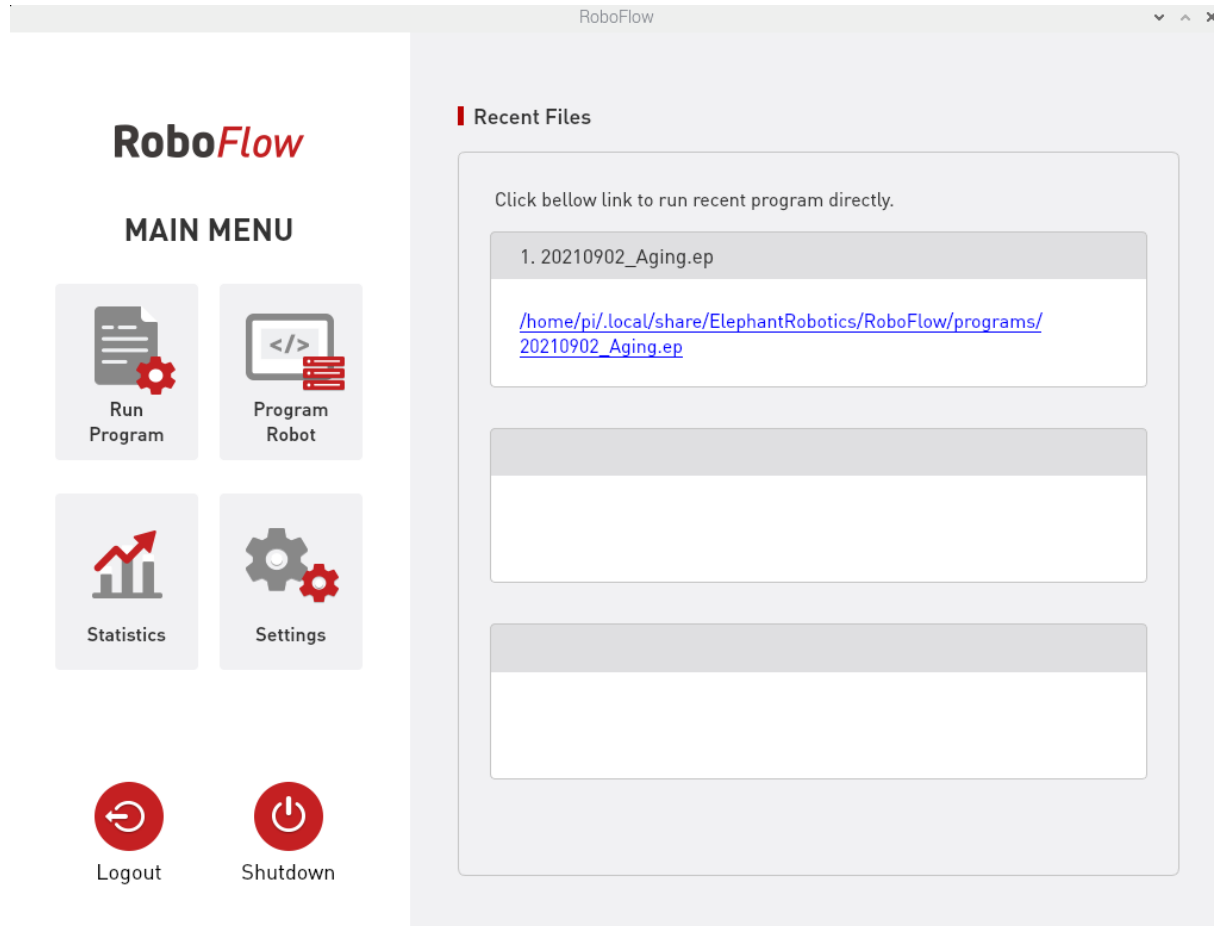


Fig. 2-8 Main Menu

On the main menu interface, if you select "Configuration Center", the page shown in Figure 2-9 is displayed (not powered on yet).

When ensuring that the emergency stop knob is not pressed, you can click the "Start Robot" button as shown in Figure 2-9. The interface will change at this point, and the "Being Powered On" icon will be displayed, as shown in Figure 2-10. If the power-on is successful, the "Normal State" will be displayed, as shown in Figure 2-11. If it fails, please check whether any steps are missing.

After completing the previous step, you can click the "< Main Menu" button in the Configuration Center to return to the main menu.

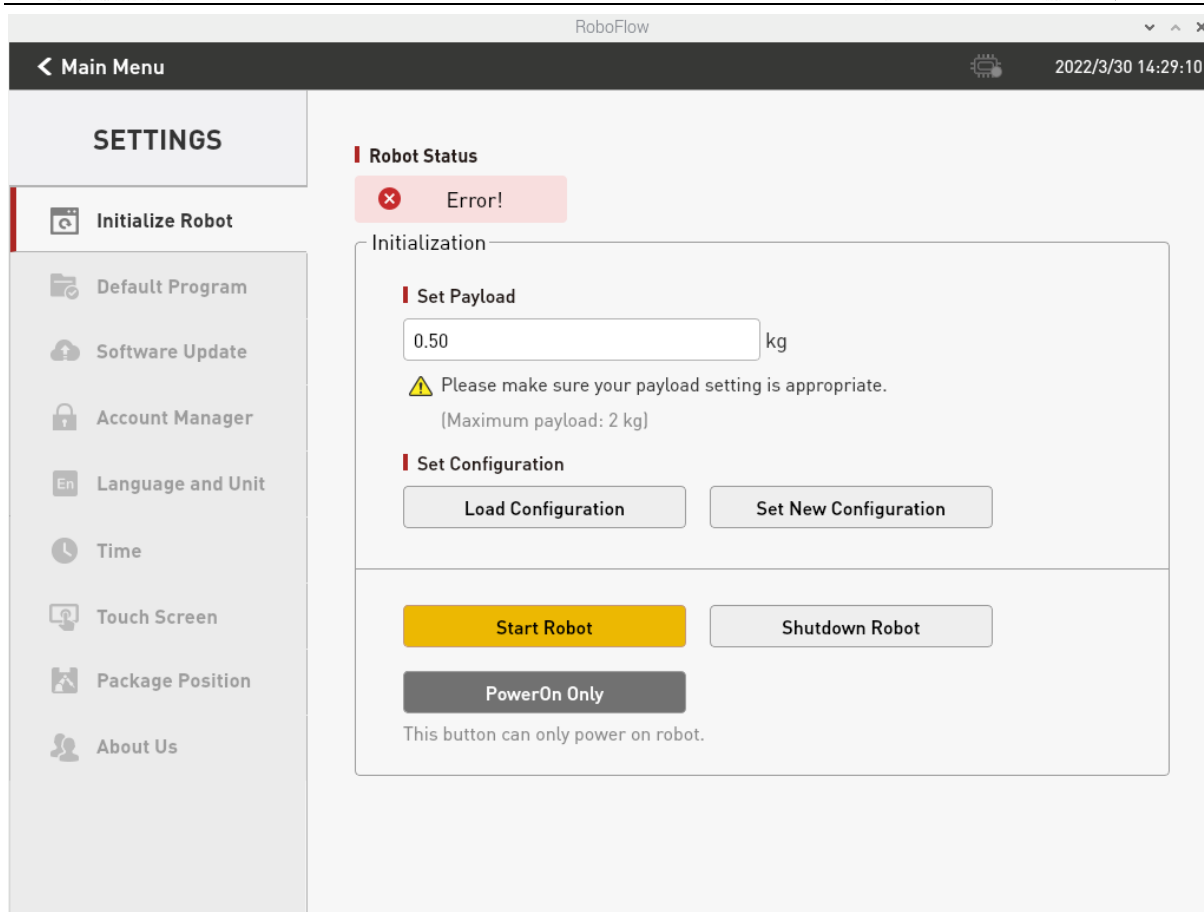


Fig. 2-9 State of Being Not Powered On

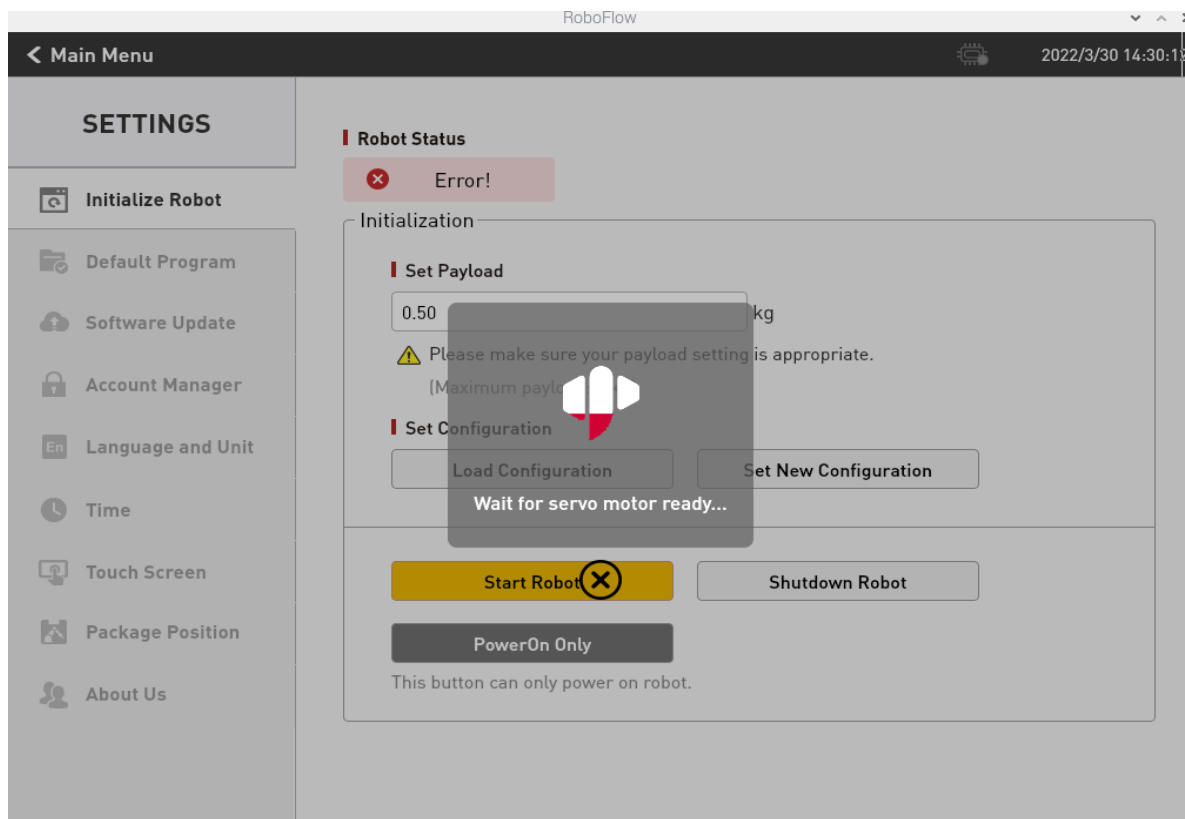


Fig. 2-10 In the Process of Being Powered On

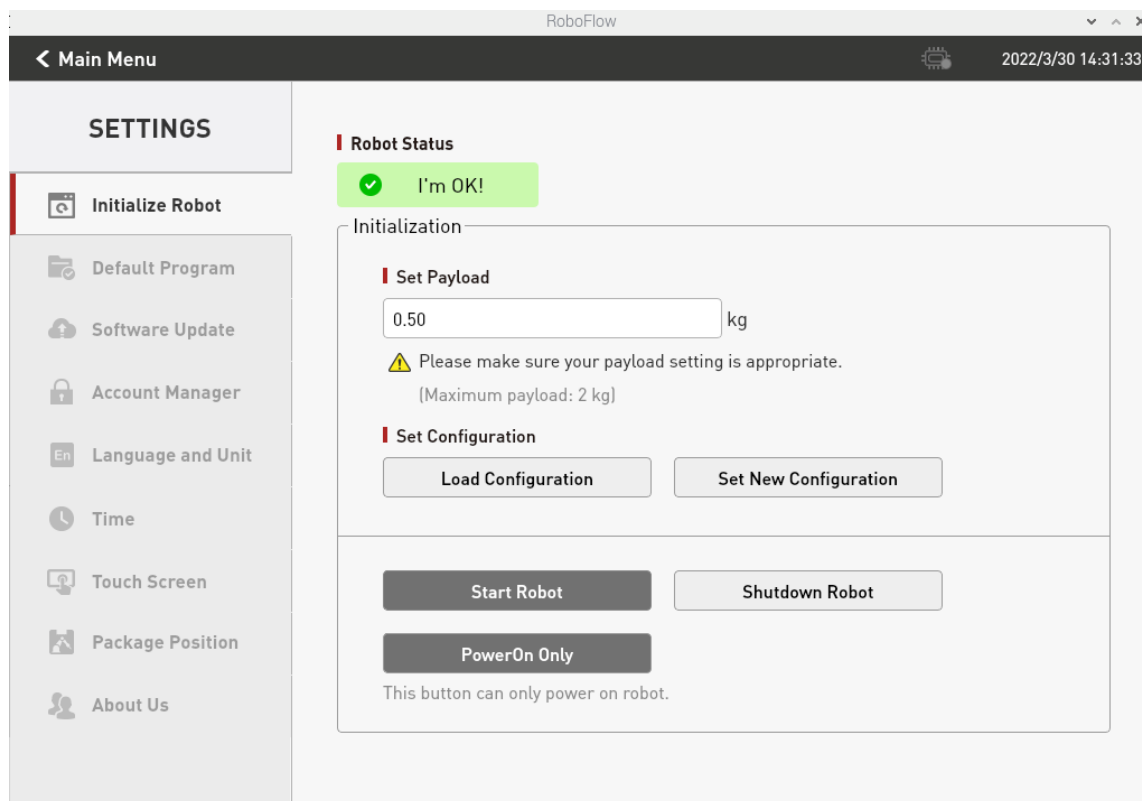


Fig. 2-11 Completion of Being Powered On

2.3.3.3 New Blank Program

As shown in Figure 2-12, click "Write Program" and then select "Blank Program".

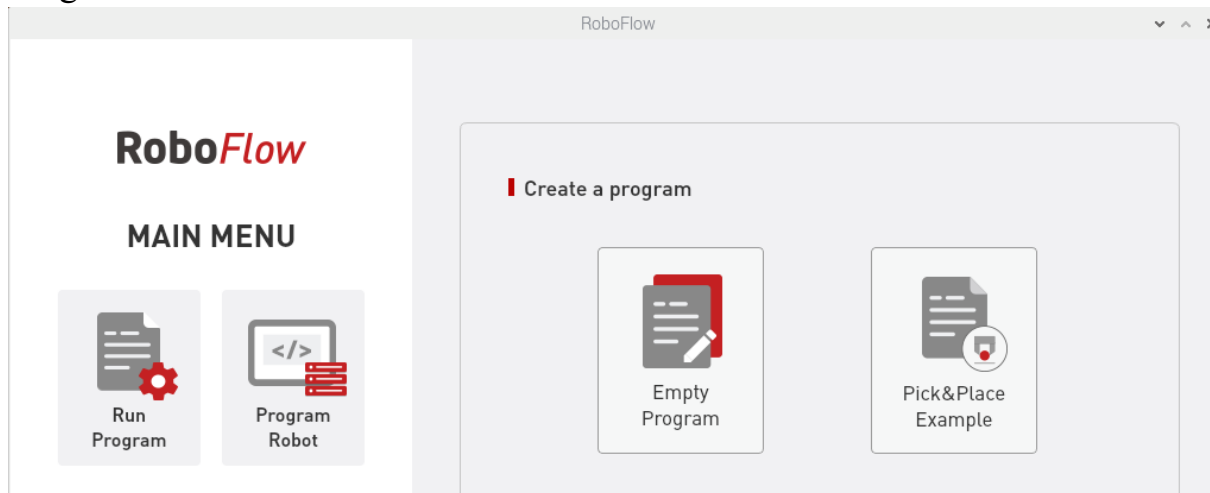


Fig. 2-12 Select "Blank Program"

After the previous step, the program editing interface will be displayed, as shown in Figure 2-13.

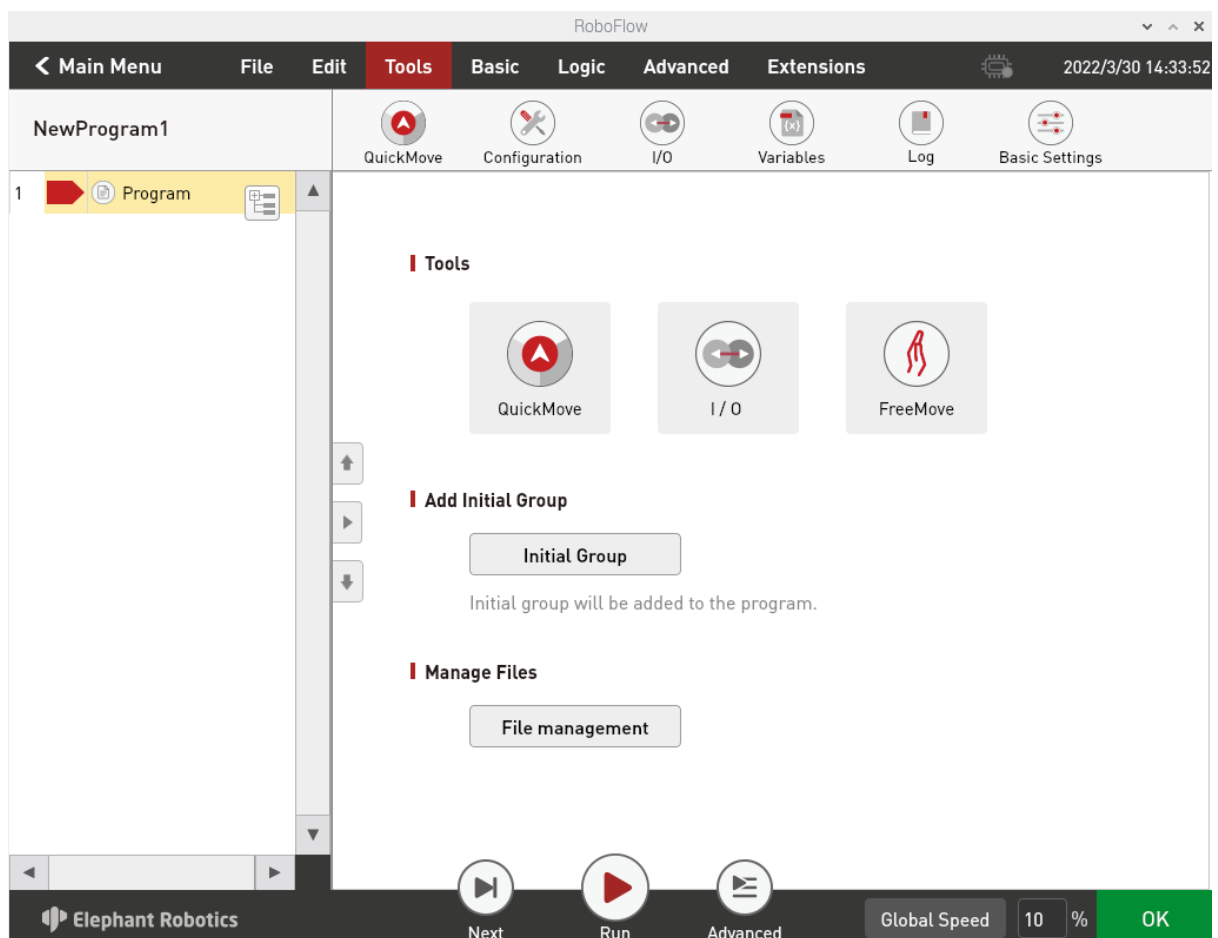


Fig. 2-13 Enter into Program Editing Interface

2.3.3.4 Quick Moving

As shown in Figure 2-13, by clicking "Quick Moving", the window as shown in figure 2-14 will pop up. The control mode is divided into Cartesian coordinate control and joint control. The movement mode of moving can be continuous movement or step movement.

The Cartesian coordinate control refers to the linear movement on xyz-axis, and the robot can be controlled to move along the direction of Cartesian coordinate system by clicking the button corresponding to the direction of Cartesian coordinate system. Note that before using Cartesian control, you need to make sure that the second joint and the third joint shows a certain Angle.

Joint control provides the keys used by the operator to manually operate the robot with visual software and control the joint movement of the robot. The control keys of each joint are divided into two directions, and the Angle data of each axis can be seen.

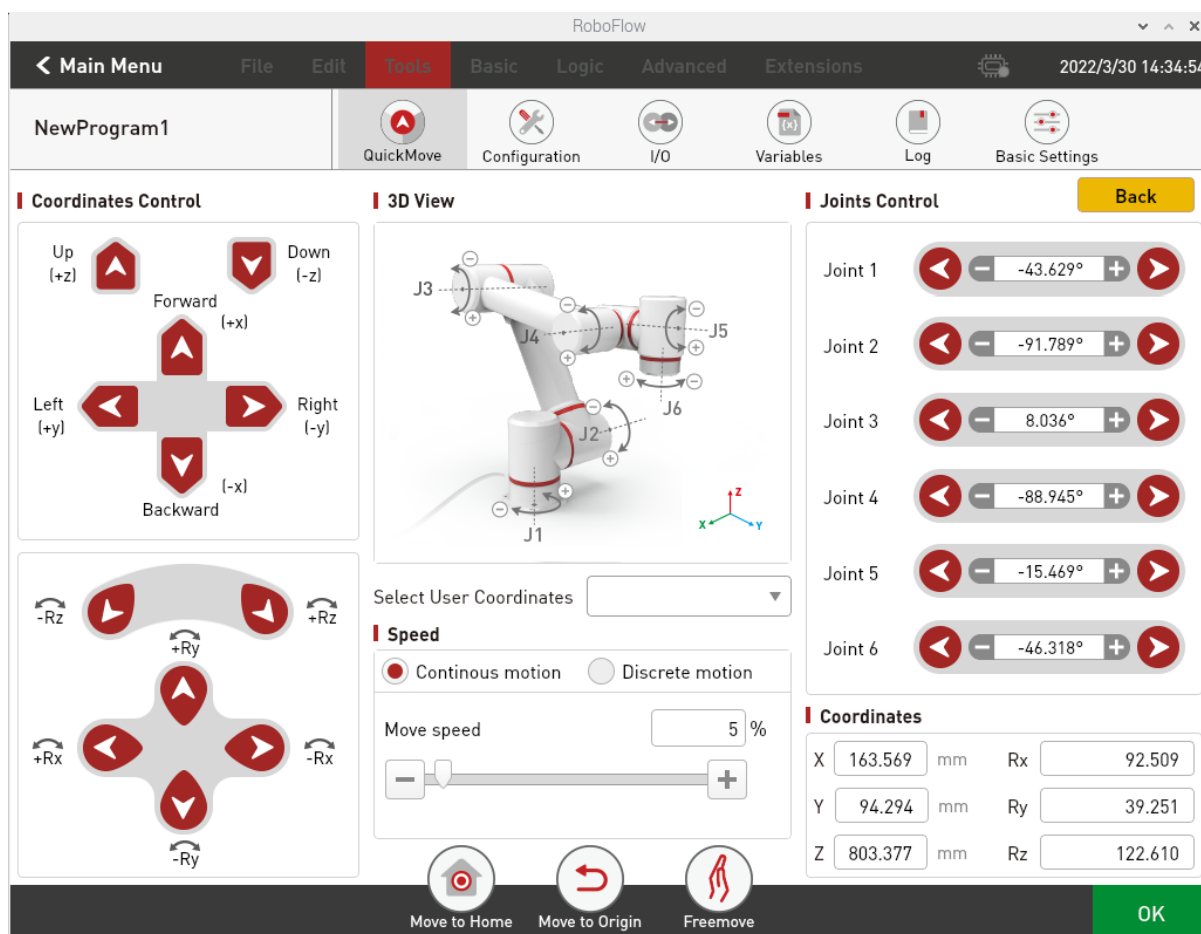


Fig. 2-14 Enter into Quick Moving Interface

2.3.3.5 Add and Edit Instructions

As shown in Figure 2-15, there are two waypoints to add: absolute point instruction and teaching two points (that is, to use the quick moving tool to manually operate the robot, control the robot movement to a certain pose, return, click "Save the Current Point". The teaching steps of the two points are the same. If you need to verify the saved point, you can manually control the robot to move to the teaching point by long pressing the "Move to this point" button.).

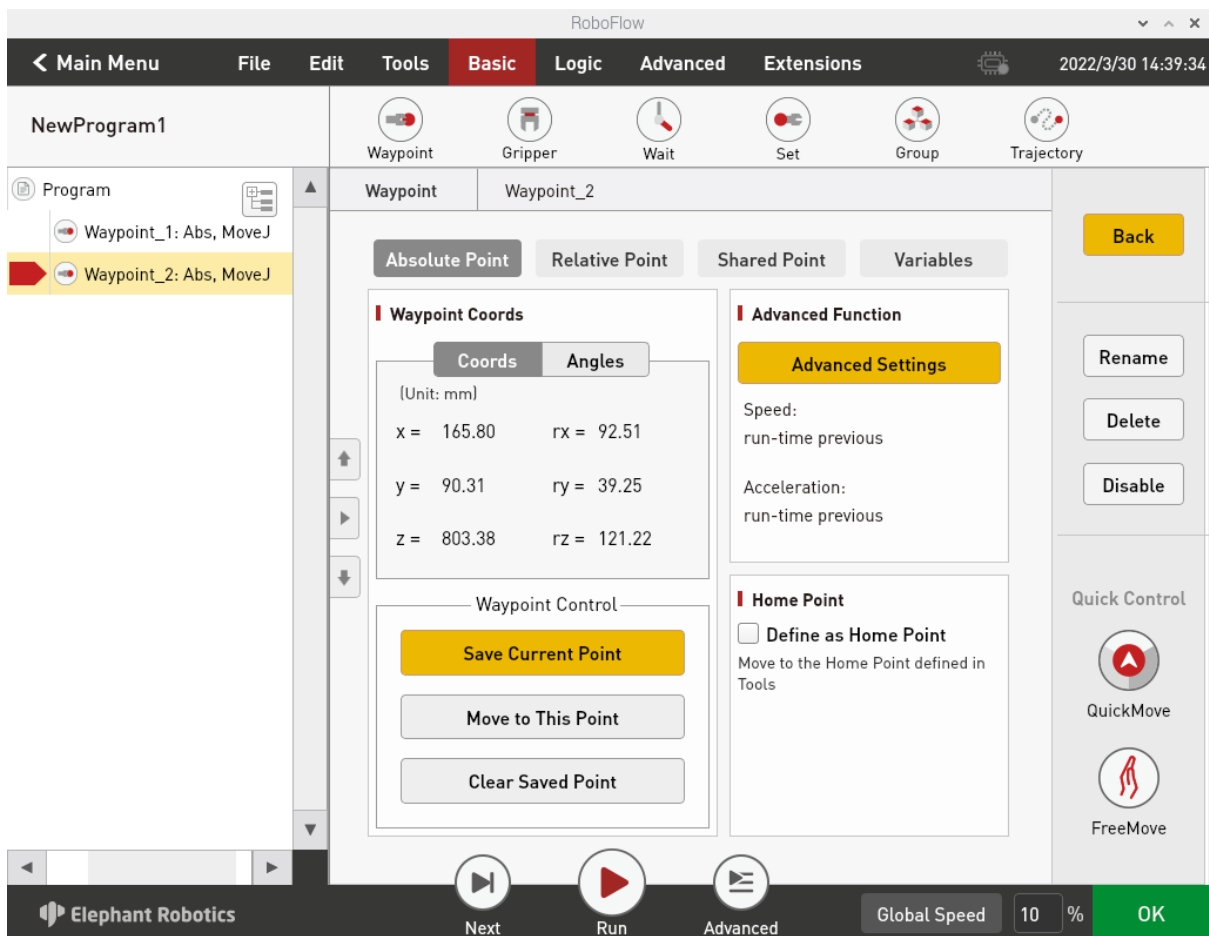


Fig. 2-15 Program Editing

2.3.3.6 Debugging Program

As shown in Figure 2-16, in addition to the "Next" and "Run" functions provided in the program running control bar, by clicking the "Advanced Functions", you can enter the interface of more settings.

Among them, the "Next" function corresponds to the step by step execution of the program, and one click only runs one step, if you need to continue to run, click "Next". The "Run" function corresponds to automatically running the program once.

In "Advanced Functions", you can set the number of times the loop runs, or you can run the loop indefinitely. You can also control whether the program runs in automatic or manual mode. In automatic operation mode, you can use "Next", "Run", and cycle operation. On the interface shown in Figure 2-17, you can select "Manual Run Mode" and then select "Run" or "Infinite Loop" in cycle operation to enter the interface in manual run mode, as shown in Figure 2-20.

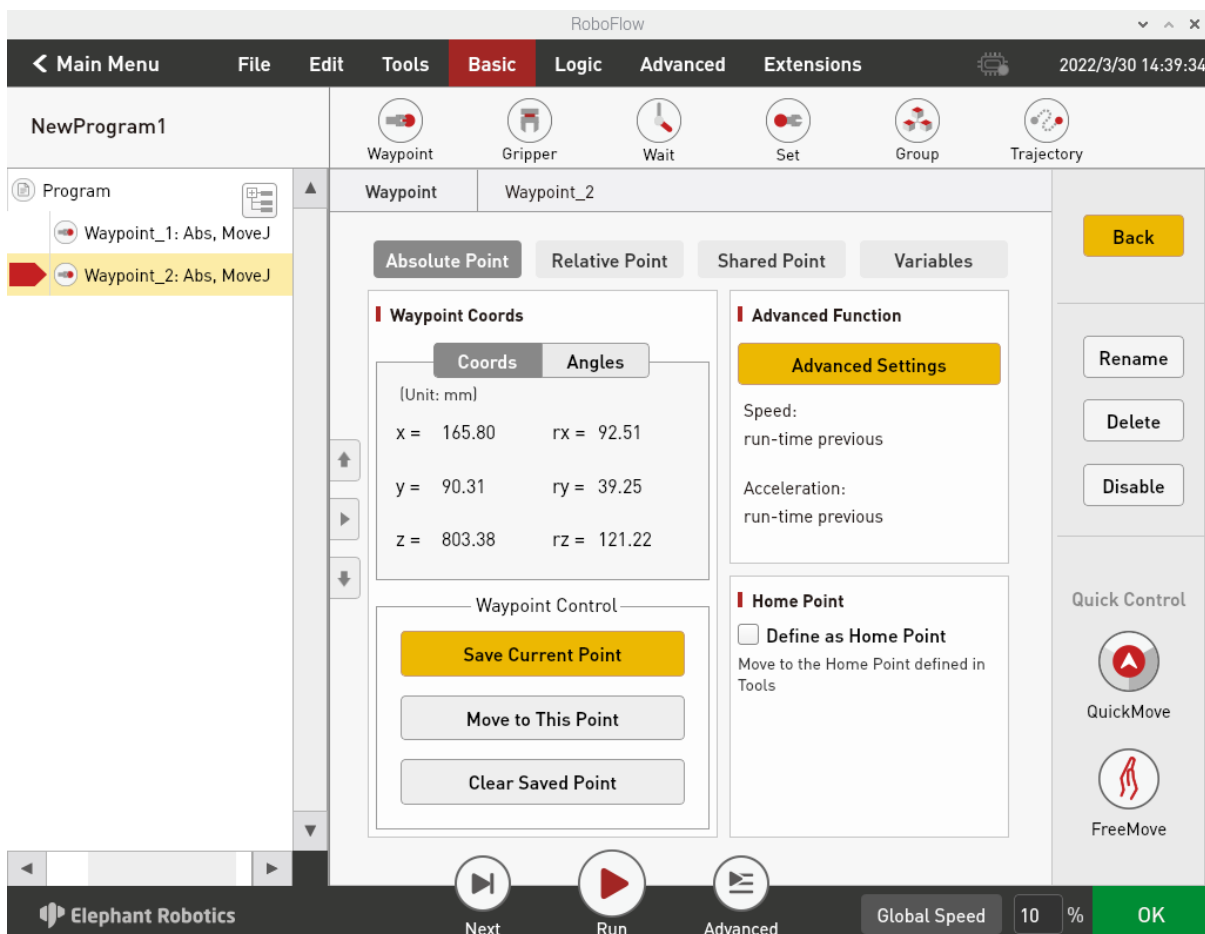


Fig. 2-16 Debugging Program

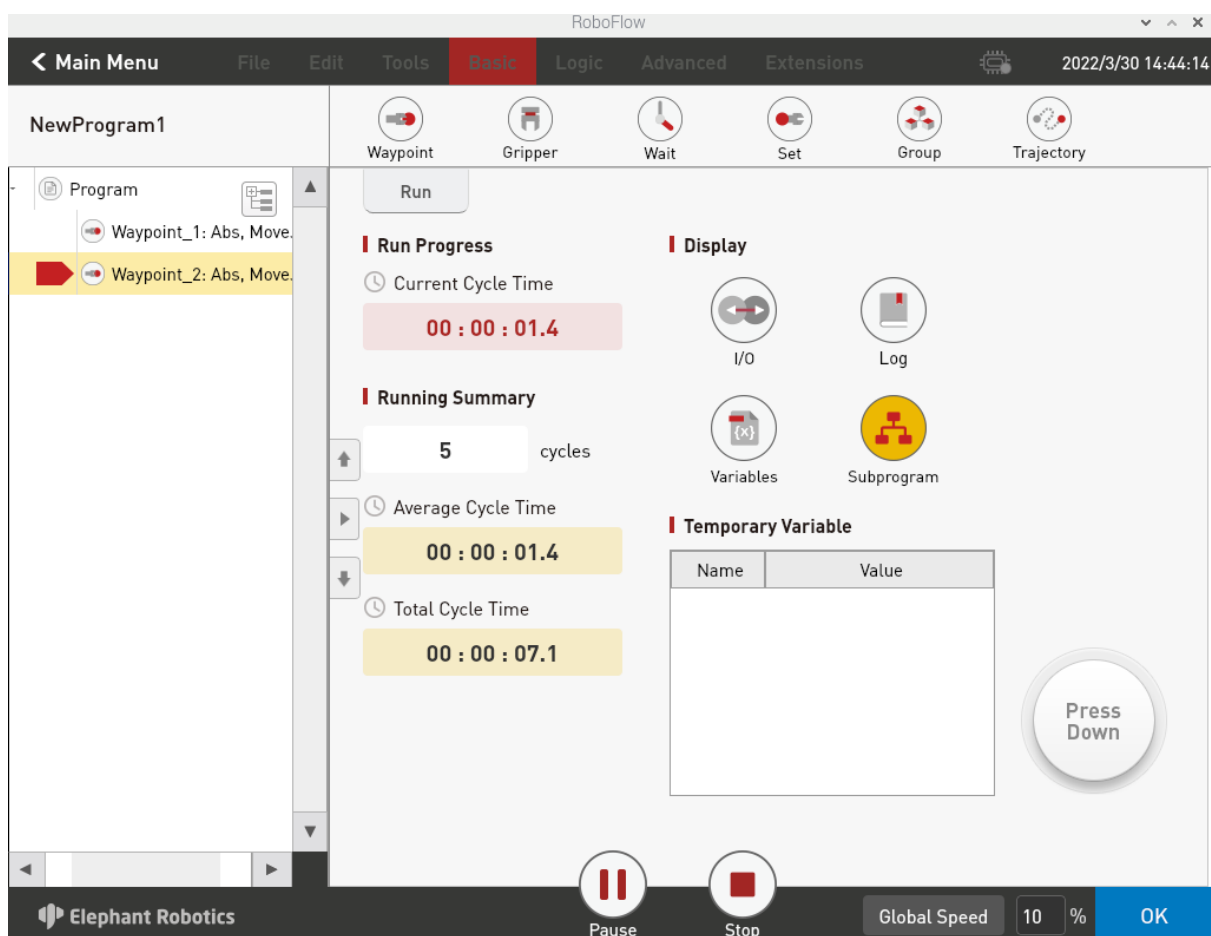


Fig. 2-17 Debugging Program in Manual Mode

If you use manual mode to debug the program, you need to press on the "Press Down" button to continue running. If you release the button, the program pauses, and press it again to continue running.

