Table of Contents

1 Product Introduction

myGripper H100 is a three-finger dexterous hand developed by Shenzhen Elephant Robotics Technology Co., Ltd. specifically for robot scientific research, teaching and other application scenarios. H100 is equipped with 6 high-performance digital servo motors, which can realize the control of multiple parameters such as output torque, movement speed, absolute position, etc. H100 supports 100Hz high-frequency communication, and can read status information in real time during the grasping process, fully meeting the needs of scientific research scenarios for data collection.

2 Compatible models

ER myCobot Pro 630

ER Mercury series

Applicable scenarios

Precision grasping: taking fruits, food, tools and equipment and other items

Physical interaction: combined with robots to realize actions such as likes, greetings, gesture control, etc.

Experimental operation: precise operation and experimental demonstration in the laboratory

Teaching demonstration: used for practical teaching in robotics courses

1 Product Introduction

myGripper H100 is a three-finger dexterous hand developed by Shenzhen Elephant Robotics Technology Co., Ltd. specifically for robot scientific research, teaching and other application scenarios. H100 is equipped with 6 high-performance digital servo motors, which can realize the control of multiple parameters such as output torque, movement speed, absolute position, etc. H100 supports 100Hz high-frequency communication, and can read status information in real time during the grasping process, fully meeting the needs of scientific research scenarios for data collection.

2 Compatible models

ER myCobot Pro 630

ER Mercury series

Applicable scenarios

Precision grasping: taking fruits, food, tools and equipment and other items

Physical interaction: combined with robots to realize actions such as likes, greetings, gesture control, etc.

Experimental operation: precise operation and experimental demonstration in the laboratory

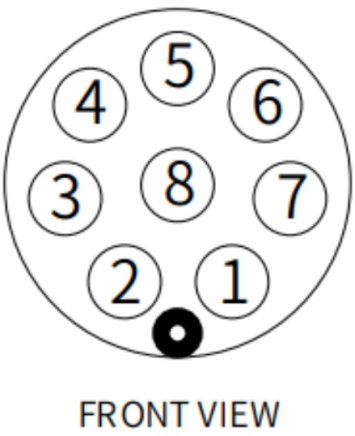
Teaching demonstration: used for practical teaching in robotics courses

Product Features

1 Product Specifications

Name	myGripper H100 three-finger dexterous hand
Gripping range	0 - 130mm
Number of fingertips	3 fingers, corresponding to the thumb, index finger and ring finger of the human hand
Service life	10W times and above
Movable joints	6
Motor type	Servo servo, supporting current, position and speed control
Weight	780g
Rated load	500g
Power supply parameters	24V2A
Fixing method	Screw fixing
Environment requirements for use	Normal temperature and pressure
Control interface	RS485 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 control IO input, numbers 6 and 7 are IO output, and numbers 4 and 8 are 485 communications, which are used to receive and send instructions with the dexterous hand

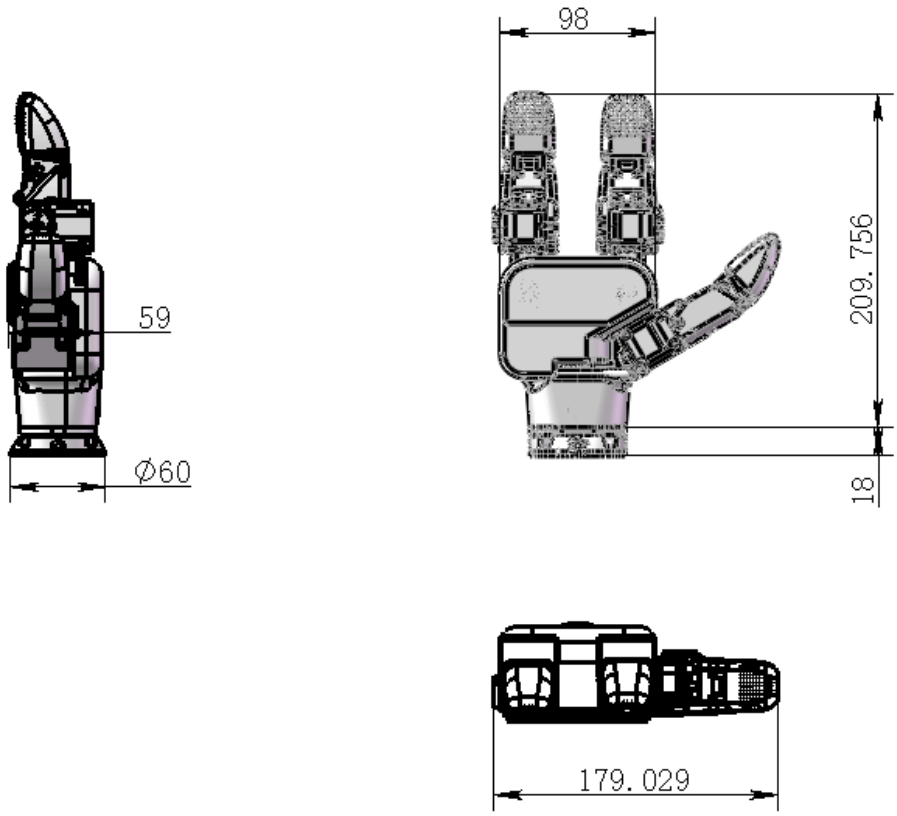
Notes:

To ensure the safe operation of the dexterous hand equipment, please strictly distinguish the line sequence according to the line mark. If the line mark is lost, detached, or forgotten, please contact the technical team through official channels immediately. If damage is caused by non-standard operation, our company will bear the corresponding repair costs according to the terms of the contract.

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



4 Motor Corresponding ID

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's 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 performed while the power is on, it may cause electric shock or malfunction.

