Welcome

Thank you for choosing Freenove products!

About Battery

First, read the document **About_Battery.pdf** in the unzipped folder.
If you have not downloaded the zip file, please download it and unzip it via link below.
https://github.com/Freenove/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/archive/master.zip

Get Support and Offer Input

Freenove provides free and responsive product and technical support, including but not limited to:

- Product quality issues
- Product use and build issues
- Questions regarding the technology employed in our products for learning and education
- Your input and opinions are always welcome
- We also encourage your ideas and suggestions for new products and product improvements

For any of the above, you may send us an email to:

support@freenove.com

Safety and Precautions

Please follow the following safety precautions when using or storing this product:

- Keep this product out of the reach of children under 6 years old.
- This product should be used only when there is adult supervision present as young children lack necessary judgment regarding safety and the consequences of product misuse.
- This product contains small parts and parts, which are sharp. This product contains electrically conductive parts. Use caution with electrically conductive parts near or around power supplies, batteries and powered (live) circuits.
- When the product is turned ON, activated or tested, some parts will move or rotate. To avoid injuries to hands and fingers, keep them away from any moving parts!
- It is possible that an improperly connected or shorted circuit may cause overheating. Should this happen, immediately disconnect the power supply or remove the batteries and do not touch anything until it cools down! When everything is safe and cool, review the product tutorial to identify the cause.
- Only operate the product in accordance with the instructions and guidelines of this tutorial, otherwise parts may be damaged or you could be injured.
- Store the product in a cool dry place and avoid exposing the product to direct sunlight.
- After use, always turn the power OFF and remove or unplug the batteries before storing.

About Freenove

Freenove provides open source electronic products and services worldwide.

Freenove is committed to assist customers in their education of robotics, programming and electronic circuits so that they may transform their creative ideas into prototypes and new and innovative products. To this end, our services include but are not limited to:

- Educational and Entertaining Project Kits for Robots, Smart Cars and Drones
- Educational Kits to Learn Robotic Software Systems for Arduino, Raspberry Pi and micro: bit
- Electronic Component Assortments, Electronic Modules and Specialized Tools
- **Product Development and Customization Services**

You can find more about Freenove and get our latest news and updates through our website:

http://www.freenove.com

Copyright

FREENOVE®

FREE YOUR INNOVATION

Raspberry Pi® is a trademark of Raspberry Pi Foundation (https://www.raspberrypi.org/).

Contents

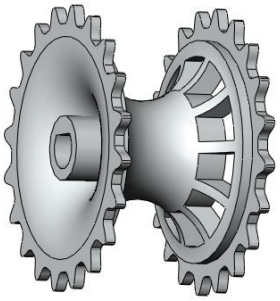
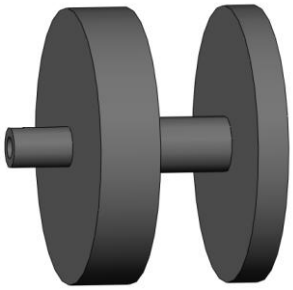
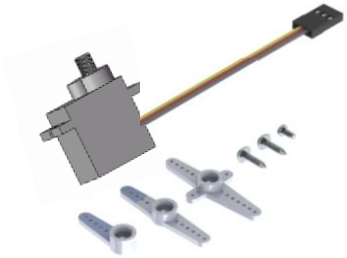
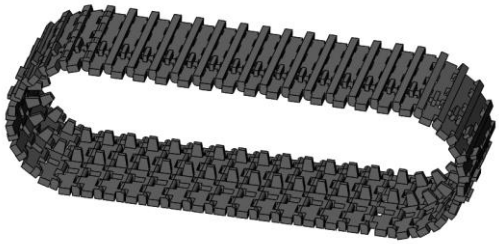
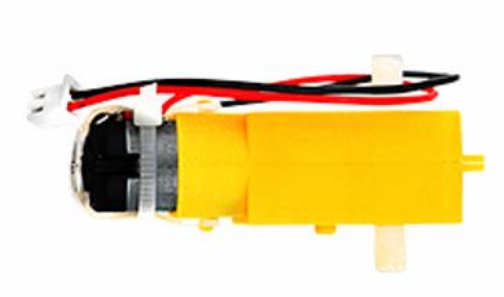
List

If you have any concerns, please feel free to contact us via support@freenove.com

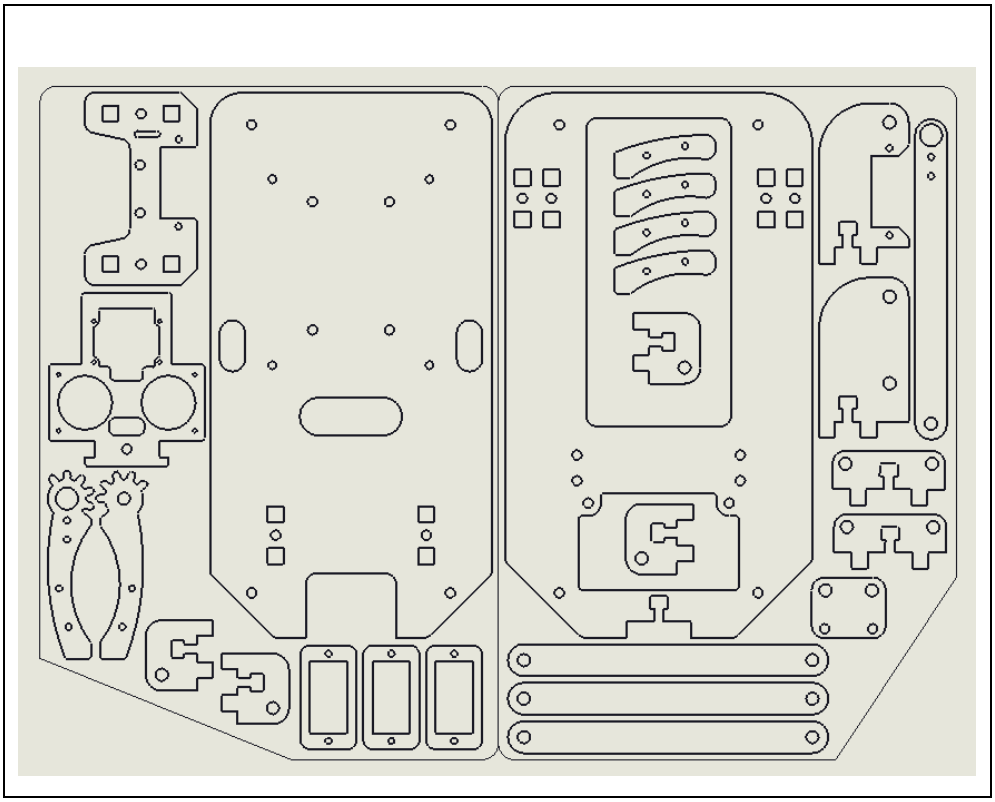
Machinery Parts

M3*24 Brass Standoff  x5 Freenove	M3*10 Brass Standoff  x3 Freenove	M2.5*11 Brass Standoff  x5 Freenove	M2.5*6+6 Screw  x5 Freenove	M4*10 Rivet  x9 Freenove
M3*12 Screw  x13 Freenove	M3*8 Screw  x18 Freenove	M2.5*7 Screw  x9 Freenove	M3*10 Countersunk Head Screw  x6 Freenove	M1.4*5 Screw  x12 Freenove
M2*20 Screw  x10 Freenove	M4*50 Half Tooth Screw  x2 Freenove	M4 Screw  x5 Freenove	M3 Nut  x19 Freenove	M2 Nut  x10 Freenove

Transmission Parts

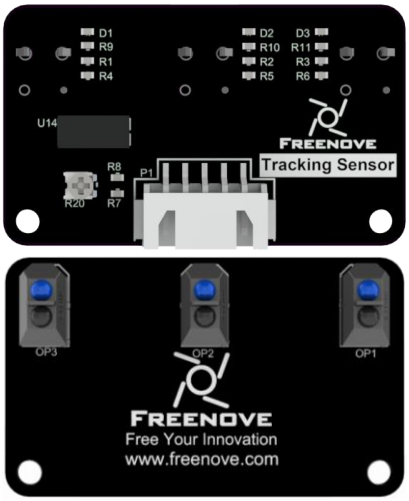
Track drive wheels x2 	Track follower wheels x2 
Servo package x2 	Track x2 
DC speed reduction motor x2 Do NOT remove the cable tie on the motor! 	Motor bracket bag x2 

Acrylic Parts



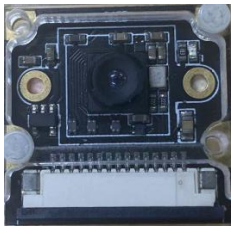
Electronic Parts

Line tracking module x1

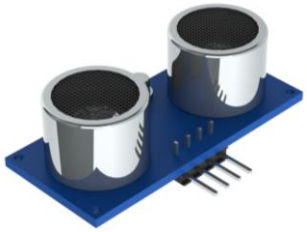


Camera x1 (OV5670 or imx219)

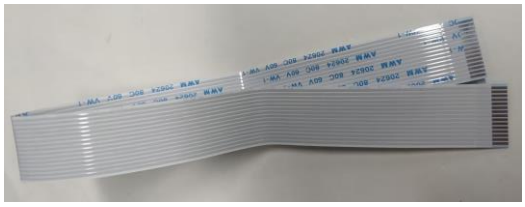
Or



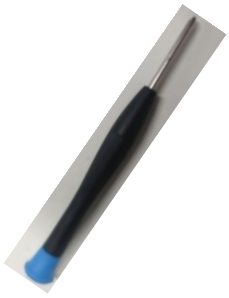
HC-SR04 Ultrasonic Module x1



Jumper Wire F/F(4) x1

XH-2.54-5Pin cable x1 
FPC cable x1 

Tools

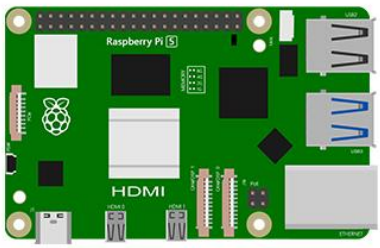
Cross screwdriver (3mm) x1 	Black tape x1 	Cable Tidy x20cm 	Red ball x1 
Cross screwdriver (2mm) x1 	Aluminum alloy coupling x2 	internal hexagonal wrench x1 	

Self-prepared Parts

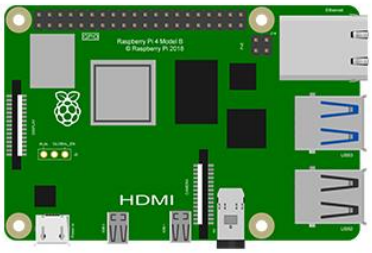
2X 3.7V 18650 lithium **rechargeable** batteries with continuous discharge current >3A.

It is easier to find proper battery on eBay than Amazon. Search "18650 high drain" on eBay.

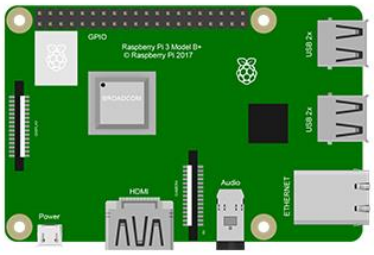
Raspberry Pi (Recommended model: Raspberry 5 / 4B / 3B+) x1



Raspberry Pi 5



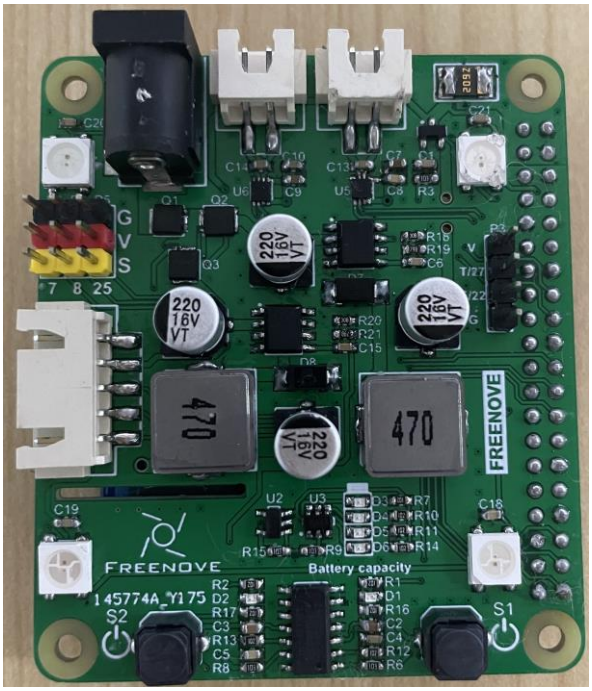
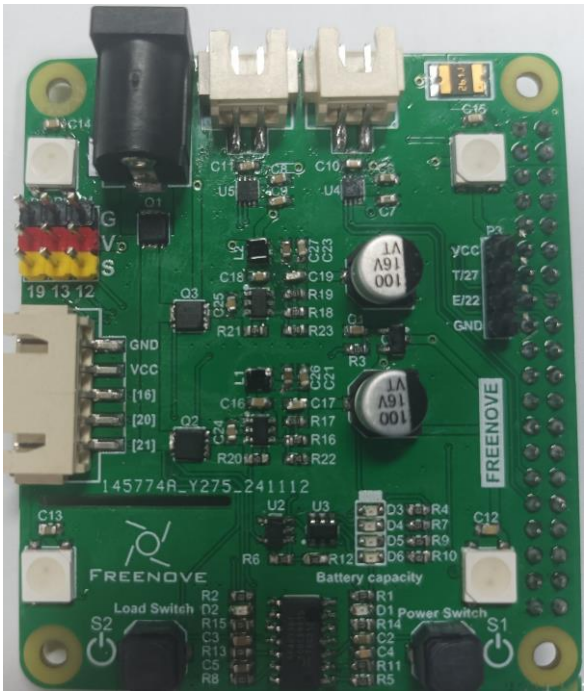
Raspberry Pi 4 Model B



Raspberry Pi 3 Model B+

Tank Smart Car Board for Raspberry Pi

Note: The images of the car board may look different from the one you receive (V1.0 or V2.0). By the way, the interface design of the two version is the same, so the operation on them are the same. You can just follow this book to use it.

Tank Smart Car Board V1.0	Tank Smart Car Board V2.0
	

Preface

Welcome to use Freenove Tank Kit for Raspberry Pi. Following this tutorial, you can make a very cool smart car with many functions.

This kit is based on Raspberry Pi, a popular control panel, so you can share and exchange your experience and design ideas with many enthusiasts all over the world. The parts in this kit include all electronic components, modules, and mechanical components required for making the smart car. And all of them are packaged individually. There are detailed assembly and commissioning instructions in this book.

And if you encounter any problems, please feel free to contact us for fast and free technical support.

support@freenove.com

The contents in this book can help enthusiasts with little technical knowledge to make a smart car. If you are very interested in Raspberry Pi, and want to learn how to program and build the circuit, please visit our website www.freenove.com or contact us to buy the kits designed for beginners:

Freenove Basic\LCD1602\Super\Ultrasonic\RFID\Ultimate Starter Kit for Raspberry Pi

Need support? ✉ support.freenove.com

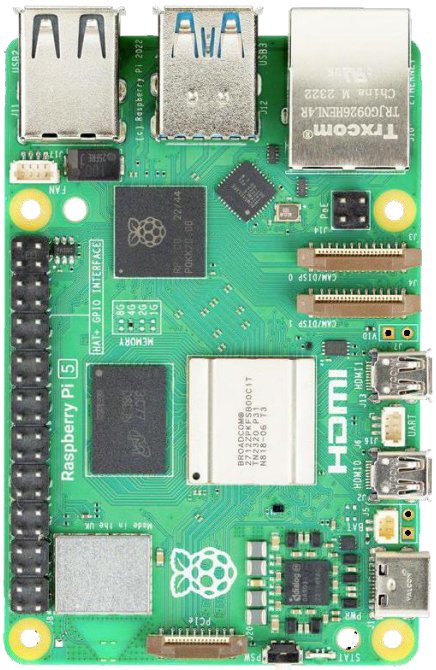
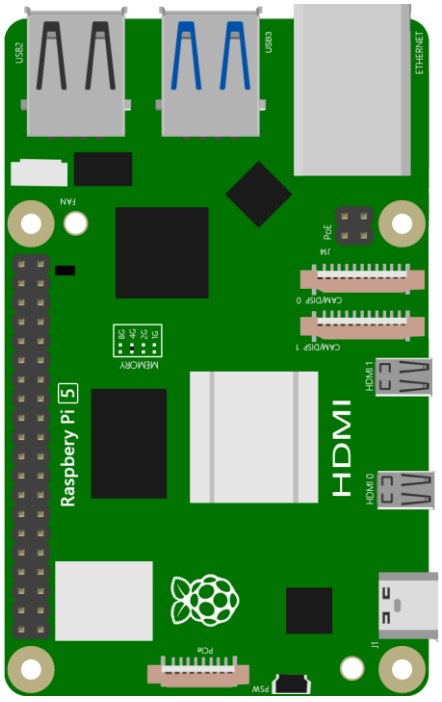
Raspberry Pi Introduction

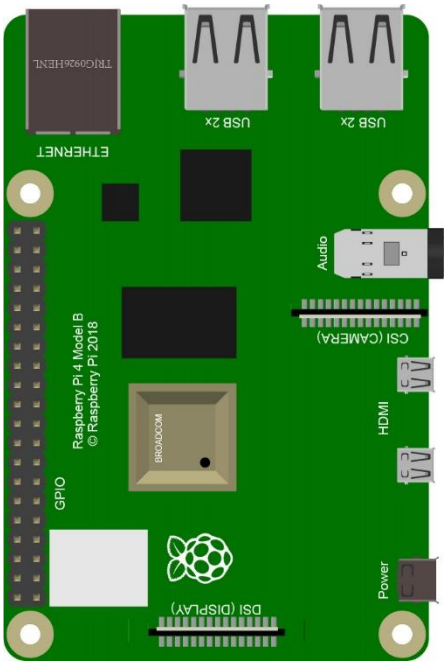
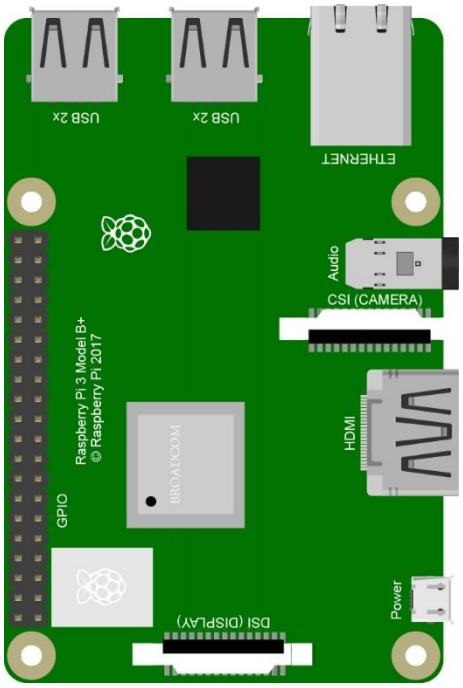
Raspberry Pi (called RPi, RPI, RasPi, these words will be used alternately later), a micro-computer with size of a card, quickly swept the world since its debut. It is widely used in desktop workstation, media center, smart home, robots, and even the servers, etc. It can do almost anything, which continues to attract fans to explore it. Raspberry Pi used to be running with Linux system and along with the release of windows 10 IoT. We can also run it with Windows. Raspberry Pi (with interfaces USB, network, HDMI, camera, audio, display and GPIO), as a microcomputer, can be running in command line mode and desktop system mode. Additionally, it is easy to operate just like Arduino, and you can even directly operate the GPIO of CPU.

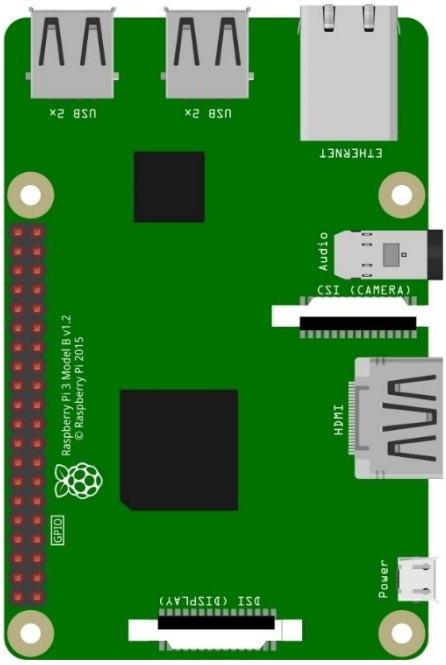
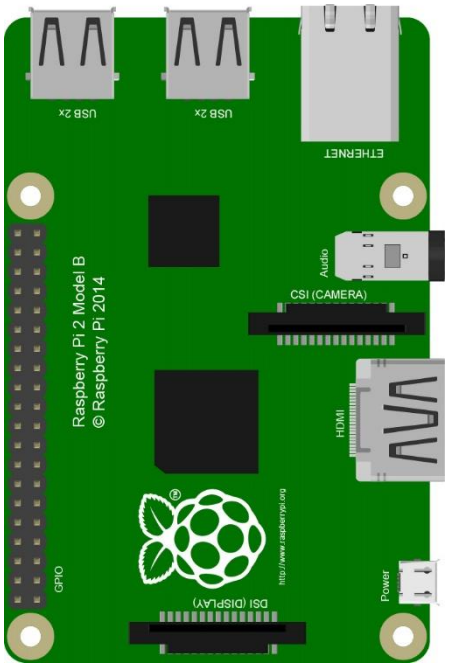
So far, at this writing, Raspberry Pi has advanced to its fifth generation product offering. Version changes are accompanied by increases in upgrades in hardware and capabilities.

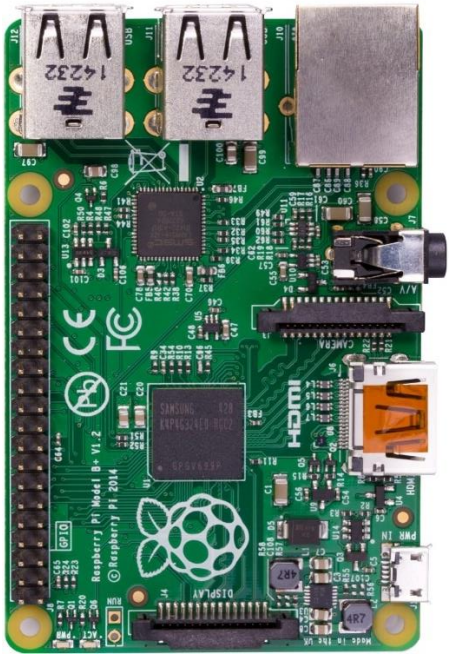
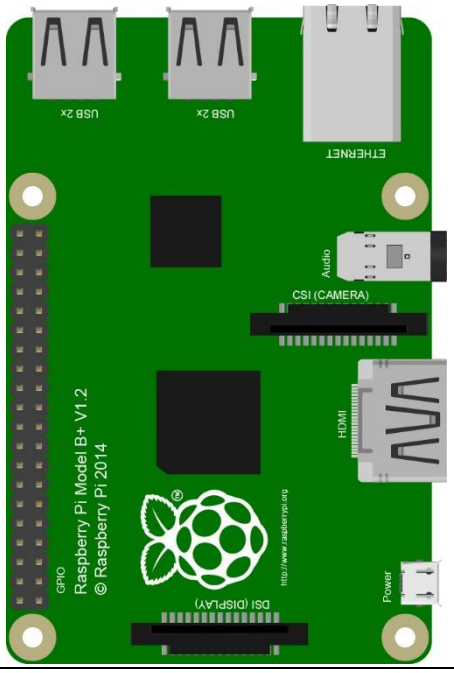
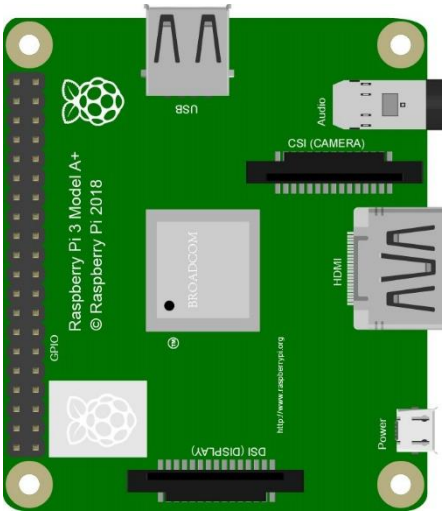
The A type and B type versions of the first generation products have been discontinued due to various reasons. What is most important is that other popular and currently available versions are consistent in the order and number of pins and their assigned designation of function, making compatibility of peripheral devices greatly enhanced between versions.

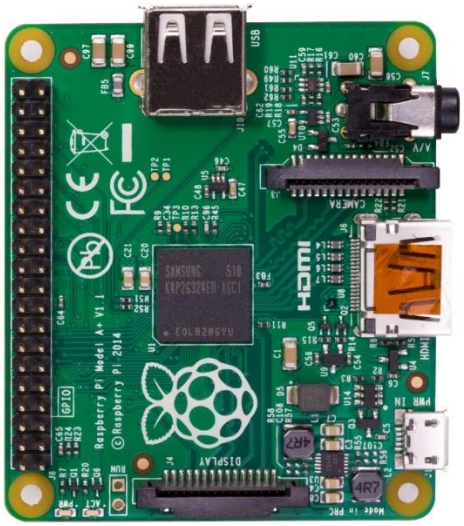
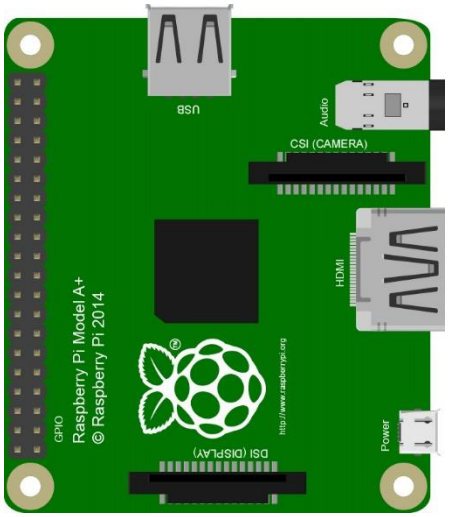
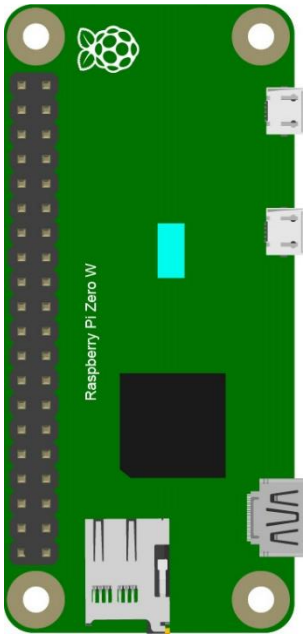
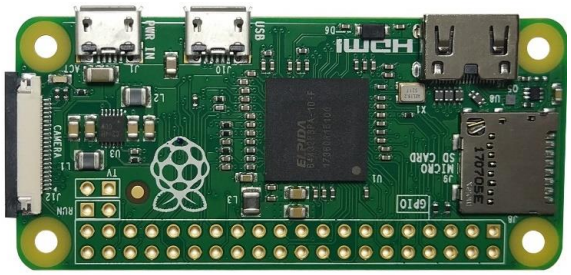
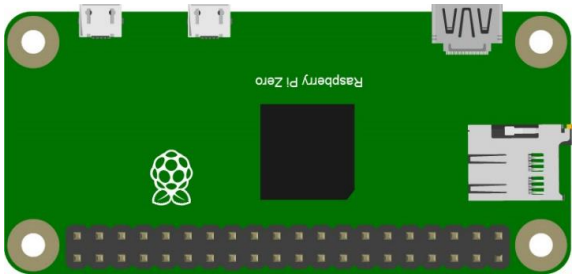
Below are the raspberry pi pictures and model pictures supported by this product. They have 40 pins.

Physical image of Raspberry Pi 5:	CAD image of Raspberry Pi 5:
	

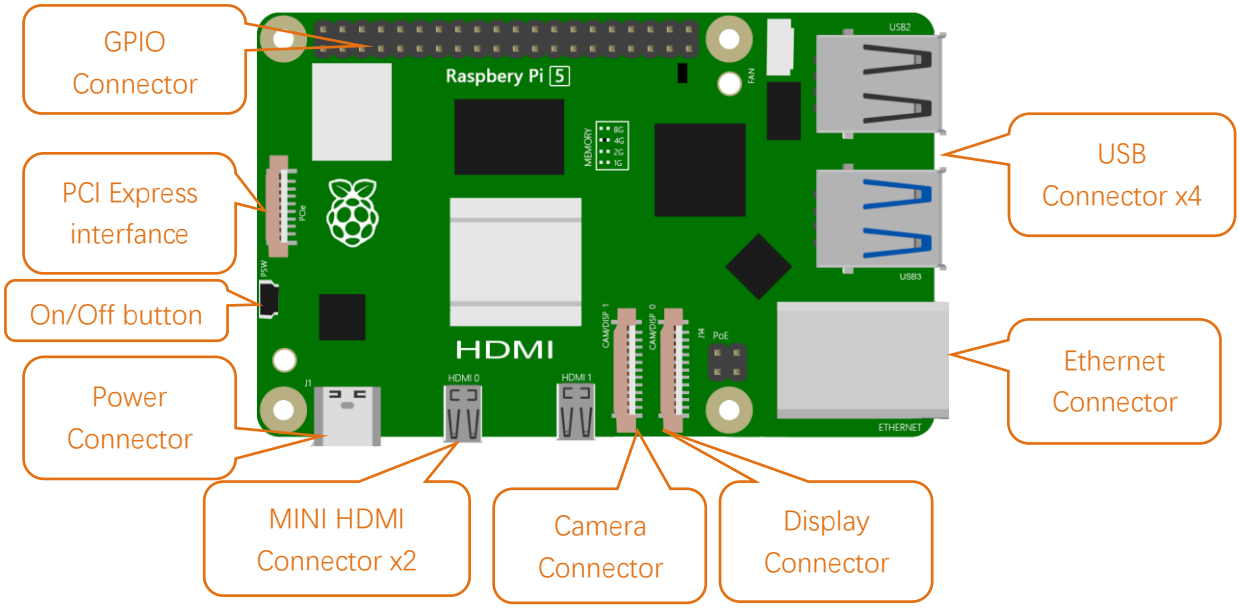
Physical image of Raspberry Pi 4 Model B:	CAD image of Raspberry Pi 4 Model B:
	
Physical image of Raspberry Pi 3 Model B+:	CAD image of Raspberry Pi 3 Model B+:
	

Physical image of Raspberry Pi 3 Model B: 	CAD image of Raspberry Pi 3 Model B: 
Physical image of Raspberry Pi 2 Model B: 	CAD image of Raspberry Pi 2 Model B: 

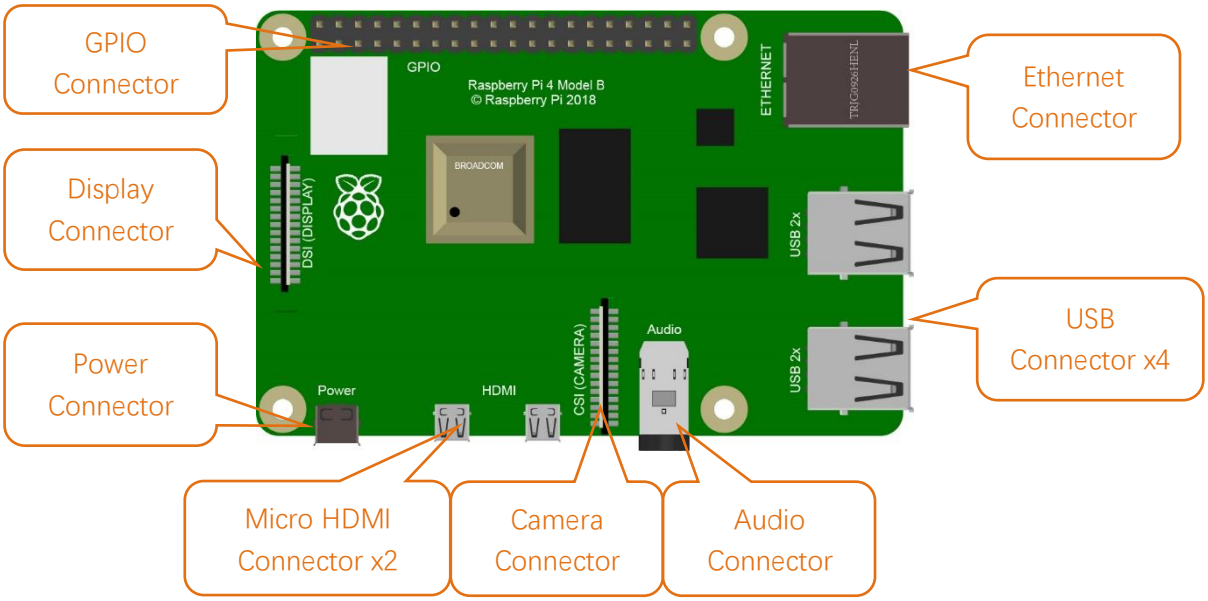
Physical image of Raspberry Pi 1 Model B+: 	CAD image of Raspberry Pi 1 Model B+: 
Physical image of Raspberry Pi 3 Model A+: 	CAD image of Raspberry Pi 3 Model A+: 

Physical image of Raspberry Pi 1 Model A+:	CAD image of Raspberry Pi 1 Model A+:
	
Physical image of Raspberry Pi Zero W:	CAD image of Raspberry Pi Zero W:
	
Physical image of Raspberry Pi Zero:	CAD image of Raspberry Pi Zero:
	

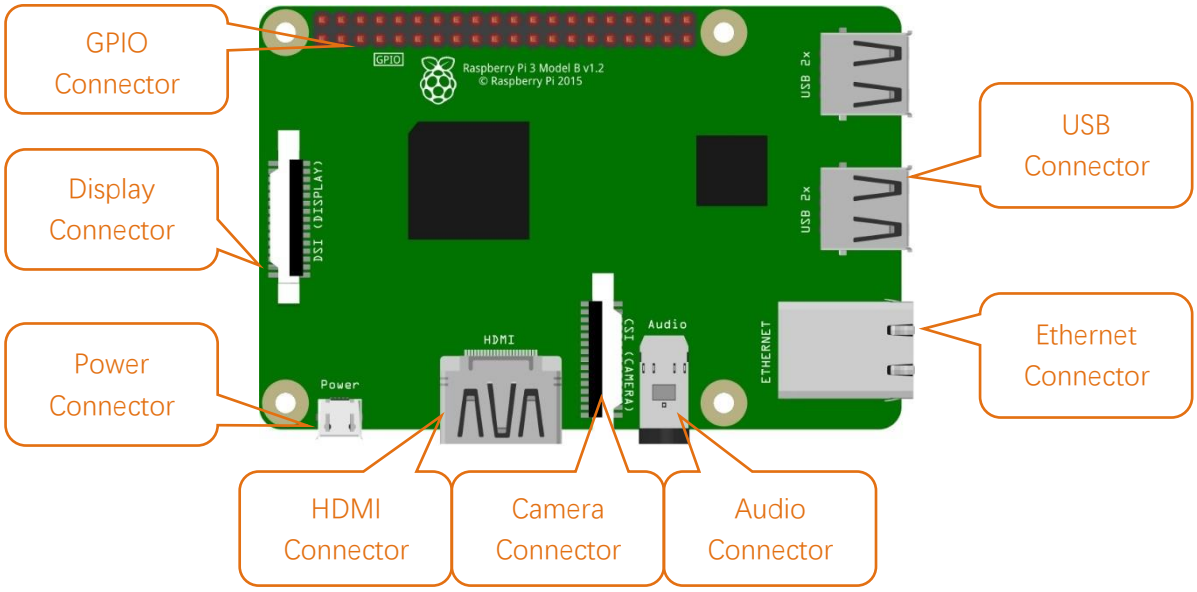
Below are the raspberry pi pictures and model pictures supported by this product. They have 40 pins.
Hardware interface diagram of RPi 5 is shown below:



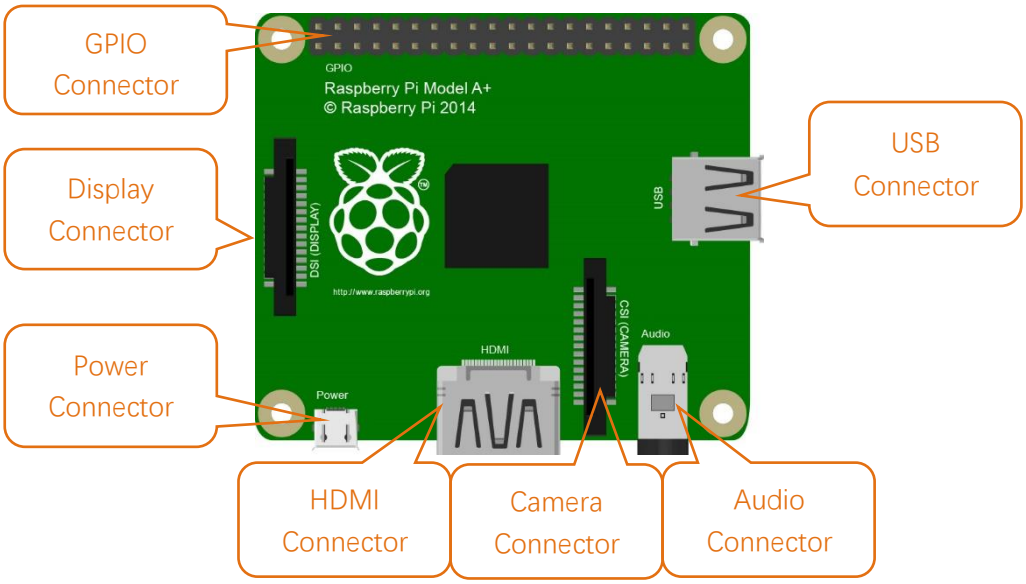
Hardware interface diagram of RPi 4B is shown below:



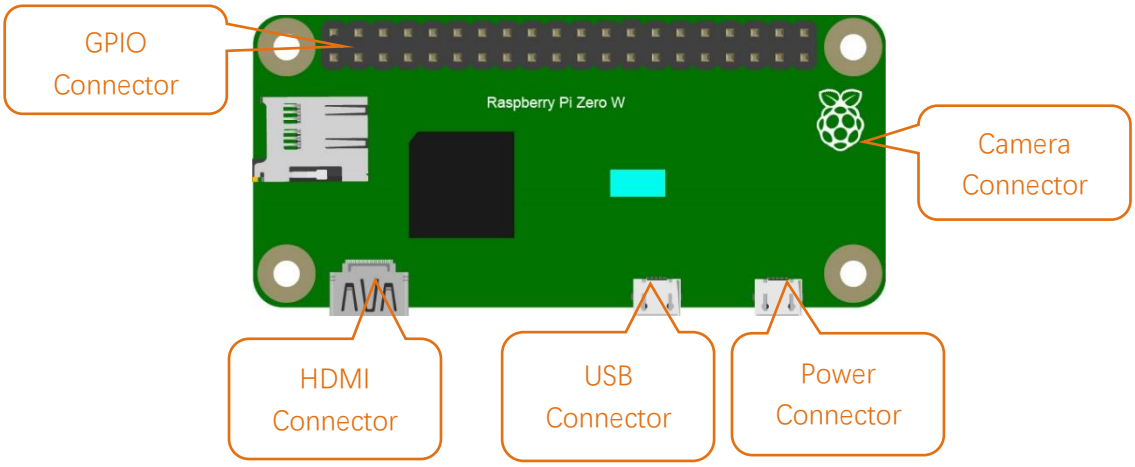
Hardware interface diagram of RPi 3B+/3B/2B/1B+:



Hardware interface diagram of RPi 3A+/A+:



Hardware interface diagram of RPi Zero/Zero W:

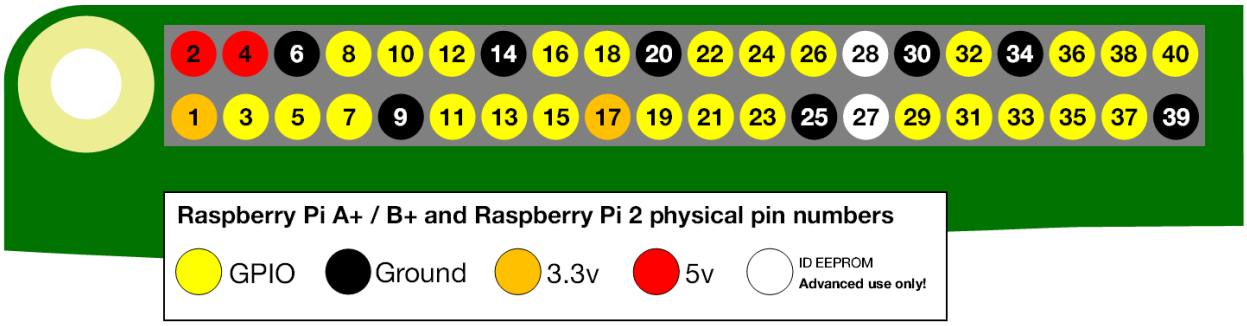


GPIO

GPIO: General purpose input/output. We will introduce the specific feature of the pins on the Raspberry Pi and what you can do with them. You can use them for all sorts of purposes. Most of them can be used as either inputs or outputs, depending on your program.
When programming the GPIO pins there are 3 different ways to refer to them: GPIO numbering, physical numbering, WiringPi GPIO Numbering.

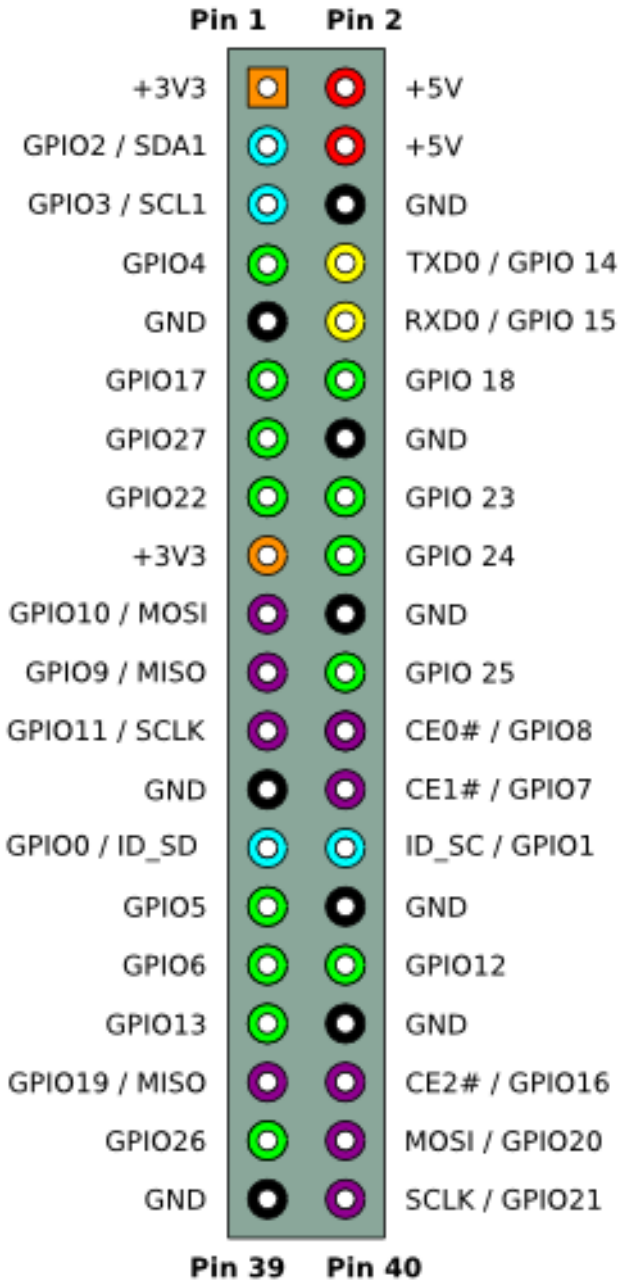
PHYSICAL Numbering

Another way to refer to the pins is by simply counting across and down from pin 1 at the top left (nearest to the SD card). This is 'physical numbering', as shown below:



BCM GPIO Numbering

Raspberry Pi CPU use BCM2835/BCM2836/BCM2837of Broadcom. GPIO pin number is set by chip manufacturer. These are the GPIO pins as that computer recognizes. The numbers are unordered and don't make any sense to humans. You will need a printed reference or a reference board that fits over the pins.
Each pin is defined as below:



For more details about pin definition of GPIO, please refer to http://pinout.xyz/

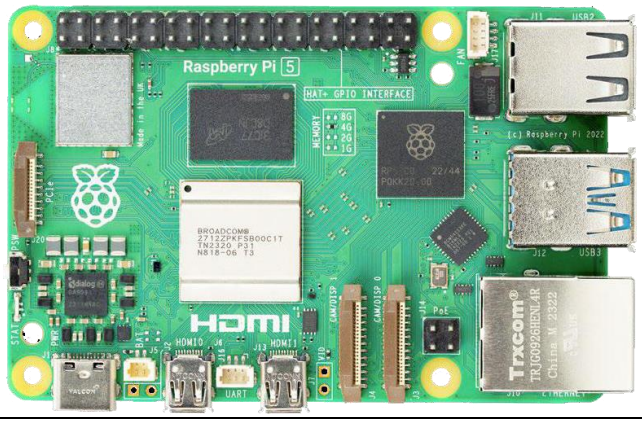
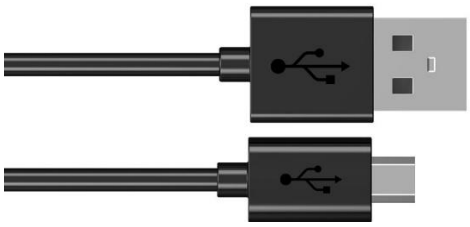
Chapter 0 Raspberry Pi Preparation

Install a System

Firstly, install a system for your RPi.

Component List

Required Components

Raspberry Pi 5 / 4B / 3B+/ 3B /3A+ (Recommended) 	5V/3A Power Adapter. Different versions of Raspberry Pi have different power requirements. 
Micro USB Cable x1 	Micro SD Card (TF Card) x1, Card Reader x1 

This robot also supports the following versions of the Raspberry Pi, but **additional accessories** need to be prepared by yourself.

Raspberry	Additional accessories
Raspberry Pi Zero W	Camera cable(>25cm) for zero w, 15 Pin 1.0mm Pitch to 22 Pin 0.5mm https://www.amazon.com/dp/B076Q595HJ/
Raspberry Pi Zero 1.3	wireless network adapter, Camera cable(>25cm) for zero w, 15 Pin 1.0mm Pitch to 22 Pin 0.5mm, OTG cable (USB Type micro B to USB Type A)
Raspberry Pi 2 Model B	wireless network adapter
Raspberry Pi 1 Model A+	wireless network adapter
Raspberry Pi 1 Model B+	wireless network adapter

Power requirements of various versions of Raspberry Pi are shown in following table:

Product	Recommended PSU current capacity	Maximum total USB peripheral current draw	Typical bare-board active current consumption
Raspberry Pi 1 Model A	700mA	500mA	200mA
Raspberry Pi 1 Model B	1.2A	500mA	500mA
Raspberry Pi 1 Model A+	700mA	500mA	180mA
Raspberry Pi 1 Model B+	1.8A	1.2A	330mA
Raspberry Pi 2 Model B	1.8A	1.2A	350mA
Raspberry Pi 3 Model B	2.5A	1.2A	400mA
Raspberry Pi 3 Model A+	2.5A	Limited by PSU, board, and connector ratings only.	350mA
Raspberry Pi 3 Model B+	2.5A	1.2A	500mA
Raspberry Pi 4 Model B	3.0A	1.2A	600mA
Raspberry Pi 5	5.0A	1.6A (600mA if using a 3A power supply)	800mA
Raspberry Pi 400	3.0A	1.2A	800mA
Raspberry Pi Zero	1.2A	Limited by PSU, board, and connector ratings only	100mA
Raspberry Pi Zero W	1.2A	Limited by PSU, board, and connector ratings only.	150mA
Raspberry Pi Zero 2 W	2A	Limited by PSU, board, and connector ratings only.	350mA

For more details, please refer to https://www.raspberrypi.org/help/faqs/#powerReqs

In addition, RPi also needs an Ethernet network cable used to connect it to a WAN (Wide Area Network).

The Raspberry Pi 5 provides 1.6A of power to downstream USB peripherals when connected to a power supply capable of 5A at +5V (25W). When connected to any other compatible power supply, the Raspberry Pi 5 restricts downstream USB devices to 600mA of power.

Optional Components

Under normal circumstances, there are two ways to login to Raspberry Pi:
1) Using independent monitor, or remote desktop to share a monitor with your PC.
2) Using a remote desktop or laptop computer monitor "sharing" the PC monitor with your RPi.

Required Accessories for Monitor

If you want to use independent monitor, mouse and keyboard, you also need the following accessories.
1.Display with HDMI interface
2.Mouse and Keyboard with USB interface

As to Pi Zero and Pi Zero W, you also need the following accessories.

1. Micro-HDMI to HDMI converter wire.
2. Micro-USB to USB-A Receptacles converter wire (Micro USB OTG wire).
3. USB HUB.
4. USB transferring to Ethernet interface or USB Wi-Fi receiver.

For different Raspberry Pi Modules, the optional items may vary slightly but they all aim to convert the interfaces to Raspberry Pi standards.