2.3.3.7 Free Movement

As shown in Figure 2-18, you can click the free movement button to make the robot arm enter the free movement mode, and at this time, you can move joint 1, joint 2 and joint 3, and if you need to move joint 4, joint 5 and joint 6, you need to press the ATOM button at the end, as shown in Figure 2-19.

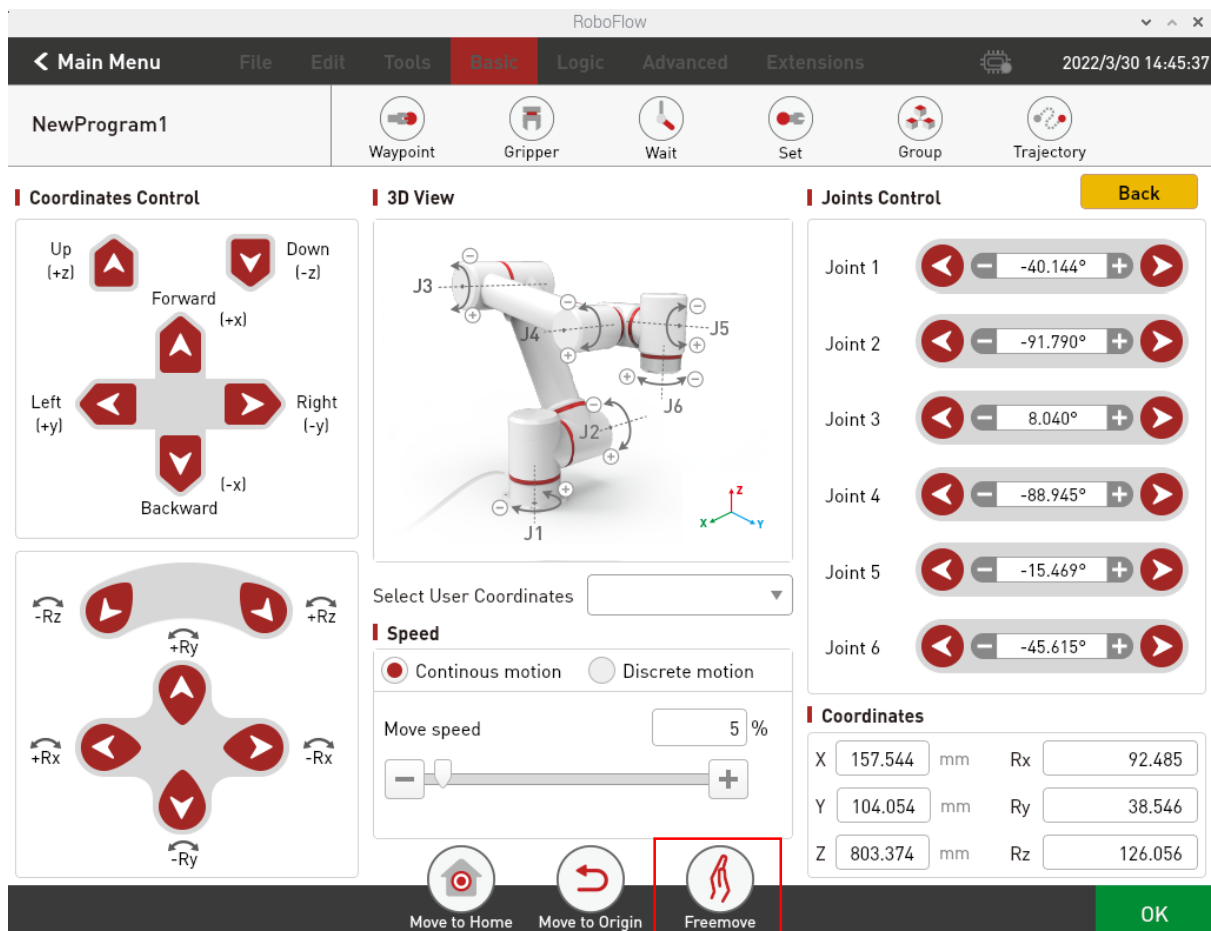


Fig. 2-18 Free Movement



Fig. 2-19 ATOM Button at the End

3 Product Introduction

3.1 Overview

Professional Six-axis Cooperative Robot Arm

It uses a Raspberry Pi microprocessor and built-in RoboFlow visual programming software, and is a teaching-oriented and commercially developed machine "assistant" for Elephant Robotics. It has rich extension interfaces, supports a variety of development languages, is easy to get started visual programming, has secondary development of the communication protocol and SDK, with a number of peripherals to choose, providing commercial teaching, commercial development of infinite possibilities.

3.2 Product Appearance and Composition

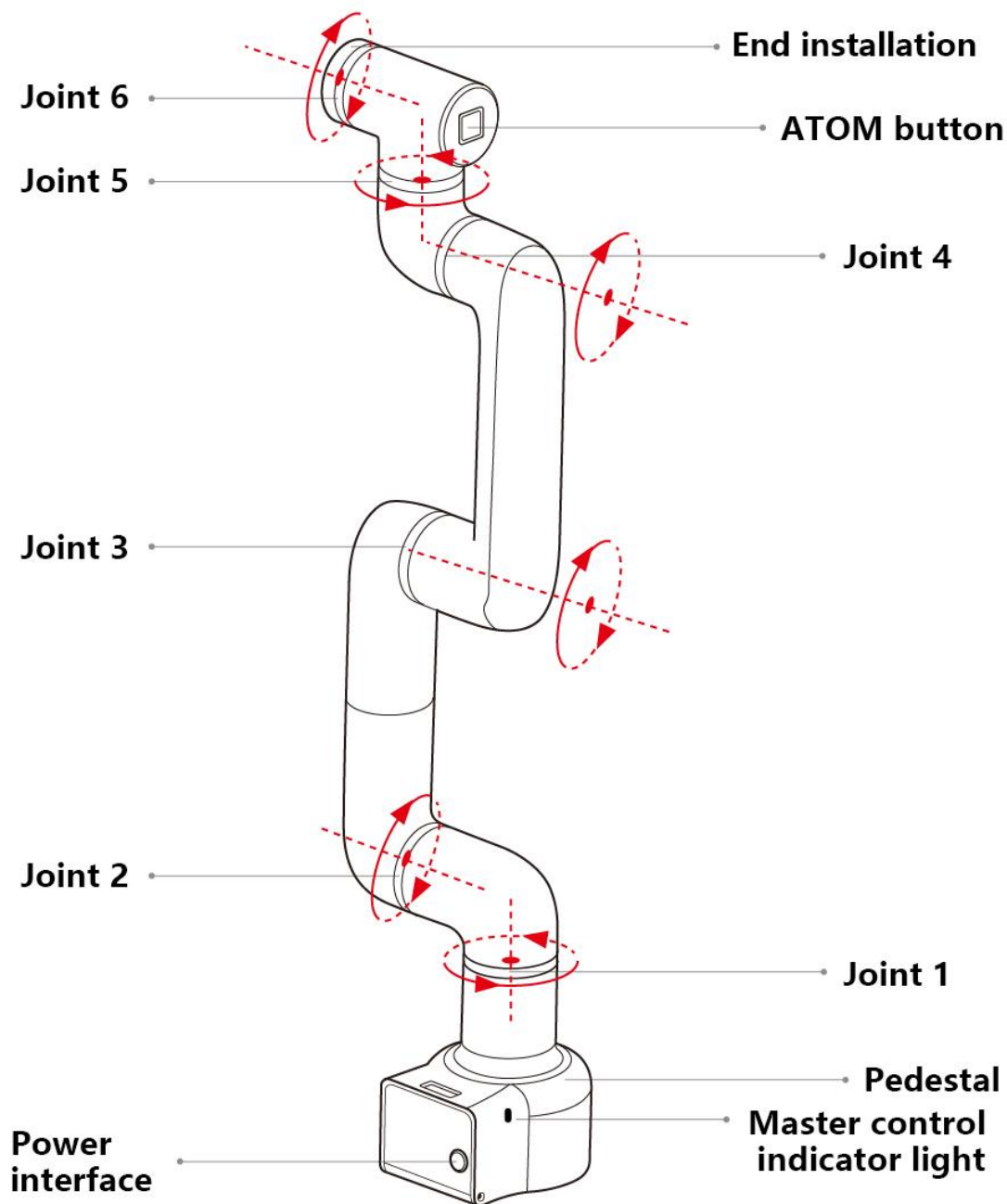
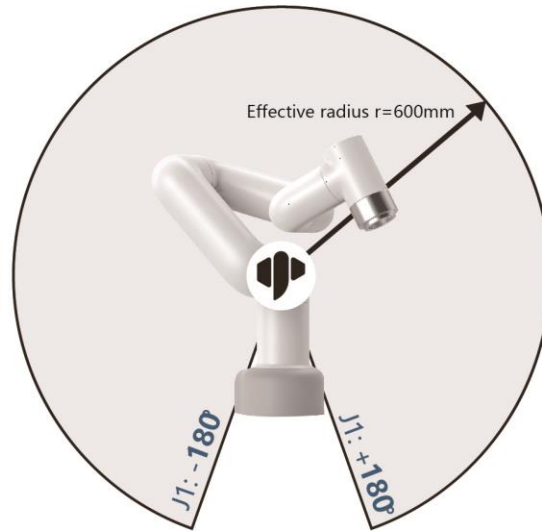
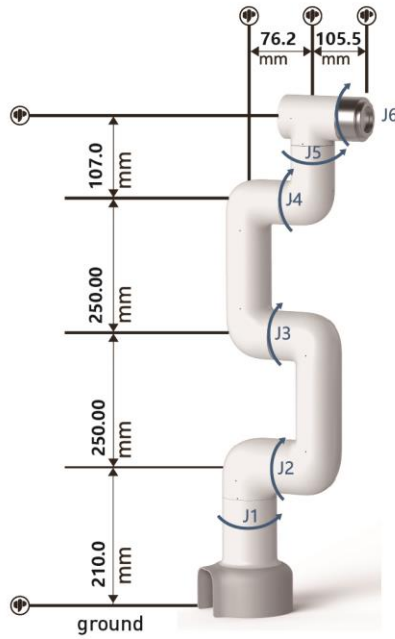


Fig. 3-1 Product Composition

3.3 Working Principles and Specifications

3.3.1 Working Space



3.3.2 Coordinate System

DH parameters and coordinate system:

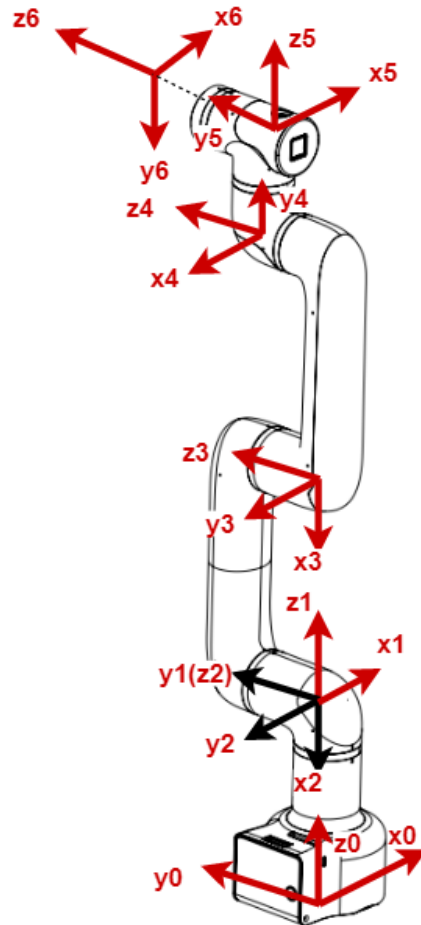


Fig. 3-2 DH Coordinate System

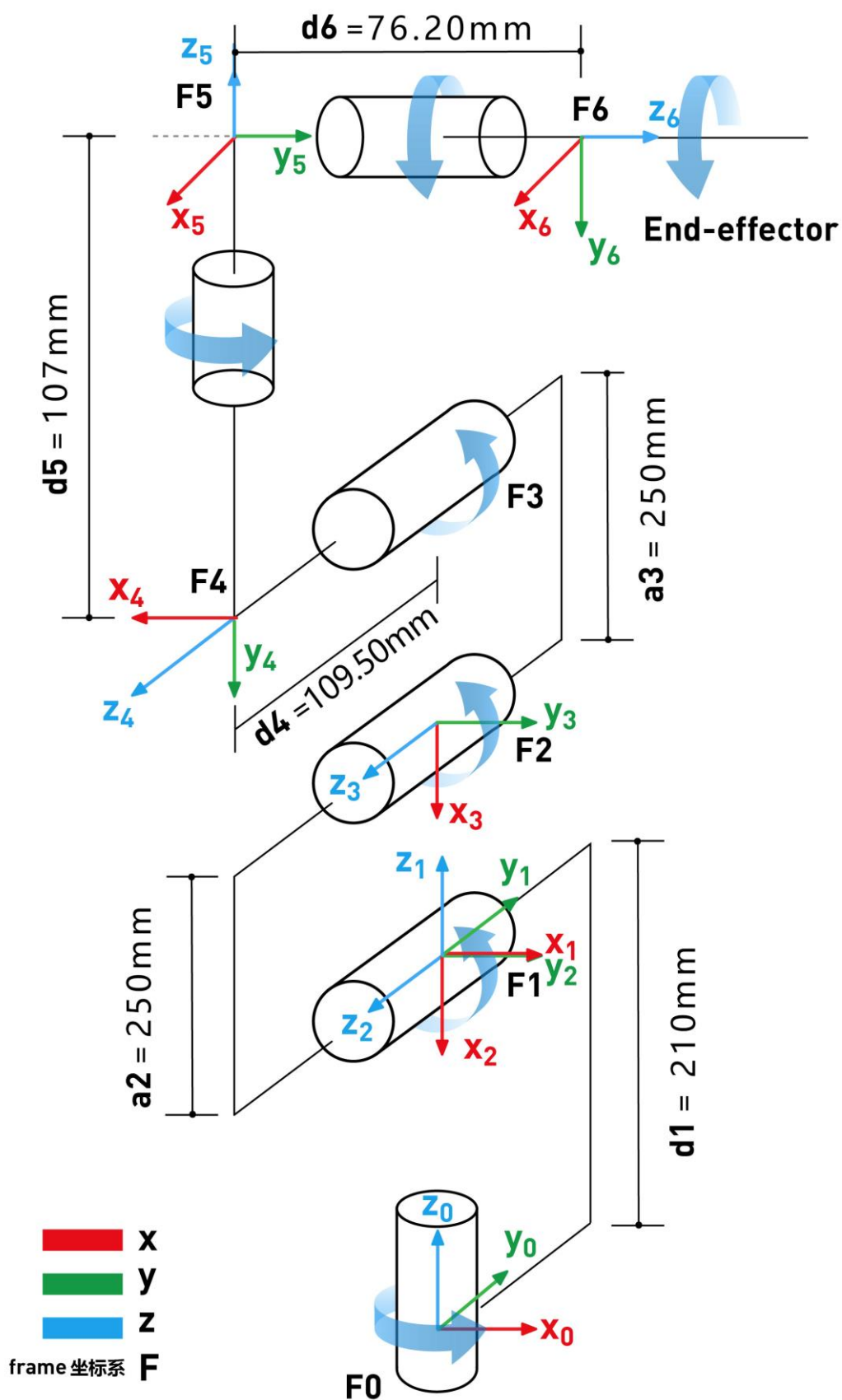


Fig. 3-2 DH Coordinate Parameter

SDH parameter list:

j	theta	d	a	alpha	offset
1	q1	210	0	-1.5708	0
2	q2	0	-250	0	1.5708
3	q3	0	-250	0	0
4	q4	109.5	0	-1.5708	1.5708
5	q5	107	0	-1.5708	3.14159
6	q6	76.2	0	0	0

Joints movement:

For the rotation of each joint in Figure 3-1, the direction of arrows in the figure is the positive direction of the joint.

Cartesian coordinate movement:

Taking the coordinate system $\{x_0, y_0, z_0\}$ in Figure 3-2 as the standard, the X axis is perpendicular to the forward direction of the fixed base, the Y axis is perpendicular to the left direction of the fixed base, and the Z axis is perpendicular to the upward direction of the fixed base.

3.3.3 Moving Functions

The movement modes of the robot arm include MoveJ, MoveL, MoveC and Jog modes

3.3.3.1 MoveJ

The joint movement moves from point A to point B, and each joint moves from the joint Angle corresponding to point A to the joint Angle corresponding to point B. In the process of joint movement, the running time of each joint axis should be consistent and the end point should be reached at the same time.

3.3.3.2 MoveL

In rectilinear movement, the path from point A to point B is straight, as shown in Figure 3-4

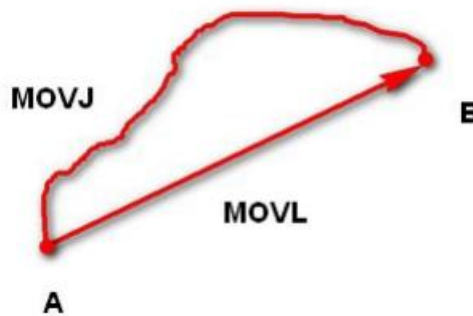


Fig. 3-4 MoveJ and MoveL Movement Modes

3.3.3.3 MoveC

The moving track of arc mode is an arc. The arc track is a space arc, which is jointly determined by the current point, the any point of the arc and the end point of the arc. An arc always starts from the starting point and passes through any point of the arc to the end point, as shown in Figure 3-5.

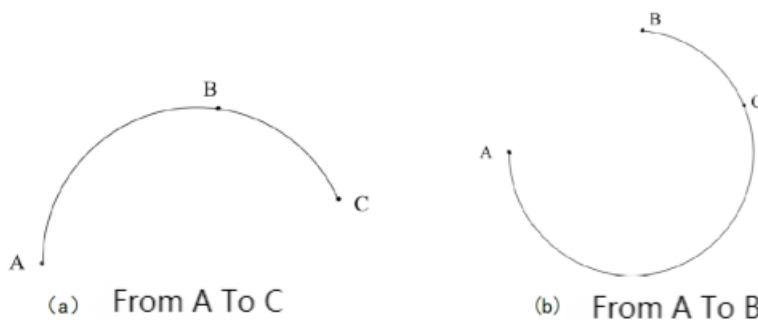


Fig. 3-5 MoveC

3.4 Technical Specifications

3.4.1 Technical Parameters

Name	myCobot Pro 600	
Maximum load	2000g	
Maximum extension distance	600mm	
Range of movement	Joint 1	-180°-180°
	Joint 2	-270°-90°
	Joint 3	-150°-150°
	Joint 4	-260°-80°
	Joint 5	-168°-168°
	Joint 6	-174°-174°
Maximum movement velocity	115 degrees per second	
Repeated positioning accuracy	±0.5mm	
Power voltage	100-240VAC 50-60Hz	
Communication mode	WIFI, Ethernet Port	
I/O interface	15 I/O interfaces	
Control software	RoboFlow	
Working environment	0°C ~ 50°C	

3.4.2 Size Parameters

Size parameters of myCobot Pro 600 are shown in Figure 3.9, and the end size parameters are shown in Figure 3.10

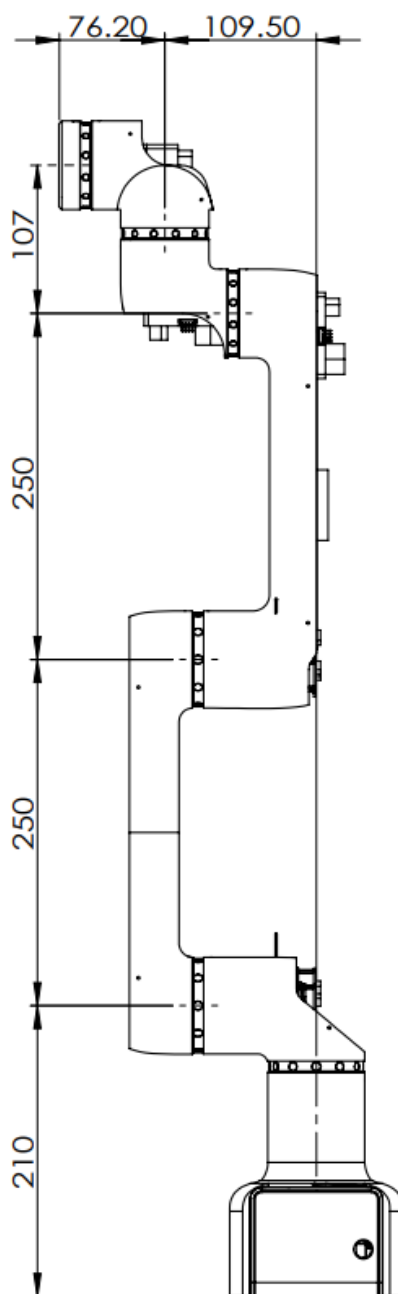


Fig. 3.9 Size Parameters of myCobot Pro

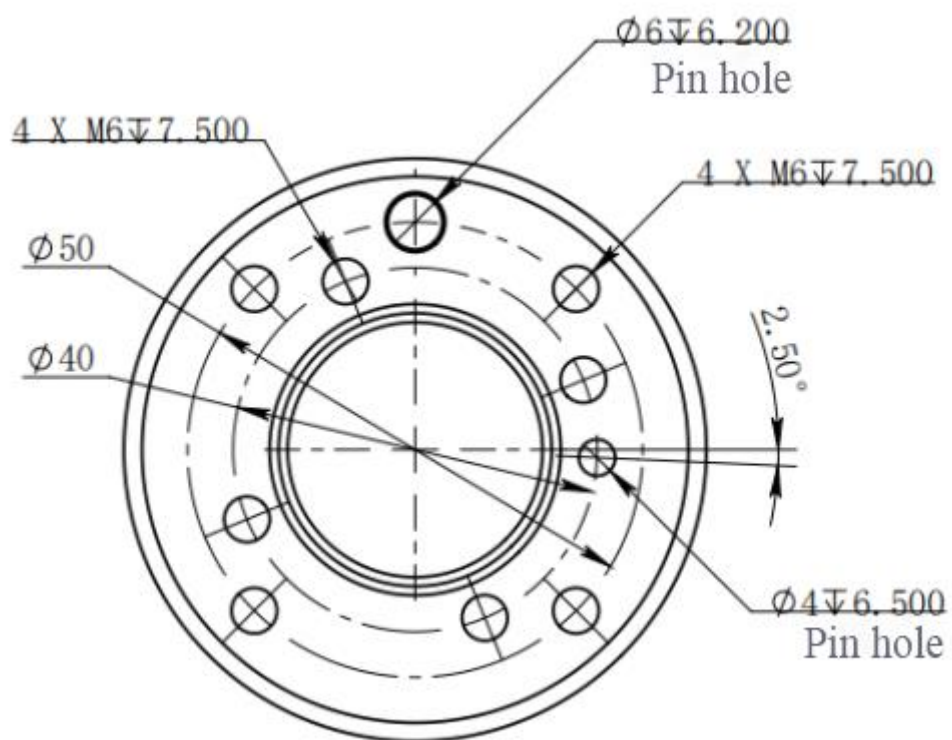


Fig. 3.10 Size Parameters of End Mounting Hole

4 Interface Specifications

4.1 Base Electrical Interfaces

4.1.1 Introduction to Base Electrical Interfaces

1 Figure 4-2 shows the electric in the front of base:

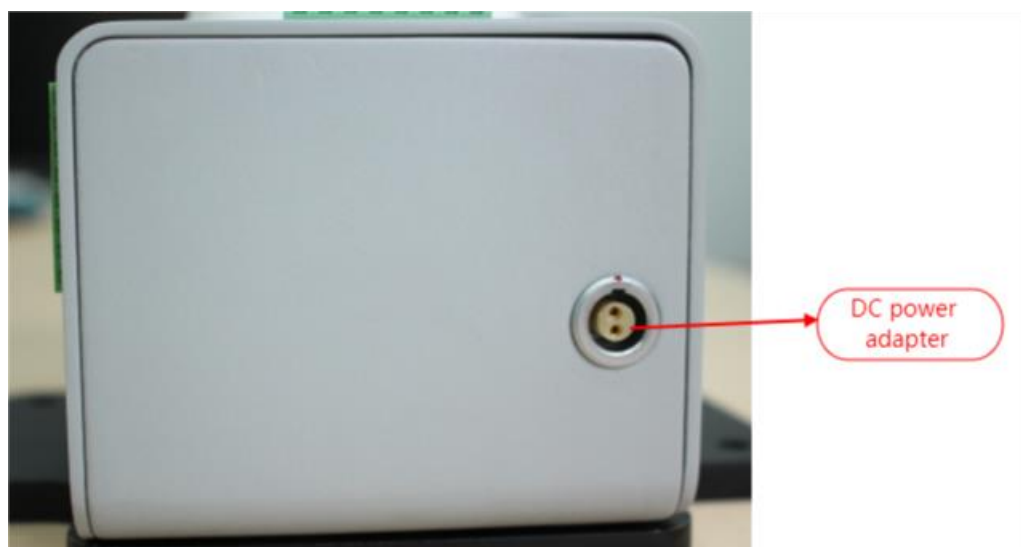


Fig. 4-1 Front View of the Base

2 Figure 4-2 shows the interfaces on the left side of the base:

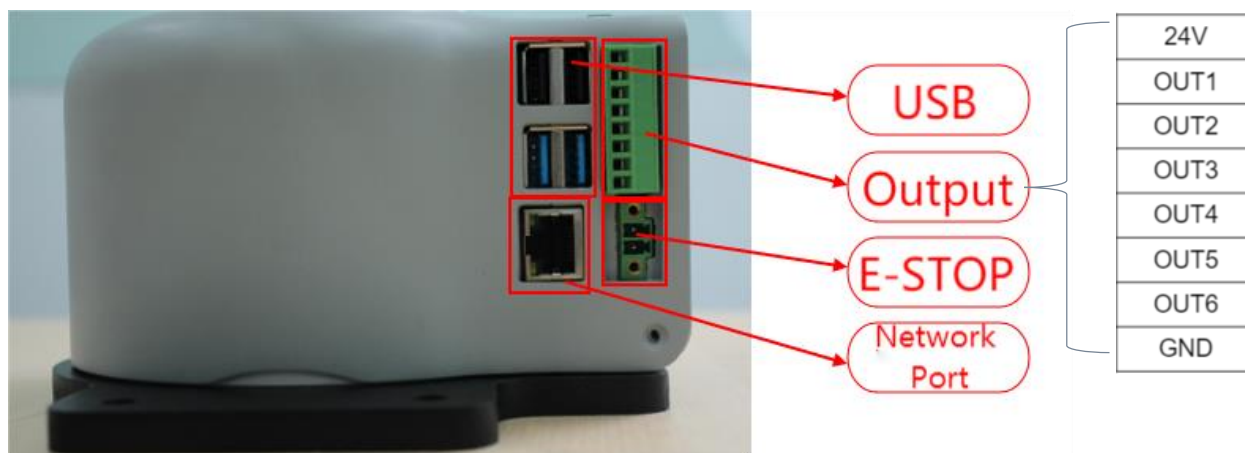


Fig. 4-2 Left Side View of the Base

3 Figure 4-3 shows the electrical interfaces on the base:



Fig. 4-3 Right Side View of the Base

4 Figure 4-4 shows the electrical interfaces on the upside of the base:

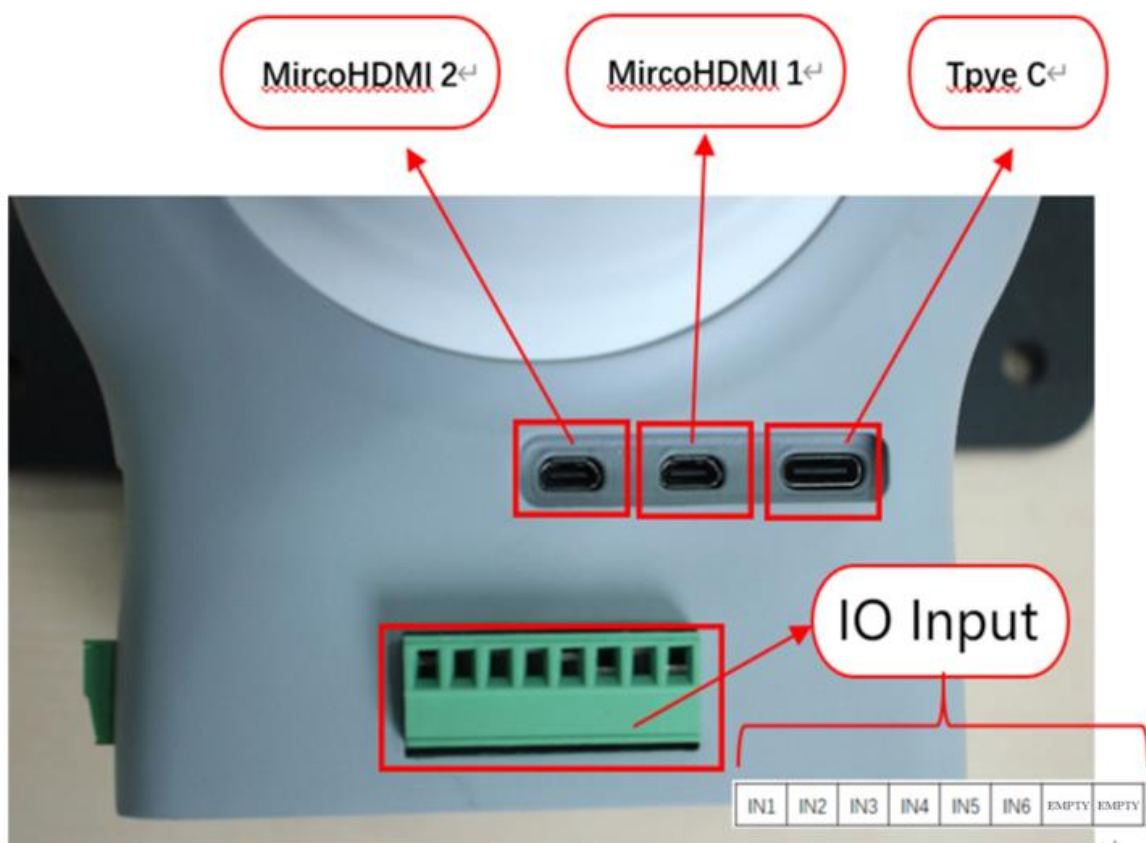


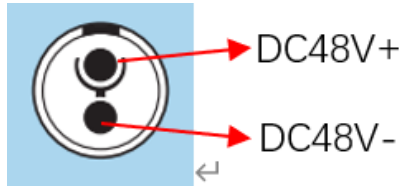
Fig. 4-4 Top View of the Base

4.1.2 Description of Base Electrical Interfaces

Serial No.	Layout Position	Types	Definition	Description
1	Front	DC power input	DC48V	External DC42V power input interface
2	Left side	USB Interface	USB2.0	Used for external extension devices such as mouse, keyboard and USB flash disk
3			USB3.0 (blue)	
4		Ethernet interface	Ethernet	Ethernet interface
5		24V	DC24V	DC24V Output
6		Digital output 1-6	OUT1	PNP Digital output signal 1
7			OUT2	PNP Digital output signal 2
8			OUT3	PNP Digital output signal 3
9			OUT4	PNP Digital output signal 4
10			OUT5	PNP Digital output signal 5
11			OUT6	PNP Digital output signal 6
12		GND	GND	GND
13		Emergency stop	ES1 +	External emergency stop control loop
14			ES1 -	
15	Upside	MircoHDMI1	MircoHDMI1	Display interface, use HDMI cable to connect the display screen
16		MircoHDMI2	MircoHDMI2	
17		Type C	Type C	Used for internal debugging
18		Digital input 1-6	IN1	PNP Digital input signal 1
19			IN2	PNP Digital input signal 2
20			IN3	PNP Digital input signal 3
21			IN4	PNP Digital input signal 4
22			IN5	PNP Digital input signal 5
23			IN6	PNP Digital input signal 6
24		Null interfaces	-	Reserved SRS485 interface
25			-	
26	Right side	Display lamp	-	Display the status of the master controller

1 DC power input interface:

This interface is connected with the DC48V power adapter interface, and its definition is shown in the following figure:



2 USB2.0 interface:

It is a serial bus standard 2.0 interface for data connection; and users can copy program files using USB interface and can also use the USB ports to connect peripherals such as mouse and keyboard.

3 USB3.0 interface (blue):

It is a serial bus standard 3.0 interface for data connection; and users can copy program files using USB interface and can also use the USB ports to connect peripherals such as mouse and keyboard.

4 Ethernet interface:

It is the port for the network data connection, and users can use Ethernet interface for communication interaction between PC and robot system, also can use it for Ethernet communication with other devices.

5 24V output:

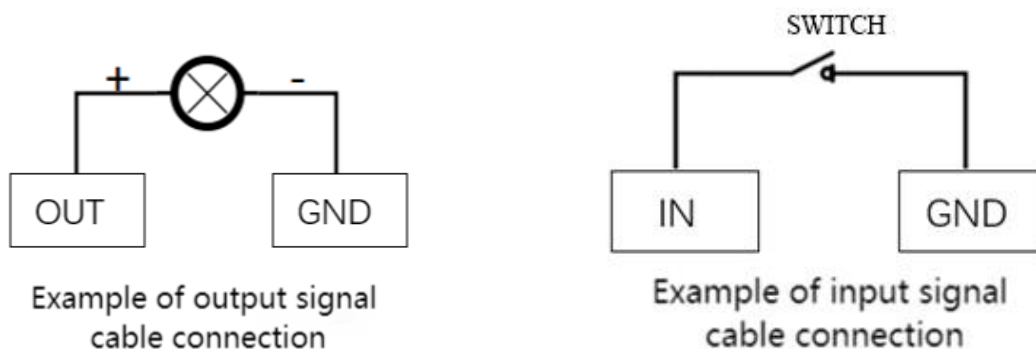
It is internal DC24V and available for user use.

6 Digital input/digital output:

It includes 6 digital input signals and 6 digital output signals, used for interaction with other equipment, which together with other equipment constitute an important part of the automation system.

For example, the user can control the electric gripper mounted on the output flange using a digital output signal or connect it to a PLC for easy signal interaction.

It should be noted that the input/output signal is in the form of PNP, and the following is the schematic diagram of external wiring:



7 Loop terminal of emergency stop:

It is connected with the emergency stop button box and can be used to control the emergency stop of the robot.