3 Precautions

The following rules should be followed when using the three-finger dexterous hand:

- 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 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 the surface of the dexterous hand is stained due to use, it is recommended to wipe it gently with clean water. To avoid coating damage, do not use alcohol solvents to clean the shell and finger sleeves.** If the surface is damaged due to the use of alcohol-based cleaners and needs to be repaired or replaced, our company will charge the corresponding repair fee according to the terms of the contract

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 Product list

Name	Quantity
Three-finger dexterous hand	1
USB-485 module	1
Burner	1
M8 aviation plug	1
Accessory package	1
Flange connector	2

2 Unboxing video

3 Notes

The following rules should be followed when using the three-finger dexterous hand:

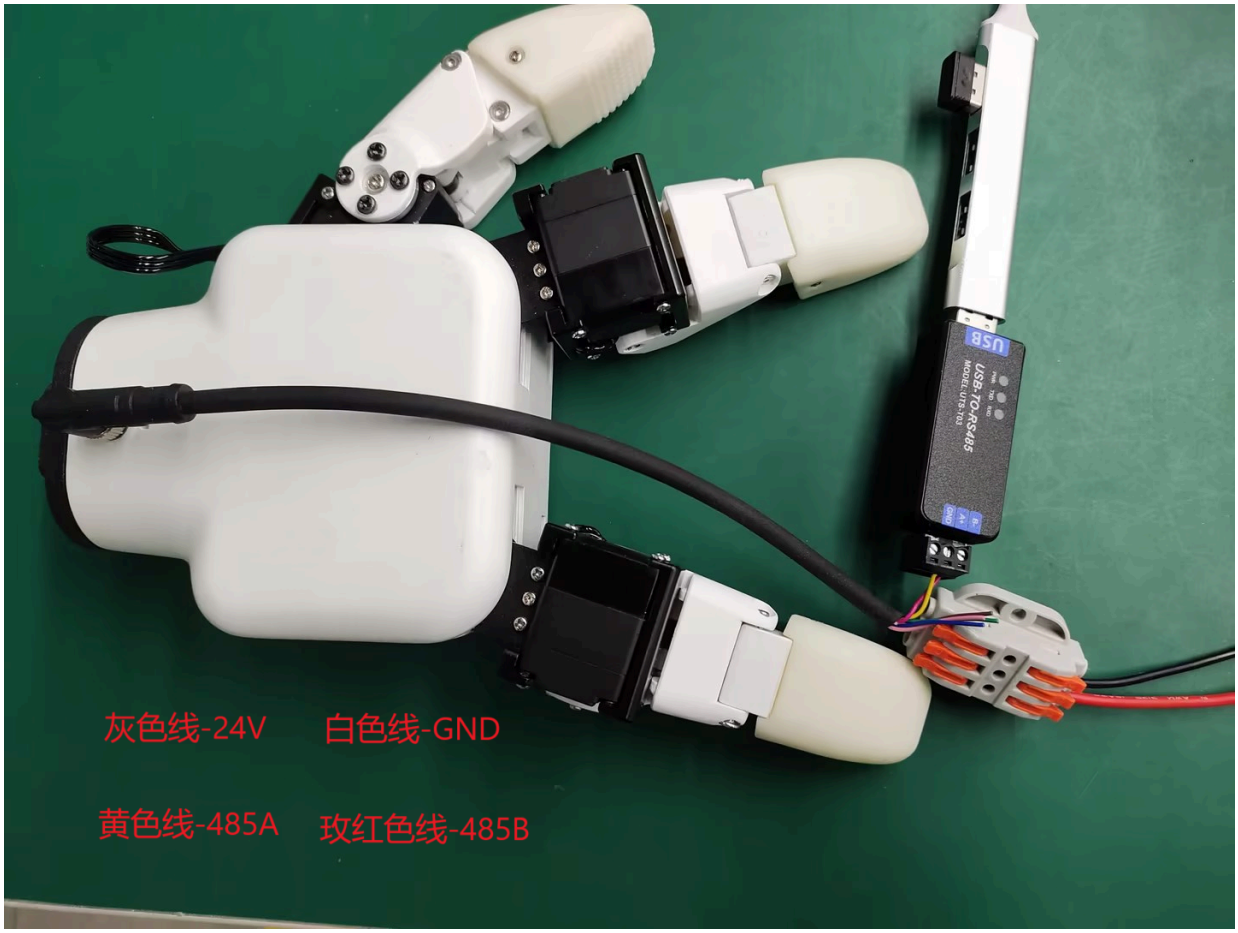
- Please distinguish the line sequence according to the line mark. If the line mark is lost, detached, or forgotten, please contact our staff to cooperate in determining the line sequence. If you do not contact our staff, **you will be responsible for the consequences if the smart hand is damaged due to the wrong wiring 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 the surface of the smart hand is stained by use, it is recommended to wipe it gently with clean water. To avoid coating damage, do not use alcohol solvents to clean the shell and finger sleeves.** If the surface is damaged due to the use of alcohol cleaners and needs to be repaired or replaced, our company will charge the corresponding repair fee according to the terms of the contract

Basic usage

Serial port control method

USB-485 module wiring:

Connect the 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) of the smart hand 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 connects to the 485 to USB module A+;

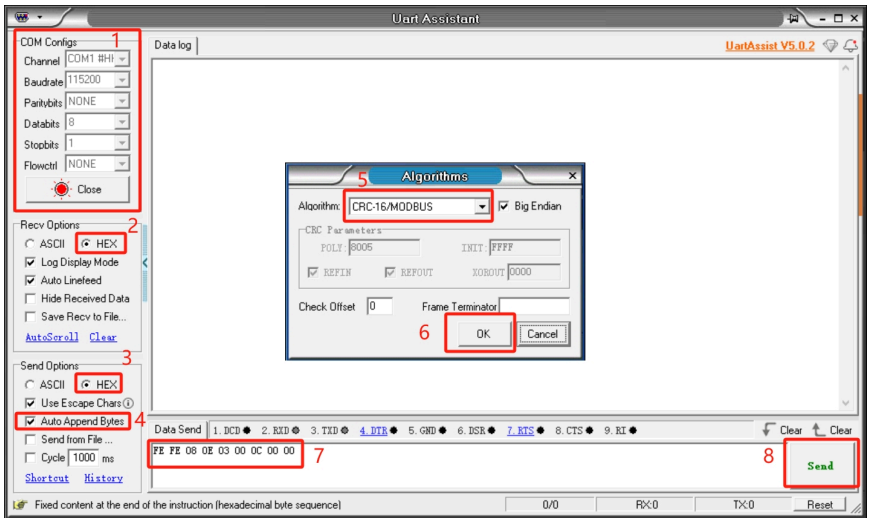
485B connects to the 485 to USB module B-;

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

GND connects 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 dexterous hand command, CRC check code does not need to be filled in, UartAssist serial port debugging assistant will automatically generate it.