<table>
<tr><th></th><th>Pi Zero</th><th>Pi A+</th><th>Pi Zero W</th><th>Pi 3A+</th><th>Pi B+/2B</th><th>Pi 3B/3B+</th><th>Pi 4B</th><th>Pi 5</th></tr>
<tr><td>Monitor</td><td colspan="8">Yes (All)</td></tr>
<tr><td>Mouse</td><td colspan="8">Yes (All)</td></tr>
<tr><td>Keyboard</td><td colspan="8">Yes (All)</td></tr>
<tr><td>Micro-HDMI to HDMI Adapter & Cable</td><td>Yes</td><td>No</td><td>Yes</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td></tr>
<tr><td>Micro-HDMI to HDMI Adapter & Cable</td><td colspan="6">No</td><td colspan="2">Yes</td></tr>
<tr><td>Micro-USB to USB-A Adapter & Cable (Micro USB OTG Cable)</td><td>Yes</td><td>No</td><td>Yes</td><td colspan="5">No</td></tr>
<tr><td>USB HUB</td><td>Yes</td><td>Yes</td><td>Yes</td><td>Yes</td><td>No</td><td>No</td><td>No</td><td>No</td></tr>
<tr><td>USB to Ethernet Interface</td><td colspan="2" rowspan="2">select one from two or select two from two</td><td colspan="2">optional</td><td>Internal Integration</td><td colspan="3" rowspan="2">Internal Integration</td></tr>
<tr><td>USB Wi-Fi Receiver</td><td colspan="2">Internal Integration</td><td>optional</td></tr>
</table>

Required Accessories for Remote Desktop

If you do not have an independent monitor, or if you want to use a remote desktop, you first need to login to Raspberry Pi through SSH, and then open the VNC or RDP service. This requires the following accessories.

	Pi Zero	Pi Zero W	Pi A+	Pi 3A+ Pi B+/2B Pi 3B/3B+/4B/5
Micro-USB to USB-A Adapter & Cable (Micro USB OTG Cable)	Yes	Yes	No	NO
USB to Ethernet interface	Yes	Yes	Yes	

Raspberry Pi OS

Headless Installation - on Windows:
https://youtu.be/XpiT_ezb_7c
Headless Installation – on macOS
https://youtu.be/I1zRHp3Deeg

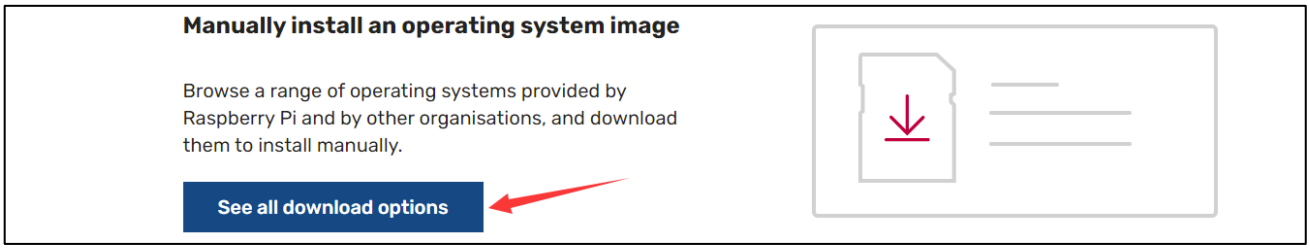
Automatically Method

You can follow the official method to install the system for raspberry pi via visiting link below:
https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/2
In this way, the system will be downloaded **automatically** via the application.

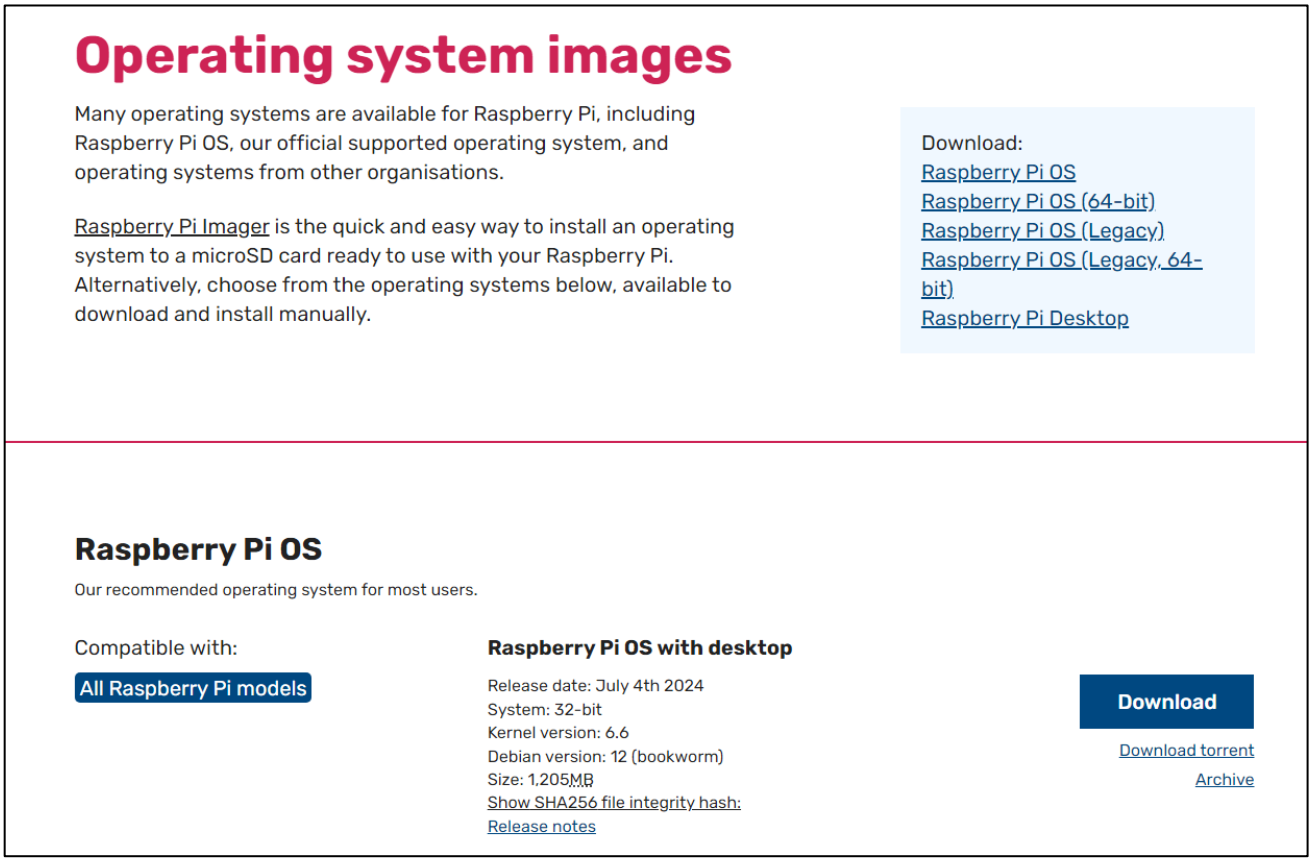
Manually Method

After installing the Imager Tool in the **link above**. You can **also** download the system **manually** first.

Visit https://www.raspberrypi.org/downloads/



You can download the latest system image. The following is just for your reference.

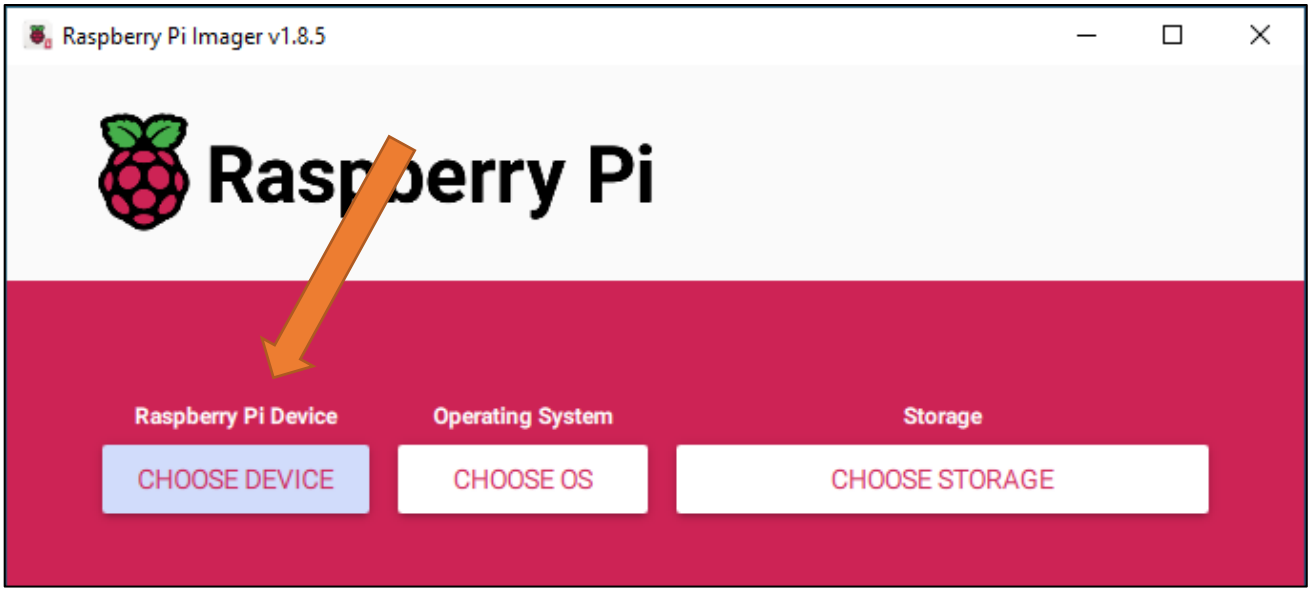


And then the zip file is downloaded.

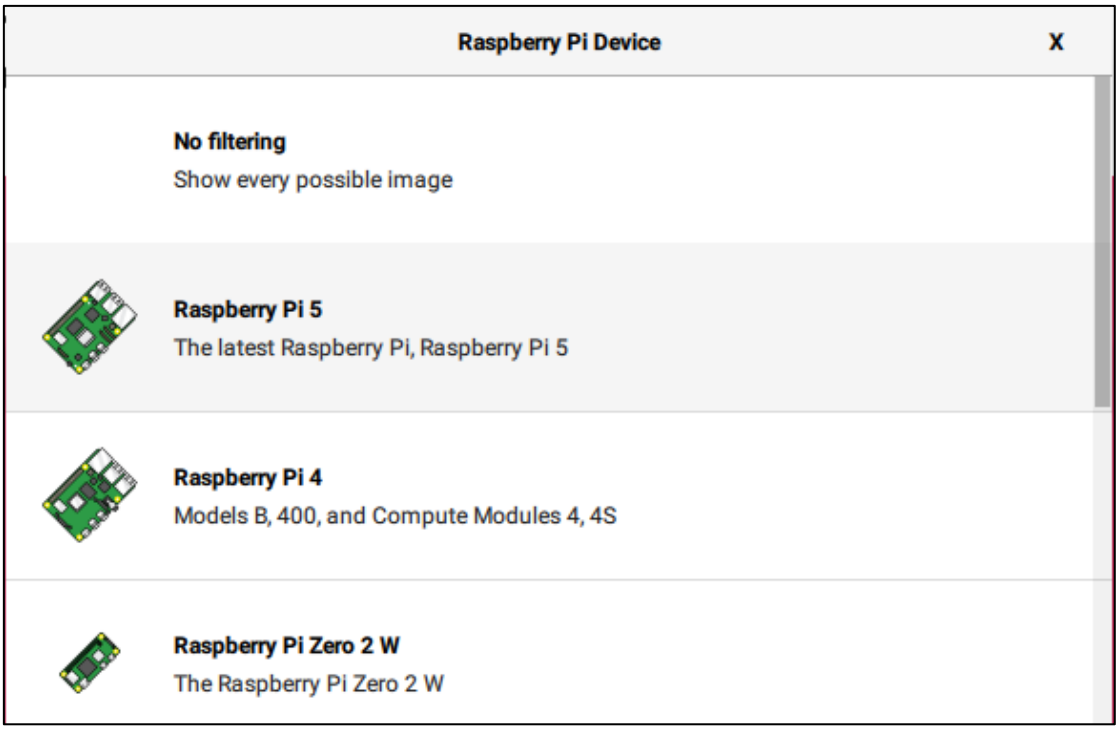
Write System to Micro SD Card

First, put your Micro **SD card** into card reader and connect it to USB port of PC.

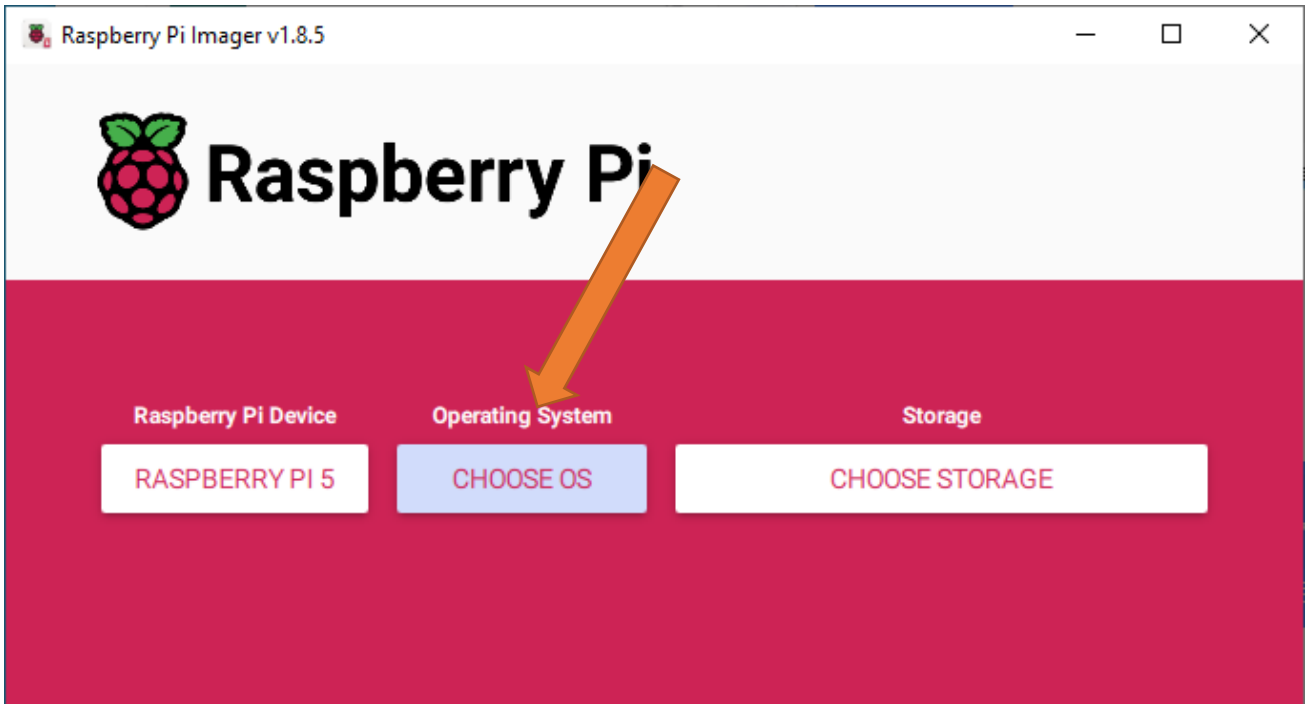
Then open imager toll. Clicked Choose Device.



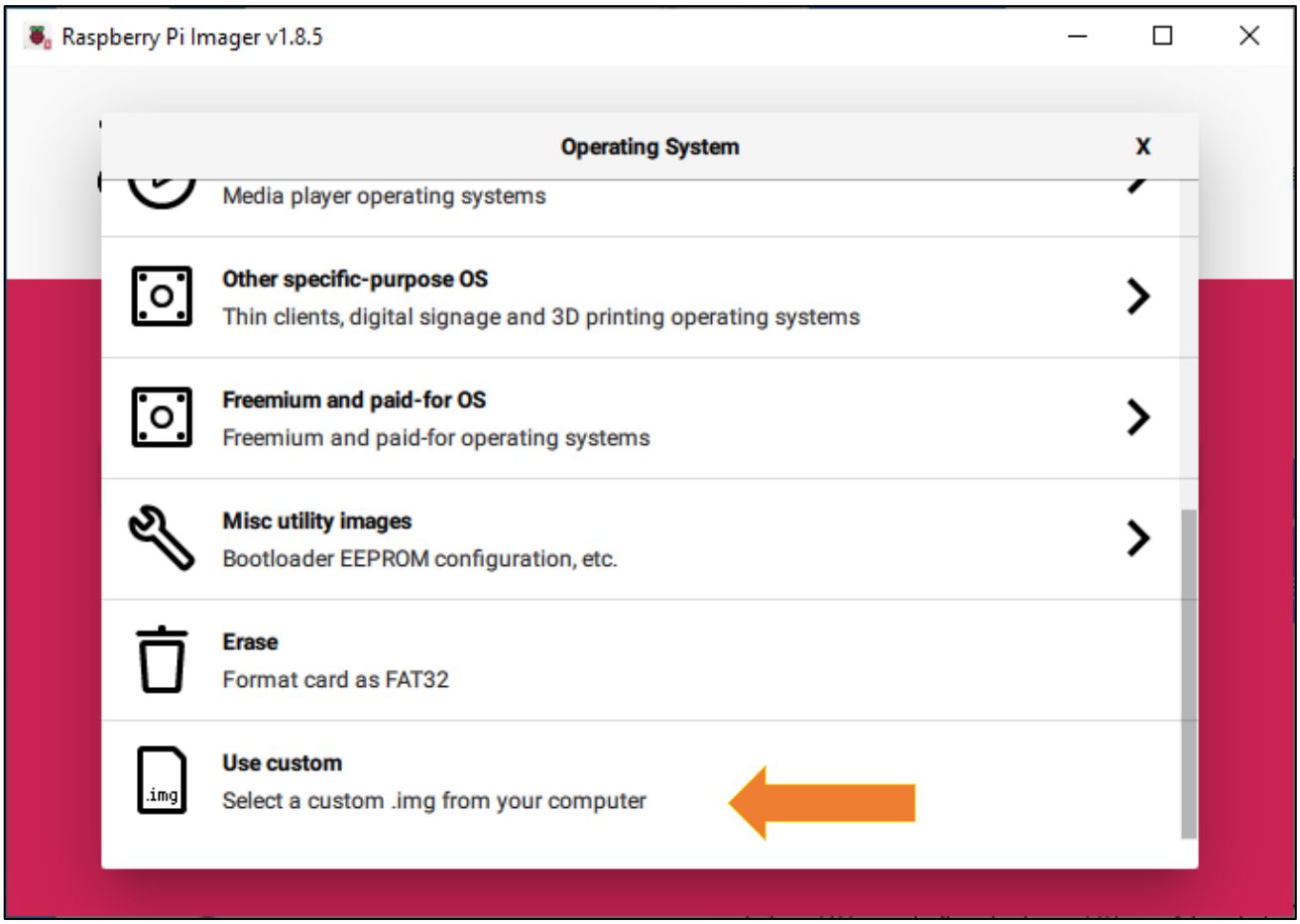
Select a Raspberry PI Device based on your Raspberry PI version. It will help us filter out the right version of the system for the Raspberry PI.



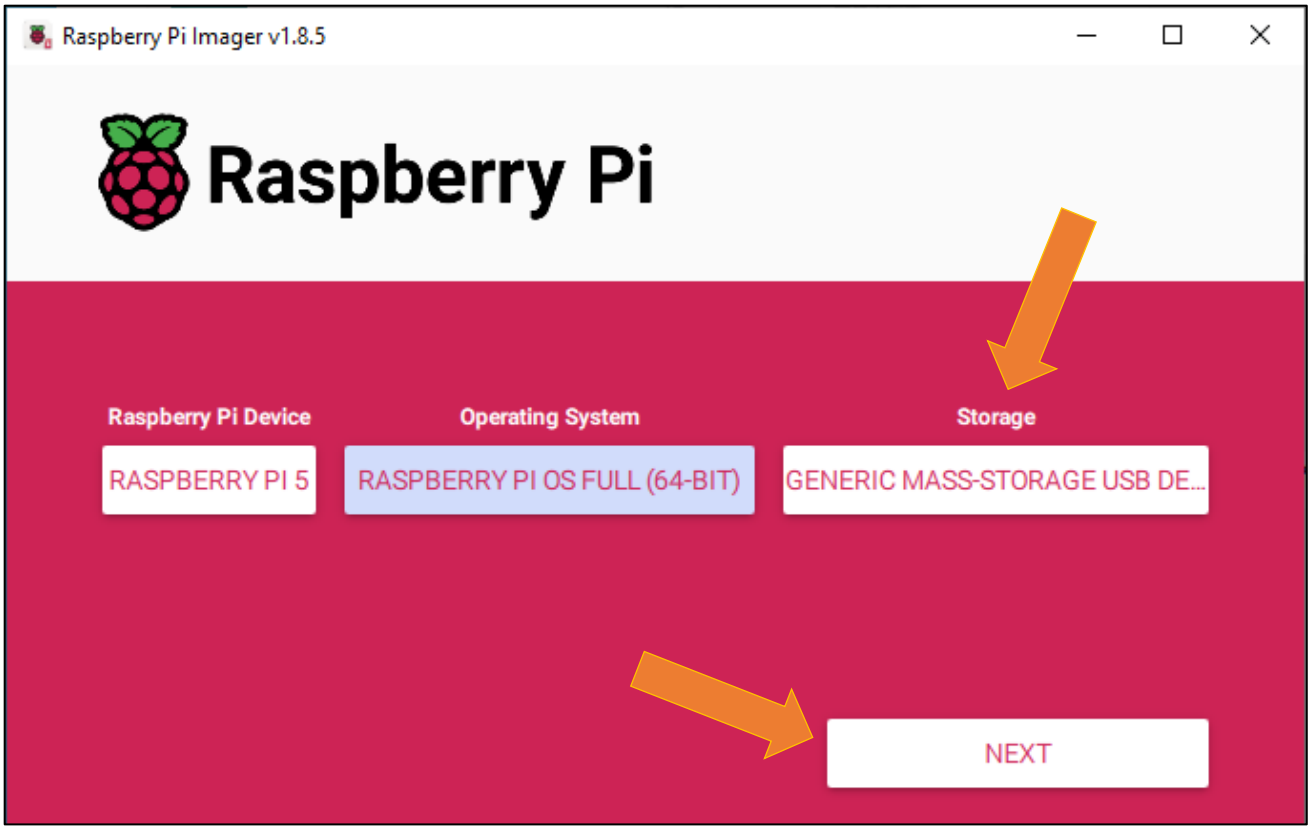
Clicked Operating System.



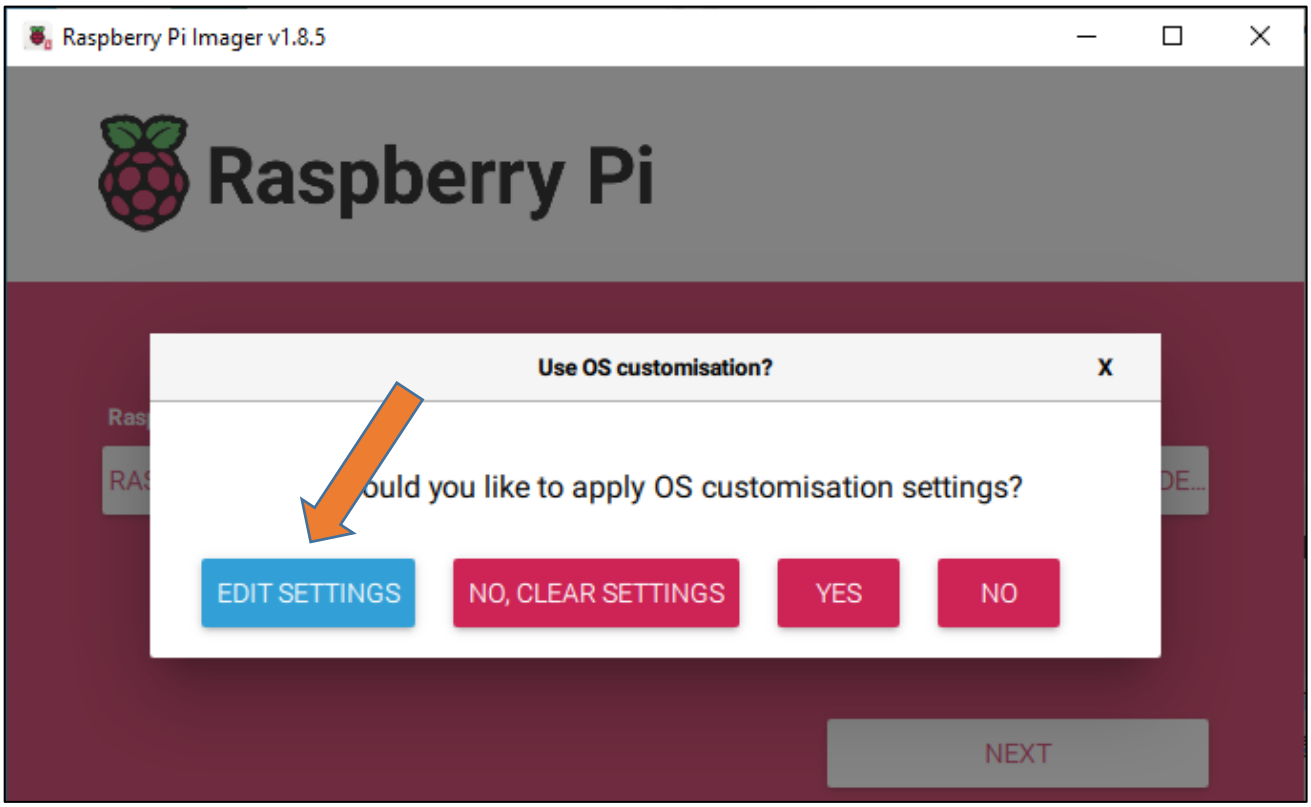
Choose system that you just downloaded in Use custom.



Choose the SD card. Then click “Next”.

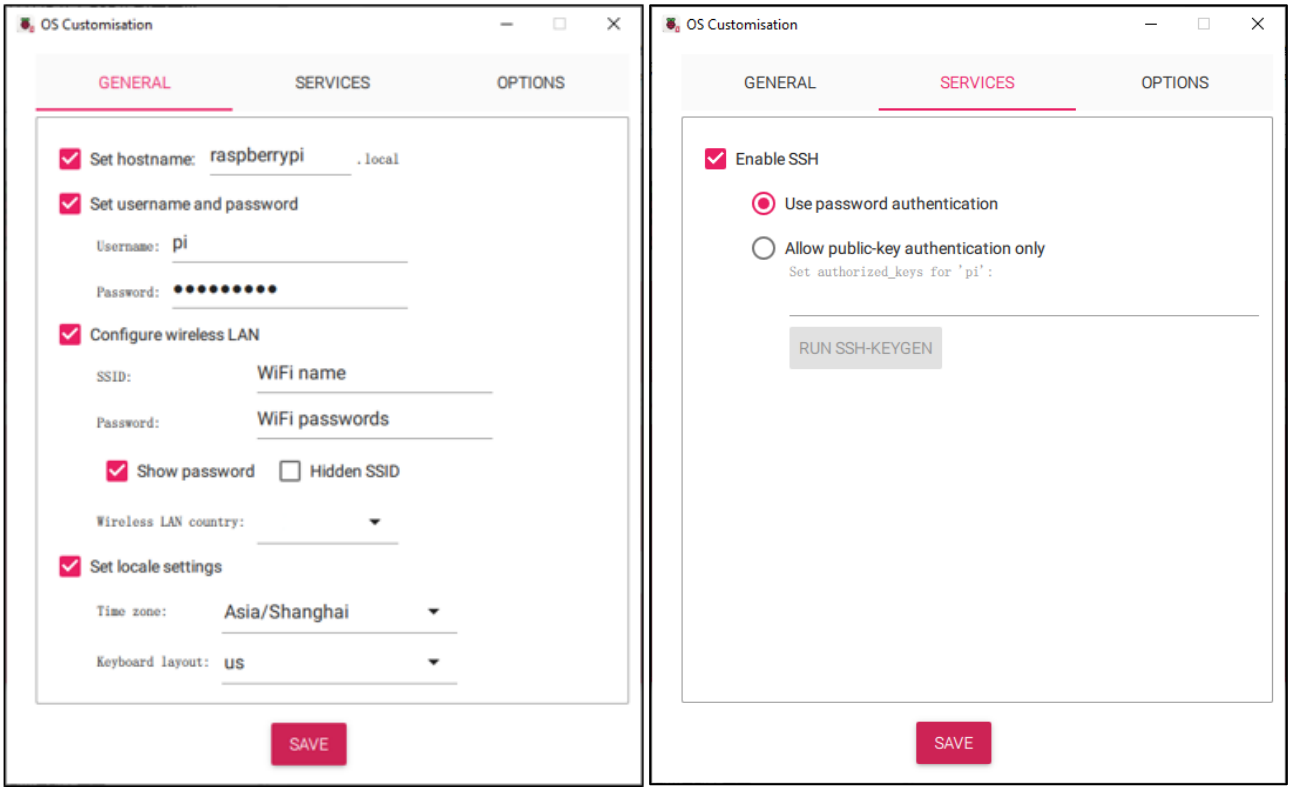


You can configure the Raspberry PI according to your needs.

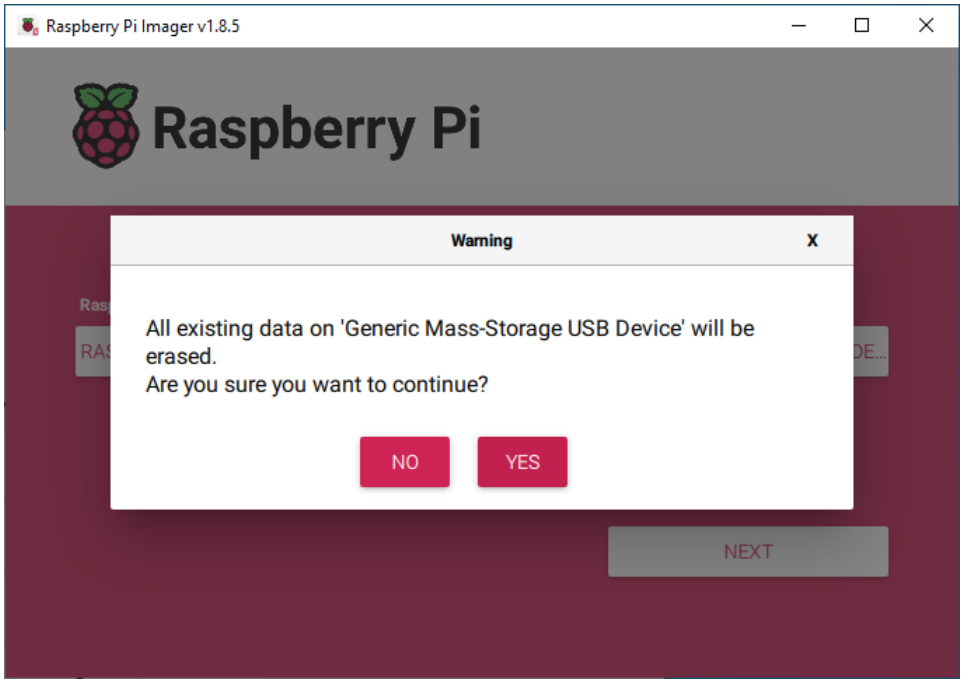


Enable ssh and configure WiFi

On the GENERAL screen, configure your information based on your actual situation.
Enable SSH on the SERVICES page.

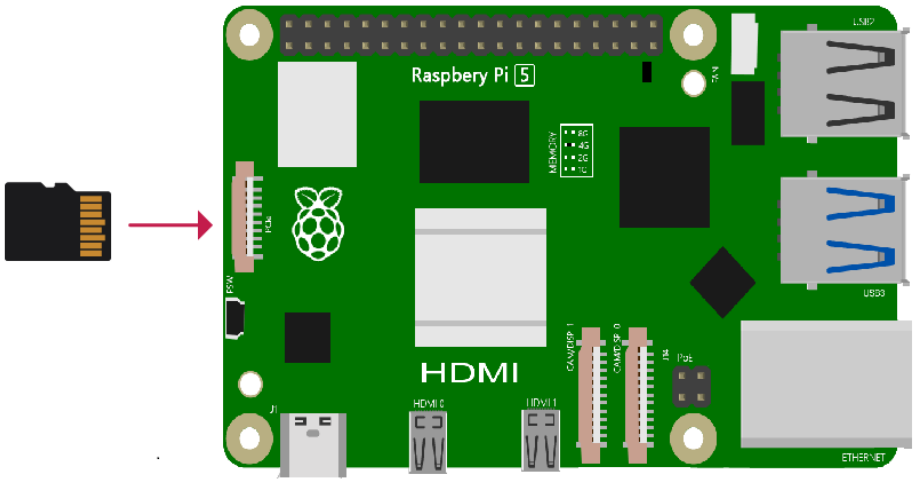


Click Save, in the new screen, click Yes, wait for SD to brush into the Raspberry system.



Insert SD card

Then remove SD card from card reader and insert it into Raspberry Pi.



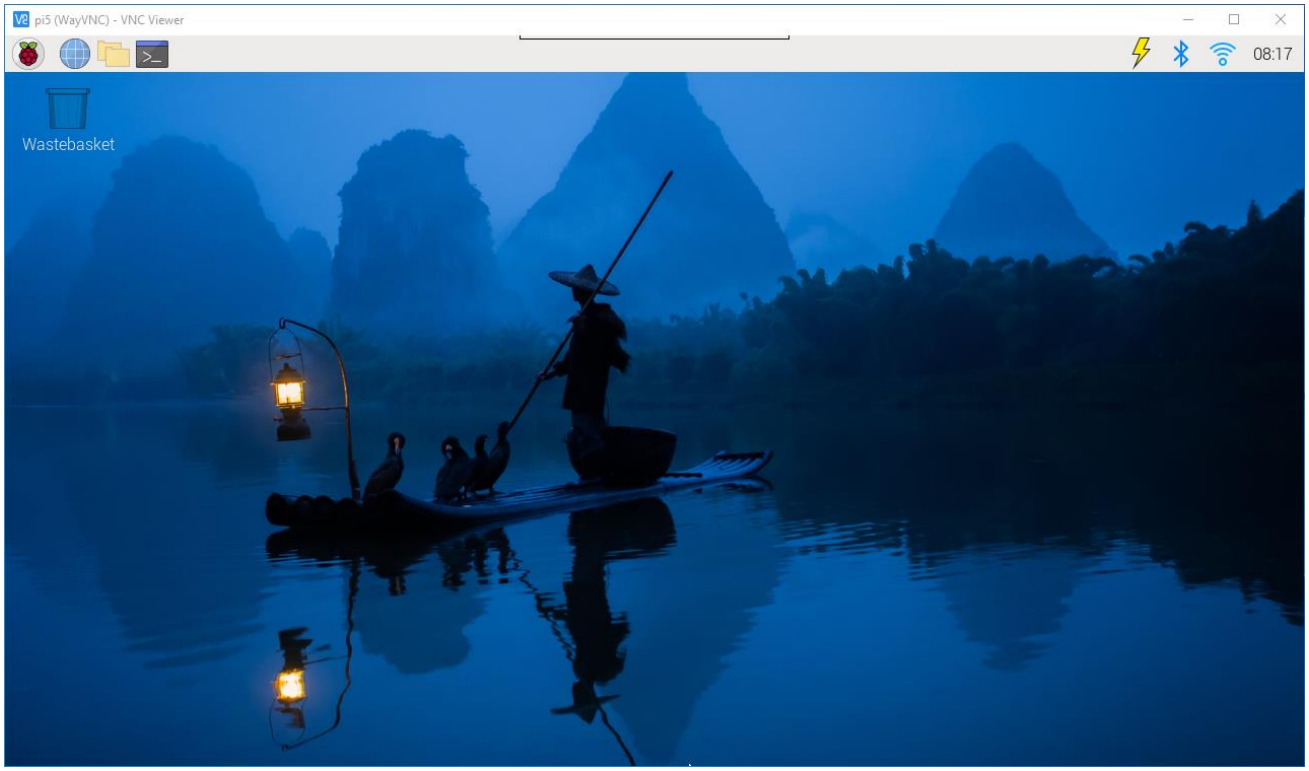
Connect to the power supply and wait for the Raspberry PI to turn on.

Getting Started with Raspberry Pi

Monitor desktop

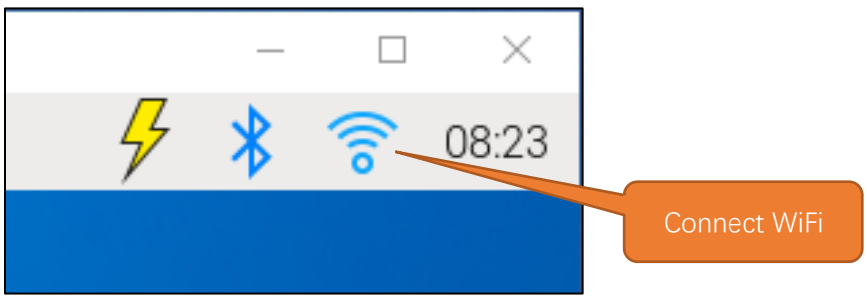
If you do not have a spare monitor, please skip to next section Remote desktop & VNC. If you have a spare monitor, please follow the steps in this section.

After the system is written successfully, take out Micro SD Card and put it into the SD card slot of RPi. Then connect your RPi to the monitor through the HDMI port, attach your mouse and keyboard through the USB ports, attach a network cable to the network port and finally, connect your power supply (making sure that it meets the specifications required by your RPi Module Version. Your RPi should start (power up). Later, after setup, you will need to enter your user name and password to login. The default user name: pi; password: raspberry. After login, you should see the following screen.

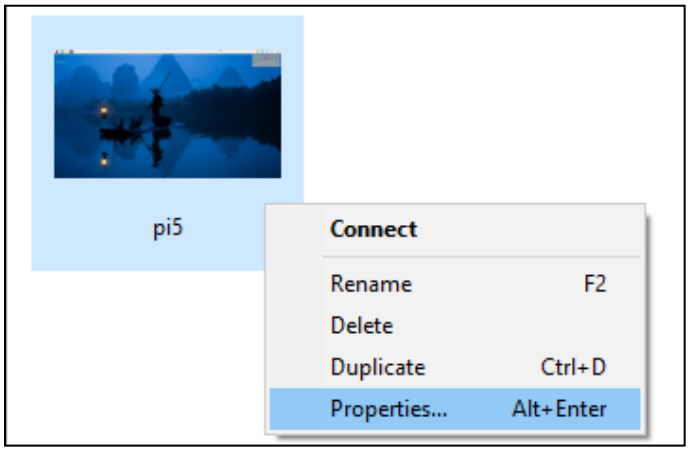


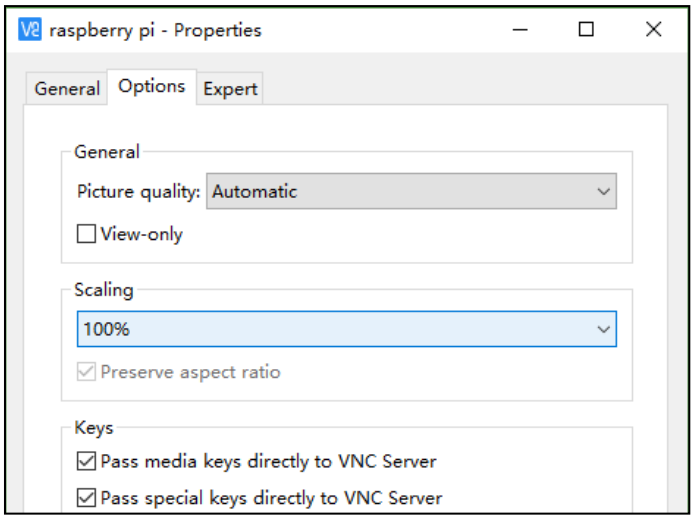
Congratulations! You have successfully installed the RASPBERRY PI OS operating system on your RPi.

Raspberry Pi 5, 4B, 3B+/3B integrates a Wi-Fi adaptor. You can use it to connect to your Wi-Fi. Then you can use the wireless remote desktop to control your RPi. This will be helpful for the following work. Raspberry Pi of other models can use wireless remote desktop through accessing an external USB wireless card.



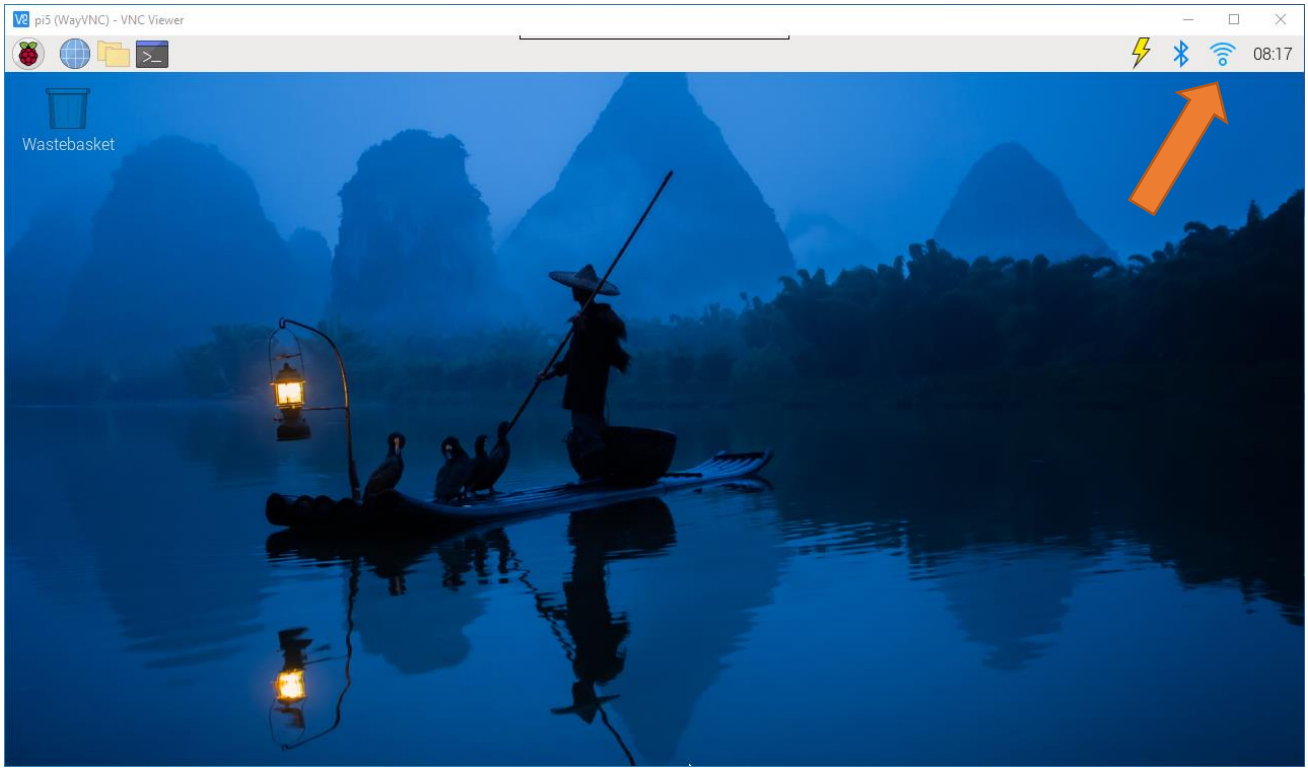
In addition, your VNC Viewer window may zoom your Raspberry Pi desktop. You can change it. On your VNC View control panel, click right key. And select Properties->Options label->Scaling. Then set proper scaling.





Here, you have logged in to Raspberry Pi successfully by using VNC Viewer and operated proper setting.

Raspberry Pi 5/4B/3B+/3B integrates a Wi-Fi adaptor.If you did not connect Pi to WiFi. You can connect it to wirelessly control the robot.


pi5 (WayVNC) - VNC Viewer
08:17
Wastebasket

Remote desktop & VNC

If you have logged in Raspberry Pi via display, you can skip to VNC Viewer.

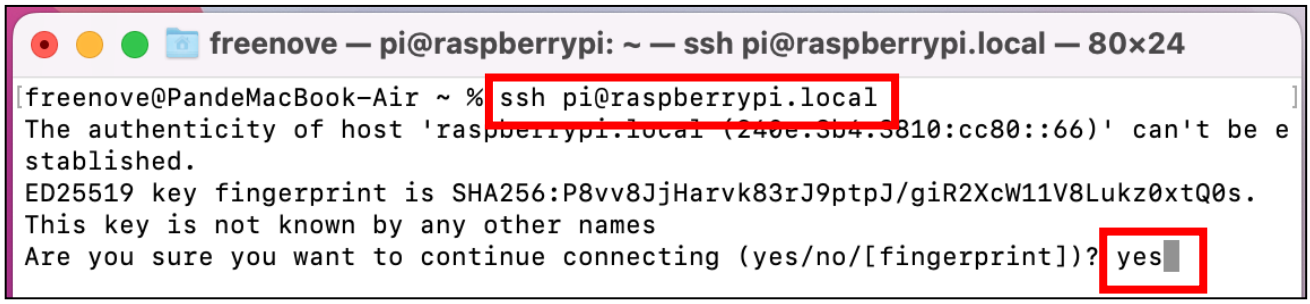
If you don't have a spare display, mouse and keyboard for your RPi, you can use a remote desktop to share a display, keyboard, and mouse with your PC. Below is how to use:
MAC OS remote desktop and Windows OS remote desktop.

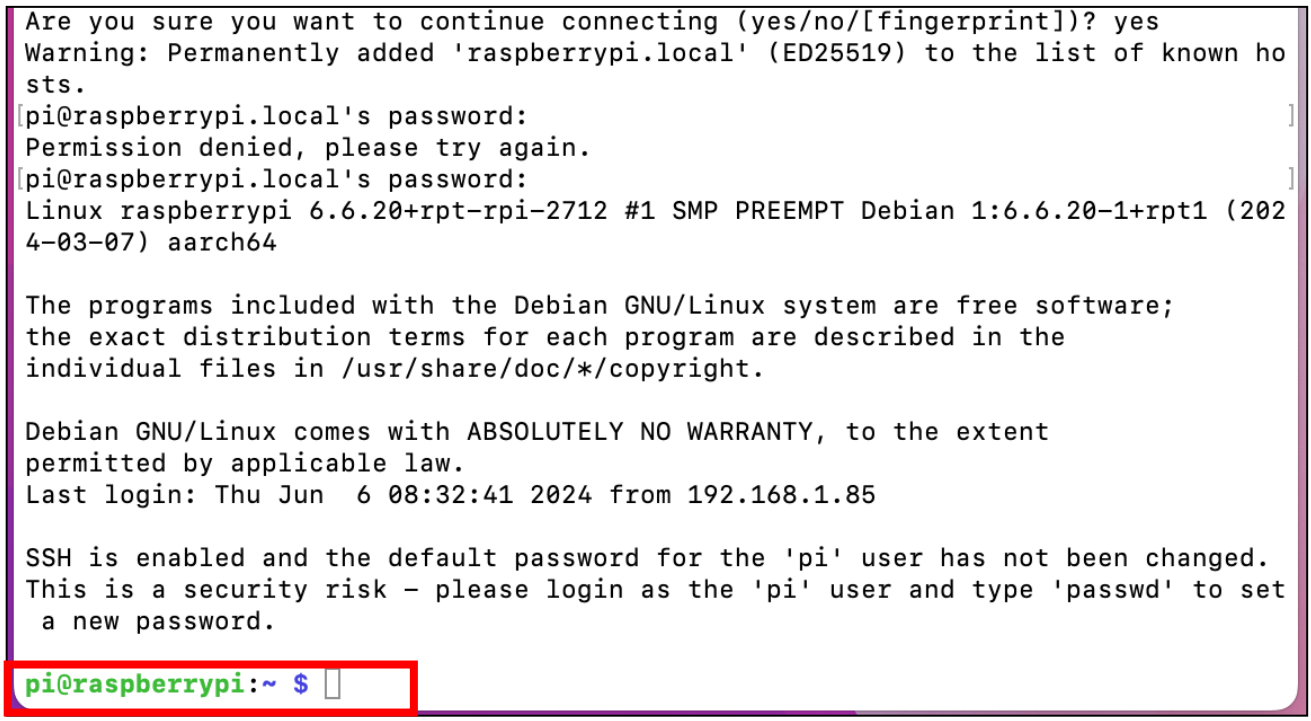
MAC OS Remote Desktop

Open the terminal and type following command. **If this command doesn't work, please move to next page.**

```
ssh pi@raspberrypi.local
```

The password is **raspberry** by default, case sensitive. You may need to type **yes** during the process.





