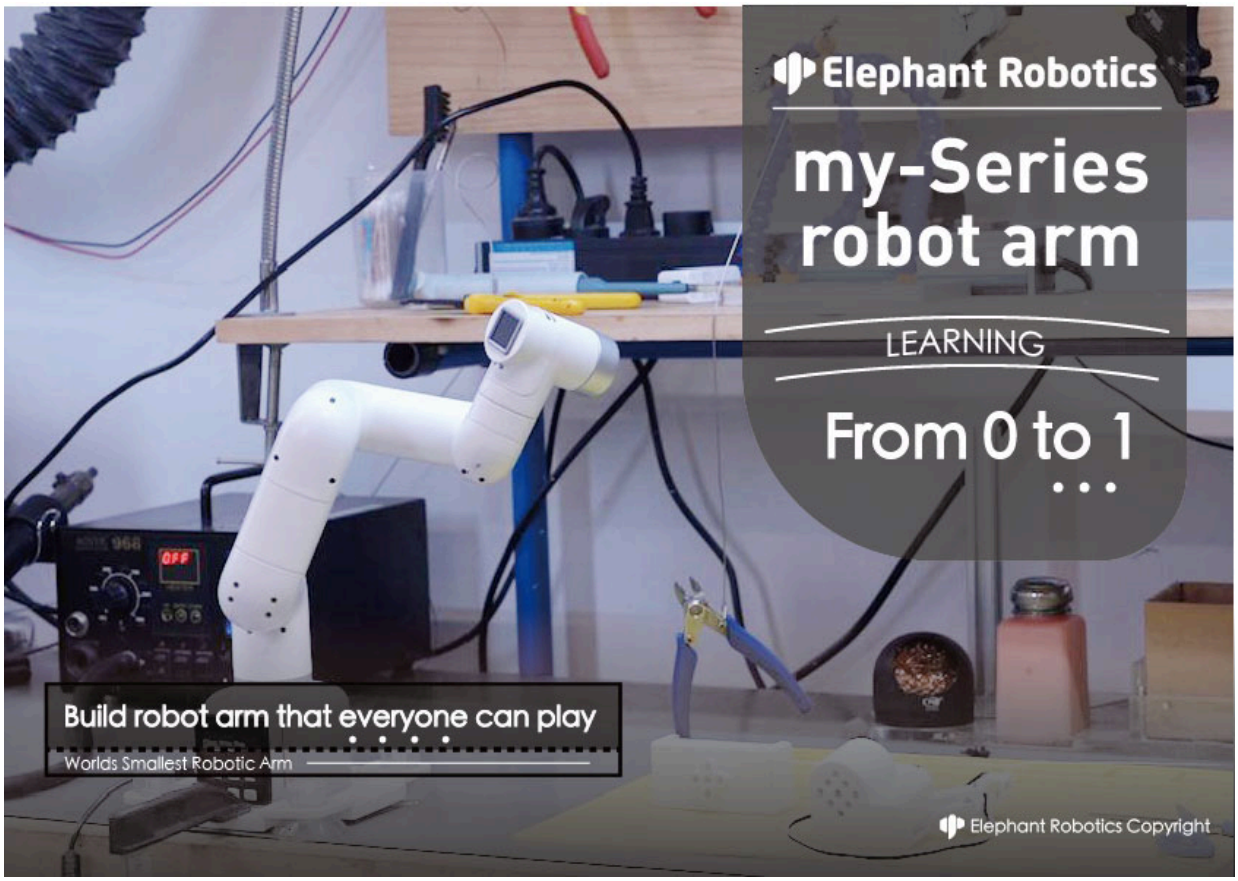
Table of Contents

myCobot: From 0 to 1



1.1 Why do we design myCobot

> **An entry-level collaborative robot arm** that everyone can learn and play

The original design of myCobot is to help friends who are interested in 6-axisseries robot to learn it from entry to master, creating unprecedented experience and teaching value.

What you can learn

> Robotics is based on rigid body kinematics and dynamics, but also an interdisciplinary subject that combines hardware, software, algorithm and control.

With myCobot, you can learn that

- **Hardware**
  - **Embedded Microcontroller Based on ESP32**
  - **Motor and Steering Gear**
  - **M5Stack Basic/ Atom**
- **Software**
  - **Arduino**
  - **C++**

    - **Python**
    - **ROS，MoveIt**
    - **Communication Data**
    - **Virtual Machines & Linux (visual system)**
- **Algorithm**
    - **Series Manipulator**
    - **Coordinate and Coordinate Transformation**
    - **DH Parameters**
    - **Kinematics**
    - **Manipulator Algorithm (e.g. dynamics)**
- **Machine Vision (Vision Set)**
    - **Color Recognition**
    - **Image Recognition**
    - **Hand-Eye Calibration**
    - **See and Grab**
- **Extended Applications**
    - **End-effector: gripper,suction pump, etc.**
    - **Robot Suit & Industry 4.0 Applications**

Parts of Gitbook

> View the directory on the left to jump

There are four major parts of Gitbook :

- **Introduction & Quick Start**
    - **Introduction** -- introduce what myCobot is and its main features, etc.
    - **How to Read** -- help you read Gitbook efficiently according to your learning level and knowledge background
    - **Use Cases** -- you can know exactly what use cases you can accomplish with
    - **Quick Start** -- learn the unboxing of your myCobot, and its first boot and use

- **Preparation before Development**
    - **Background Knowledge** -- learn about tools, industrial robots, algorithms, software, hardware,etc.
    - **Hardware Learning** -- learn about embedded hardware, structural components, electronic components, etc.
    - **Purpose of Use** -- identify the purpose you want to use it for, and complete the study related to your task

- **Development and Use**
    - **Development Environment** -- learn to use Arduino, ROS, uiFlow, roboFlow, python and others development environment to develop myCobot
    - **Accessories** -- learn to use myCobot with different accessories, such us bases, grippers, suction pumps and so on
    - **Machine Vision** -- learn to control myCobot under the guidance of machine vision
    - **Robot Modification** -- learn how to modificate myCobot into a 4 or 5 axis manipulator

- **myCobot Suit**
  - **Intelligent Warehouse:** learn how to use myCobot to carry different objects
  - **Artificial Intelligence:** learn how to control myCobot to grasp objects intelligently under the guidance of machine vision
  - **Industry 4.0:** learn how to grasp and place objects intelligently by simulating production line

Information Source

- **Official website:** www.elephantrobotics.com
- **Tutorial video:** https://www.youtube.com/channel/UC68l2RaRF2Mp8fzpCTzNBfA
- **Shop website:** https://shop.elephantrobotics.com
- Download PDF

Contact Us

> **If you have any other questions, you can contact us as follows.**
>
> **We will answer you as soon as possible (working day 9:30-18:30)**

- Twitter: myCobot Official\@CobotMy
- Facebook: https://www.facebook.com/MyCobot-116558893805177
- Mail: support@elephantrobotics.com

Product Display

1 Product Introduction

myCobot Pro force-controlled gripper is a high-performance robot end effector designed for multi-functional grasping needs. Made of PC and PBT materials, it is manufactured through precision injection molding to ensure the firmness and durability of the product. The gripping range of the gripper is 0-100 mm (default fingertip), and it supports multi-speed torque adjustment to meet the grasping needs of different forces. The rated load is 500 grams, and the repeatability accuracy is 0.5 mm, which can adapt to various robot grasping operation scenarios.

2 Applicable models

ER myCobot 320 series

ER Mercury A/B/X series

ER myCobot Pro 630

Other general robots

3 Applicable scenarios

Experimental operation: In scientific research experiments, complete the grasping and moving of test tubes, utensils, etc. to ensure the safety and accuracy of the experiment.

Educational demonstration: As a teaching tool, it helps students understand the principles of robot grasping and cultivate practical skills.

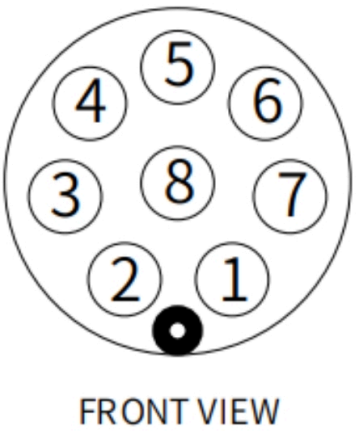
Material handling: In simulated production lines or warehouses, materials of various specifications are handled to improve work efficiency.

Product Features

1 Product Specifications

Name	myGripper F100 force-controlled gripper
Material	PC, PBT
Dimensions	156X106X61mm
Process technology	Injection molding
Gripping range	0-100 mm (default fingertip)
Repeatability	0.5 mm
Service life	300,000 openings and closings
Drive mode	Electric drive
Transmission mode	Gear + connecting rod
Weight	340 g
Rated load	500g
Working voltage	24V
Fixing method	Screw fixing
Environment requirements	Normal temperature and pressure
Control interface	RS485/IO control/button control
Cable interface model	M8-8PIN

Pin sequence description



编号	信号	解释	配套M8线颜色
1	GND	DC24V 负极	白
2	IN1	工具输出接口1	褐
3	IN2	工具输出接口2	绿
4	485A	485标准接口A	黄
5	24V	DC24V 正极	灰
6	OUT1	工具输入接口1	粉
7	OUT2	具输入接口2	蓝
8	485B	485标准接口B	紫

Numbers 1 and 5 connect GND and 24V, numbers 2 and 3 are for controlling IO input, numbers 6 and 7 are for IO output, and numbers 4 and 8 are for 485 communication, which is to receive and send instructions with the gripper

