Welcome

Thank you for choosing Freenove products!

About Battery

First, read the document **About Battery.pdf** in the unzipped folder.
If you did not download the zip file, please download it and unzip it via link below.
https://github.com/Freenove/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/archive/master.zip

Get Support & Custommer Service

You may find somethings missing or broken, or some difficulty to learn the kit.
Freenove provides free and quick support, including but not limited to:

- Quality problems of products
- Problems in using products
- Questions for learning and technology
- Opinions and suggestions
- Ideas and thoughts

If you have any concerns, please send email to us:

support@freenove.com

And suggestions and feedbacks are welcomed. Many customers offered great feedbacks. According to that, we are keeping updating the kit and the tutorial to make it better. Thank you.

Safety

Pay attention to safety when using and storing this product:

- Do not expose children under 6 years of age to this product. Put it out of their reach.
- Children lack safety ability should use this product under the guardianship of adults.
- This product contains small and sharp parts. Do not swallow, prick and scratch to avoid injury.
- This product contains conductive parts. Do not hold them to touch power supply and other circuits.
- Some parts will rotate or move when it works. Do not touch them to avoid being bruised or scratched.
- The wrong operation may cause overheat. Do not touch and disconnect the power supply immediately.
- Operate in accordance with the requirements of the tutorial. Otherwise, the parts may be damaged.
- Store the product in a dry place and avoid direct sunlight.
- Turn off the power of the circuit before leaving.

About

Freenove provides open source electronic products and services.

Freenove is committed to helping customers learn programming and electronic knowledge, quickly realize their creative ideas and product prototypes and launching innovative products. Our services include:

- Kits of robots, smart cars and drones
- Kits for learning Arduino, Raspberry Pi and micro:bit
- Electronic components and modules, tools
- **Product customization service**

You can learn more about us or get our latest information through our website:

http://www.freenove.com

Copyright

FREENOVE®

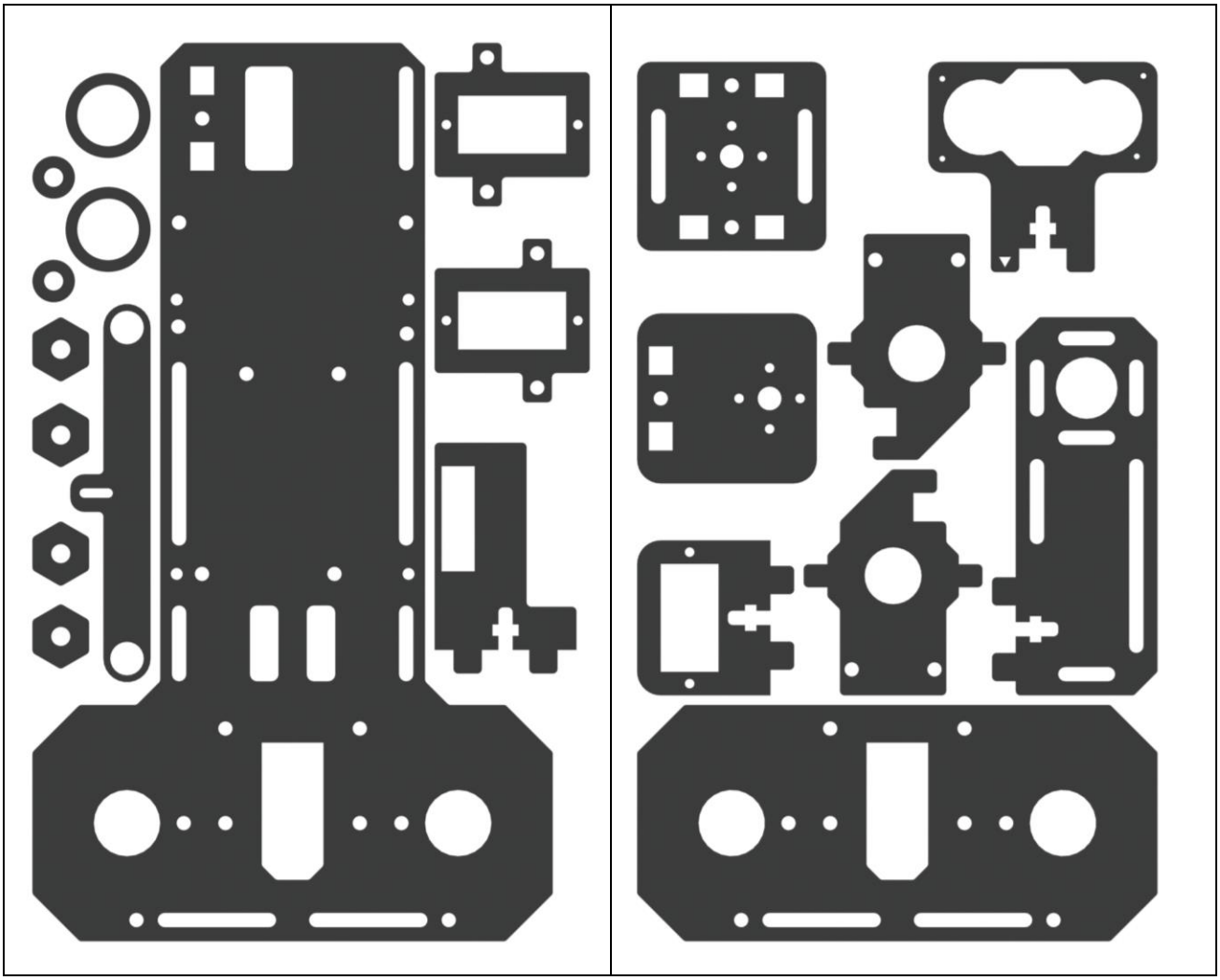
FREE YOUR INNOVATION

Contents

Welcome 1
Contents I
List 1
    Acrylic Sheet 1
    Machinery Parts 2
    Transmission Parts 3
    Electronic Parts 4
    Tools 5
    Self-prepared Parts 5
Preface 6
    Raspberry Pi 7
Preparation 16
    Install the System 16
    Remote desktop & VNC 23
Smart Car Shield for RPi 34
Chapter 0 Software installation and Test 36
    Step 0.1 Obtain the Code 36
    Step 0.2 Configure I2C 38
    Step 0.3 Install mjpg-streamer 41
    Step 0.4 Install PyQt5 45
    Step 0.5 Test 46
    Step 0.6 Client 52
    Android app 56
Chapter 1 Smart video car 57
    Assembly 57
    Run the code 76
    Automatic Start 82
    Android App 85
Chapter 2 Ultrasonic Radar Car 87
    Assembly 87
    Run the Code 93
Chapter 3 Part of the Code 97
    Server and Client 97
Chapter 4 Contributions of other developers 103
    Extensions for Scratch 2 103
Chapter 5 Free your innovation 106
    Program 106
    Code Example 109
    Related Functions 111
What's next? 112

List

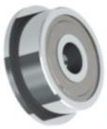
Acrylic Sheet



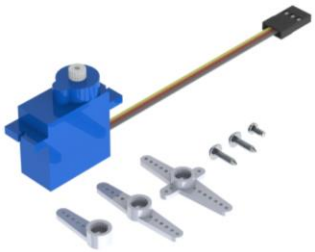
Surface of the acrylic sheet is covered with a layer of protective film. You need to remove it first before using. Some holes of the acrylic sheets may have residues, so you may need to clean them before using.

Machinery Parts

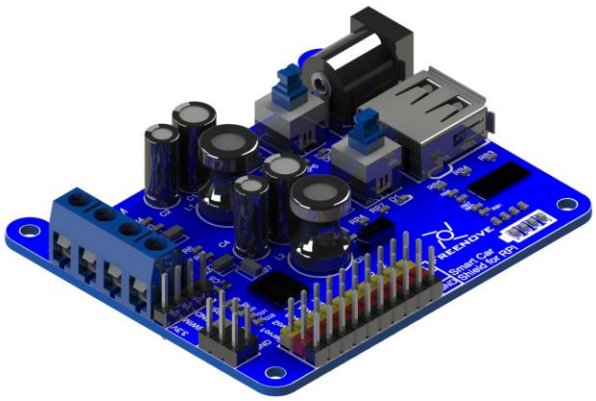
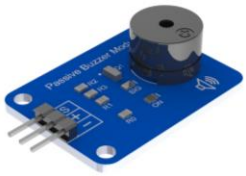
In addition to bearing F624ZZ and F687ZZ, the number of each following part provided is more than required.