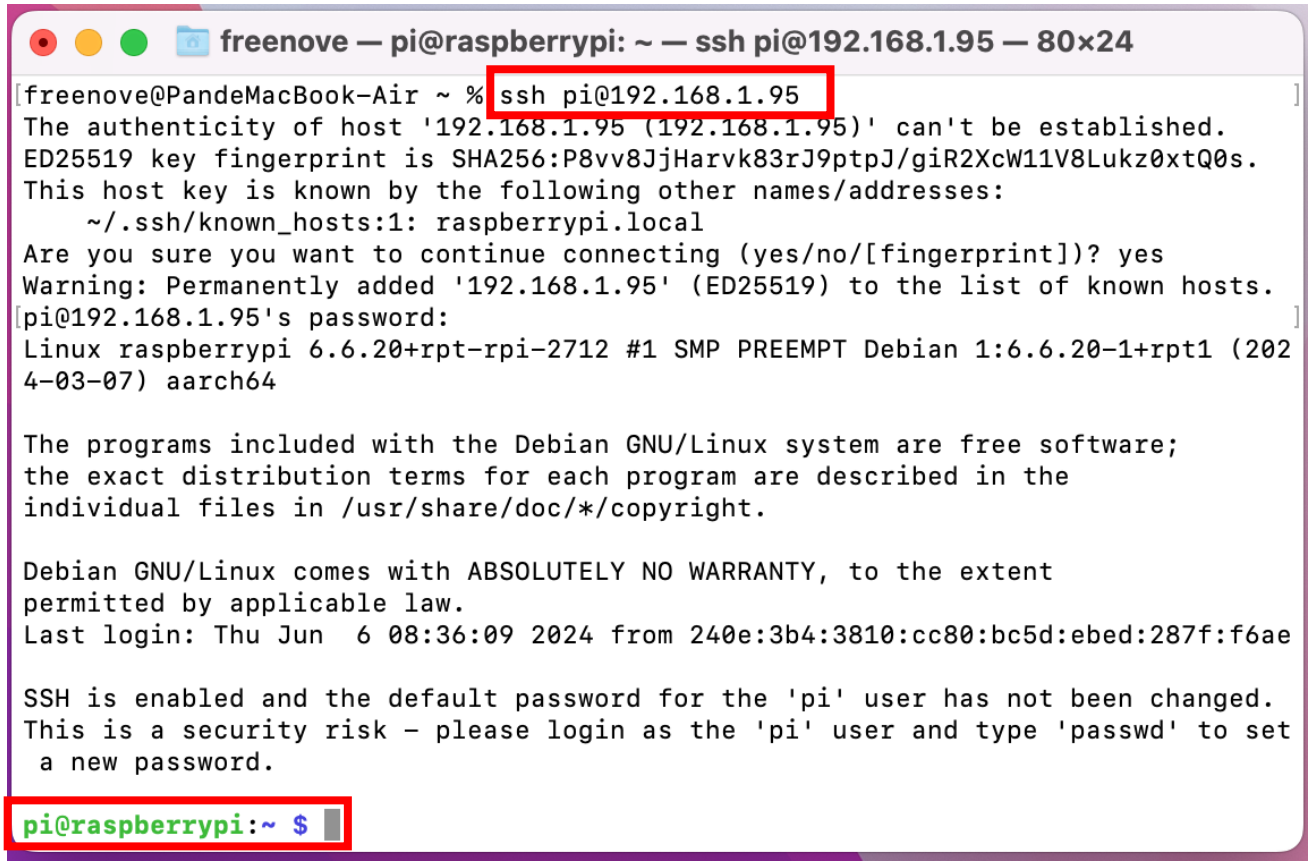
You can also use the IP address to log in Pi.

Enter **router** client to **inquiry IP address** named "raspberry pi". For example, I have inquired to **my RPi IP address, and it is "192.168.1.95".**

Open the terminal and type following command.

```
ssh pi@192.168.1.95
```

When you see pi@raspberrypi:~ $, you have logged in Pi successfully. Then you can skip to next section.



Then you can skip to VNC Viewer.

Windows OS Remote Desktop

If you are using win10, you can use follow way to login Raspberry Pi without desktop.

Press **Win+R**. Enter **cmd**. Then use this command to check IP:

```
ping -4 raspberrypi.local
```

```
C:\Users\freenove-14>ping -4 raspberrypi.local

Pinging raspberrypi.local [192.168.1.95] with 32 bytes of data:
Reply from 192.168.1.95: bytes=32 time=29ms TTL=64
Reply from 192.168.1.95: bytes=32 time=5ms TTL=64
Reply from 192.168.1.95: bytes=32 time=6ms TTL=64
Reply from 192.168.1.95: bytes=32 time=7ms TTL=64

Ping statistics for 192.168.1.95:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 5ms, Maximum = 29ms, Average = 11ms
```

Then 192.168.1.95 is my Raspberry Pi IP.

Or enter **router** client to **inquiry IP address** named "**raspberrypi**". For example, I have inquired to **my RPi IP address, and it is "192.168.1.95".**

```
ssh pi@xxxxxxxxxxx(IP address)
```

Enter the following command:

```
ssh pi@192.168.1.95
```

The default password is "raspberry", case sensitive.

When you see the following prompt, type in yes, as shown below:

```
C:\Users\freenove-14>ssh pi@192.168.1.95
The authenticity of host '192.168.1.95 (192.168.1.95)' can't be established.
ED25519 key fingerprint is SHA256:x/9Im3QqkIWnDZFBUkPK8VWIbPRhHATJA43JHo5k+zQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

"pi@raspberrypi: ~ $" indicates that you have successfully connected to Rpi, as shown below.

```
C:\Users\freenove-14>ssh pi@192.168.1.95
pi@192.168.1.95's password:
Permission denied, please try again.
pi@192.168.1.95's password:
Linux raspberrypi 6.6.51+rpt-rpi-2712 #1 SMP PREEMPT Debian 1:6.6.51-1+rpt3 (2024-10-08) aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Nov 30 14:17:31 2024

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $
```

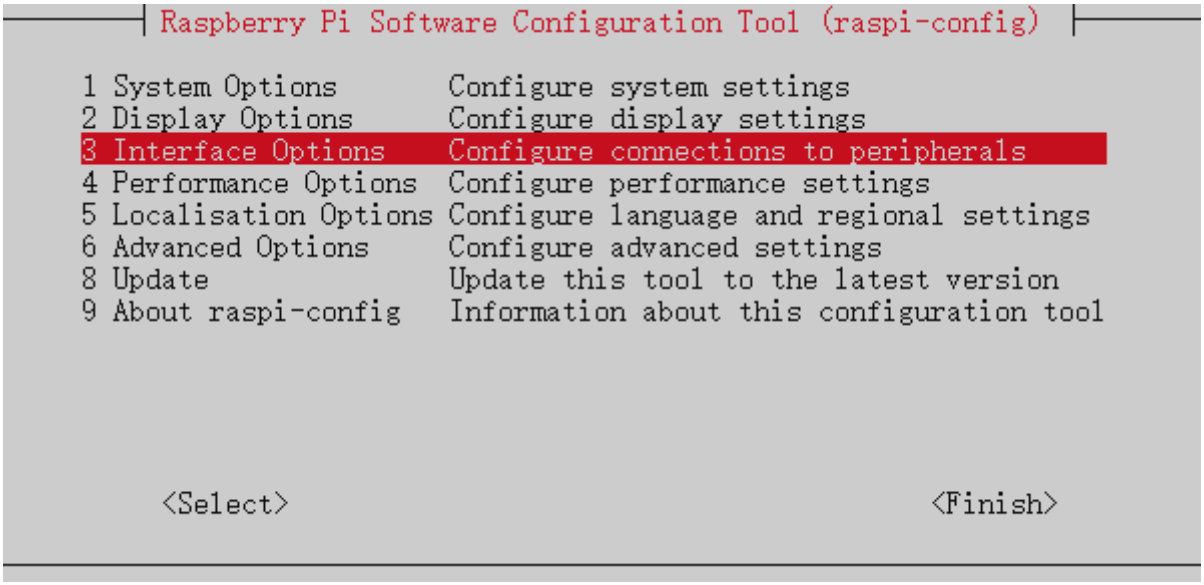
Please note that the typing is invisible when you enter the password. Make sure you type it correctly.

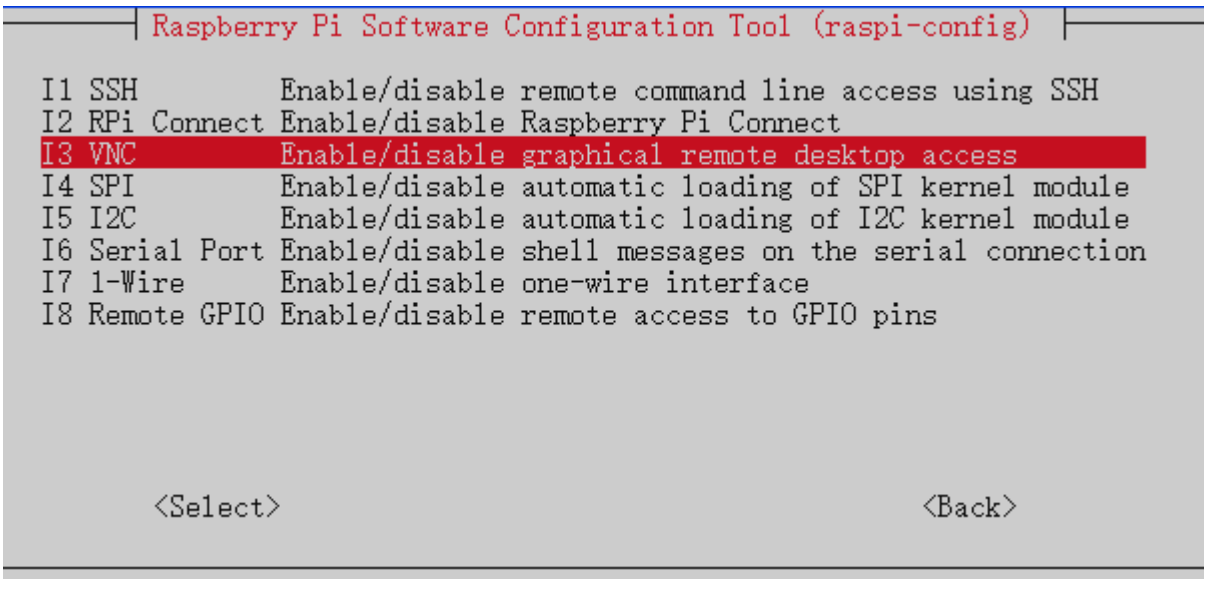
Enable VNC Viewer

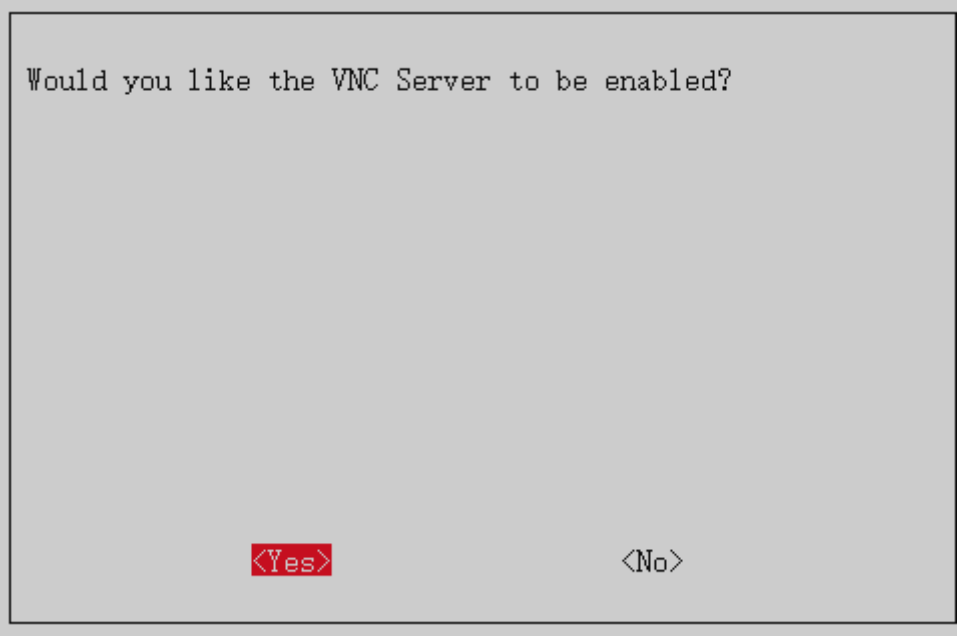
Type the following command. And select Interface Options→P5 VNC → Enter→Yes→OK. Here Raspberry Pi may need be restarted, and choose ok. Then open VNC interface.

```
sudo raspi-config
```

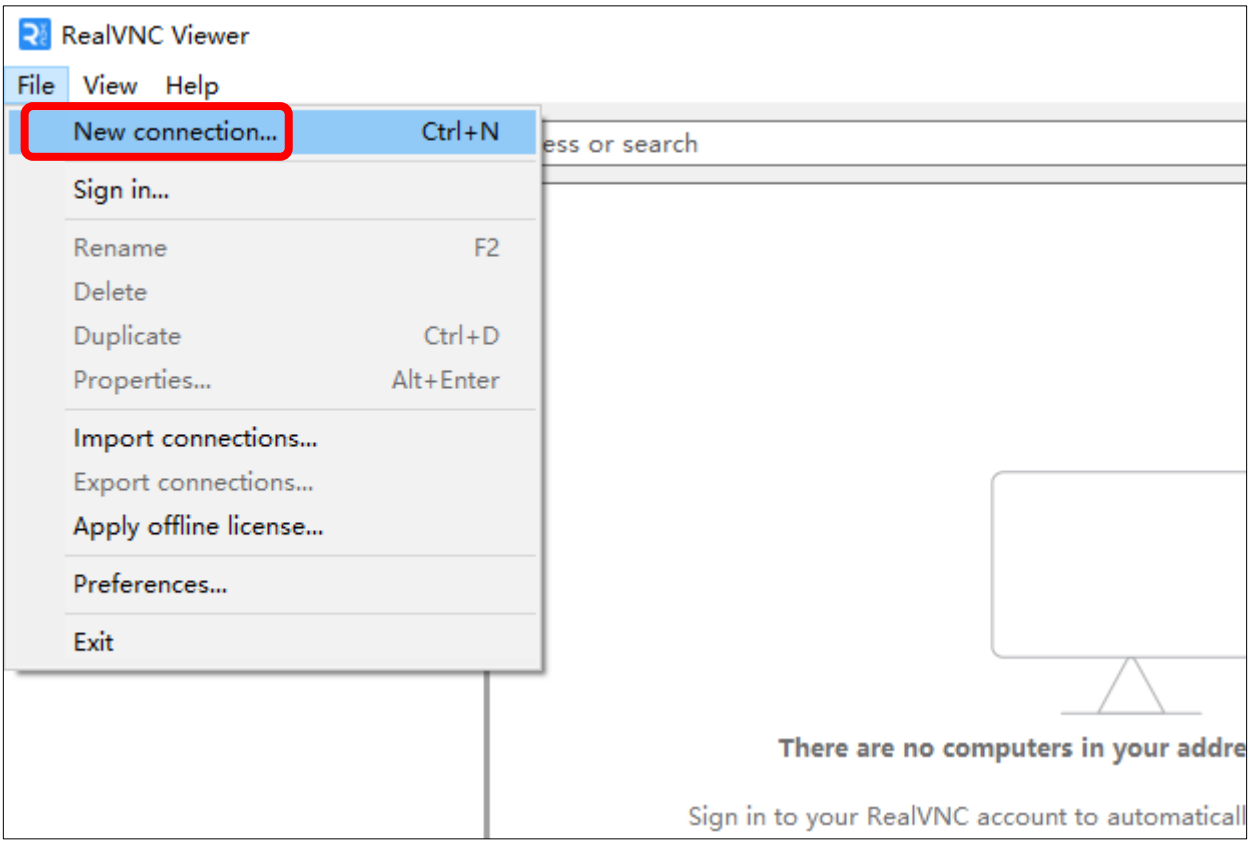
```
SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $ sudo raspi-config
```

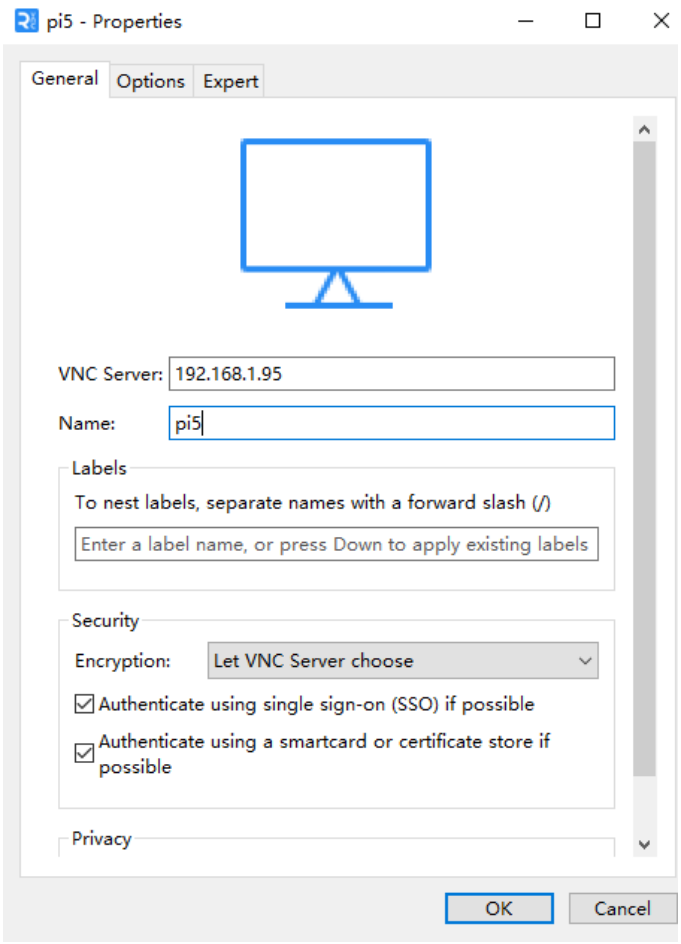






Then download and install VNC Viewer according to your computer system by click following link:
https://www.realvnc.com/en/connect/download/viewer/
After installation is completed, open VNC Viewer. And click File → New Connection. Then the interface is shown below.

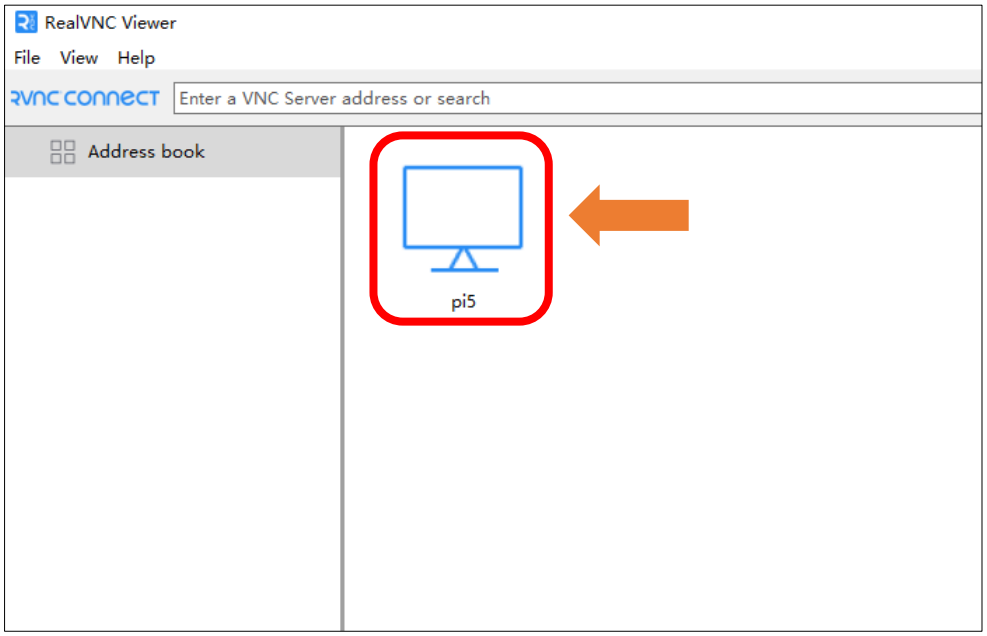




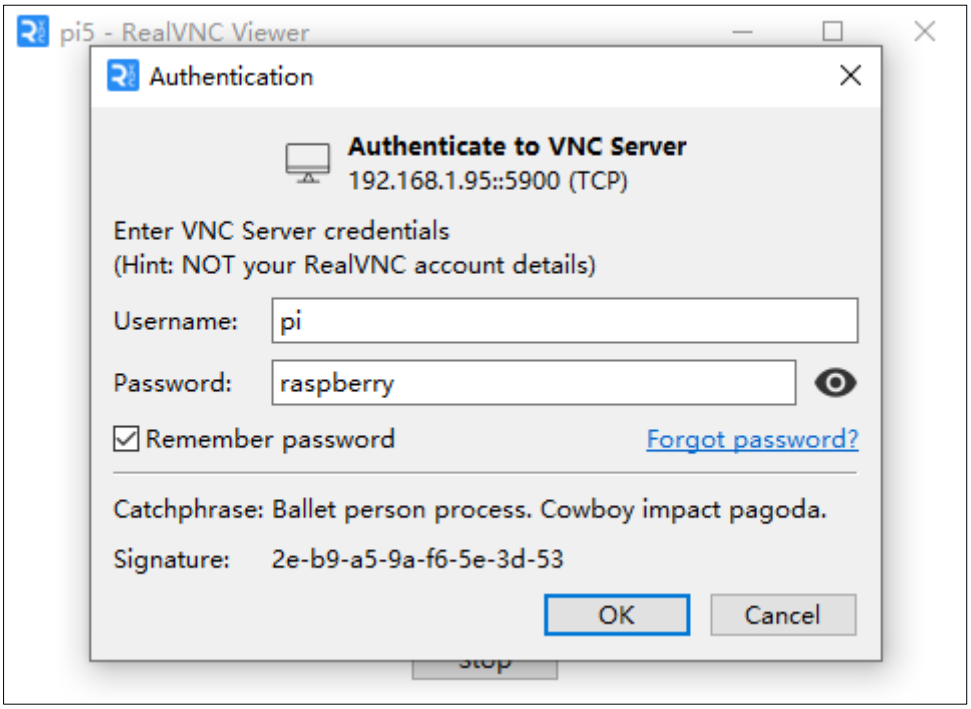
Enter ip address of your Raspberry Pi and fill in a name. Then click OK.

.

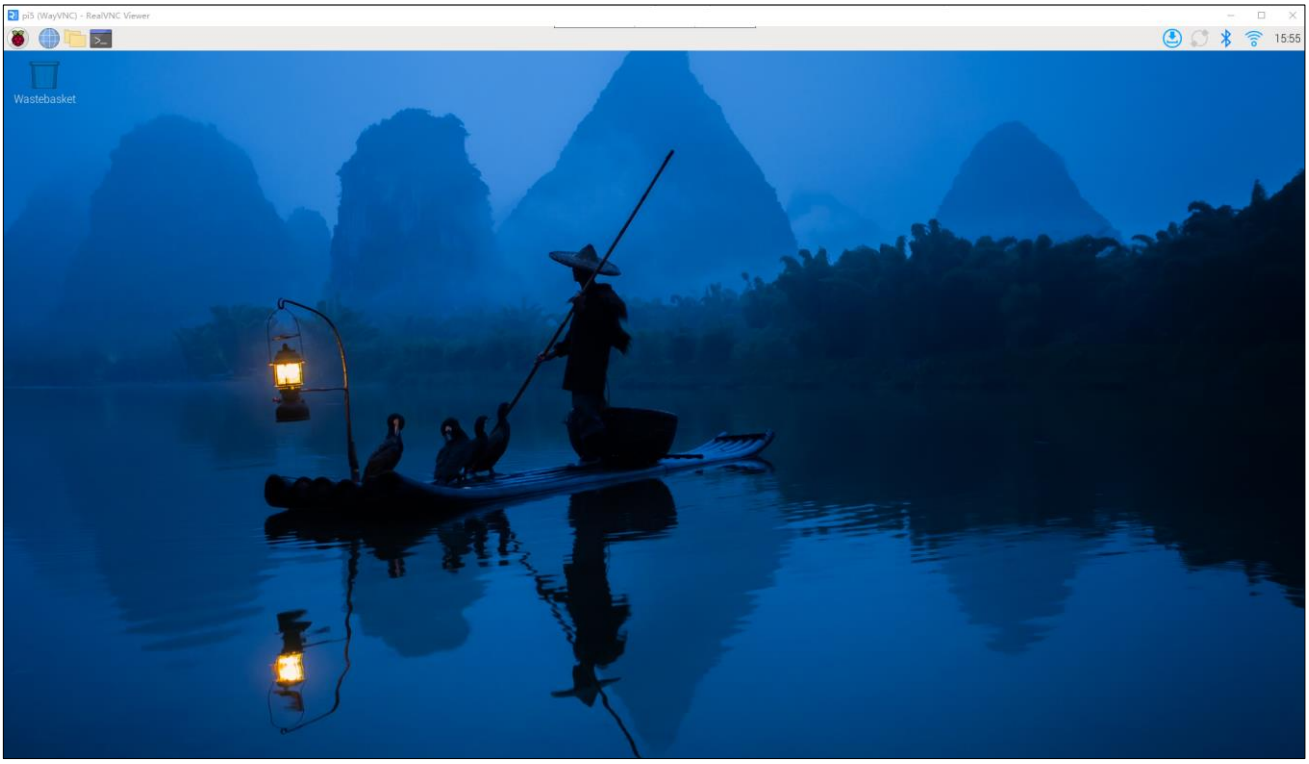
Then on the VNC Viewer panel, double-click new connection you just created,



and the following dialog box pops up.



Enter username: **pi** and Password: **raspberry**. And click OK.



Here, you have logged in to Raspberry Pi successfully by using VNC Viewer.

Chapter 1 Software installation and Test (necessary)

If you have any concerns, please feel free to contact us via support@freenove.com

In this chapter, we will make some necessary preparation: start your Pi Raspberry and install some necessary libraries. Then test some parts. Batteries are needed when driving peripherals such as motors, servos, LEDs, etc.

Note:

1, **Please use Raspberry Pi OS with Desktop**

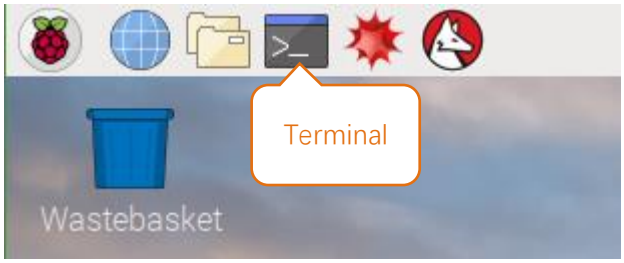
2, The installation of libraries takes much time. **You can power Raspberry Pi with a power supply Cable**.

3, If you are using **remote desktop** to login Raspberry Pi, you need to use VNC viewer.

Vedio: https://youtu.be/T7gFezHcqZU

Step 1 Obtain the Code

To download the code, you can power Raspberry Pi with a power supply cable **or** switch on S1 (Power Switch). Then open the Raspberry Pi and the terminal. You can open the terminal by clicking as shown below, or you can press "CTRL + ALT + T" on the desktop.



The terminal is shown below:

Open the terminal and type the following commands to obtain the car code. And the code will be placed in the directory "Pi". (Note: Here are two commands. Please execute them in order.)

```
cd ~
git clone --depth 1 https://github.com/Freenove/Freenove_Tank_Robot_Kit_for_Raspberry_Pi.git
```

Downloading takes some time. Please wait with patience.

You can also find and download the code by visiting our official website (http://www.freenove.com) or our GitHub repository (https://github.com/freenove).

If you have never learned python before, you can learn some basics through the following link:

https://python.swaroopch.com/basics.html

Step2 Install the Needed Libraries

We have wrriten a Python script for you to install all dependency libraries automatically.

1. Change the root.

```
sudo chmod 755 -R ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi
```

2. Execute following commands to enter directory of "setup.py".

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code
```

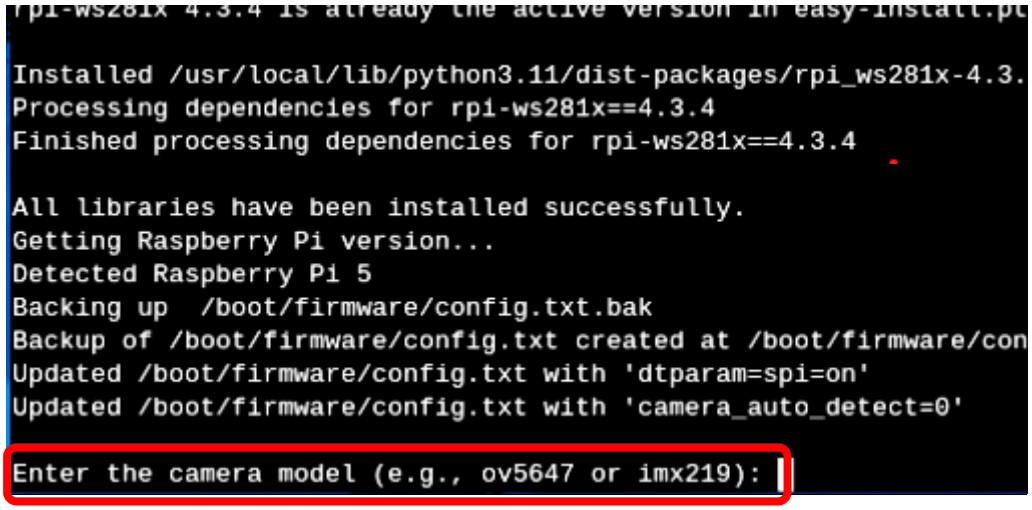
```
pi@raspberrypi:~ $ cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/
```

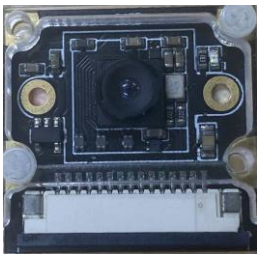
3. Run setup.py

```
sudo python setup.py
```

```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code $ sudo python setup.py
```

If it is your first time to run the script, it will ask you to input the camera type. At this point, input the content according to the camera you have. (The one included in this kit is of the type ov5647).

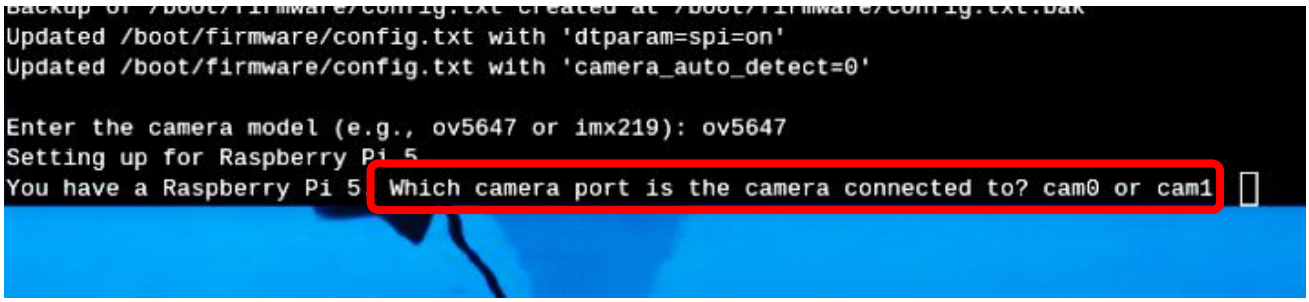


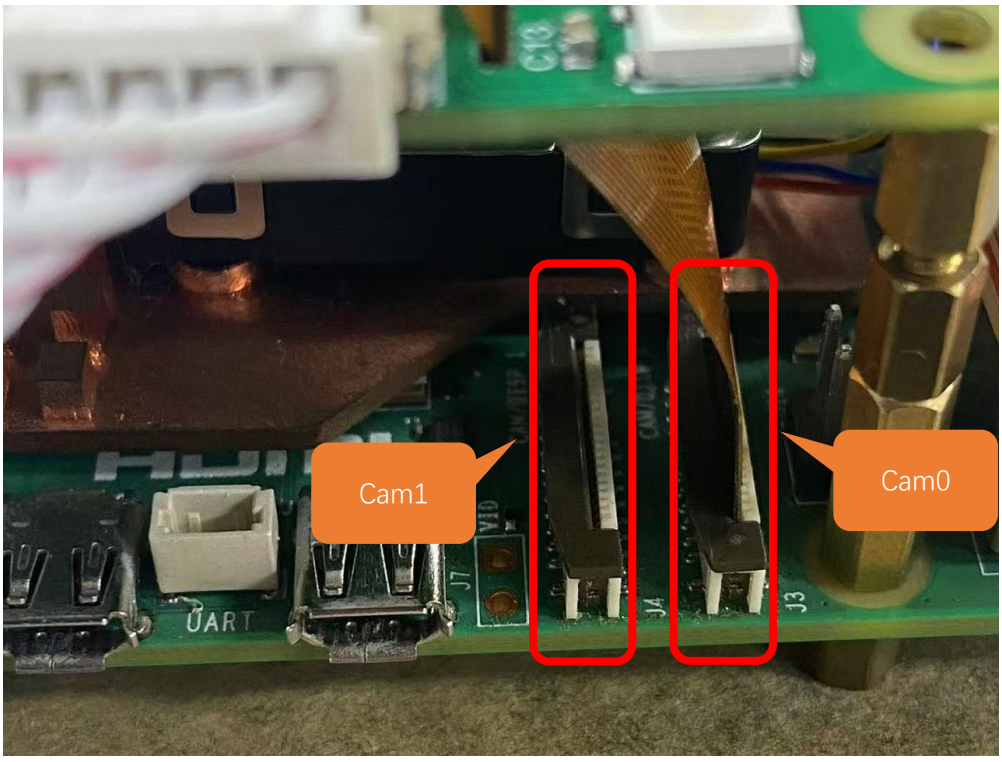
OV5670	Imx219
	

If your Rpi is a Pi 5, it will ask you which camera interface to use. You can type in **cam0** or **cam1** based on the interface you want to use.

Note: Make sure you connect the camera cable to the corresponding interface.

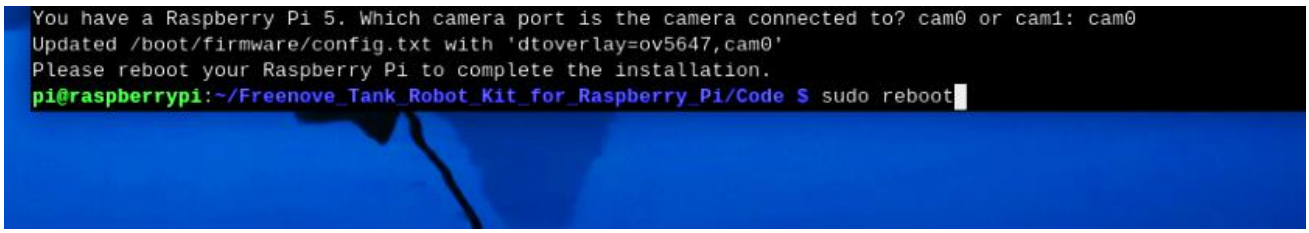
If it is not Raspberry Pi, this setting is not needed, as there is only camera interface available.





Reboot your Raspberry Pi after installing the libraries.

```
sudo reboot
```



The reboot of the pi takes some time, please wait with patience.

If the installation fails, please rerun setup.py. After the installation is completed, restart the Raspberry Pi. Most installation failures are caused by network reasons.

```
sudo python setup.py
```

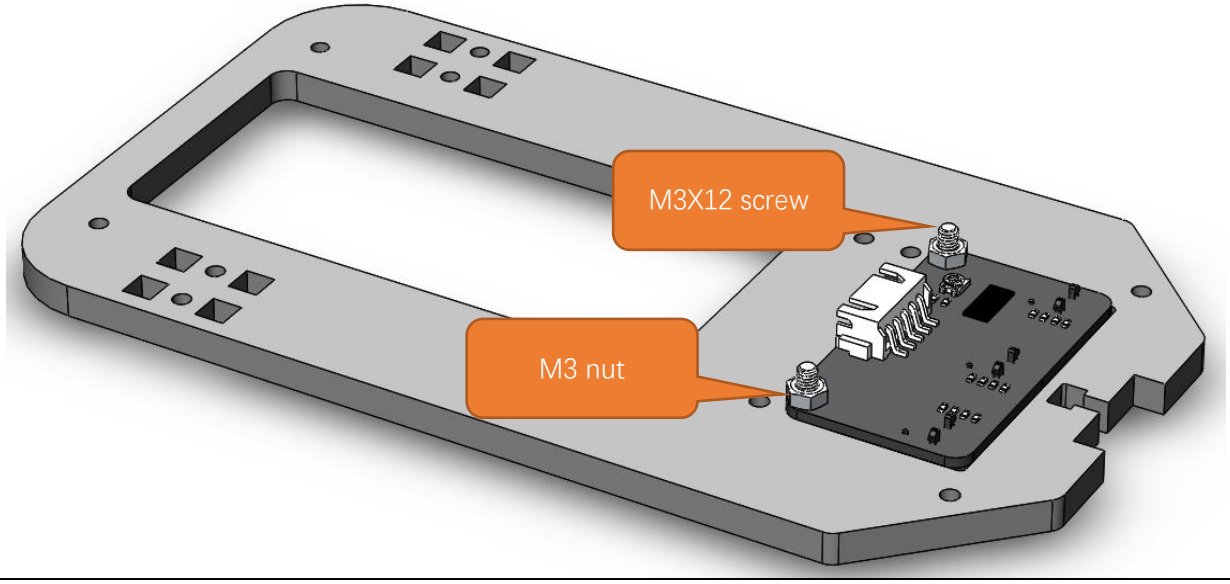
Chapter 2 Assemble Smart Car

Vedio: https://youtu.be/44PiaImoPNo

If you have any concerns, please feel free to contact us via support@freenove.com

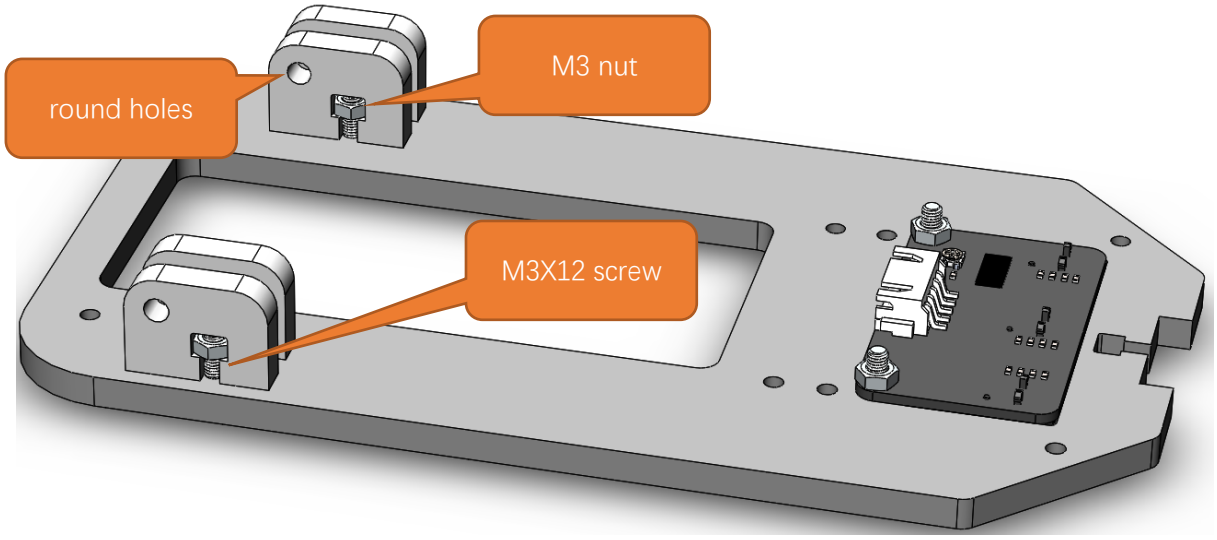
Step 1 Install line tracking sensor.

Secure the line tracking sensor to the acrylic using two M3X12 screws and two M3 nuts.

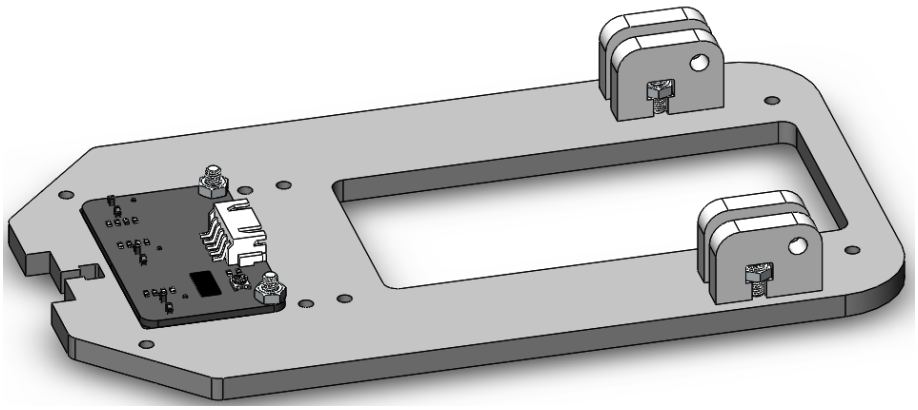


Step 2

Attach four acrylic support frames to the acrylic using four M3X12 screws and four M3 nuts. Note the position of the round holes in the acrylic support frame.

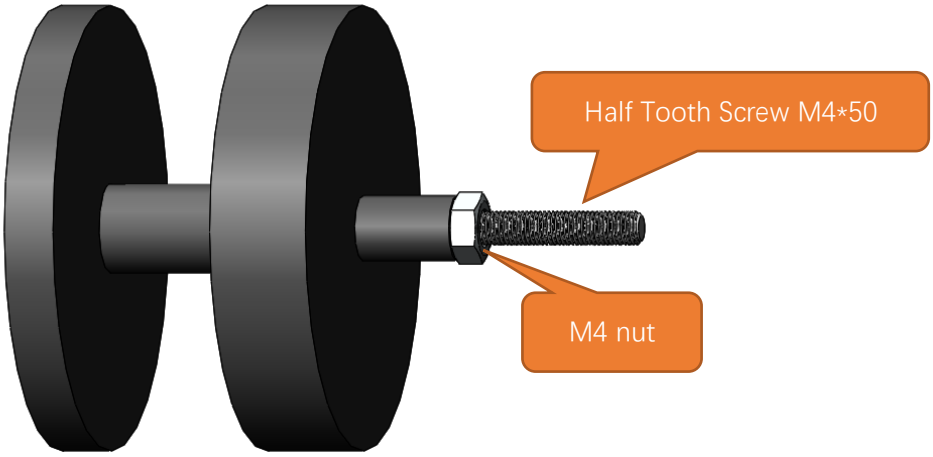


It looks like below after assembled



Step 3

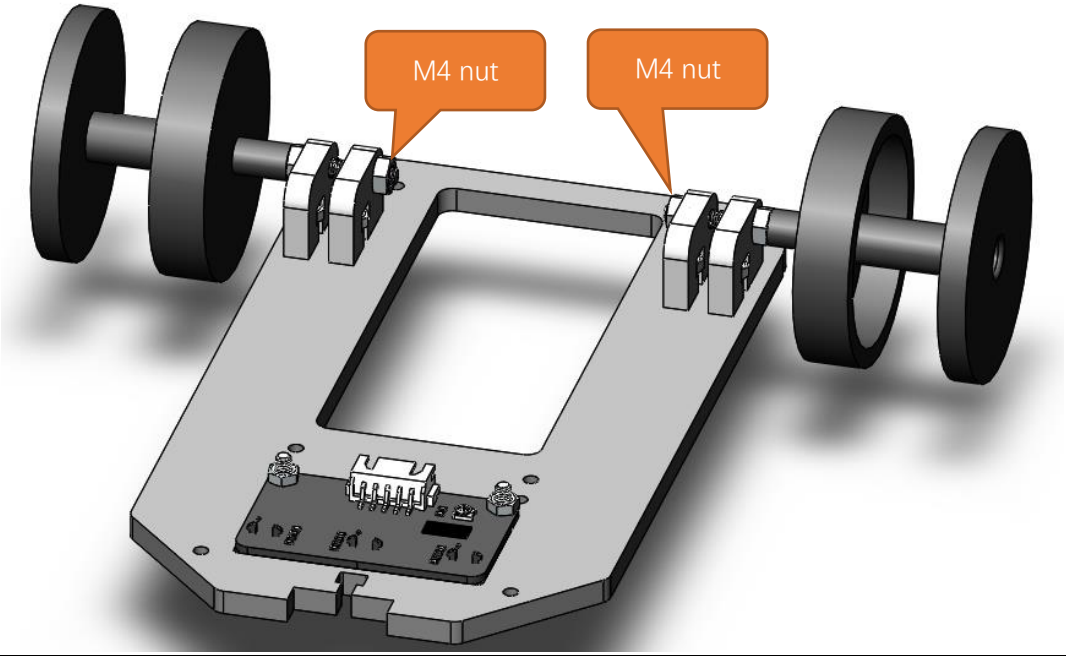
Using a Half Tooth Screw M4*50 and an M4 nut, assemble the drive wheel as shown below.



Same steps to assemble the other drive wheel.

Step 4

Secure the driving wheel to the acrylic support frame using 2 M4 nuts.

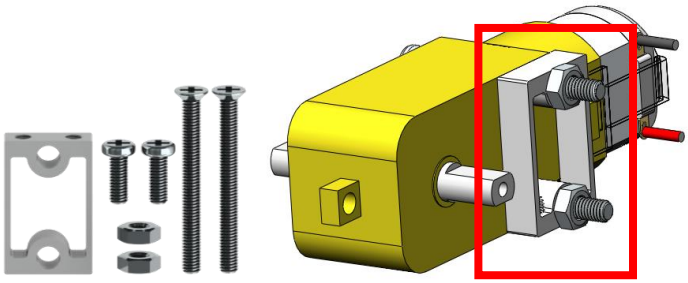


Step 5

There is a special fixed bracket to fix motor, which contains an aluminum bracket, two M3*30 screws, two M3*8 screws, and two M3 nuts, as shown below:

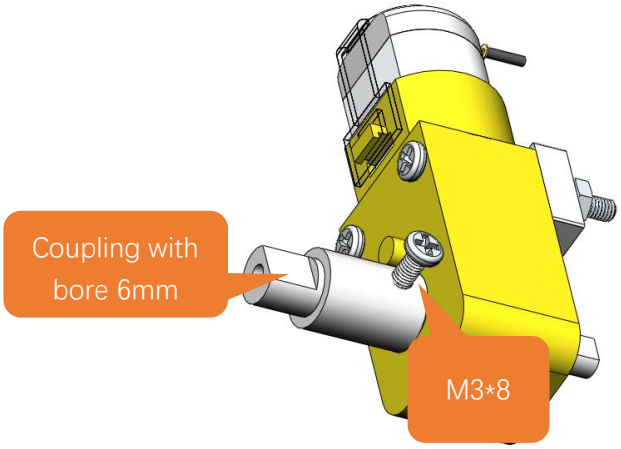
Use 2 M3x30 screws and 2 M3 nuts.

Do NOT remove the cable tie on the motor!



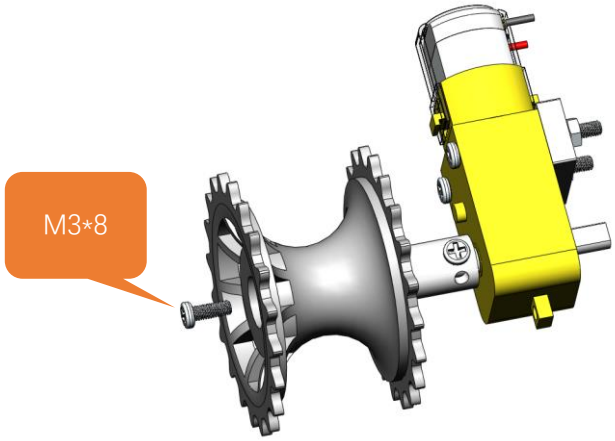
Step 6 Connect coupling with motor.

Use an M3x8 screw to connect coupling with motor. (Use the screw in the M3x8 screw bag.)



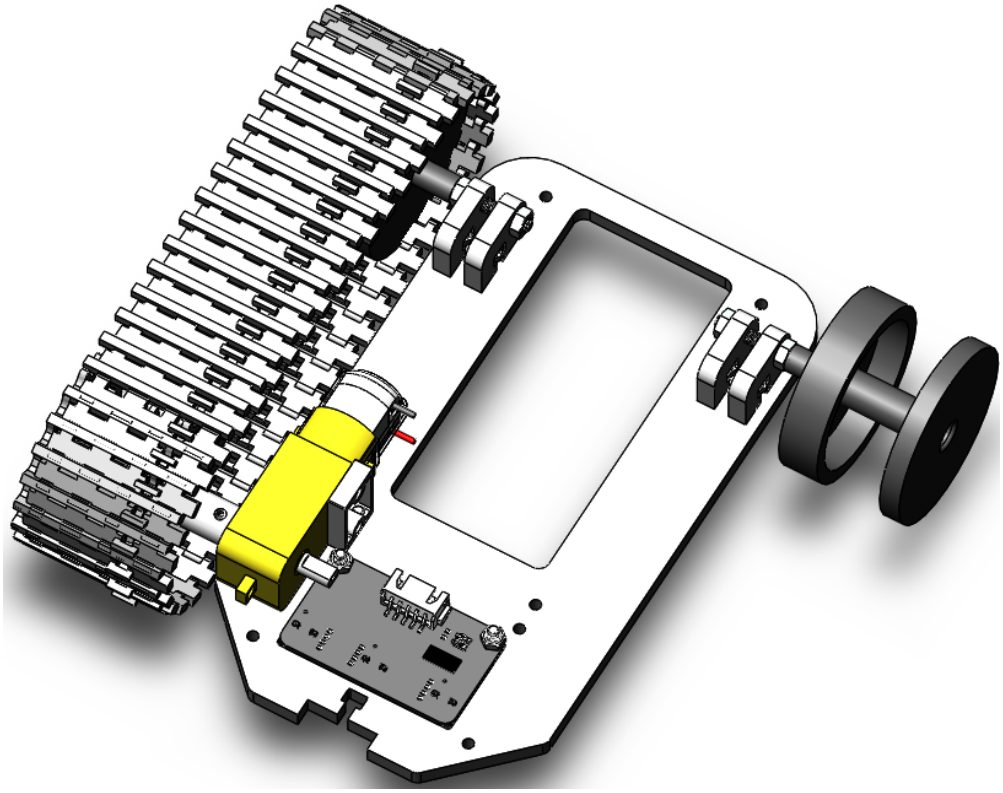
Step 7

Attach the track driver to the motor using an M3x8 screw. (Use the screw in the M3x8 screw bag.))