Default configuration of the dexterous hand serial port

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

The dexterous hand 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 (2*n bytes), and check code (2byte). Let's take reading the angle of the dexterous hand 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 01	71 01

Frame header: 254 254

Length: Command length 08

ID: 0E, can be modified in the device, the default ID is 14, 0E represents the current ID of the dexterous hand 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 Register address corresponding to the dexterous hand function

Register parameter: 00 01 If reading, just fill in the servo ID, if setting, fill in the servo ID and the written parameters

CRC check: 71 01 Ensure that the terminal device does not respond to data that changes during transmission, ensure the security and efficiency of the system, and the CRC check adopts the 16-bit cyclic remainder method. Verify all the hexadecimal values of the frame header, length, function code, register address, and register parameters directly to get the check code 71 01.

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 command also needs to be modified to be the same to communicate

- Read

  Command: 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 returns 00 0E, indicating the current smart hand 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: 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

    Instruction: 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, corresponding to the baud rate value of the setting parameter

- Set the dexterous hand enable state

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

  Function code: 06 Write operation

  Parameter: 00 00, 00 means disconnection, 00 01 means enablement

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

  Note: Success returns 00 01, failure returns 00 00

- Set/read the dexterous hand angle
  - Set

    Instruction: FE FE 0A 0E 06 00 0B 00 01 00 64 FF 39

Function code: 06 Write operation

Parameter: 00 01 00 64, set the servo 1 angle to 100

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

Note: Success returns 00 01, failure returns 00 00

- Read

  Instruction: FE FE 08 0E 03 00 0C 00 01 71 01

  Function code: 03 Read operation

  Parameter: None

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

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

- Set the servo zero position

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

  Function code: 06 Write operation

  Parameter: 00 01 Set the servo 1 zero position

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

  Note: Success returns 00 01, failure returns 00 00

- Read the gripping status of the dexterous hand

  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

  Note: 00 01, return data 0 means moving ,1: Stop motion, no clamping detected, 2: Stop motion, clamping detected, 3: Clamping detected, object dropped

- Set/read the servo P value

  - Set

    Command: FE FE 0A 0E 06 00 0F 00 01 00 64 3F C8

    Function code: 06 Write operation

    Parameter: 00 01 00 64, set the servo 1 P value to 100, setting range (1-254)

    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 01 B7 C0

    Function code: 03 Read operation Parameter: 00 01 Servo 1 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 servo D value

- Set

  Command: FE FE 0A 0E 06 00 11 00 01 00 64 3D 60

  Function code: 06 Write operation

  Parameter: 00 01 00 64, set the servo 1 D value to 100, setting range (1-254)

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

  Note: Success returns 00 01, failure returns 00 00

  If the dexterous hand shakes, the D value can be appropriately increased

- Read

  Command: FE FE 08 0E 03 00 12 00 01 77 61

  Function code: 03 Read operation

  Parameter: 00 01 Servo 1

  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 servo I value
  - Set

    Command: FE FE 0A 0E 06 00 13 00 01 00 00 16 18

    Function code: 06 Write operation

    Parameter: 00 01 00 00, set servo 1 I value to 0, setting range (1-254)

    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 01 76 81

    Function code: 03 Read operation

    Parameter: 00 01 Servo 1

    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 running error of the servo
  - Set

    Command: FE FE 0A 0E 06 00 15 00 01 00 03 17 D0

    Function code: 06 Write operation

    Parameter: 00 01 00 03, set the error value of servo 1 to 3, setting range (0-16)

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

    Mark: Success returns 00 01, failure returns 00 00

  - Read

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

Function code: 03 Read operation

Parameter: 00 01 Servo 1

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

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

- Set/read the counterclockwise operational error of the servo
  - Set

    Command: FE FE 0A 0E 06 00 17 00 01 00 03 D7 A9

    Function code: 06 Write operation

    Parameter: 00 01 00 03, set the error value of the servo 3, 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 01 75 41

    Function code: 03 Read operation

    Parameter: 00 01 Servo 1

    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 servo
  - Set

    Command: FE FE 0A 0E 06 00 19 00 01 00 18 1D 80

    Function code: 06 Write operation

    Parameter: 00 01 00 18 Set the value of servo 1 to 24

    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 01 B5 E0

    Function code: 03 Read operation

    Parameter: 00 01 Servo 1

    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 servo torque
  - Set

    Command: FE FE 0A 0E 06 00 1B 00 01 00 64 3C F8

Function code: 06 Write operation

Parameter: 00 01 00 64 Set the torque of servo 1 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 01 B4 00

Function code: 03 Read operation

Parameter: 00 01 Servo 1

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

- Set/read servo speed
  - Set

Command: FE FE 0A 0E 06 00 20 00 01 00 14 1D 1C

Function code: 06 Write operation

Parameter: 00 01 00 14 Set the speed of servo 1 to 20, 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

Instruction: FE FE 08 0E 03 00 21 00 01 78 91

Function code: 03 Read operation

Parameter: 00 01 Servo 1

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

Note: 00 32 Data returned is 50

- Set the dexterous hand gesture
  - Set

Command: FE FE 08 0E 06 00 34 01 01 00 35 EC

Function code: 06 Write operation

Parameter: 01 01 Set gesture (0-4), gesture closure degree (0-5)

Return: FE FE 08 0E 06 00 34 00 01 B8 0C

Note: Success returns 00 01, failure returns 00 00

  - Set/read the dexterous hand angle
  - Set

Command: FE FE 12 0E 06 00 2D 00 00 00 00 00 00 00 00 00 00 00 00 00 14 23 FC

Function code: 06 Write operation

Parameter: 00 00 00 00 00 00 00 00 00 00 00 00 00 14 The first 12 bytes represent the angle (0-100), and the last two bytes represent the speed (0-100)