Note: the robot must be connected to the emergency stop switch, and ensure that the switch circuit of the emergency stop is connected.

8 MicroHDMI display interface:

Users can connect the MicroHDMI display interface to display the operation page to other device terminals.

9 Master control display lamp:

It can be used to determine whether the master controller of the robot arm is working properly, and after the robot is powered on, the red light will be on and the yellow light

4.2 Electrical Interfaces of Robot Arm End

4.2.1 Introduction to Robot Arm End

1 Figure 4-5 shows the side interfaces at the end of the robot arm:

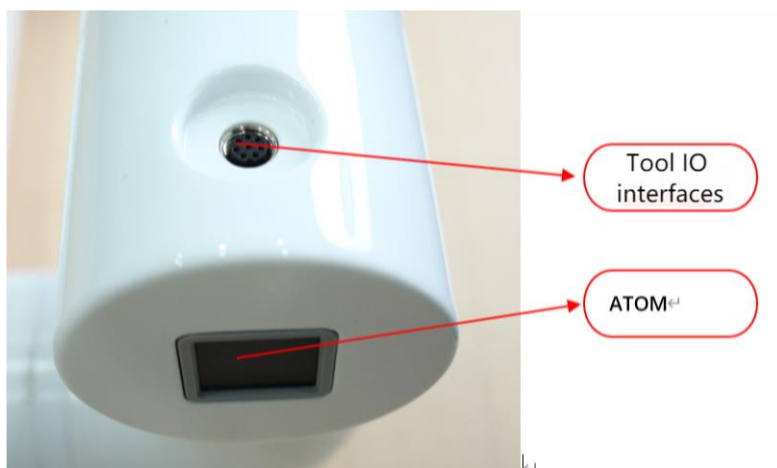
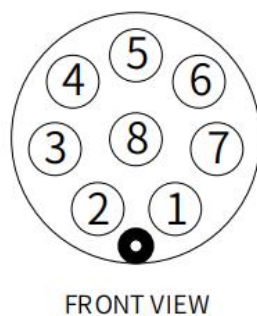


Fig. 4-5 Electrical Diagram of Robot Arm End

4.2.2 End Electrical Specification

1) Tool I/O interface:

Here's the tool I/O drawing as shown in the figure, with the Mycobot Pro600 robot providing one input and two outputs.



Tool I/O Drawing

Each tool I/O interface is defined in the following table, and it is noted that the input and output of tool I/ OS are all PNP types and the connection mode is the same as that of the bottom I/O interface.

Serial No.	Signals	Explanation	Color of Matching Line M8
1	GND	DC24V negative electrode	White
2	OUT1	Tool output interface 1	Brown
3	OUT2	Tool output interface 2	Green
4	485A	Reserved, undeveloped	Yellow
5	24V	DC24V positive electrode	Grey
6	IN1	Tool input interface 1	Pink
7	IN2	N/A	Blue
8	485B	Reserved, undeveloped	Purple

5 Operation Guide

5.1 RoboFlow Software Instructions

5.1.1 Overview

The raspberry is the operating system of the Elephant Collaborative Robot. It provides a human-computer interaction interface, which is convenient for operators to interact with the elephant robot and use the elephant robot correctly. That is to say, when the user uses the robot, most of the time is achieved by using the raspberry operating system. For example, since the raspberry operating system runs in the teach pendant, the user can use the carrier of the teach pendant to perform manual robotics, programming, and other operations. The operating system OS can also be used to communicate with other robots or devices. All in all, with the advantages of friendly interface and rich functions, the appearance of the raspberry operating system makes it easier for users to start using the elephant robot. It makes everyone a commander of robots.

5.1.2 Main interface introduction

5.1.2.1 User login interface

When the controller is powered up and the system startup button on the teach pendant is pressed, the login page is entered. Figure 2-1 shows the login page of the RoboFlow operating system.

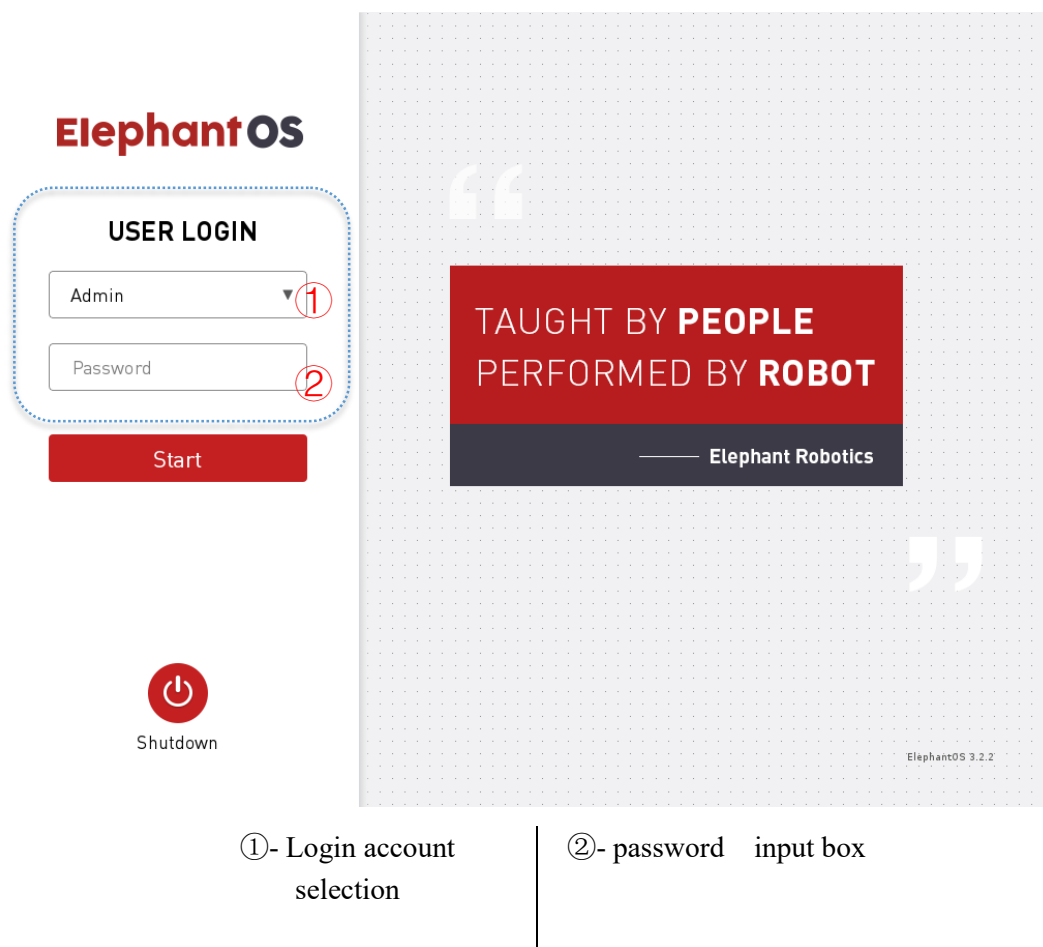


Figure 2- 1 login interface

As shown in the login page, "TAUGHT BY PEOPLE, PERFORMED BY ROBOT", this is the concept that the elephant robotics has always insisted on making the operator become the commander of the robot. Let robots replace people with simple but repetitive tasks, work in harsh working conditions, and work that people can't do well (such as scenes with very high operational accuracy).

There are two types of login users for the RoboFlow operating system, one is the administrator and the other is the operator. The administrator has the highest

authority to perform all operations, programming and setup. The operator can only load and run existing programs and check the statistical data information.

Administrators can add and modify multiple accounts in the settings, including operator accounts.

By clicking on the "Shutdown" button, the RoboFlow operating system can be turned off, and then the power supply can be turned off, thus the robot system can be shut down.

5.1.2.2 MAIN MENU

When the login is successful, it will go to the main menu page. The main menu of the RoboFlow operating system is shown in Figure 2-2.

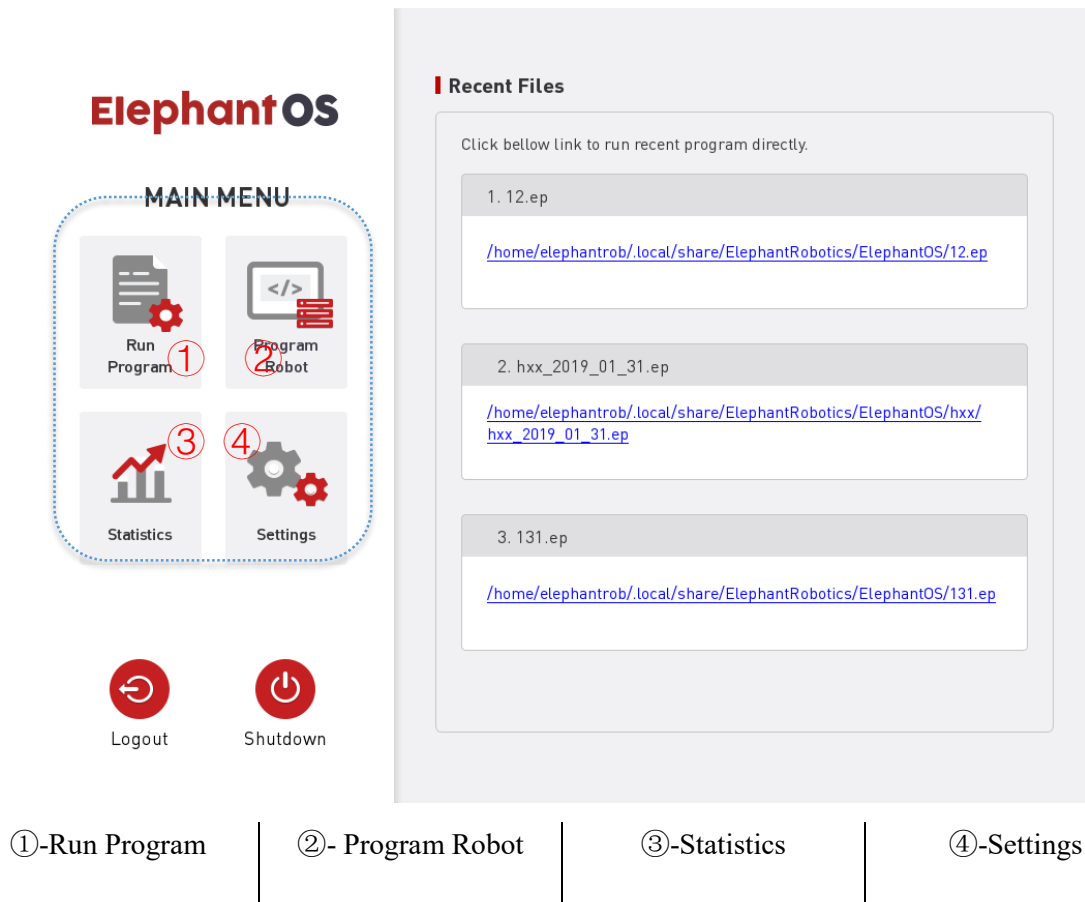


Figure 2- 2 Main menu

On the left side of the main menu, there are four different options available:


1, Run Program

Load an existing program directly and control the program to run. In this window, the user is not allowed to edit the program, but can only control the program running (such as control program running, pausing, stopping). At the same time, you can view the log and other related information during the running process of the program.

2, Edit Program

Users can choose to load an existing program in this window for modification, or they can choose to create a new blank program for editing.

This window is the most frequently used function window for users. Besides programming, it can also perform other operations, such as manual manipulation of robots with "fast moving" function, forced control of IO signals, new variables, etc.

 Attention	If you need to leave this window in the process of writing the program, please pay attention to saving the program.
	It is suggested that the program be debugged in the "trial run" mode before the first run, so as to avoid collision caused by point or path inconsistencies with expectations!
	Non-operators should not use this window to avoid modifying important parameters and causing unexpected behavior during operation.
	It is recommended to use low speed during commissioning, which can effectively reduce the negative impact in the event of an emergency.

3, Statistics

In this window, users can not only view the existing running data of the system, but also view related information saved before.

4, Settings

In this window, the user can make basic settings for the robot. Such as robot open, robot off, account management, default program settings, etc.

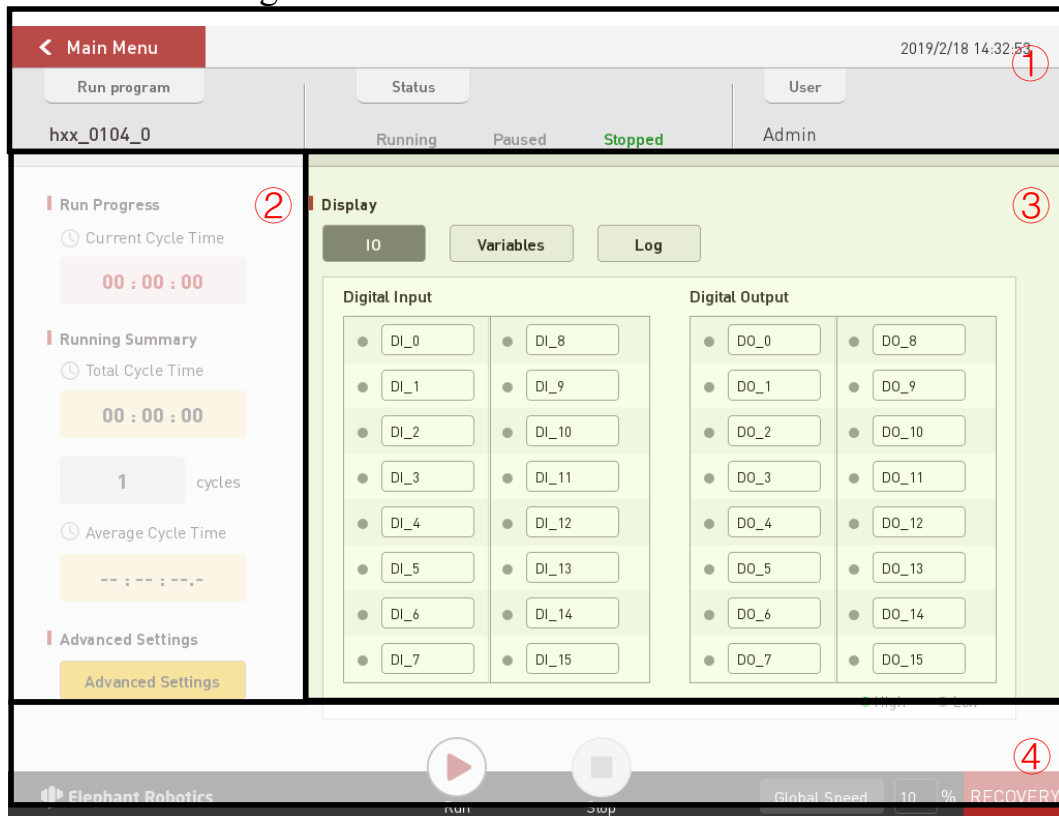
In addition to these four main options, in the right window of the main menu, the user can see and open the most recently run program files. It is convenient for users to quickly find the most recently run program and control the program to run.

Click the "Shutdown" button to close the RoboFlow operating system; click

the "Logout" button to log out.

5.1.2.3 Run Program

If the user selects "Run Program" in the main menu, it will enter the Run Program window. The running program window of the RoboFlow operating system is shown in Figure 2-3.



- | | | | |
|--------------------------------------|---------------------------|-------------------|----------------------------|
| ①- Running program basic information | ②- Operational statistics | ③- Display window | ④- Program run control bar |
|--------------------------------------|---------------------------|-------------------|----------------------------|

Figure 2- 3 Program editing window option

Users can enter the program window by loading the program they need to run. In this window, users can:

- 1) Get the basic information of the current (ready) running program, including program name, running status, user type.
- 2) Understand the statistical information of the current running program, such as the total number of runs and the rhythm, etc.
- 3) Read the relevant information of the current running program through the display window, such as IO, variables, logs, etc.

- 4) The most important thing is that the running program window is the channel for the user to load and run the program that has been debugged.

5.1.2.4 Edit Program

As shown in Figure 2-4, if the user selects "Write Program" in the main menu, two options will appear in the right window. The first is to create a program (optional blank or template) and the second is to load the program.

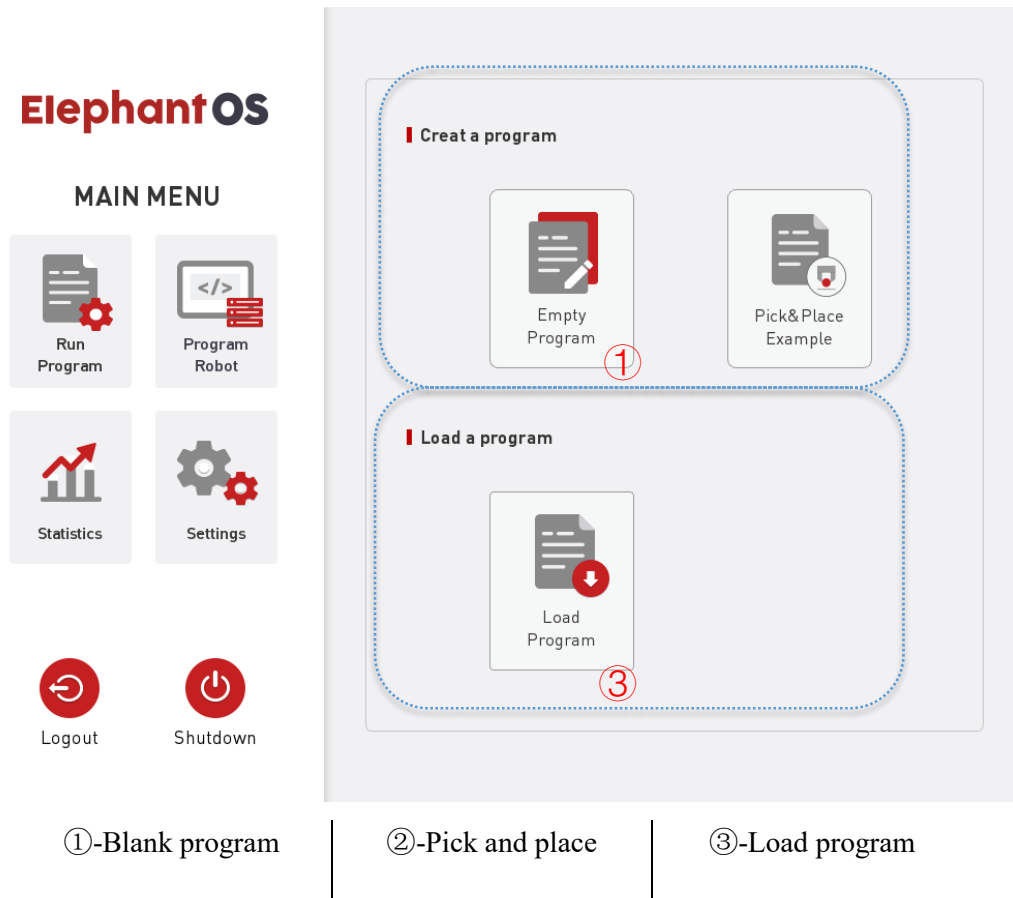


Figure 2- 4 Program Editing Window Options

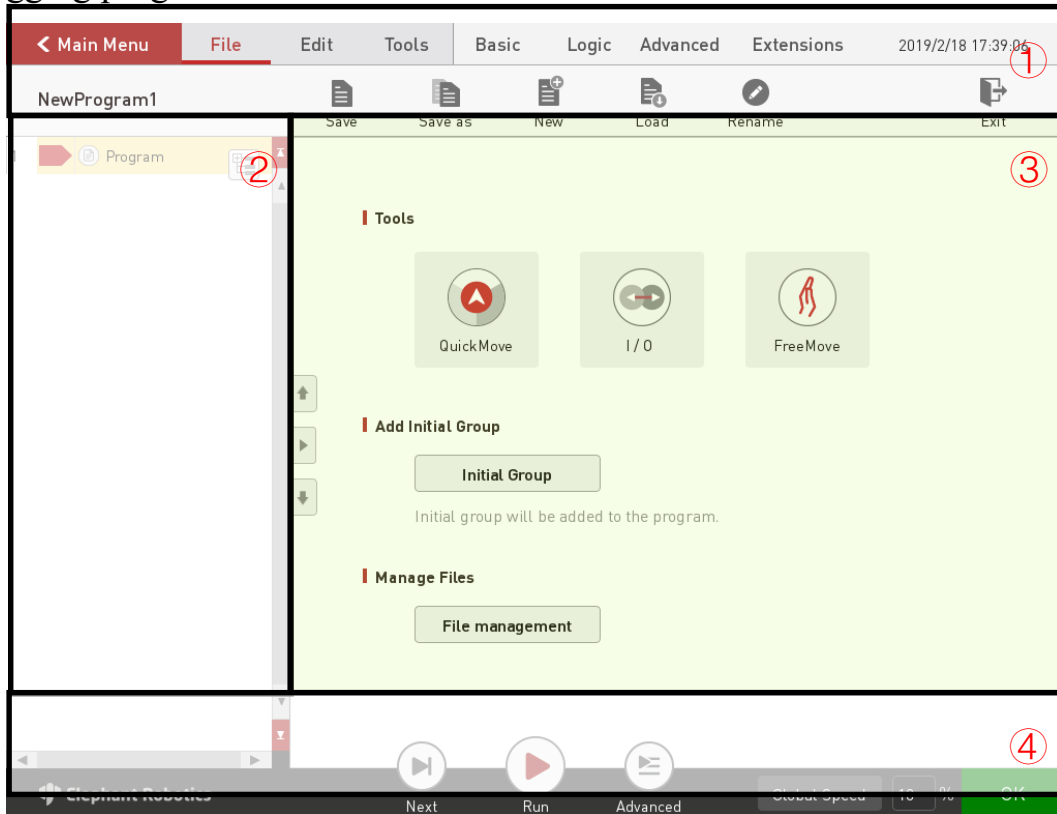
The user selects one as needed and then enters the program editing page as shown in Figure 2-5.

When first entering the program edit page, the user sees the initial page as shown in figure 2-5. In this page, common tools, initialization group and file management functions are provided. The role of the initialization group is to make it easy for the user to set the program content to run at the beginning of the program and run only once. For example, set the initial point, Io State, and so on before the robot starts formal work. File management provides users with a way

to manage files. Users can manage program files here, and can copy them to the U disk, or from the U disk to the system memory. If the user wants to go back to the initial page during the programming process, click "Back".

After entering the program editing page, users can save, create and save files. Users can also edit programs in this interface.

At the same time, users are allowed to teach points, create new variables, view IO, view logs, set important parameters, and add functional instructions, debugging programs and so on.



- | | | | |
|-----------------|---------------------------|------------------------------|--------------------------------|
| ①- Function bar | ②- Program Display Window | ③- Functional Editing Window | ④- Program Running Control Bar |
|-----------------|---------------------------|------------------------------|--------------------------------|

Figure 2- 5 Program Editing Interface

The program editing page is divided into four parts:

1, Function bar

As shown in Figure 2-6, the function bar has seven sub-options, which are divided into two categories, one is the program editing toolbar, and the other is

the function editing column.

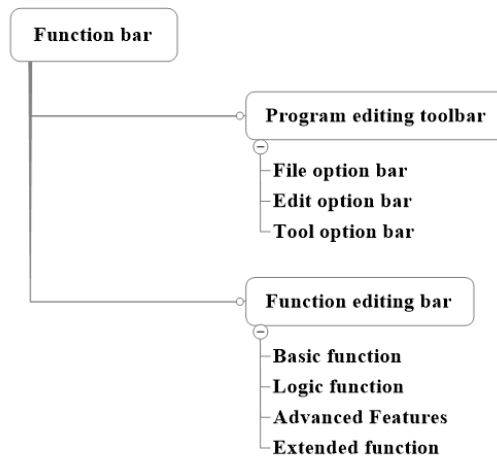


Figure 2- 6 Function bar

1) Program editing toolbar: Includes file option bar, edit option bar, and tool options bar.

A. File: As shown in Figure 2-7, you can edit the program file. There are several operation options: Save, Save As, New, Load, Rename, and Exit.

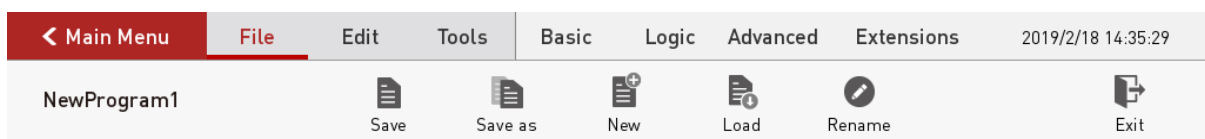


Figure 2- 7 File option bar

B. Edit: As shown in Figure 2-8, you can edit the specific command content in the program file. There are cut, copy, paste, delete, disable, delete all, redo, undo options.

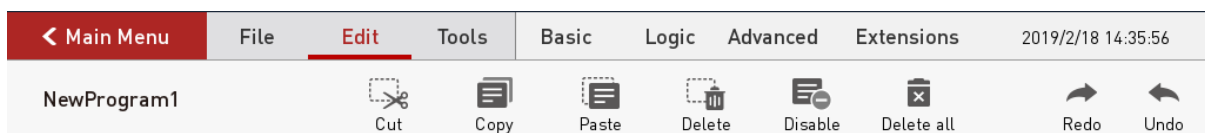


Figure 2- 8 Edit option bar

C. Tool options bar: As shown in Figure 2-9, it is a shortcut toolbar.

When editing a robot program, the user often uses other tools to operate the robot. The tool options bar provides tools commonly used in program editing. Tools provided include: Quickmove, install, input and output, variables, logs, basic settings. For example, when editing a motion command, the user needs to manually operate the robot to a working position and teach the point. Then, the “Quickmove” tool in the toolbar can be selected to manually operate the robot to move to the position.

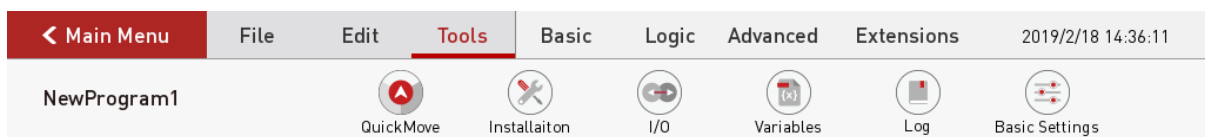


Figure 2- 9 Tool options bar

2) Functional Editing Window

The RoboFlow operating system provides a rich set of features that allow users to perform complex functions with simple operations. Simple, but not simple functions, thus reducing the time workers to learn programming, efficient accomplish their goals. The function editing bar includes basic functions, logic functions, advanced functions, and extended functions.

A. Basic functions : As shown in Figure 2-10, the basic functions include Waypoint, Gripper, Wait, Set, and Group, which are some

basic functions commonly used by users.

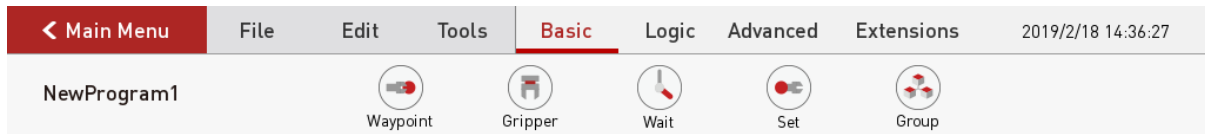


Figure 2- 10 Basic functions

- a) Waypoint: “Create new waypoints → Manually operate the robot to move the robot to the target point → Save current point → Running program”. With this series of operations, the user completes the goal of controlling the movement of the robot to the target point. If you create multiple waypoints, the motion of the robot will form a trajectory when you run the program.
 - b) Gripper: The user can use this function to set the end effector. For example, it holds the workpiece or releases the workpiece.
 - c) Wait: Users can use this function to delay, or wait for signals, conditions, and so on.
 - d) Set: Users can use this function to set the input and output signals and custom conditions.
 - e) Group: Users can use this function to edit the programs in the group.。
- B. Logic function: As shown in Figure 2-11, the logic functions include Loop, If/Else, Subprogram, Thread, Halt, Switch, to complete the program running process control.

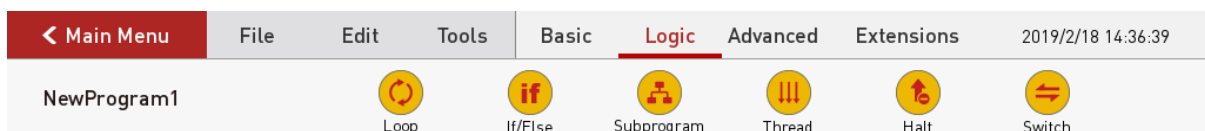


Figure 2- 11 Logic function

- a) Loop: The user can use this function to set a block to run cyclically multiple times.
 - b) If/Else: The user can use this function to make conditional judgments, such as the determination of an input signal.
 - c) Subprogram: The user can use this function to call a subroutine.
 - d) Thread: Users can use this function to achieve robot multi-thread control.
 - e) Halt: The user can use this function to control the program to pause, stop, restart, and pop up the window to display the corresponding prompt information.
 - f) Switch: The user can use this function to make a condition selection and determine the content to be executed according to the value of the selected object.
- C. Advanced function: As shown in Figure 2-12, advanced functions include Pallet, Assign to Var, Script, Popup, and Sender, all of which perform more complex operations.

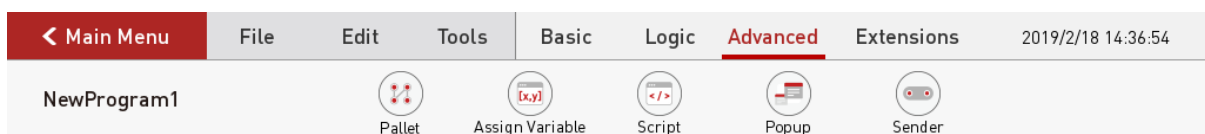


Figure 2- 12 Advanced function

- a) Pallet: Users can use this function to realize the robot to perform

regular point movements. For example, the handling of workpieces in pallets, palletizing, etc. It is also possible to implement the fixed but irregular rendezvous motion of the robot in sequence.

- b) Assign to Var: Users can use this function to implement the assignment of a variable.
 - c) Script: With the scripting feature, users can use the other common functions to achieve simple tasks while using the elephant robot, and can also use script programming to complete more complex tasks.
 - d) Popup: Users can use this function to customize the pop-up window to display related information. This helps the operator to analyze the status of the current robot running program.
 - e) Sender: Users can use this function to achieve TCP/IP communication between the elephant robot and other devices.
- D. Extended function: To adapt to different application scenarios, the RoboFlow operating system provides some extension functions, and even customizes functions according to important application scenarios proposed by users.



Figure 2- 13 Extended function



Attention

These extensions do not exist in every version, but are chosen based on user needs.

2, Program Display Window

On the left side of the program editing page, there is a program display window as shown in Figure 2-14. The upper part is the name of the currently open program file, and the lower part is the program tree, which records the specific instructions and related information.

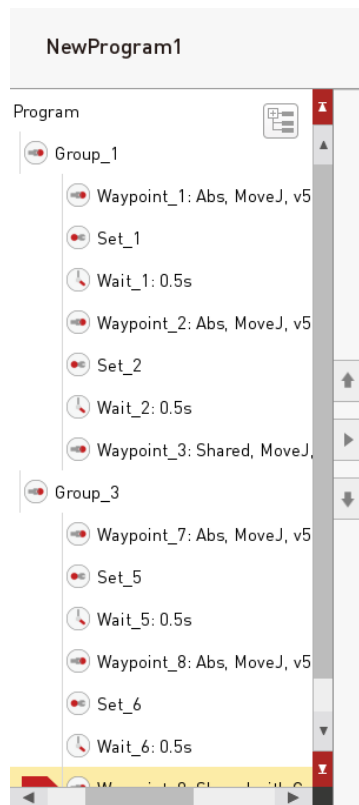


Figure 2- 14 Program Display Window

3, On the right side of the program editing page, there is a function editing window as shown in Figure 2-15, which shows the specific contents of the function instructions.

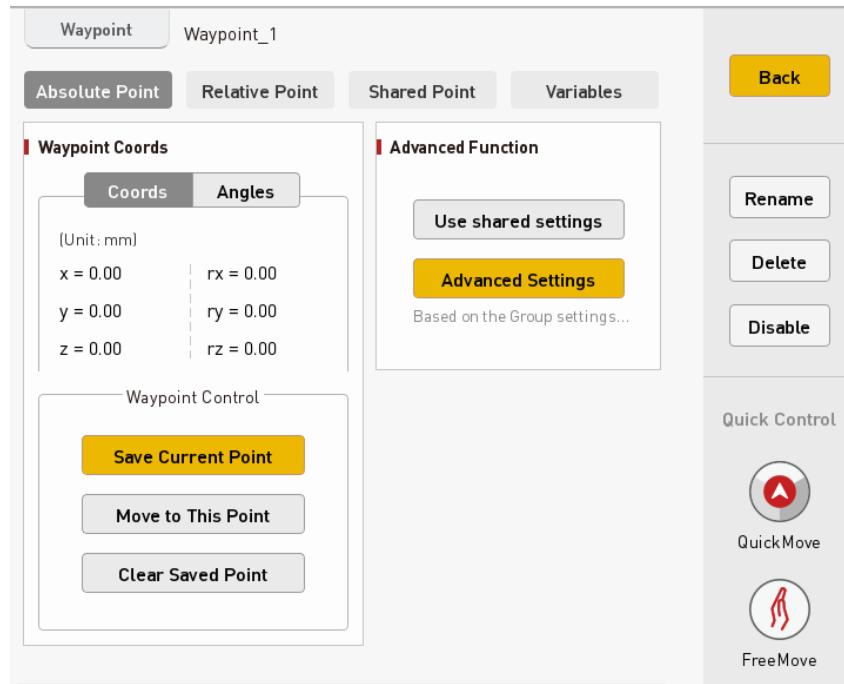


Figure 2- 15 Functional Editing Window

The user can make specific settings for the function instructions in this window. Quick control and current command renaming, deletion, and disabling are also provided here.

4, At the bottom of the program editing page, there is a program running control bar as shown in Figure 2-16. When debugging a program, users can use it to run, pause, stop and limit the running speed of the program.

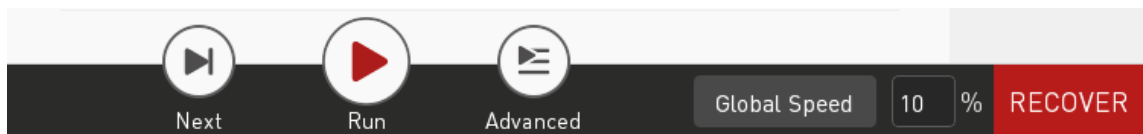


Figure 2- 16 Program run control bar

5.1.2.5 Statistic

When users use elephant robots, they can not only program and control the

robot to complete the corresponding tasks, but also get some valuable statistical data in the statistical report window for analysis and statistics.

The statistical report window is divided into four sub-windows.

As shown in Figure 2-17, the general class counts the total running time, the number of active programs, and the specific information of active programs.