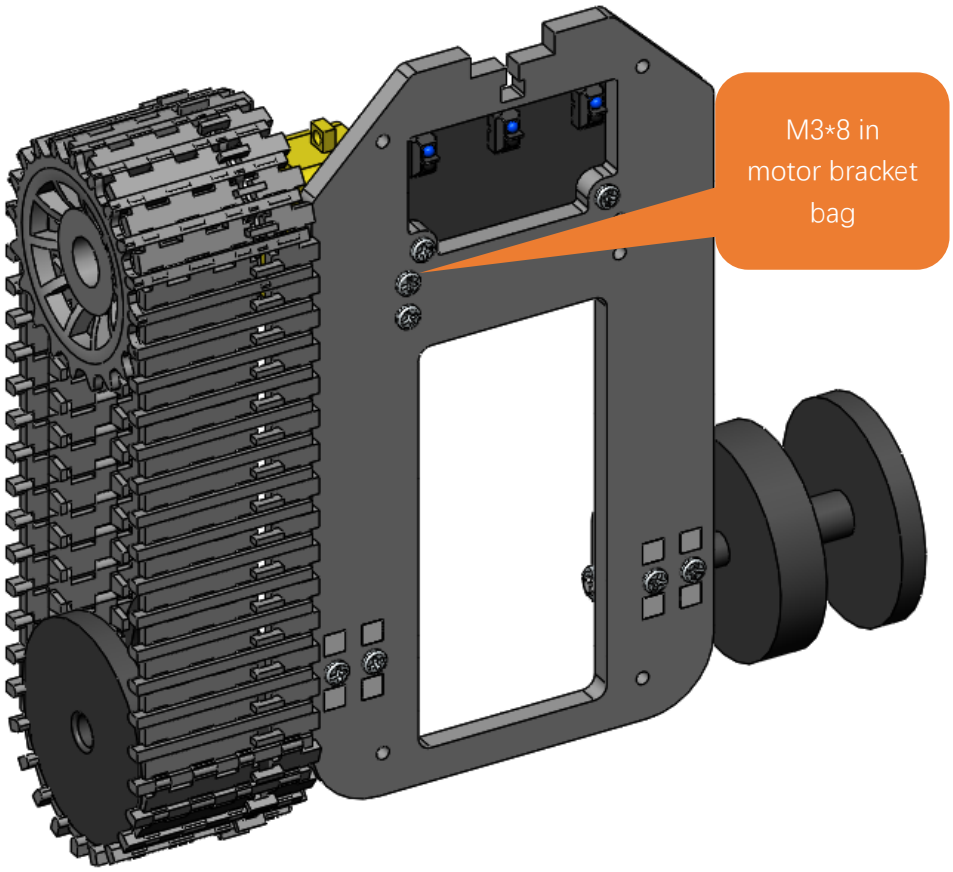


Step 8

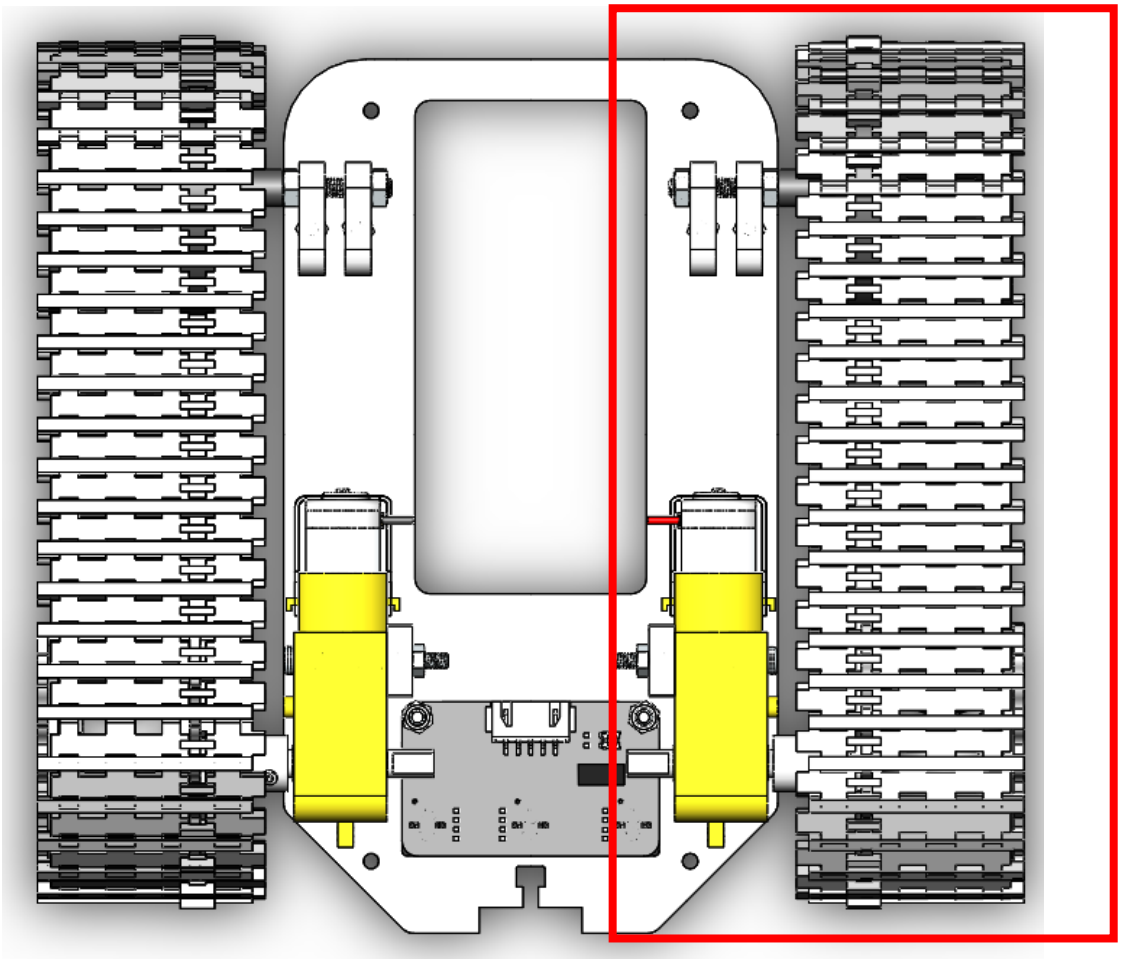
Put the track on as shown in the picture below.



Use two M3x8 screws from the motor bracket bag at the bottom to hold the motor in place.

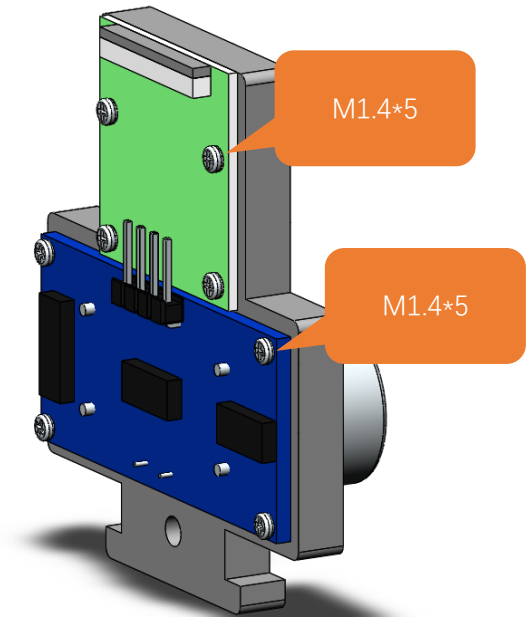


Step 9 With the same steps, install the motor and track on the other side.



Step 10

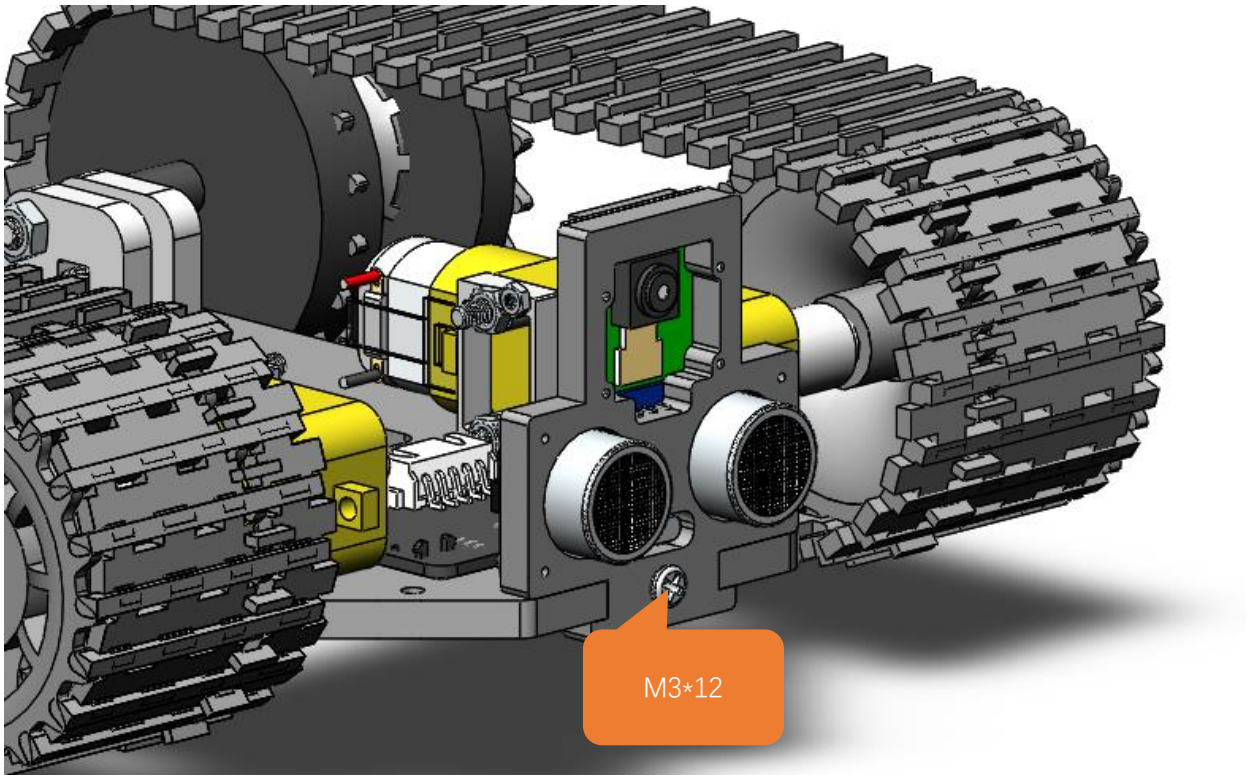
Attach the camera and ultrasound to the acrylic using eight M1.4x5 screws.



If they cannot be installed, please flip the acrylic board. The hole sizes are different on two sides.

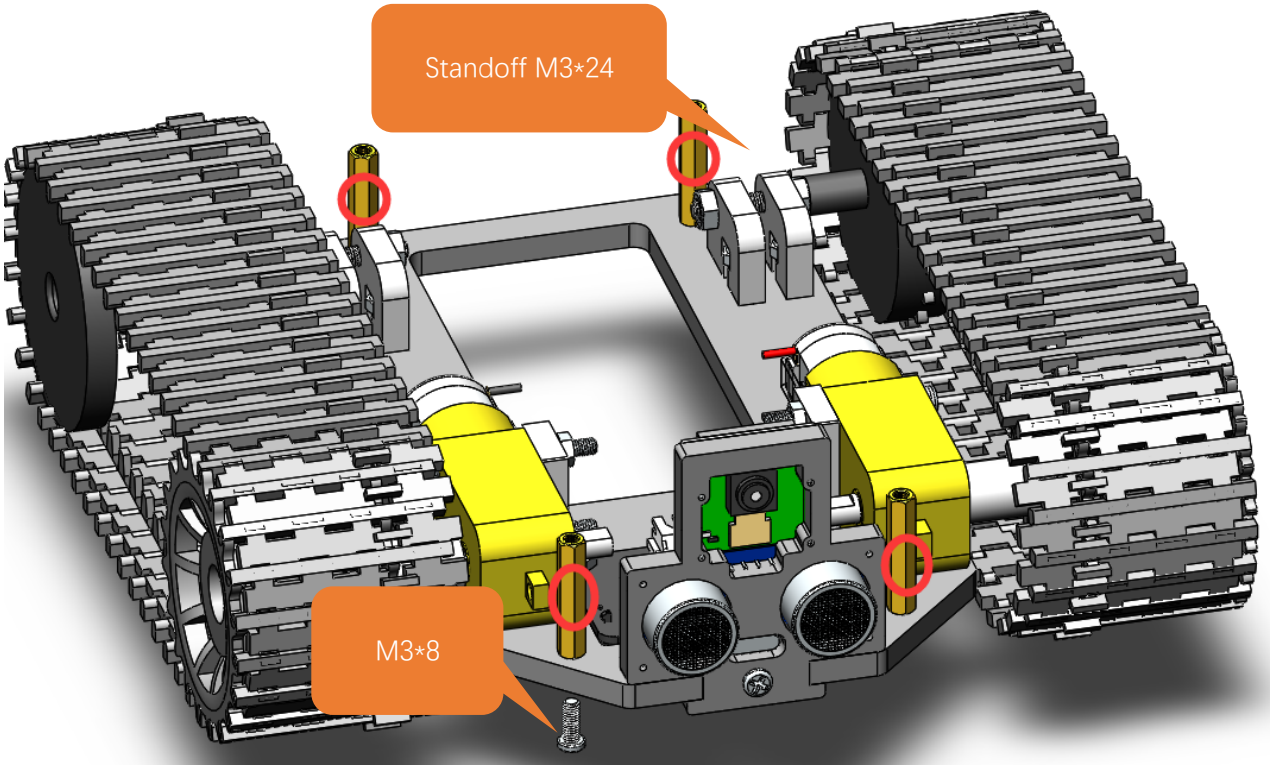
Step 11

Mount the camera and ultrasound onto the acrylic using one M3x12 screw and one M3 nut.



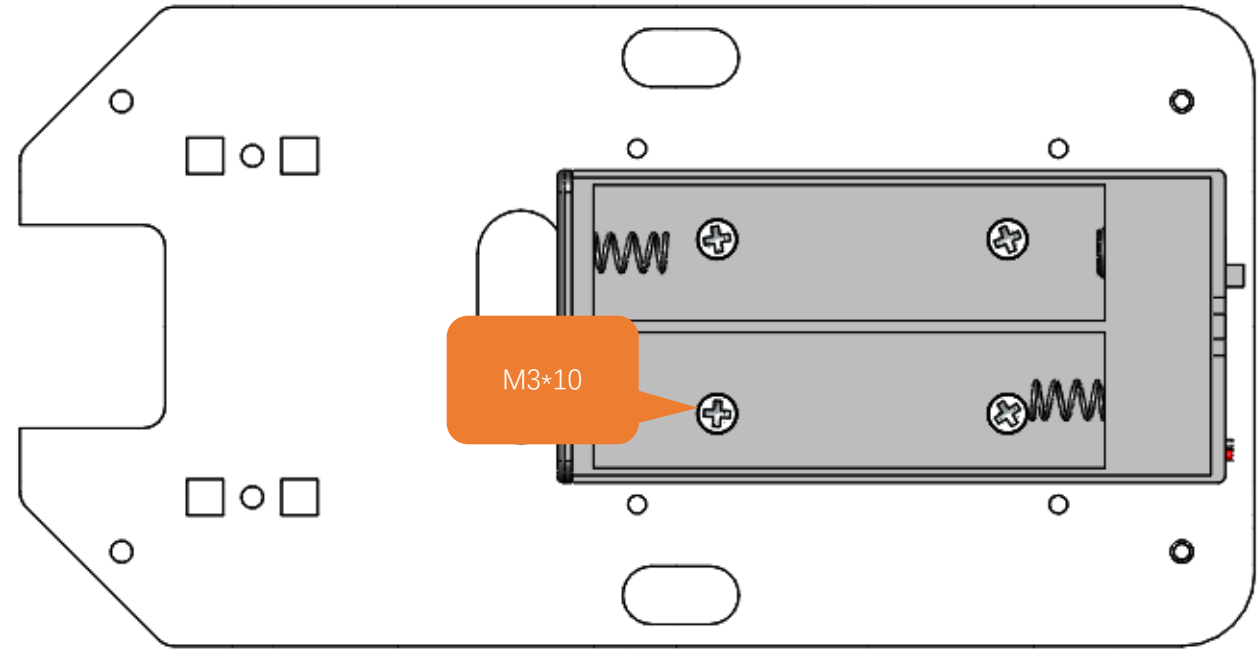
Step 12

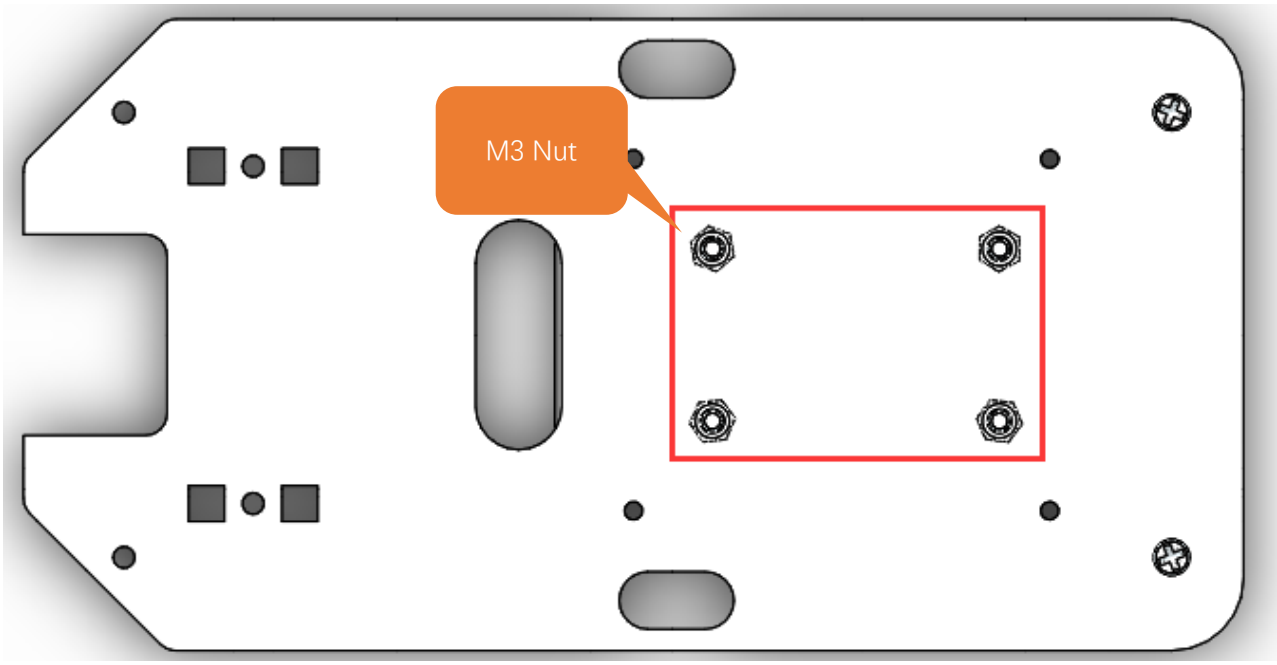
Use four M3x8 screws to fix four M3x24 standoff onto the acrylic.



Step 13

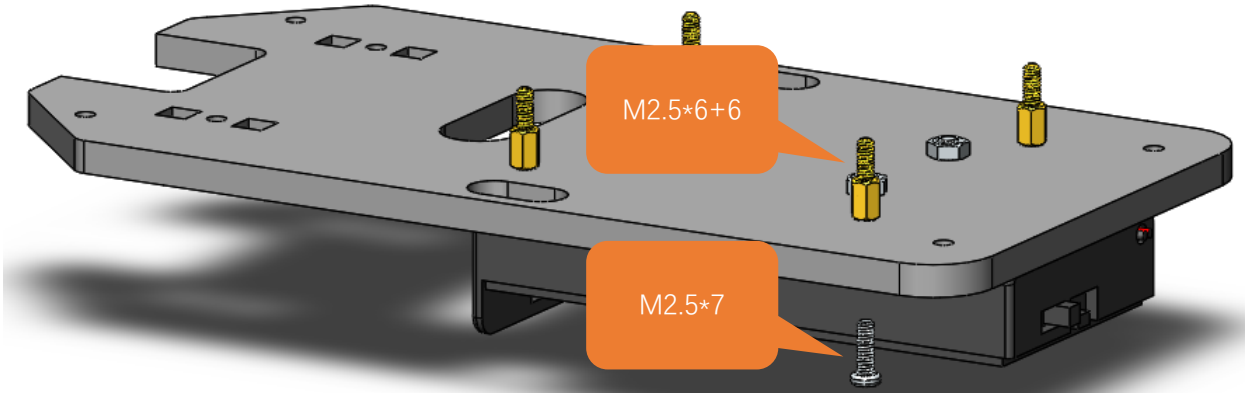
Use 4 M3x10 screws and 4 M3 nuts to secure the battery case to another piece of acrylic.





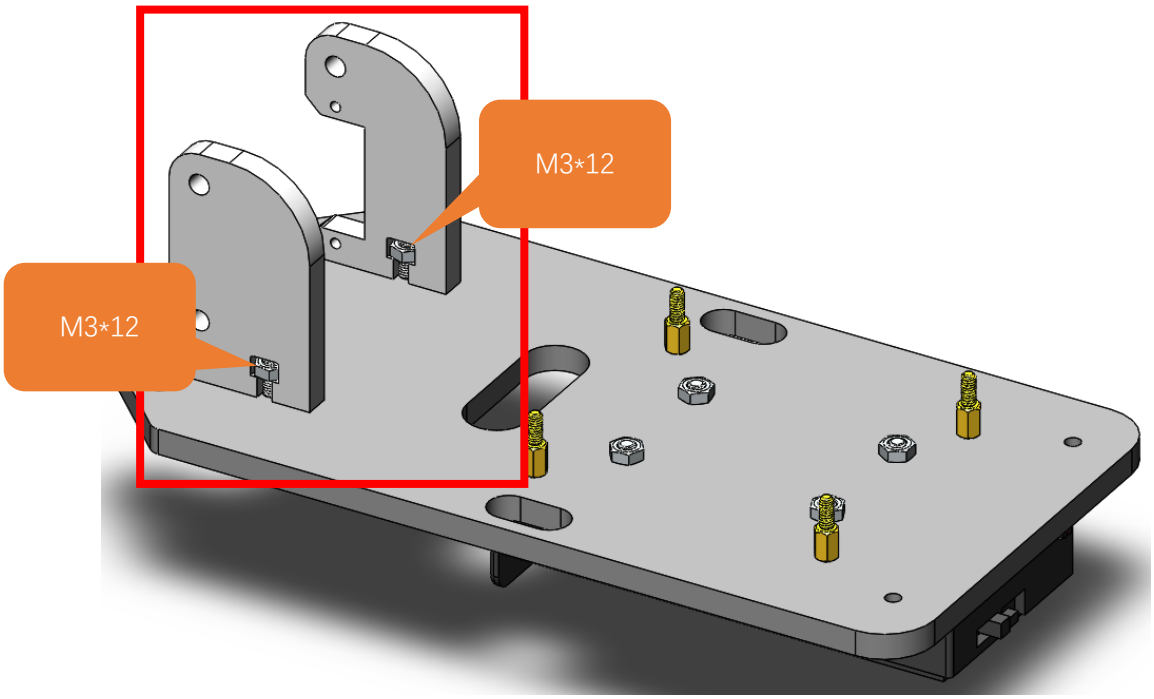
Step 14

Attach 4 M2.5x6+6 screws to the other side of the acrylic using 4 M2.5x7 screws.



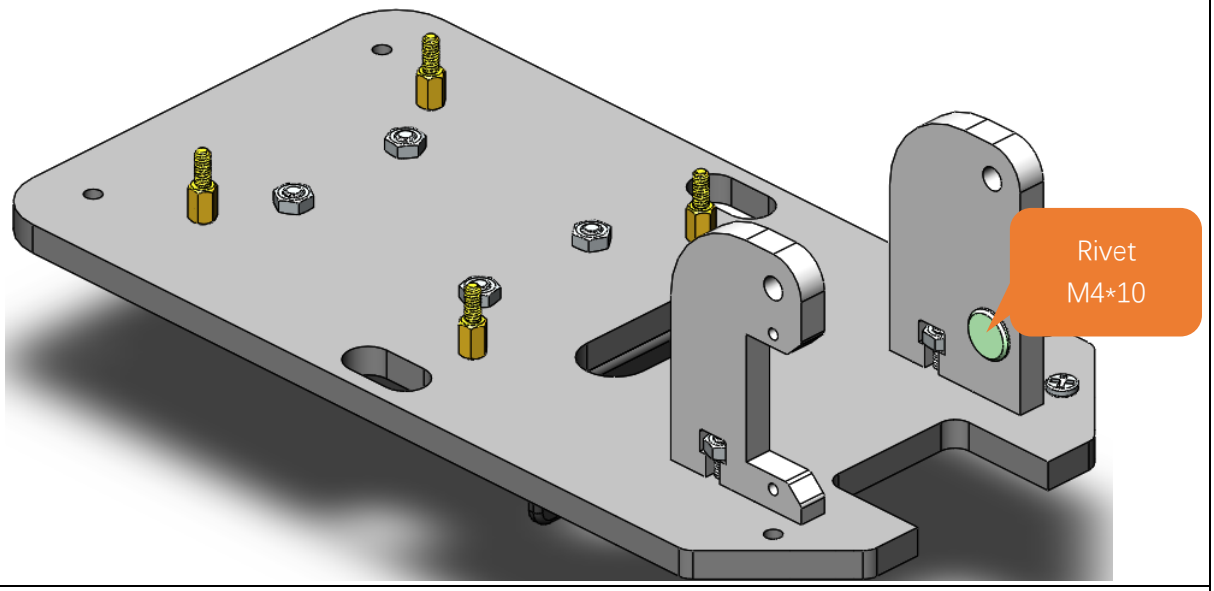
Step 15

Using 2 M3x12 screws and 2 M3 nuts, secure the two acrylic connectors to the acrylic sheet. Please note that the shape of the acrylic is not the same, and their position is fixed.



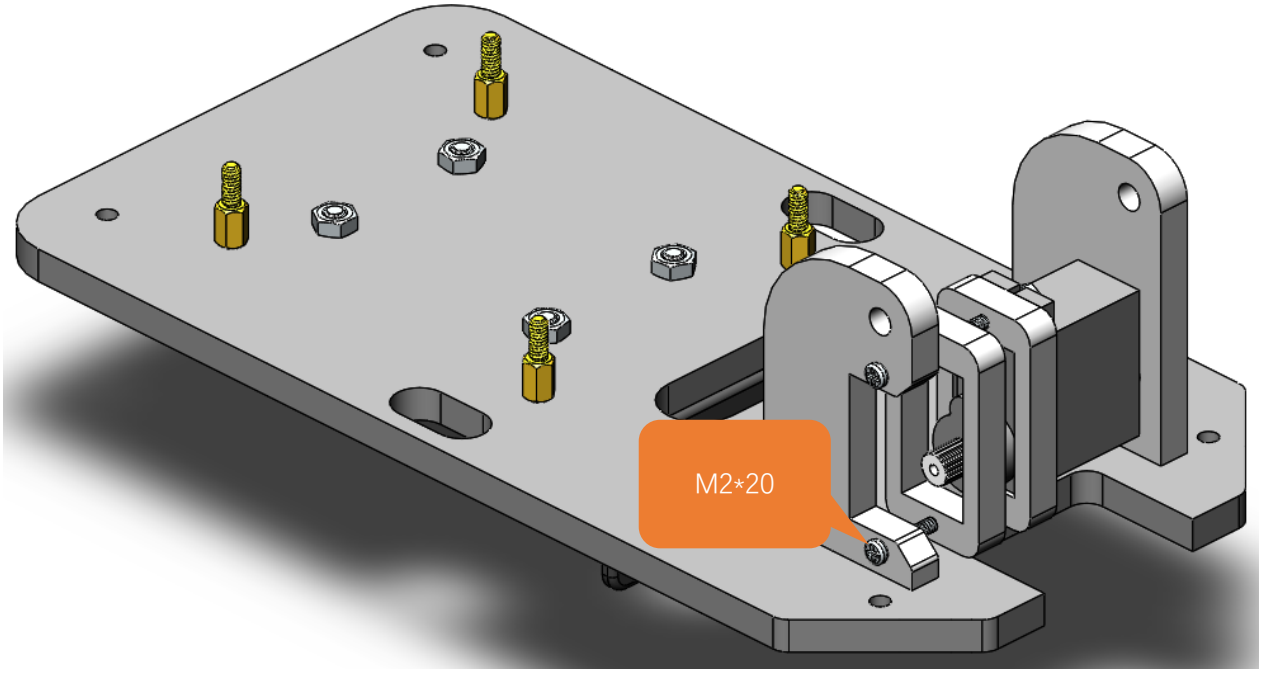
Step 16

Attach an M4x10 Rivet to the acrylic as shown in the image below.

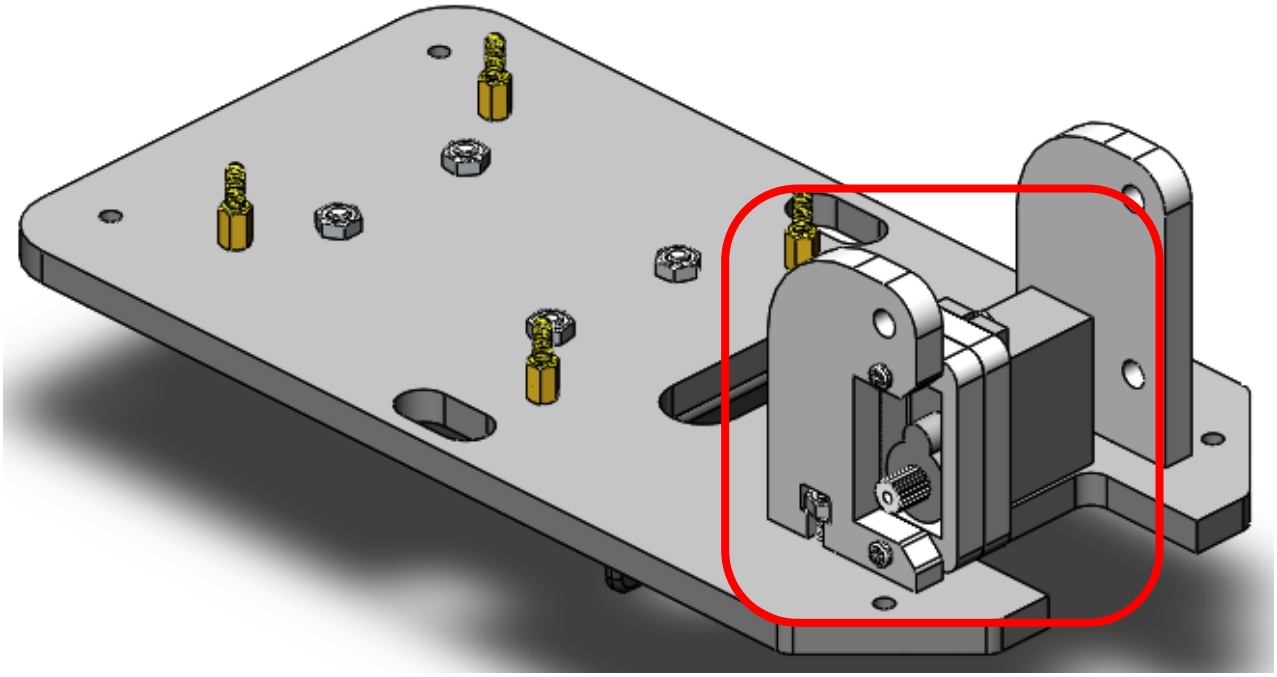


Step 17

Using 2 M2x20 screws and 2 M2 nuts, attach the Opaque Black Servo and 2 acrylics to the bottom acrylic. Servo shaft is below.

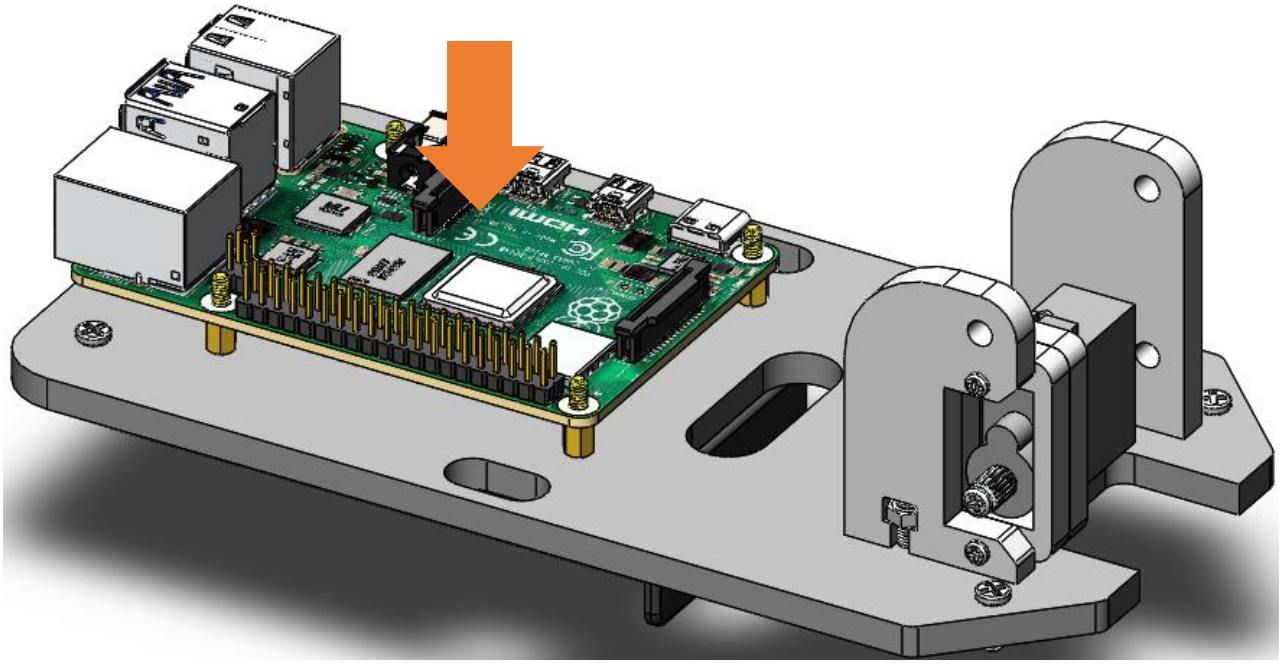


The installation is complete as shown in the following figure.



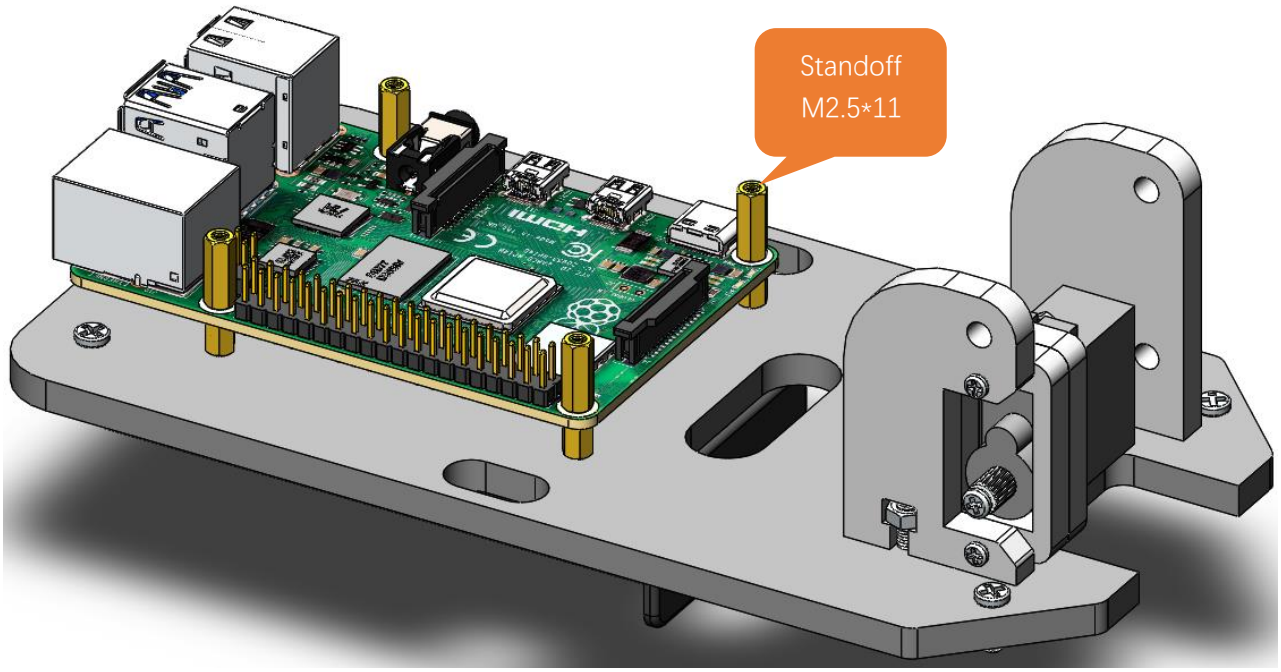
Step 18

Install Raspberry PI.



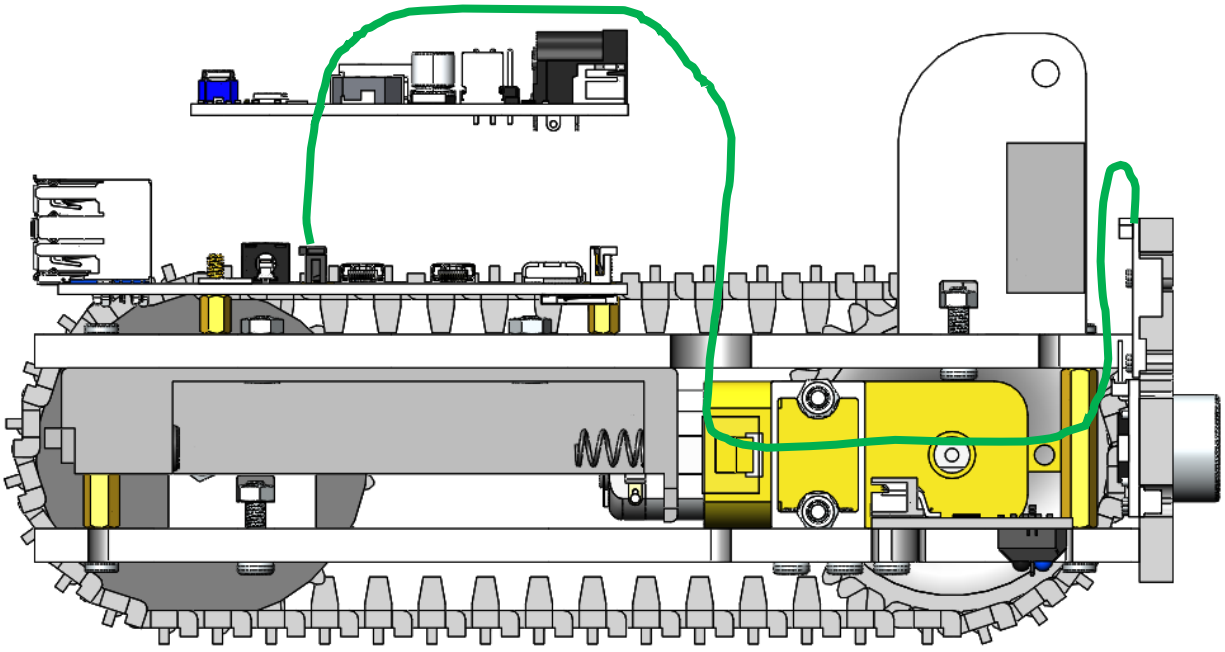
Step 19

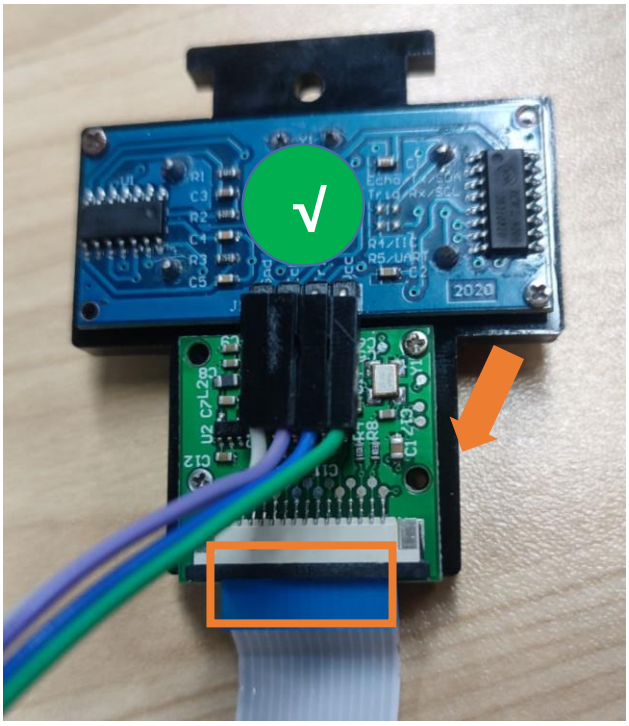
Install 4 M2.5x11 standoff onto the Raspberry PI.

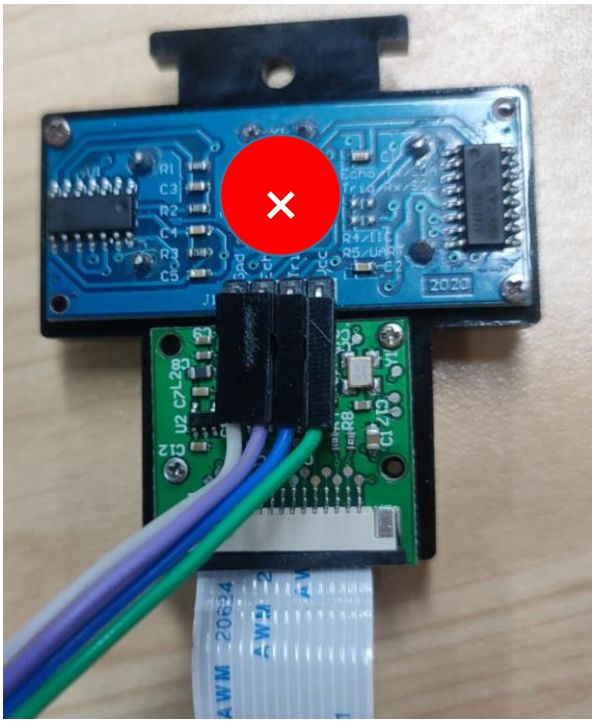


Step 20 Conect camera.

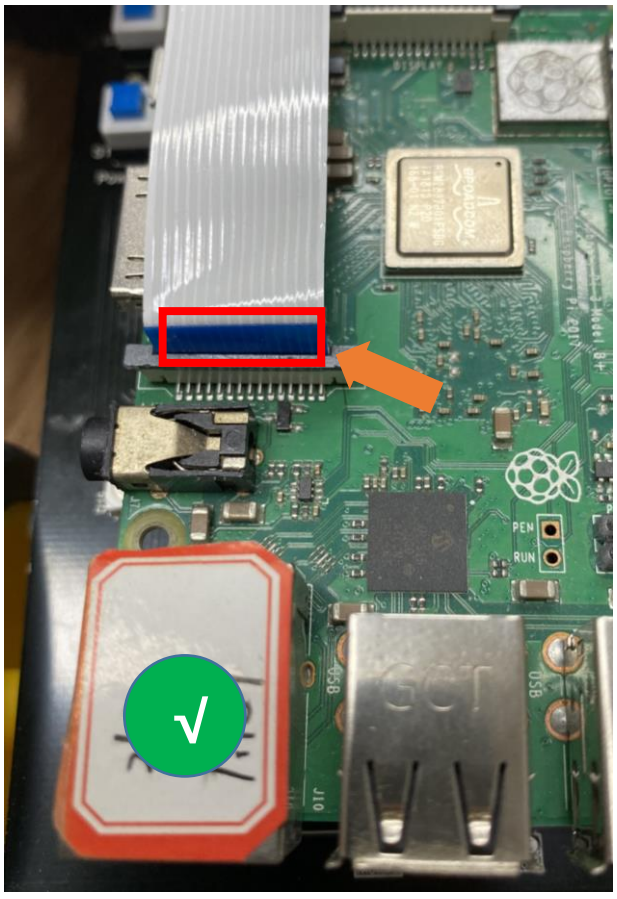
The CSI camera must be connected or disconnected under no power and when Raspberry Pi is shut down, or the camera may be burned. The board was attached to the Raspberry PI using four M3x8 screws.。

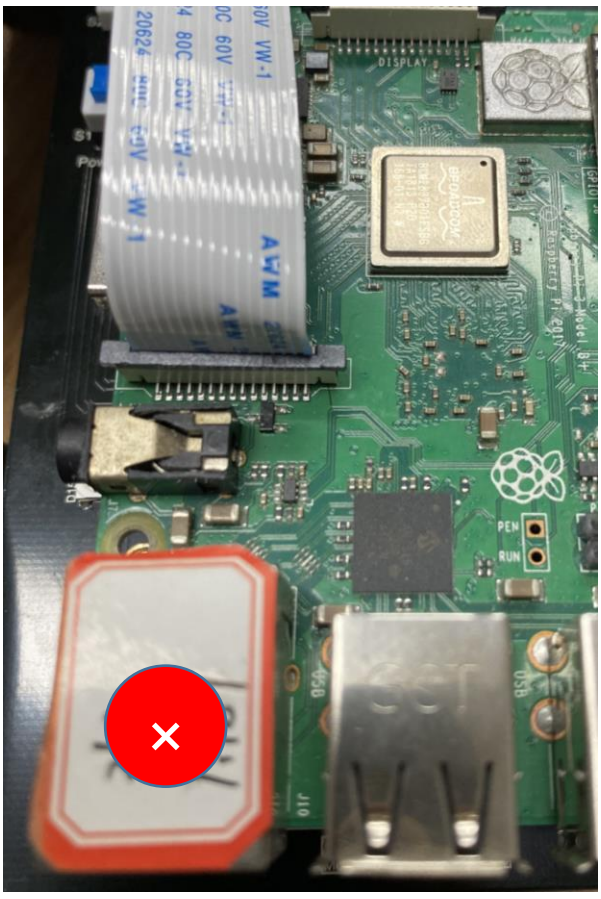




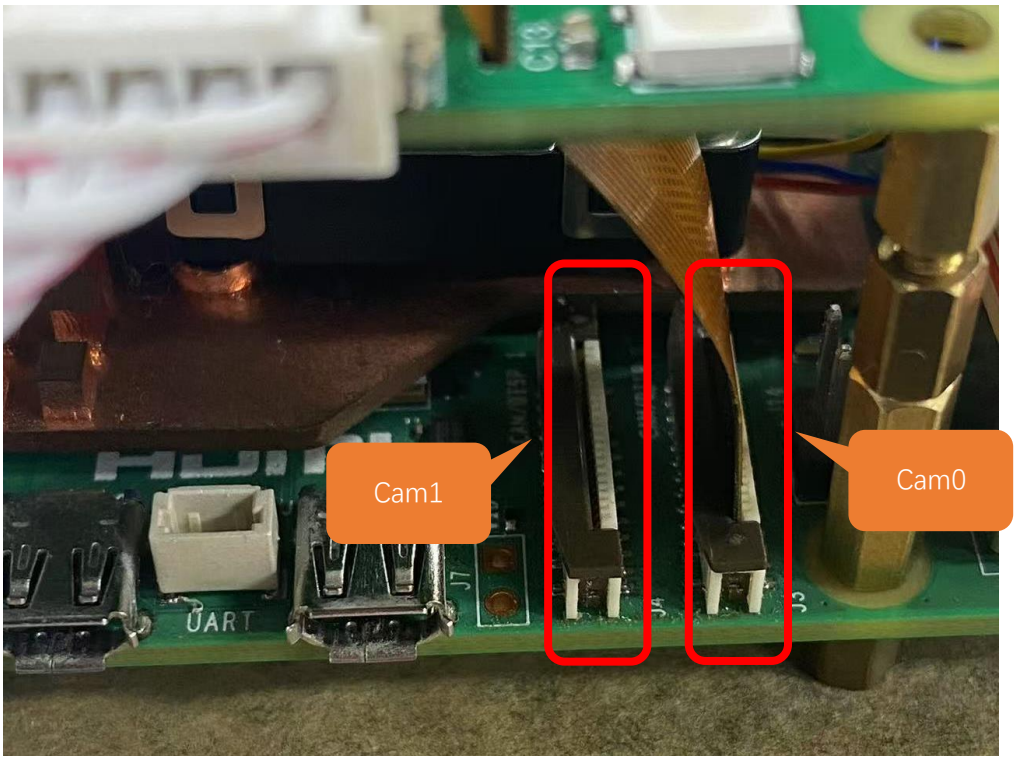


Pay attention to **the Blue bar of cable.**



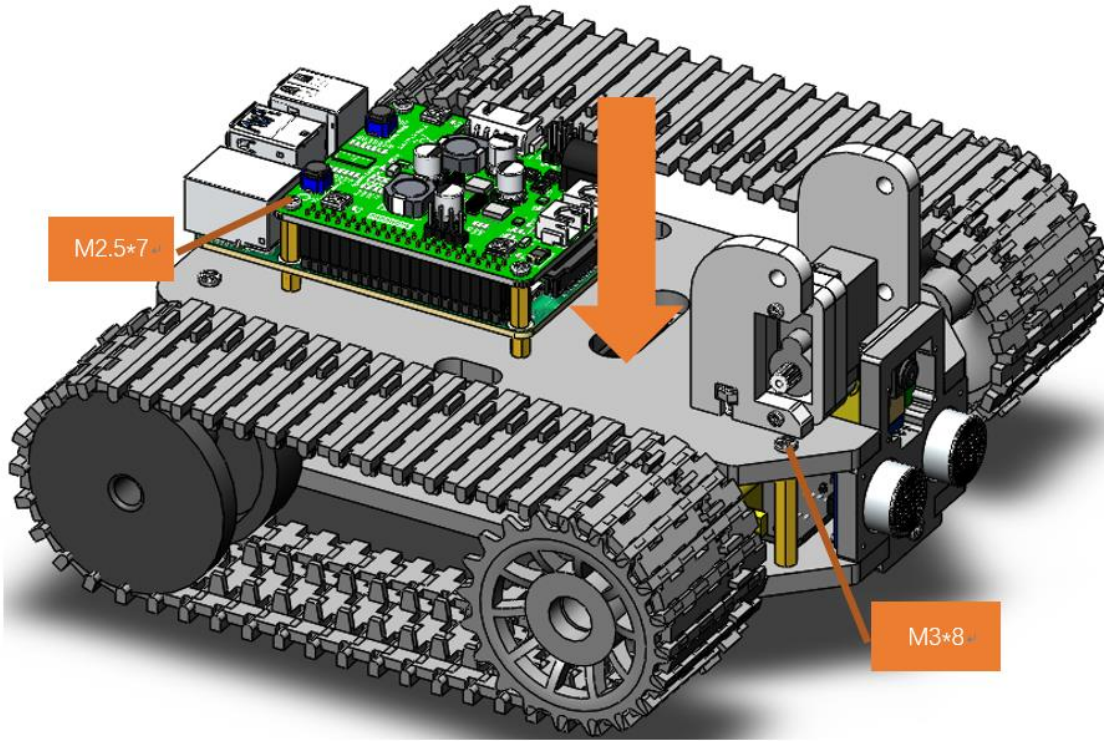


Attention for Pi 5 users: As we have set the camera interface when installing the libraries in the previous chapter, please connect the camera to the corresponding interface.