Return: FE FE 08 0E 06 00 2D 00 01 7B 9D

Note: Success returns 00 01, failure returns 00 00

- Read

Instruction: FE FE 08 0E 03 00 32 00 00 7D A1

Function code: 03 Read operation

Parameter: None

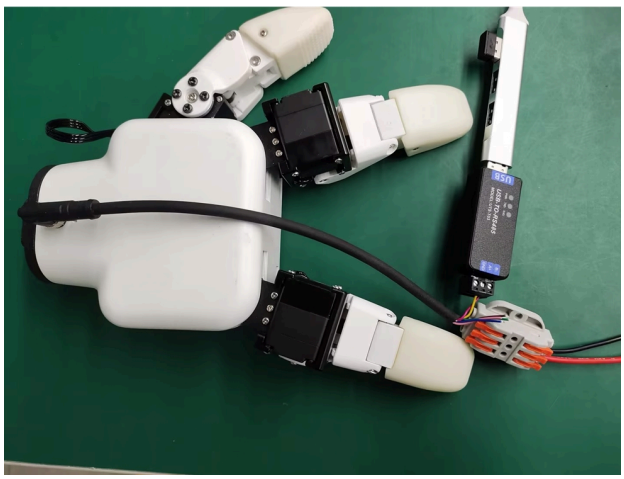
Return: FE FE 0C 0E 03 00 32 00 00 00 01 00 00 00 00 00 01 00 00 84 74

Note: 00 00 00 01 00 00 00 00 00 01 00 00 The data returned is [0, 1, 0, 0, 1, 0]

python development

USB-485 module wiring:

Connect the 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) of the smart hand 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



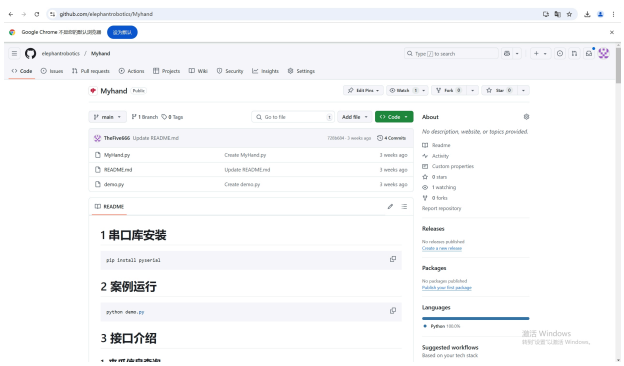
485A connects to the 485 to USB module A+;

485B connects to the 485 to USB module B-;

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

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

Driver library installation Click to download the driver library



Serial port dependency library installation

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

```
pip install pyserial
```

API description

get_gripper_firmware_version()

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

get_gripper_modified_version()

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

get_gripper_gripper_Id()

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

get_gripper_gripper_baud()

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

get_gripper_joint_angle(id)

- **Function:** Get the current position data of the gripper
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` Current position data of the gripper joint ID

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 the object was detected, the object fell

get_gripper_joint_speed(id)

- **Function:** Get the current speed of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` Current speed of the gripper joint ID

get_gripper_joint_P(id)

- **Function:** Get the P value of the PID of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` The P value of the PID of the gripper joint ID

get_gripper_joint_I(id)

- **Function:** Get the I value of the PID of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` The I value of the PID of the gripper joint ID

get_gripper_joint_D(id)

- **Function:** Get the D value of the PID of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` The D value of the PID of the gripper joint ID

get_gripper_joint_cw(id)

- **Function:** Get the clockwise runnable error of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` The clockwise runnable error of the gripper joint ID

get_gripper_joint_cww(id)

- **Function:** Get the counterclockwise runnable error of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` Anti-clockwise runnable error of the gripper joint ID

get_gripper_joint_mini_pressure(id)

- **Function:** Get the minimum starting force of the gripper joint ID
- **Parameter:** `id` : `(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` Minimum starting force of the gripper joint ID

get_gripper_joint_mini_pressure(id)

- **Function:** Get the minimum starting force of the gripper joint ID
- **Parameter:** `id`：`(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` Minimum starting force of the gripper joint ID

get_gripper_angles()

- **Function:** Get the angles of the 6 joints of the gripper
- **Parameter:** `id`：`(int)` Gripper joint ID, value range `1-6`
- **Return:** `(list)` Angles of the 6 joints of the gripper

set_gripper_Id(value)

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

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`: Success

set_gripper_enable(value)

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

set_gripper_joint_calibration(id)

- **Function:** Set the gripper joint ID zero calibration
- **Parameter:** `id`：`(int)` Gripper joint ID, value range `1-6`
- **Return:** `(int)` 0-1
  - `0`：Failed
  - `1`：Success

set_gripper_joint_P(id,value)

- **Function:** Set the P value of the PID of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` P value, value range `0-254`
- **Return:** `(int)` 0-1
  - `0`：Failed
  - `1`：Success

set_gripper_joint_I(id,value)

- **Function:** Set the I value of the PID of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` I value, value range `0-254`
- **Return:** `(int)` 0-1
  - `0`：Failed
  - `1`：Success

set_gripper_joint_D(id,value)

- **Function:** Set the D value of the PID of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` D value, value range `0-254`
- **Return:** `(int)` 0-1
  - `0`：Failed
  - `1`：Success

set_gripper_joint_cw(id,value)

- **Function:** Set the clockwise runnable error of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` Error, value range `0-16`
- **Return:** `(int)` 0-1
  - `0`：Failed
  - `1`：Success

set_gripper_joint_cww(id,value)

- **Function:** Set the counterclockwise runnable error of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` Error, value range `0-16`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_joint_mini_pressure(id,value)

- **Function:** Set the minimum starting force of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` Minimum starting force, value range `0-254`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_joint_torque(id,value)

- **Function:** Set the torque of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `value`：`(int)` Torque, value range `0-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_joint_speed(id,speed)

- **Function:** Set the speed of the gripper joint ID
- **Parameters:**
  - `id`：`(int)` Joint ID, value range `1-6`
  - `speed`：`(int)` Speed, value range `1-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

set_gripper_angles(angles,speed)

- **Function:** Set the angle of all joints of the gripper
- **Parameters:**
  - `angles`：`(list)` 6 joint angles, each joint angle has a value range of `0-100`
  - `speed`：`(int)` speed, value range of `1-100`
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Successful

set_gripper_action(value)

- **Function:** Set the gripper pinching action
- **Parameters:**
  - `value`：`(int)` action, value range of `0-3`
    - `0` : Index finger and thumb pinching
    - `1` : Middle finger and thumb pinching
    - `2` : Three-finger gripping
    - `3` : Two-finger gripping
- **Return:** `(int)` 0-1
  - `0` : Failed
  - `1` : Success