M4 Washer  x4 Freenove	M2 Nut  x8 Freenove	M3 Nut  x11 Freenove	M4 Nut  x2 Freenove	M2*10 Screw  x8 Freenove	M3*4 Screw  x6 Freenove
M3*8 Screw  x22 Freenove	M3*12 Screw  x5 Freenove	M3*30 Screw  x6 Freenove	M4*40 Screw  x2 Freenove	M3*10 Countersunk Head Screw  x4 Freenove	M1.4*6 Self-tapping Screw  x4 Freenove
F624ZZ Bearing  x2 Freenove	F687ZZ Bearing  x4 Freenove	M3*6 Copper Standoff  x6 Freenove	M3*12 Copper Standoff  x6 Freenove	M3*30 Copper Standoff  x6 Freenove	M4 Spring Washer  x4 Freenove
M2.5*8 Scew  x10 Freenove	M2.5*6+6 M-F Copper Standoff  x6 Freenove	M2.5*14 Copper Standoff  x6 Freenove			

Transmission Parts

Servo x3 	Driven wheel x1 
DC speed reduction motor x2 	Driving wheel x2 

Electronic Parts

Freenove Smart Car Shield for RPi x1 	Freenove Passive Buzzer Module x1 	Freenove RGB LED Module x1 
18650x2 Battery Holder x1 	USB Camera x1 	HC-SR04 Ultrasonic Module x1 
Micro USB Cable x1 		
Type-C USB Cable x1 		
Jumper Wire M/M x10 		

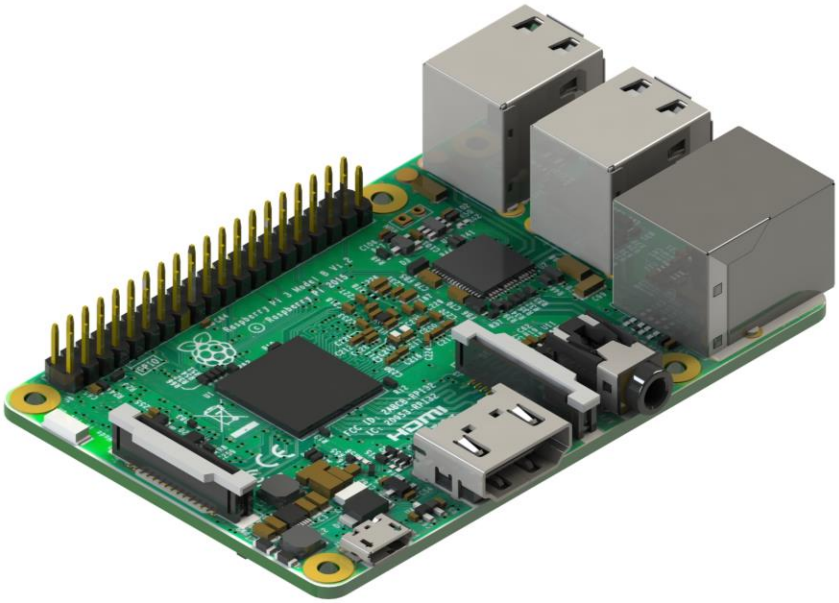
Tools

Cross screwdriver x1	Slotted screwdriver x1	Multifunctional Spanner x1
		

Self-prepared Parts

Two 18650 lithium batteries without protection board, with **continuous discharge current** > 3A.
It would be easier to find proper battery on eBay by search "18650 high drain".

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



Preface

Welcome to use Freenove Three-wheeled Smart Car Kit for Raspberry Pi. Whether you are a senior maker, or have little technical knowledge, by using this tutorial, you can make a very cool smart car with video surveillance and ultrasonic radar.