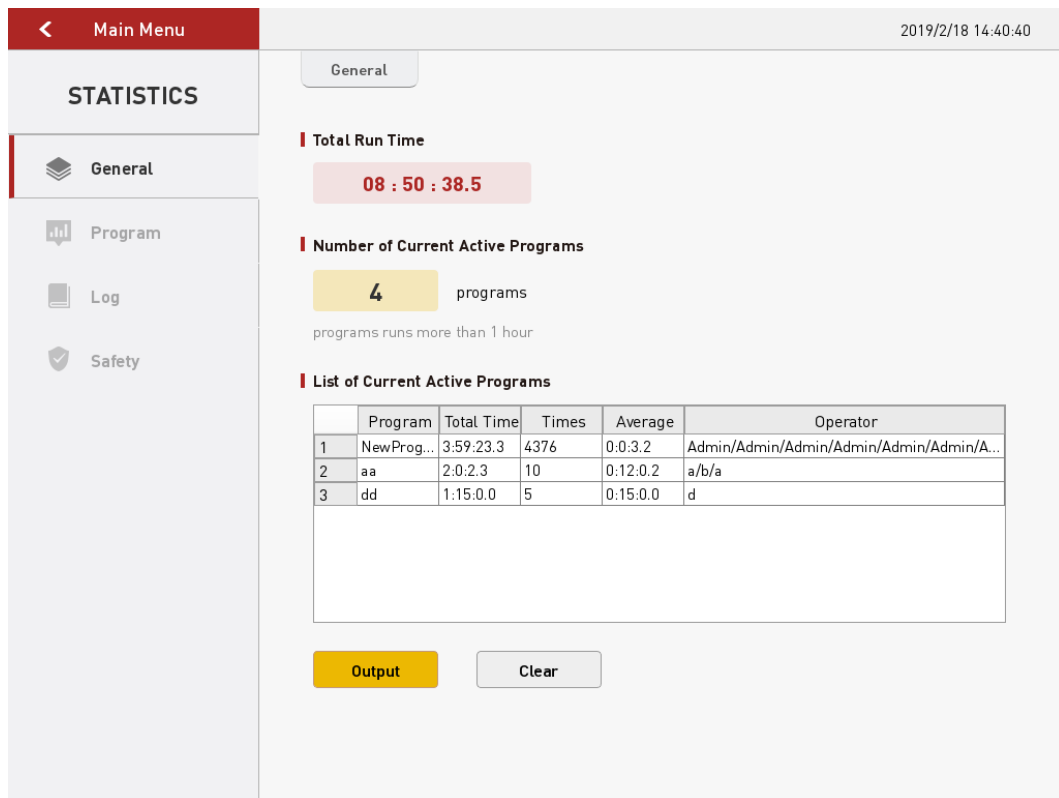


Figure 2- 17 Conventional statistics

As shown in Figure 2-18, the program class counts the total running time and times of different programs.

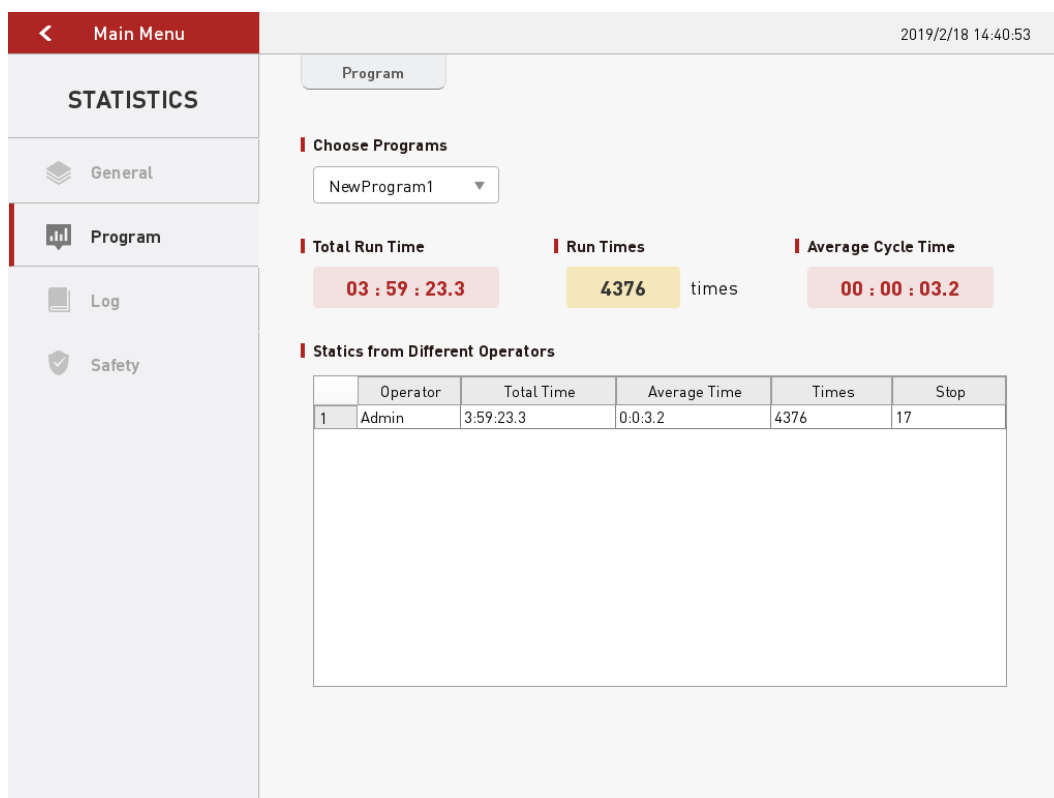


Figure 2- 18 Procedural statistics

As shown in Figure 2-19, the log lists the general information, warning information and error information recorded by the system during the user's use of RoboFlow operating system. This information helps users to determine what changes and feedback the system has made during the operation of the RoboFlow operating system.

In particular, error information can help users quickly locate the possible causes of errors, so as to solve problems according to error information and resume normal use.

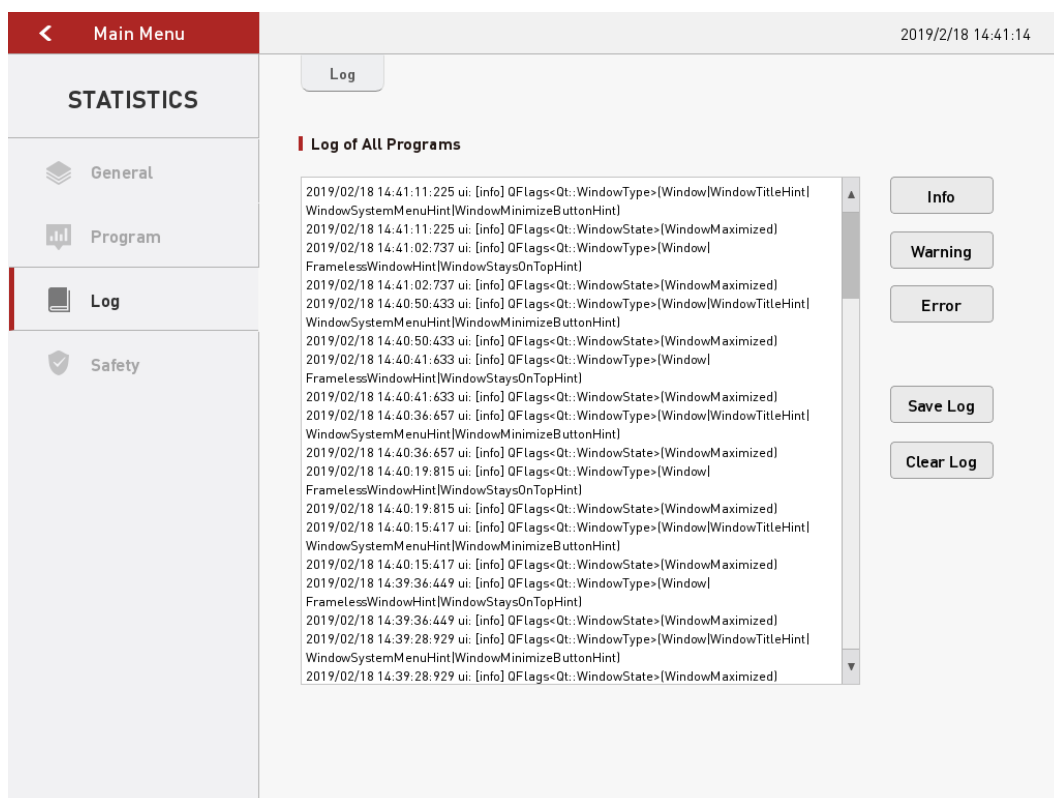


Figure 2- 19 Log statistics

As shown in Figure 2-20, security statistics can help users to count security-related information, such as collision information, number of stops, etc.

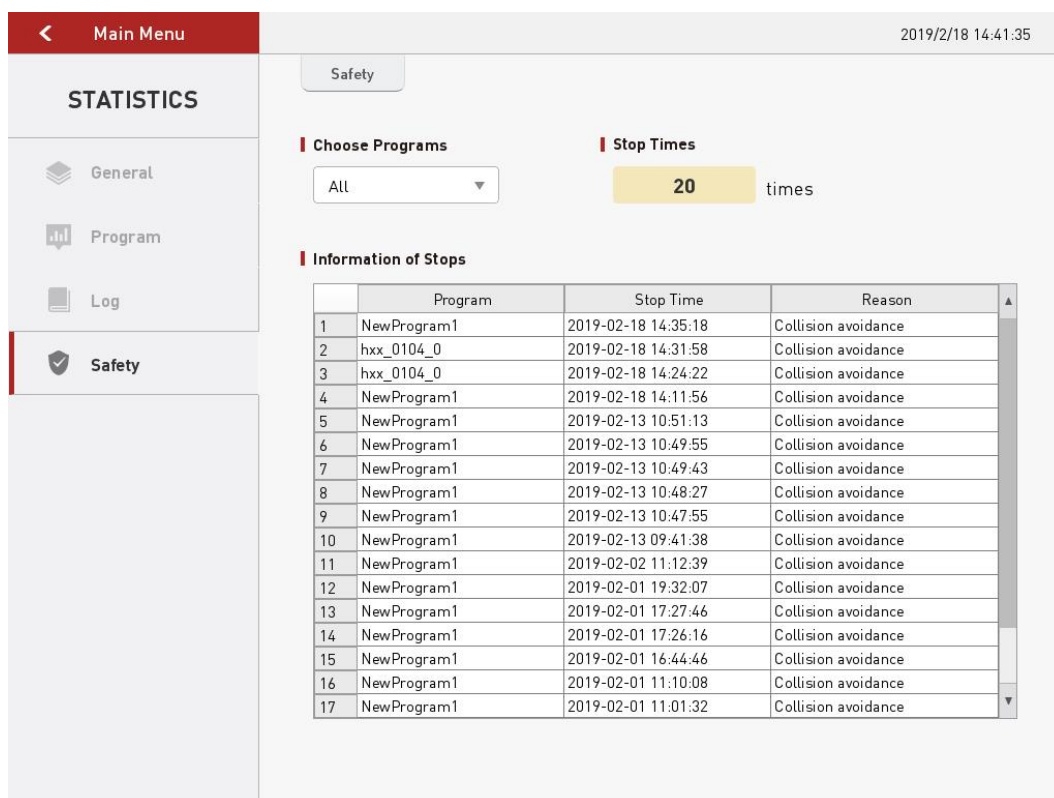


Figure 2- 20 Security statistics

5.1.2.6 Setting

In the configuration center, users can configure the robot. For example, power the robot, turn off the robot, set the load, time, network and so on.

1, Initialization

The initialization configuration page is shown in Figure 2-21.

When robot movement is required, the user needs to enter the configuration center → initialize the robot, or shut down the robot. In the initialization page, you can also set the load and installation, these two are important configuration content before other operations, such as configuration errors may cause unexpected situations.

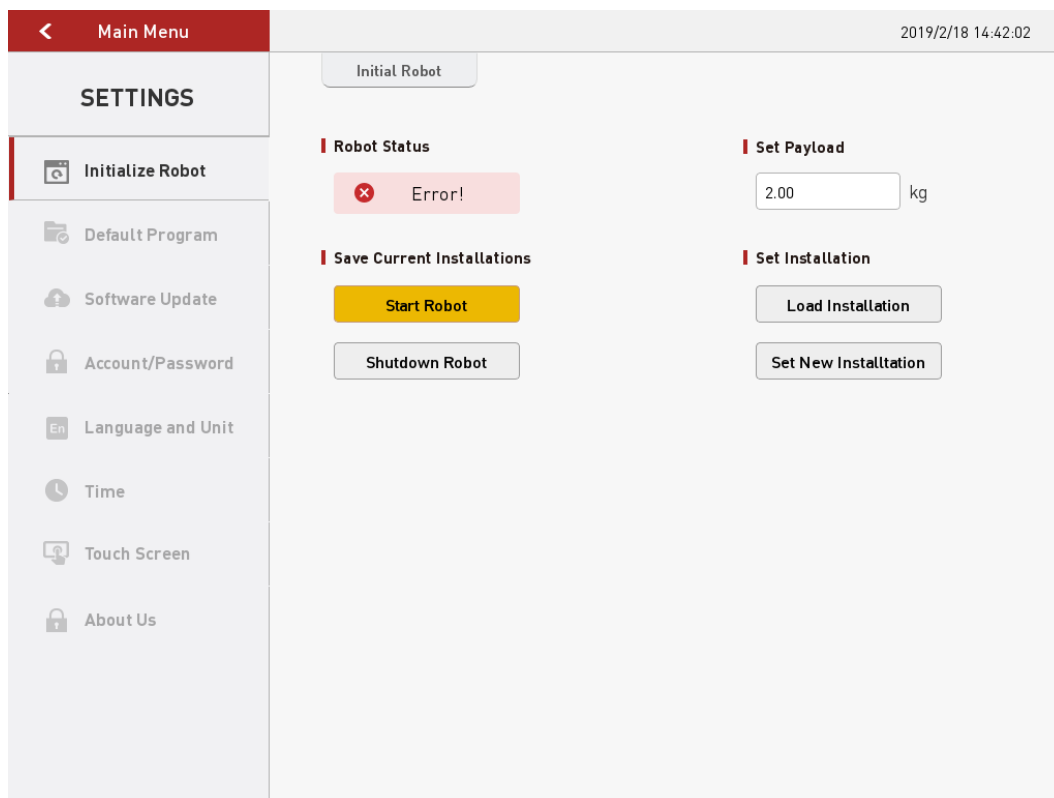


Figure 2- 21 Initialization

2, Default program

Figure 2-22 shows the default program settings page.

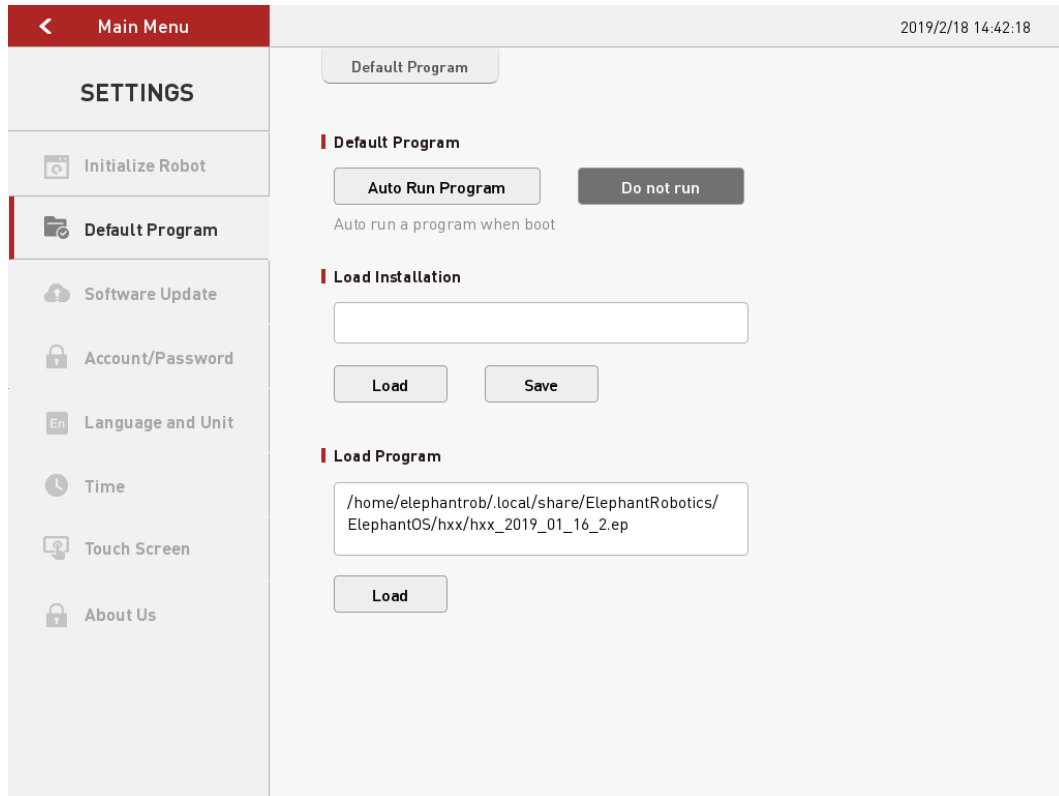


Figure 2- 22 Default program

This function allows the user to set a default running program. As long as the system starts, the robot directly enters the running program window, and can start running the program and perform corresponding actions to complete the specified task.

If the user does not want the system to start and the startup program starts running, you can choose not to run.



Attention

This function is mainly for the program that the user has debugged according to the project content, and directly enters the running program interface after each boot, simplifying the operation steps and facilitating the user. If the program has not been debugged, please choose not to run.

3, Version update

Figure 2-23 shows the version update settings page.

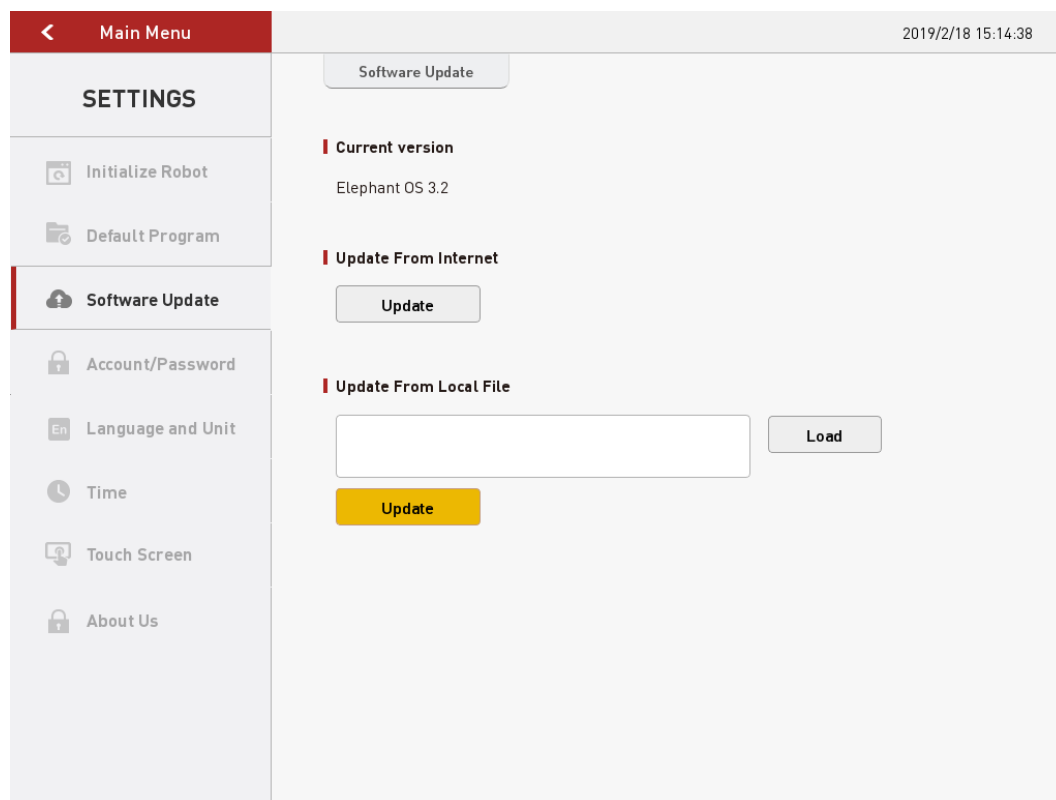


Figure 2- 23 Version update

This page allows users to update the RoboFlow operating system in two ways, one for local file updates and one for network updates.

4, Account management

Figure 2-24 shows the account management page.

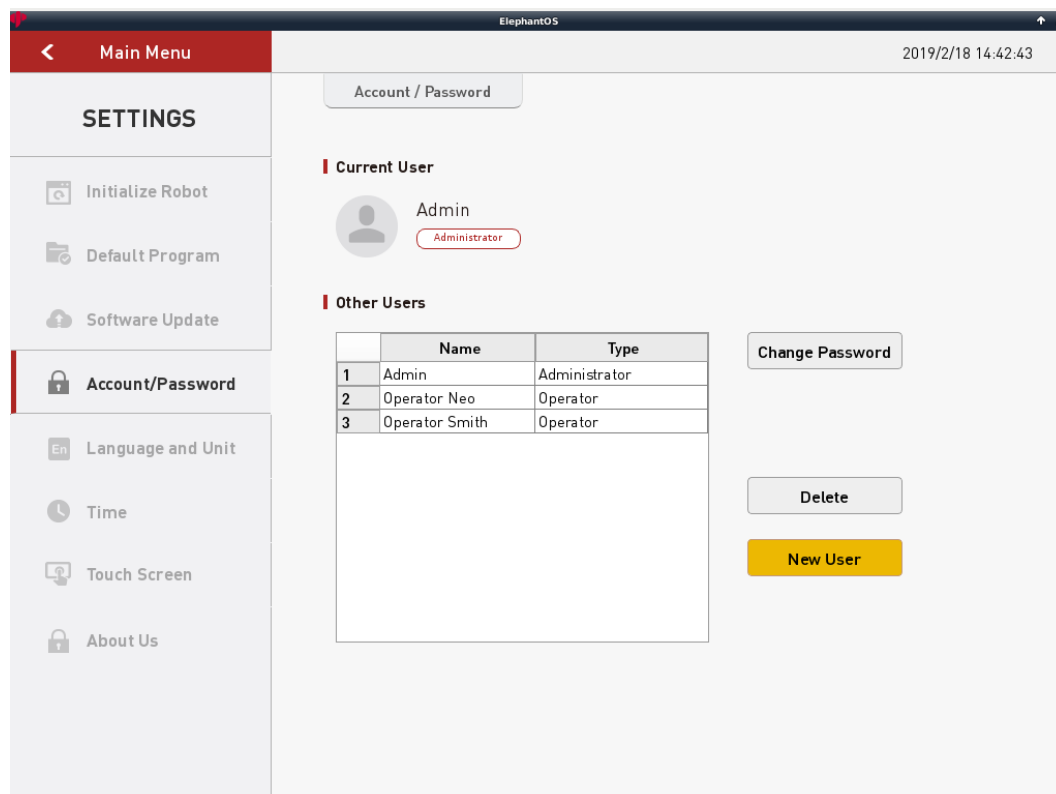


Figure 2- 24 Account management

Users can add new users, delete expired users, or change passwords on this page. On this page, the user can get all the account information.

5, Language and unit

The language and unit settings page are shown in Figure 2-25. At present, the RoboFlow operating system supports Chinese and English and metric units. Other languages and units are increasing, so stay tuned!

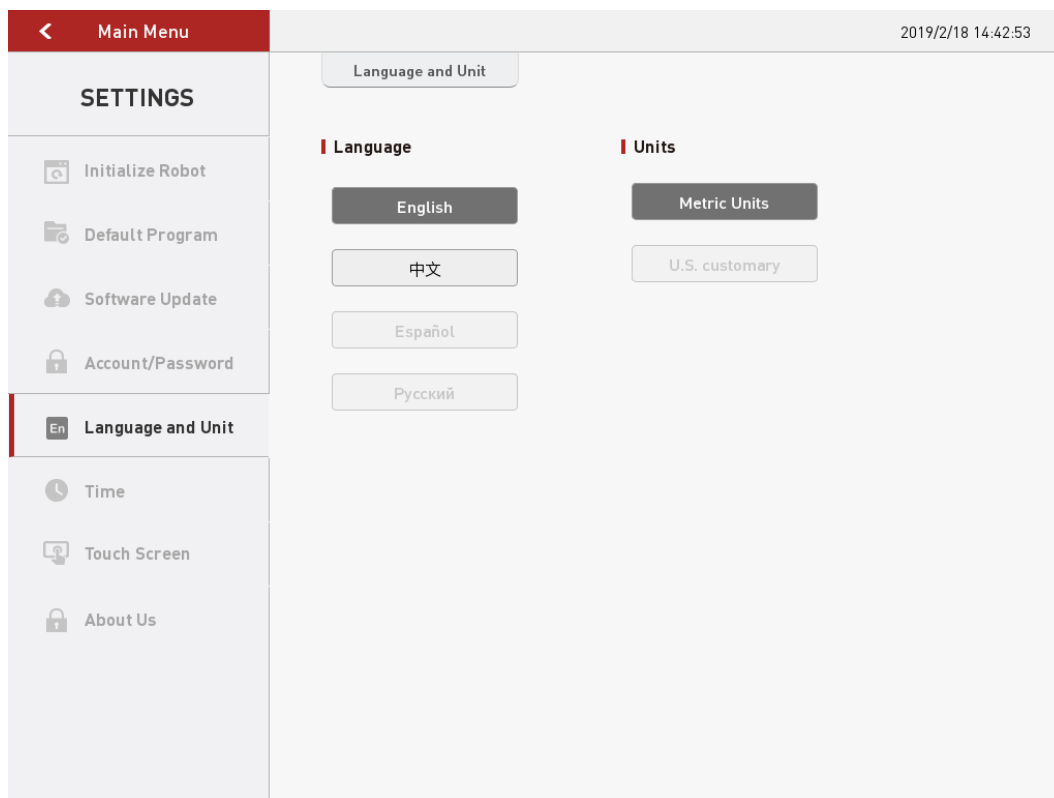


Figure 2- 25 Language and unit

6, Time

Figure 2-26 shows the time setting page.

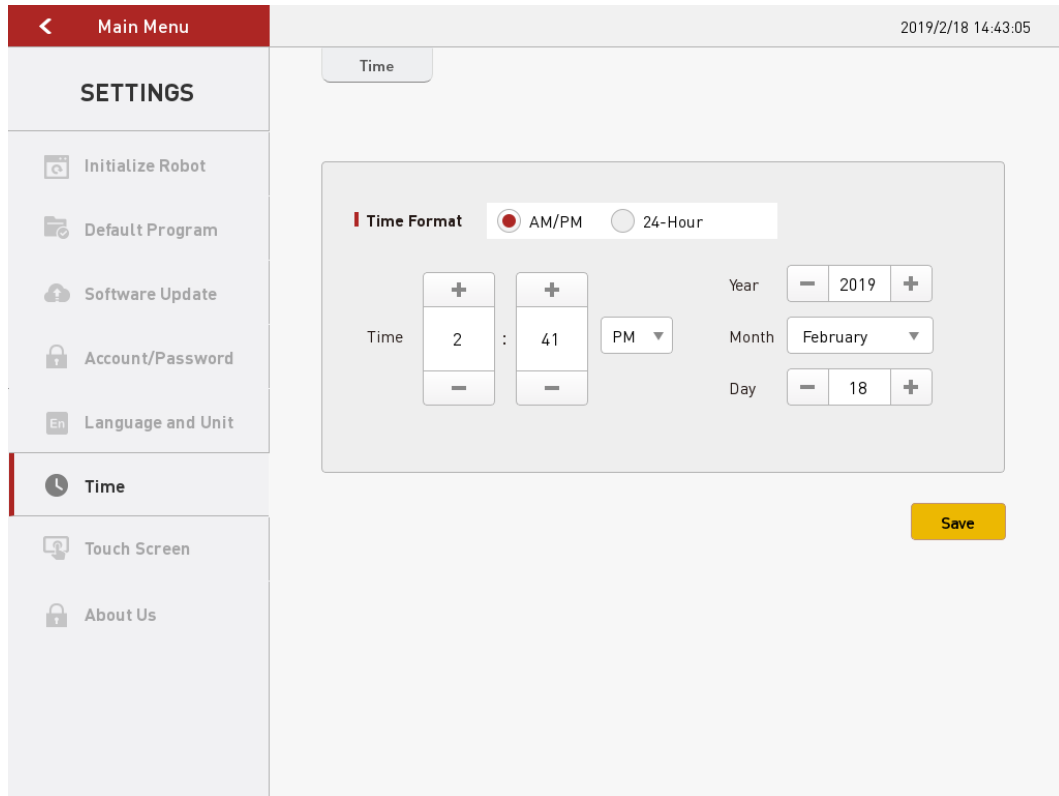


Figure 2- 26 Time

The user can set the system time on the current page. If the "24-hour system" is not checked, the time display format defaults to 12-hour system.

7, Touch screen calibration

Figure 2-27 shows the touch screen calibration instructions. The user clicks on "Start to Calibrate" to enter the calibration interface. The calibration interface will appear in sequence with four circles, as shown in the figure. The user needs to click the center of the circle with a touch pen, and each time the button is clicked, the next circle will appear until all four circles appear. A pop-up window will appear indicating that the calibration is complete, and you can exit the calibration screen after confirming the pop-up.

If the calibration times out or the steps are wrong, a pop-up prompts the

calibration failure. At this point, you can confirm to exit the calibration interface and return to the page in Figure 2-27 to recalibrate.

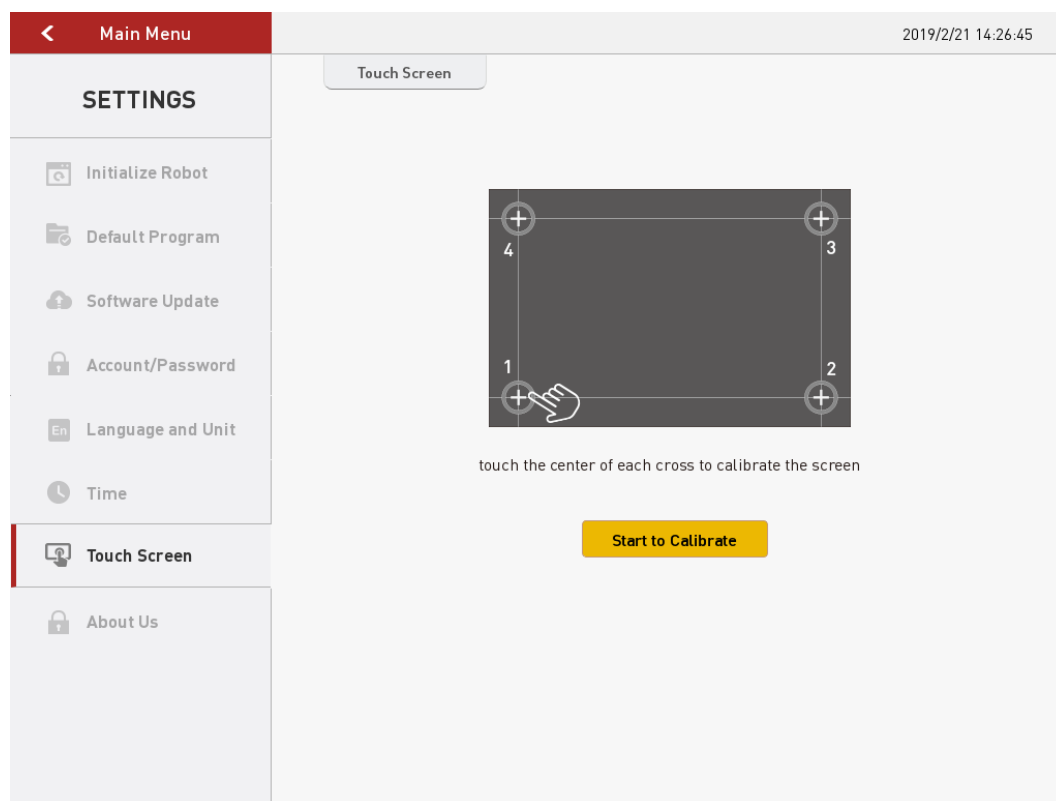


图 2- 27 Touch Screen

8, About us

As shown in Figure 2-28, it is about our page.

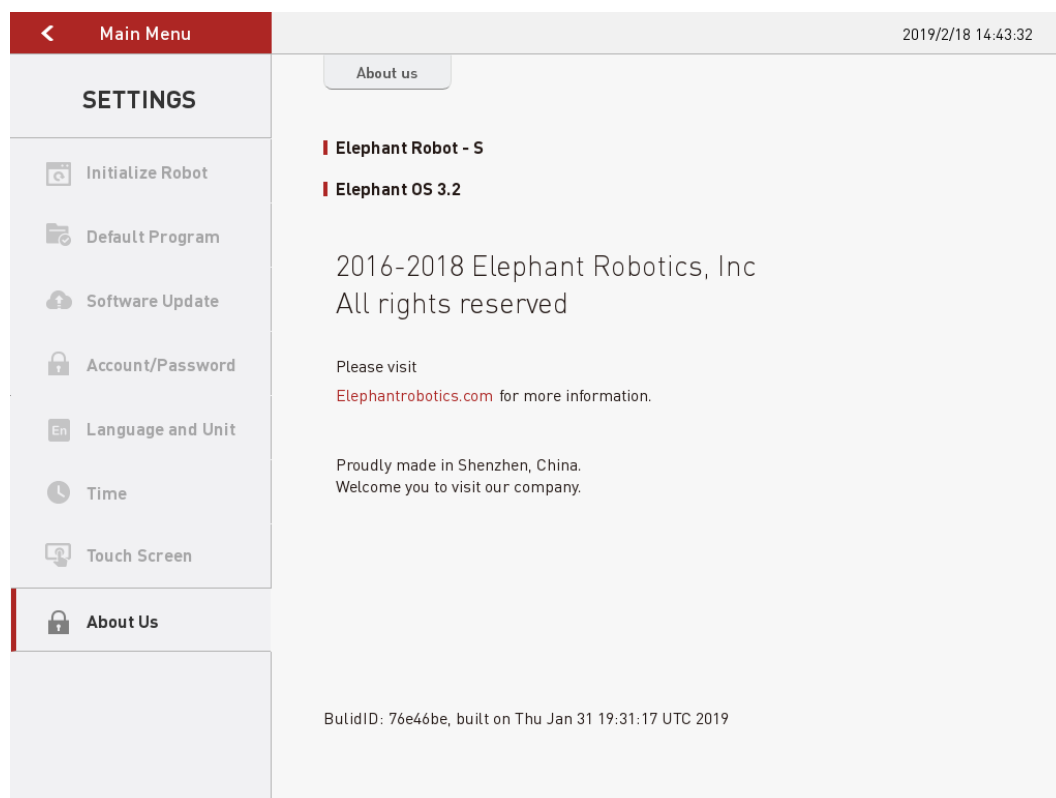


Figure 2- 28 About us

This page shows basic information about the operating system of the RoboFlow operating system. For example, the model of the robot used is the Elephant series, version information, and so on.