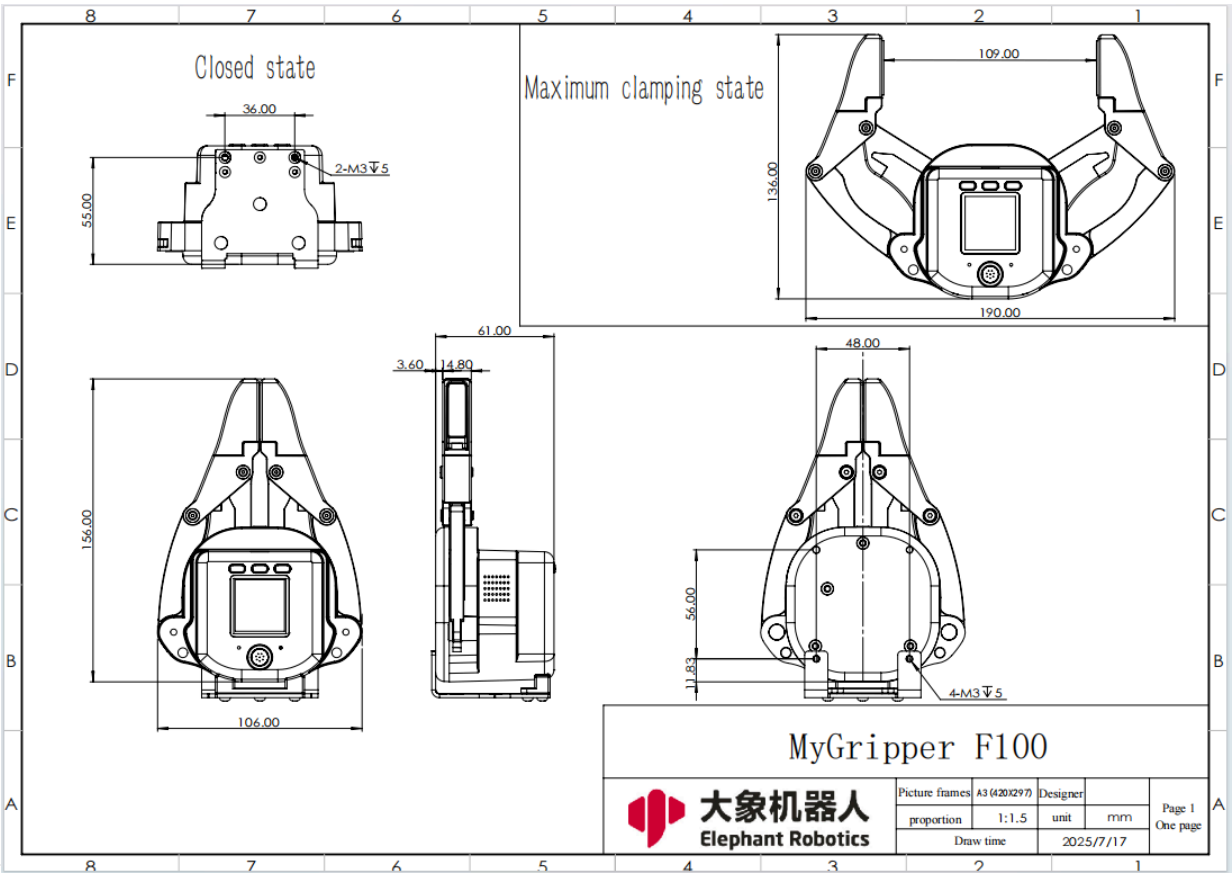
Notes:

Please distinguish the line sequence according to the line mark. If the line mark is lost, detached, or forgotten, please contact our staff to cooperate in determining the line sequence. If you do not contact our staff, you will be responsible for the consequences of the damage to the gripper due to the wrong wiring sequence.

2 Main controller specification parameter table

Name	ESP32
Core parameters	240MHz dual core. 600 DMIPS, 520KB SRAM. Wi-Fi, dual mode Bluetooth
Flash	4MB

3 Structural parameters



User Notice

1 Introduction

This chapter details general safety information for personnel who perform installation, maintenance, and repair work on the Elephant robot. Please fully read and understand the contents and precautions in this chapter before handling, installing, and using it.

2 Hazard Identification

The safety of collaborative robots is based on the correct configuration and use of the robots. Moreover, even if all safety instructions are followed, harm or injury to the operator may still occur. Therefore, it is very important to understand the safety hazards of robot use, which is conducive to preventing them before they occur. The following tables 1-1~3 are common safety hazards that may exist in the context of using robots:

Table 1-1 Dangerous safety hazards


1. Personal injury or robot damage caused by incorrect operation during robot handling.
2. Failure to fix the robot as required, such as lack of screws or screws not tightened, insufficient base locking capacity to stably support the robot for high-speed movement, etc., causing the robot to fall over and cause personal injury or robot damage.
3. Failure to configure the correct safety function of the robot, or insufficient installation of safety protection tools, etc., causing the robot's safety function to fail to function, thereby causing danger.

Table 1-2 Warning-level safety hazards


1. Do not stay in the robot's range of motion when debugging the program. Improper safety configuration may not be able to avoid collisions that may cause personal injury.
2. Connecting the robot to other equipment may cause new dangers, and a comprehensive risk assessment needs to be re-performed.
3. Scratches and punctures caused by sharp surfaces such as other equipment or the robot's end effector in the working environment.
4. The robot is a precision machine, and stepping on it may cause damage to the robot.
5. Failure to remove the clamped object before clamping it in place or turning off the robot power or air source (not confirming whether the end effector is firm and the clamped object falls due to loss of power) may cause dangers, such as damage to the end effector and injuries to people.
6. The robot has the risk of accidental movement. Do not stand under any axis of the robot under any circumstances!
7. The robot is a precision machine. If it is not placed stably during transportation, it may cause vibration and damage to the internal parts of the robot.
8. Compared with ordinary mechanical equipment, the robot has more degrees of freedom and a larger range of motion. Failure to meet the range of motion may cause unexpected collisions.

Table 1-3 Safety hazards that may cause electric shock


1. Using non-original cables may cause unknown dangers.
2. Electrical equipment contacting liquids may cause leakage hazards.
3. There may be a risk of electric shock when the electrical connection is wrong.
4. Please be sure to turn off the power of the controller and related devices and unplug the power plug before replacing. If the operation is carried out while the power is on, it may cause electric shock or malfunction.

3 Precautions

The following rules should be followed when using the force-controlled gripper:

- Please distinguish the line sequence according to the line mark. If the line mark is lost, detached, or forgotten, please contact our staff to cooperate in determining the line sequence. If you do not contact our staff, **you will**

be responsible for the damage to the dexterous hand due to the wrong line sequence

- Please do not burn other product drivers without authorization, or use unofficial recommended methods to burn firmware. **If the device is damaged due to the user's personal burning of other firmware, it will not be covered by after-sales service.**

If you have any questions or suggestions about the content, you can log in to the official website of Elephant Robotics to submit relevant information:

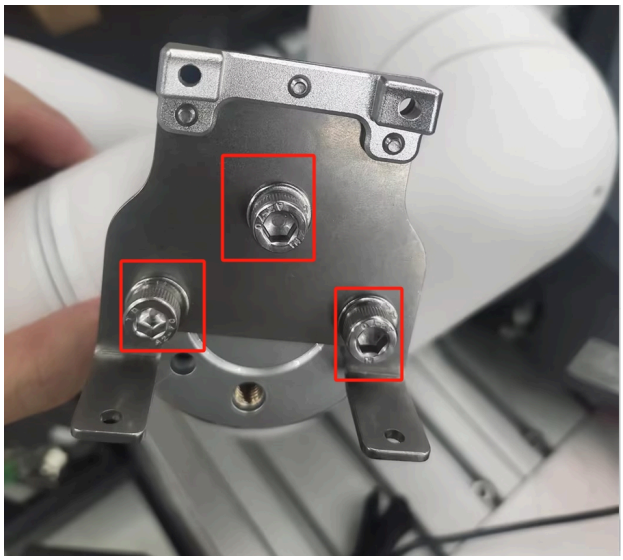
https://www.elephantrobotics.com

First installation and use

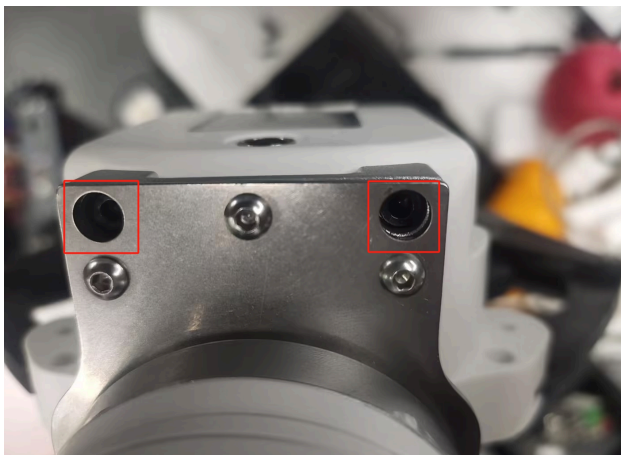
1 Direct connection between the gripper and the end of the robot

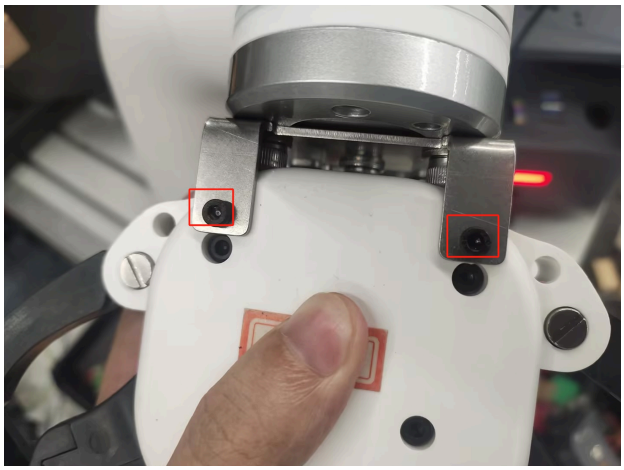
Note: Only the ER myCobot 320 series, ER Mercury A/B/X series, ER myCobot Pro 630, and ER myCobot Pro 600 can be directly connected to the gripper through the M8 aviation plug line. For other manufacturers' robots, refer to the wiring sequence of the gripper to rewire

Take mycobot320 as an example, other models can refer to this step for installation Use screws and gaskets to install the gripper connector to the end flange of the robot