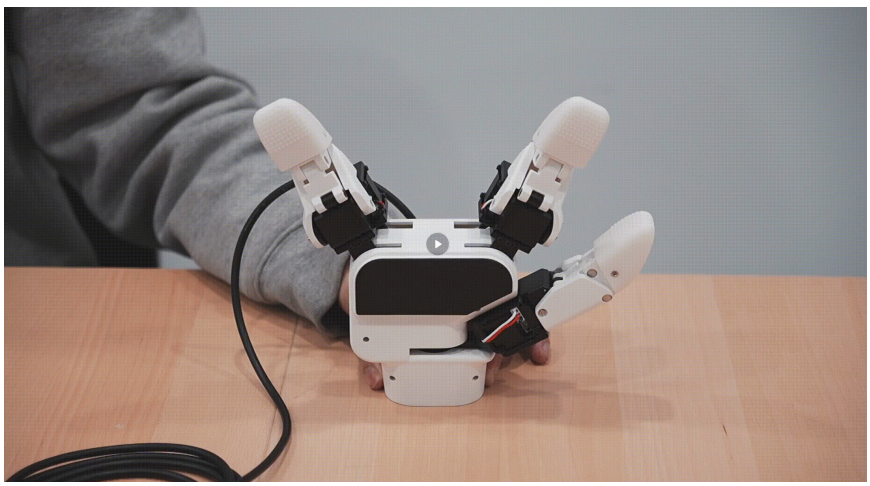
set_gripper_pose(pose,value,flag)

- **Function:** Set the gripper pinching action and opening and closing degree
- **Parameters:**
  - `pose`：`(int)` Action, value range `0-4`
    - `0` : All joints return to zero
    - `1` : Index finger and thumb pinching
    - `2` : Middle finger and thumb pinching
    - `3` : Middle finger and index finger pinching
    - `4` : Three-finger pinching
  - `value`：`(int)` Opening and closing degree, value range `0-15`, closing degree, the higher the level, the closer it is
  - `flag`：`(int)` Idle flag, when flag 1, the idle finger can be freely controlled
- **Return:** `(int)` 0-1
  - `0` : Failure
  - `1` : Success

Testing Procedure

```
from MyHand import MyGripper_H100
import time
if __name__=="__main__":
    hand=MyGripper_H100("COM8")
    hand.set_gripper_pose(0,0)
    time.sleep(2)
    hand.set_gripper_pose(1,5)
    time.sleep(5)
    hand.set_gripper_pose(2,5)
    time.sleep(5)
    hand.set_gripper_pose(3,5)
    time.sleep(5)
    hand.set_gripper_pose(4,15)
    time.sleep(5)
    hand.set_gripper_pose(0,0)
    time.sleep(2)
```

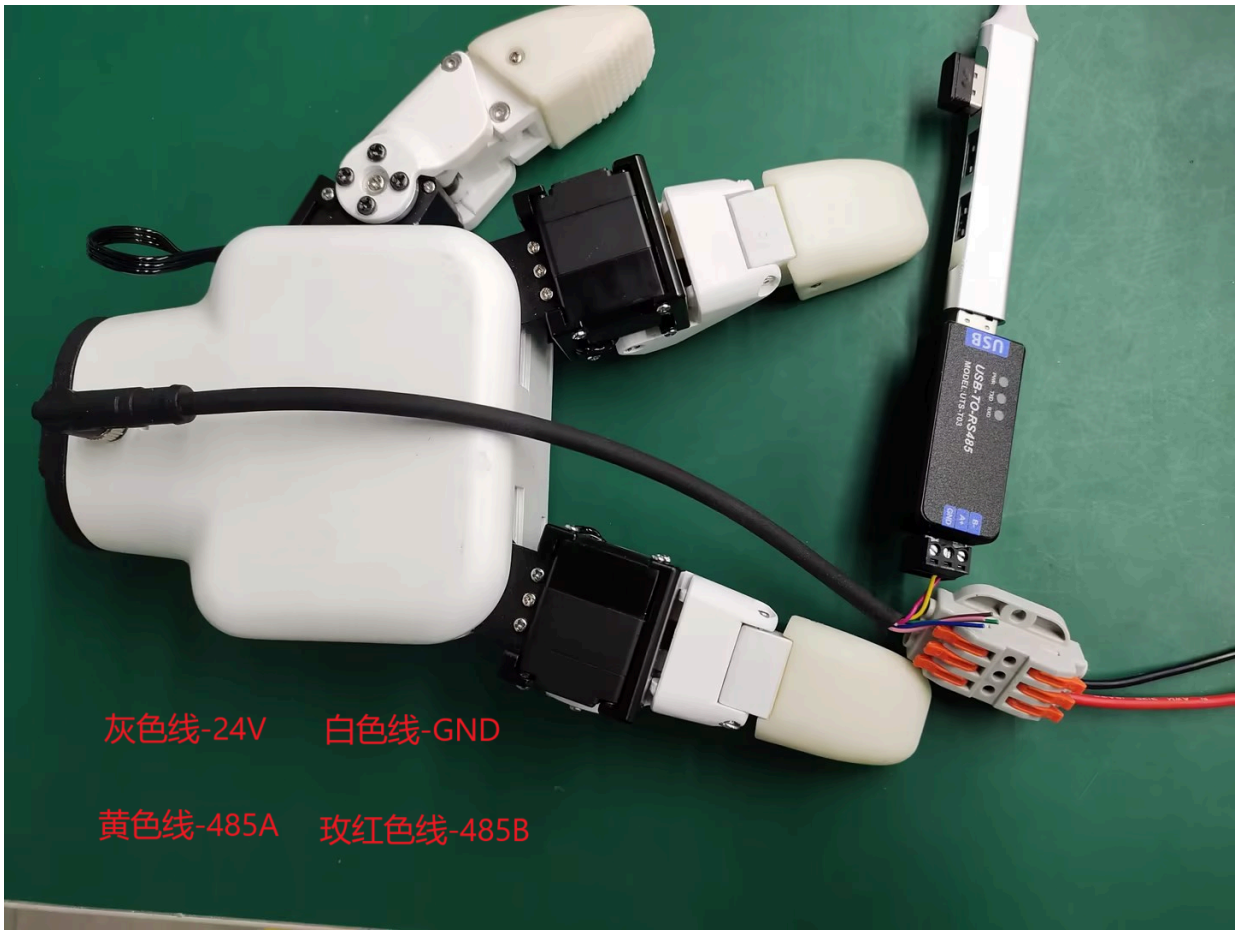
Effect display



Python USB-485 library control

Hardware connection

Connect the 24V, GND, 485_A (T/R+, 485+), 485_B (T/R-, 485-) of the smart hand 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 connects to the 485 to USB module A+;