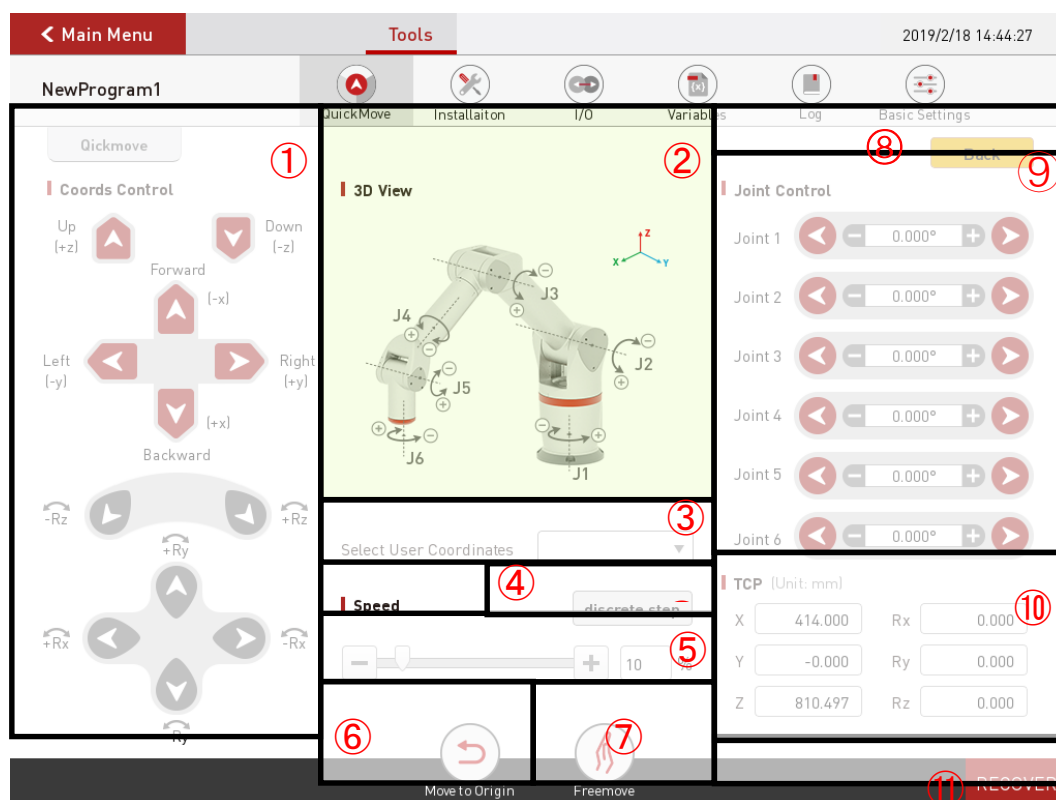
For more information, please visit the official website <https://www.elephantrobotics.cn>.

5.1.3 Introduction to common tools

5.1.3.1 Quickmove

Quickmove is a tool that users use frequently when they operate the robot quickly and manually. Therefore, every user must be very familiar with the use of Quickmove using methods. The wrong operation may result in damage to the robot and its peripheral equipment, and even injuries to personnel.

As shown in Figure 3-1, Quickmove are mainly composed of 11 parts, which are described below.



①- Coordinate control

②- 3D View

③- User coordinate system

④- Stepping motion	⑤- Speed	⑥- Move to origin
⑦- Freemove	⑧-Return	⑨- Joint control
⑩- Coordinate position	⑪- Status display button	

Figure 3- 1 Quickmove

1, Motion Control Mode in Cartesian Coordinate System

As shown in Figure 3-2, over-fixed-point O, three axes perpendicular to each other, all with O as the origin and generally with the same unit of length. These three axes are called x-axis (horizontal axis), y-axis (vertical axis), and z-axis (vertical axis), which are collectively referred to as coordinate axes. The x-axis and y-axis are usually arranged on a horizontal plane, while the z-axis is a plumb line. Their positive direction is in accordance with the right-hand rule, that is, holding the z-axis with the right hand. When the four fingers of the right hand turn from the positive x-axis to the positive y-axis from the $\pi/2$ angle, the thumb is pointed to the positive direction of the z-axis. Such three axes form a spatial Cartesian coordinate system, and point O is called the coordinate origin. This constitutes a Cartesian coordinate.

There are three planes in the three-dimensional Cartesian coordinate system, XY-plane, YZ-plane, and XZ-plane. These three planes divide the three-dimensional space into eight parts, called octant spaces. The three coordinates of each point of the first limit are positive values.

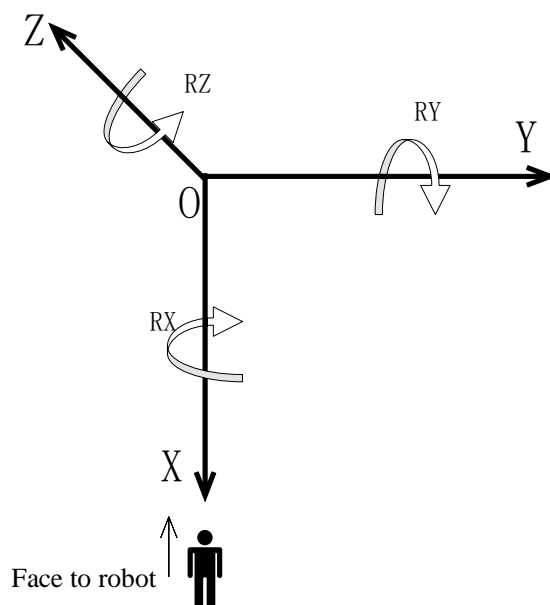


Figure 3- 2 Cartesian coordinate system Direction callout diagram

As shown in Figure 3-3, the robot can be controlled to move in the direction of the Cartesian coordinate system by clicking the key corresponding to the direction of the Cartesian coordinate system.

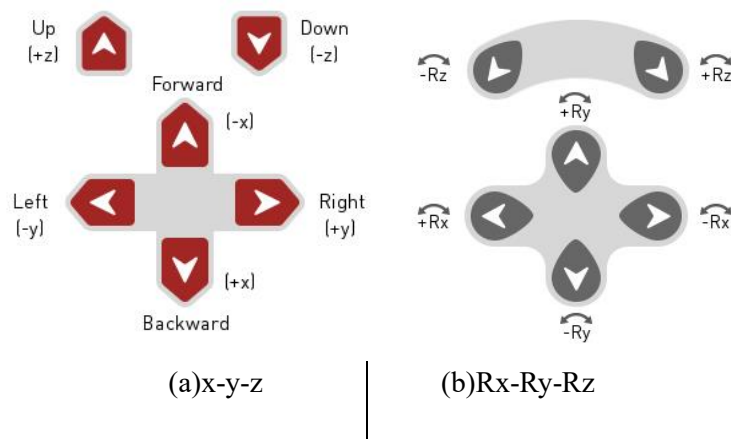


Figure 3- 3 Cartesian coordinate system motion control mode button



Attention

It should be noted that if the coordinate control button is pressed and the robot does not respond, please release the button and press it again.

During the process of pressing the button, the user needs to observe the movement trend of the robot to prevent the robot from hitting other equipment or obstacles and causing damage.

	The time when the coordinate control button is pressed determines the distance the robot moves. Do not press and hold it.
--	---

2, 3D View

This window marks the direction of movement of the six joints of the robot.

3, User coordinate system

User coordinate system can be selected under this window. This function is under debugging. Please look forward to it!

4, Motion mode switching

There are two main motion modes for manual manipulation robots.

- 1) Continuous motion mode: The user presses the motion control button and allows the robot to move until the user releases the button and the robot stops. For example, if you press the + X direction motion control button, you need to hold the button all the time. The time of pressing the motion control key determines the distance of the robot in the + X direction.
- 2) Stepping motion mode: Manual manipulation robot step motion, click "step motion" and open the step setting window as shown in Figure 3-4. Then the user chooses the step in this window and clicks the key of the target control direction. Every time he clicks, the robot takes a step. For example, choose a 1 mm step, click the X-direction movement control button, and every time you click the button, the robot will move 1 mm in the + X direction.




Figure 3- 4 Step-by-step motion step setting window

5, Speed

As shown in Figure 3-5, the control speed of the manual manipulator can be set here. Speed can be set from 0 to 100%.



Figure 3- 5 Speed setting window

	<p>The speed here refers to the speed of the manual manipulation robot, including continuous motion control speed and stepping motion control speed.</p>
	<p>Please choose the right speed according to the actual needs. If you are not familiar with fast movement control, please try to choose low speed.</p>

6, Move to origin

By selecting the icon shown in Figure 3-6, the robot can be controlled to return to its original position and posture.



Figure 3- 6 Move to origin



Attention

In the process, you need to keep the state of the icon button selected, otherwise the robot will stop in the middle.

7, Freemove

Select the icon shown in Figure 3-7 to switch to the drag mode.



Figure 3- 7 Freemove

8, Return

Click on the icon shown in Figure 3-8 to return to the programming operation window.



Figure 3- 8 Return

9, Joint control

Serial robot is an open kinematic chain of the robot. It is formed by a series of connecting rods connected in series with a rotating joint or a moving joint. The elephant cooperative robot belongs to a 6-axis serial robot. It drives the relative motion of the connecting rod by using motor drivers to drive the movement of 6 joints, allowing the end operator to reach the right posture. The Joint control window shown in Figure 3-9 provides the keys used by the operator to manually

manipulate the robot and control the robot for joint movement using the instructor. The control buttons for each joint are divided into 2 directions, and the angle data of each axis can be seen.



Figure 3- 9 Joint motion mode control window

10, Coordinate position

As shown in Figure 3-10, this window displays the coordinate position corresponding to the coordinate control.

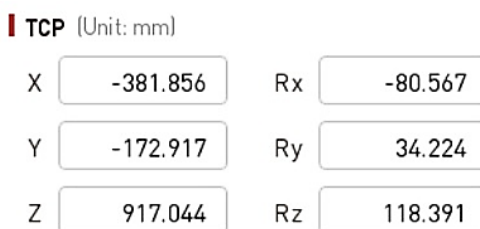


Figure 3- 10 Coordinate position display window

11, Status display button: The button has two states, "OK" (displays green)

and "Reset" (displays red). When the display is normal, it indicates that the robot is working properly, and when the reset is displayed, the robot is abnormal, the anomaly needs to be lifted and the key is clicked for reset.

5.1.3.2 Installation

As shown in Figure 3-11, there are three submenus inside the installation tool. It is used to implement the loading/saving installation configuration, security configuration, and network configuration of the elephant robot.

- 1, Load/save: As shown in Figure 3-11, users can choose to save or load the installation configuration on this page.

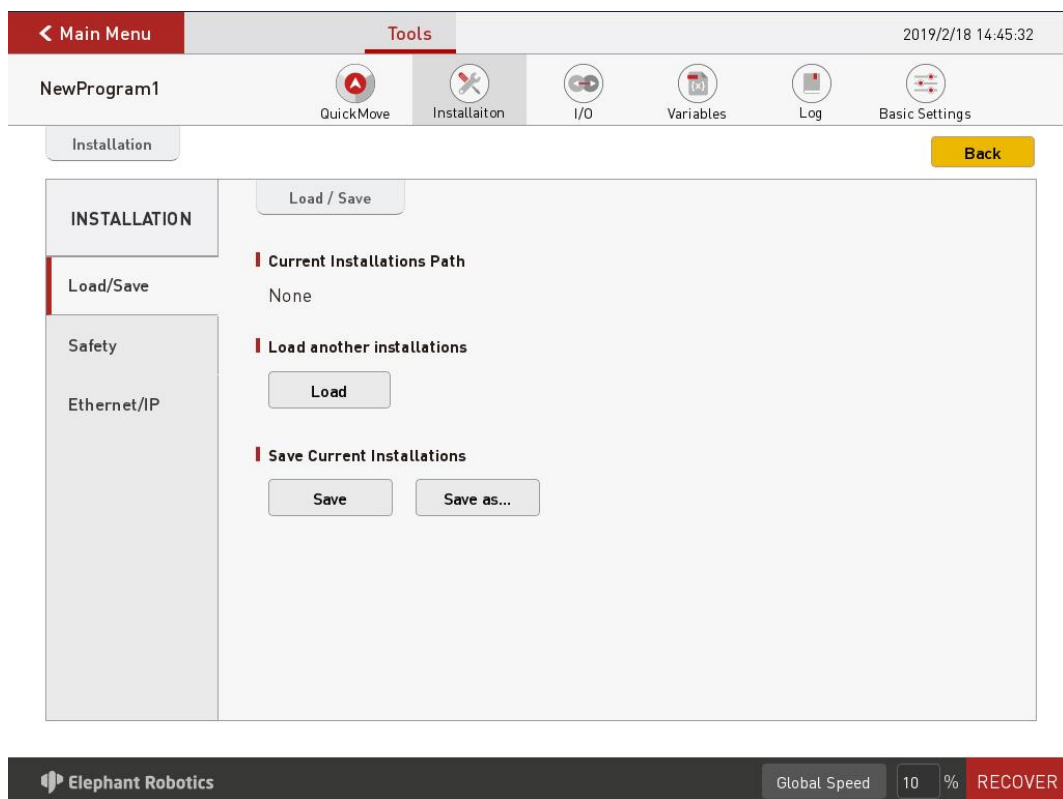


Figure 3- 11 Load/save installation

2, Security configuration: As shown in Figure 3-12, set the torque limit and brake control of the elephant robot.

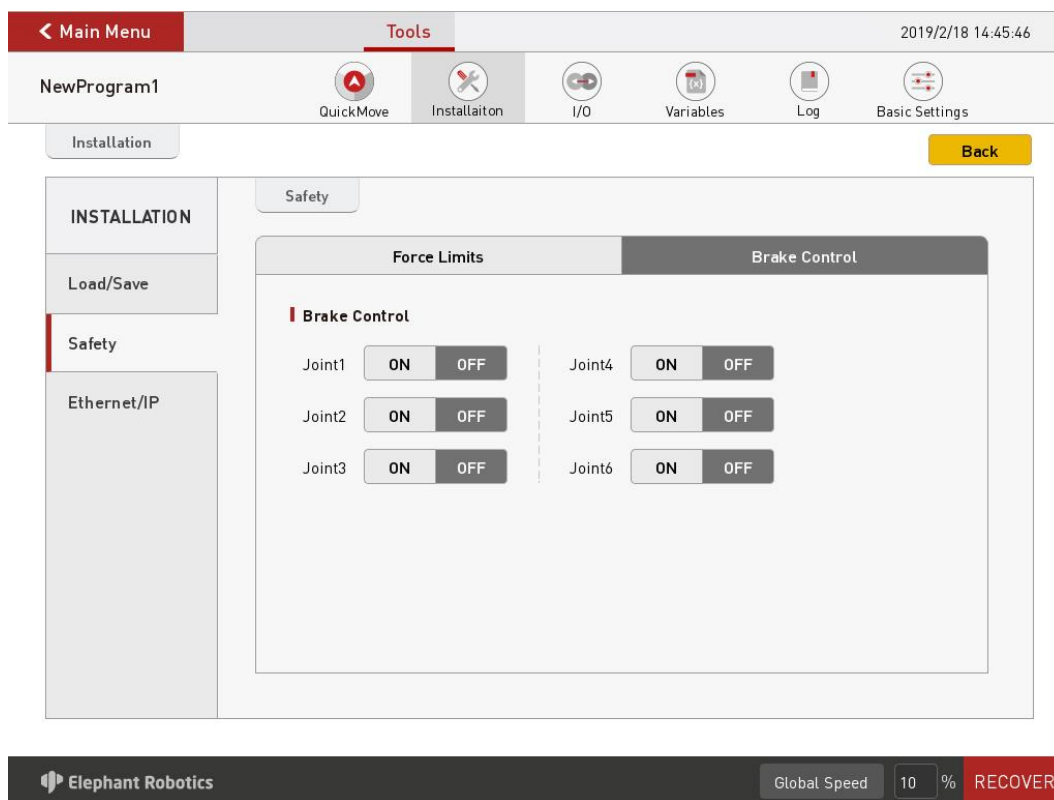


Figure 3- 12 Security configuration

3, Network Configuration: As shown in Figure 3-13, configure the IP address and port number of the Ethernet communication here.

Figure 3- 13 Network settings

5.1.3.3 Input and output configuration

The robot system has a total of 6 digital input signals and 6 digital output signals. As shown in Figure 3-14, the input and output signals can be configured and monitored in this window, and the output signals can be forcibly output. IO configuration files can also be saved and loaded on this page. As shown in Figure 3-15, it is an input/output interface description corresponding to the page shown in Figure 3-14.

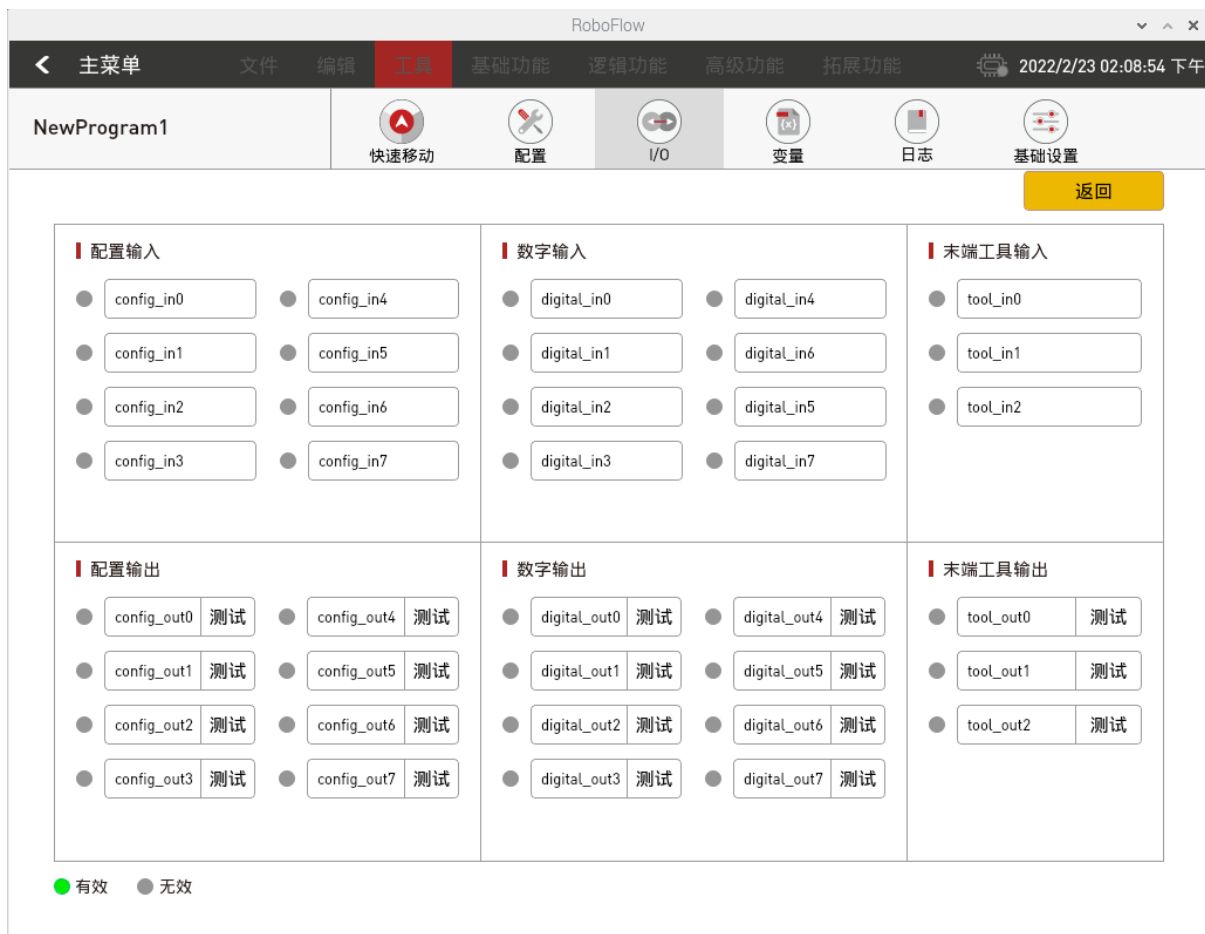


Figure 3- 14 Input and output configuration

Serial No.	Layout Position	Types	Definition	Description
1	Front	DC power input	DC48V	External DC42V power input interface
2	Left side	USB Interface	USB2.0	Used for external extension devices such as mouse, keyboard and USB flash disk
3			USB3.0 (blue)	
4		Ethernet interface	Ethernet	Ethernet interface
5		24V	DC24V	DC24V Output
6		Digital output 1-6	OUT1	PNP Digital output signal 1
7			OUT2	PNP Digital output signal 2
8			OUT3	PNP Digital output signal 3
9			OUT4	PNP Digital output signal 4
10			OUT5	PNP Digital output signal 5
11			OUT6	PNP Digital output signal 6
12		GND	GND	GND
13		Emergency stop	ES1 +	External emergency stop control loop
14			ES1 -	
15	Upside	MircoHDMI1	MircoHDMI1	Display interface, use HDMI cable to connect the display screen
16		MircoHDMI2	MircoHDMI2	
17		Type C	Type C	Used for internal debugging
18		Digital input 1-6	IN1	PNP Digital input signal 1
19			IN2	PNP Digital input signal 2
20			IN3	PNP Digital input signal 3
21			IN4	PNP Digital input signal 4
22			IN5	PNP Digital input signal 5
23			IN6	PNP Digital input signal 6
24		Null interfaces	-	Reserved
25			-	SRS485 interface
26	Right side	Display lamp	-	Display the status of the master controller

Figure 3- 15 Input and output interface description

It should be noted that the input public terminal needs to be connected to 24V power supply. It can be determined whether the input is active high or active low according to the common configuration (hardware connection determines 24V or 0V). As shown in Figure 3-16, when the common terminal is connected to 24V, once an external device inputs 0V, the input signal is in the “High” state, otherwise it is in the “Low” state; vice versa.

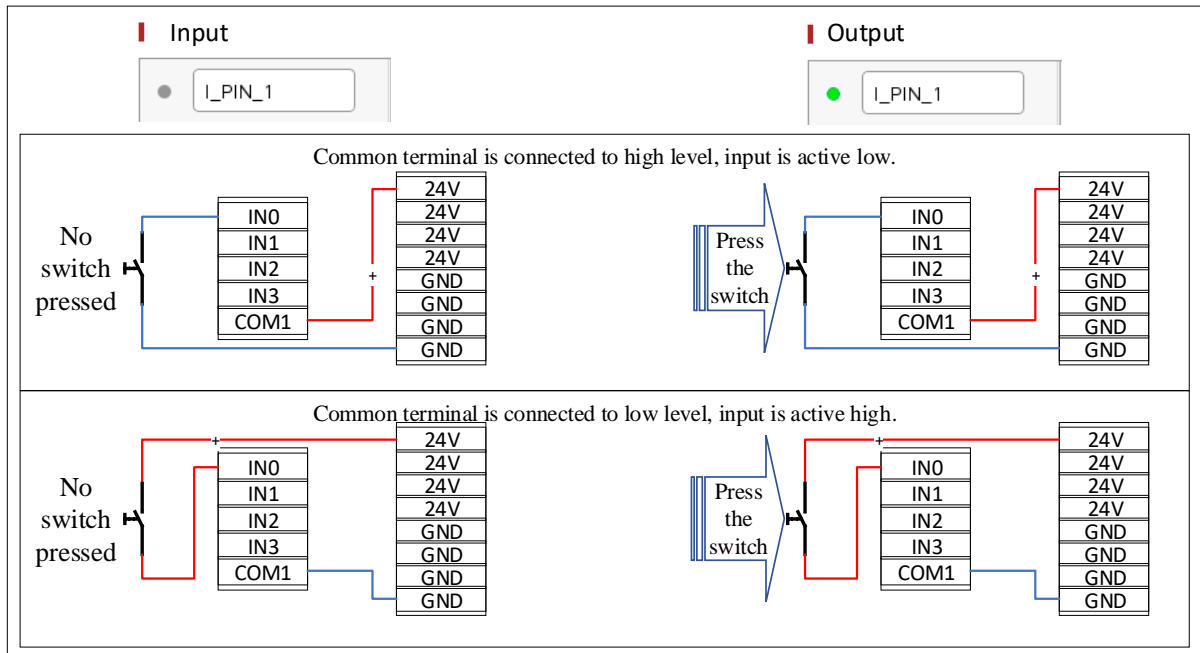


Figure 3- 16 Input signal application diagram

As shown in Figure 3-17, the output is 24V when there is no output. Once the output is turned on (that is, the output is High), the output is 0V.

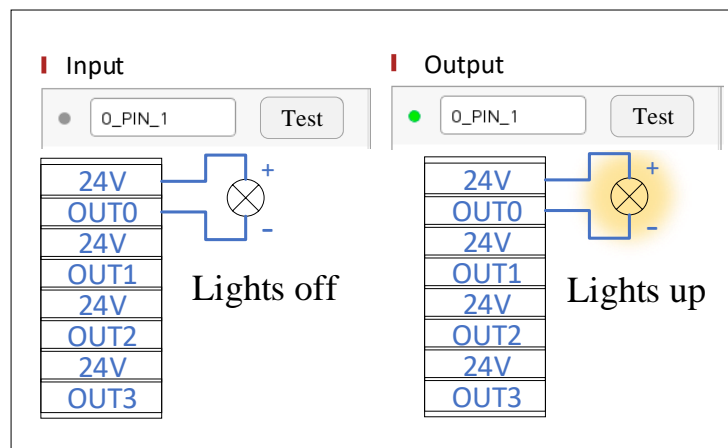


Figure 3- 17 Output signal application diagram

5.1.3.4 Variable

As shown in Figure 3-18, in the variable editing window, you can add, edit,

and delete variables.

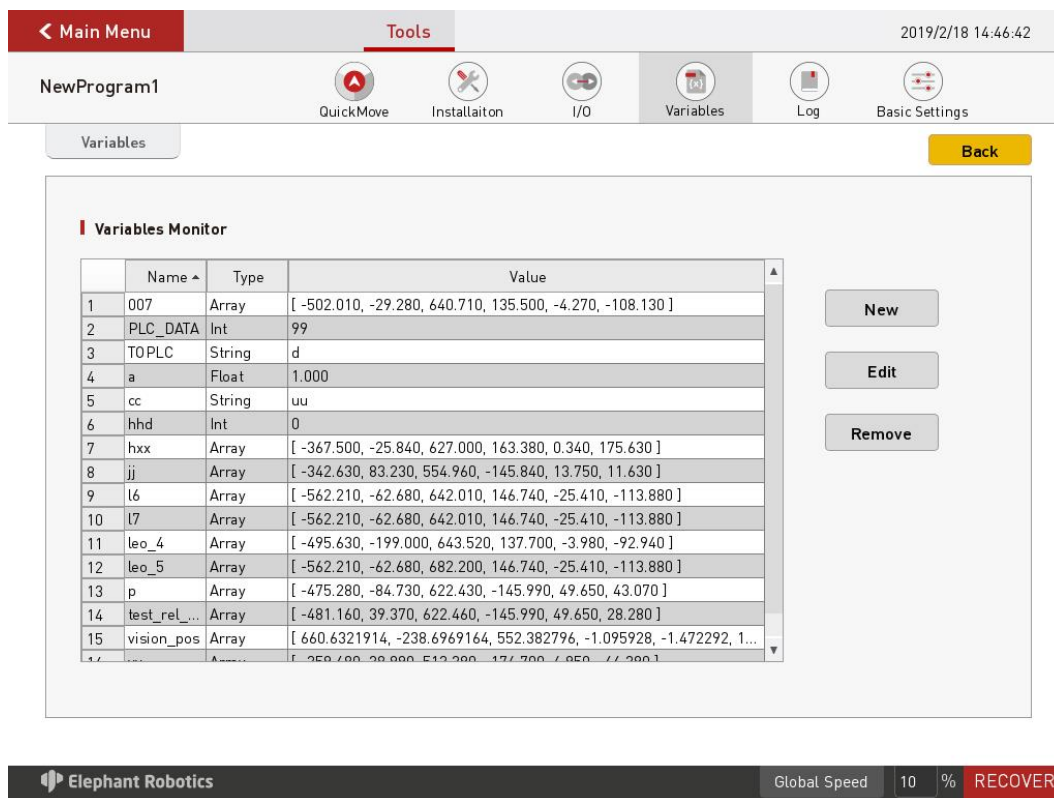
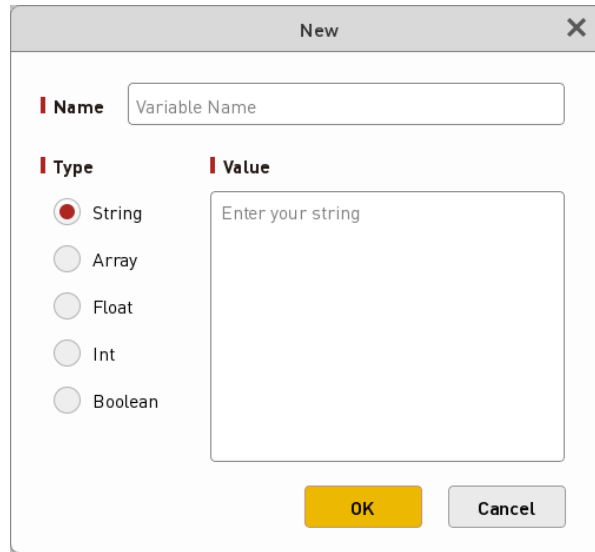


Figure 3- 18 Variable editing

As shown in Figure 3-19, there are 5 types of editable variable types. They are string variables, pose variables, floating point variables, integer variables, and Boolean variables. On this page, you can edit the variable name and initial value.



The image shows a 'New' dialog box for creating a variable. It has a title bar with 'New' and a close button. Inside, there are three main sections: 'Name', 'Type', and 'Value'. The 'Name' section has a text input field with 'Variable Name' as a placeholder. The 'Type' section has five radio button options: 'String' (selected), 'Array', 'Float', 'Int', and 'Boolean'. The 'Value' section has a large text area with the placeholder 'Enter your string'. At the bottom right, there are two buttons: 'OK' (yellow) and 'Cancel' (gray).

Figure 3- 19 New variable interface

5.1.3.5 Log

As shown in Figure 3-20, you can view information about the robot running status, error information, and alarm information in the running log window. Click the "Information", "Warning" and "Error" buttons to sort the corresponding logs.

Users can save logs to a local folder. Log files are a record of how the system is performing, helping users to have a clearer understanding of the system and also helping to troubleshoot errors.

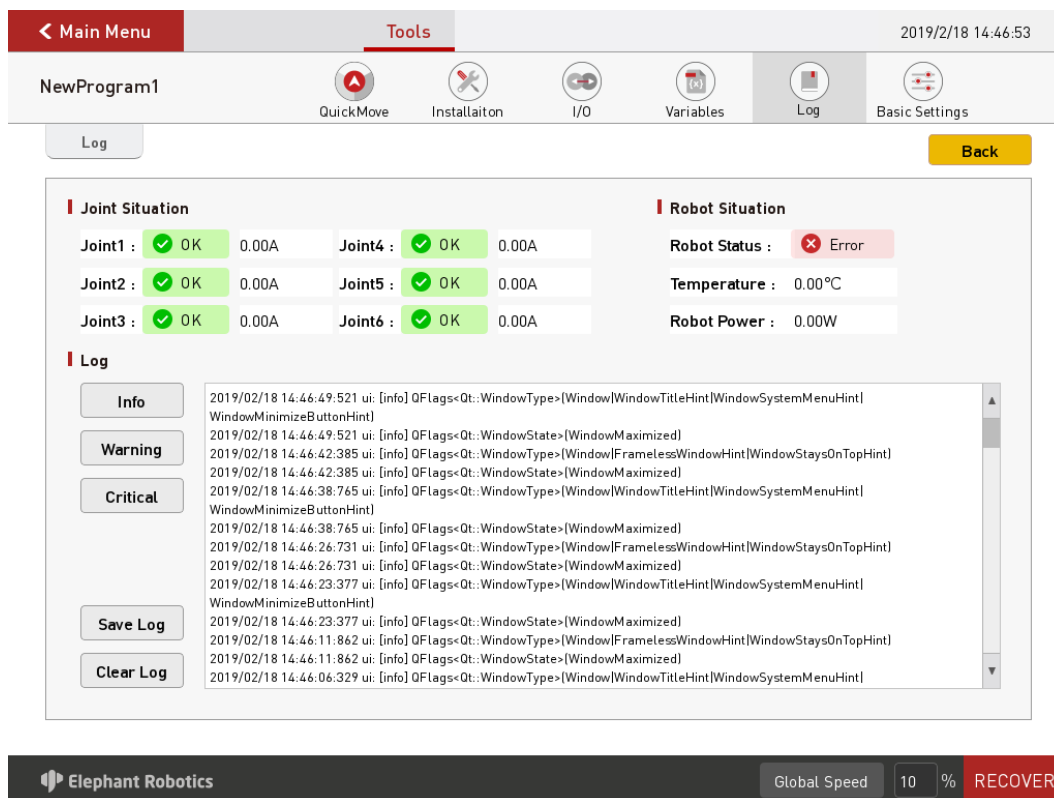


Figure 3- 20 Running log

5.1.3.6 Basic Settings

As shown in Figure 3-21, the basic settings page provides a common setting channel, allowing the user to quickly set up some functions, such as free movement related parameter settings, even when leaving the programming window while writing the program.

< Main Menu
Tools
2019/2/18 14:47:06

NewProgram1

QuickMove
Installaiton
I/O
Variables
Log
Basic Settings

Basic Settings
Back

Initialize Robot

Start Robot

Start Robot

Shutdown Robot

Shutdown Robot

Robot Status

✖
Error!

Set Freemove

Payload kg
(Maximum Payload: 5kg)

TCP

X mm

Y mm

Z mm

☒ Check the Payload when start Freemove function.

Elephant Robotics

Global Speed

10

%

RECOVER

Figure 3- 21 Basic Settings

5.1.4 Function instruction

5.1.4.1 Basic function

5.1.4.1.1 Waypoint

There are four types of waypoints: Absolute points, Relative points, Shared points, and Variables. These four types are side-by-side. Under one waypoint command, you can only choose one.

1, Absolute point: The absolute point is a description of the actual pose of the robot.

That is, as long as the robot records the absolute point, the next time the instruction is executed, regardless of the position of the robot (other settings unchanged), will reproduce the original teaching of the absolute point of posture.

The specific configuration page for absolute points is shown in Figure 4-1.