This kit is based on the popular control panel Pi Raspberry, 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 problem, you can always look for fast and free technical support through support@freenove.com to ensure that your smart car is successfully assembled and run.

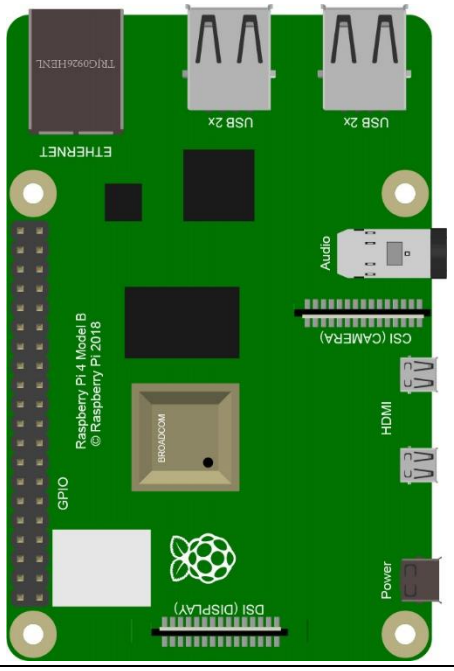
The contents in this book can ensure enthusiastic with little technical knowledge to make the smart car. If you are very interested in Raspberry Pi, and want to learn how to program and build the circuit, please visit our website www.freenove.com or contact us to buy the kits designed for beginners:
Freenove Basic\LCD1602\Super\Ultrasonic\RFID\Ultimate Starter Kit for Raspberry Pi

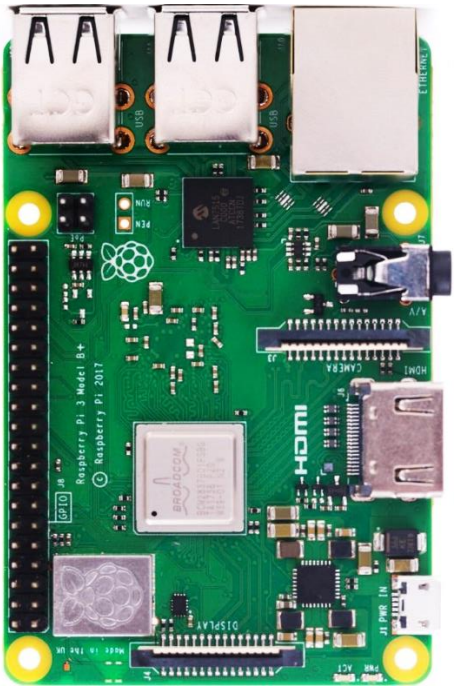
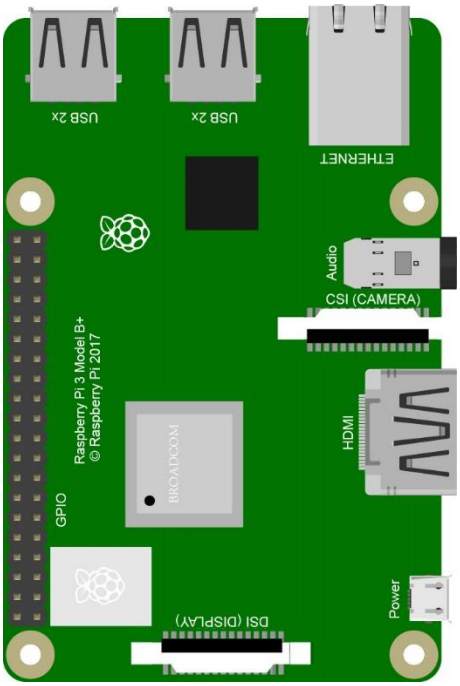
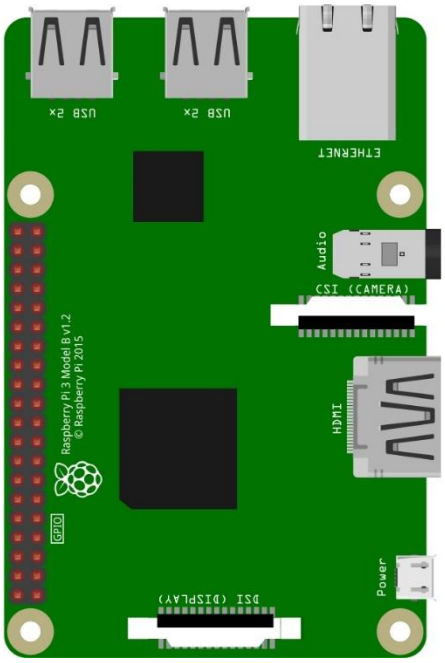
Raspberry Pi