Then use screws to install the gripper on the connector





Finally, use the M8 aviation line to connect the gripper and the robot

2 Gripper and USB485 module connection

USB-485 module wiring:

Connect 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) at the gripper end, a total of 4 wires, the power supply is a 24V DC regulated power supply, insert the module's USB port into the computer's USB port

485A connected to 485 to USB module A+;

485B connected to 485 to USB module B-;

24V connected to 24V DC regulated power supply positive pole;

GND connected to 24V Negative pole of DC regulated power supply

Screen control

The myCobot Pro force-controlled gripper has a screen display function, and there are buttons under the screen for easy operation of the screen. By selecting the interface, you can more conveniently control the gripper and understand the gripper information. Screen information and Overall function description

Interface	Function
Main interface	View the gripper information, such as: Position Set speed Current Input IO Output IO
Setting interface	View and set the gripper data
Set baud rate interface (Modbus Config)	View and change the device baud rate
IO enable interface (IO Config)	View and enable IO, the default IO is enabled
Gripper control interface (Manual Config)	Open-control the gripper to open Close-control the gripper to close Release-disable the gripper (the gripper will not have torque at this time, and the gripper can swing freely) Hold - enable the gripper (enable the gripper, and the gripper has torque at this time)
Version information interface (Information)	Firmware - Gripper Version Number and Gripper ID Servomotor - Servo Version Number and Servo ID

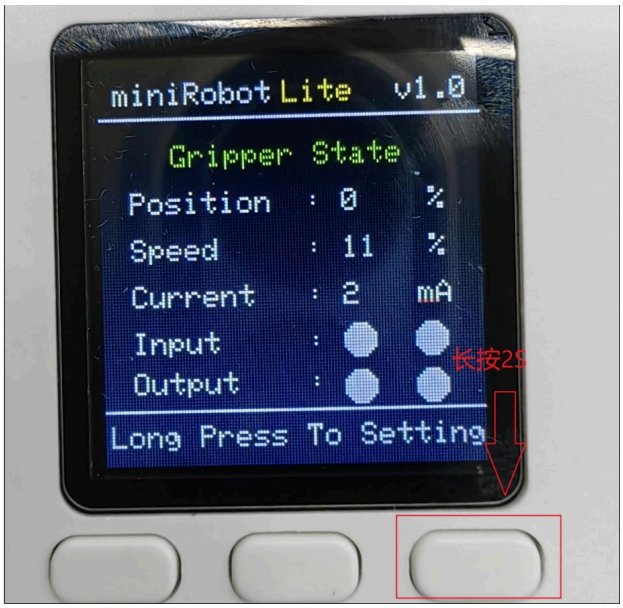
Main interface

Entering the gripper main interface, we can see some real-time information of the gripper on the screen, which makes it easier to know the gripper information. The main interface mainly has the following information as shown in the figure

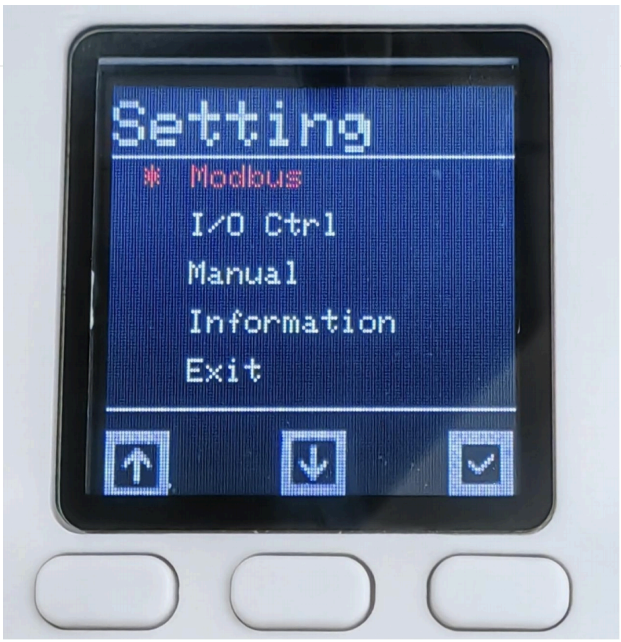
The main interface displays the gripper angle, speed, current, input and output IO level in real time

Setting interface

Press and hold the confirmation button for 2s in the main interface to enter the main interface

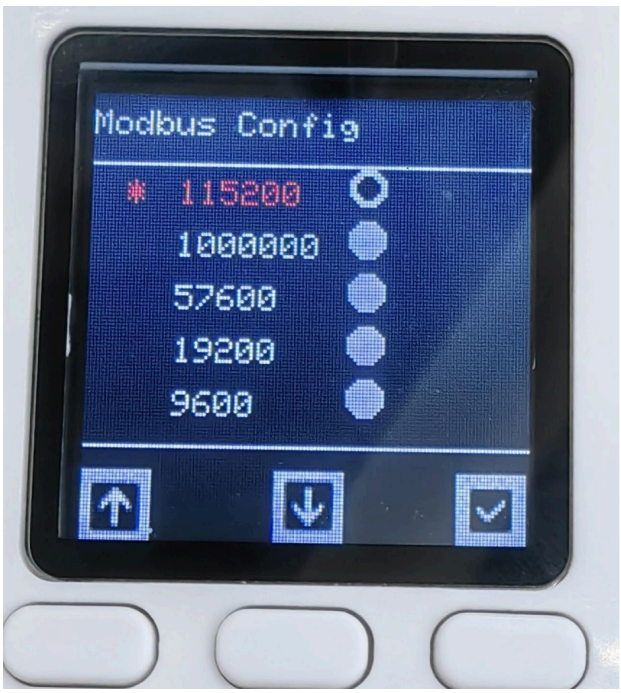


In the setting interface, you can move up and down and select by pressing the button



Baud rate setting

Click Modbus in the setting interface to enter the Modbud Config interface. Click Baudrate to enter the interface, select the baud rate you want to set and click Confirm. When the color of the origin next to the baud rate changes, it proves that the setting is successful



After setting, find Exit and click to exit the current interface

IO Enable settings

Select I/O Ctrl in the settings interface to enter the I/O Config interface. By default, IO is enabled. If it does not work, use IO control. You can select Disable to turn off IO control

Manual interface

Select Manual in the settings interface to enter the Manual Control interface to quickly and conveniently control the gripper

Select Quick Move to enter the gripper control interface, as shown in Figure 4.7. Select Gripper Init to enter the gripper configuration interface

Open-set the gripper to open, Close-set the gripper to close, Pelease-set the gripper to disable, Hold-set the gripper to enable, Exit-exit the current interface

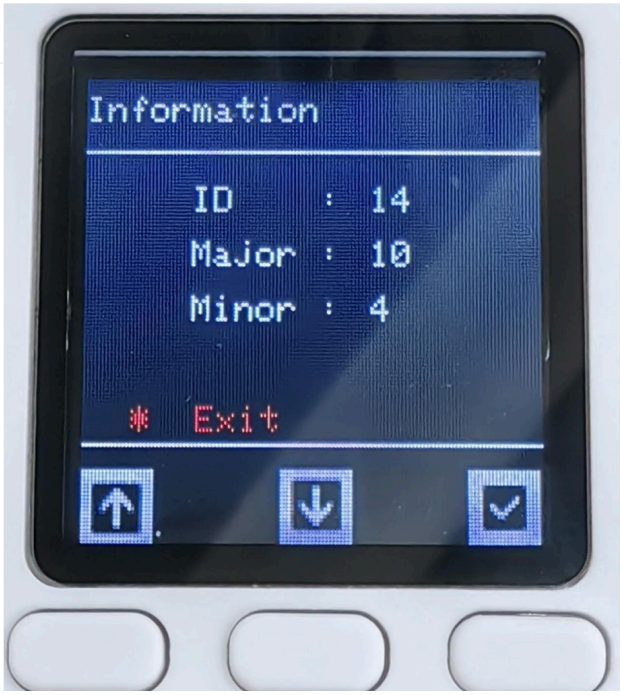
Select Auto Init to set the zero position of the gripper

Information interface

By selecting Information to enter the interface, we can see two options, Firmware and Servomotor. The former is the firmware version number. Click it to view the current firmware major and minor version numbers. The latter is to view the click information. Click it to see the ID and the main and minor versions of the motor.

Click Firmware to view the firmware version and the ID in the device ID command

Click Servomotor to view the motor version and ID. The ID cannot be modified.



IO control

According to the above pin sequence description, find the line corresponding to OUT1 OUT2, input 24V, and when a high and a low level are input, the gripper will move. On the contrary, when a high and a low level are given, the gripper will move in the opposite direction. The IO input status can be observed on the main screen interface. When there is no IO input, it will appear white, and when there is IO input, it will appear black.

Customized protocol control method

USB-485 module wiring:

Connect 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) at the gripper end, a total of 4 wires, the power supply is a 24V DC regulated power supply, insert the module's USB port into the computer's USB port

485A connected to 485 to USB module A+;

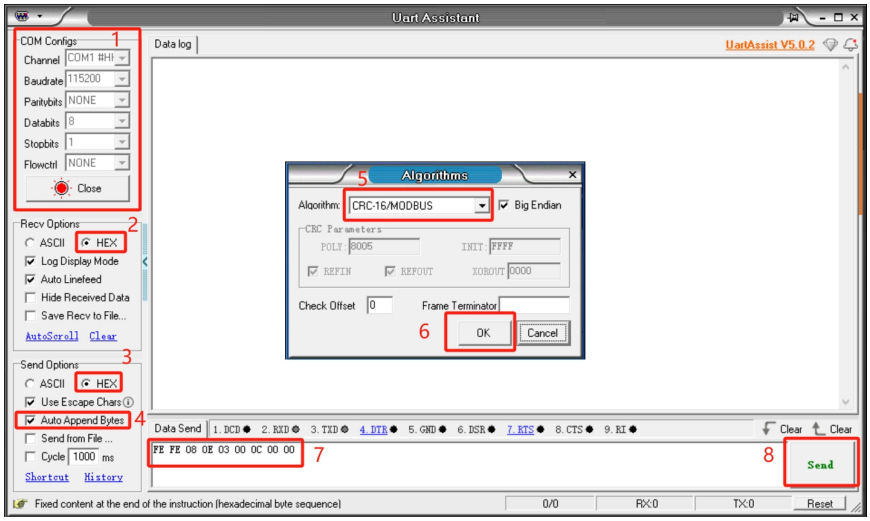
485B connected to 485 to USB module B-;

24V connected to 24V DC regulated power supply positive pole;

GND connected to 24V Negative pole of DC regulated power supply

Serial port debugging assistant debugging:

Users can use UartAssist serial port debugging assistant, refer to the figure below to send the corresponding gripper command, CRC checksum does not need to be filled in, UartAssist serial port debugging assistant will automatically generate it.