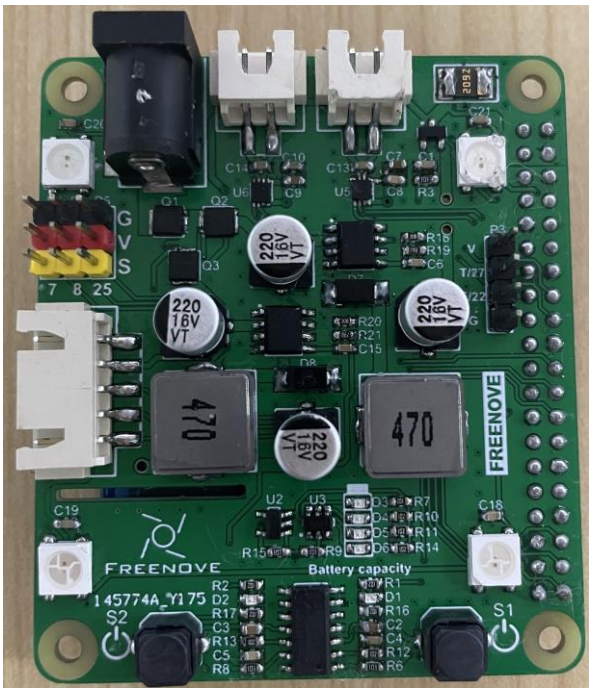
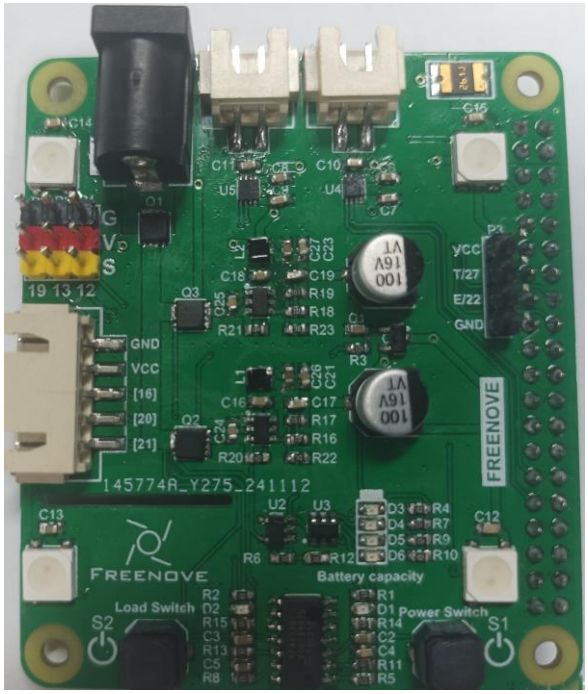


If you want to change the interface, please repeat the library installation steps to set another interface.
Pay attention to the Blue bar of cable.

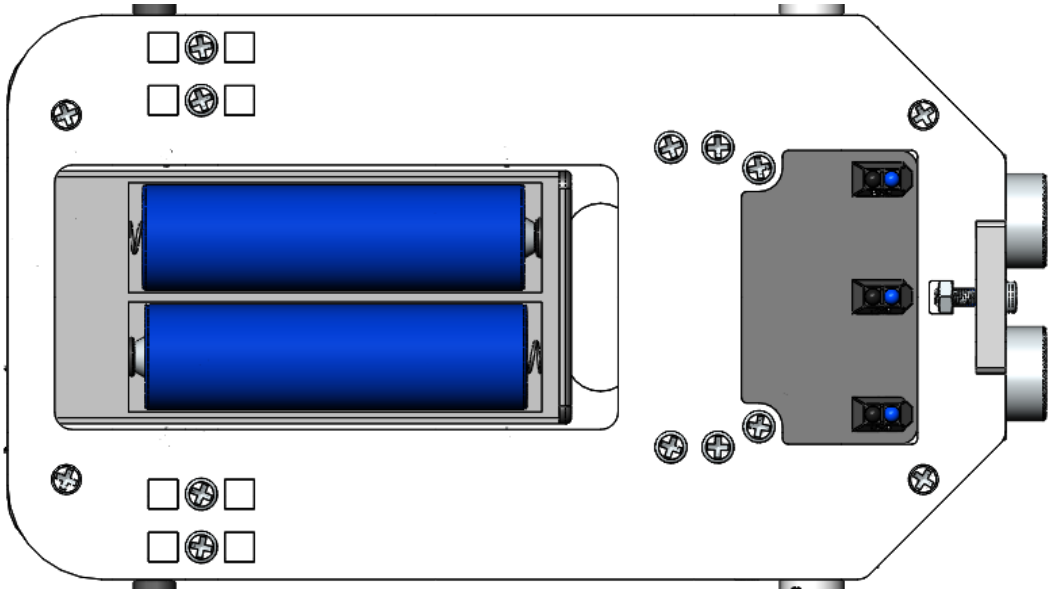
Step 21 Install the upper and lower acrylic sheets together using four M3x8 screws.



Note: The images of the car board may look different from the one you receive (V1.0 or V2.0). By the way, the interface design of the two version is the same, so the operation on them are the same. You can just follow this book to use it.

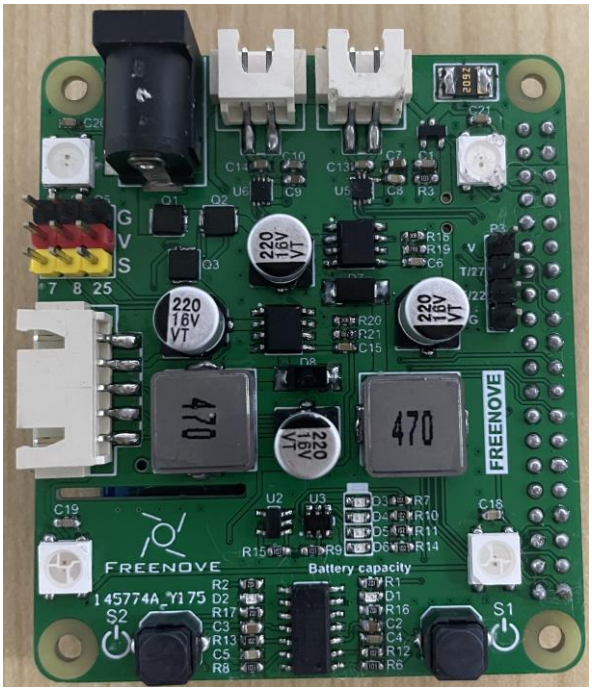
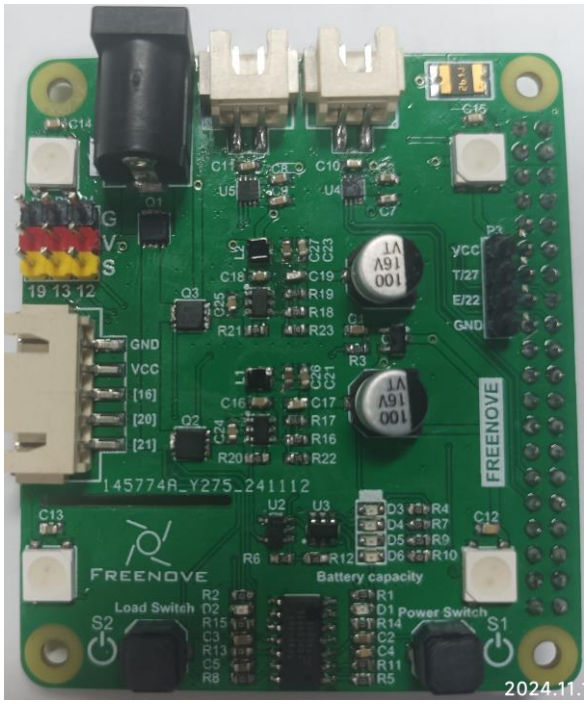
Tank Smart Car Board V1.0	Tank Smart Car Board V2.0
	

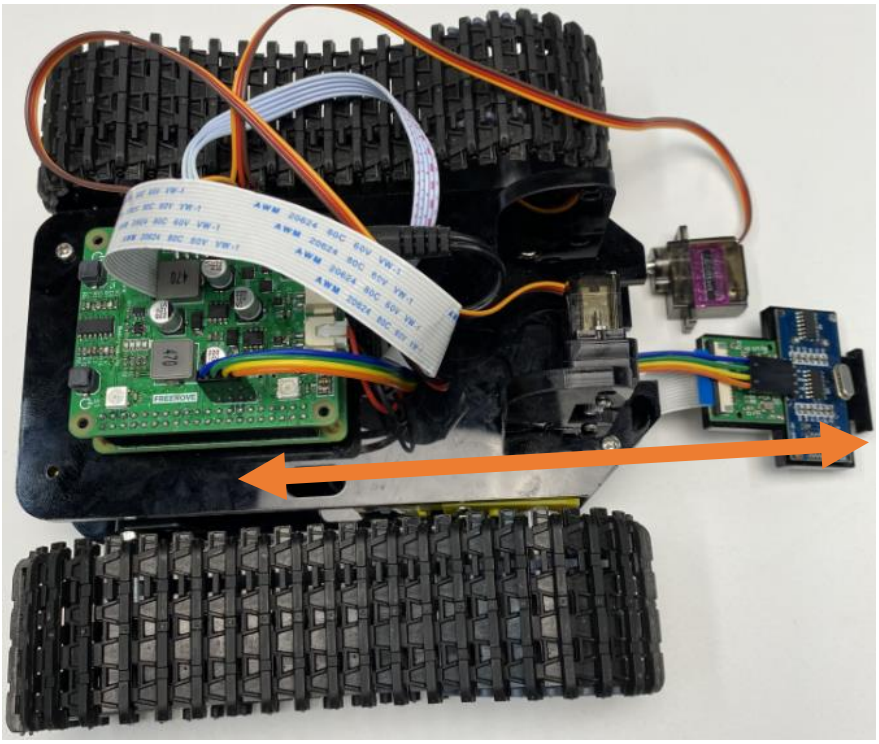
Step 22 Install the battery according to the battery box instructions.



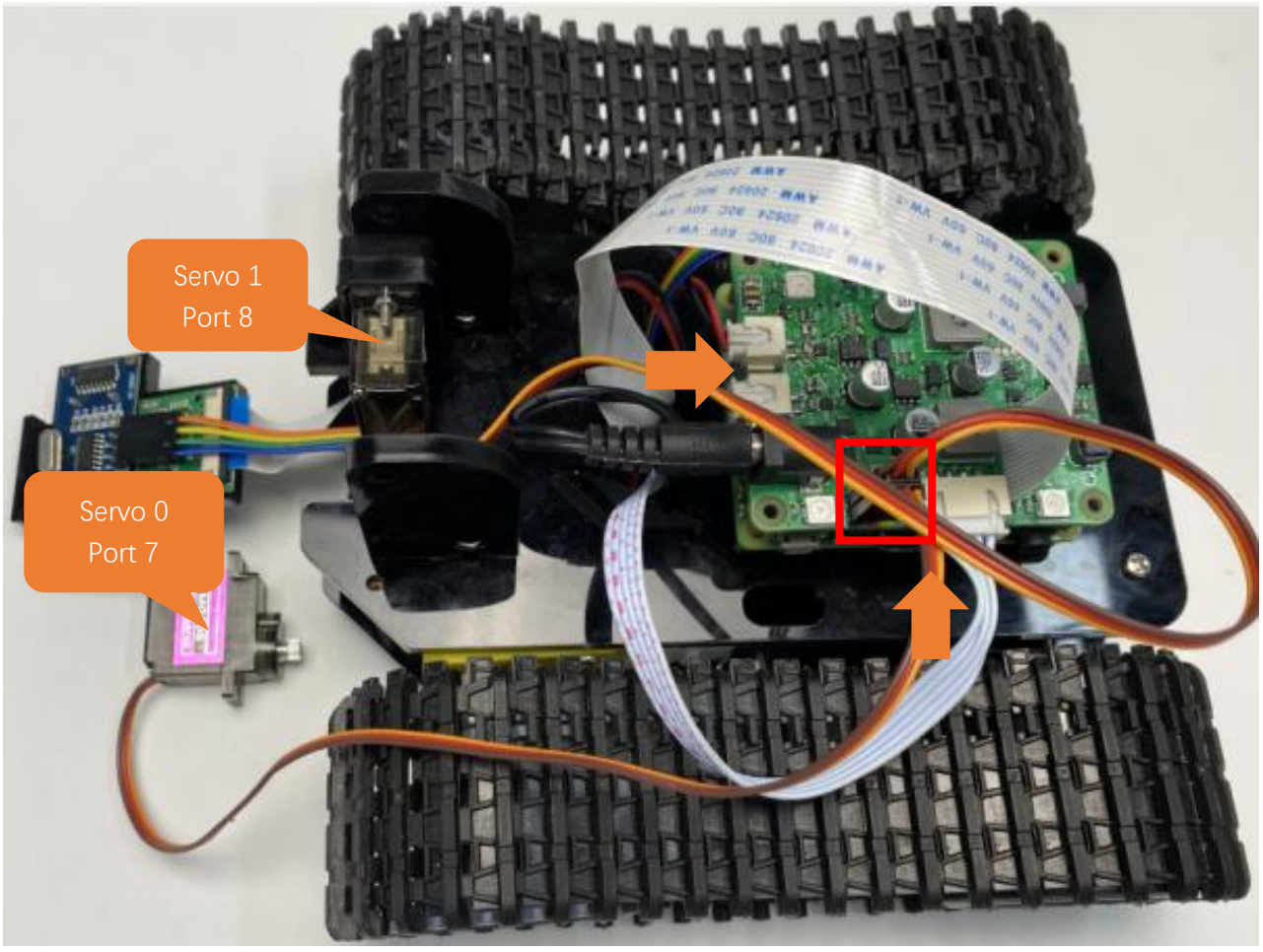
Turn on the switch of battery holder.

Step 23 Complete other wiring.

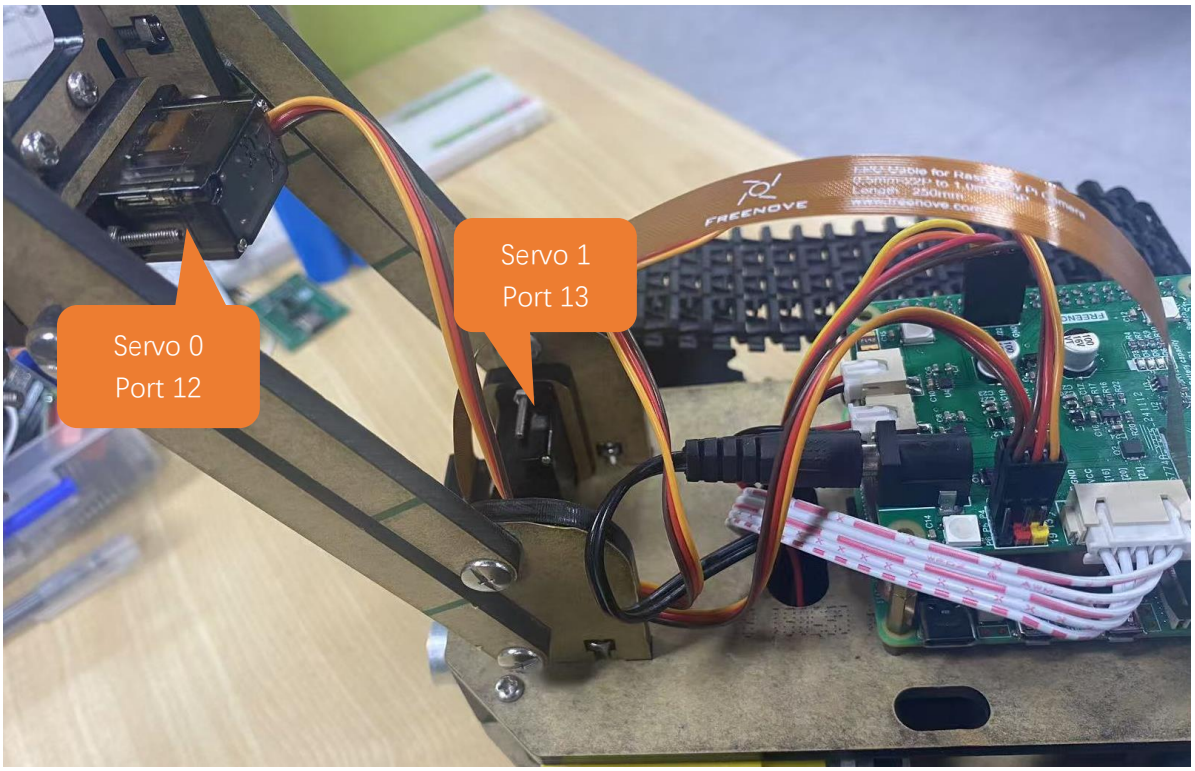
Tank Smart Car Board V1.0	Tank Smart Car Board V2.0
	



The servo wiring varies on the two board versions.
If your board is Version 1.0, please connect as shown below:



If it is version 2.0, connect in the following way.



Step 24 Make servo rotate to 90°.

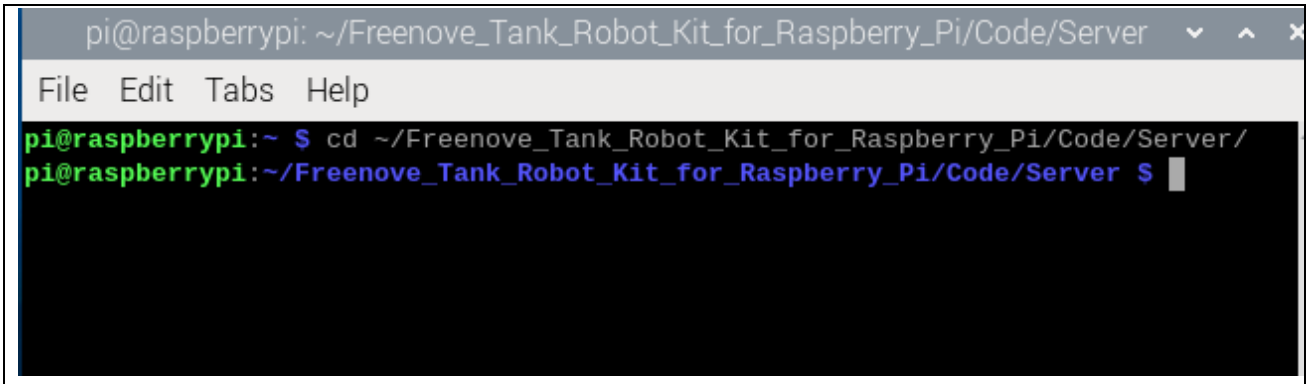
Turn on the two switches.

Execute following commands in terminal one by one.

assembling the servos, please perform the following steps to adjust the servos to appropriate angles; otherwise it will affect the final effect.

1. Run the following command to enter the directory Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server/
```



2. Run the following command the current directory.

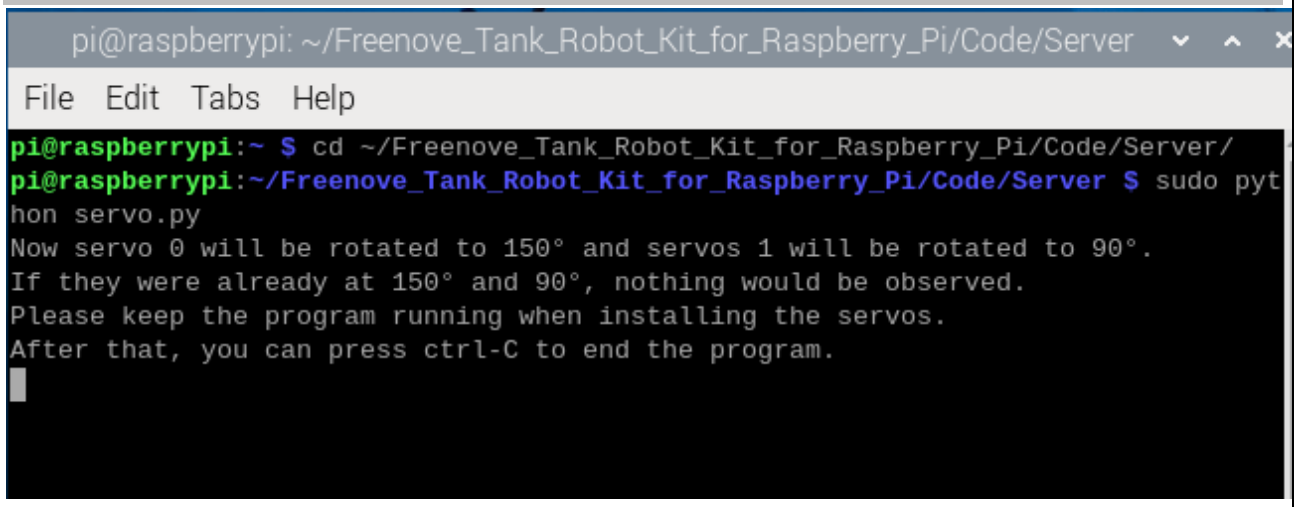
```
sudo python servo.py
```

If your Raspberry pie is not Pi5 and your PCB version is V1.0, run the following command before running the servo.py.

```
sudo pigpiod
```

If you want to end the pigpio process, run the following command.

```
sudo killall pigpiod
```



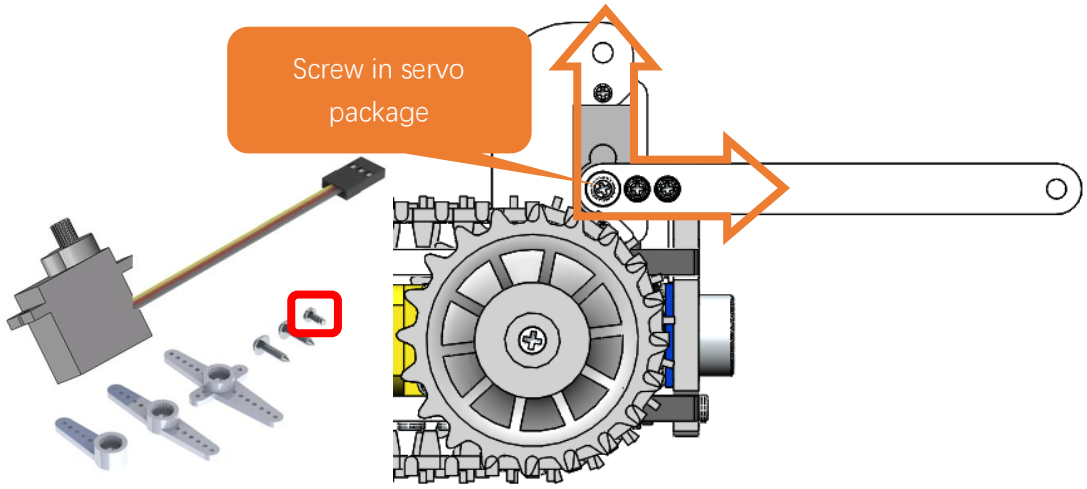
Result:

The two servos will spin to the designated angles. If the servos are already at that position, nothing will be observed upon running the code.

Keep the program running during the servo assembly process to avoid assembly offset.

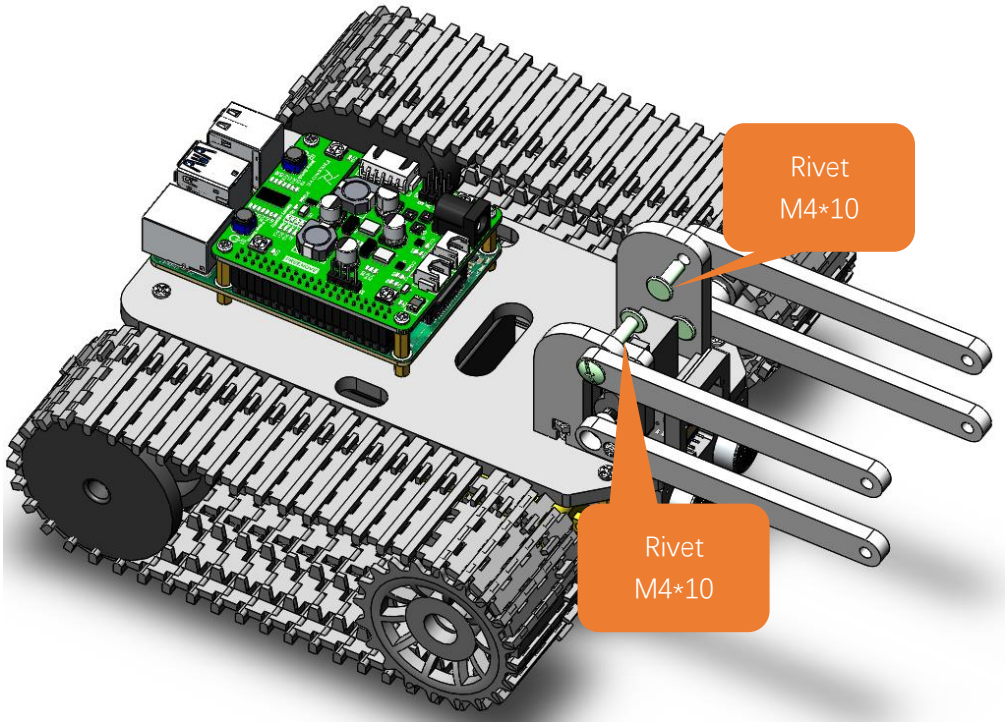
Step 26

Install using the screws in the Servo package as shown below.



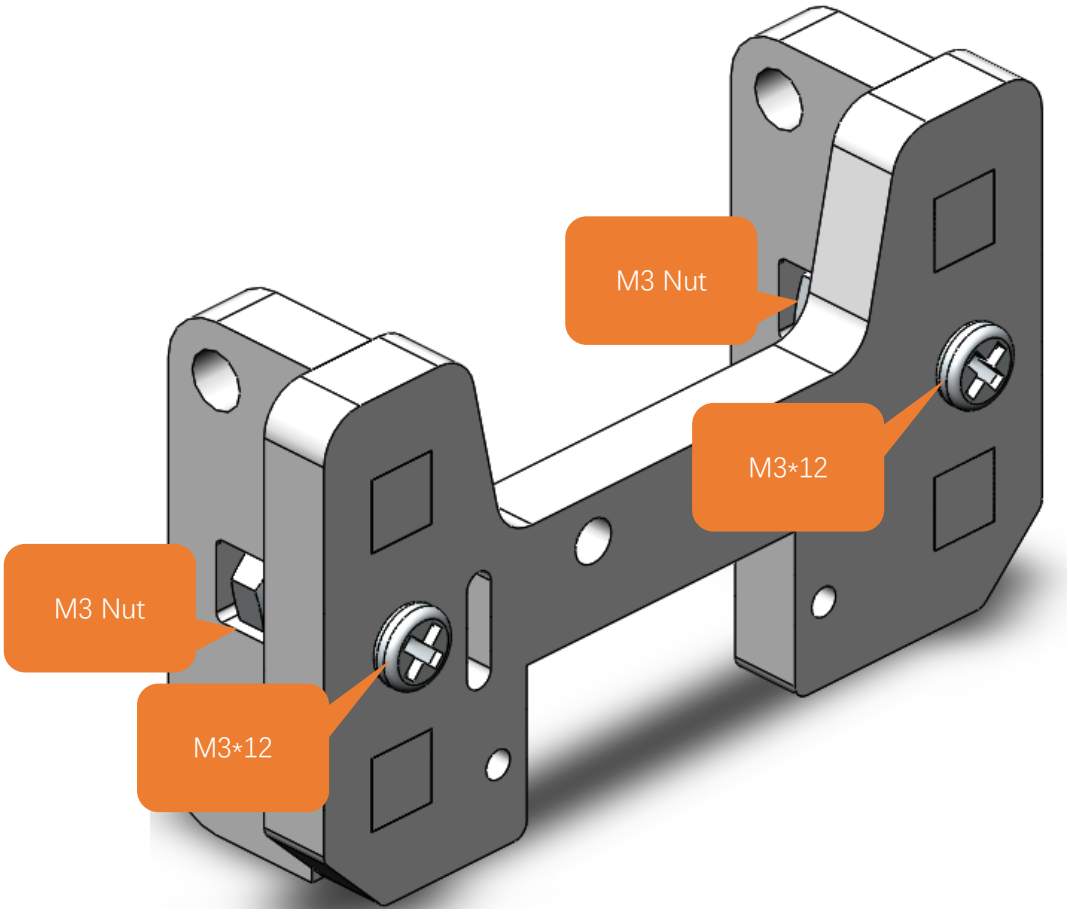
Install only after Servo has been rotated to a specified Angle.

Step 27 Mount the acrylic lever to the trolley using 2 sets of M4x10 Rivet.

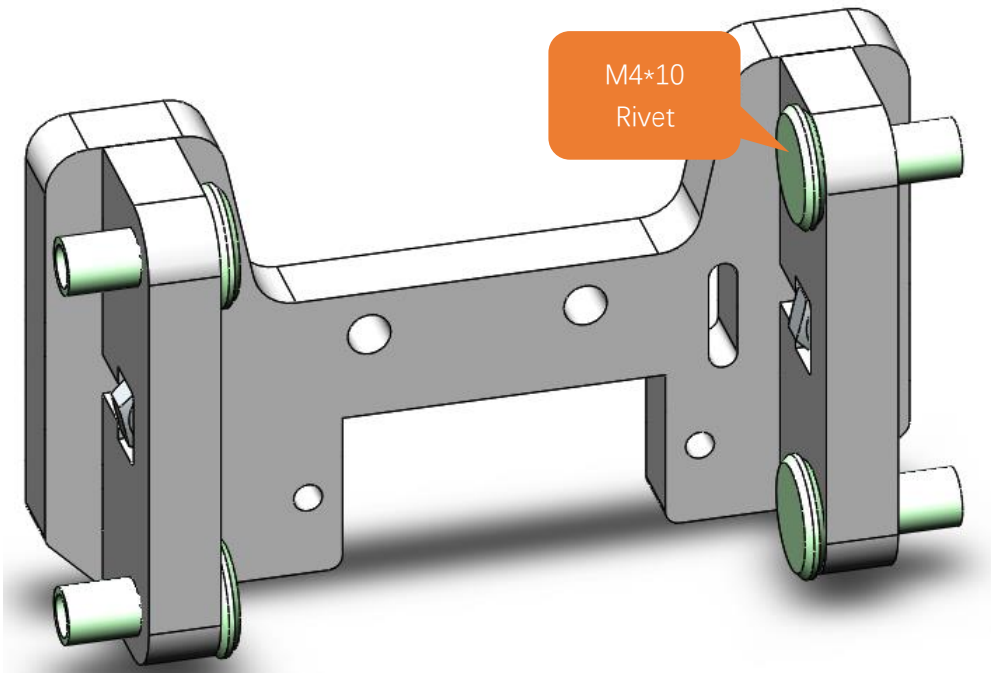


Step 28

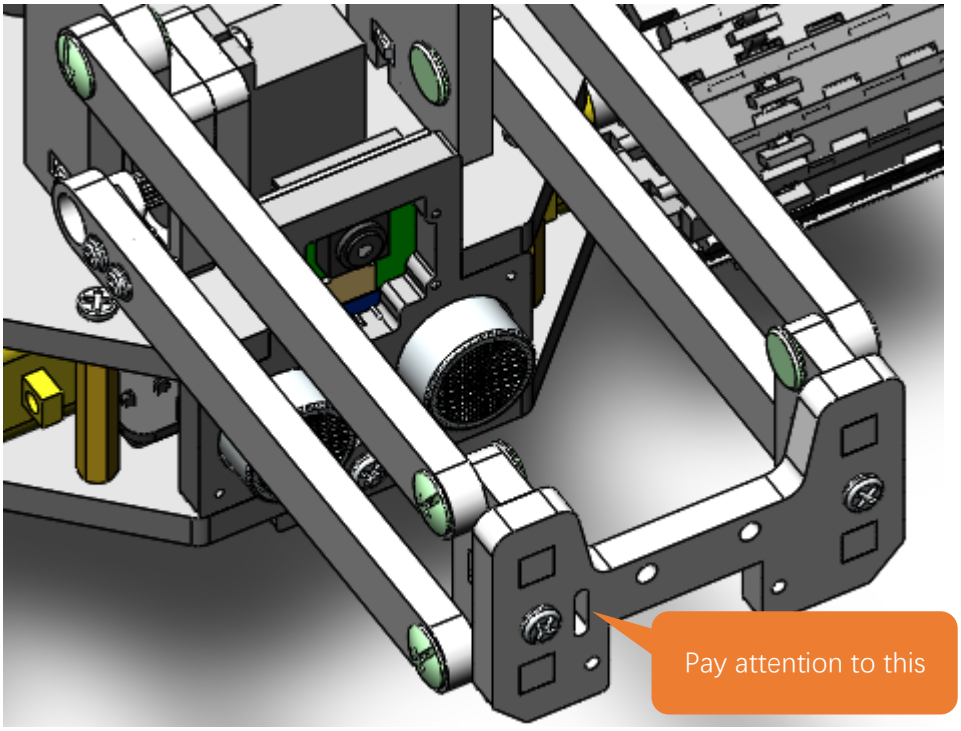
Using 2 M3x12 screws and 2 M3 nuts, install the acrylic as shown below.



Step 29 Install using 4 M4x10 rivets as shown below.



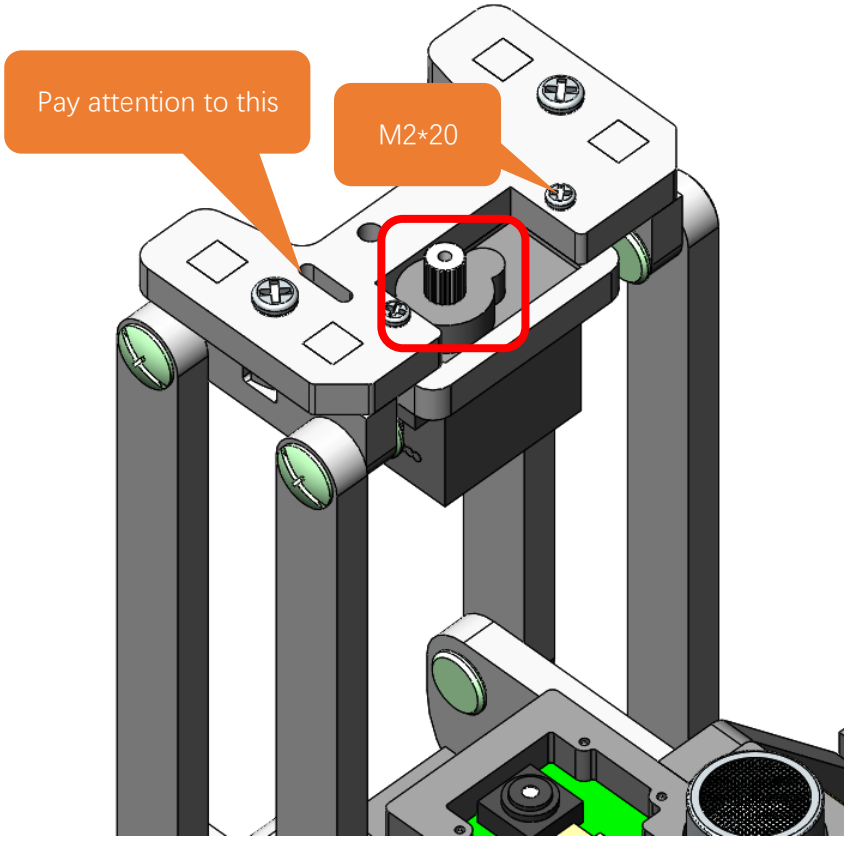
Step 30 Install the structure of the previous step on the trolley. Note the direction of the structure.



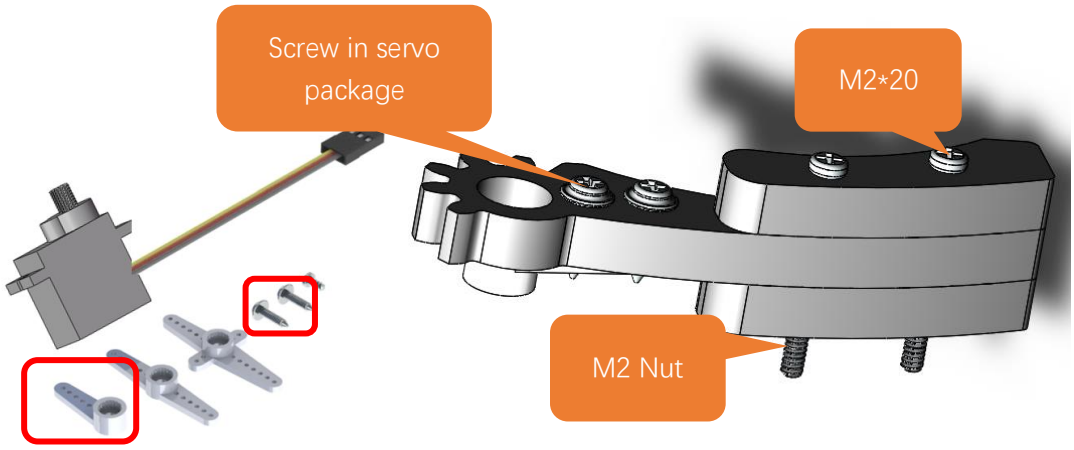
Step 31

Install the clear black steering gear to the trolley using 2 M2x20 screws and 2 M2 nuts.

Notice that the Servo axis of rotation is on the left.

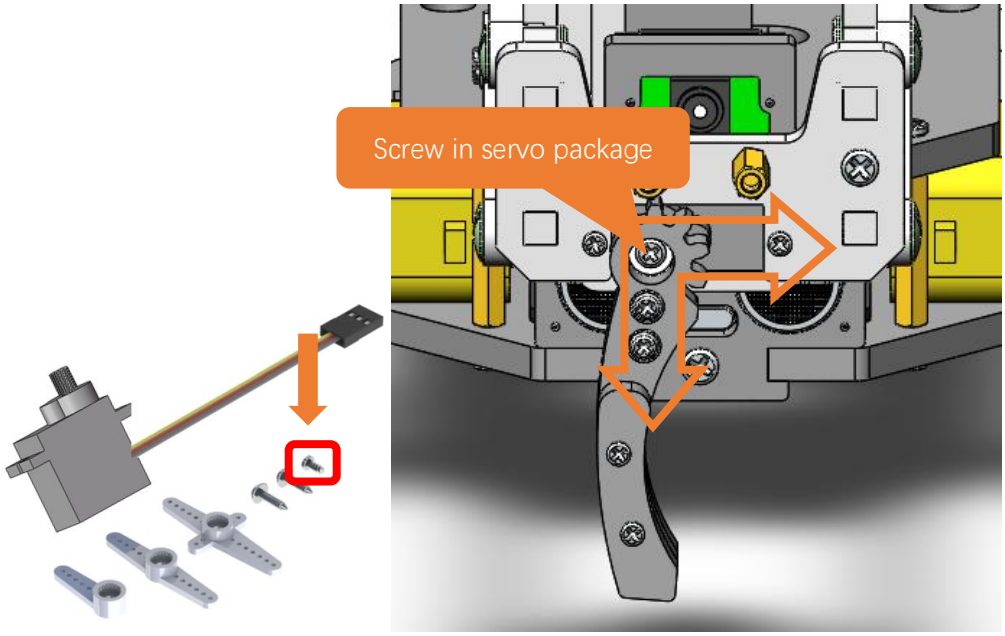


Step 32 Using 2 M2x20 screws and 2 M2 nuts, secure the three acrylics together. At the same time, install using the steering wheel and screws in the Servo package as shown below.



Step 33

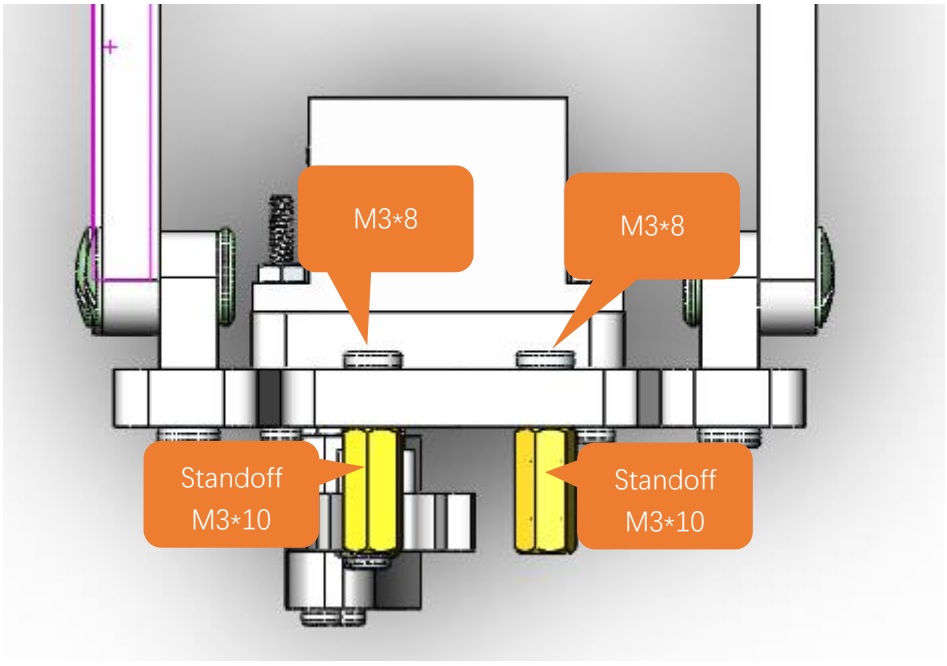
Install using the screws in the Servo package as shown below.



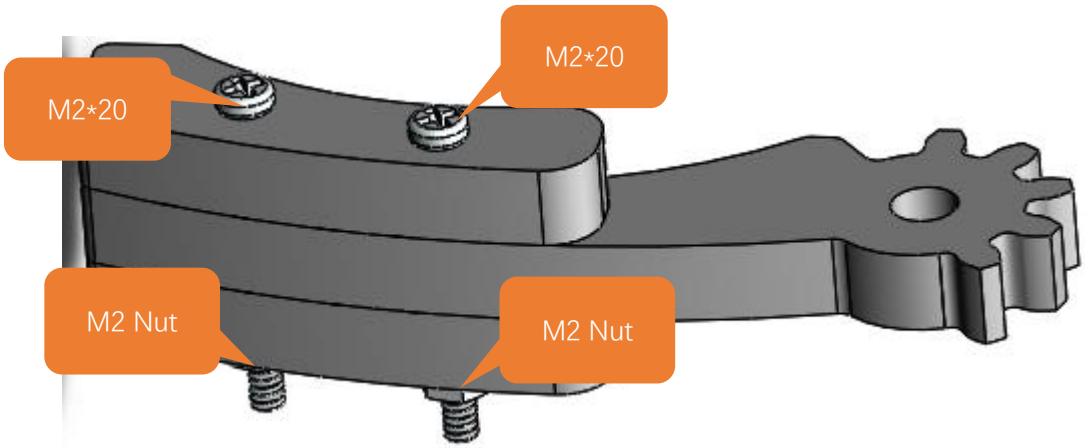
Install only after Servo has been rotated to a specified Angle.

You can press “Ctrl+C” to end the servo.py program after finishing servo assembly.

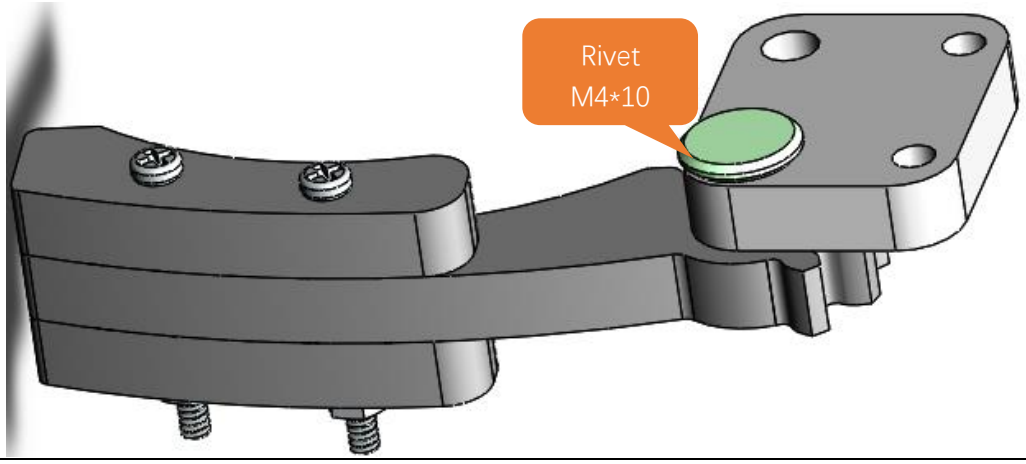
Step 34 Use two M3x8 screws to fix two M3x10 standoff into the acrylic.



Step 35 Using 2 M2x20 screws and 2 M2 nuts, secure the three acrylics together.

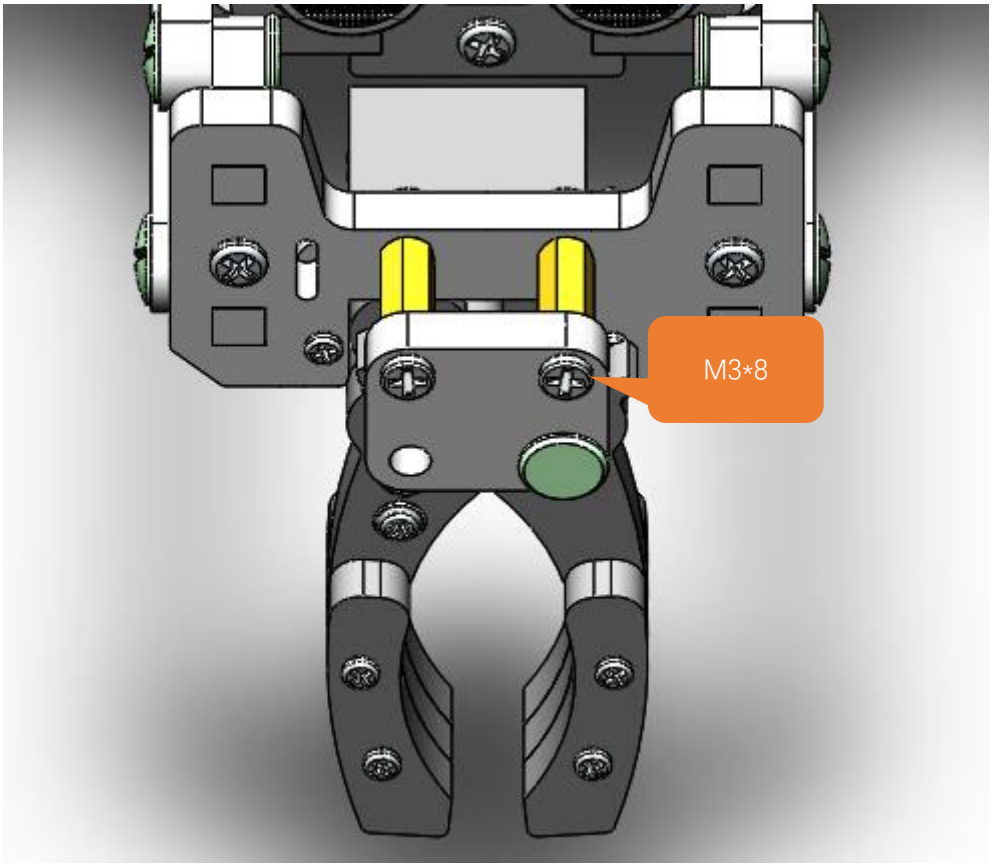


Step 36 Install using a set of M4x10 Rivet as shown below.



Step 37

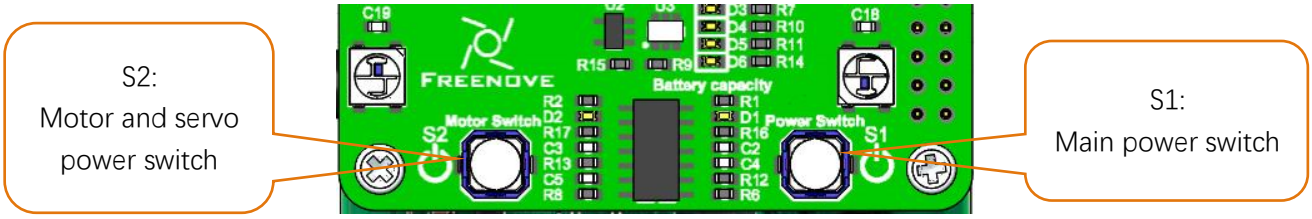
Use one M3x8 screw to install the structure of the previous step to the trolley, as shown below.



Chapter 3 Module test (necessary)

If you have any concerns, please feel free to contact us via support@freenove.com

This section requires that the car be equipped with batteries, and S1 power switch and S2 motor switch should be pressed until the corresponding power indicator lights up.



During the test, the motor will work. So you can disconnect the wheels or put it on the ground to avoid that it falls down and is damaged. Next, test RGB LED, motor, ultrasonic module, servo, etc.

Button S1 is the Raspberry PI power supply switch. When you only press S1, you can perform other tests except motor and servo.

Button S2 is the motor and servo power switch. When you press S1 and S2, you can test the motor and servo.

You can still power Raspberry Pi with a power supply Cable when switches are pressed.

If you have never learned python before, you can learn some basic knowledge via the link below:

https://python.swaroopch.com/basics.html

Motor

Run program

Before running the program, please make sure all needed libraries are installed; otherwise, the program will fail to run.

Open the terminal of Raspberry Pi. Enter the following commands to test the motor.

1. Use the cd command to enter the directory where test.py is located.

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Execute test.py command:

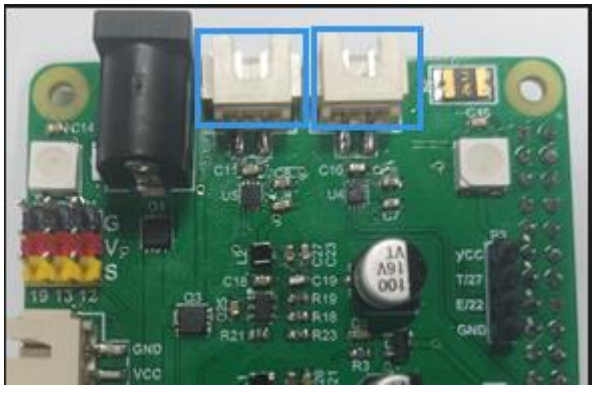
```
sudo python test.py Motor
```

Result:

The car moves forward for 1 seconds, then moves back for 1 seconds, then turns left for 1 seconds, turns right for 1 seconds, then stops. You can press **"Ctrl + C"** to end the program ahead of time. **If the car doesn't work normally, please check if both switches are pressed.**

```
pi@raspberrypi: ~/Freenove_Tank_R...ot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo python
test.py Motor
Program is starting ...
The car is moving forward
The car is going backwards
The car is turning left
The car is turning right

End of program
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

If the direction is reversed, it moves back then move forward, please exchange wiring of the two motors.