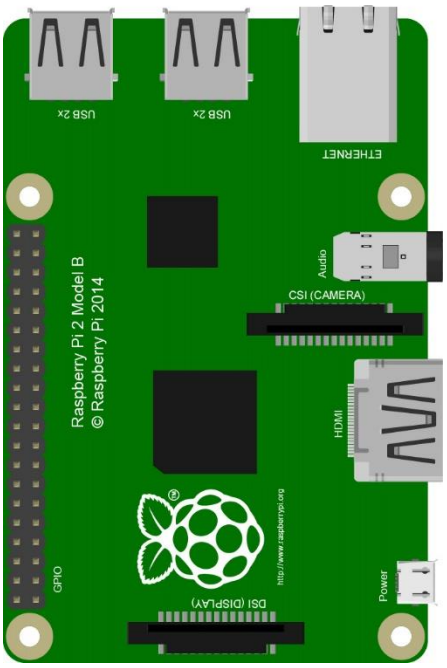
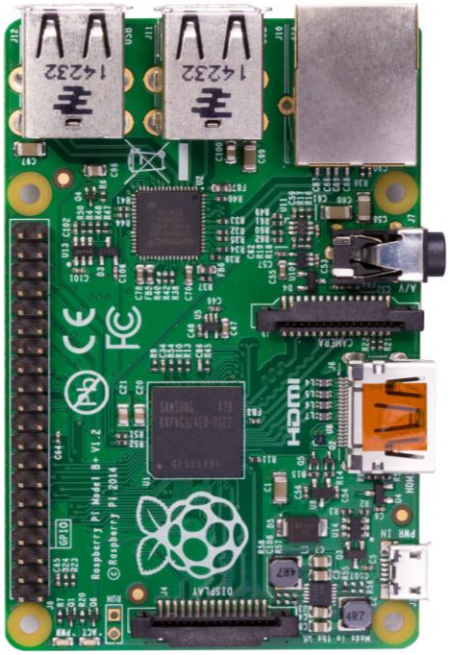
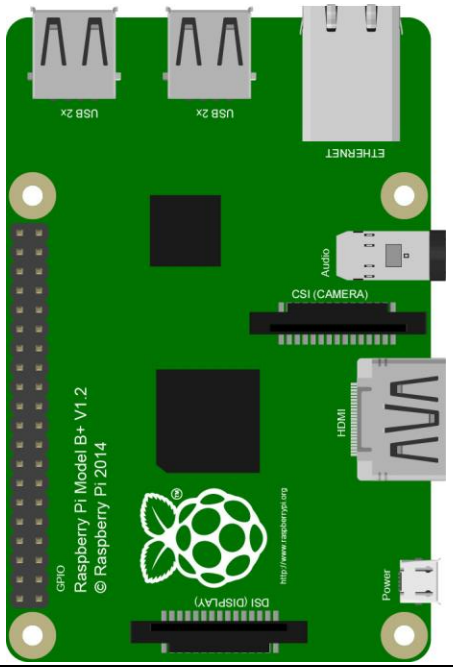
Raspberry Pi (called RPi, RPI, RasPi, the text these words will be used alternately behind), a micro computer with size of a card, quickly swept the world since its debut. It is widely used in desktop workstation, media center, smart home, robots, and even the servers, etc. It can do almost anything, which continues to attract fans to explore it. Raspberry Pi used to be running in Linux system and along with the release of windows 10 IoT, we can also run it in Windows. Raspberry Pi (with interfaces for 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.