Default configuration of the gripper serial port

- Gripper ID: 14
- Baud rate: 115200
- Data bit: 8
- Stop bit: 1
- Check bit: No check bit

Protocol usage instructions

The gripper uses a custom protocol. An instruction consists of a frame header (2byte), length (1byte), address code ID (1byte), function code (1byte), register address (2byte), register data (2byte), and check code (2byte). Let's take reading the gripper angle as an example.

Frame header	Length	ID	Function code	Register address	Register data/parameter	CRC-16MODBUS check code
Fe Fe	08	0e	03	00 0C	00 00	B1 C0

Frame header: 254 254

Length: Instruction length 08

ID: 0E, can be modified in the device, the default ID is 14, 0E represents the current ID of the gripper is 14

Function code: Identify whether the instruction is a setting or acquisition function, 06 (write operation to the register)/03 (read operation to the register).

Register address: 00 0c Gripper function corresponding register address

Register parameter: 00 00, no parameters are required in the read instruction, 00 00 can be given, and data is written to the gripper address for the parameter in the setting instruction.

CRC check: B1 C0, to ensure that the terminal device does not respond to data that has changed during the transmission process, to ensure the security and efficiency of the system, the CRC check adopts the 16-bit cyclic remainder method. Verify all hexadecimal values of frame header, length, function code, register address and register parameters directly to get B1 C0 verification code.

Command Overview

- Read the firmware major version number

  Command: fe fe 08 0e 03 00 01 00 00 72 51

  Function code: 03 Read operation

  Parameter: None

  Return: fe fe 08 0e 03 00 01 00 01 B2 90

  Note: Data return 00 01, return version number is 1
- Read the firmware minor version number

  Command: FE FE 08 0E 03 00 02 00 00 72 A1

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 02 00 01 B2 60

  Note: Data return 00 01, indicating that the return version number is 1
- Set/read device ID number
  - Set

    Command: FE FE 08 0E 06 00 03 00 0E 76 BD

    Function code: 06 write operation

    Parameter: 00 0E, ID setting range (1-254)

Return: FE FE 08 0E 06 00 03 00 01 72 FD

Note: Success returns 00 01, failure returns 00 00, after the device ID is modified, the ID in the instruction also needs to be modified to be the same to communicate

- Read

  Instruction: FE FE 08 0E 03 00 04 00 00 73 41

  Function code: 03 read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 04 00 0E B7 C0

  Note: Data return 00 0E, indicating the current gripper ID number

- Set/read 485 baud rate
  - Set

    Instruction: FE FE 08 0E 06 00 05 00 00 73 1D

    Function code: 06 write operation

    Parameter: parameter 00 00, 0-115200, 1-1000000, 2-57600, 3-19200, 4-9600, 5-4800, If set to 1000000, the parameter is changed to 00 01

    Return: FE FE 08 0E 06 00 05 00 01 72 FD

    Note: Success returns 00 01, failure returns 00 00

  - Read

    Command: FE FE 08 0E 03 00 06 00 00 B3 E0

    Function code: 03 read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 06 00 00 B3 E0

    Note: Return data 00 00, the baud rate value corresponding to the setting parameter

- Set the gripper enable state

  Command: FE FE 08 0E 06 00 0A 00 00 B0 EC

  Function code: 06 write operation

  Parameter: 00 00, 00 means disconnect enable, 00 01 means enable

  Return: FE FE 08 0E 06 00 0A 00 01 70 2D

  Note: Success returns 00 01, failure returns 00 00

- Set/read the gripper angle
  - Set

    Command: FE FE 08 0E 06 00 0B 00 64 9B BC

    Function code: 06 write operation

    Parameter: Parameter 00 64, set the angle to fully open

    Return: FE FE 08 0E 06 00 0B 00 01 B0 7C

Mark: Success returns 00 01, failure returns 00 00

- Read

  Command: FE FE 08 0E 03 00 0C 00 00 B1 C0

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 0C 00 64 5A C1

  Mark: Return data 00 64, indicating that the current angle is 100, fully open state

- Set the gripper zero position

  Command: FE FE 08 0E 06 00 0D 00 00 71 5D

  Function code: 06 Write operation

  Parameter: None

  Return: FE FE 08 0E 06 00 0D 00 01 B1 9C

  Mark: Success returns 00 01, failure returns 00 00

  Note: When setting the zero position, the gripper will move by itself. If there is an object blocking the movement, the setting will fail. Please check whether there are any obstacles around the gripper before setting.

- Read the gripper clamping status

  Command: FE FE 08 0E 03 00 0E 00 00 71 61

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 0E 00 01 B1 A0 Mark: 00 01, return data 0 is moving, 1 is stopped and no clamping is detected, 2 is stopped and clamping is detected, 3 is detected and the object falls

- Set/read the gripper P value

  - Set

    Command: FE FE 08 0E 06 00 0F 00 64 5A FD

    Function code: 06 Write operation

    Parameter: Parameter 00 64, set the P value 100, setting range (1-150)

    Return: FE FE 08 0E 06 00 0F 00 01 71 3D

    Note: Success returns 00 01, failure returns 00 00

  - Read

    Command: FE FE 08 0E 03 00 10 00 00 77 01

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 10 00 64 9C 00

    Note: Return data 00 64, indicating that the current P value is 100

- Set/read the gripper D value

- Set

  Command: FE FE 08 0E 06 00 11 00 64 5C 9D

  Function code: 06 Write operation

  Parameter: Parameter 00 64, set the D value 100, setting range (1-150)

  Return: FE FE 08 0E 06 00 11 00 01 77 5D

  Note: Success returns 00 01, failure returns 00 00

  If the gripper shakes, the D value can be appropriately increased

- Read

  Command: FE FE 08 0E 03 00 12 00 00 B7 A0

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 12 00 64 5C A1

  Note: Return data 00 64, indicating that the current D value is 100

- Set/read gripper I value
  - Set

    Command: FE FE 08 0E 06 00 13 00 00 77 3D

    Function code: 06 Write operation

    Parameter: Parameter 00 00, set I value to 0, setting range (1-150)

    Return: FE FE 08 0E 06 00 13 00 01 B7 FC

    Note: Success returns 00 01, failure returns 00 00

  - Read Command: FE FE 08 0E 03 00 14 00 00 B6 40

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 14 00 00 B6 40

    Note: Return data 00 00, indicating that the current I value is 0

- Set/read the clockwise runnable error of the gripper
  - Set

    Command: FE FE 08 0E 06 00 15 00 01 B6 1C

    Function code: 06 Write operation

    Parameter: Parameter 00 01, setting value is 1, setting range (0-16)

    Return: FE FE 08 0E 06 00 15 00 01 B6 1C

    Note: Success returns 00 01, failure returns 00 00

  - Read

    Command: FE FE 08 0E 03 00 16 00 00 76 E1

Function code: 03 Read operation

Parameter: None

Return: FE FE 08 0E 03 00 16 00 01 B6 20

Note: Return data 00 01, indicating that the current value is 1

- Set/read the clockwise runnable error of the gripper
  - Set

    Command: FE FE 08 0E 06 00 17 00 01 76 BD

    Function code: 06 Write operation

    Parameter: Parameter 00 01, setting value is 1, setting range (0-16)

    Return: FE FE 08 0E 06 00 17 00 01 76 BD

    Note: Success returns 00 01, failure returns 00 00
  - Read

    Command: FE FE 08 0E 03 00 18 00 00 B5 80

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 18 00 01 75 41

    Note: Return data 00 01, indicating that the current value is 1
- Set/read the minimum starting force of the gripper
  - Set

    Command: FE FE 08 0E 06 00 19 00 18 7F 1D

    Function code: 06 Write operation

    Parameter: 00 18

    Return: FE FE 08 0E 06 00 19 00 01 B5 DC

    Note: Success returns 00 01, failure returns 00 00
  - Read

    Command: FE FE 08 0E 03 00 1A 00 00 75 21

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 1A 00 18 7F 21

    Note: Return data 00 18, indicating that the current value is 24
- Set/read torque
  - Set

    Command: FE FE 08 0E 06 00 1B 00 64 F8 BC

    Function code: 06 Write operation

Parameter: 00 64 Set the torque to 100, parameter range (0-100)

Return: FE FE 08 0E 06 00 1B 00 01 75 7D

Note: Success returns 00 01, failure returns 00 00

- Read

  Command: FE FE 08 0E 03 00 1C 00 00 74 C1

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 1C 01 2C 39 C1

  Note: Return data 01 2C, indicating that the current value is 300, the greater the torque, the greater the current, and the gripper strength will also increase

- io output setting

  Command: FE FE 08 0E 06 00 1D 00 10 78 5D

  Function code: 06 Write operation

  Parameter: 00 10 Set IO output is 10, parameter range (00, 01, 10, 11)

  Return: FE FE 08 0E 06 00 1D 00 01 74 9D

  Note: Success returns 00 01, failure returns 00 00

- Set io opening angle

  - Set

    Command: FE FE 08 0E 06 00 1E 00 32 61 2D

    Function code: 06 Write operation

    Parameter: 00 32 Set IO opening angle to 50, parameter range (0-100)