The code for motor in test.py is as below:

```
def test_Motor():
    from motor import tankMotor              # Import the tankMotor class from the motor
module
    import time                              # Import the time module for sleep functionality
    print('Program is starting ... ')        # Print a start message
    PWM = tankMotor()                        # Initialize the tankMotor instance
    try:
        PWM.setMotorModel(2000, 2000)        # Move the car forward
        print("The car is moving forward")   # Print a status message
        time.sleep(1)                        # Wait for 1 second
        PWM.setMotorModel(-2000, -2000)      # Move the car backward
        print("The car is going backwards")  # Print a status message
        time.sleep(1)                        # Wait for 1 second
        PWM.setMotorModel(-2000, 2000)       # Turn the car left
        print("The car is turning left")     # Print a status message
        time.sleep(1)                        # Wait for 1 second
        PWM.setMotorModel(2000, -2000)       # Turn the car right
        print("The car is turning right")    # Print a status message
        time.sleep(1)                        # Wait for 1 second
        PWM.setMotorModel(0, 0)              # Stop the car
        print("\nEnd of program")            # Print an end message
    except KeyboardInterrupt:                # Handle keyboard interrupt (Ctrl+C)
        PWM.setMotorModel(0, 0)              # Stop the car
        print("\nEnd of program")            # Print an end message
```

If you are interested in the underlying code, you can refer to motor.py for more details.

Reference

setMotorModel(data1,data2)

This function has two input parameters, which control the left motor and the right motor respectively. When the input parameter is in the range of 0~4095, the motor rotates forward. In the range of -4095~0, the motor rotates in reverse. The larger the absolute value, the faster the motor. When the input is 0, the motor will stop. If the function input is as follows: setMotorModel(2000,2000), the two motors will rotate forward and the car will move forward.

Infrared Line tracking module

Run program

Enter the following command in the terminal to test line tracking module.

If the terminal displays the directory as below (where test.py is located), you can **directly** execute the test.py command.

```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Execute test.py command:

```
sudo python test.py Infrared
```

```
pi@raspberrypi: ~/Freenove_Tank_R...ot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo python
test.py Infrared
Program is starting ...
```

Result:

When the black tape is at the left of the sensor, it will print “Left” on the terminal; when it is at the middle, print “Middle”; and at the right, print “Right”. You can press “CTRL+C” to end the program.

```
pi@raspberrypi: ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo pyt
hon test.py Infrared
Program is starting ...
Left
Left
Middle
Right
Left
Middle
Right
^C
End of program
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

The code is as below:

```
def test_Infrared():
    from infrared import Infrared      # Import the Infrared class from the infrared module
    import time                        # Import the time module for sleep functionality
    print('Program is starting ... ')  # Print a start message
    infrared = Infrared()              # Initialize the Infrared instance
    try:
        while True:
            if infrared.read_one_infrared(1) == 0 and infrared.read_one_infrared(2) == 1 and
infrared.read_one_infrared(3) == 0:
                print('Middle')        # Print a middle detection message
            elif infrared.read_one_infrared(1) == 1 and infrared.read_one_infrared(2) == 0 and
infrared.read_one_infrared(3) == 1:
                print('Middle')        # Print a middle detection message
            elif infrared.read_one_infrared(1) == 0 and infrared.read_one_infrared(2) == 0 and
infrared.read_one_infrared(3) == 1:
                print('Right')         # Print a right detection message
            elif infrared.read_one_infrared(1) == 1 and infrared.read_one_infrared(2) == 1 and
infrared.read_one_infrared(3) == 0:
                print('Right')         # Print a right detection message
            elif infrared.read_one_infrared(1) == 1 and infrared.read_one_infrared(2) == 0 and
infrared.read_one_infrared(3) == 0:
                print('Left')          # Print a left detection message
            elif infrared.read_one_infrared(1) == 0 and infrared.read_one_infrared(2) == 1 and
infrared.read_one_infrared(3) == 1:
                print('Left')          # Print a left detection message
            time.sleep(0.1)            # Wait for 0.1 seconds
    except KeyboardInterrupt:          # Handle keyboard interrupt (Ctrl+C)
        print("\nEnd of program")      # Print an end message
```

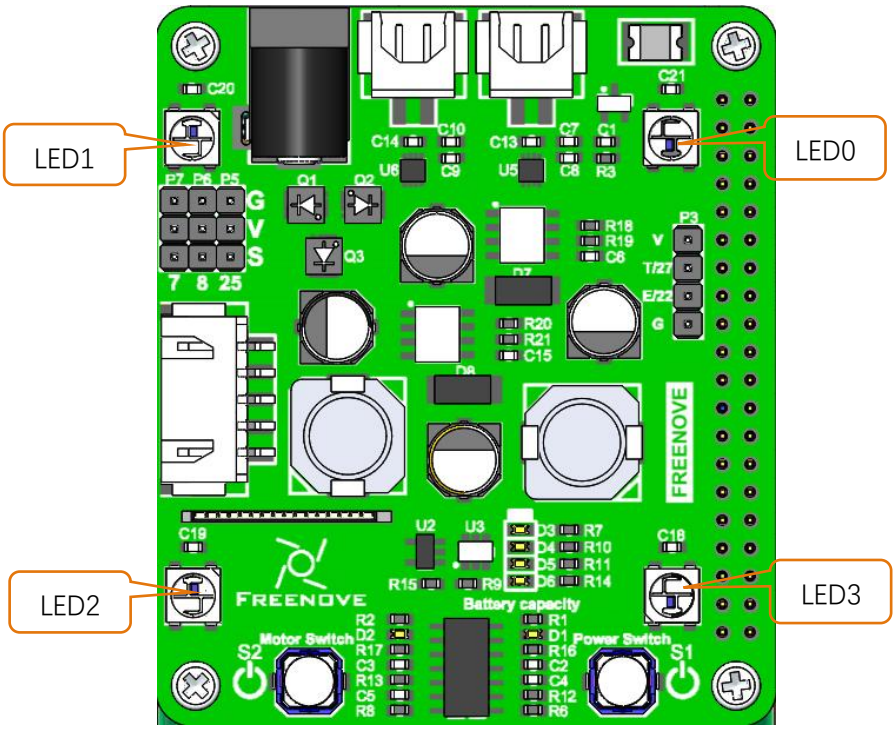
you are interested in underlying code, please refer to infrared.py file.

Reference

GPIO.input(IO)
This function has an input parameter. If the IO input is high level, GPIO.input(IO) returns True. If the IO input is low level, GPIO.input(IO) returns False.

LED

There are 4 RGB LEDs on the smart car board, as shownbelow. You can control them separately.



Run program

Enter the following commands to test LEDs.

If the terminal displays the directory as below (where test.py is located), you can **directly** execute the test.py command.

```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

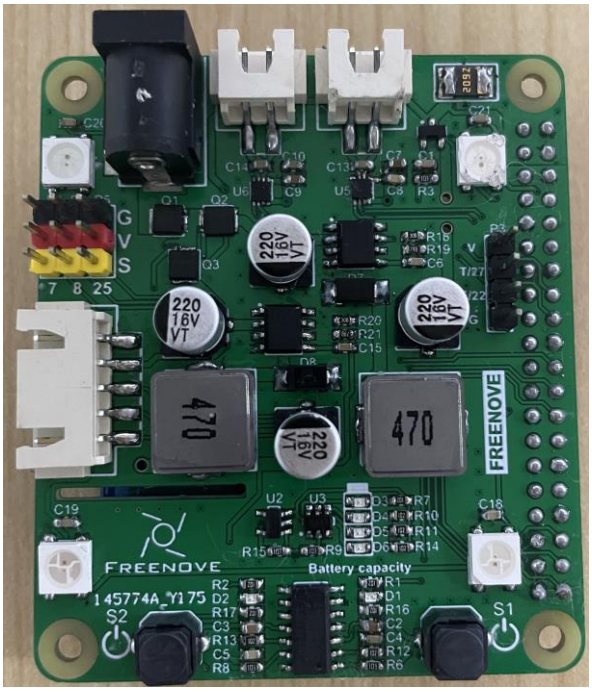
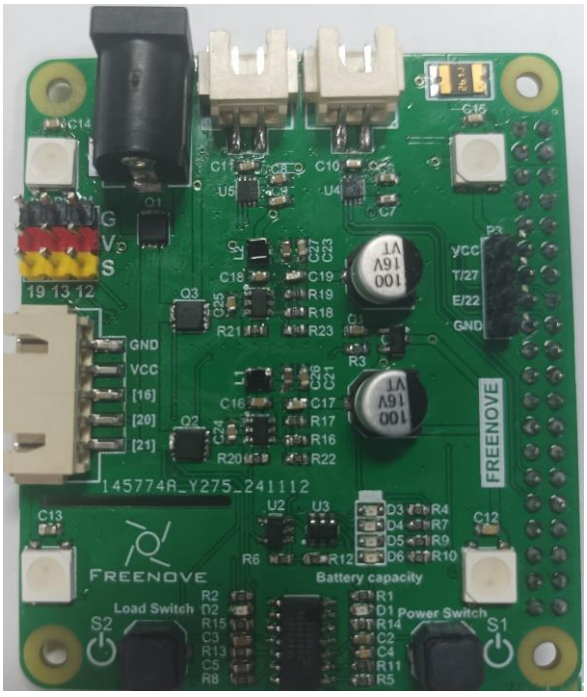
```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Execute test.py command:

```
sudo python test.py Led
```

```
pi@raspberrypi: ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo pyt
hon test.py Led
Program is starting ...
ledIndex test
colorWipe test
theaterChaseRainbow test
rainbow test
```

Note: On the board V1.0, the WS821x LEDs are connected to GPIO 18, which can be used on Raspberry Pi series 1-4, but not on Rpi 5.
On version 2.0, we connect them to GPIO10, which applies SPI interface, so it is compatible with Raspberry Pi series 1-5.

Tank Smart Car Board V1.0	Tank Smart Car Board V2.0
	

At the first time you run the code, the system will check your hardware version. Please enter the version according to the board you receive. You can refer to the images above to check the version.

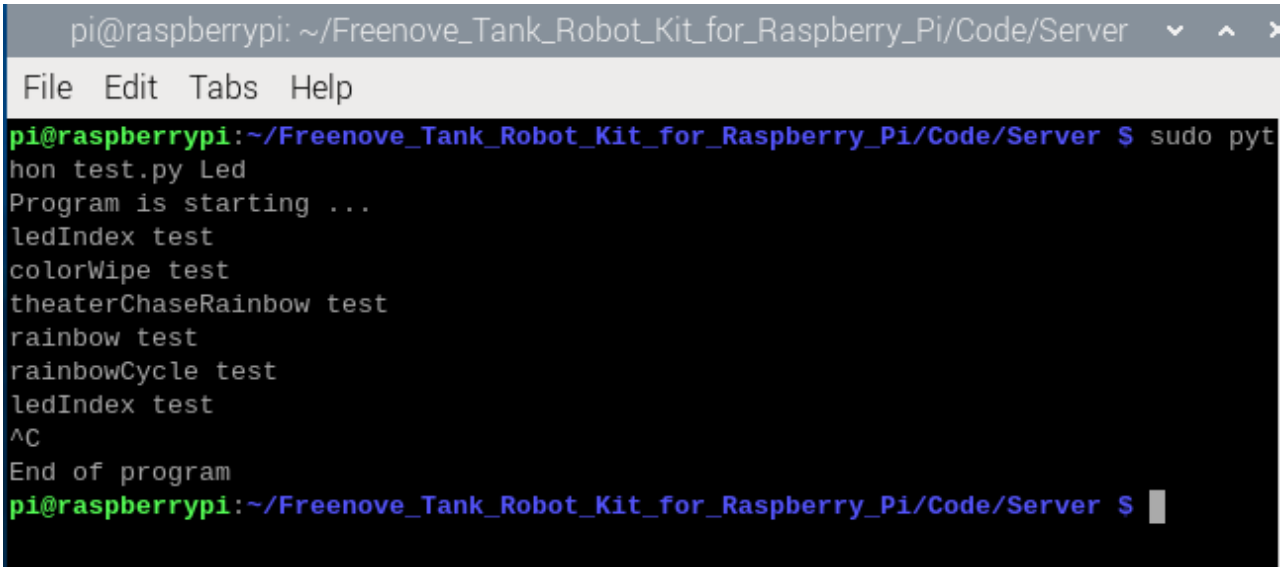
```
pi@raspberrypi: ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server/
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo python test.py led
Program is starting ...
Parameter file params.json does not exist or contains invalid parameters.
Please enter the hardware versions.
Enter PCB Version (1 or 2):
```

After you input the version (here we take verion 2 as an example), the LED test program will be run. If the LEDs do not respond, please check whether you type in a wrong version.

```
pi@raspberrypi: ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server/
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo python test.py led
Program is starting ...
Parameter file params.json does not exist or contains invalid parameters.
Please enter the hardware versions.
Enter PCB Version (1 or 2): 2
ledIndex test
colorWipe test
theaterChaseRainbow test
```

Result:

When the program is running, the LEDs will blink with different modes and colors. The order of the modes is: ledIndex->colorWipe->theaterChaseRainbow->rainbow->rainbowCycle
You can press CTRL+C to end the program.

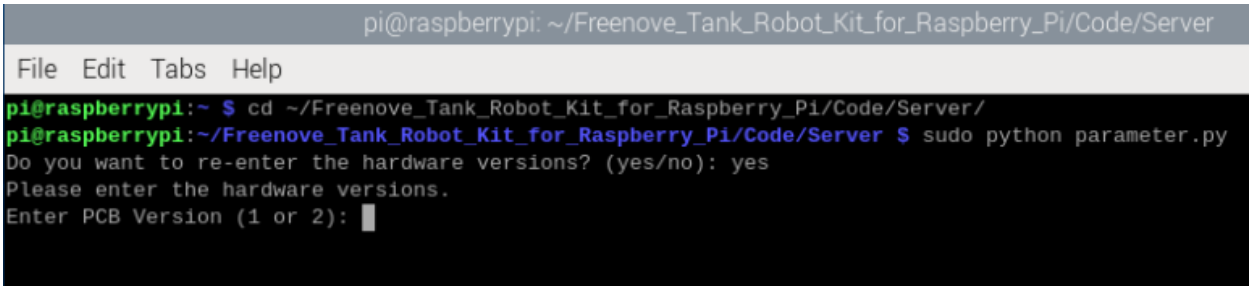


Note, the PCB version must be correctly input, which depends whether the code can run successfully. If you input a correct version, please follow the steps below to configure it again.

1. Enter the directory /Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server/, and run the parameter.py under Server folder.

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server/
sudo python parameter.py
```

2. Input the number according to the board version.



The code of test.py is as below:

```
1  def test_Led():
2      from led import Led                    # Import the Led class from the led module
3      import time                            # Import the time module for sleep
4  functionality
5      print('Program is starting ... ')      # Print a start message
6      led = Led()                            # Initialize the Led instance
7      try:
8          while True:
9              print("ledIndex test")         # Print a test message
```

```
10              led.ledIndex(0x01, 255, 0, 0)       # Set LED 1 to red
11              led.ledIndex(0x02, 0, 255, 0)       # Set LED 2 to green
12              led.ledIndex(0x04, 0, 0, 255)       # Set LED 3 to blue
13              led.ledIndex(0x08, 255, 255, 255)   # Set LED 4 to white
14              time.sleep(3)                       # Wait for 3 seconds
15
16              print("colorWipe test")             # Print a test message
17              led.colorWipe((255, 0, 0))          # Perform a red color wipe
18              led.colorWipe((0, 255, 0))          # Perform a green color wipe
19              led.colorWipe((0, 0, 255))          # Perform a blue color wipe
20              time.sleep(1)                       # Wait for 1 second
21
22              print("theaterChaseRainbow test")   # Print a test message
23              led.theaterChaseRainbow()           # Perform a theater chase rainbow effect
24              print("rainbow test")               # Print a test message
25              led.rainbow()                       # Perform a rainbow effect
26              print("rainbowCycle test")          # Print a test message
27              led.rainbowCycle()                  # Perform a rainbow cycle effect
28
29              led.colorWipe((0, 0, 0), 10)        # Turn off all LEDs
30      except KeyboardInterrupt:                   # Handle keyboard interrupt (Ctrl+C)
31          led.colorWipe((0, 0, 0), 10)            # Turn off all LEDs
32          print("\nEnd of program")               # Print an end message
```

If you are interested in the underlying code, you can check the led.py file.

Servo

Run program

Enter the following commands in the terminal to test servos.

If the terminal displays the directory as below (where test.py is located), you can directly execute the test.py command.

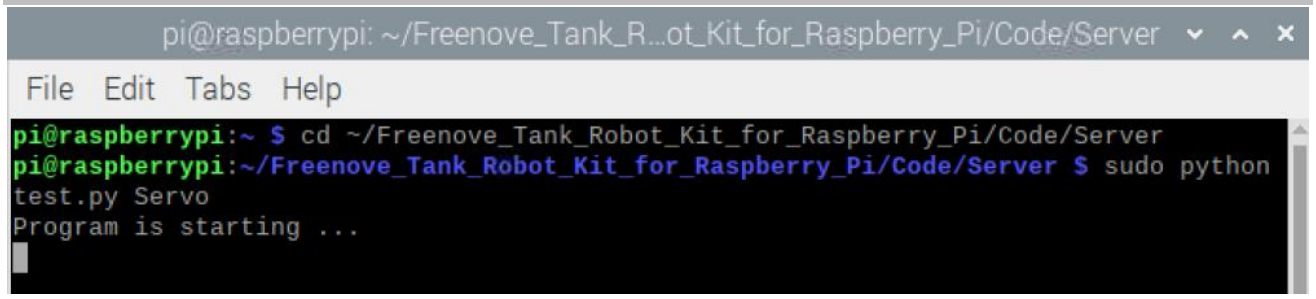
```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Execute test.py command:

```
sudo python test.py Servo
```



If your Raspberry pie is not Pi5 and your PCB version is V1.0, run the following command before running the servo.py.

```
sudo pigpiod
```

If you want to end the pigpio process, run the following command.

```
sudo killall pigpiod
```

The servo will repeat the following: servo 0 is closed gradually, servo 1 from top to bottom, and then from bottom to top, servo 0 is opened gradually. You can press Ctrl + C to end the program.

When testing servo steering gear, you need to observe whether servo 0 is closed to the minimum, whether servo 1 hits the ground during the descent,

When servo 0 is just closed to the minimum and servo 1 drops close to the ground without touching the ground, it means that your servo 0 and servo 1 are properly installed.

When other conditions occur, please check that the installation steps of the relevant servo are correct.

For details, see Step 24 in Chapter 2.

The code is as below:

```
1  def test_Servo():
2      from servo import Servo              # Import the Servo class from the servo module
3      import time                          # Import the time module for sleep functionality
4      print('Program is starting ... ')   # Print a start message
5      servo = Servo()                      # Initialize the Servo instance
6      try:
7          while True:
```

```
8             for i in range(90, 150, 1):
9                 servo.setServoAngle('0', i)  # Set servo 0 to angle i
10                time.sleep(0.01)             # Wait for 0.01 seconds
11            for i in range(140, 90, -1):
12                servo.setServoAngle('1', i)  # Set servo 1 to angle i
13                time.sleep(0.01)             # Wait for 0.01 seconds
14            for i in range(90, 140, 1):
15                servo.setServoAngle('1', i)  # Set servo 1 to angle i
16                time.sleep(0.01)             # Wait for 0.01 seconds
17            for i in range(150, 90, -1):
18                servo.setServoAngle('0', i)  # Set servo 0 to angle i
19                time.sleep(0.01)             # Wait for 0.01 seconds
20    except KeyboardInterrupt:                # Handle keyboard interrupt (Ctrl+C)
21        servo.setServoAngle('0', 90)           # Set servo 0 to 90 degrees
22        servo.setServoAngle('1', 140)          # Set servo 1 to 140 degrees
23        print("\nEnd of program")             # Print an end message
```

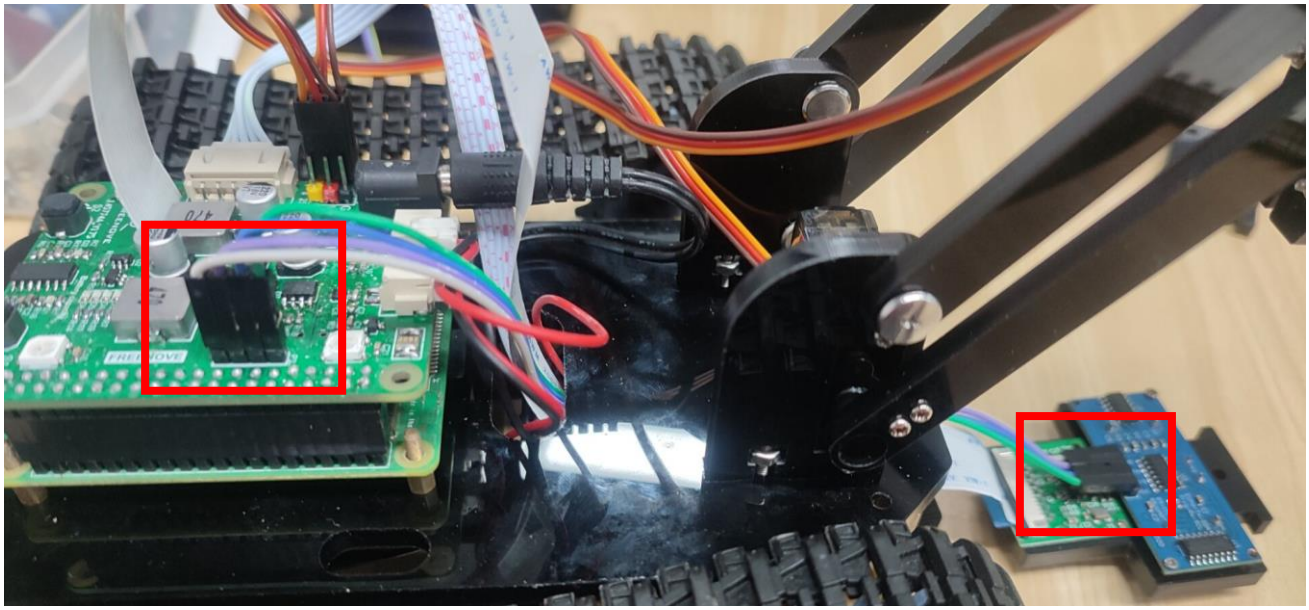
Reference

setServoAngle(Servo,angle)

There are 2 parameters.
The first one is related to servo index.
The second one is related to the angle of servos.
For example,
setServoAngle ('0',90) makes servo0 rotate to 90°.
setServoAngle ('1',90) makes servo1 rotate to 90°.

Ultrasonic module

Next, use jumper wires F/F to connect ultrasonic module with pins on smart car board.



When connecting the ultrasonic module, you should keep the silk screen of the ultrasonic module and the smart car board consistent. Vcc should be connected to 5V, Trig to TRIG, Echo to ECHO, and Gnd to GND. If the connection is wrong, for example, if Vcc is connected to GND, and Gnd is connected to 5V, it will cause the damage to ultrasonic module. After the wiring is completed, you can start testing.

Run program

Enter following command in the terminal:

If the terminal displays the directory as below (where test.py is located). You can **directly** execute the test.py command.

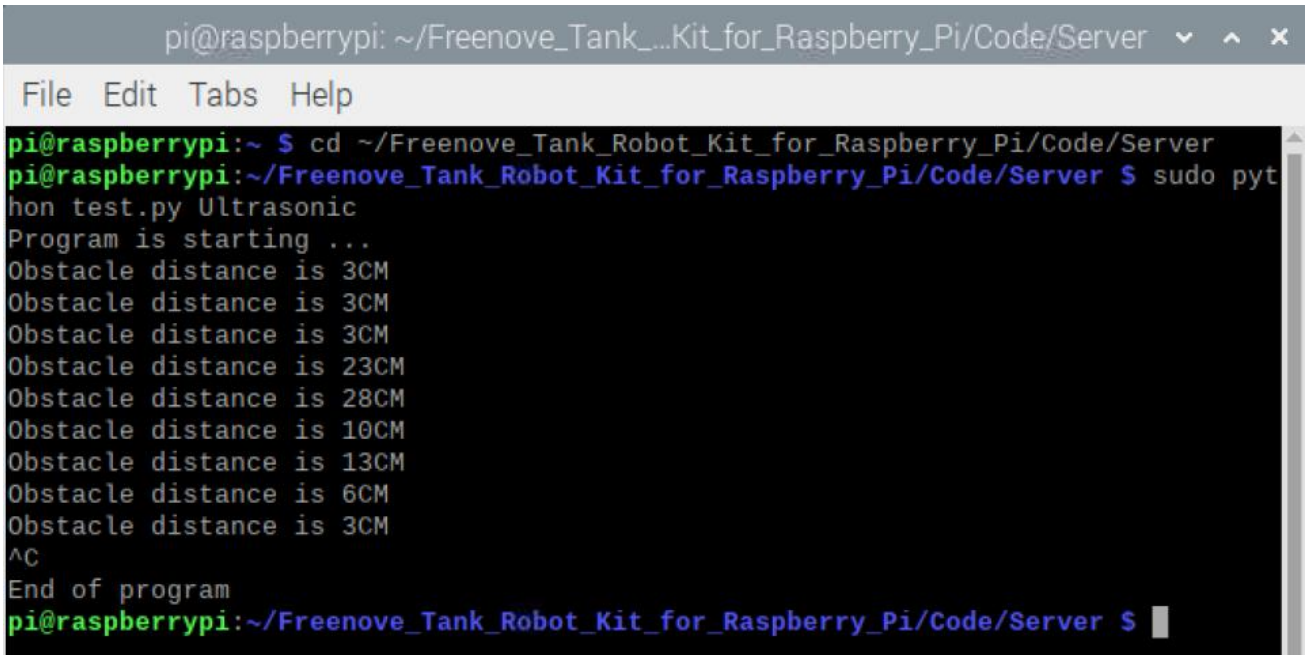
```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Execute test.py command:

```
sudo python test.py Ultrasonic
```



Result:

Every 0.3s, the distance between the obstacle and the ultrasonic module will be printed out, and you can press "Ctrl + C" to end the program.

The code is as below:

```
1  def test_Ultrasonic():
2      from ultrasonic import Ultrasonic # Import the Ultrasonic class from the ultrasonic module
3      import time                       # Import the time module for sleep functionality
4      print('Program is starting ... ') # Print a start message
5      ultrasonic = Ultrasonic()         # Initialize the Ultrasonic instance
6      try:
7          while True:
8              distance = ultrasonic.get_distance()  # Get the distance to the obstacle
9              print("Obstacle distance is " + str(distance) + "CM")  # Print the distance
10             time.sleep(0.3)                       # Wait for 0.3 seconds
11     except KeyboardInterrupt:                     # Handle keyboard interrupt (Ctrl+C)
12         print("\nEnd of program")                  # Print an end message
```

Reference

get_distance()
This function is used to obtain the distance between ultrasonic module and obstacles in front of it, with unit CM.

Camera

Next let us connect the camera to smart car board.

First **turn off S1** (Power Switch) and **S2**, **shut down Raspberry Pi** and disconnect power cable. If the data cable is used to power the Raspberry Pi, disconnect the data cable and install the CSI camera to the Raspberry Pi camera interface when the Raspberry Pi is powered off.

The CSI camera must be connected or disconnected under no power and when Raspberry Pi is shut down, or the camera may be burned.

Run Program

If the terminal displays the directory as below, you can **directly** run the Led.py.

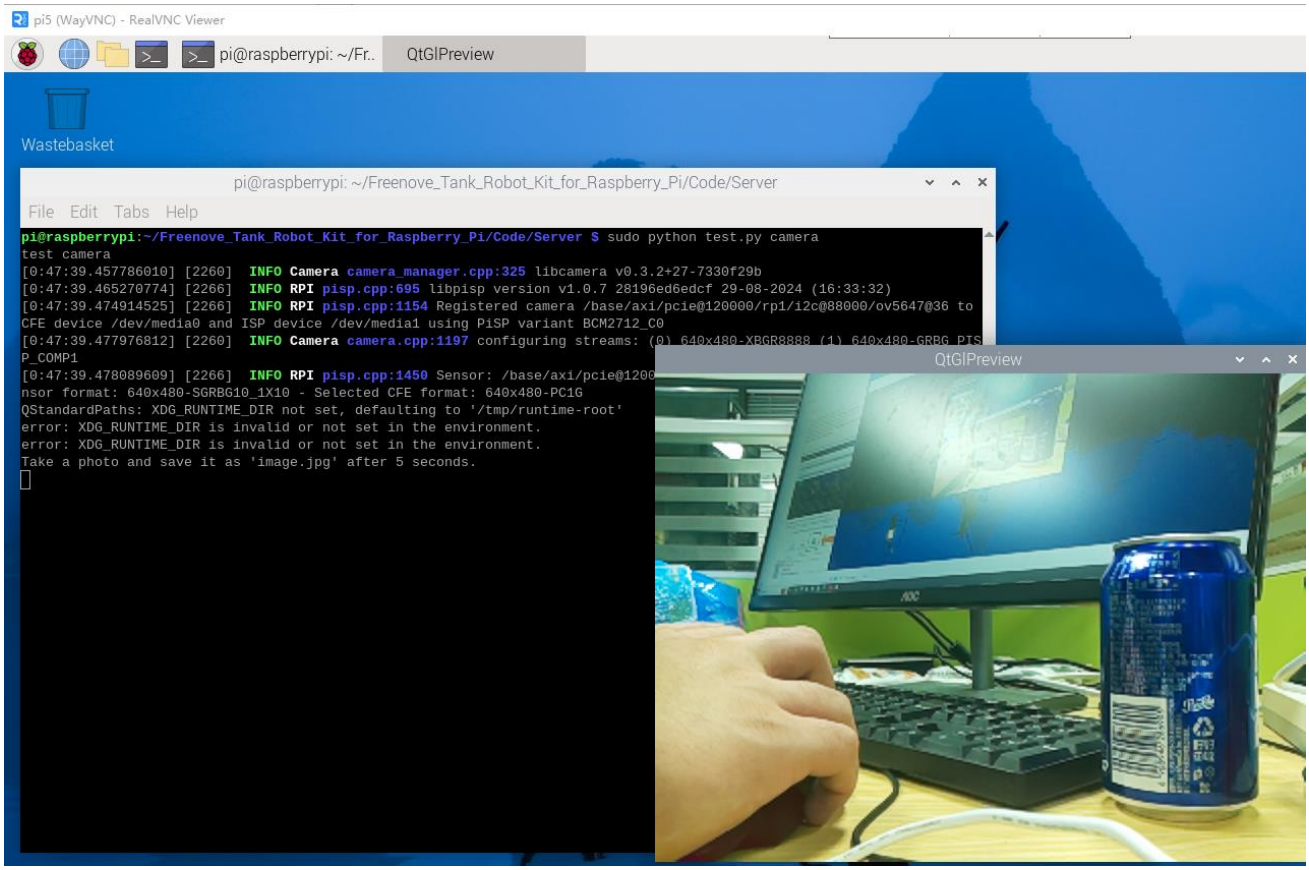
```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Execute test.py command:

```
sudo python text.py camera
```



Results:

Five seconds after the program runs, it will take a picture in jpg format and save it to the current directory. You can run the ls command to see if it is there.

```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo python test.py camera
test camera
[0:47:39.457786010] [2260]  INFO Camera camera_manager.cpp:325 libcamera v0.3.2+27-7330f29b
[0:47:39.465270774] [2266]  INFO RPI pisp.cpp:695 libpisp version v1.0.7 28196ed6edcf 29-08-2024 (16:33:32)
[0:47:39.474914525] [2266]  INFO RPI pisp.cpp:1154 Registered camera /base/axi/pcie@120000/rp1/i2c@88000/ov5647@36 to
CFE device /dev/media0 and ISP device /dev/media1 using PiSP variant BCM2712_C0
[0:47:39.477976812] [2260]  INFO Camera camera.cpp:1197 configuring streams: (0) 640x480-XBGR8888 (1) 640x480-GRBG_PI
P_COMP1
[0:47:39.478089609] [2266]  INFO RPI pisp.cpp:1450 Sensor: /base/axi/pcie@120000/rp1/i2c@88000/ov5647@36 - Selected s
nsor format: 640x480-SGRBG10_1X10 - Selected CFE format: 640x480-PC1G
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
error: XDG_RUNTIME_DIR is invalid or not set in the environment.
error: XDG_RUNTIME_DIR is invalid or not set in the environment.
Take a photo and save it as 'image.jpg' after 5 seconds.
Camera test finished
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ ls
camera.py   image.jpg    main.py     parameter.py  rpi_ledpixel.py  servo.py        test.py
car.py      infrared.py   message.py  params.json   server.py        spi_ledpixel.py  ultrasonic.py
command.py  led.py        motor.py    __pycache__   server_ui.py     tcp_server.py
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

Part of code is as below:

```
1   def test_Camera():
2       import time
3       from camera import Camera          # Import the Camera class from the camera module
4       print("test camera")               # Print a test message
5       camera = Camera()                  # Initialize the Camera instance
        camera.start_image()               # Start the camera
6       print("Take a photo and save it as 'image.jpg' after 5 seconds.")
7       time.sleep(5)
8       camera.save_image("image.jpg")  # Capture an image and save it as test.jpg
9       camera.close()                     # Close the camera
10      print("Camera test finished")      # Print a finish message
```

Chapter 4 Ultrasonic Obstacle Avoidance Car

If you have any concerns, please feel free to contact us via support@freenove.com

Description

The obstacle avoidance function of the vehicle mainly uses HC-SR04 ultrasonic module. The ultrasonic module will detect the distance between obstacles and the robot in real time, and then control the tank robot to move according to different distances.

Run program

If the terminal displays the directory as below, you can **directly** run the Ultrasonic.py.

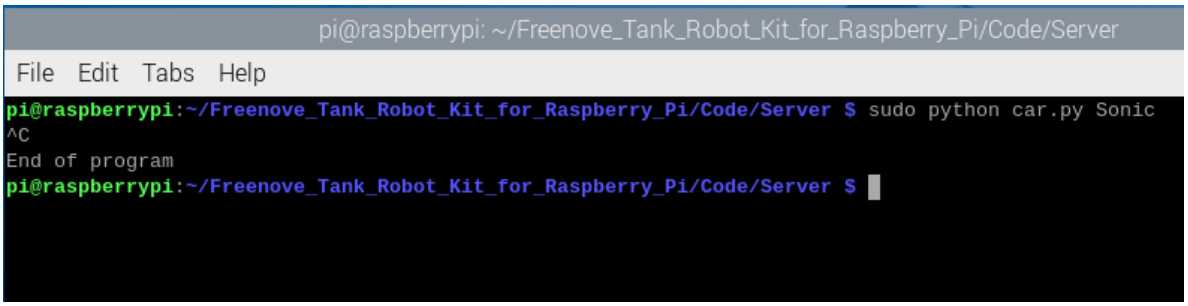
```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Run the ultrasonic program.

```
sudo python car.py Sonic
```



You can press "Ctrl + C" to end the program.

Part of code is as below:

```
1     def mode_ultrasonic(self):
2         # Get distance from ultrasonic sensor
3         distance = self.sonic.get_distance()
4         # print("Ultrasonic distance is " + str(distance) + "CM")
5
6         # Check if distance is valid
7         if distance != 0:
8             # If distance is less than 45 cm, move backward and turn left
9             if distance < 45:
10                self.motor.setMotorModel(-1500, -1500)
11                time.sleep(0.4)
12                self.motor.setMotorModel(-1500, 1500)
13                time.sleep(0.2)
14            # Otherwise, move forward
15            else:
```

```
16                  self.motor.setMotorModel(1500, 1500)
17          # Sleep for a short duration
18          time.sleep(0.2)
```

Result analysis

When the tank robot detects that the distance between the obstacle and the car is less than 45cm, it will successively retreat for a period of time and then turn left for a period of time. When the robot detects that the distance between the obstacle and the robot is greater than or equal to 45cm, it will move forward.

In addition, you need to pay attention to the following:

In obstacle avoidance mode, only a single ultrasonic module is used for recognition, so in this mode, you need to observe the movement of the car. When the car is misidentified, please stop the car as soon as possible or pick up the car and place it in an open space. Otherwise, the car's manipulator servo could be damaged due to a collision, and may even burn your Raspberry PI expansion board or your raspberry PI.

Chapter 5 Infrared tracking automatic wrecker

If you have any concerns, please feel free to contact us via support@freenove.com

Description

The circuit tracking function of the tank robot mainly uses infrared module. When the sensor detects a black line, the corresponding LED lights up to control the tank robot's movement according to the values of the three sensors. The tank robot detects whether there are obstacles on the road through the ultrasonic sensor while finding the line. When there are obstacles, the tank robot starts the obstacle clearing function. After the obstacle is cleared, the tank robot continues to find the line.

Run program

If the terminal displays the directory as below, you can **directly** run the program.

```
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

1. If not, execute the cd command:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Run the infrared car program.

```
sudo python car.py Infrared
```

```
pi@raspberrypi: ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
File  Edit  Tabs  Help
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $ sudo python car.py Infrared
^C
End of program
pi@raspberrypi:~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server $
```

If your Raspberry pie is not Pi5 and your PCB version is V1.0, run the following command before running the code.

```
sudo pigpiod
```

If you want to end the pigpio process, run the following command.

```
sudo killall pigpiod
```

You can press "Ctrl + C" to end the program.

Before running the project, you should use black tape to make a suitable closed track, place the car on the track after the completion of the production, and then run the relevant code. After 1.5S, the car will start the function of automatically removing obstacles.

When the car is in the task, you need to observe the movement of the car, when the car runs out of the designated track, should stop the car or put the car on the original designated track.

The code is as below:

```
1  def mode_infrared(self):
2          # Get distance from ultrasonic sensor
3          distance = self.sonic.get_distance()
4          # Read all infrared sensors
```

```
        infrared_value = self.infrared.read_all_infrared()
        # print("distance:", distance, "infrared:", infrared_value)

        # Control motor based on infrared sensor values
        if infrared_value == 2:
            self.motor.setMotorModel(1200, 1200)     # Move forward
        elif infrared_value == 4:
            self.motor.setMotorModel(-1500, 2500)    # Turn left
        elif infrared_value == 6:
            self.motor.setMotorModel(-2000, 4000)    # Turn left more sharply
        elif infrared_value == 1:
            self.motor.setMotorModel(2500, -1500)    # Turn right
        elif infrared_value == 3:
            self.motor.setMotorModel(4000, -2000)    # Turn right more sharply
        elif infrared_value == 7:
            self.motor.setMotorModel(0, 0)           # Stop

        # If distance is between 5.0 and 12.0 cm, perform clamp operations
        if distance > 5.0 and distance <= 12.0:
            self.motor.setMotorModel(0, 0)           # Stop motor
            self.set_mode_clamp(1)                   # Set clamp mode to 1 (up)
            while self.get_mode_clamp() == 1 and self.infrared_run_stop == False:
                self.mode_clamp()                    # Perform clamp up operation
            if self.infrared_run_stop == True:
                self.motor.setMotorModel(0, 0)       # Stop motor if infrared run stop is True
                return
            self.motor.setMotorModel(-1500, 1500)    # Turn left
            time.sleep(1.5)
            self.motor.setMotorModel(0, 0)           # Stop motor
            self.set_mode_clamp(2)                   # Set clamp mode to 2 (down)
            while self.get_mode_clamp() == 2 and self.infrared_run_stop == False:
                self.mode_clamp()                    # Perform clamp down operation
            if self.infrared_run_stop == True:
                self.motor.setMotorModel(0, 0)       # Stop motor if infrared run stop is True
                return
            self.motor.setMotorModel(1500, -1500)    # Turn right
            time.sleep(1.4)
```

Result analysis

The car has the functions of line inspection and automatic obstacle clearance. The specific realization of the car's line inspection function is as follows:

There are three sensors on the left, center and right. When the black line is detected by the sensor, it will display a high level, or it is low. When the sensor left: high, middle: low, right: low, the car gently turns left. When the sensor is left: high, middle: high, right: low, the car turns left. When the sensor left: low, medium: high, right: low, the car goes straight. When the sensor left: low, medium: low, right: high, the car gently turns

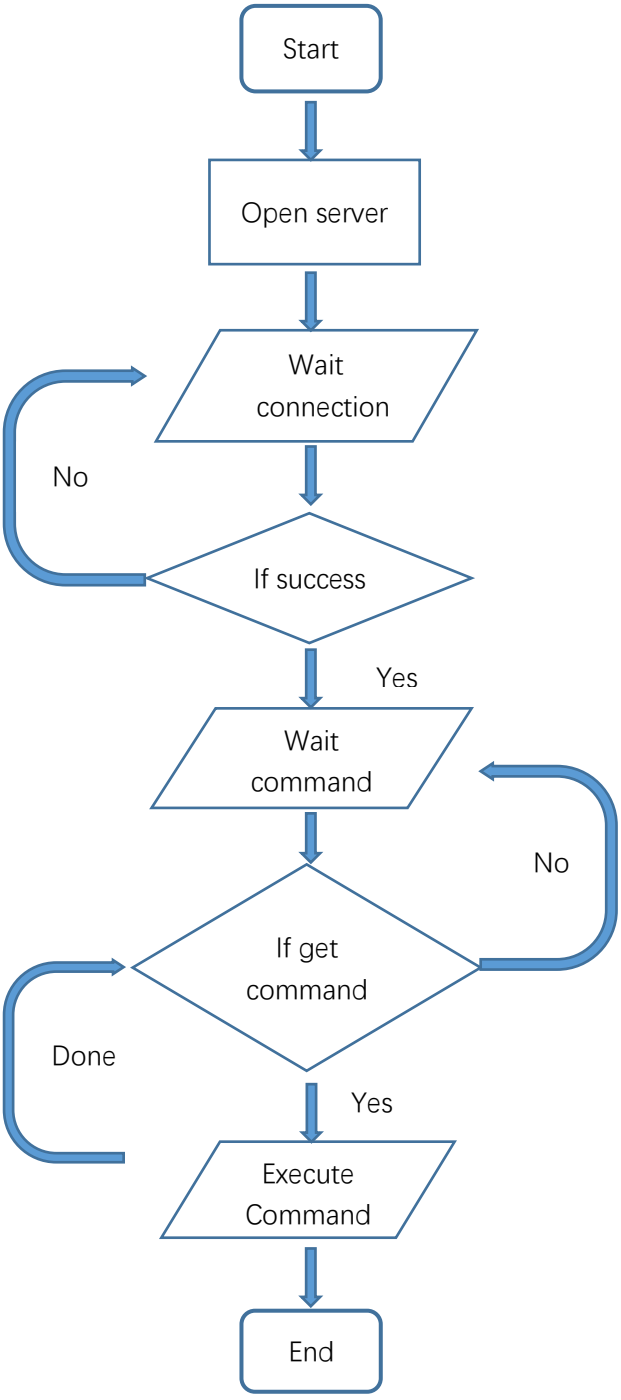
right. When the sensor is left: low, middle: high, right: high, the car turns right.

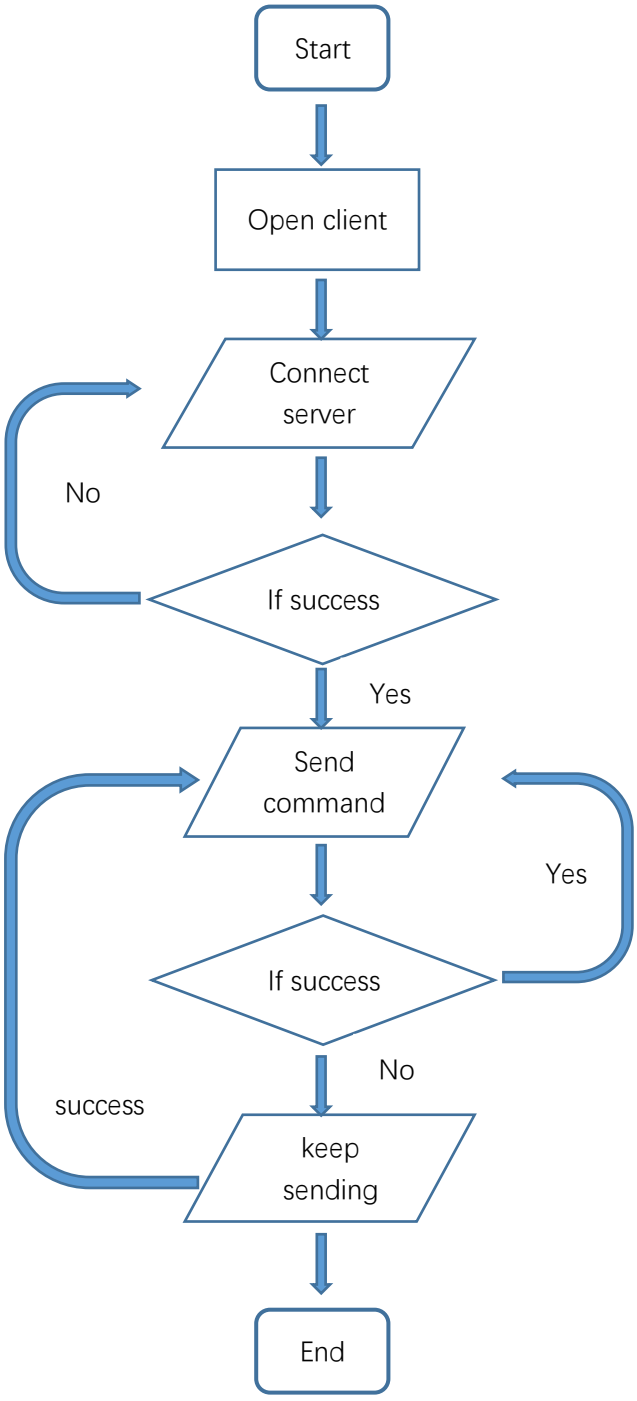
The car's automatic obstacle clearance function is as follows: under the condition of line inspection, the car detects whether there are obstacles in front of the car in real time through the ultrasonic module. When the car detects obstacles in front of the car, the car stops and removes the obstacles through the steering gear. After clearing, the car continues to perform the function of line inspection.

Chapter 6 Smart video car

If you have any concerns, please feel free to contact us via support@freenove.com

The smart video car integrates the previous functions of obstacle avoidance, line tracking and obstacle clearing, video transmission, ball tracking, LED and so on. It consists of a server and a client, so it can be controlled remotely.





Server

Open Server

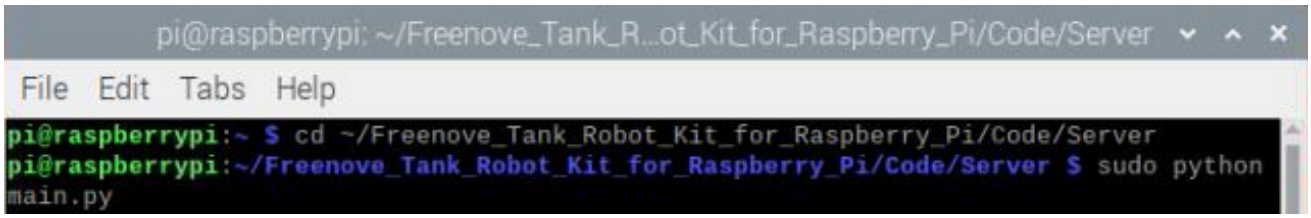
Enter the following command in the terminal.

1. Use cd command to enter directory where main.py is located:

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server
```

2. Run main.py:

```
sudo python main.py
```



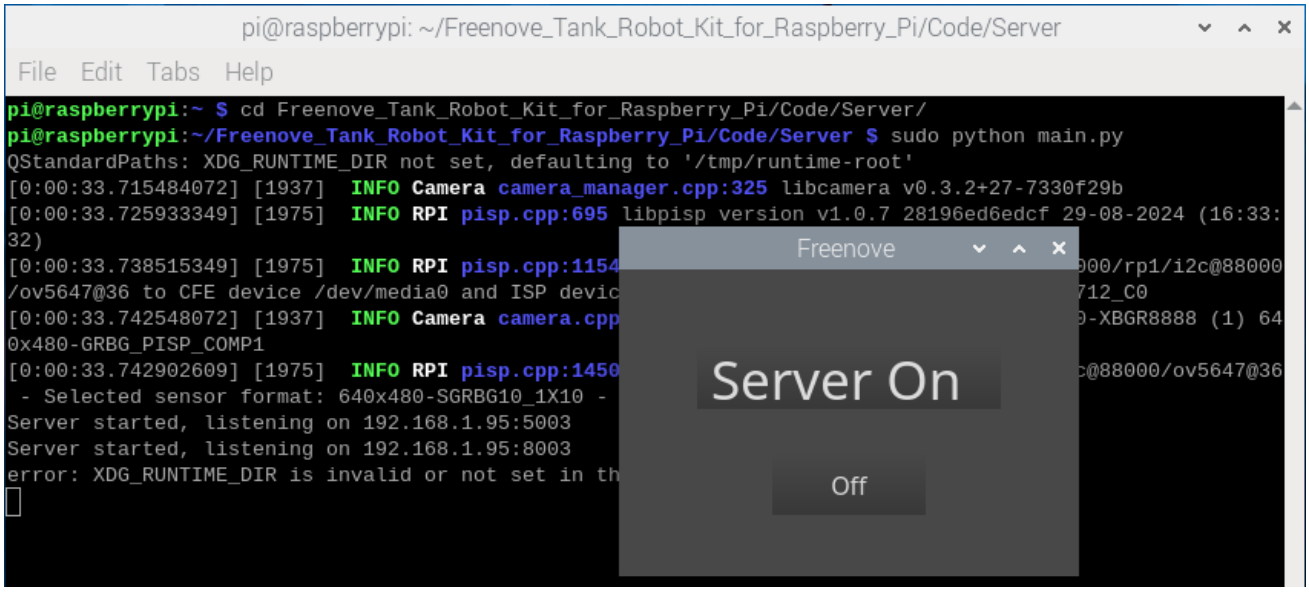
If your Raspberry pie is not Pi5 and your PCB version is V1.0, run the following command before running the code.

```
sudo pigpiod
```

If you want to end the pigpio process, run the following command.

```
sudo killall pigpiod
```

The interface is as below:



The Server is ON when the code is run. To turn it off, you can click the “Off” button.