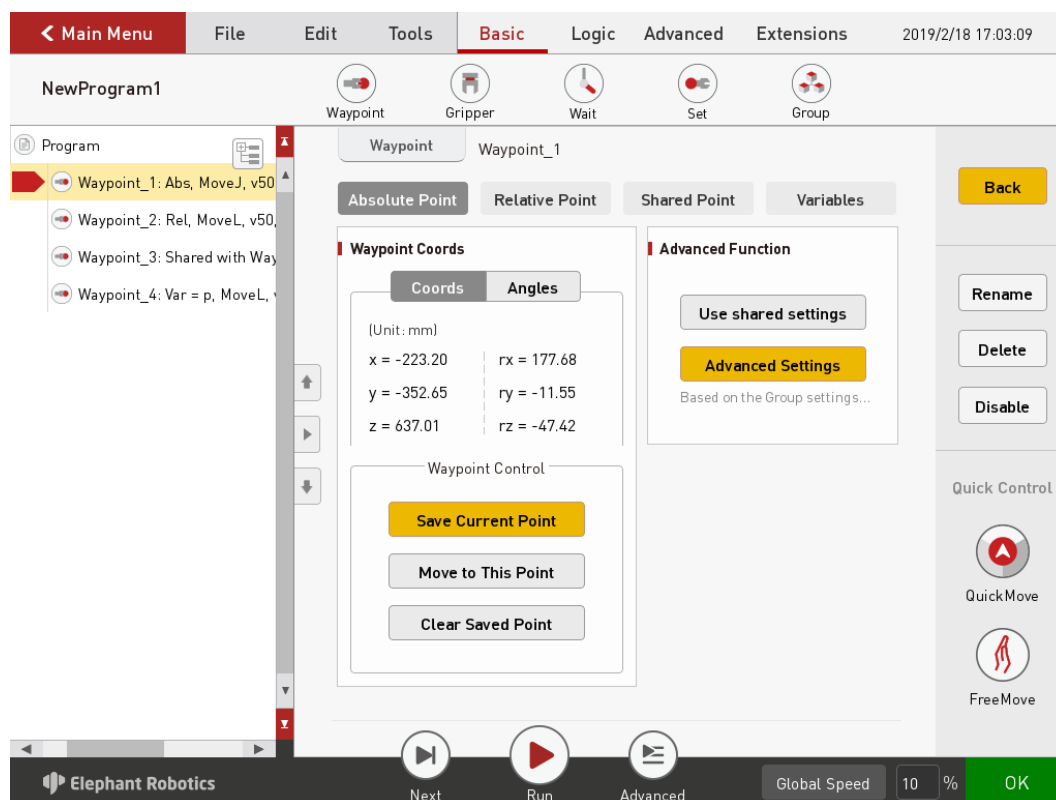


Figure 4- 1 Absolute point

1) Road Point coordinates

As shown in Figure 4-2, there are two formats for the representation of absolute points, namely Cartesian coordinate system coordinate values and joint angles. Among them, the Cartesian coordinate system coordinate value records the position and attitude of the robot TCP relative to the base coordinate system (in mm),.The joint angle is a direct record of the actual angle of each axis (in degree, degrees).

Coords	Angles	Coords	Angles
(Unit: mm)		(Unit: degree)	
x = -223.20	rx = 177.68	J1 = 53.85	J4 = 10.86
y = -352.65	ry = -11.55	J2 = 94.72	J5 = 91.68
z = 637.01	rz = -47.42	J3 = -11.03	J6 = -77.74
①- Coordinate		②- Angle	

Figure 4- 2 Absolute point Position data

2) Waypoint control

A. Save current point

This button is used to save the current pose data of the robot.

B. Move to this point

If you need to verify the teaching point or move to the teaching point for some operations, press and hold the button until the robot moves to the current teaching point.

C. Clear saved points

If the current teaching point is no longer needed, this button is used to clear the current teaching point.

1) Advanced Features

A. Shared configuration: This feature is being debugged, so stay tuned!

B. Advanced configuration

As shown in Figure 4-3, in the advanced configuration page, the user can set the movement mode, proximity mode, command speed, and torque limit.

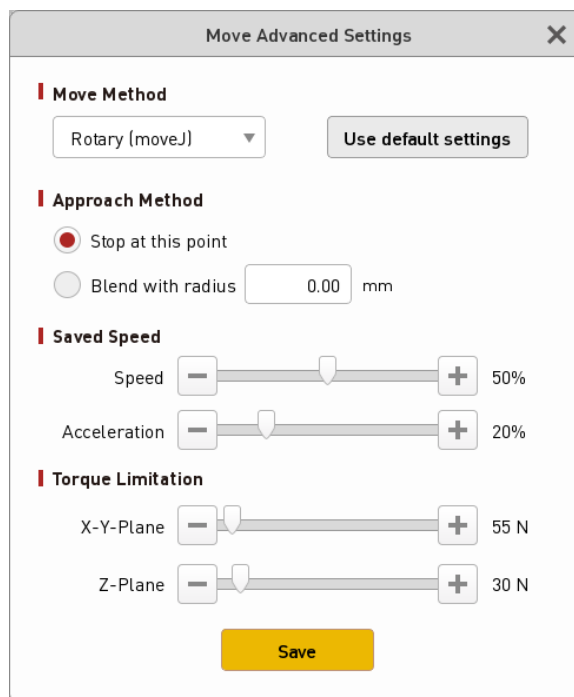


Figure 4- 3 Advanced configuration

2, Relative point: Figure 4-4 shows the specific configuration page of the relative point.

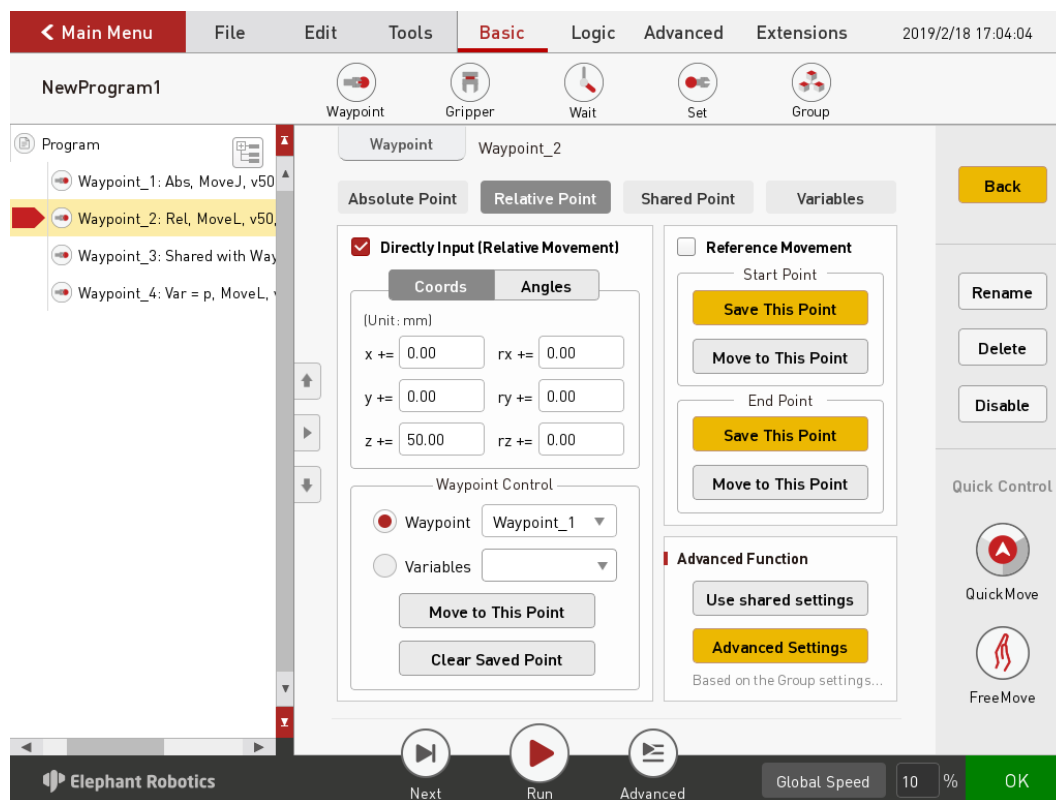


Figure 4- 4 Relative point

The relative point is used in a situation where a certain displacement is required based on a corresponding point of the movement instruction on the robot/an absolute point/variable point offset. The displacement can be a distance in a single direction, or a superposition of displacements in multiple directions, and can also teach a segment to offset.

1) Direct input (relative movement)

As shown in Figure 4-5, you can directly enter the coordinate value / joint angle.

①- Directly enter coordinate values

②- Directly input joint angle

Figure 4- 5 Two forms of direct input

Regardless of whether the coordinate value or the joint angle is input, one or more of the six values are selected according to the offset requirement, and not every value is required to be input.

For example, as shown in Figure 4-6, in the actual pickup and placement process, it is necessary to set a transition point above the target placement position. At this time, we can set a path command as absolute point, control the

robot (at this time the robot should be holding the state of the workpiece) to move to the placement point, click to save the current point, which generates the command line 2 shown in Figure 4-6. Then click on the basic function - waypoint: select the relative point, set the z-direction of the icon to increase the relative point of 50mm, then the robot will move to the position of the transition point after running the last sentence. In the actual pickup and placement process, other instructions, such as setting instructions, may be added between the two instructions to open the gripper.

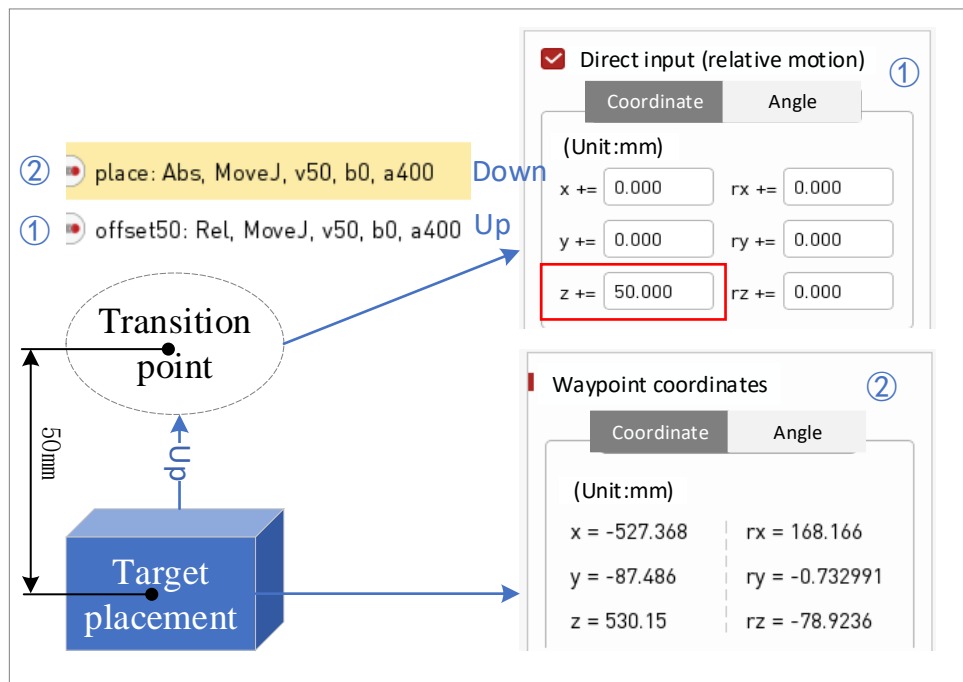


Figure 4- 6 Application examples of direct input coordinate values

In addition to offset based on the position of the last motion instruction, relative point instruction can also be offset based on a Waypoint or Variable point.

The "Move to this" button verifies the offset motion, and "Clear Saved Points" clears the currently entered content.

2) Reference move: By teaching two points, a path is generated, based on the current point, and the track is reproduced.

3) Advanced Features: The advanced configuration of the absolute point is not repeated here.

1, Shared point: The share point can use the location of other waypoints. Figure 4-7 shows the specific configuration page of the share point.

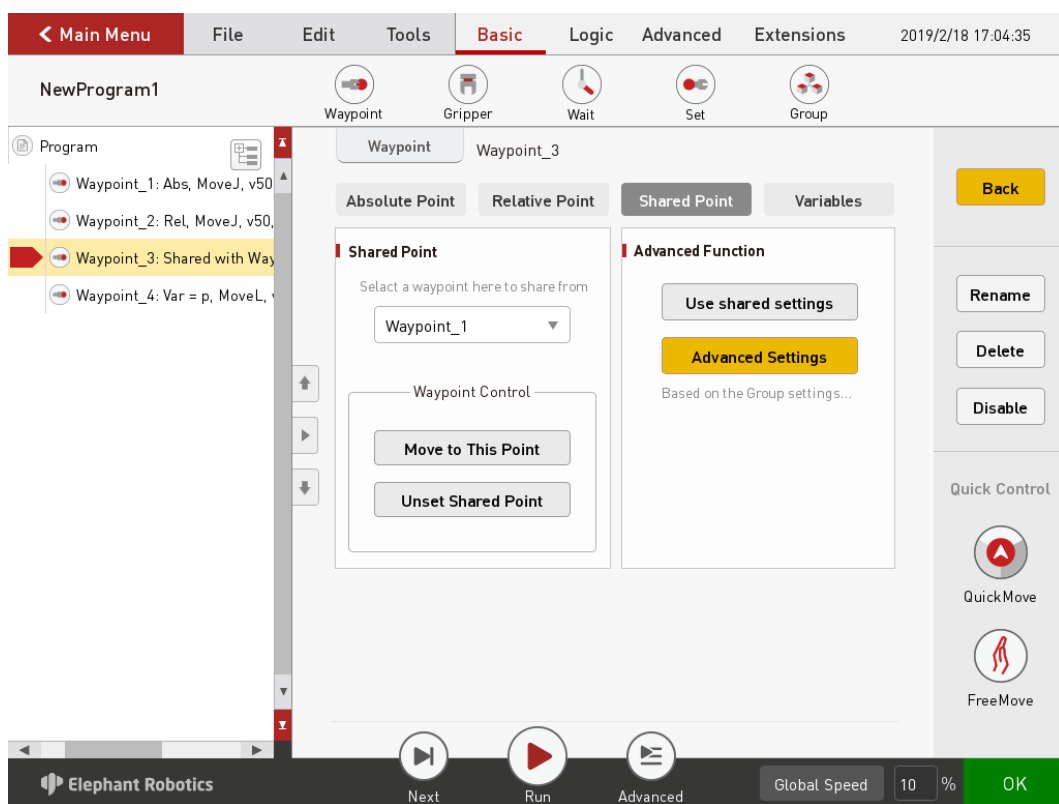


Figure 4- 7 Shared point

1) Shared point: Select the point you want to share in the box, you can keep pressing the "Move to this point" button to control the robot to move to that point. If you click "Clear Saved Points" to clear the current share point.

2) Advanced Features: The advanced configuration of the absolute point is

not repeated here.

- 3, Variable: The waypoint can be assigned by a variable. The user can use the communication method to obtain the waypoint location from other devices.

Figure 4-8 shows the specific configuration page of the variable point.

- 1) Variable assignment: The user can select the associated pose variable, and "Move to this point" can check whether the pose is the target pose.
- 2) Advanced Features: The advanced configuration of the absolute point is not repeated here.

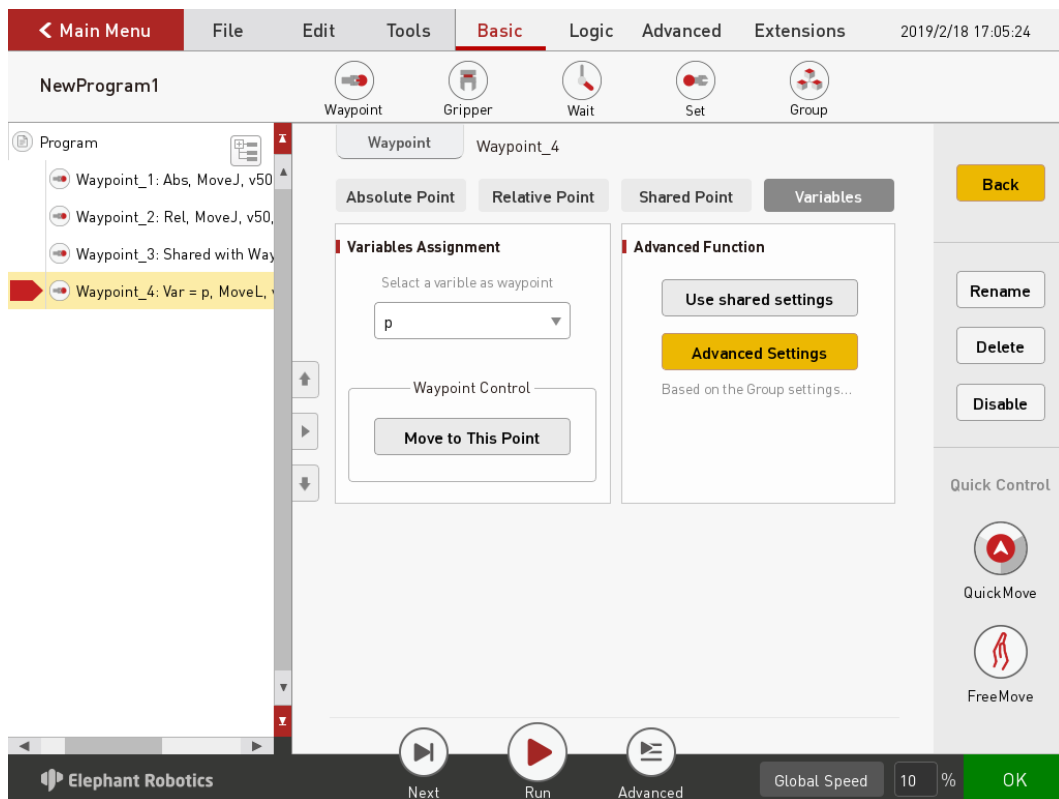


Figure 4- 8 Variable

5.1.4.1.2 Gripper

Figure 4-9 shows the specific configuration page of the gripper.

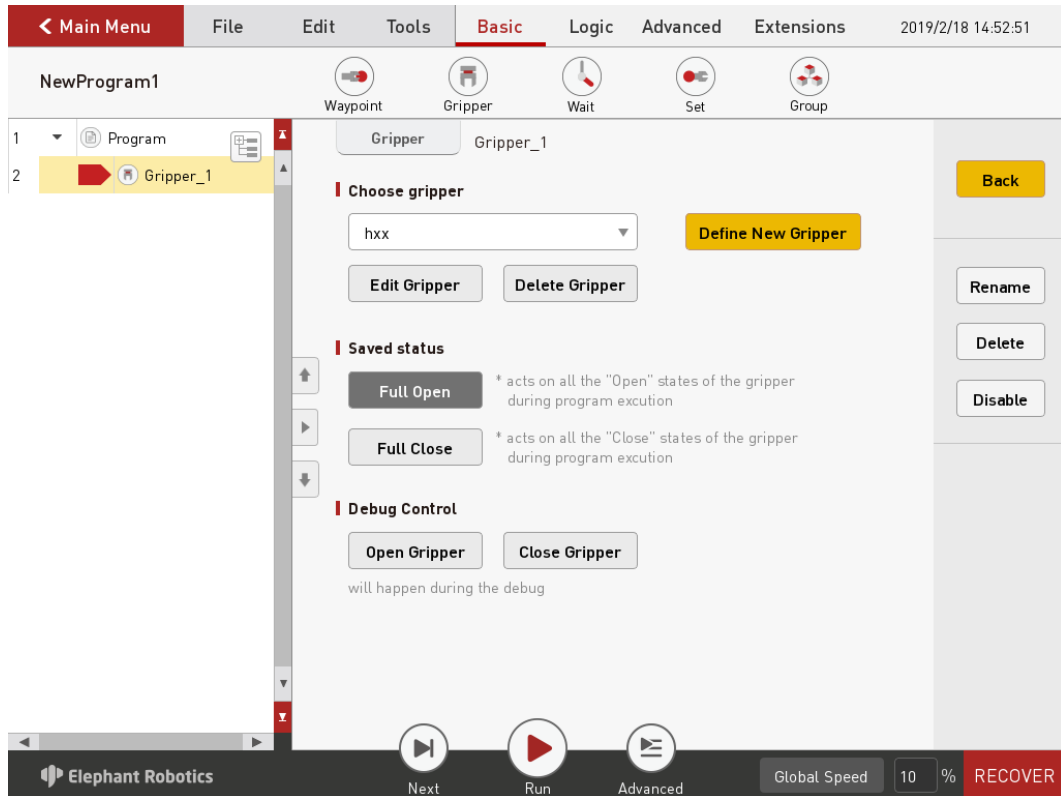


Figure 4- 9 Gripper

The user defines and controls the gripper through a simple function.

1, Select gripper

1) Set existing grippers

Select the gripper, you can edit or delete the existing gripper.

2) Define new grippers

As shown in Figure 4-10, the gripper can be named and multiple input signals can be controlled simultaneously. What needs to be set is: Set the number of output signals that need to be controlled, select the set of several signals in settings, set the status (related to the specific execution corresponding to the "open" or "off" function), set the corresponding output signal. After the setup is complete, you can also select a wait

condition.

define new gripper [X]

Name this gripper

Gripper Name [] Clean Name []

Connect signal

Set IO

Set Pin Count [- 0 +] No. [] States of Pin [Open] [Close]

Set Pin

Pin [DO_0] [Keep] keep this status

Signal (Output) [High] [Low] [Delay] 0 ms [Once] active for 200ms

Wait

☒ Wait time [0.00] s ☐ Wait Digital Input

Pin [DI_0] is [High] [Low]

Save []

Figure 4- 10 Define new grippers

2, Set the saved state

- 1) Fully open: The option in the execution gripper definition is the "open" state.
- 2) Completely off: The option in the execution gripper definition is "off" status.

3, Debug control

- 1) Open the gripper: Manual operation performs the option of the "Open"

state in the gripper definition.

- 2) Close the gripper: Manual operation performs the option of the "Close" state in the gripper definition.

5.1.4.1.3 Wait

As shown in Figure 4-11, there are four modes for waiting for instructions.

- 1, Waiting time: The delay time can be set in seconds.
- 2, Waiting for the input signal: The state of the input signal is judged and waits until it meets the set input signal state condition.
- 3, Waiting for the output signal: The state of the output signal is judged and waits until it meets the set output signal state condition.
- 4, Waiting conditions: You can customize the wait condition and wait until the wait condition is met.

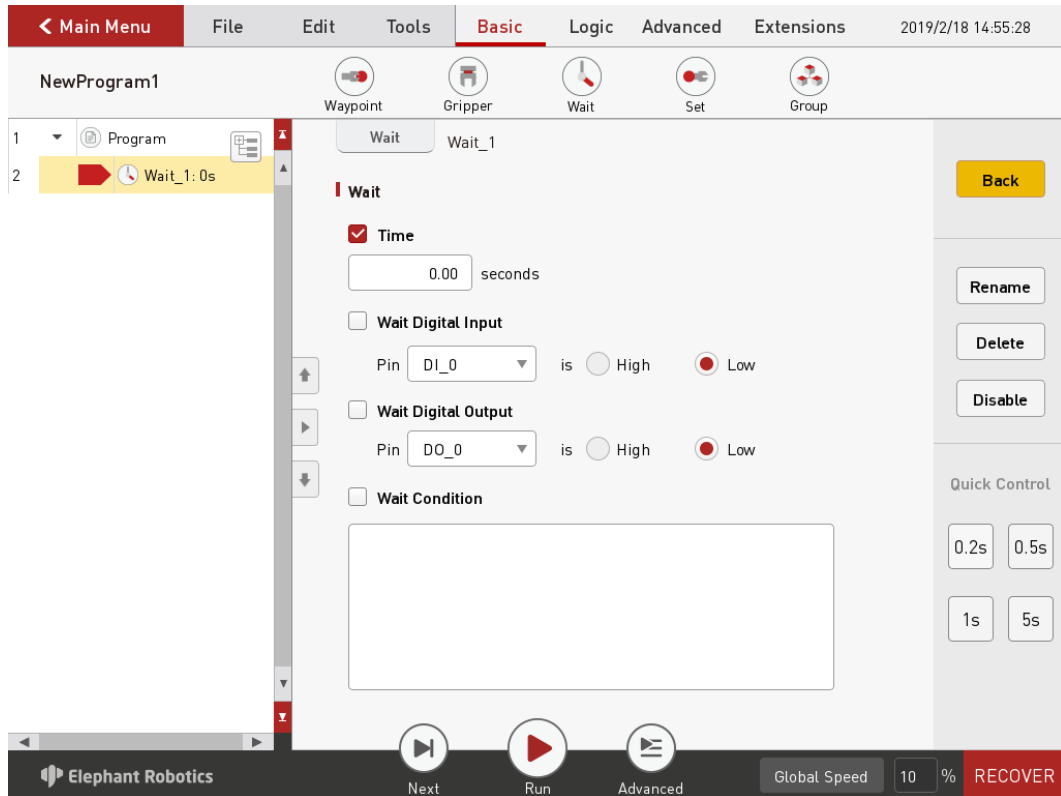


Figure 4- 11Wait

5.1.4.1.4 Set

As shown in Figure 4-12, the setup command has four modes of selection.

- 1, Set IO: Set the state of the output signal. In addition to selecting the set output signal to determine whether it is on or off, you can also set the time that the signal is held.
- 2, Set conditions: Customize the content of the settings.
- 3, Set TCP (i.e. tool center point).
- 4, Set the load.

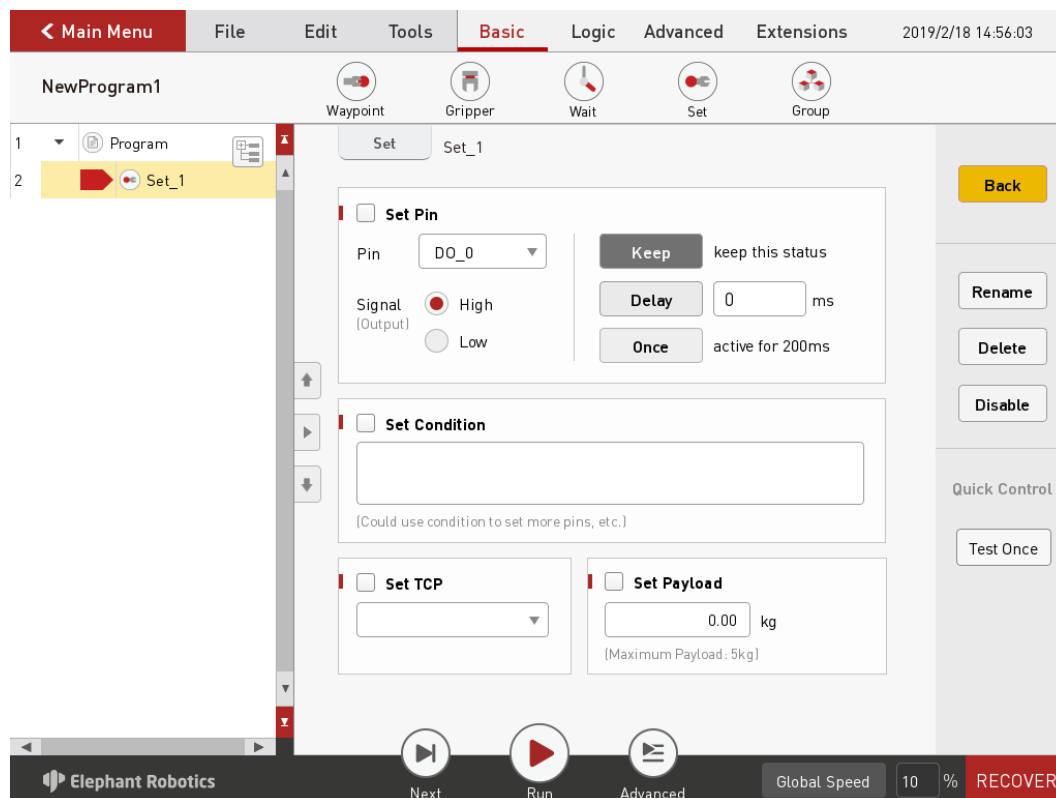


Figure 4- 12 Set

5.1.4.1.5 Group

As shown in Figure 4-13, the group instructions provide common combination templates, such as grabbing and placing combinations.

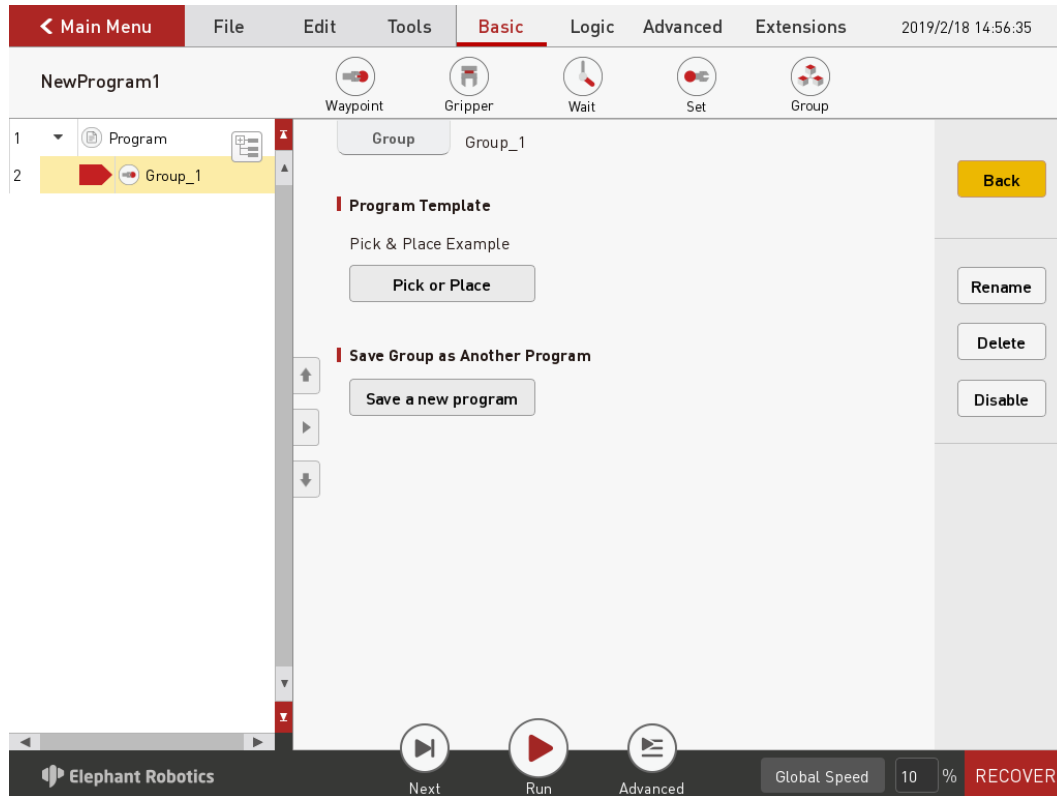


Figure 4- 13 Group

When users use group instruction, such as grabbing and placing combinations, they can modify parameters and teach Waypoints directly on the basis of template programs, or they can add or delete instructions freely according to their needs.

The user can simplify the process of finding instructions by using the group instruction. And it is more convenient and quicker to complete the programming of the corresponding project.

5.1.4.2 Logic function

5.1.4.2.1 Loop

Loop instructions can repeat all instructions within a loop for a certain

number of times. As shown in Figure 4-14, the number of loops can be represented by a constant or a variable or an expression.

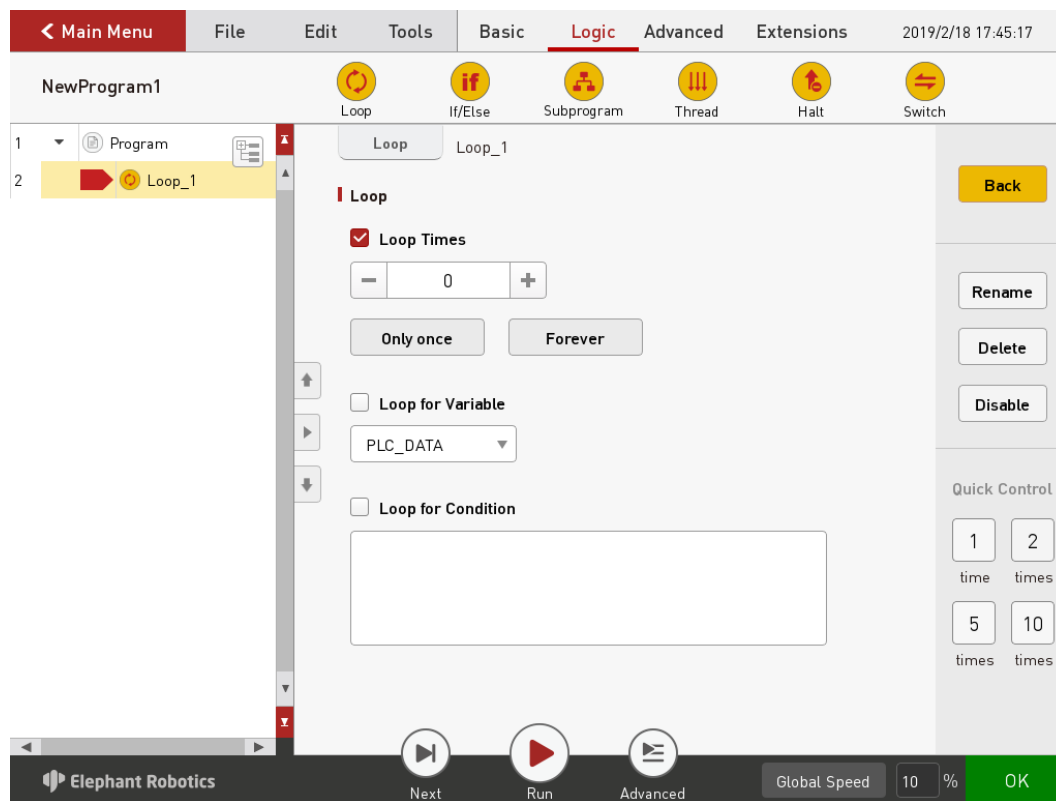


Figure 4- 14 Loop

5.1.4.2.2 If/Else

Judging the set conditions allows the program to read the data, determine and determine what to do next.

If/Else can be used to determine the I/O signal and can also be used to determine other conditions.

If/Else consists of three parts: If, Else If, and Else. The relationship between these three parts is as follows:

- 1, Except that If is an integral part, the remaining two are optional parts;
- 2, If both If, Else If, and Else exist, the program will first read If, then read Else

If, Else If ... Else. The relationship between the three is shown in Figure 4-15:

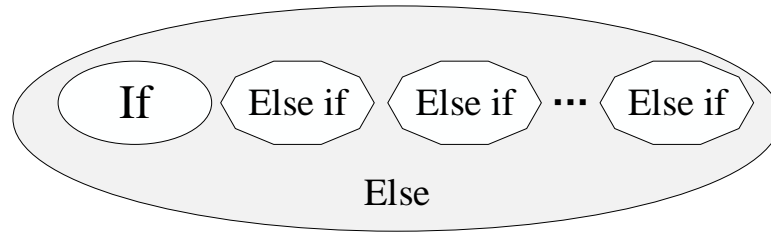


Figure 4- 15 Relationship between If, Else If, and Else

Both If and Else If judge the determined conditions, and Else corresponds to all cases except the above conditions.

- 3, There can be more than one Else If, but there is only one If, and if you choose to add Else, you can only have one Else.
- 4, You can delete Else If or Else, but if you delete If, delete all Else If and Else.

Figure 4-16 shows the setting page of the conditional judgment command.