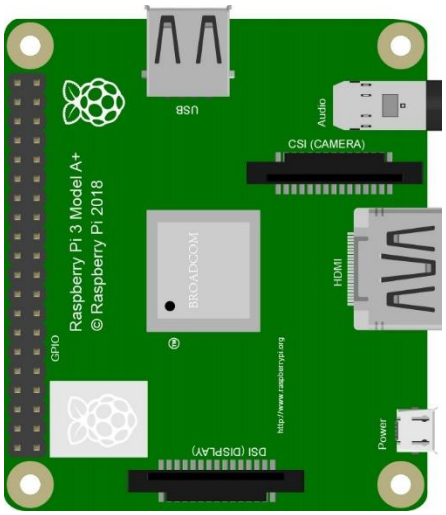
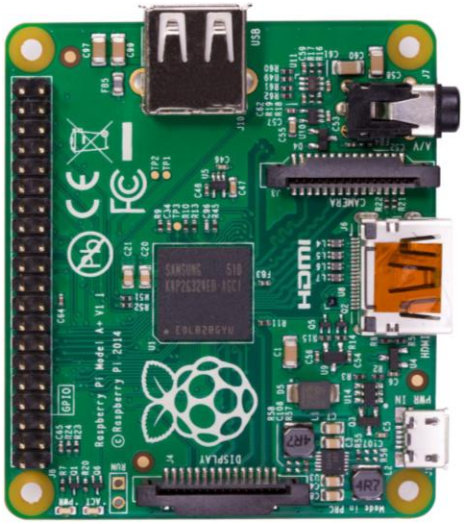
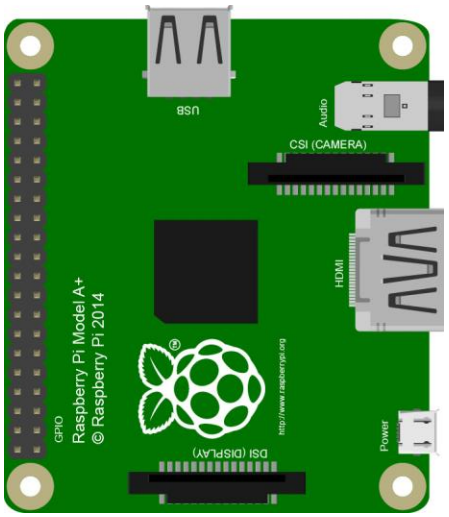
So far, Raspberry Pi has developed to the fourth generation. Changes in versions are accompanied by increase and upgrades in hardware. A type and B type, the first generation of products, have been stopped due to various reasons. Other versions are popular and active and the most important is that they are consistent in the order and number of pins, which makes the compatibility of peripheral devices greatly enhanced between different versions.

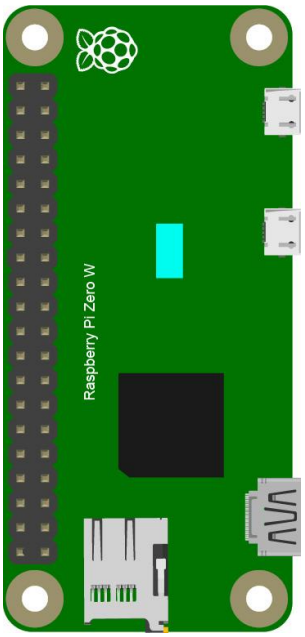
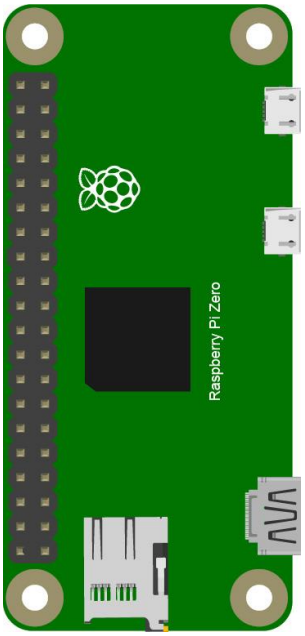
Here are some practicality pictures and model diagrams of Raspberry Pi:

Practicality picture of Raspberry Pi 4 Model B:	Model diagram of Raspberry Pi 4 Model B:
	

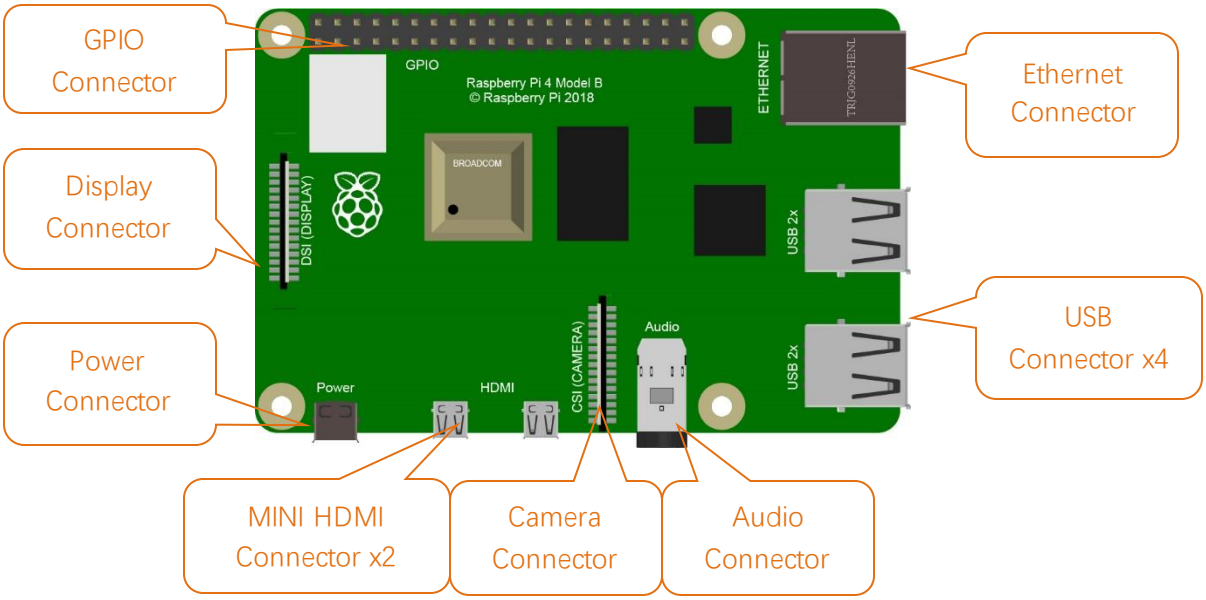
Practicality picture of Raspberry Pi 3 Model B+: 	Model diagram of Raspberry Pi 3 Model B+: 
Practicality picture of Raspberry Pi 3 Model B: 	Model diagram of Raspberry Pi 3 Model B: 

Practicality picture of Raspberry Pi 2 Model B:	Model diagram of Raspberry Pi 2 Model B:
	
Practicality picture of Raspberry Pi 1 Model B+:	Model diagram of Raspberry Pi 1 Model B+:
	