To exit the server, you can close the server interface or run CTRL+C on the Terminal.

If you are very interested in the implementation of the code, you can view the code: Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server

Sever Auto Start

1 Open the terminal and execute the following two commands respectively to create a “start.sh” file.

```
cd ~
sudo touch start.sh
```

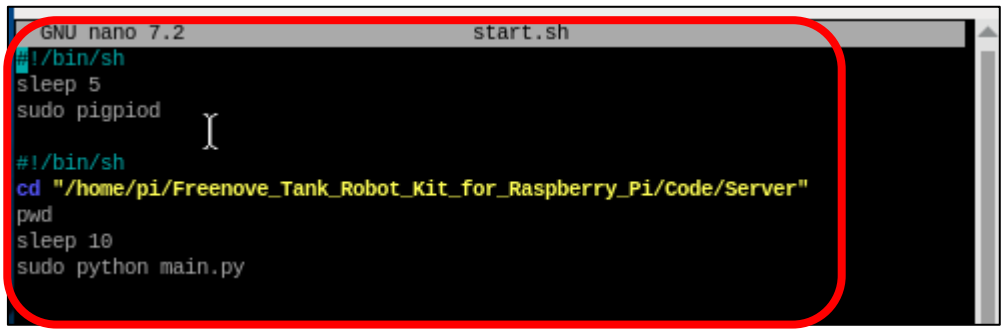
2 Open “start.sh”.

```
sudo nano start.sh
```

3 Add the following contents to “start.sh” file.

```
#!/bin/sh
cd "/home/pi/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server"
pwd
sleep 10
sudo python main.py
```

Press Ctrl + O and then press Enter to save it. Press Ctrl+X to exit.



4 Modify permissions.

```
sudo chmod 777 start.sh
```

5 Enter the following command to create a directory.

```
mkdir ~/.config/autostart/
```

6 create and open “start.desktop” file

```
sudo nano .config/autostart/start.desktop
```

7 Add the following content to “start.desktop” file.

```
[Desktop Entry]
Type=Application
Name=start
NoDisplay=true
Exec=/home/pi/start.sh
```

Press Ctrl + O and then press Enter to save it. Press Ctrl+X to exit.

8 Modify permissions.

```
sudo chmod +x .config/autostart/start.desktop
```

9 Finally enter the following content to reboot Raspberry Pi.

```
sudo reboot
```

Note: To cancel auto start, please delete the files “start.sh” and “start.desktop” created above.

Client

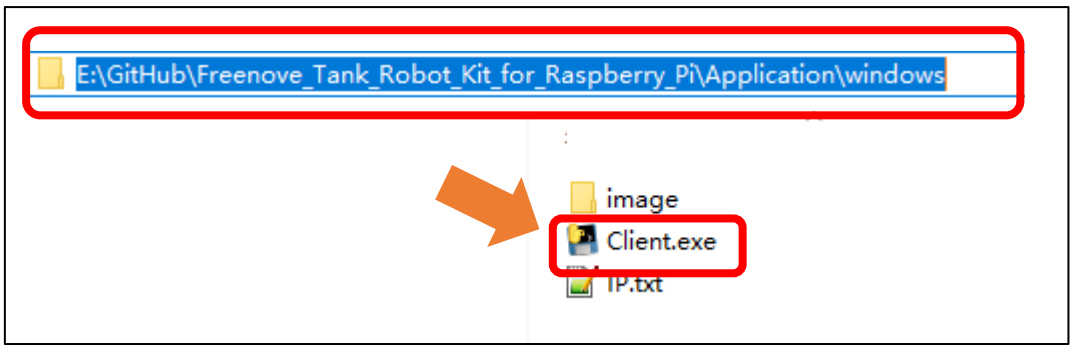
The client connects to the server through TCP, which receives the video stream and other commands from the server. And it also sends commands to the server to control the car.
Clients can run on different systems such as windows, macOS, and Linux. You need to install related software and libraries.

Run client on windows system

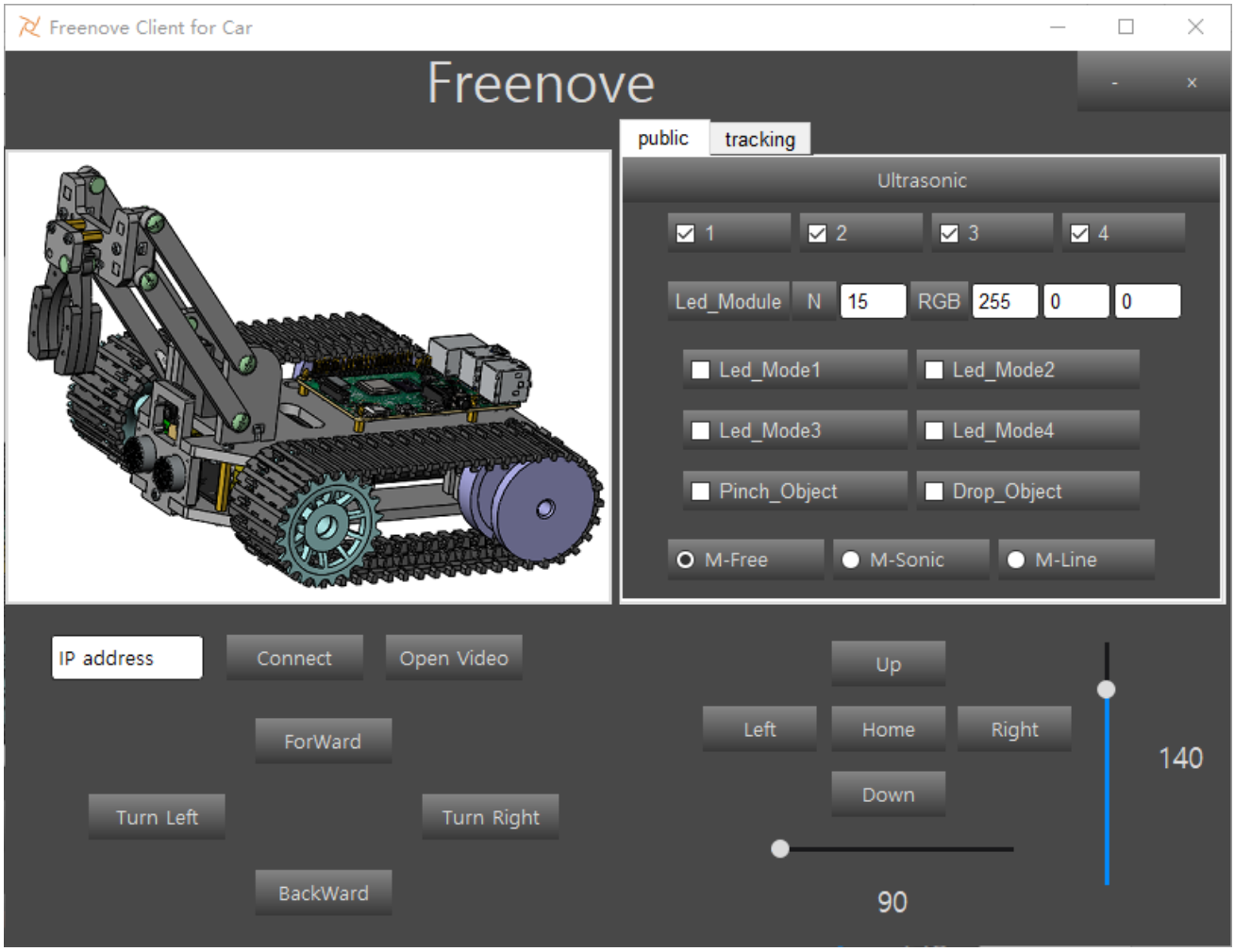
There are two ways to run Client on Windows.

Option 1 Running executable file directly

Find the "Client.exe" file in the specified directory, double click it and the Client is opened.



The client interface is shown as below:

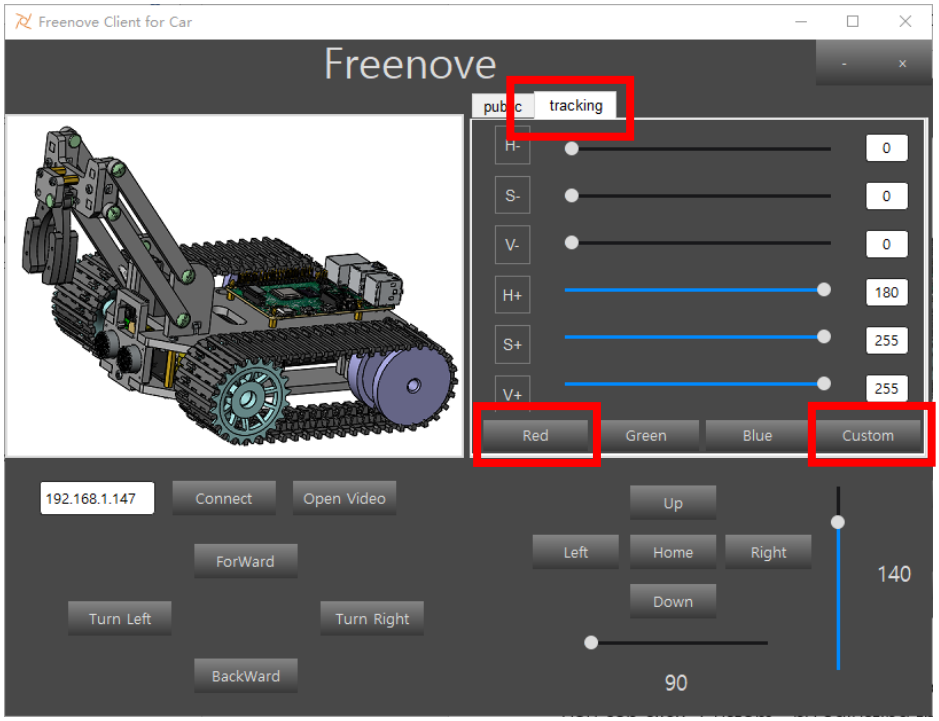


After the client opens successfully, you need open the Raspberry Pi and **open server first**, then enter the IP address of the Raspberry Pi in the white IP edit box, and then click "Connect" to connect smart car to Raspberry Pi. After the connection is successful, you can click on the controls on the interface to operate the car.

Note: when Raspberry Pi is shut down, server will be closed. You need open server again the next time.

If pressing forward but the car moves backward, please refer to page 66 to modify the code.

The control interface of tank robot tracking ball is as follows:



Choose the color of the ball in your hand, take the Red ball as an example, when you click the button "red", into the tank robot tracking red ball control program, when your environment is poor recognition of red, you can click "Custom", by adjusting the HSV parameters to adjust the accuracy of recognition,
This recognition program is for color recognition, can track red objects not limited to red ball.

When the car motion control is abnormal, you can click the color button again to make the car stop moving.

After the color threshold is displayed, the recognition effect can be observed and the appropriate data can be selected according to the following HSV color range table to improve the recognition accuracy.
The HSV color range is listed below:

	black	gray	white	red		orange	yellow	green	cyan	blue	purple
H-	0	0	0	0	156	11	26	35	78	100	125
H+	180	180	180	10	180	25	34	77	99	124	155
S-	0	0	0	43		43	43	43	43	43	43
S+	255	43	30	255		255	255	255	255	255	255
V-	0	46	221	46		46	46	46	46	46	46
V+	46	220	255	255		255	255	255	255	255	255

Using red as an example, set HSV to the following:
[H-, S-, V-, H+, S+, V+] [0, 118, 31, 6, 255, 255]
You can adjust the value range to make the tracking better.

Option 2 Install python3 and some related python libraries to run the client

If you want to modify the client, please follow this section.
This section will be completed in your **computer with windows system, not Raspberry Pi.**
There are many relevant software and libraries needed to be installed in Windows system, which takes a long time. At this time, it does not need to run Server or use Raspberry Pi. You can shut down Raspberry Pi first. After the installation is completed, you need to open Raspberry Pi and server again.

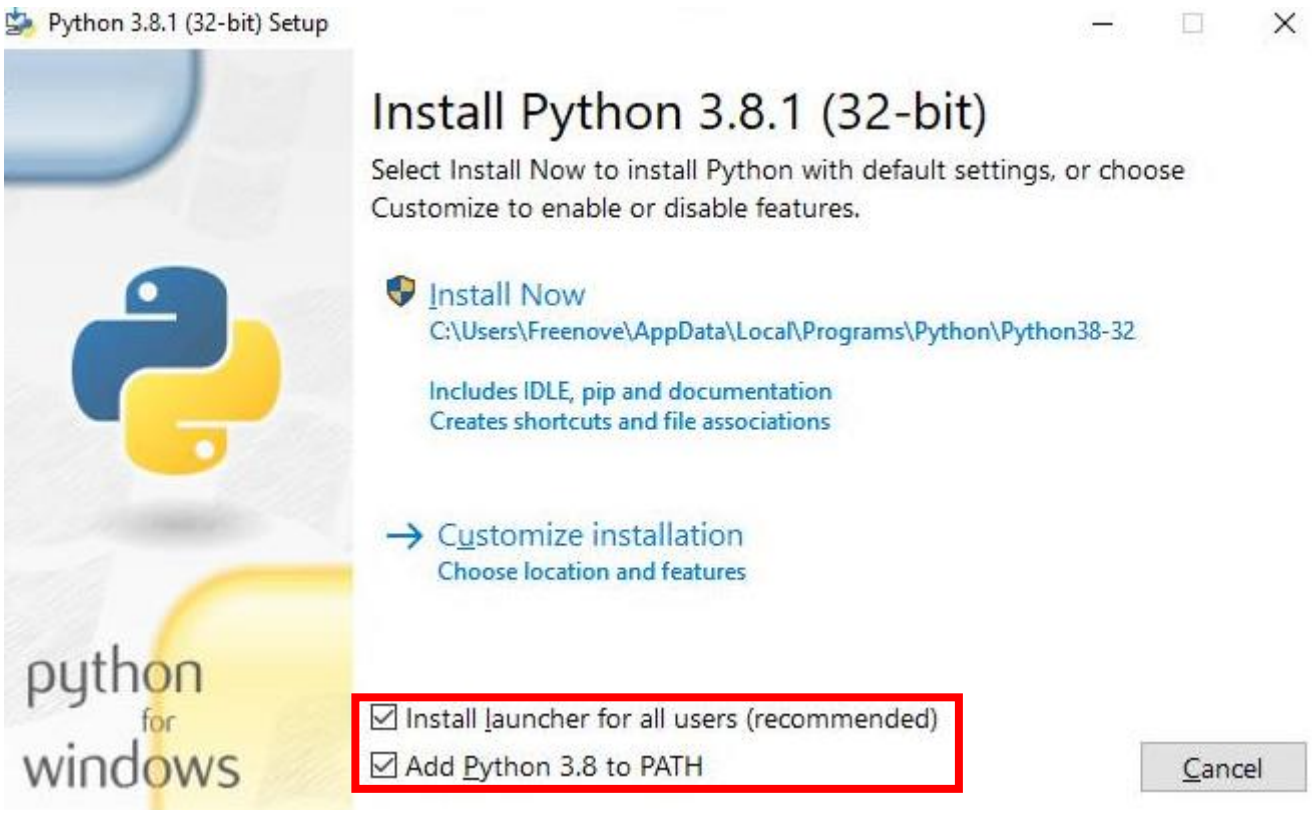
Install python3

Download the installation file:
https://www.python.org/downloads/windows/

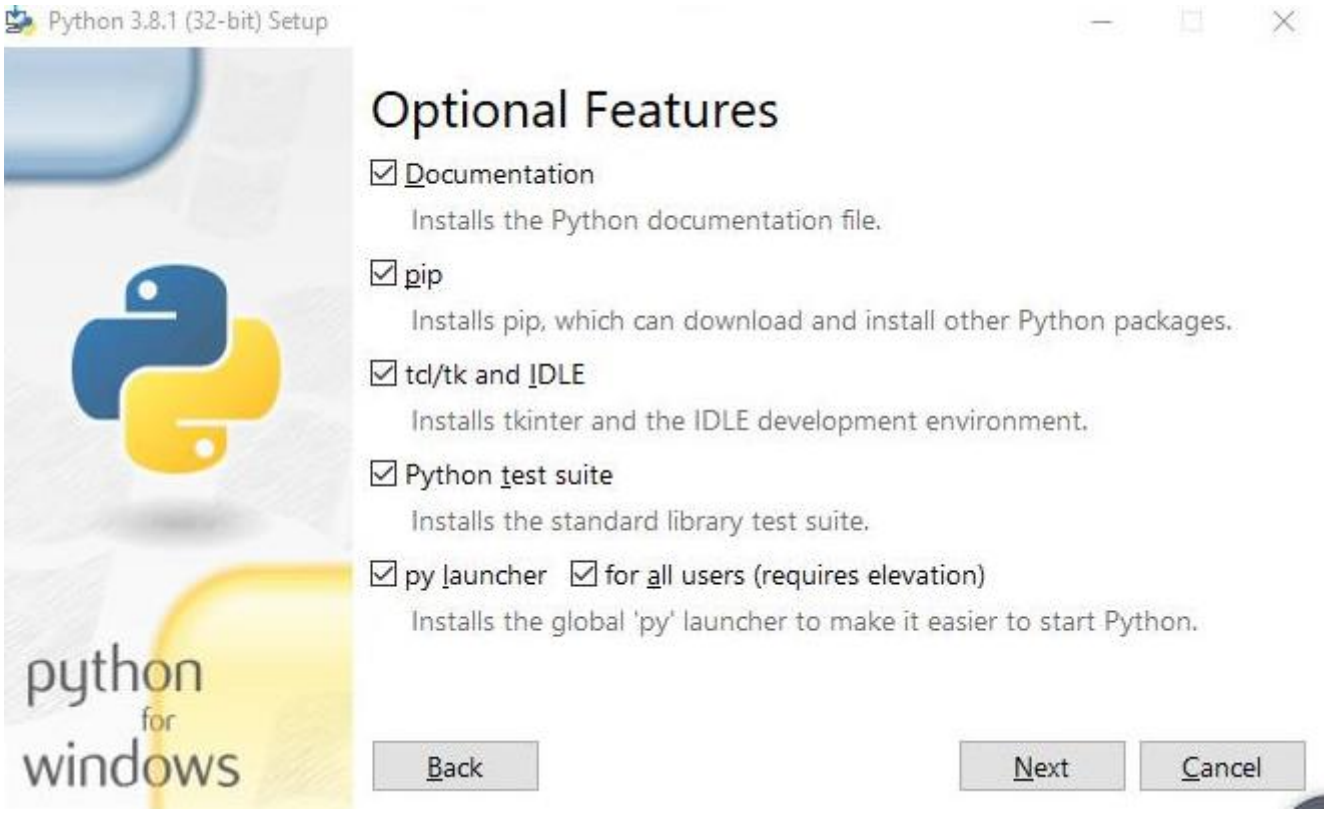
Click Latest Python 3 Release - Python 3.8.1

Version	Operating System	Description
Gzipped source tarball	Source release	
XZ compressed source tarball	Source release	
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later
Windows help file	Windows	
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64
Windows x86 embeddable zip file	Windows	
Windows x86 executable installer	Windows	
Windows x86 web-based installer	Windows	

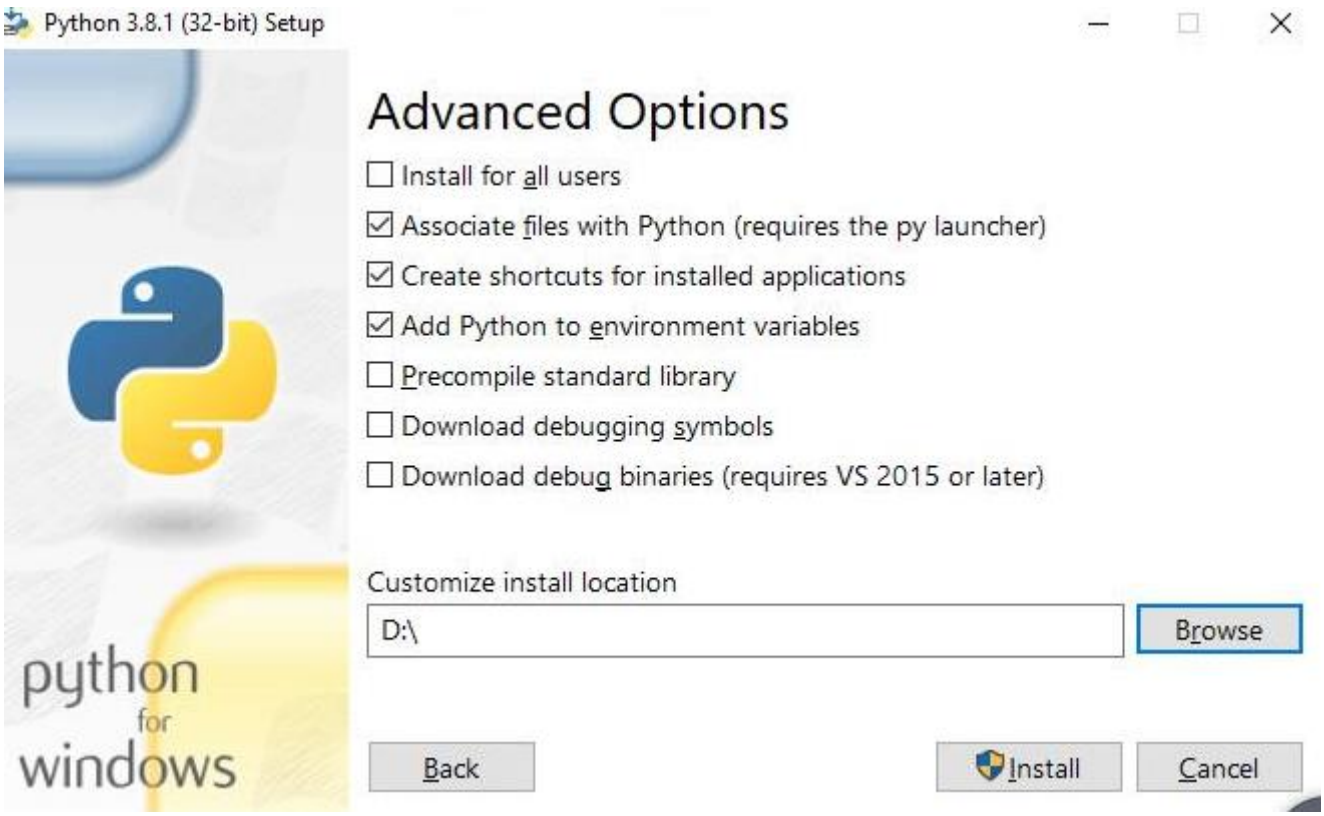
Choose and download Windows x86 executable installer. After downloading successfully, install it.



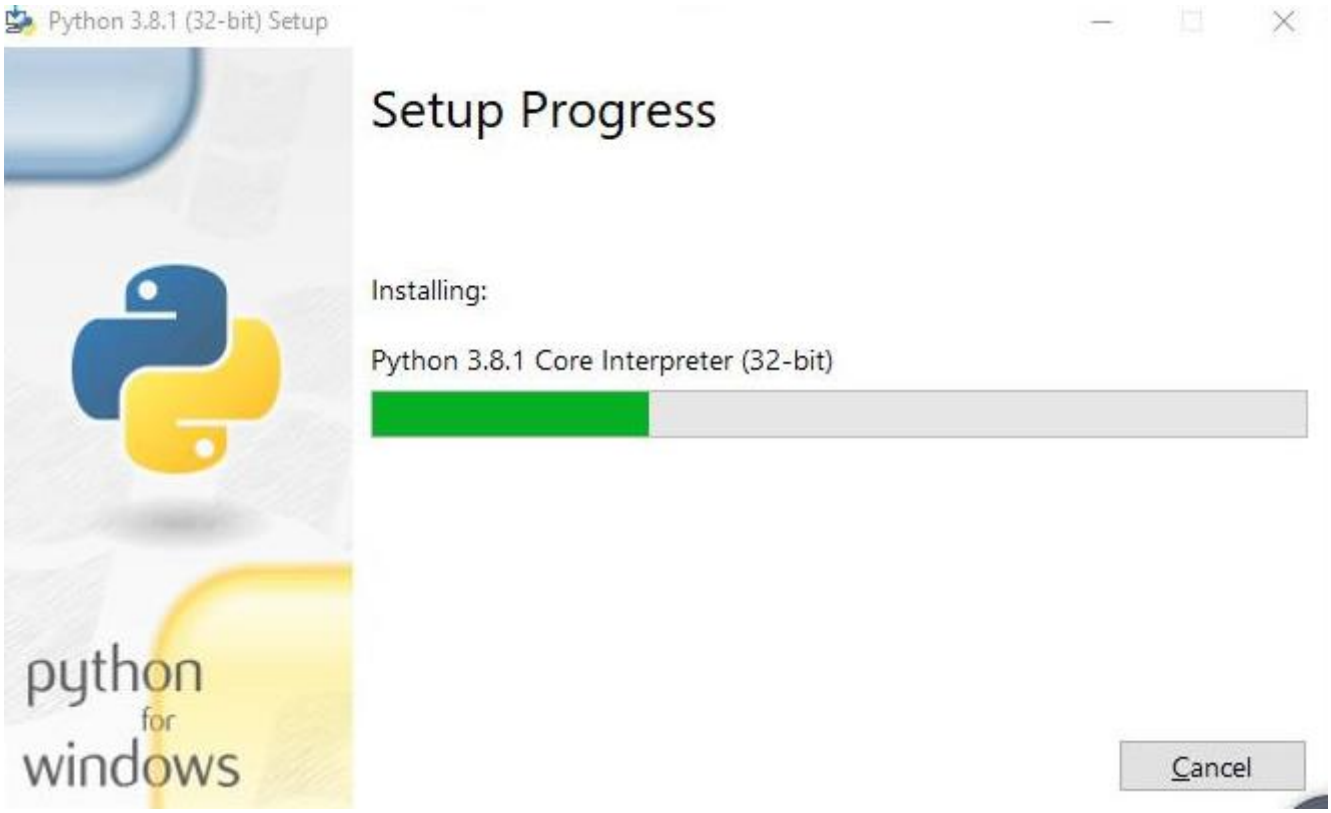
Select "**Add Python 3.8 to PATH**". You can choose other installation features.



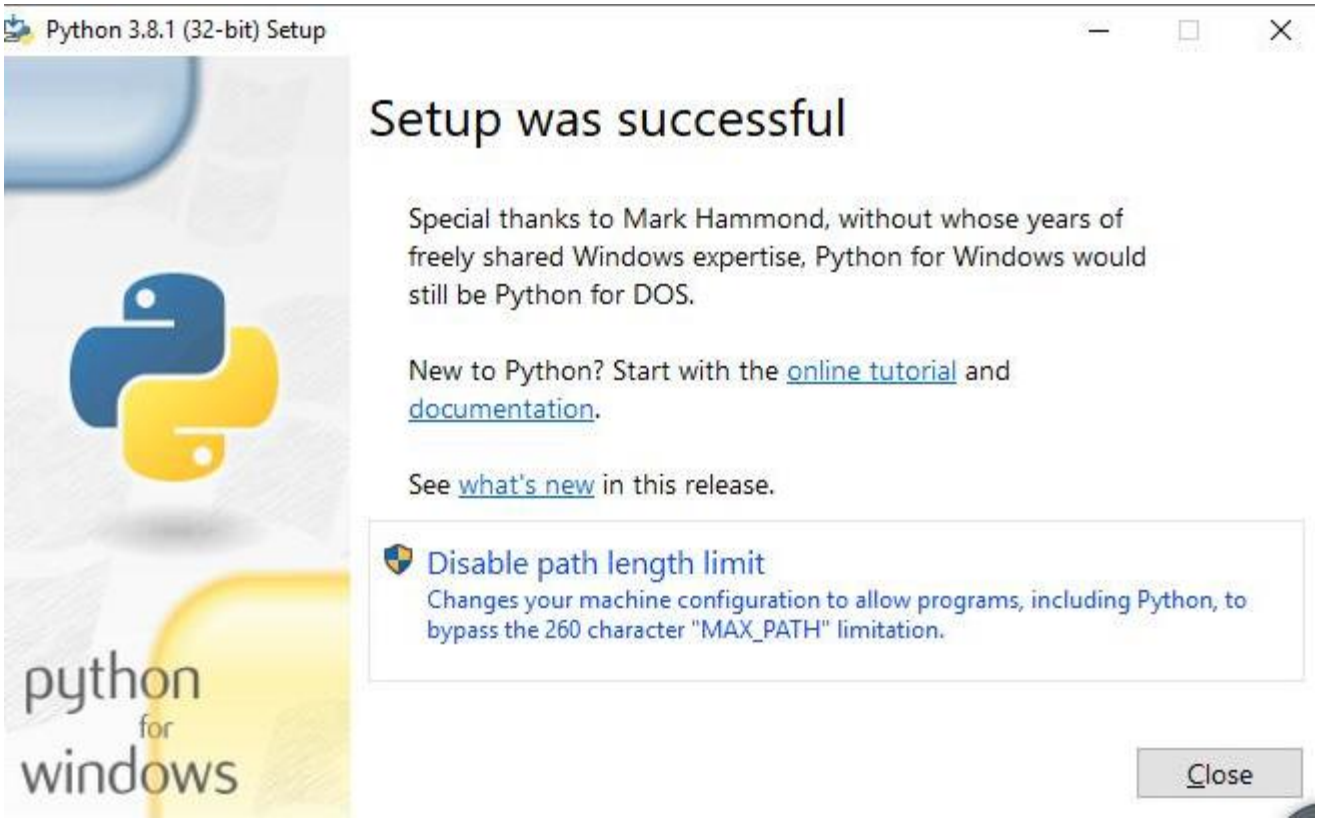
Select all options and click "Next".



Here, my install location is D. You can also choose other location. Then click "Install".



Wait installing.



Now, installation is completed.

Install PyQt5、opencv、numpy and other libraries.

If have not download the zip file, do so via:
https://github.com/Freenove/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/archive/master.zip
Then unzip it and delete "-master" to rename it to "Freenove_Tank_Robot_Kit_for_Raspberry_Pi".
Then put it into D disk for example.

You can also place it into other disks (like E), but the path in the following command should be modified accordingly (replace D: by E:).
Press "win + R" and enter cmd, and click ok. Then enter following commands.

1. Enter D disk. (If you put it into E, it should be E:)

```
D:
```

2. Enter directory where setup_windows.py is located: (If you put it into E, it should be E:)

```
cd D:\Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code
```

3. Run setup_windows.py:

```
python3 setup_windows.py
```

```
C:\Users\Freenove>D:

D:\>cd D:\Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code

D:\Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code>python3 setup_windows.py
```

Or enter the unzipped directory Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code\Client.
If your python3 fails to execute, you can try using python with the following command:

```
python setup_windows.py
```

And double-click **setup_client.py** or open it with python3.
Installation will take some time. Just wait patiently. For successful installation, it will prompt "All libraries installed successfully":

```
Package        Version
-------------- -------------
Click          7.0
numpy          1.18.1
opencv-python  4.1.2.30
Pillow         7.0.0
pip            19.2.3
PyQt5          5.13.2
PyQt5-sip      12.7.0
pyqt5-tools    5.13.2.1.6rc1
python-dotenv  0.10.3
setuptools     41.2.0
```

If not all installations are successful, it will prompt "Some libraries have not been installed yet. Please run ' Python3 setup_windows.py ' again", then you need to execute the Python3 setup_windows.py command again. Most of the installation failures are caused by poor networks. You can check your network before installing.

Open client

Press “win + R” and enter cmd, and click ok. Then enter following commands.

1. Enter D disk. If you put it into E, it should be E:

```
D:
```

2. Enter directory where Main.py is located:

```
cd D:\Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code\Client
```

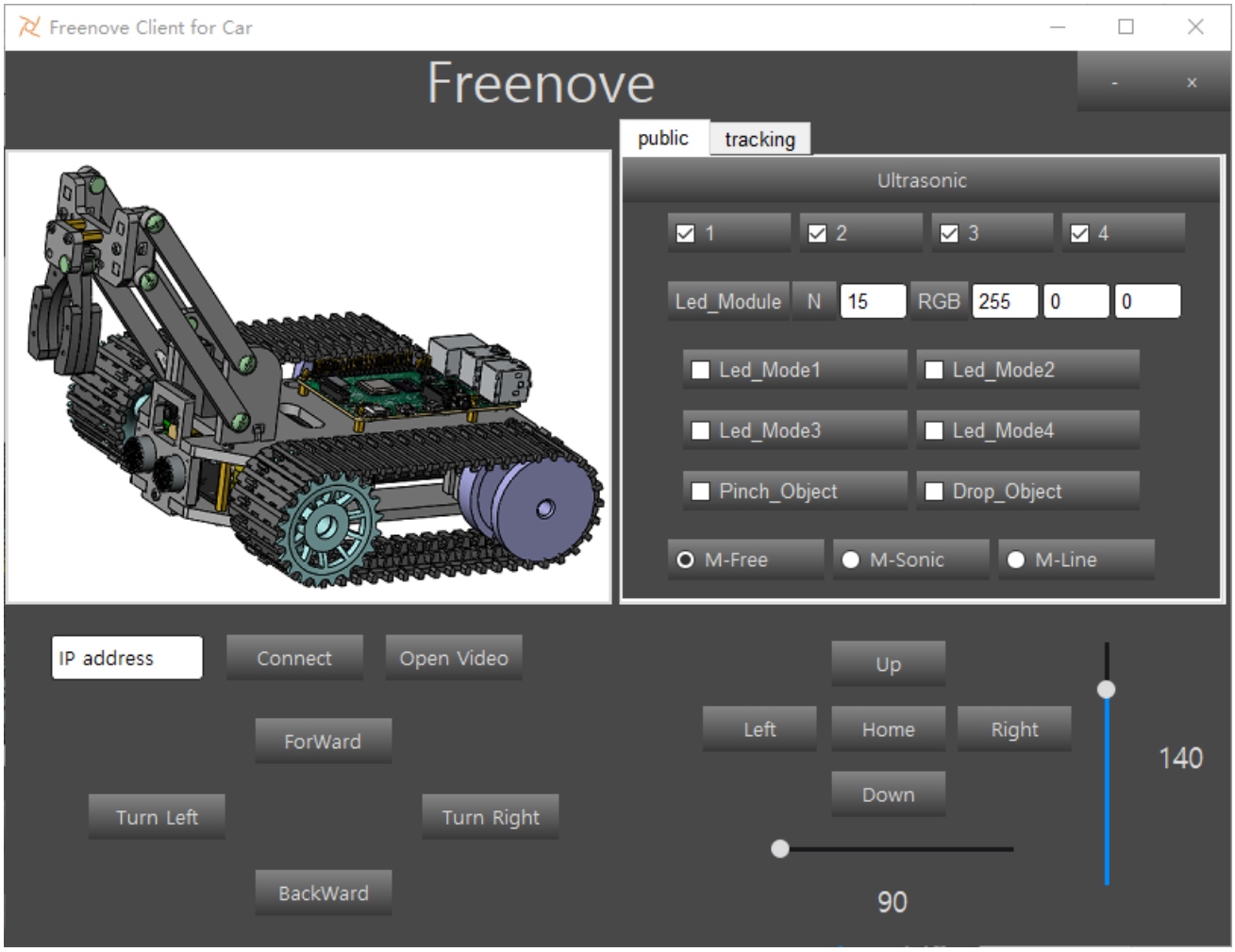
3. Run Main.py:

```
python Main.py
```

```
C:\Users\Freenove>D:

D:\>cd D:\Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code\Client

D:\Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code\Client>python Main.py
```

Or enter the unzipped directory and enter following directory:
Freenove_Tank_Robot_Kit_for_Raspberry_Pi\Code\Client. And double-click **Main.py** or open it with python to open the client.

The client interface is shown as below:



After the client opens successfully, you need open the Raspberry Pi and **open server first**, then enter the IP address of the Raspberry Pi in the white IP edit box, and then click "Connect" to connect smart car to Raspberry Pi. After the connection is successful, you can click on the controls on the interface to operate the car.

Note: when Raspberry Pi is shut down, server will be closed. You need open server again the next time.

If pressing forward but the car moves backward, please refer to page 63 to modify the code.

Control

And you can also control the car with following blue keys.

The car has three work modes:

Mode	Function
M-Free (Mode1)	Free control mode
M-Sonic (Mode2)	Ultrasonic obstacle avoidance mode
M-Line (Mode3)	Infrared tracking automatically clears obstacles mode

The following is the corresponding operation of the buttons and keys.

Button on Client	Key	Action
ForWard	W	Move
BackWard	S	Back off
Turn Left	A	Turn left
Turn Right	D	Turn right
Left	left arrow	Turn camera left
Right	right arrow	Turn camera right
Up	up arrow	Turn camera up
Down	down arrow	Turn camera down
Home	Home	Turn camera back Home
Connect/ Disconnect	C	On/off Connection
Open Video/ Close Video	V	On/off Video
Led_Mode 1,2,3,4	L	Switch Led Mode
The car work modes	Q	Switch the car work modes
Pinch_Object	O	Pinch Object
Drop_Object	P	Drop Object

The function of SliderBar is below:

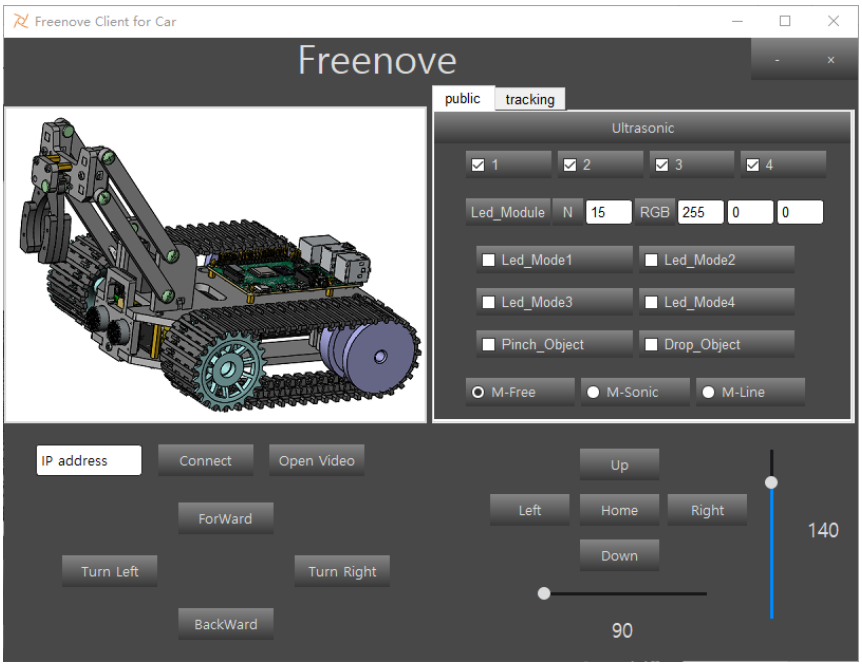
SliderBar	Function
Servo 1,2,	SliderBar Servo 1, 2 are used to slightly adjust the angle. you can slightly tune it via the SliderBar.

Other control information:

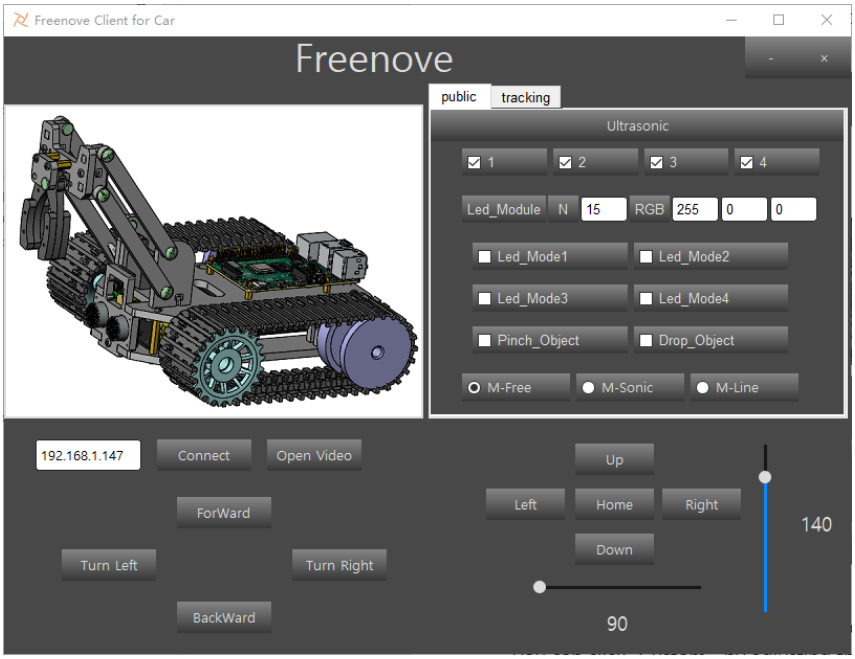
Control	Function
IP address Edit box	Enter IP address of Raspberry Pi
R,G,B Edit box	Control the color of LED selected.
Button “Ultrasonic“	Show the distance from obstacle.

When you enter the IP address for the first time, the program saves the IP address. When you close the program and open the program again, the program automatically fills in the last IP address. The IP address is saved in the “IP.txt” file.

When you run the program for the first time, the contents of the "IP.txt" file are: "IP address", then the program page is opened as follows:



After you enter the address, for example, 192.168.1.147, close the program and open the program page again as follows:



Run client on macOS system

Here take MacOS 10.13 as an example. To run the client on MacOS, you need to install some software and libraries. At this time, it does not need to run the server or use the Raspberry Pi. So you can turn off the Raspberry Pi first. After the installation is complete, turn on the Raspberry Pi and run the server. MacOS 10.13 comes with python2, but no python3. However, the programs in this project need run under python3, so you need to install it first.

Install python3

Download installation package, link: https://www.python.org/downloads/

Python 3.8.1	Dec. 18, 2019	Download
Python 3.7.6	Dec. 18, 2019	Download

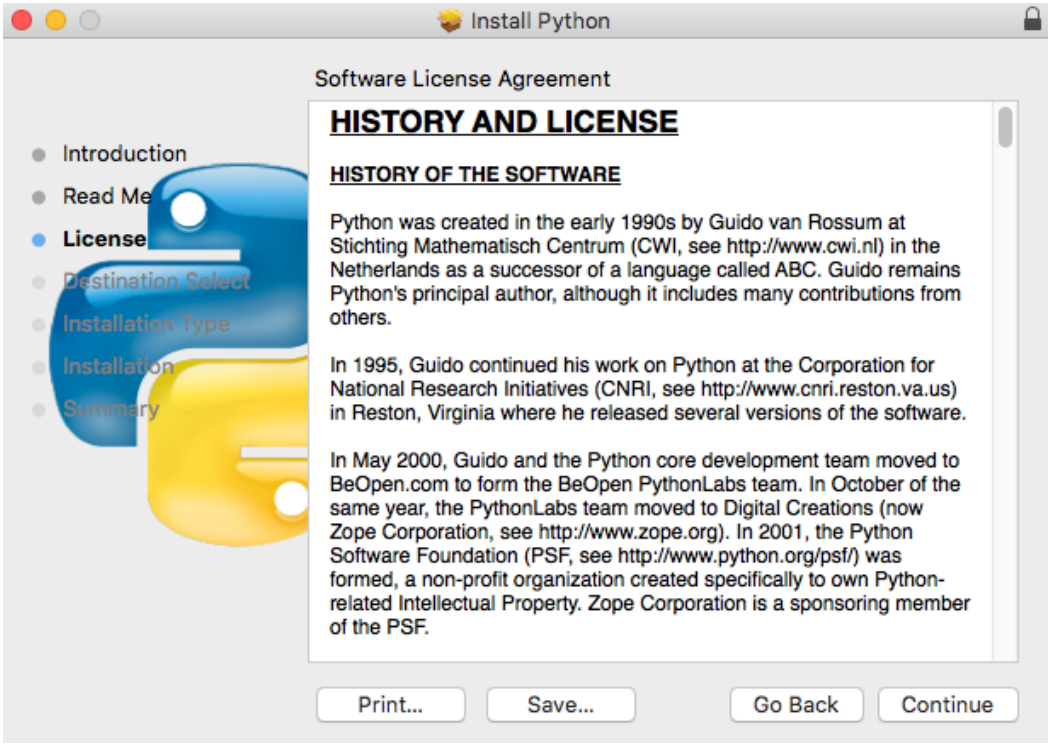
If your macOS is 11. Like 11.0, please install python 3.9.