    Return: FE FE 08 0E 06 00 1E 00 01 74 6D

    Note: Success returns 00 01, failure returns 00 00

  - Read

    Command: FE FE 08 0E 03 00 22 00 00 B8 A0

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 22 00 32 6D 21

    Note: 00 32 Data return is 50 degrees

- Set io closing angle

  - Set

    Command: FE FE 08 0E 06 00 1F 00 00 74 FD

    Function code: 06 Write operation

    Parameter: 00 32 Set IO closing angle to 0, parameter range (0-100)

    Return: FE FE 08 0E 06 00 1F 00 01 B4 3C

Note: Success returns 00 01, failure returns 00 00

- Read

  Command: FE FE 08 0E 03 00 23 00 00 78 F1

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 23 00 00 78 F1

  Note: 00 00 data return is 0 degrees

- Set/read the gripper speed
  - Set

    Command: FE FE 08 0E 06 00 20 00 32 AD 4C

    Function code: 06 Write operation

    Parameter: 00 32 Set the speed to 50, parameter range (1-100)

    Return: FE FE 08 0E 06 00 20 00 01 B8 0C

    Note: Success returns 00 01, failure returns 00 00

  - Read

    Command: FE FE 08 0E 03 00 21 00 00 B8 50

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 21 00 32 6D D1

    Note: 00 32 data return is 50

- Set absolute angle Command: FE FE 08 0E 06 00 24 00 64 52 8D

  Function code: 06 Write operation

  Parameter: 00 64 Set absolute angle to 100, parameter range (0-100)

  Return: FE FE 08 0E 06 00 24 00 01 79 4D

  Note: Success returns 00 01, failure returns 00 00 **Due to the long response time, if the user keeps sending this command, it will be put into the queue and executed one by one. The absolute angle will wait for the gripper to move to the specified position before returning the command. If there is an object blocking the movement during the movement, the gripper cannot move to the specified position and the failure command will be returned after the waiting timeout**

- Pause movement

  Command: FE FE 08 0E 06 00 25 00 00 79 DD

  Function code: 06 Write operation

  Parameter: None

  Return: FE FE 08 0E 06 00 25 00 01 B9 1C

  Mark: Success returns 00 01, failure returns 00 00

Pause motion acts on setting absolute angle. After sending this command, the absolute angle will not be executed from the queue Command

- Resume motion

  Command: FE FE 08 0E 06 00 26 00 00 79 2D

  Function code: 06 Write operation

  Parameter: None

  Return: FE FE 08 0E 06 00 26 00 01 B9 EC

  Mark: Success returns 00 01, failure returns 00 00 **Resume motion acts on setting absolute angle. This command can resume the execution of the absolute angle queue**

- Stop motion

  Command: FE FE 08 0E 06 00 27 00 00 B9 7C

  Function code: 06 Write operation

  Parameter: None

  Return: FE FE 08 0E 06 00 27 00 01 79 BD

  Note: Success returns 00 01, failure returns 00 00 **Stop motion acts on the absolute angle, this instruction will clear the absolute instruction queue**

- Get the amount of data in the current queue

  Instruction: FE FE 08 0E 03 00 28 00 00 BA 80

  Function code: 03 Read operation

  Parameter: None

  Return: FE FE 08 0E 03 00 28 00 00 BA 80

  Note: 00 00, indicating that the number of instructions in the absolute angle queue is 0 **This instruction acts on setting the absolute angle, and this instruction can get the number of instructions in the absolute angle queue**

- Set the virtual position value of the servo
  - Set

    Command: FE FE 08 0E 06 00 29 00 08 BC 1C

    Function code: 06 Write operation

    Parameter: 00 08 Set the gripper position error range, parameter range (0-254)

    Return: FE FE 08 0E 06 00 29 00 01 BA DC

    Note: Success returns 00 01, failure returns 00 00

  - Read

    Command: FE FE 08 0E 03 00 2A 00 00 7A 21

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 2A 00 08 BC 20

Note: 00 08 Data return 8 **The default value is 8. The larger the virtual position setting, the lower the accuracy of the gripper position determination**

- Set/read gripping current
  - Set

    Command: FE FE 08 0E 06 00 2B 00 FE 3A 3D

    Function code: 06 Write operation

    Parameter: 00 FE Set gripping current to 254, parameter range (0-254)

    Return: FE FE 08 0E 06 00 2B 00 01 7A 7D

    Note: Success returns 00 01, failure returns 00 00
  - Read

    Command: FE FE 08 0E 03 00 2C 00 00 7B C1

    Function code: 03 Read operation

    Parameter: None

    Return: FE FE 08 0E 03 00 2C 00 FE FB 40

    Note: 00 FE Data return 254, when setting the torque, the current of the clamped object will adapt accordingly. This command can set the clamping current by yourself.

modbus-rtu protocol control method

Currently, the ModBus-RTU protocol of the force control gripper only supports function codes 03 and 06. The force control gripper is a ModBus-RTU slave, and the register length of the read function code 03 is only 1. The function code 03 represents read, and the function code 06 represents write

USB-485 module wiring:

Connect the 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) of the gripper end, a total of 4 wires, the power supply is a 24V DC regulated power supply, and insert the USB port of the module into the USB port of the computer

485A is connected to the 485 to USB module A+;

485B is connected to the 485 to USB module B-;

24V is connected to the positive pole of the 24V DC regulated power supply;

GND is connected to the negative pole of the 24V DC regulated power supply

Read the holding register (function code 0x03)

Take the example of reading the jaw opening angle

Master station request frame format:

Slave station address	Function code	Register address (high bit)	Register address (low bit)	Number of registers (high bit)	Number of registers (low bit)	CRC16 check high bit	CRC16 check low bit
0x0E	0x03	0x00	0x0C	0x00	0x01	crc1	crc2

Slave station response frame format: | Slave station address | Function code | Number of bytes | Register (high bit) | Register (low bit) | CRC16 check high bit |CRC16 check low bit| | -------- | ------ | -------- | ---------- | ---------- | ----- |----- | | 0x11 | 0x03 | 0x02 | 0x00 | 0x64 | crc1 | crc2|

Set the holding register (function code 0x06)

Set the jaw opening angle as an example Master request frame format:

Slave address	Function code	Start register (high bit)	Start register (low bit)	Data content (high bit)	Data content (low bit)	CRC16 check high bit	CRC16 check low bit
0x0E	0x06	0x00	0x0B	0x00	0x64	crc1	crc2

Slave response frame format:

Slave address	Function code	Start register (high bit)	Start register (low bit)	Data content (high bit)	Data content (low bit)	CRC16 check high bit	CRC16 check low bit
0x0E	0x06	0x00	0x0B	0x00	0x01	crc1	crc2

Register table

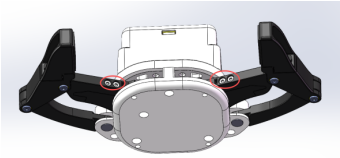
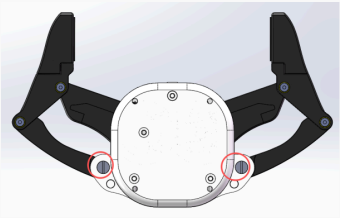
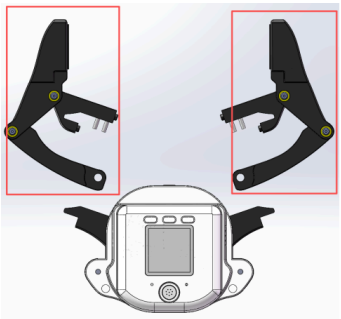
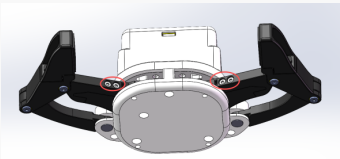
Register function	Register address	Read/write permission	Description
Read firmware main version number	1	Read	Version number*10 returns, starting from 1.0. For example, 1.0 returns 10
Read the firmware minor version number	2	Read	0-255. Each time the major version number remains unchanged, submit the test minor version number + 1
Set device ID number	3	Write	1 - 254
Read device ID number	4	Read	Return ID
Set 485 baud rate	5	Write	0 - 115200 (default) 1 - 1000000 2 - 57600 3 - 19200 4 - 9600 5 - 4800

Register function	Register address	Read/write permission	Description
Read 485 baud rate	6	Read	0 - 115200 1 - 1000000 2 - 57600 3 - 19200 4 - 9600 5 - 4800
Set the gripper enable status	10	Write	0 - Enable off 1 - Enable on
Set the gripper angle	11	Write	Angle value: 0-100
Read the gripper angle	12	Read	Angle value: 0-100
Set the gripper zero position	13	Write	0 - Failed 1 - Success
Read the gripper clamping status	14	Read	0 - Moving 1 - Stopped, no object detected 2 - Stopped, object detected 3 - After detecting that the object is clamped, the object falls
Set the gripper P value	15	Write	Value range 0-254
Read the gripper P value	16	Read	Value range 0-254

Register function	Register address	Read/write permission	Description
Set the gripper D value	17	Write	Value range 0-254
Read the gripper D value	18	Read	Value range 0-254
Set the gripper I value	19	Write	Value range 0-254
Read the gripper I value	20	Read	Value range 0-254
Set the gripper clockwise operable error	21	Write	Value range 0-16
Read the gripper clockwise Runnable error	22	Read	Value range 0-150
Set the counterclockwise runnable error of the gripper	23	Write	Value range 0-16
Read the counterclockwise runnable error of the gripper	24	Read	Value range 0-150
Set the minimum starting force of the gripper	25	Write	Value range 0-254
Read the minimum starting force of the gripper	26	Read	Value range 0-150
Set torque	27	Write	Value range 0-100
Read torque	28	Read	Value range 100-300
IO output setting	29	Write	Value range： 0,1,16,17
Set io opening angle	30	Write	Angle value: 0-100
Read io opening angle	31	Write	Angle value: 0-100
Set gripper speed	32	Write	Speed: 1-100
Read gripper default speed	33	Write	Speed: 1-100
Read Io opening angle	34	Write	Angle value: 0-100
Read Io closing angle	35	Write	Angle value: 0-100
Set absolute angle	36	Write	Angle value: 0-100
Pause motion	37	Write	Value: 0
Resume motion	38	Write	Value: 0
Stop motion	39	Write	Value: 0
Get the amount of data in the current queue	40	Read	Return the amount of data in the current absolute control queue