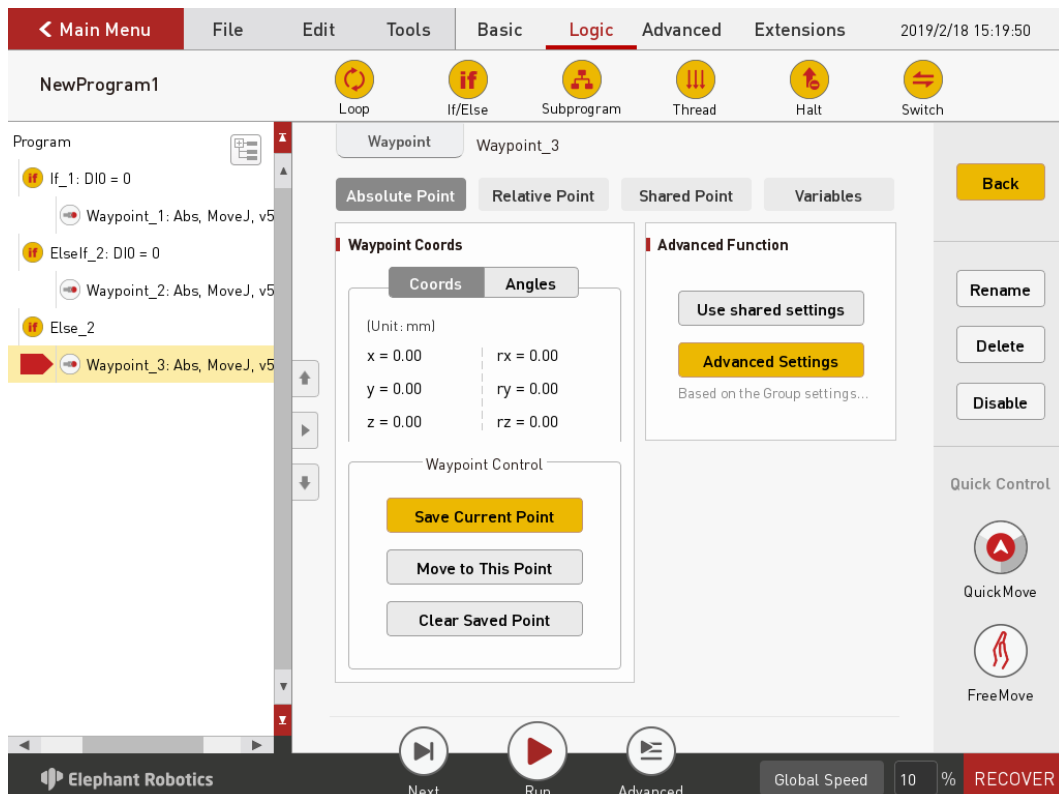


Figure 4- 16 If/Else

As shown in the figure above, if the condition following "If" is met, the robot will move to waypoint 1; if it meets the condition followed by "Else if", it will move to waypoint 2; if both conditions are not met, the "Else" corresponding block will be executed, that is, the robot will move to the waypoint 3.

5.1.4.2.3 Subprogram

As shown in Figure 4-17, other subroutines can be called using this instruction. The main program can use multiple subroutines, but there are no subroutines in the subprogram.

As shown in Figure 4-18, you can view and edit subroutines in the main program. If you edit the subroutine, please note that it will not take effect until it is saved.

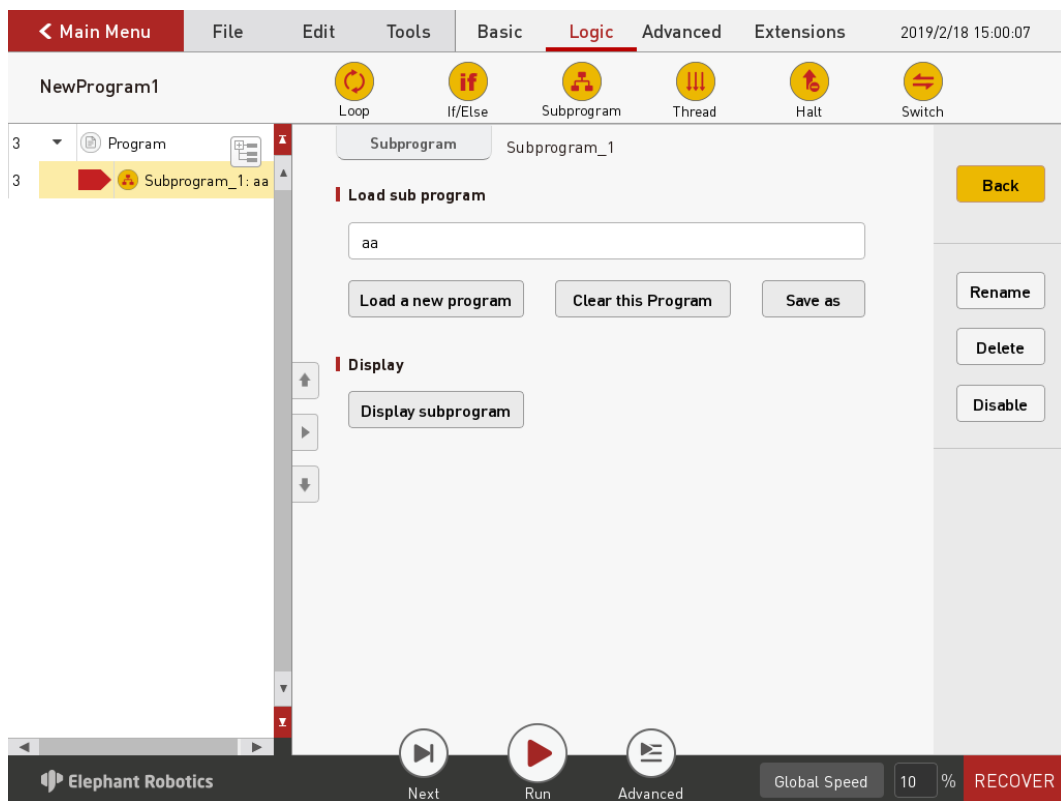


Figure 4- 17 Subprogram

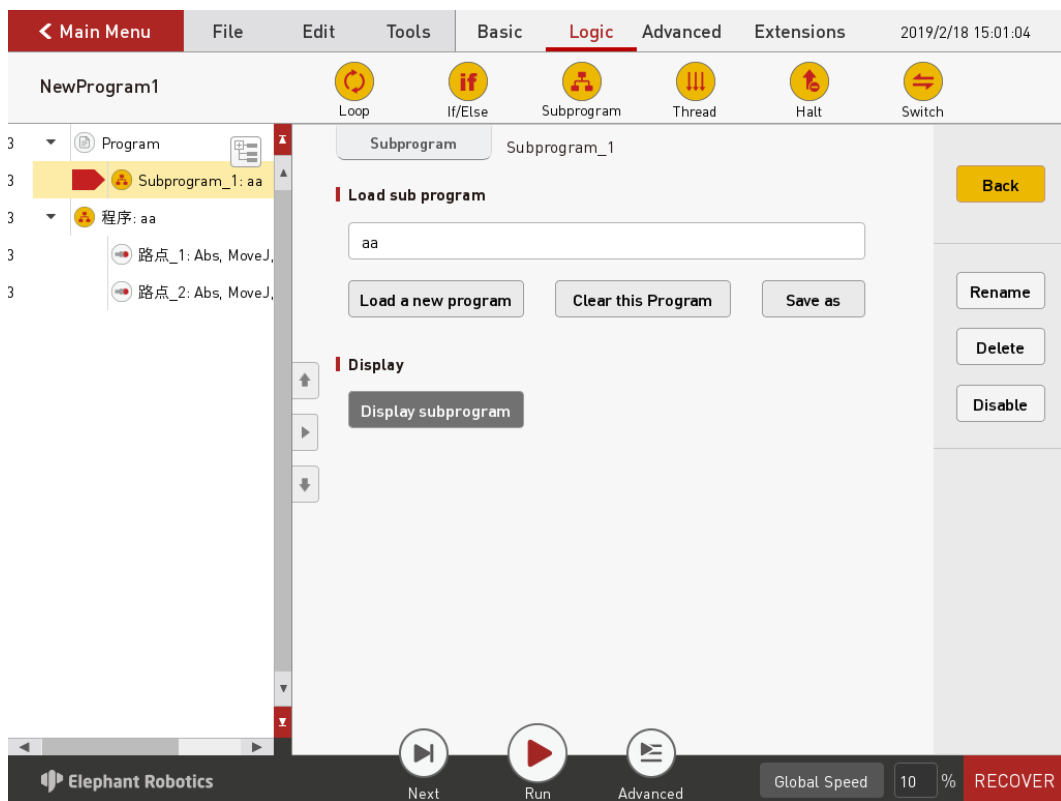


Figure 4- 18 Display subroutine

5.1.4.2.4 Thread

The thread runs along the main program. It is used to check signals such as emergency buttons or safety light curtains. As shown in Figure 4-19, you can set the running interval of threads.

Note that motion instructions are not allowed in threads.

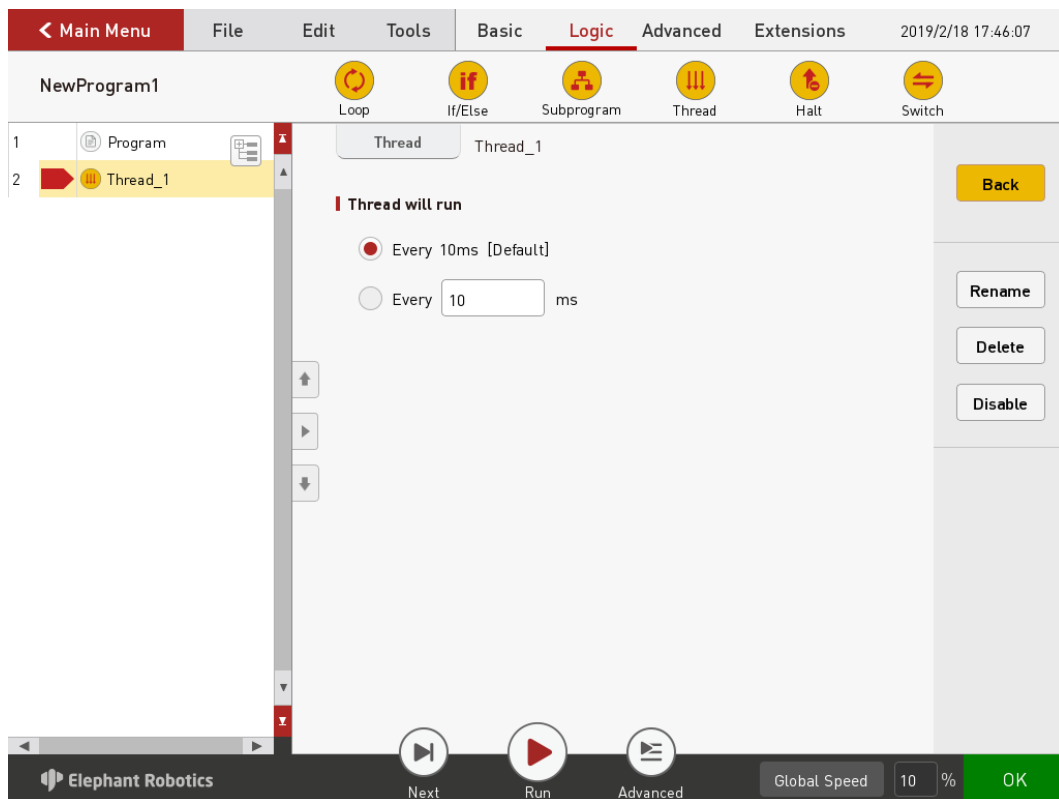


Figure 4- 19 Thread

5.1.4.2.5 Halt

The pause command is used to control the robot to pause, stop, and resume. Figure 4-20 shows the specific configuration page for the pause command.

When setting the pause and stop status, you can also select “Show Popup” to customize the contents displayed by the popup.

Set the restart state. When the program runs to this instruction, it will start running again from the first instruction at the beginning.

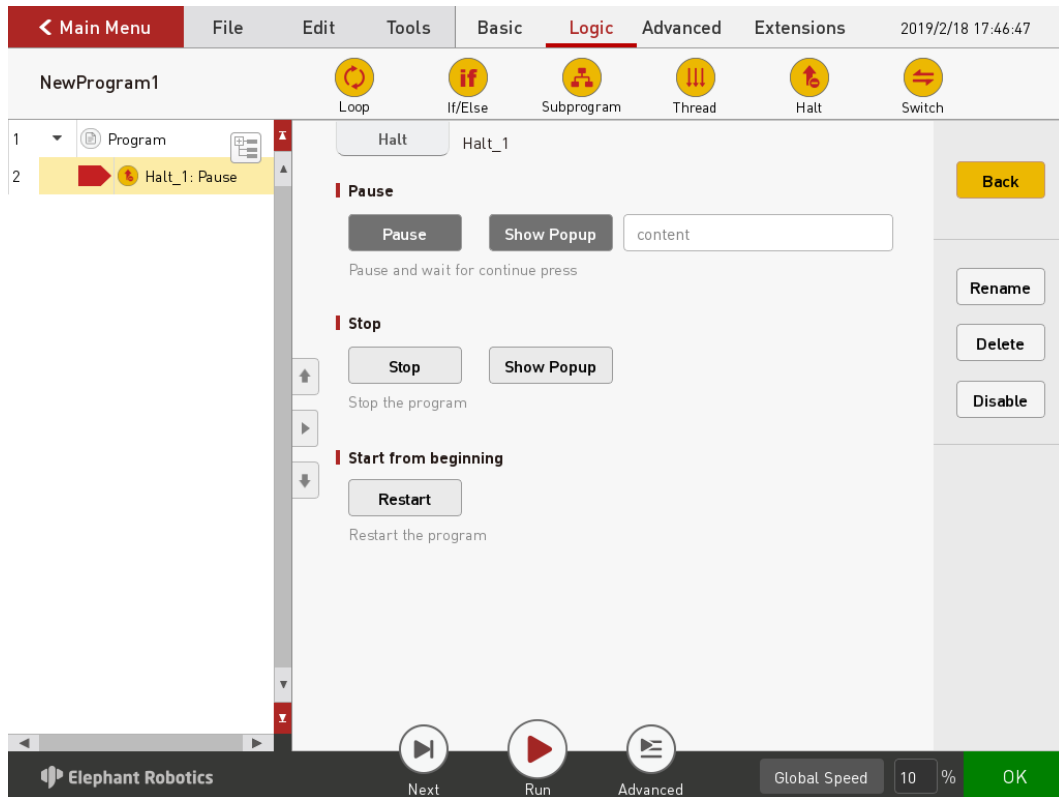


Figure 4- 20 Halt

5.1.4.2.6 Switch

As shown in Figure 4-21, the conditional selection instruction is used to judge the value of a variable.

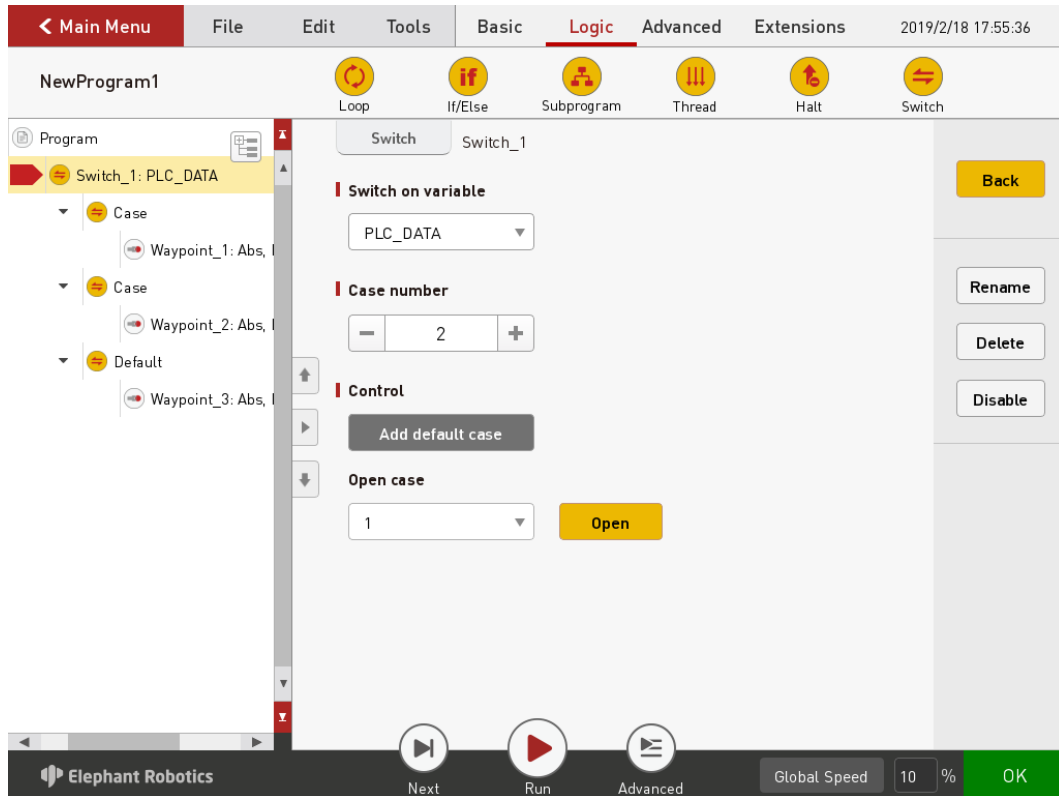


Figure 4- 21 Switch

Corresponding to different conditional values, how many conditional values need to be judged to add how many cases, you can open each case, increase the corresponding execution instructions. For example, to judge the integer variable A, set two cases, if A is 1, execute the first route instruction, if A is 2, execute the second route instruction.

If only a few variables are judged, and other cases are handled uniformly, we need to select "default" and add corresponding instructions to the switch.

5.1.4.3 Advanced function

5.1.4.3.1 Pallet

The pallet instruction allows the user to teach only a few points, through

which the position of the other points can be calculated by the robotic system.

Running this instruction can control the movement of the robot to these points.

As shown in Figure 4-22, you can select a line, plane, cube, discrete point.

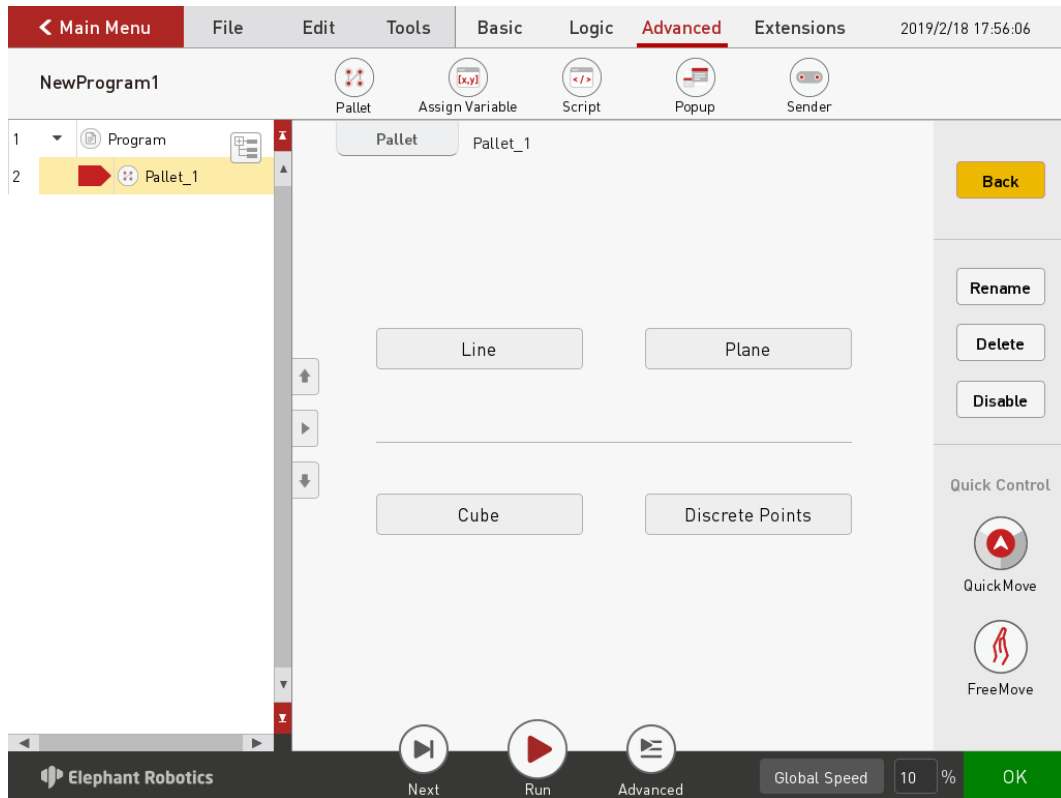


Figure 4- 22 Pallet type Selection

As shown in Figure 4-23, after you select line, select the number of points, and the line will be split evenly based on the number of points. These points are the split point. The user determines this line by teaching two points.

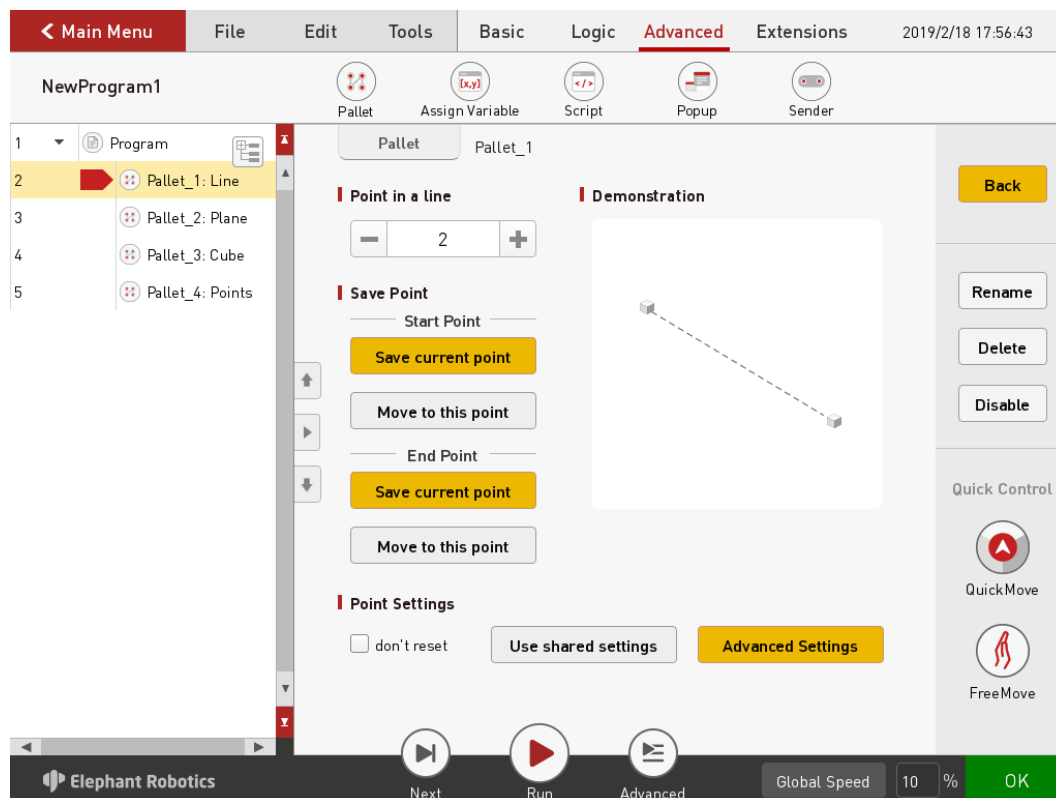


Figure 4- 23 Line

As shown in Figure 4-24, after selecting “Plane”, select the number of points of the two axes, and the plane is divided equally. These points are the dividing points. This plane is determined by teaching four points.

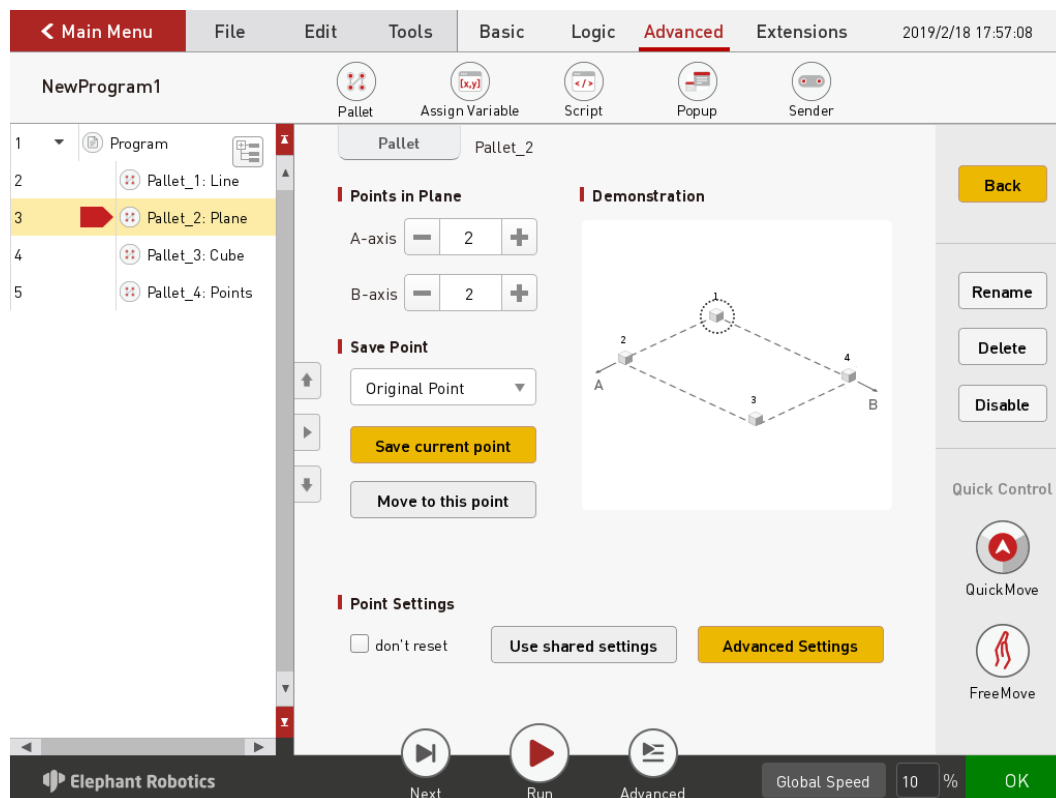


Figure 4- 24 Plane

As shown in Figure 4-25, after selecting "Cube", select the number of points of the three axes, and the cube is divided equally. These points are the dividing points. Determine this cube by teaching eight points.

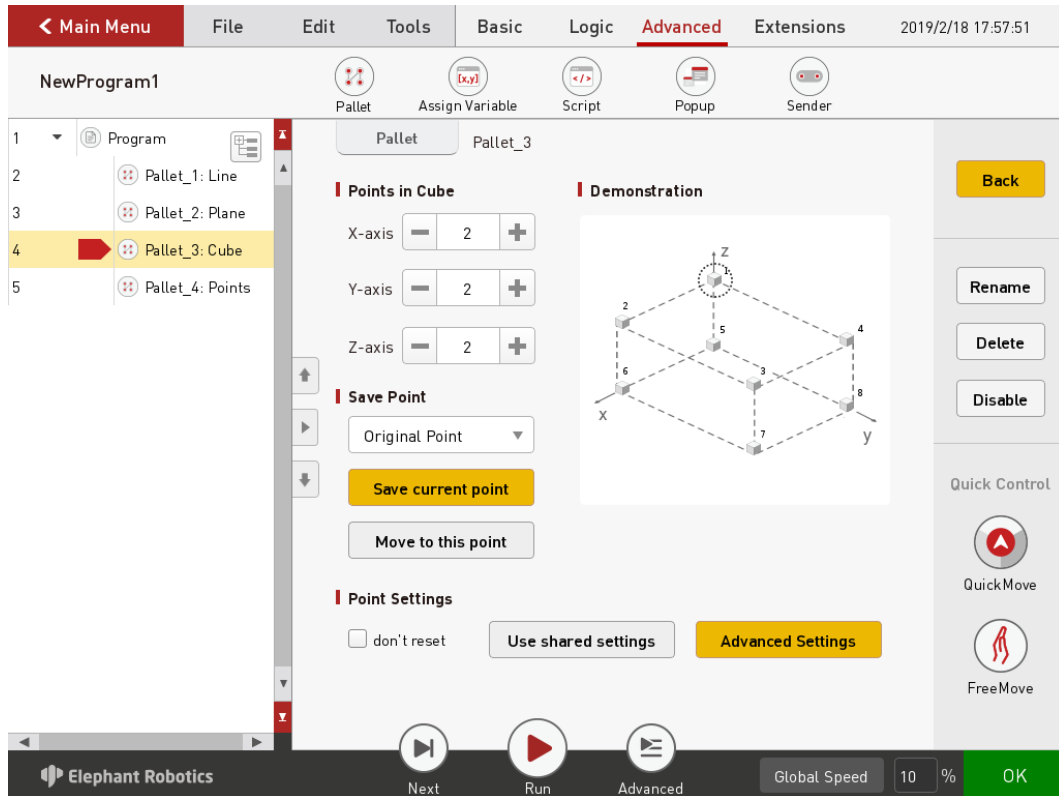


Figure 4- 25 Cube

As shown in Figure 4-26, when “Discrete Point” is selected, the number of points is selected to teach different points. That is, a discrete point is a collection of multiple points.



Attention

Whichever type is chosen, the robot will move to the first point when this instruction is executed for the first time. The second time, the robot moves to the second point. The nth time the instruction is run, it moves to the nth point. Until all the points have arrived, the robot will start again from the first point.

Note that if you need to control the position of the robot to reach each point separately, this command needs to be used in conjunction with the loop instruction. This instruction is only executed once and only controls the robot to reach the first point.

If you select "Do not reset", then stop running the program before all points have been executed. When you run it again, it will start from the point of the last interruption, not from the first position.

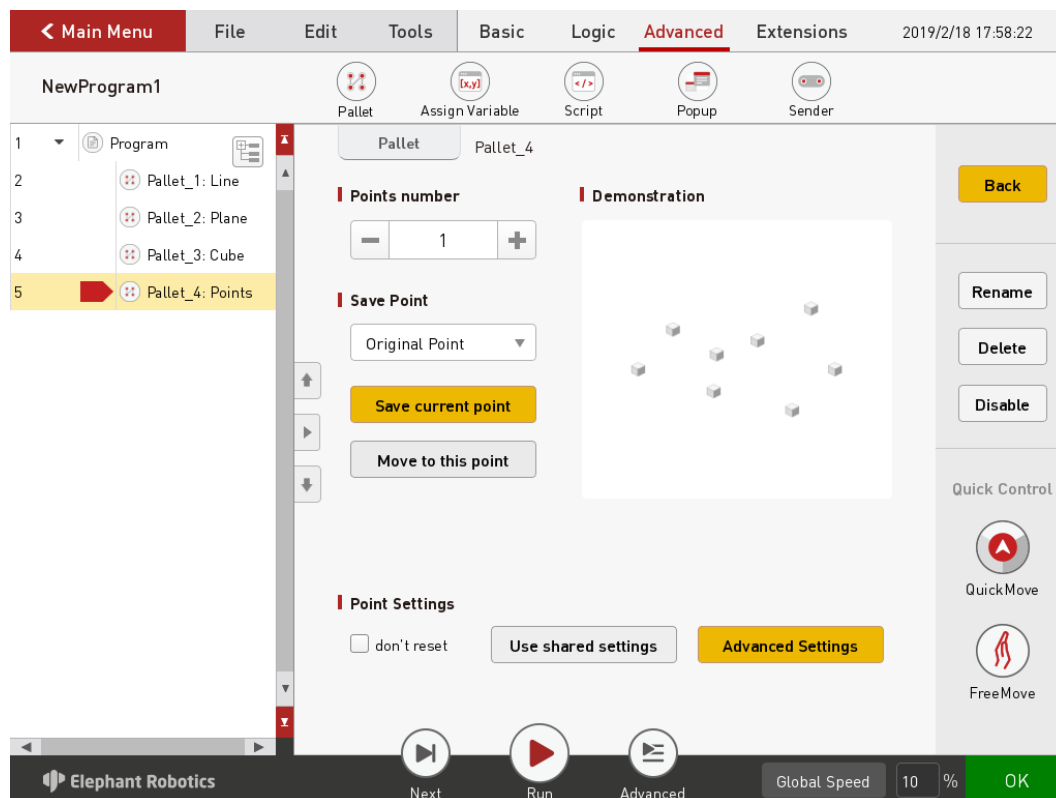


Figure 4- 26 Discrete point

5.1.4.3.2 Assign to var

As shown in Figure 4-27, this command can assign values to integer variables and string variables. You can also use the "set variables" to directly set the value of the variable according to the instruction.

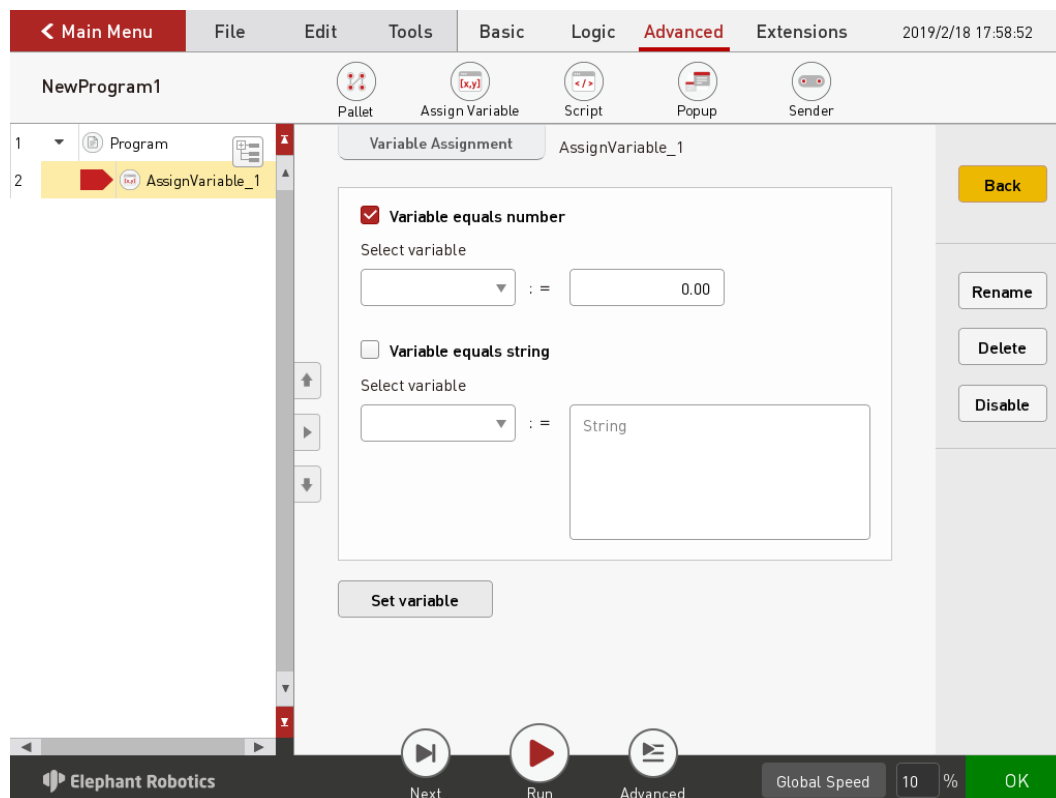


Figure 4- 27 Assign to var

5.1.4.3.3 Script

Script instructions can be used to edit complex instructions, providing a richer set of functional instructions. Figure 4-28 shows the specific configuration page of the script command. There are two types of setup scripts, one is a single-line expression and the other is a multi-line script.