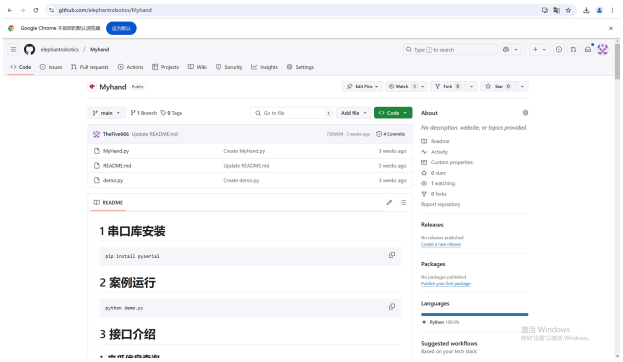
485B connects to the 485 to USB module B-;

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

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

Software installation

Driver library installation Click to download the driver library



Serial port dependency library installation

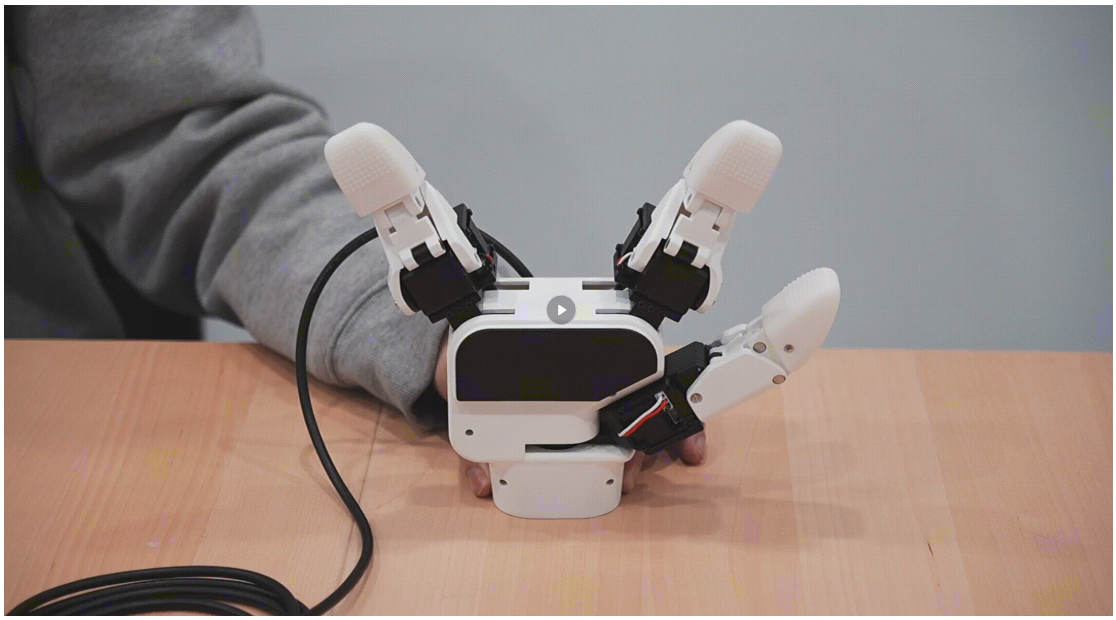
Execute the following command in the computer terminal to install the dependency library

```
pip install pyserial
```

Example program

```
from MyHand import MyGripper_H100
import time
if __name__=="__main__":
    hand=MyGripper_H100("COM8")
    hand.set_gripper_pose(0,0)
    time.sleep(2)
    hand.set_gripper_pose(1,5)
    time.sleep(5)
    hand.set_gripper_pose(2,5)
    time.sleep(5)
    hand.set_gripper_pose(3,5)
    time.sleep(5)
    hand.set_gripper_pose(4,15)
    time.sleep(5)
    hand.set_gripper_pose(0,0)
    time.sleep(2)
```