Register function	Register address	Read/write permission	Description
Set the servo virtual position value	41	Write	Value: 0-100
Read the servo virtual position value	42	Read	Value: 0-100
Set the clamping current	43	Read	Value: 100-300
Read the clamping current	44	Read	Value: 1-300

Instructions for Replacing Flexible Fingertips

Serial Number	Operation Diagram	Detailed Operation Instructions
1		Use an Allen wrench with a diagonal width of 2mm to remove the 4 screws shown in the left diagram, and keep the removed screws as they will be used for the subsequent installation of the flexible gripper.
2		Use a flat-head screwdriver to remove the two flat-head screws at the lower end of the gripper.
3		After the two sets of screws mentioned above are removed, the original gripper part can be separated.
4		Replace the flexible fingertips and place them at the positions of the left and right movement rods respectively. Use an Allen wrench with a diagonal width of 2mm to re-tighten the screws that were originally removed from the same positions

pymycobot library control

Note: Currently only mycobot 320 series and Mercury series can use the following pymycobot library API function to control the gripper. When using, connect the gripper to the end of the robot arm, and the time interval between each gripper instruction call must be greater than 1.5 seconds.When calling the `set_pro_gripper()` interface, you must install the latest version of `pymycobot` to resolve the issue regarding the inverted positioning of the ID parameter.

`

`set_pro_gripper(gripper_id, address, value)`

- **Function**: Set the Pro force control gripper parameters, and you can set a variety of parameter functions.You must install the latest version of `pymycobot` to resolve the issue regarding the inverted positioning of the ID parameter. For details, please see the following table.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
  - `address` (`int`): Gripper instruction number.
  - `value` : Parameter value corresponding to the instruction number.

Function	gripper_id	address	value
Set gripper ID	14	3	1 ~ 254
Set gripper Band	14	5	0 ~ 5 0-115200，1-1000000，2-57600，3-19200，4-9600，5-4800
Set gripper enable status	14	10	0 or 1, 0 - off enable; 1 - on enable
Set gripper angle	14	11	0 ~ 100
Zero position calibration	14	13	0
Set gripper clockwise runnable error	14	21	0 ~ 16
Set the anti-clockwise running error of the gripper	14	23	0 ~ 16
Set the minimum starting force of the gripper	14	25	0 ~ 254
IO output setting	14	29	0, 1, 16, 17
Set gripper torque	14	27	0 ~ 100
Set IO opening angle	14	30	0 ~ 100
Set IO closing angle	14	31	0 ~ 100
Set gripper speed	14	32	0 ~ 100
Set servo virtual position value	14	41	0 ~ 100
Set the clamping current	14	43	1 ~ 254

- **Return value**:
  - Please check the following table:

Function	Return value
Set the gripper ID	0 - Failure; 1 - Success
Set the gripper enable status	0 - Failure; 1 - Success
Set the gripper angle	0 - Failure; 1 - Success
Set the clockwise runnable error of the gripper	0 - Failure; 1 - Success
Zero position calibration	0 - Failure; 1 - Success
Set the counterclockwise runnable error of the gripper	0 - Failure; 1 - Success
Set the minimum starting force of the gripper	0 - Failure; 1 - Success
IO output setting	0 - Failure; 1 - Success
Set IO opening angle	0 - Failure; 1 - Success
Set IO closing angle	0 - Failure; 1 - Success
Set gripper torque	0 - Failure; 1 - Success
Set gripper speed	0 - Failure; 1 - Success
Set the servo virtual position value	0 - Failure; 1 - Success
Set the clamping current	0 - Failure; 1 - Success

`get_pro_gripper(gripper_id, address)`

- **Function**: Get the parameters of the Pro force-controlled gripper, which can get a variety of parameter functions. Please refer to the following table for details.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
  - `address` (`int`): The command number of the gripper.

Function	gripper_id	address
Read firmware major version number	14	1
Read firmware minor version number	14	2
Read gripper ID	14	4
Read gripper Band	14	6
Read gripper angle	14	12
Read gripper status	14	14
Read gripper clockwise runnable error	14	22
Read the anti-clockwise running error of the gripper	14	24
Read the minimum starting force of the gripper	14	26
Read gripper torque	14	28
Read gripper speed	14	33
Read the IO opening angle	14	34
Read the IO closing angle	14	35
Get the amount of data in the current queue	14	40
Read the servo virtual position value	14	42
Read the clamping current	14	44

- **Return value**:
  - Check the following table (if the return value is -1, it means that no data can be read):

Function	Return value
Read the firmware major version number	Major version number
Read the firmware minor version number	Minor version number
Read the gripper ID	1 ~ 254
Read the gripper Band	0 ~ 5 0-115200, 1-1000000, 2-57600, 3-19200, 4-9600, 5-4800
Read the gripper angle	0 ~ 100
Read the gripper clockwise runnable error	0 ~ 254
Read the gripper counterclockwise runnable error	0 ~ 254
Read the minimum starting force of the gripper	0 ~ 254
Read the gripper torque	0 ~ 100
Read the gripper speed	0 ~ 100
Read the IO opening angle	0 ~ 100
Read the IO closing angle	0 ~ 100
Get the amount of data in the current queue	Return the amount of data in the current absolute control queue
Read the servo virtual position value	0 ~ 100
Read the clamping current	1 ~ 254

`set_pro_gripper_angle(gripper_id, gripper_angle)`

- **Function**: Set the force-controlled gripper angle.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
  - `gripper_angle` (`int`): Gripper angle, value range 0 ~ 100.
- **Return value**:
  - 0 - Failed
  - 1 - Successful

`get_pro_gripper_angle(gripper_id)`

- **Function**: Read the force-controlled gripper angle.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value**: `int` 0 ~ 100

`set_pro_gripper_open(gripper_id)`

- **Function**: Open the force-controlled gripper.

- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value**:
  - 0 - Failed
  - 1 - Successful

`set_pro_gripper_close(gripper_id)`

- **Function**: Close the force-controlled gripper.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value**:
  - 0 - Failure
  - 1 - Success

`set_pro_gripper_calibration(gripper_id)`

- **Function**: Set the zero position of the force-controlled gripper. (The zero position needs to be set first when using it for the first time)
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value**:
  - 0 - Failure
  - 1 - Success

`get_pro_gripper_status(gripper_id)`

- **Function**: Read the gripping status of the force-controlled gripper.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value:**
  - `0` - Moving.
  - `1` - Stopped moving, no object was detected.
  - `2` - Stop motion, detect that the object is clamped.
  - `3` - After detecting that the object is clamped, the object falls.

`set_pro_gripper_torque(gripper_id, torque_value)`

- **Function**: Set the torque of the force-controlled gripper.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
  - `torque_value` (`int`): Torque value, value range 1 ~ 100.
- **Return value**:
  - 0 - Failed
  - 1 - Successful

`get_pro_gripper_torque(gripper_id)`

- **Function**: Read the torque of the force-controlled gripper.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value:** (`int`) 100 ~ 300

`set_pro_gripper_speed(gripper_id, speed)`

- **Function**: Set the force-controlled gripper speed.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
  - `speed` (int): Gripper movement speed, value range 1 ~ 100.
- **Return value**:
  - 0 - Failed
  - 1 - Successful

`get_pro_gripper_default_speed(gripper_id, speed)`

- **Function**: Read the default speed of the force-controlled gripper.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, value range 1 ~ 254.
- **Return value**: Default movement speed of the gripper, range 1 ~ 100.

`set_pro_gripper_abs_angle(gripper_id, gripper_angle)`

- **Function**: Set the absolute angle of the force-controlled gripper.
- **Parameter**:
  - `gripper_id` (`int`): Gripper ID, default 14, range 1 ~ 254.
  - `gripper_angle` (`int`): Gripper angle, range 0 ~ 100.
- **Return value**:
  - 0 - Failed
  - 1 - Successful

`set_pro_gripper_pause(gripper_id)`

- **Function**: Pause movement.
- **Parameter**:
  - `gripper_id` (`int`) Gripper ID, default 14, range 1 ~ 254.
- **Return value**:
  - 0 - Failure
  - 1 - Success

`set_pro_gripper_resume(gripper_id)`

- **Function**: Resume motion.
- **Parameter**:
  - `gripper_id` (`int`) Gripper ID, default 14, value range 1 ~ 254.
- **Return value**:
  - 0 - Failure
  - 1 - Success

`set_pro_gripper_stop(gripper_id)`

- **Function**: Stop motion.
- **Parameter**:
  - `gripper_id` (`int`) Gripper ID, default 14, value range 1 ~ 254.
- **Return value**:
  - 0 - Failure
  - 1 - Success

python USB-485 library control

USB-485 module wiring:

Connect the 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) wires at the gripper end, a total of 4 wires, the power supply is a 24V DC regulated power supply, insert the module's USB port into the computer's USB port

485A connected to 485 to USB module A+;

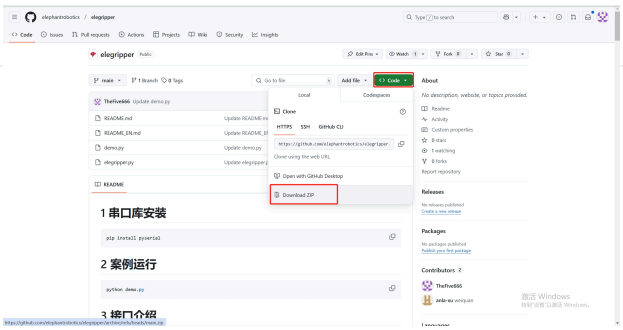
485B connected to 485 to USB module B-;

24V connected to the positive pole of the 24V DC regulated power supply;

GND connected to 24V Negative pole of DC regulated power supply

Driver library installation

Click to download the driver library



Execute the following command in the computer terminal to install the dependent library

```
pip install pyserial
```