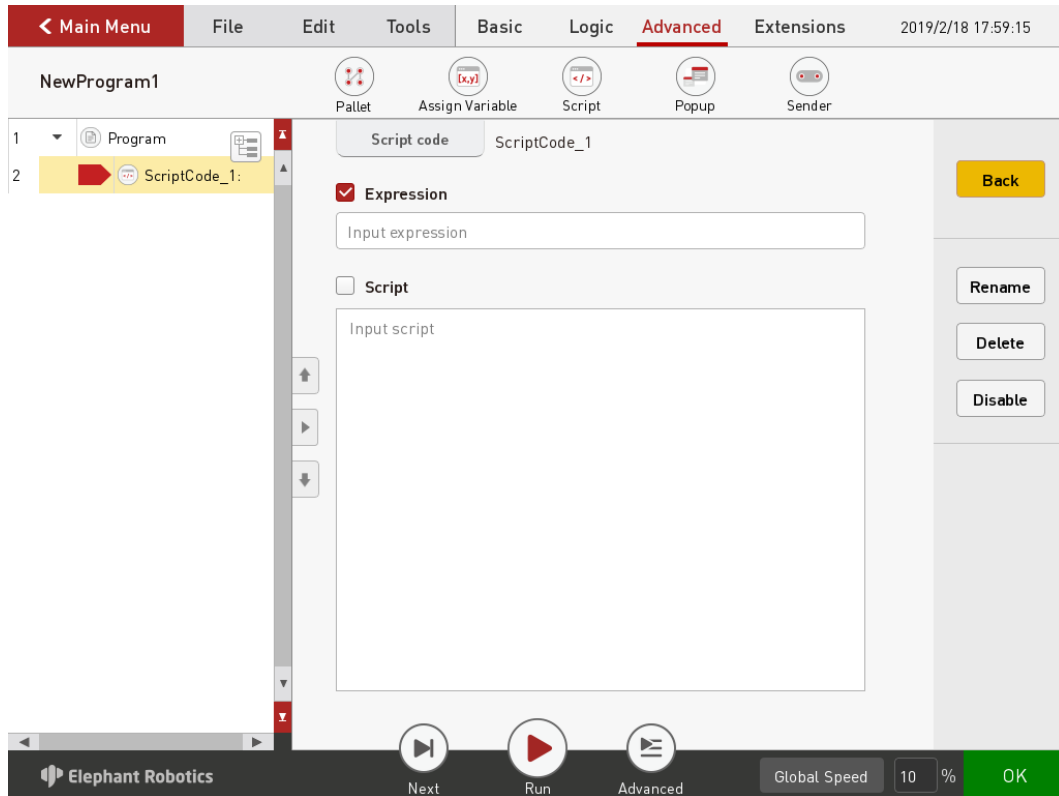


Figure 4- 28 Script

5.1.4.3.4 Popup

The pop-up command allows the user to customize the pop-up window. In other words, when this command is executed, a pop-up window appears, and the pop-up content is user-defined content. As shown in Figure 4-29, there are three types of pop-up windows, information, warnings, and errors. The user selects one and customizes the pop-up content.

There are also three kinds of pop-up window control: continue the program (logging), that is, do not pop the window, just display the contents of the pop-up window to the log, and the program continues to run; When the window is popped, the program is paused, that is, the pop-up window appears, and the program is suspended; When the window is popped, the program stops, and the

pop-up window appears, and the program stops running.

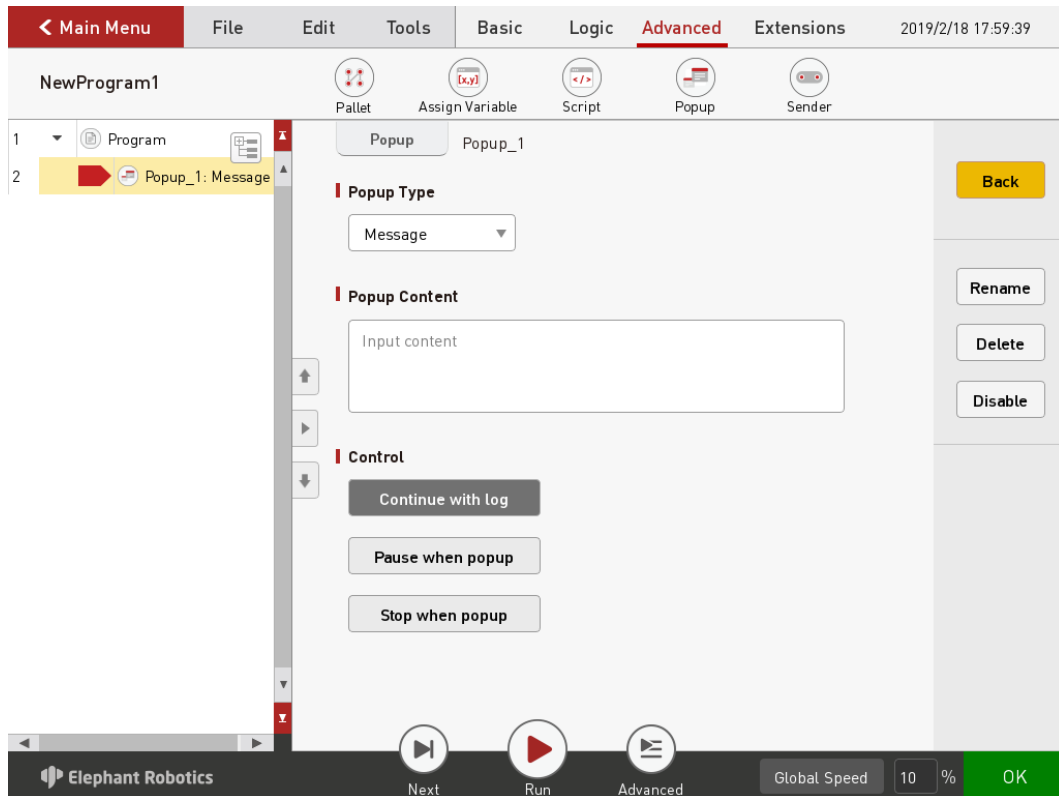


Figure 4- 29 Popup

5.1.4.3.5 Sender

If TCP/IP communication is to be performed, the robot system must set the IP and port number as a client or server to communicate with other devices.

The sender allows the user to set up a TCP/IP connection. Figure 4-30 shows the specific configuration page of the Sender instruction.

If the robot system acts as a client, the IP address filled in is the IP address of the external device of the server, and the port number corresponds to the port number assigned to the robot system by the server. When the server is in the state of monitoring, it can communicate with the server by clicking the "connection" button.

If the robot system serves as a server, the IP address filled in is the local IP address, and the port number corresponds to the port number assigned to the client device. Click on the "monitor" button, at which point the client device can connect to the robot system. In the client list, you can view the IP addresses and port numbers of all clients.

After establishing communication, data can be sent and received.

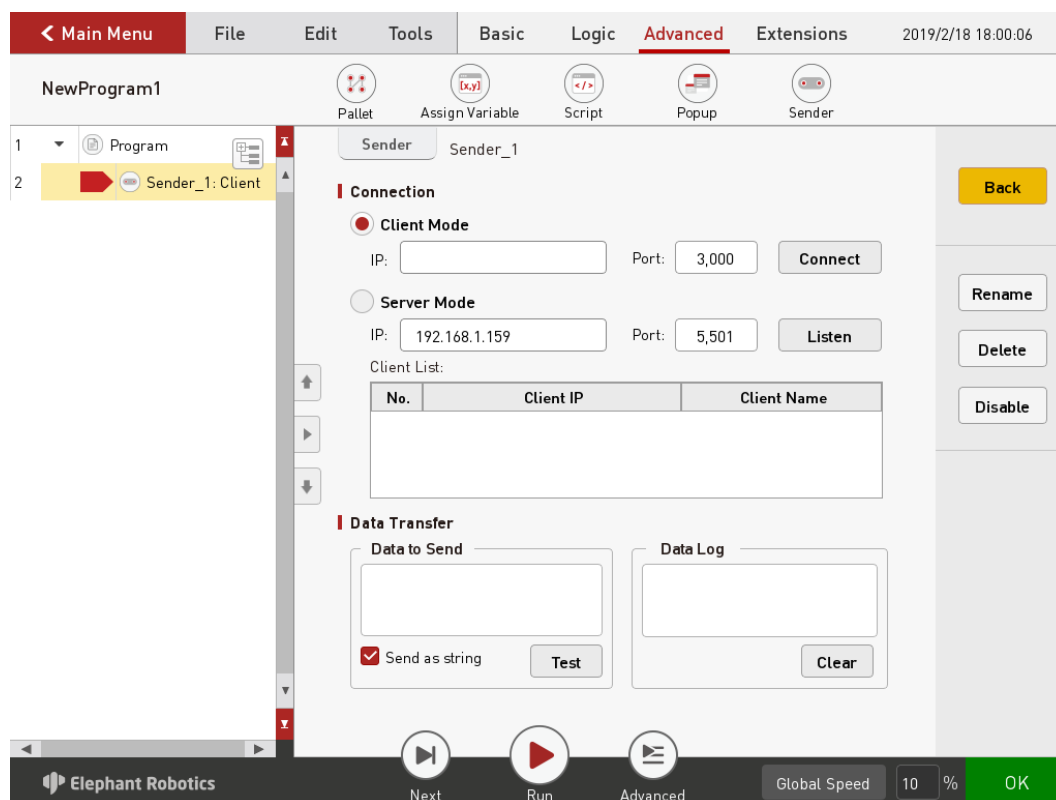


Figure 4- 30 Sender

5.2 API Interface Description

5.2.1 Overview

Elephant robot allow users to control the robot from remote, one way is use socket. We use tcp protocol to communicate between the client and the robot, you can send the formated string through tcp to get or set some property/state of the robot, the format for each function are introduced as bellow.

5.2.2 Socket String format rules

1 get current angles of robot

Socket string format: `get_angles()`
the return string is formated in key-value pair, the key is the funciton name, the value is the value from robot, like this: `get_angles:[0.174058, 0.520382, -0.07874, 0.092855, 0.0, 0.030356]`. If any error occurred, `InvalidAngles()` (defined as `[-1.0, -2.0, -3.0, -4.0, -1.0, -1.0]`) will be returned.

2 set the angles of robot

Socket string format: `set_angles(joint1_angle, joint2_angle, joint3_angle, joint4_angle, joint5_angle, joint6_angle, speed)`
Example: `set_angles(10.0, 11.0, 12.2, 12.3, 11.1, 16.0, 500)`
the return string is formated in key-value pair, the key is the funciton name, the value is the value from robot, like this: `set_angles:[ok]`. If any error occurred, you will get `set_angles:error_message`.

3 set the angle of one joint

Socket string format: `set_angle(joint, angle, , speed)`
example: `set_angle(J1, 50.5, 500)`
the return string is formated in key-value pair, the key is the funciton name, the value is the value from robot, like this: `set_angle:[ok]`. If any error occurred, you will get `set_angle:error_message`.

4 get current coordinates of robot

Socket string format: `get_coords()`
the return string is formated in key-value pair, the key is the funciton name, the value is the value from robot, like this: `get_coords:[0.174058, 0.520382, -0.07874, 0.092855, 0.0, 0.030356]`. If any error occurred, `InvalidCoords()` (defined as `[-1.0, -2.0, -3.0, -4.0, -1.0, -1.0]`) will be returned.

5 set the coordinates of robot

Socket string format: `set_coords(axis_x_coord,axis_y_coord,axis_z_coord,axis_rx_coord,axis_ry_coord,axis_rz_coord,speed)`

Example: `set_coords(10.0,11.0,12.2,12.3,.11.1,16.0,500)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `set_coords:[ok]`. If any error occurred, you will get `set_coords:error_message`.

6 set the coordinate of one axis

Socket string format: `set_coord(axis,coordinate ,speed)`

example: `set_coord(x,50.5,500)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `set_coord:[ok]`. If any error occurred, you will get `set_coord:error_message`.

7 get the signal of digital out pin

Socket string format: `get_digital_out(pin_number)`

example: `get_digital_out(1)` the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `get_digital_out:1`. If any error occurred, you will get `get_digital_out:error_message`.

8 set the signal of digital out pin

Socket string format: `set_digital_out(pin_number,signal)`

example: `set_digital_out(1,1)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `set_digital_out:[ok]`. If any error occurred, you will get `set_digital_out:error_message`.

9 get the signal of digital in pin

Socket string format: `get_digital_in(pin_number)`

example: `get_digital_in(1)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `get_digital_in:1`. If any error occurred, you will get `get_digital_in:error_message`.

10 set the signal of analog out pin

Socket string format: `set_analog_out(pin_number,signal)`

example: `set_digital_out(1,1.5)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: set_analog_out:[ok]. If any error occurred, you will get set_analog_out:error_message.

11 change the coordinate of one axis in one direction continuously

Socket string format: jog_coord(axis,direction,speed)

example: jog_coord('x', 1, 500)

The direction can be -1, 0, 1, -1 means in negative direction, 1 means positive direction, 0 means stop. the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: jog_coord:[ok]. If any error occurred, you will get jog_coord:error_message.

12 change the angle of one joint in one direction continuously

Socket string format: jog_angle(joint,direction,speed)

example: jog_coord('J1', 1, 500)

The direction can be -1, 0, 1, -1 means in negative direction, 1 means positive direction, 0 means stop. the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: jog_angle:[ok]. If any error occurred, you will get jog_angle:error_message.

13 enable the system

Socket string format: state_on()

example: state_on()

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: state_on:[ok]. If any error occurred, you will get state_on:error_message.

14 disable the system

Socket string format: state_off()

example: state_off()

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: state_off:[ok]. If any error occurred, you will get state_off:error_message.

15 stop the task

Socket string format: task_stop ()

example: task_stop()

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: task_stop:[ok]. If any error

occurred, you will get state_off:error_message.

16 set feed rate

Socket string format: set_feed_rate(speed)

example: set_feed_rate(50.0)

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success, it will return: set_feed_rate: 0. otherwise it means failed.

17 make the robot 'sleep' for some seconds

Socket string format: wait(seconds)

example: wait(10.5)

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success, it will return: wait:[ok]. This function will make the robot 'sleep' for given seconds, seem like use sleep in your code.

18 mount the robot upside down

Socket string format: set_upside_down(up_dn)

example: set_upside_down(1)

1 means upside down, 0 means not.

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: set_upside_down:[ok]. If any error occurred, you will get set_upside_down:error_message.

19 power on the robot

Socket string format: power_on()

example: power_on()

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: power_on:[ok]. If any error occurred, you will get power_on:error_message

20 power off the robot

Socket string format: power_off()

example: power_off()

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: power_off:[ok]. If any error occurred, you will get power_off:error_message

21 get the speed the robot

Socket string format: `get_speed()`

example: `get_speed()`

the speed unit is mm/s

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `get_speed:500`. If any error occurred, you will get `get_speed:error_message`

22 check the state of the robot

Socket string format: `state_check()`

example: `state_check()`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if the robot is in normal state, you will get: `state_check:1`, if the robot is not in normal state, you will get: `state_check:0`. If any error occurred, you will get `get_speed:error_message`

23 check if the robot is running

Socket string format: `check_running()`

example: `check_running()`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if the robot is running, you will get: `check_running:1`, if the robot is not running, you will get: `check_running:0`. If any error occurred, you will get `get_speed:error_message`

24 set the torque limit of the robot

Socket string format: `set_torque_limit(axis,torque)`

example: `set_torque_limit(x,10.0)`

the axis can be x,y or z

the unit of torque is N

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: `set_torque_limit:[ok]`. If any error occurred, you will get `set_torque_limit:error_message`.

25 open a g_code formatted text file

Socket string format: `program_open(file_path_name)`

example: `program_open(/usr/a.txt)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: `program_open:0`. If any error occurred, you will get `program_open:error_message`.

26 run a g_code formatted text file from the given line

Socket string format: `program_run(line_number)`

example: `program_run(0)`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: `program_run:0`. If any error occurred, you will get `program_run:error_message`.

27 get the robot error

Socket string format: `read_next_error()`

example: `read_next_error()`

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: `read_next_error:message`.

28 set the payload of the robot

Socket string format: `set_payload(payload)`

example: `set_payload(5.0)`

the unit of payload is kg

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: `set_payload:[ok]`. If any error occurred, you will get `set_payload:error_message`.

29 set the acceleration of the robot

Socket string format: `set_acceleration(acc)`

example: `set_acceleration(50)`

the acceleration must be an integer, the unit of acceleration is mm/s^2 .

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: `set_acceleration:[ok]`. If any error occurred, you will get `set_acceleration:error_message`.

30 get the acceleration of the robot

Socket string format: `get_acceleration()`

example: `get_acceleration()`

the unit of acceleration is mm/s^2

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: `get_acceleration:50`.

31 assign variable

Socket string format: `assign_variable('variable_name',value)`

example: `assign_variable('A',10)` or `assign_variable('B',"ABC")`

the variable name need to be quoted with single quote mark('), if the value is a string, need to be quoted with double quotation marks("").

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: assign_variable:[ok]. If any error occurred, you will get assign_variable:error_message.

32 get the value of a variable

Socket string format: get_variable('variable_name')

example: get_variable('A',10)

the variable name need to be quoted with single quote mark('')

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, like this: get_variable:10. If any error occurred, you will get get_variable:error_message.

33 wait for command done

Socket string format: wait_command_done()

example: wait_command_done()

this function will wait until the previous command finish.

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: wait_command_done:0. If any error occurred, you will get get_variable:error_message.

34 pause the program

Socket string format: pause_program()

example: pause_program()

this function will pause the running program.

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: pause_program:[ok]. If any error occurred, you will get pause_program:error_message.

35 resume the program

Socket string format: resume_program()

example: resume_program()

this function will resume the paused program.

the return string is formatted in key-value pair, the key is the function name, the value is the value from robot, if success: resume_program:[ok]. If any error occurred, you will get resume_program:error_message.

5.2.3 Socket API Usage Example

1, Create Blank Program

As shown in Figure 5-86, after powering on the robot, by clicking "Write Program" and "Blank Program" successively, you can enter the program creation page.

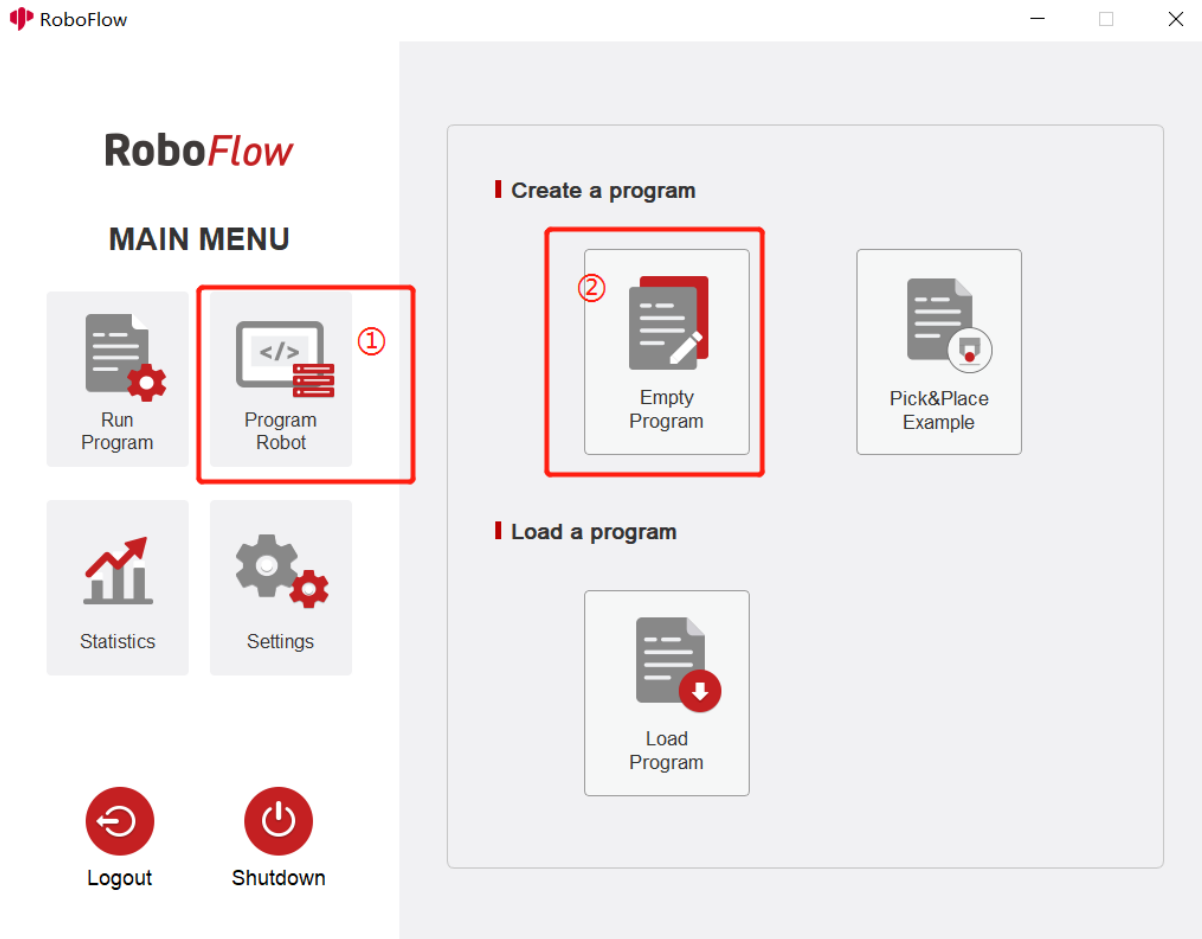


Fig. 5-86 Create Blank Program

2, Start to Monitor

As shown in Figure 5-87, by clicking "Tools", "Configuration", and "Network/Serial Port" in sequence, you can enter the API monitor page. You can enter the IP address of the TCP Server and set the port number, and you can run the ifcong command to view the IP address. You are advised to set the port number to 1024 to 49151.

Then by clicking the "Start" button, you can start to monitor the Socket API.

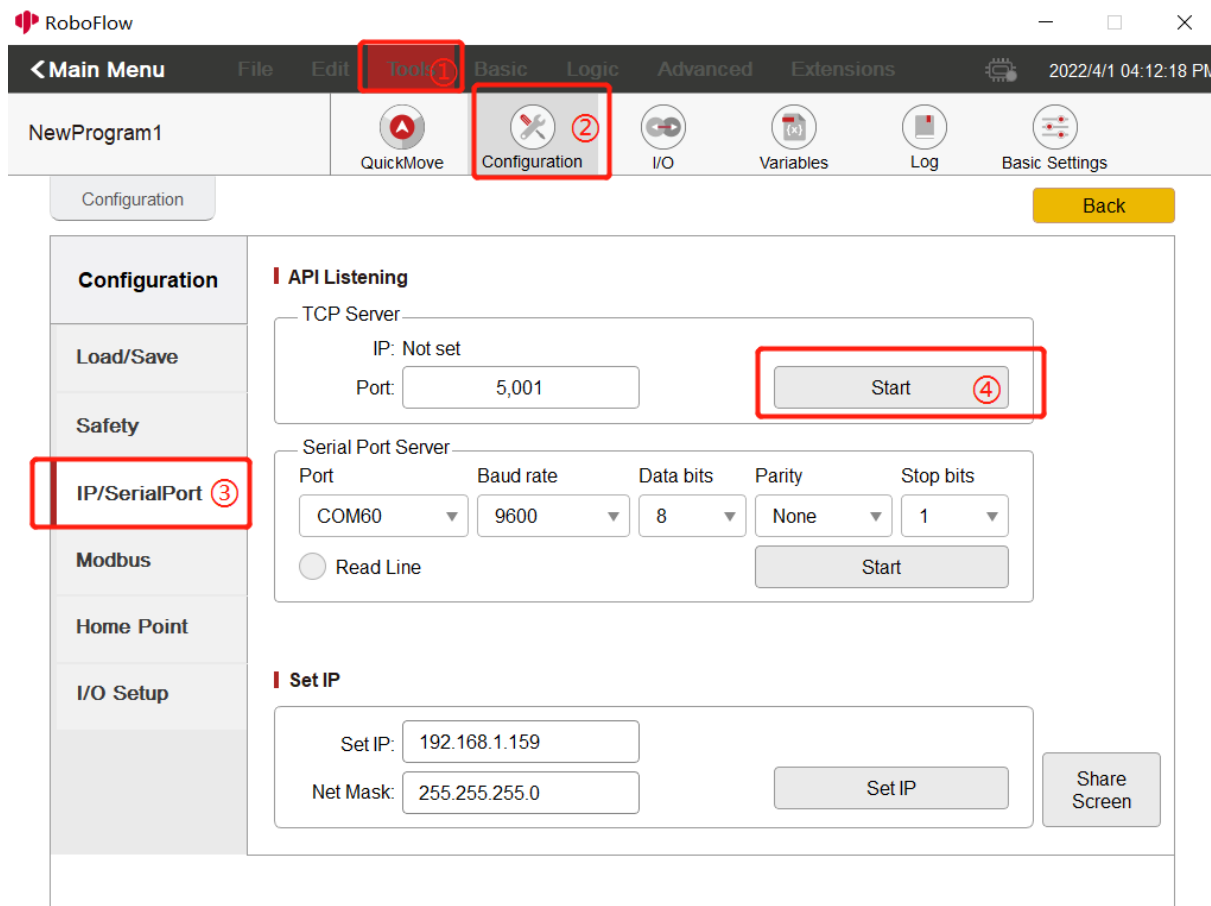


Fig. 5-87 Monitor Socket

3, Connecting Robot with PC

As shown in Figure 5-88, you can open the software "Sokit" on your PC terminal and click "Client" at the top of the software to enter the network setting page. The server address and port number can be the values set in RoboFlow and then you can click TCP connection to connect to the robot. If the connection is successful, information as shown in Figure 5-89 will be displayed.

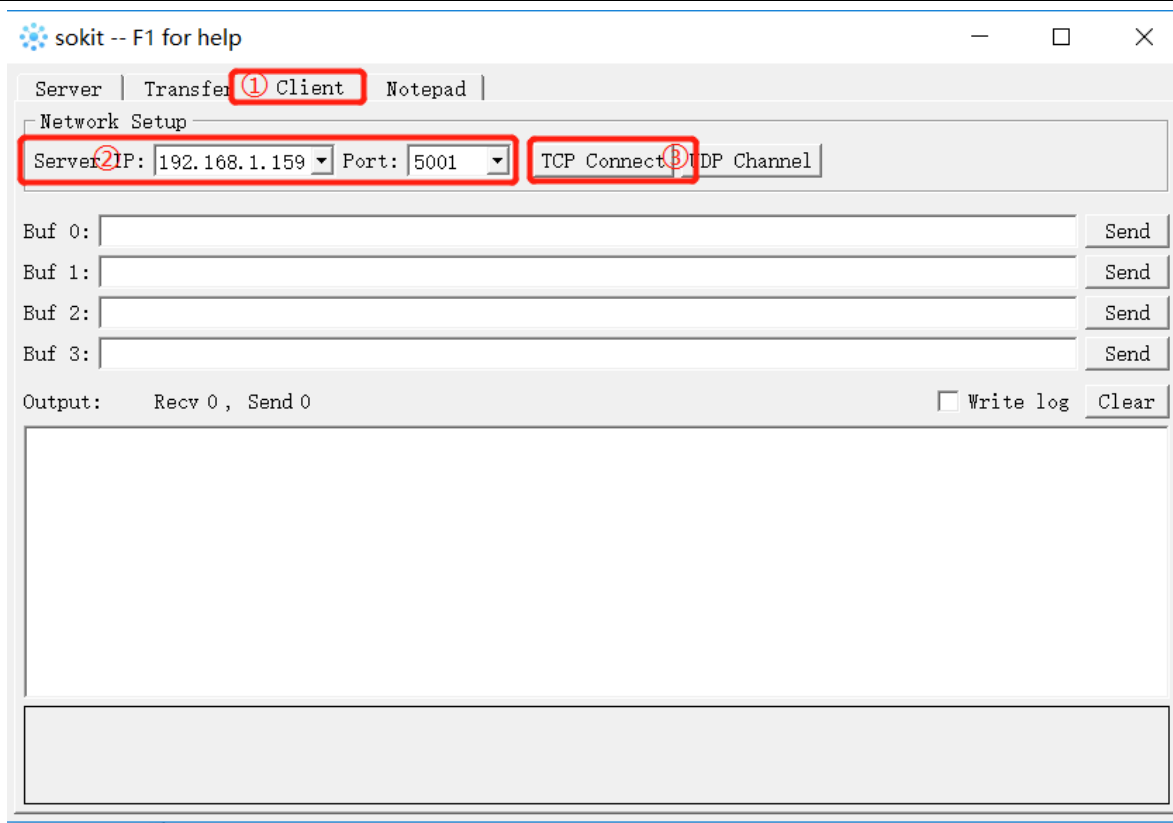


Fig. 5-88 Connect to RoboFlow

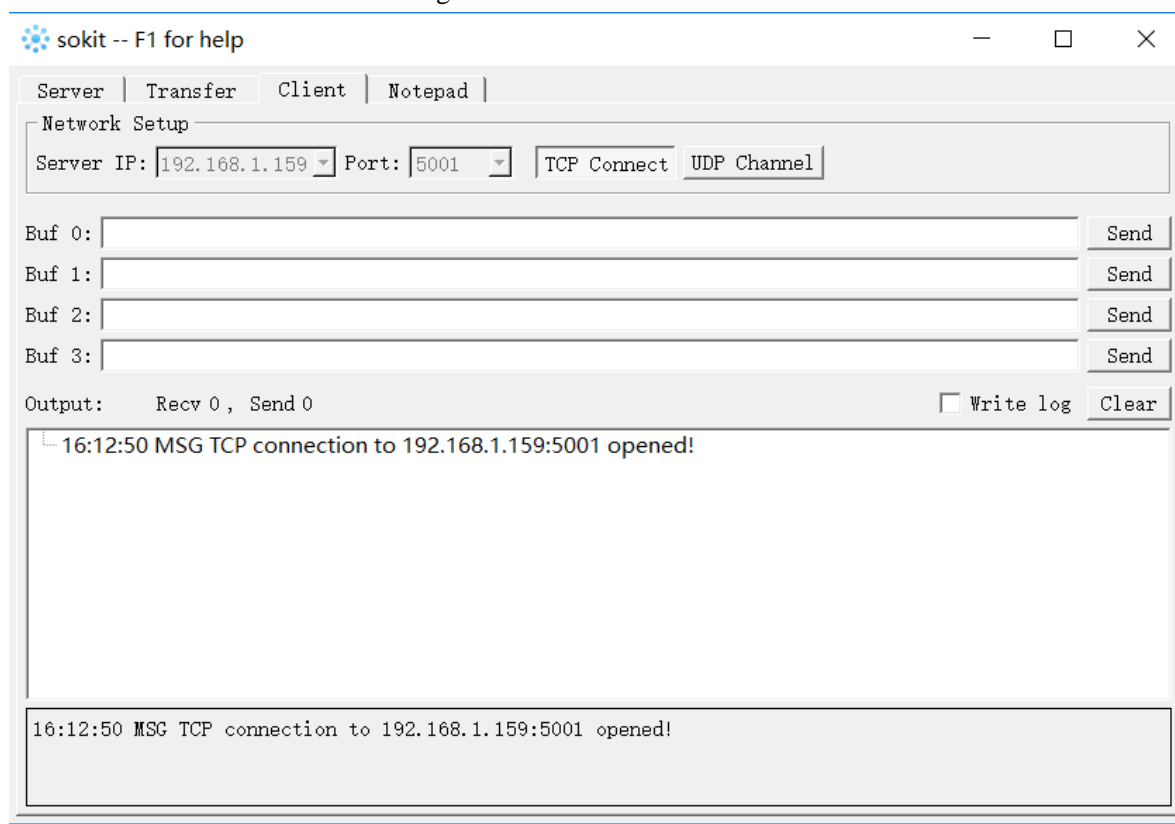


Fig. 5-89 Connected Successfully

4, Send Socket API

As shown in Figure 5-90, just by entering the Socket API that you want to send in the data window and clicking Send, the received information will be displayed in the sending and receiving records.

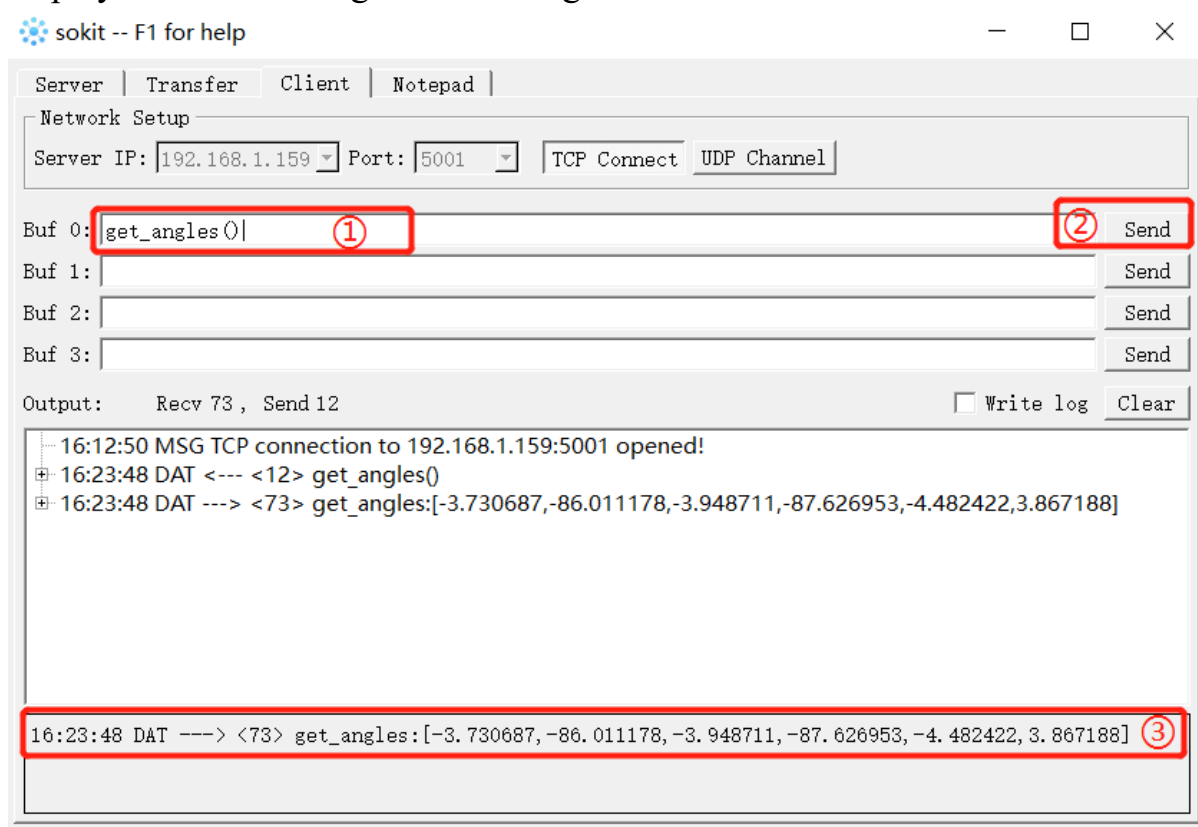


Fig. 5-90 Sent Successfully