If your macOS is NOT 11, like 10.15, please install python 3.8. If you have installed python 3.9. You need uninstall it first.

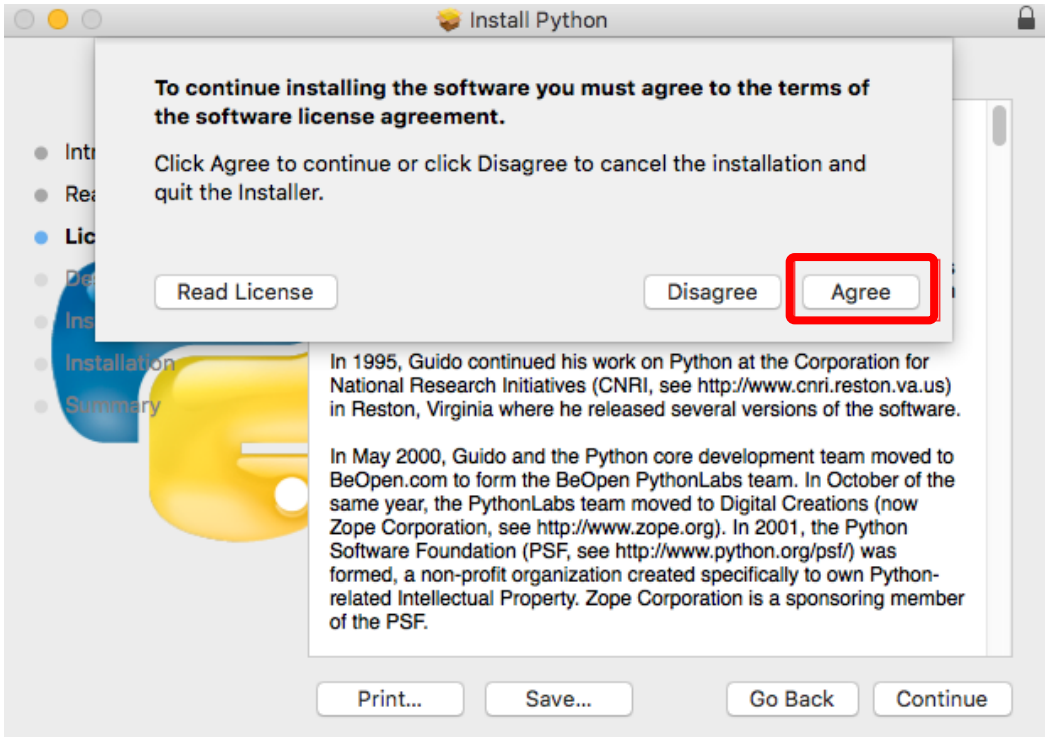
Version	Operating System	Description
Gzipped source tarball	Source release	
XZ compressed source tarball	Source release	
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later
Windows help file	Windows	
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64
Windows x86 embeddable zip file	Windows	
Windows x86 executable installer	Windows	
Windows x86 web-based installer	Windows	

At bottom of the page, click macOS 64-bit installer and download installation package.

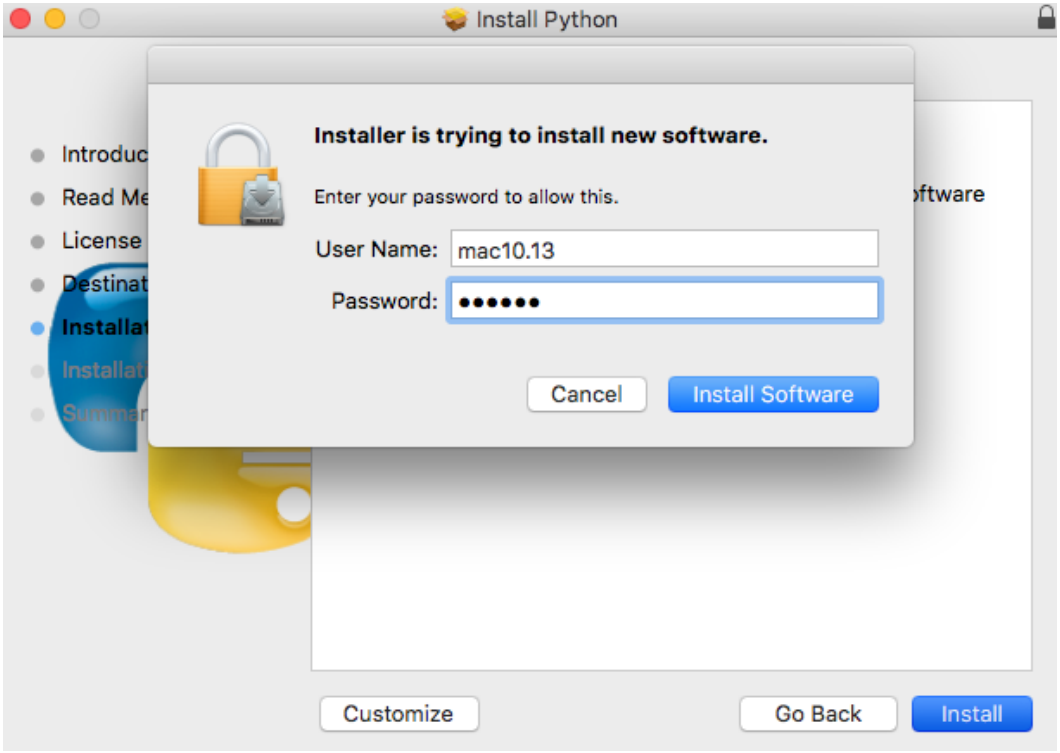
Click Continue.



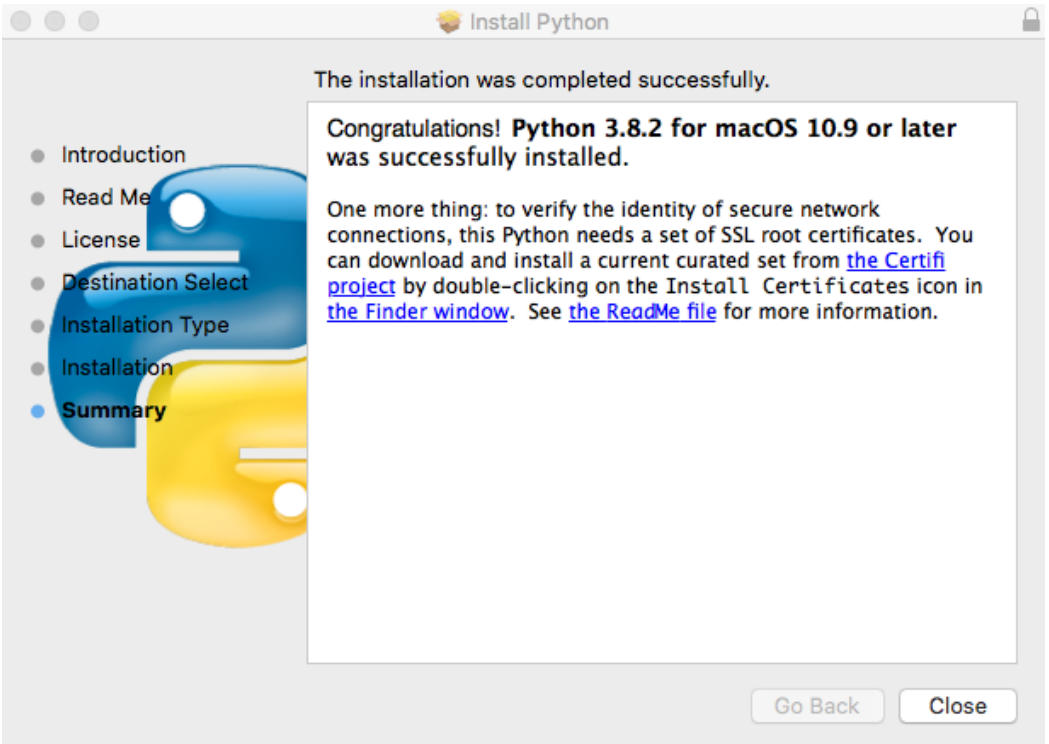
Click Continue



Click Agree.



Click Install. If your computer has a password, enter the password and Install Software.



Now the installation succeeds.

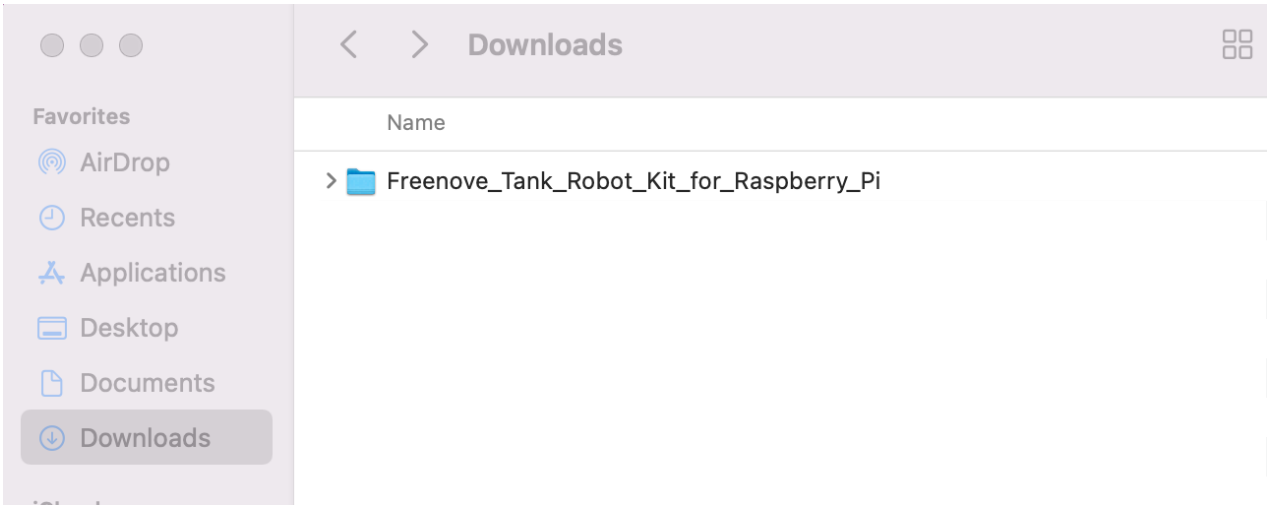
You can find it in Aapplications.

Install PyQt5、opencv、numpy and other libraries

If there is no code for this car in your macOS system device, you can download it via the link below:

https://github.com/Freenove/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/archive/master.zip

After downloaded successfully, you can find it under Downloads.



Open the Terminal.

Type following commands in Terminal.

1.Enter “Downloads“, (Where the Car code is located. If your location for it is different, please enter the

location in your device.)

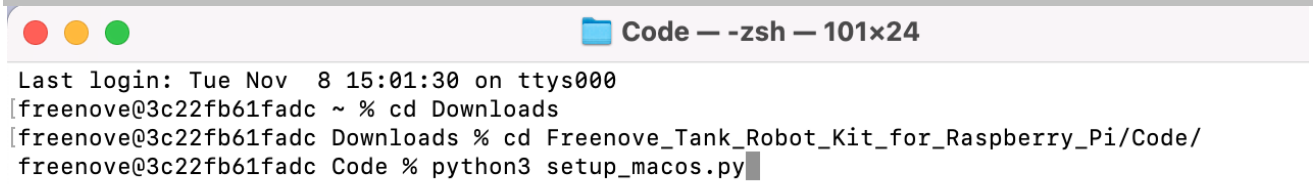
```
cd Downloads
```

2.Enter directory where setup_macos.py is located:

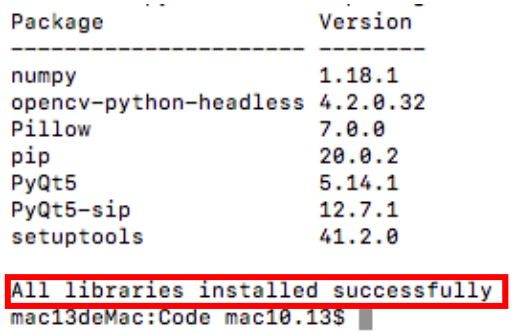
```
cd Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/
```

3.Run setup_macos.py:

```
python3 setup_macos.py
```



Installation will take some time. Just wait patiently. For successful installation, it will prompt "All libraries installed successfully":



If not all installations are successful, it will prompt "Some libraries have not been installed yet. Please run 'python3 setup_macos.py' again", then you need to execute the python3 setup_macos.py command again. Most of the installation failures are caused by poor networks. You can check your network before installing.

If you are using macOS under 11.0, like 10.15. **Just skip to "Open client".**

If you are using macOS 11.0 or later version. Please run commands below:

```
pip3 uninstall PyQt5
pip3 install PyQt5
```

Open client

Following the previous step, after the installation is completed, you are now in the directory where setup_macos.py is located.

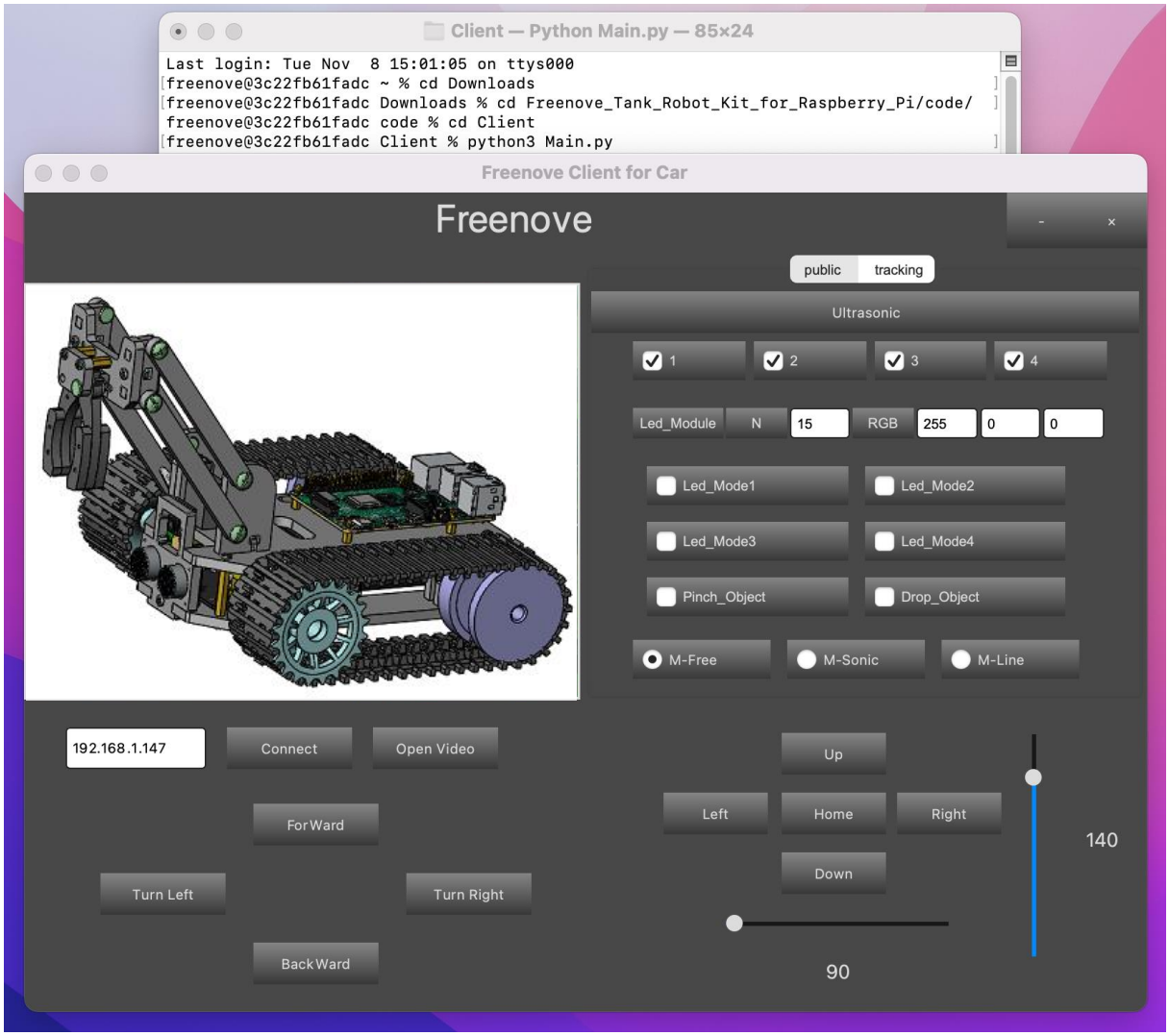
```
Package                 Version
----------------------- --------
numpy                   1.18.1
opencv-python-headless  4.2.0.32
Pillow                  7.0.0
pip                     20.0.2
PyQt5                   5.14.1
PyQt5-sip               12.7.1
setuptools              41.2.0

All libraries installed successfully
mac13deMac:Code mac10.13$
```

1.Type following command to enter Client folder.

```
cd Client/
```

2.Type following command to run Main.py.

```
python3 Main.py
```



The control way of Raspberry Pi macOS System client is same with Windows (Control).

Run client in Raspberry Pi (Linux system)

Install Opencv library

Execute the following commands in the terminal to install Opencv library:

1. Install opencv development environment:

```
sudo apt-get install -y libopencv-dev python3-opencv
```

2. Install some tools:

```
sudo apt-get install -y  python3-pil python3-tk
```

Run client

Enter the following commands at the terminal.

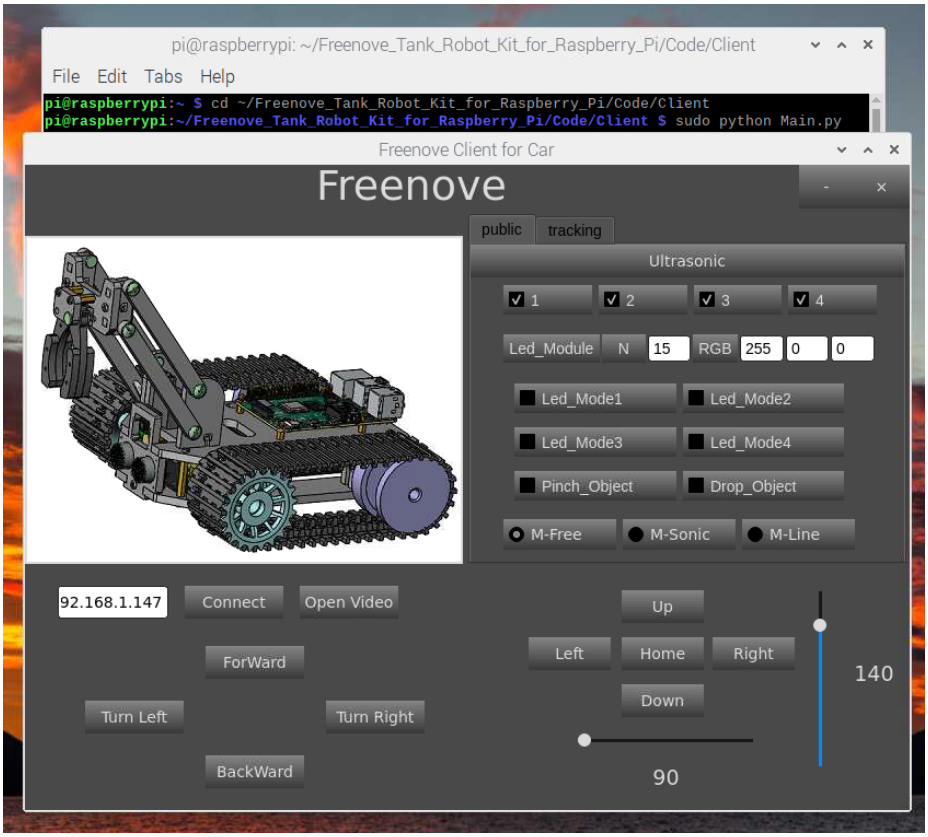
1. Use the cd command to go to the directory where Main.py is located.

```
cd ~/Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Client
```

2. Run Main.py:

```
sudo python Main.py
```

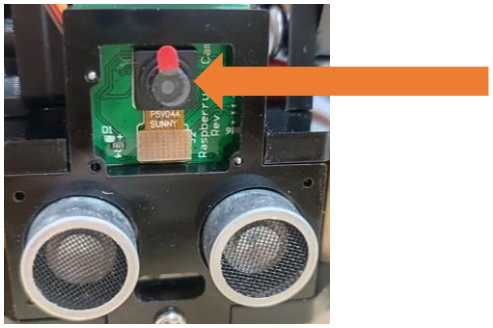
The interface is shown below:



The control mode of client on Linux is the same as that of Windows.

If the image is not clear, please check whether the camera protective film is torn off.

Trouble shooting



If the car works abnormally, it may be caused by following reasons: raspberry pi system is stuck or batteries have no power.

You need check batteries power indicator or recharge batteries.

If the batteries are OK, raspberry pi system is stuck. You need wait some time to check if the client works. Or reopen the server and client.

The latest Raspberry Pi official system is not stable. It occasionally is stuck. The old version is more stable.

If the raspberry pi system is stuck for a long time, you need reboot raspberry pi.

If you have any concerns, please feel free to contact us with pictures:

support@freenove.com

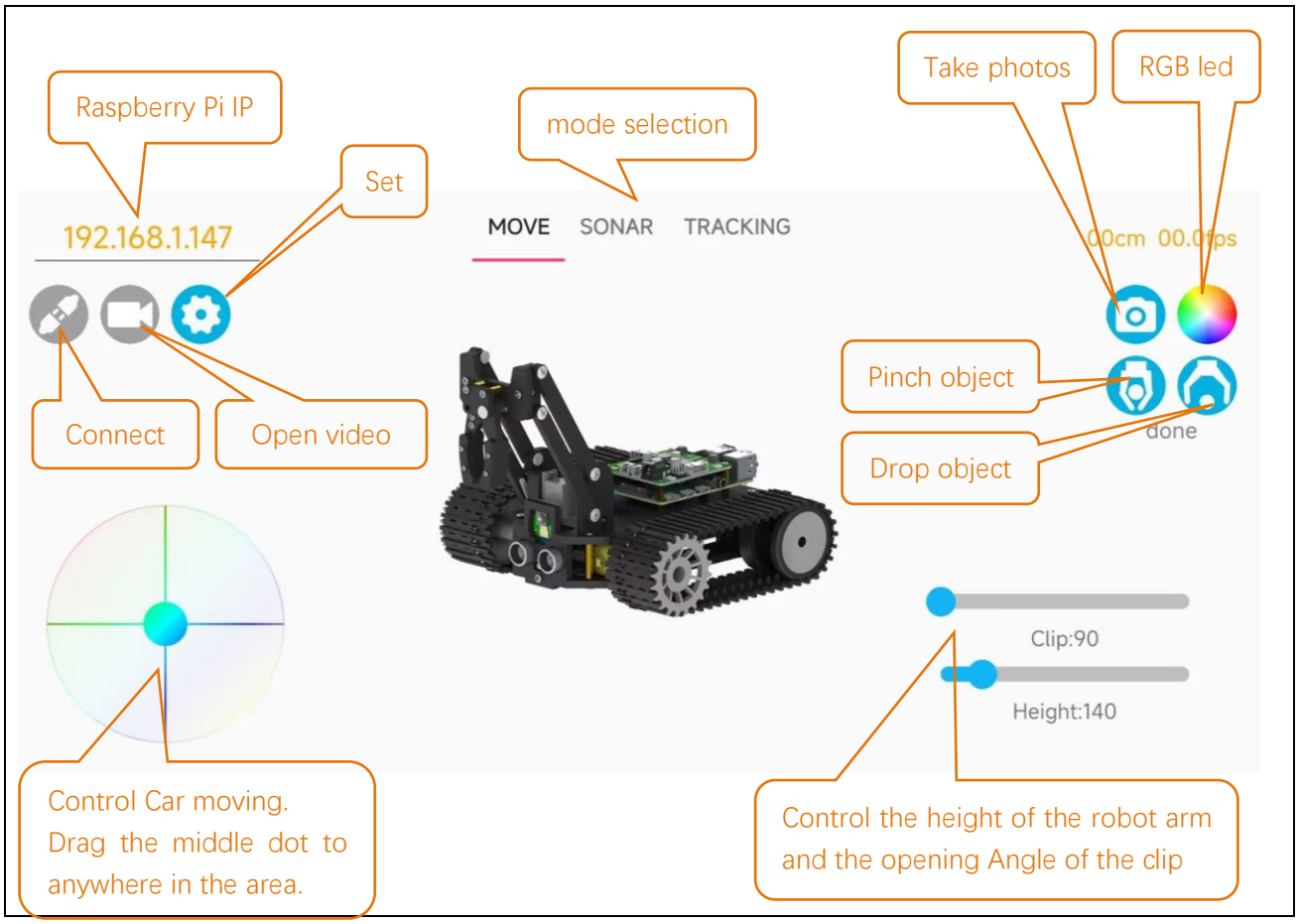
Android and iOS app

You can download and install the Freenove Android app from below:
On Google play:
https://play.google.com/store/apps/details?id=com.freenove.suhayl.Freenove
On GitHub:
https://github.com/Freenove/Freenove_App_for_Android
In this github repository, you can find the App instruction (Tutorial.pdf).

You can download and install the Freenove **iPhone ios app** by searching **freenove** in app store.

Open the app and select the car.

Open the server in Raspberry Pi car first. And enter your Pi IP.



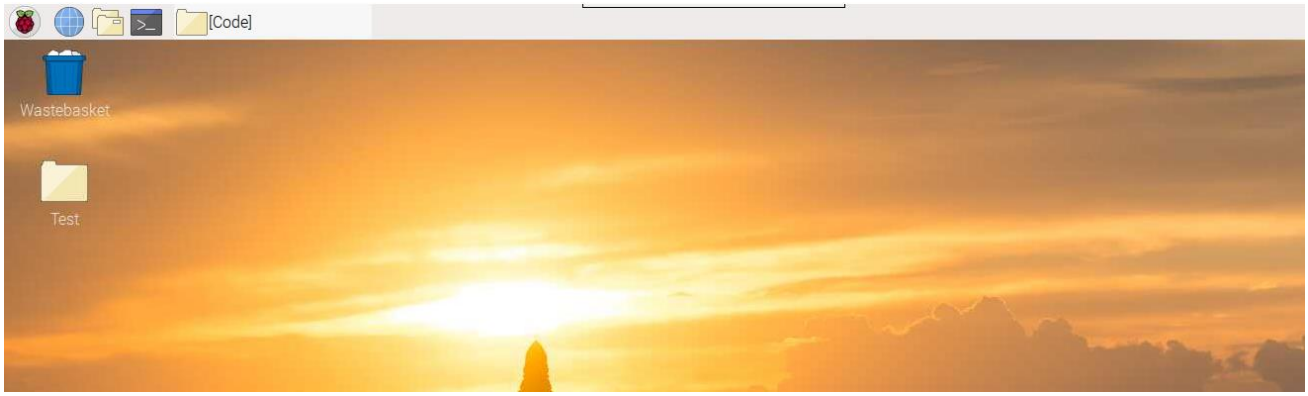
Free innovation

If you have any concerns, please feel free to contact us via support@freenove.com

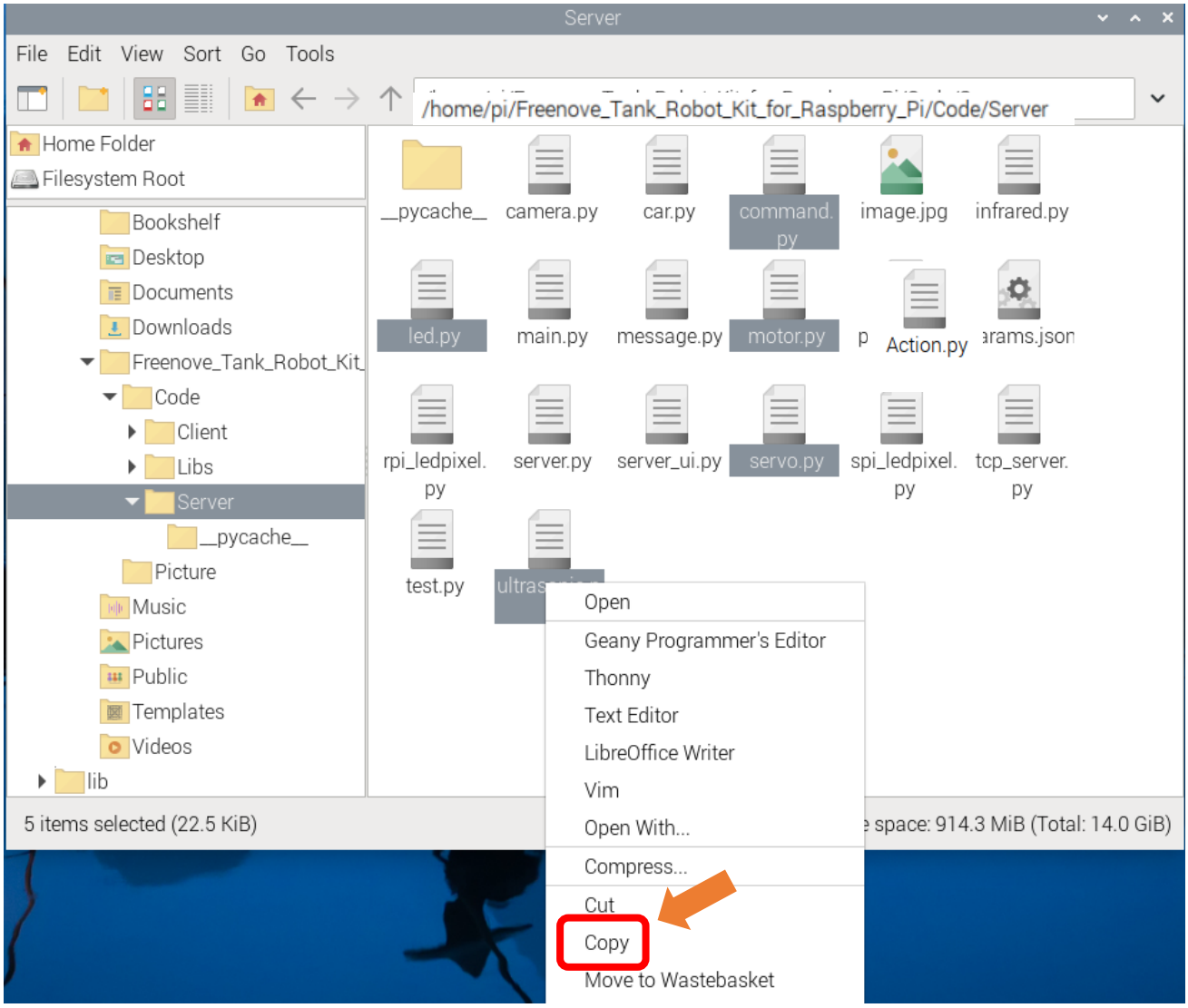
If you want to write your own program to control the car, just follow this section. We will teach you how to program this car.

If you have never learned python before, you can learn some basic knowledge via the link below: https://python.swaroopch.com/basics.html

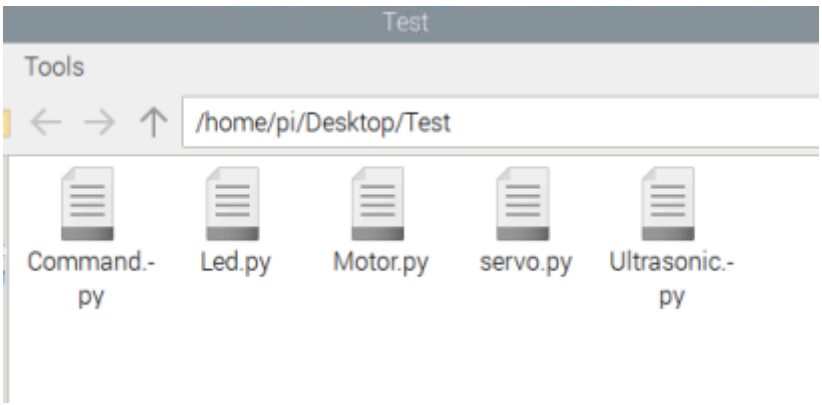
First, turned on S1 and S2. Then open Raspberry Pi, right click and create a new folder on the desktop: Test



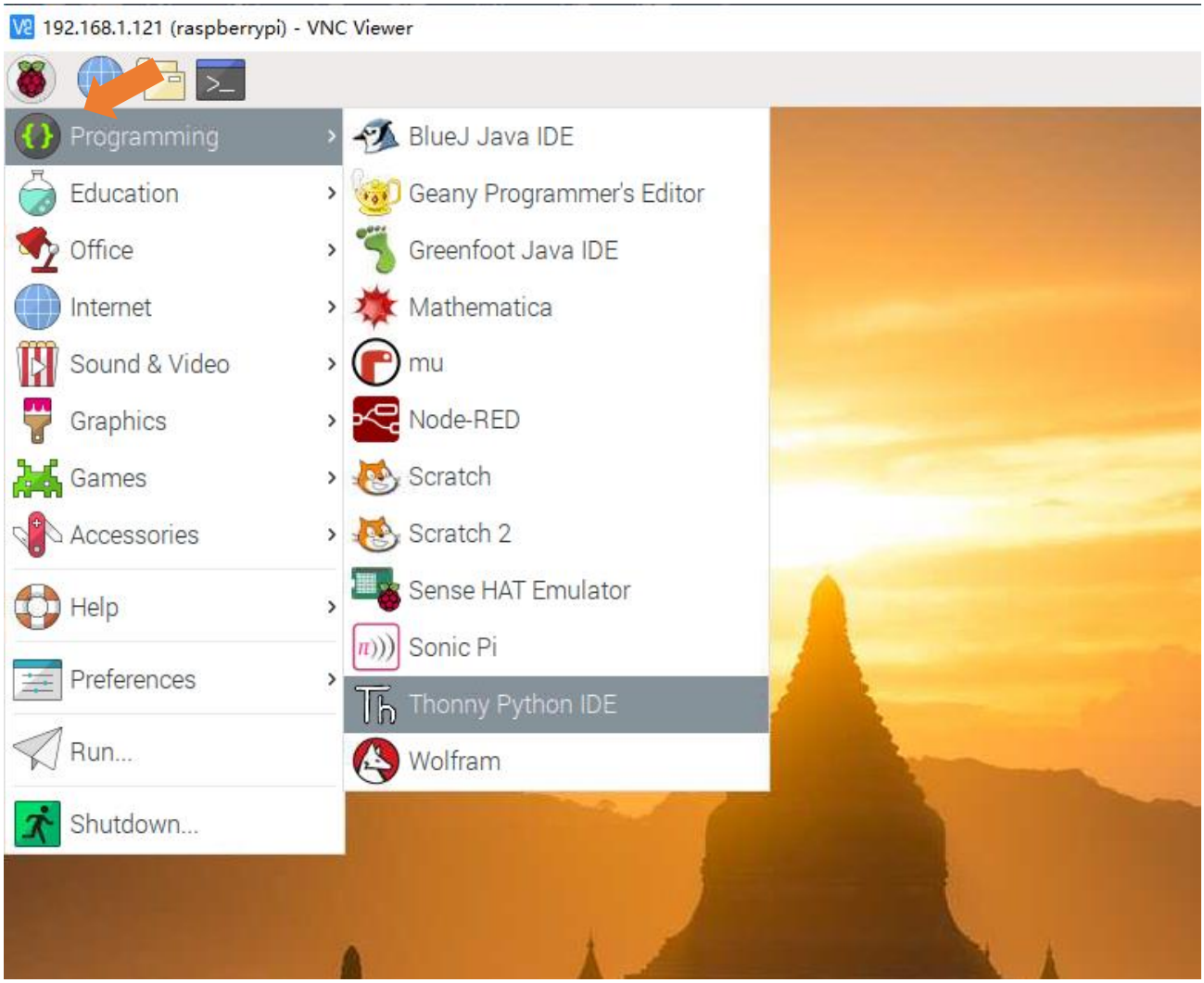
Open Freenove_Tank_Robot_Kit_for_Raspberry_Pi/Code/Server in your Raspberry Pi and copy the following **5 files** into the Test folder we created.

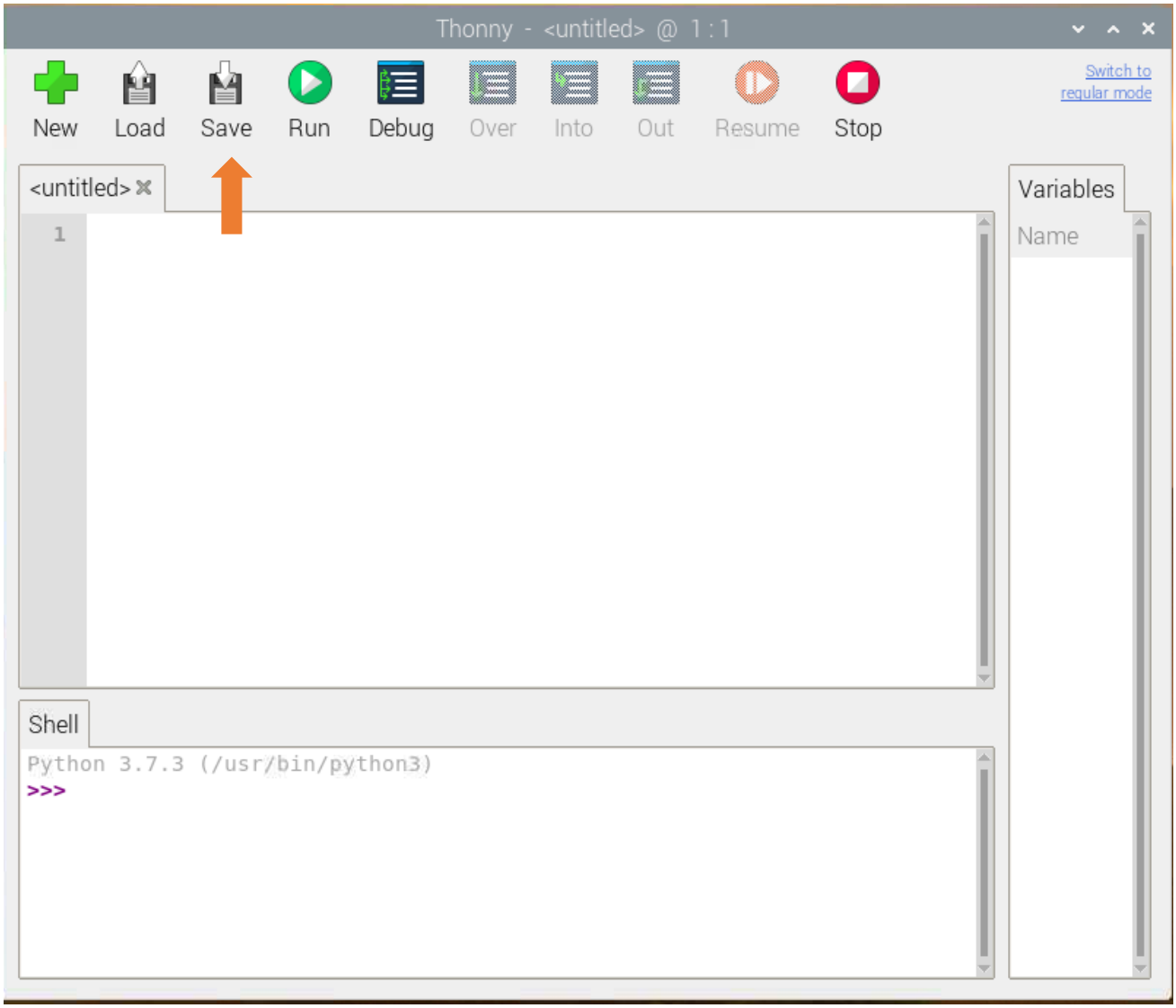


Paste them in Test folder.

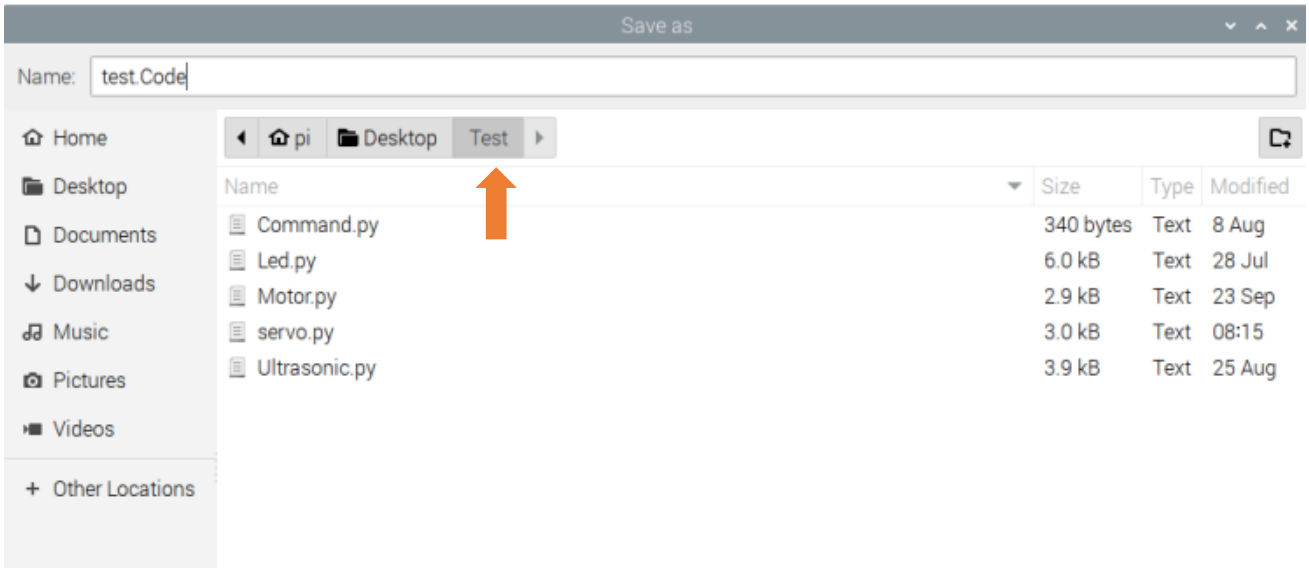


Run Thonny Python IDE

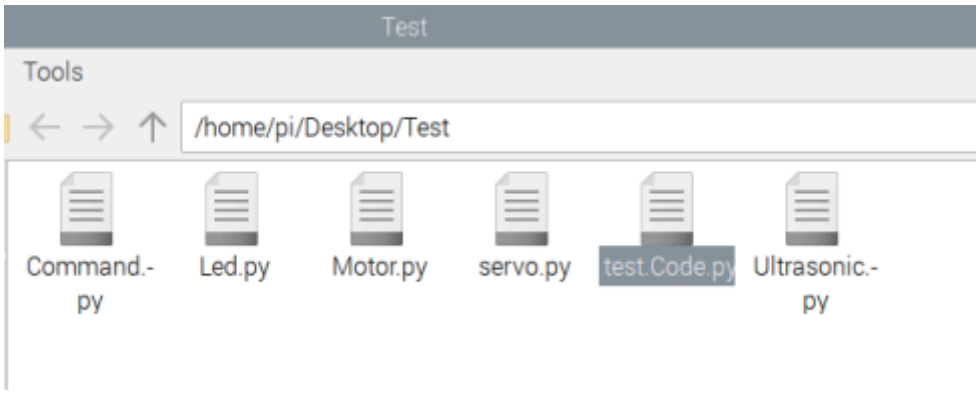

192.168.1.121 (raspberrypi) - VNC Viewer
Programming
Education
Office
Internet
Sound & Video
Graphics
Games
Accessories
Help
Preferences
Run...
Shutdown...
BlueJ Java IDE
Geany Programmer's Editor
Greenfoot Java IDE
Mathematica
mu
Node-RED
Scratch
Scratch 2
Sense HAT Emulator
Sonic Pi
Thonny Python IDE
Wolfram



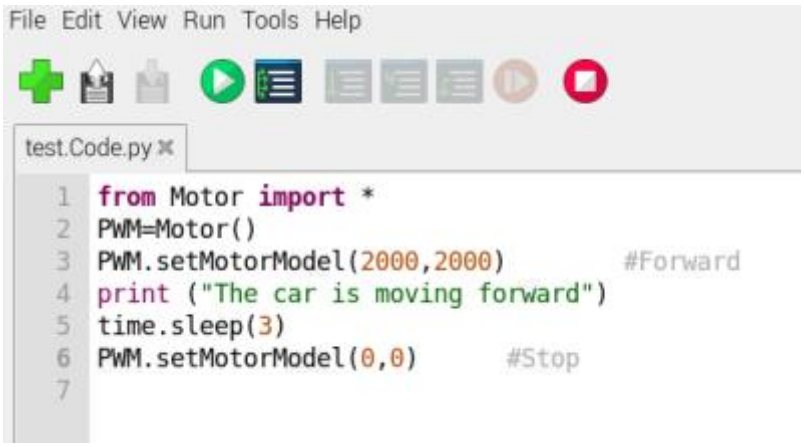
Click Save and save it into the Test folder, with name: test_Code.



Now you can see the file test_Code.py we created.



Then write code in test_Code.py, then click save.



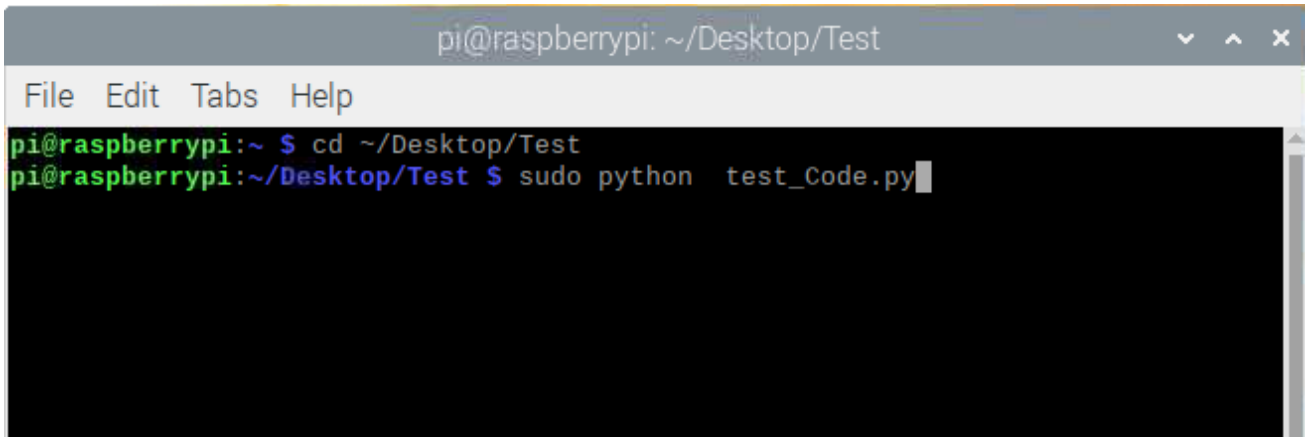
Note: the code and library are written by **Python 3.** You need execute the code with **python 3**.

Open the terminal and use the following command to enter the directory where test_Code.py is located:

```
cd ~/Desktop/Test
```

Run test_Code.py:

```
sudo python  test_Code.py
```



Code example

Following are code example for the parts. For more detail, please refer to Module test section.

For more details, please refer to Motor.

```
from Motor import *                          #import Motor
PWM=Motor()                                  #create an object
PWM.setMotorModel(2000,2000)     #Forward
print("The car is moving forward")
time.sleep(3)                                #waiting 3 second
PWM.setMotorModel(0,0)                  #Stop
```

LED. For more details, please refer to LED.

```
from Led import *                        #import Led
led=Led()                                #create an object
led.ledIndex(0x04,255,255,0)               #yellow
led.ledIndex(0x80,0,255,0)                 #green
time.sleep(5)                            #wait 5s
led.colorWipe(led.strip, Color(0,0,0))  #turn off
```

Servo. For more details, please refer to Servo.

```
from servo import *   #import Led
pwm = Servo()            #create an object
#Servo rotates from 90 degrees to 150 degrees
for i in range(90, 150, 1) :
    pwm.setServoPwm('0', i)
    time.sleep(0.01)
#Servo rotates from 140 degrees to 90 degrees
for i in range(145, 90, -1) :
    pwm.setServoPwm('0', i)
    time.sleep(0.01)
```

Ultrasonic module. For more details, please refer to Ultrasonic module.

```
from Ultrasonic import *          #import Led
ultrasonic=Ultrasonic()           #create an object
data=ultrasonic.get_distance()    #Get the value
print ("Obstacle distance is "+str(data)+"CM")
```

These codes can be integrated into one code to achieve your requirement.

What's next?

Thanks for your reading.

This book is all over here. If you find any mistakes, missions or you have other ideas and questions about contents of this book or the kit and ect., please feel free to contact us, and we will check and correct it as soon as possible.

After completing the contents in this book, you can try to reform this smart car, such as purchasing and installing other Freenove electronic modules, or improving the code to achieve different functions. We will also try our best to add more new functions and update the code on our github (https://github.com/freenove).

If you want to learn more about Arduino, Raspberry Pi, smart cars, robots and orther interesting products in science and technology, please continue to focus on our website. We will continue to launch cost-effective, innovative and exciting products.
www.freenove.com

Thank you again for choosing Freenove products.