API description

get_firmware_version()

- **Function:** Get the main version number of the gripper firmware
- **Parameter:** None
- **Return:** `(int)` Firmware main version number

get_modified_version()

- **Function:** Get the sub-version number of the gripper firmware
- **Parameter:** None
- **Return:** `(int)` Firmware sub-version number

get_gripper_Id()

- **Function:** Get the gripper ID
- **Parameter:** None
- **Return:** `(int)` Gripper ID

get_gripper_baud()

- **Function:** Get the baud rate of the gripper
- **Parameter:** None
- **Return:** `(int)` 0-5
  - `0` : 115200
  - `1` : 1000000
  - `2` : 57600
  - `3` : 19200
  - `4` : 9600
  - `5` : 4800

get_gripper_value()

- **Function:** Get the current position data of the gripper
- **Parameter:** None
- **Return:** `(int)` The current position data of the gripper

get_gripper_status()

- **Function:** Get the current status of the gripper
- **Parameter:** None
- **Return:** `(int)` 0-3
  - `0` : Moving
  - `1` : Stopped moving, no object was detected
  - `2` : Stopped moving, object was detected
  - `3` : After detecting that the object is clamped, the object falls

get_gripper_speed()

- **Function:** Get the current speed of the gripper
- **Parameter:** None
- **Return:** `(int)` The current speed of the gripper

get_gripper_P()

- **Function:** Get the P value of the gripper PID
- **Parameter:** None
- **Return:** `(int)` The P value of the gripper PID

get_gripper_I()

- **Function:** Get the I value of the gripper PID
- **Parameter:** None
- **Return:** `(int)` The I value of the gripper PID

get_gripper_D()

- **Function:** Get the D value of the gripper PID
- **Parameter:** None
- **Return:** `(int)` The D value of the gripper PID

get_gripper_cw()

- **Function:** Get the clockwise runnable error of the gripper
- **Parameter:** None
- **Return:** `(int)` Clockwise runnable error of the gripper

get_gripper_cww()

- **Function:** Get the counterclockwise runnable error of the gripper
- **Parameter:** None
- **Return:** `(int)` Counterclockwise runnable error of the gripper

get_gripper_mini_pressure()

- **Function:** Get the minimum starting force of the gripper
- **Parameter:** None
- **Return:** `(int)` Minimum starting force of the gripper

get_gripper_io_open_value()

- **Function:** Get the opening angle of the gripper Io
- **Parameter:** None
- **Return:** `(int)` Opening angle of the gripper Io

get_gripper_io_close_value()

- **Function:** Get the closing angle of the gripper Io
- **Parameter:** None
- **Return:** `(int)` Get the closing angle of the gripper Io

get_gripper_queue_count()

- **Function:** Get the amount of data in the current queue of the gripper
- **Parameter:** None
- **Return:** `(int)` The amount of data in the current queue of the gripper

get_gripper_vir_pos()

- **Function:** Get the virtual position value of the gripper servo
- **Parameter:** None
- **Return:** `(int)` The virtual position value of the gripper servo

get_gripper_protection_current()

- **Function:** Get the gripper clamping current
- **Parameter:** None
- **Return:** `(int)` The gripper clamping current

set_gripper_Id(value)

- **Function:** Set the gripper ID
- **Parameters:**
  - `value`：`(int)` Gripper ID, value range `1-254`
- **Return:** `(int)` 0-1
  - `0`：Failed

- `1` : Success

set_gripper_baud(value)

- **Function:** Set the gripper baud rate
- **Parameter:**
  - `value` : `(int)` Gripper baud rate, value range `0-5`
    - `0` : 115200
    - `1` : 1000000
    - `2` : 57600
    - `3` : 19200
    - `4` : 9600
    - `5` : 4800
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Successful

set_gripper_enable(value)

- **Function:** Set the gripper enable state
- **Parameter:**
- `value` : `(int)` Enable state, value range `0-1`
  - `0` : Disabled
  - `1` : Enabled
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_value(value,speed)

- **Function:** Set the gripper to rotate to the specified position at the specified speed
- **Parameter:**
  - `value` : `(int)` Position, value range `0-100`
  - `speed` : `(int)` Speed, value range `1-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_calibration()

- **Function:** Set the gripper Zero Calibration
- **Parameter:** None
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_P(value)

- **Function:** Set the P value of the gripper PID
- **Parameters:**
  - `value` : `(int)` P value, value range `0-254`

- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_I(value)

- **Function:** Set the I value of the gripper PID
- **Parameters:**
  - `value`：`(int)` I value, value range `0-254`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_D(value)

- **Function:** Set the D value of the gripper PID
- **Parameters:**
  - `value`：`(int)` D value, value range `0-254`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_cw(value)

- **Function:** Set the clockwise running error of the gripper
- **Parameter:**
  - `value`：`(int)` Error, value range `0-16`
- **Return:** `(int)` 0-1
  - `0` : Failure
  - `1` : Success

set_gripper_cww(value)

- **Function:** Set the counterclockwise running error of the gripper
- **Parameter:**
  - `value`：`(int)` Error, value range `0-16`
- **Return:** `(int)` 0-1
  - `0` : Failure
  - `1` : Success

set_gripper_mini_pressure(value)

- **Function:** Set the minimum starting force of the gripper
- **Parameter:**
  - `value`：`(int)` Minimum starting force, value range `0-254`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_torque(value)

- **Function:** Set gripper torque
- **Parameter:**
  - `value`：`(int)` Torque, value range `0-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_output(value)

- **Function:** Set gripper IO
- **Parameter:**
  - `value`：`(int)` Gripper IO, value range `0-3`
    - `0` : out1 off,out2 off
    - `1` : out1 on,out2 off
    - `2` : out1 off,out2 on
    - `3` : out1 on,out2 on
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_io_open_value(value)

- **Function:** Set the gripper Io open position
- **Parameter:**
  - `value`：`(int)` position, value range `0-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_io_close_value(value)

- **Function:** Set the gripper Io closed position
- **Parameter:**
  - `value`：`(int)` position, value range `0-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_speed(speed)

- **Function:** Set the gripper speed
- **Parameters:**
  - `speed`：`(int)` speed, value range `1-100`
- **Return:** `(int)` 0-1
  - `0` : failed
  - `1` : successful

set_abs_gripper_value(value,speed)

- **Function:** Set the gripper to rotate to the specified absolute position at the specified speed
- **Parameters:**
- `value`：`(int)` position, value range `1-100`
- `speed`：`(int)` speed, value range `1-100`
- **Return:** `(int)` 0-1
  - `0`: failed
  - `1`: successful

set_gripper_vir_pos(value)

- **Function:** Set the virtual position value of the gripper servo
- **Parameters:**
  - `value`：`(int)` virtual position, value range `0-100`
- **Return:** `(int)` 0-1
  - `0`: Failed
  - `1`: Success

set_gripper_protection_current(value)

- **Function:** Set the gripper gripping current
- **Parameter:**
  - `value`：`(int)` Virtual position, value range `1-254`
- **Return:** `(int)` 0-1
  - `0`: Failed
  - `1`: Success

set_gripper_pause()

- **Function:** Set the gripper to pause motion
- **Remarks:** Only valid for set_abs_gripper_value()
- **Parameter:** None
- **Return:** `(int)` 0-1
  - `0`: Failed
  - `1`: Success

set_gripper_resume()

- **Function:** Set the gripper to resume motion
- **Remarks:** Only valid for set_abs_gripper_value()
- **Parameters:** None
- **Return:** `(int)` 0-1
  - `0`: Failure
  - `1`: Success

set_gripper_stop()

- **Function:** Set the gripper to stop moving and clear the message queue
- **Remarks:** Only valid for set_abs_gripper_value()
- **Parameters:** None

- **Return:** `(int)` 0-1
  - `0` : Failure
  - `1` : Success

ros 控制

USB-485模块接线：

连接夹爪端的 24V，GND, 485_A(T/R+,485+) , 485_B(T/R-,485-)共 4 根线，电源为24V直流稳压电源，将模块的 USB 插口插入到电脑的 USB 接口

485A 接入 485 转 USB 模块 A+;

485B 接入 485 转 USB 模块 B-;

24V 接入 24V 直流稳压电源正极;