Effect display



Mercury A1 Control

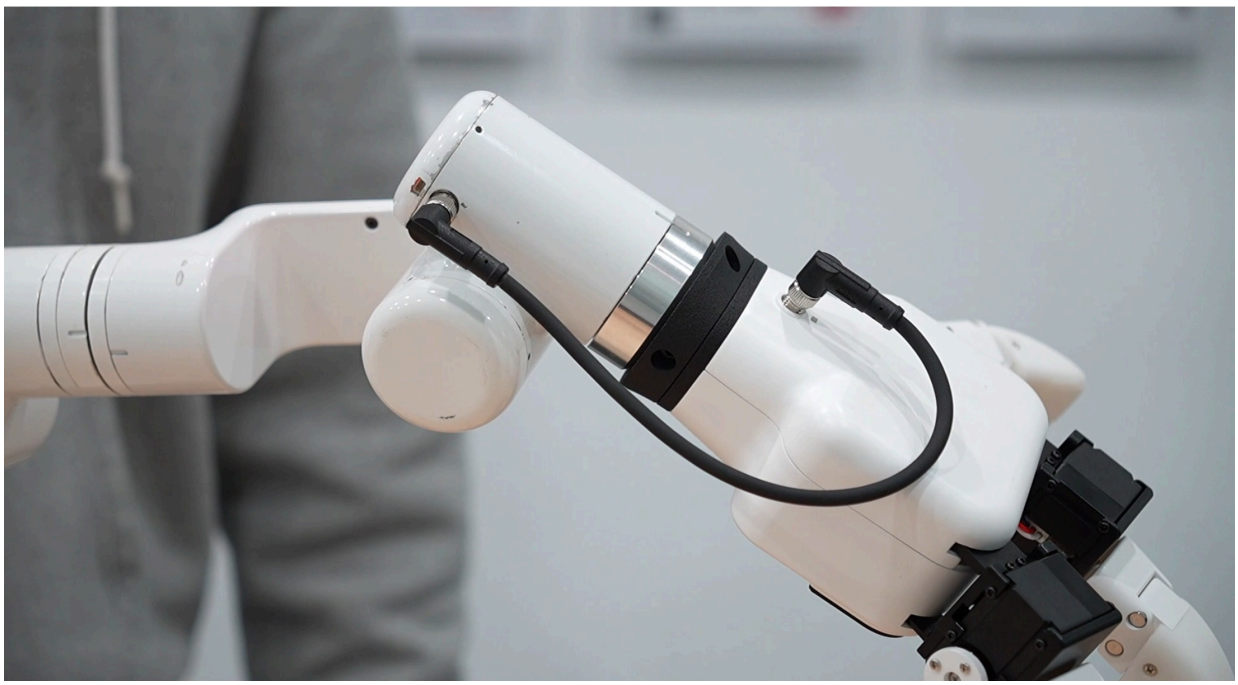
Hardware Installation

First install the connector on the smart hand

Then install another connector on the end flange of the robot arm

Then install the smart hand on the end connection flange, and fix the 4 holes on the top, bottom, left and right with screws

Finally, connect and tighten with M8 aviation plug wire



Software Installation

Dependency Library Installation

```
pip install pymycobot --upgrade
```

Example program

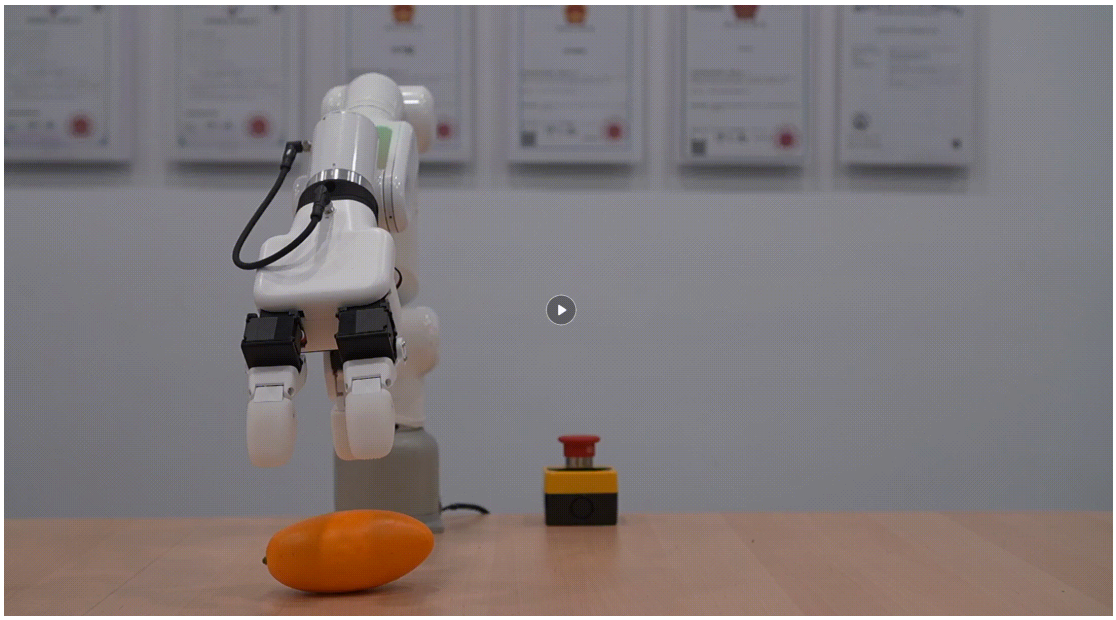
```
from pymycobot import Mercury
import time
me=Mercury()
me.set_hand_gripper_pinch_action_speed_consort(14,4,1)
me.power_on()
target_angles=[
[-11.326, 0.245, -0.909, -122.652, 5.459, 167.24, 0.38],
[-11.327, 13.361, -0.902, -122.646, 5.46, 167.24, 0.381],
[-11.319, 6.808, 16.44, -117.003, 5.461, 167.24, 0.39],
[-11.324, 18.251, 16.324, -117.001, 5.461, 167.241, 0.39]

]

def wait():
    time.sleep(0.2)
    while me.is_moving()==1:
        pass

for i in range(len(target_angles)):
    me.send_angles(target_angles[i],20)
    wait()
    if i ==1:
        me.set_hand_gripper_pinch_action_speed_consort(14,4,8)
        time.sleep(2)
        me.send_angles(target_angles[i-1],20)
        wait()
    elif i ==3:
        me.set_hand_gripper_pinch_action_speed_consort(14,4,1)
        time.sleep(2)
        me.send_angles(target_angles[i-1],20)
        wait()
```

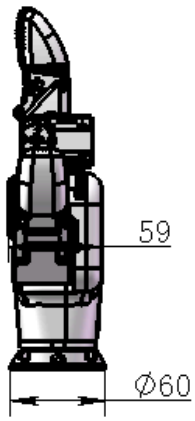
Effect display

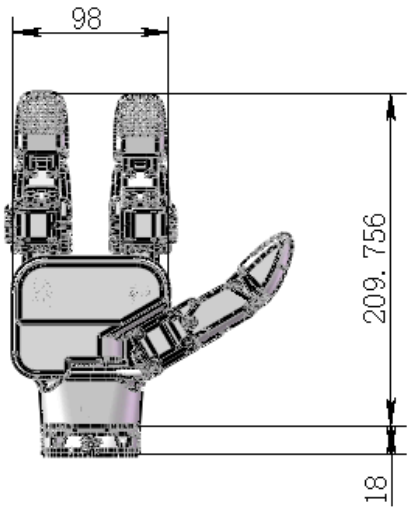


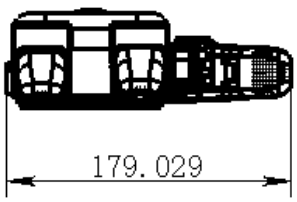
Related material download

Download address for the 3D model of the right hand

Download address for the 3D model of the left hand







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.