GND 接入 24V 直流稳压电源负极

使用环境

Linux Ubuntu20.04 ROS Noetic，具体环境搭建内容请查看 ROS环境搭建

pro_gripper_ros 包安装

`pro_gripper_ros` 是 ElephantRobotics 推出的，适配旗下myGripper F100 力控夹爪的 ROS 包。

项目地址：http://github.com/elephantrobotics/pro_gripper_ros

注意： 在安装包之前，请保证拥有 ROS 工作空间，默认的 ROS 工作空间是 catkin_ws。

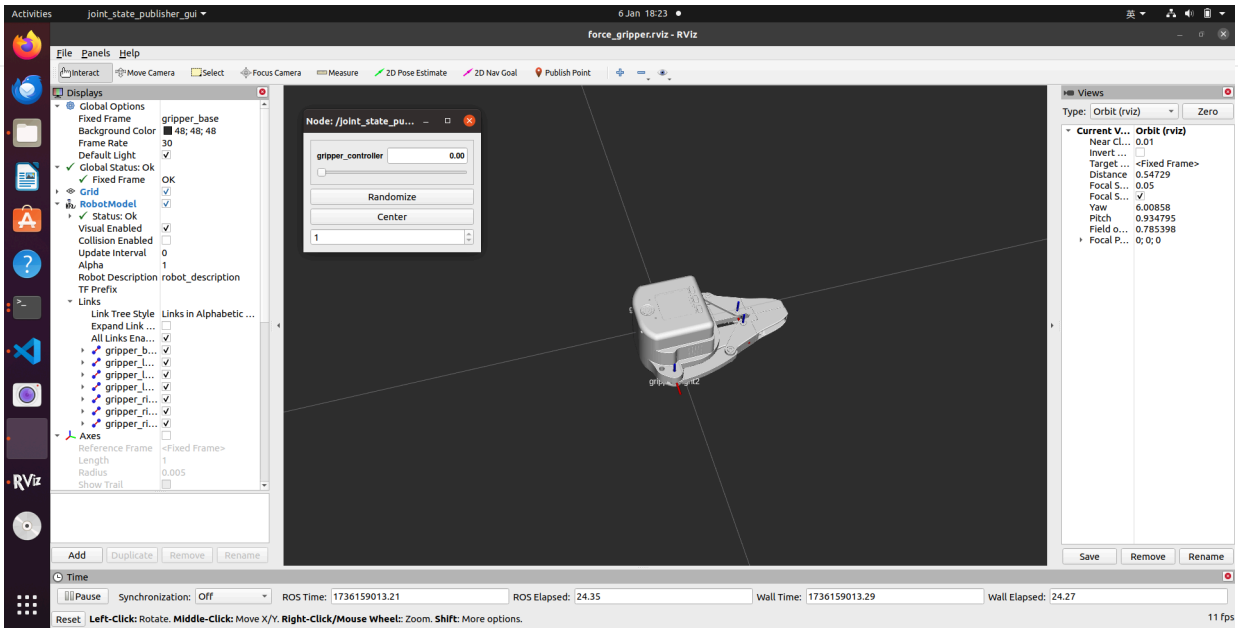
```
cd ~/catkin_ws/src  # 进入工作区的src文件夹中
# 克隆github上的代码
git clone https://github.com/elephantrobotics/pro_gripper_ros.git
cd ..       # 返回工作区
catkin_make # 构建工作区中的代码
cd ..
source devel/setup.bash # 添加环境变量
```

案例使用

打开终端命令行运行：

```
roslaunch pro_gripper_f100 force_gripper_slider.launch
```

打开 rviz 和一个滑块组件，您将看到如下界面：



然后你就可以在 rviz 中 控制模型，通过拖动滑块使其打开或关闭。如果想让真实的夹爪随着模型打开或关闭，则需要打开另一个终端命令行并运行：

```
# 确保在运行之前授予串行端口权限，默认串口为/dev/ttyACM0，波特率为115200，可根据实际串口修改
rosrun pro_gripper_f100 force_gripper_slider.py _port:=/dev/ttyACM0 _baud:=115200
```

Scenario Case

Mycobot320 force-controlled gripper use case, More case content

C3 force-controlled gripper use case

MyCobot pro630 force-controlled gripper use case


高性能 多用途
力矩

Download related materials

2D Drawing Download

MyGripper F100 Drawing jump link

MyGripper F100 Formal Lanhua blueprint jump link

MyGripper F100 Side mounted hair orchid drawing jump link

3D Drawing Download

MyGripper F100.STEP File jump link

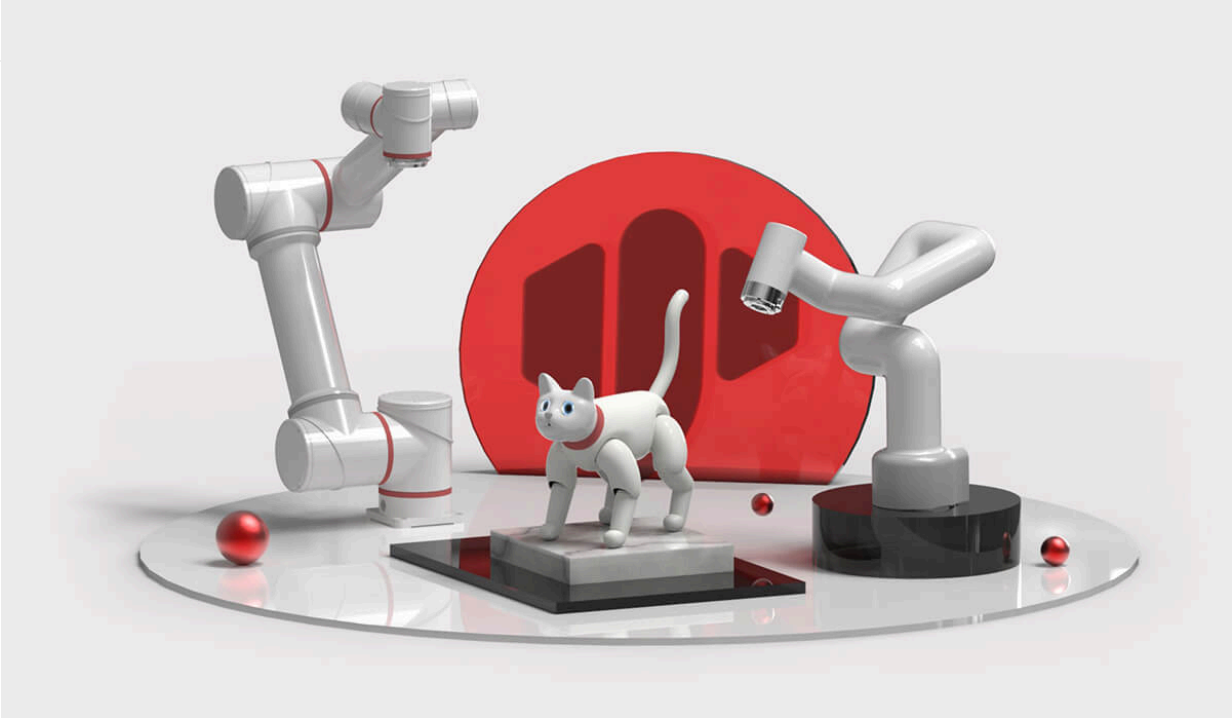
About Us

This chapter mainly provides information about the company profile and development history, providing users with additional background on the team and robot. The "Contact Us" section provides detailed contact information, including email format, instructions to describe the problem, and a template for reproduction steps. This enables users to communicate effectively with the team and solve problems. Overall, the goal of this chapter is to establish an efficient communication channel with users while presenting company and team information.

Elephant Robotics

1 Company Profile

Elephant Robotics is based in Shenzhen, China, and is a high-tech enterprise focusing on robot R&D, design and automation solutions. We are committed to providing highly flexible collaborative robots, easy-to-learn operating systems and intelligent automation solutions for robot education and scientific research institutions, commercial scenarios and industrial production. Its product quality and smart solutions have been unanimously recognized and praised by several factories from the world's top 500 companies in South Korea, Japan, the United States, Germany, Italy, Greece and other countries. Elephant Robotics adheres to the vision of "Enjoy Robots World", advocates the collaborative work of people and robots, makes robots a good helper for human work and life, helps people free themselves from simple, repetitive and boring work, gives full play to the advantages of human-machine collaboration, and thus improves work efficiency and helps humans create a better new life. In the future, Elephant Robotics hopes to promote the development of the robotics industry through a new generation of cutting-edge technology, and work together with customers and partners to open a new era of automation and intelligence.



2 Development History

2016.08 -----Elephant Robotics Co., Ltd. was officially established 2016.08 -----Entered the HAX incubator and received seed round investment from SOSV 2016.08 -----Started to develop the Elephant S industrial collaborative robot 2017.01 -----Rated as "Top 10 Most Innovative Companies in China at CES" 2017.04 -----Attended the Hannover Industrial Fair and the Korea Automation Exhibition 2017.07 -----The two founders were selected as "30 Business Elites Under 30" by Forbes Asia 2017.10 -----The fifth-generation single-arm industrial collaborative robot Elephant S was launched 2018.04 -----Received angel round investment from "Yun Angel Fund" 2018.06 -----First public appearance at the 2018 Hannover World Industrial Fair 2018.06 -----Won the "Smart Manufacturing Entrepreneurship MBA Award" from Cheung Kong Graduate School of Business 2018.06 -----Won the "Entrepreneurship Accelerator X-elerator Award" from Tsinghua School of Economics and Management 2018.11 -----Won the second place in the Shenzhen Division of the Asian Smart Hardware Competition 2018.11 -----Won the "Most Investment Enterprise Award" from the Gaogong Golden Globe Award 2019.03 -----Won the "Leadership Award" from the Gaogong Golden Globe Award 2019.04 -----March 2019 Catbot won the "Industrial Robot Innovation Award" 2019.09 -----Attended the Huawei European Ecosystem Conference (HCE) and officially became a member of Huawei's ecological partner 2019.11 -----Elephant Robot and Harbin Institute of Technology attended the IROS International Intelligent Robot and System Conference 2019.12 -----The "Intelligent Robot Joint Development Laboratory" of Elephant Robot and South China University of Technology was officially unveiled 2019.12 -----Won the 2019 "Innovation Technology Award" from Gaogong 2019.12 -----Won the "Top Ten Fast-Growing Enterprises" of Gaogong in 2019 2019.12 -----Won the "New Enterprise Award" in the Industrial Robot Segment of Shenzhen Equipment Industry 2019.12 -----The world's first bionic robot cat MarsCat was launched 2020.05 -----The founder won the Shenzhen Robot Newcomer Award in 2019 2020.10 -----The world's lightest and smallest six-axis collaborative robot myCobot was launched 2021.03 -----The smallest collaborative robot for scientific research myCobotPro 320 was launched 2021.05 -----Mars bionic cat MarsCat received competitive coverage from Xinhua Finance, China Daily, Nanjing Daily, Harbin Daily and other media 2021.07 -----Released the smallest composite robot chassis - the Little Elephant mobile robot myAGV 2021.09 -----The world's first fully enclosed four-axis robot arm - the Little Elephant palletizing robot myPalletizer was launched

3 Related links

- **Official website**: https://www.elephantrobotics.com
- **Purchase link**
- **Taobao**: https://shop504055678.taobao.com
- **shopify**: https://shop.elephantrobotics.com/
- Video
- **bilibili**: https://space.bilibili.com/2126215657
- **youtube**: https://www.youtube.com/c/Elephantrobotics

9.2 Contact us

If you have any other questions, you can contact us through the following methods

- **Email** E-mail : `sales@elephantrobotics.com`

We will reply within 1~2 working days

- **WeChat**

We only provide WeChat 1-on-1 service to users who have purchased myCobot.