大象机器人

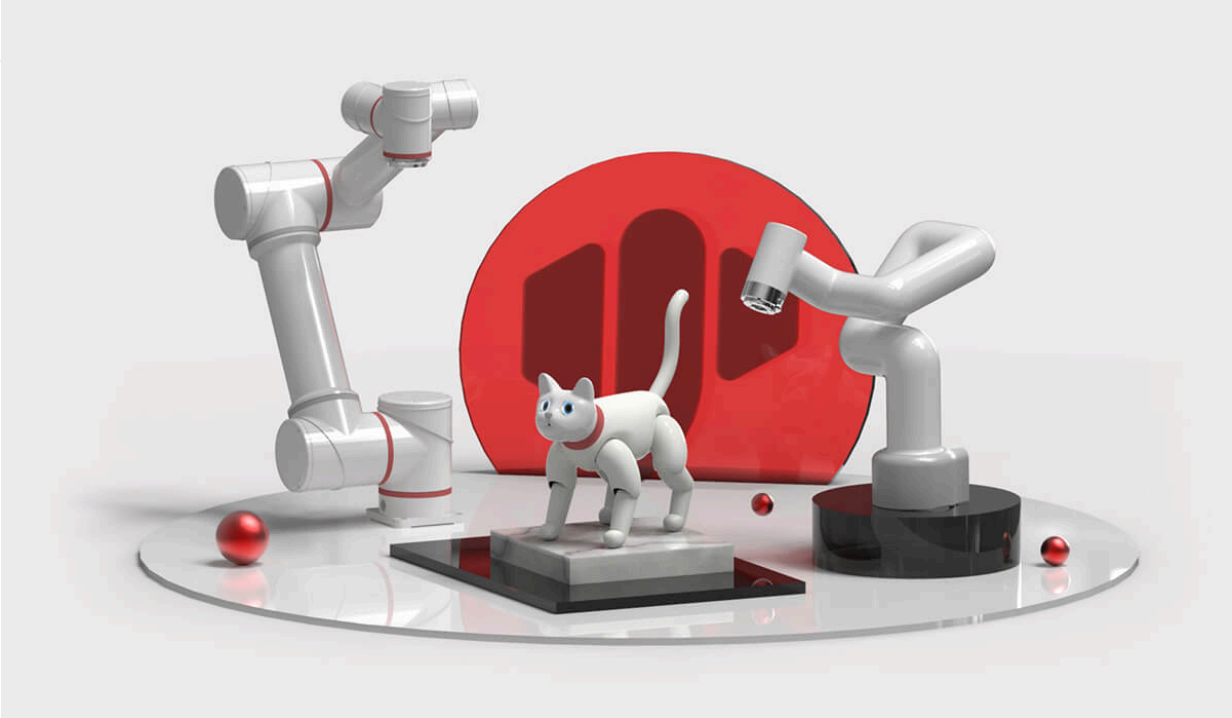
1 Company Introduction

Elephant Robotics is based in Shenzhen, China, a high-tech company which focuses on the design, research and development of robots and automation solutions.

We are devoted to providing highly flexible robots, easy-to-learn operating systems and intelligent automation solutions for robot education, scientific research institutions, business situations and industrial production. Our product quality and intelligent solutions have obtained unanimous acceptance and favorable comments from a number of world top 500 enterprises & factories in South Korea, Japan, America,Germany, Italy, Greece, etc.

Abiding by the vision of "enjoy robots world, Elephant Robotics initiates collaborative work between human and robots to make robots be good life and work helpers for human so as to free people from simple,repeated and dull jobs and give full play to the advantages of human-robot coordination, thus improving work efficiency and helping human to create a nice, new life.

In the future, Elephant Robotics hopes to promote the development of the robot industry through a new generation of cutting-edge technology, and starts a new era of automation and intelligence with its customers hand in hand.



2 Development History

In August 2016, Elephant Robotics was established.

In August 2016, entered HAX Incubator and obtained SOSV seed round investment.

In July 2017, the two founders were included in Forbes Asia\'s "30 Business Elites under Age 30".

In October 2017, published the fifth generation of single-arm industrial cobot called Elephant S.

In April 2018, obtained angel round investment from Cloud Angel Fund.

In June 2018, was awarded "Intelligent Manufacture Entrepreneurship MBA Award" by CKGSB.

In June 2018, was awarded "Startup Accelerator X-elerator Award" operated by Tsinghua University.

In November 2018, won the second place in the Asian Smart Hardware Competition in Shenzhen Division

In November 2018, obtained the \"Most Invested Company Award\" in GaogongGold Globe Award.

In March 2019, obtained the \"Leading Person Award \" in Gaogong Gold Globe Award.

In April 2019, obtained Catbot "Industrial Robot Innovation Award".

In September 2019, attended Huawei European Eco-Conference (HCE) and became a member of Huawei eco-partners.

In November 2019, Elephant Robotics attended the IROS International Conference on Intelligent Robots and Systems jointly with Harbin Institute of Technology.

In December 2019, obtained "Gaogong 2019 Innovation Technology Award".

In December 2019, was awarded as one of the Gaogong 2019 Top 10 Fast Growing Enterprises.

In December 2019, was awarded the \"Emerging Enterprise Award" in the industrial robotics segment field of Shenzhen equipment industry.

In December 2019, launched the fist type of bionic robotic cat called MarsCat in the world.

In May 2020, the founders obtained \"Shenzhen Robot Emerging Talent Award" in 2019.

In October 2020, launched the smallest six-axis cobot named myCobot in the world.

In March 2020, launched the smallest cobot named myCobotPro 320 for scientific research in the world.

In May 2021, the Mars bionic cat named MarsCat was reported by several media such as Xinhua Finance, China Daily, Nanjing Daily, Harbin Daily, etc.

In July 2021, published the seat for the smallest hybrid robot, a baby elephant moving robot called myAGV.

In September 2021, launched the world's first type of fully wrapped four-axis robot arm, a tiny elephant palletizing robot arm called myPalletizer.

3 Related Links

- Official website: https://www.elephantrobotics.com
- Purchase link
  - Shopify : https://shop.elephantrobotics.com/
- Video
  - Bilibili : https://space.bilibili.com/2126215657
  - Youtube : https://www.youtube.com/c/Elephantrobotics
- Facebook : https://www.facebook.com/mycobotcreator/
- Linkedin : https://www.linkedin.com/company/18319865
- X (Twitter) : https://twitter.com/CobotMy
- Discord : https://discord.gg/2MAherp7nt
- Hackster : https://www.hackster.io/elephant-robotics

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**

  我们只对已经购买过myCobot的用户进行微信1对1服务。