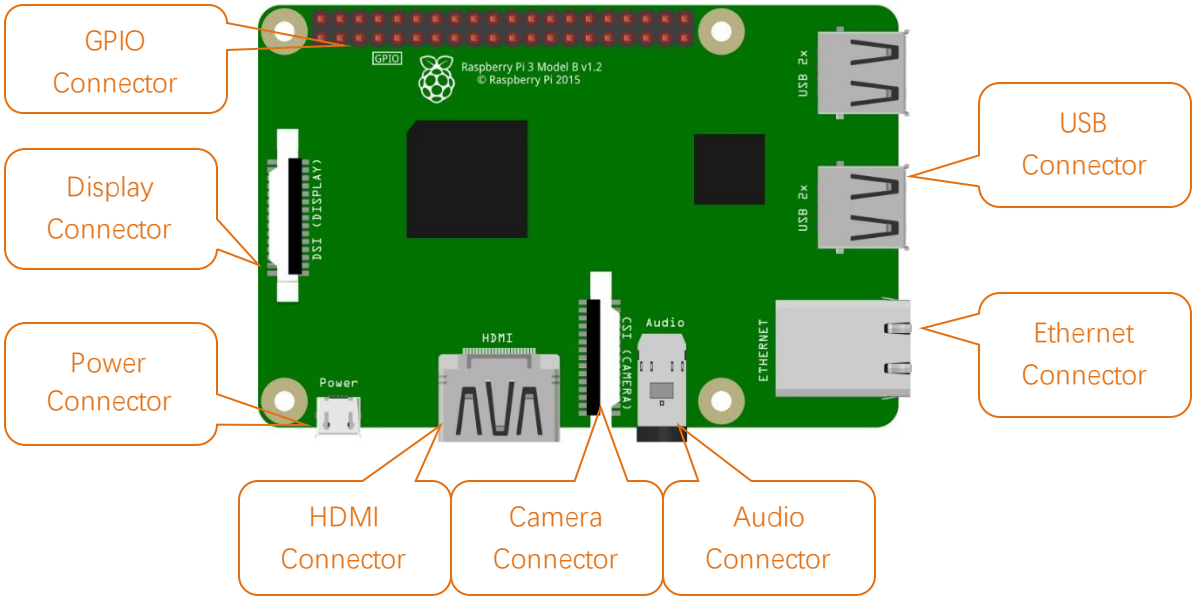
Practicality picture of Raspberry Pi 3 Model A+:	Model diagram of Raspberry Pi 3 Model A+:
	
Practicality picture of Raspberry Pi 1 Model A+:	Model diagram of Raspberry Pi 1 Model A+:
	

Practicality picture of Raspberry Pi Zero W:	Model diagram of Raspberry Pi Zero W:
	
Practicality picture of Raspberry Pi Zero:	Model diagram of Raspberry Pi Zero:
	

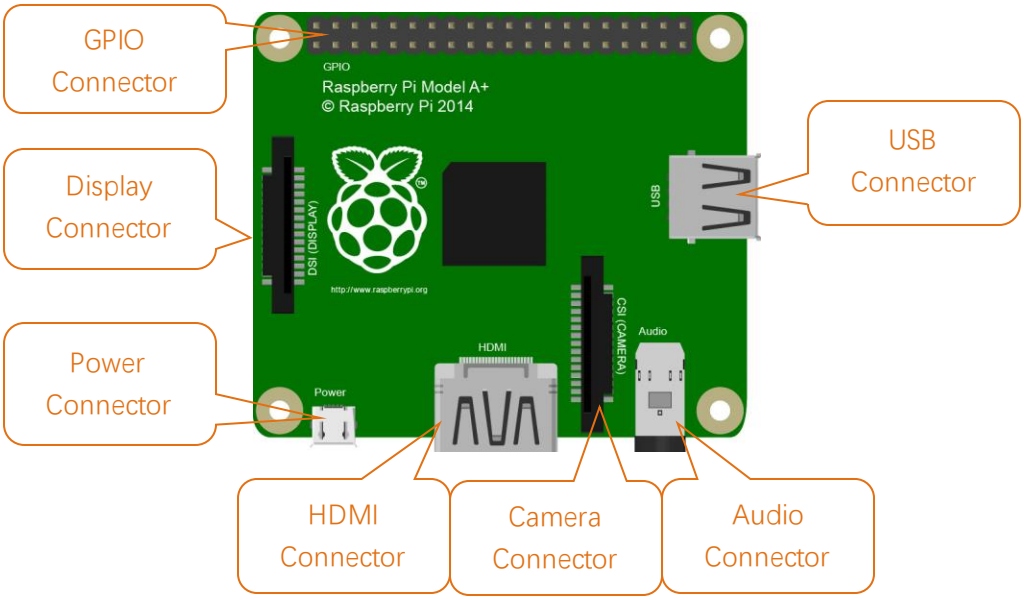
Hardware interface diagram of RPi 4B is shown below:



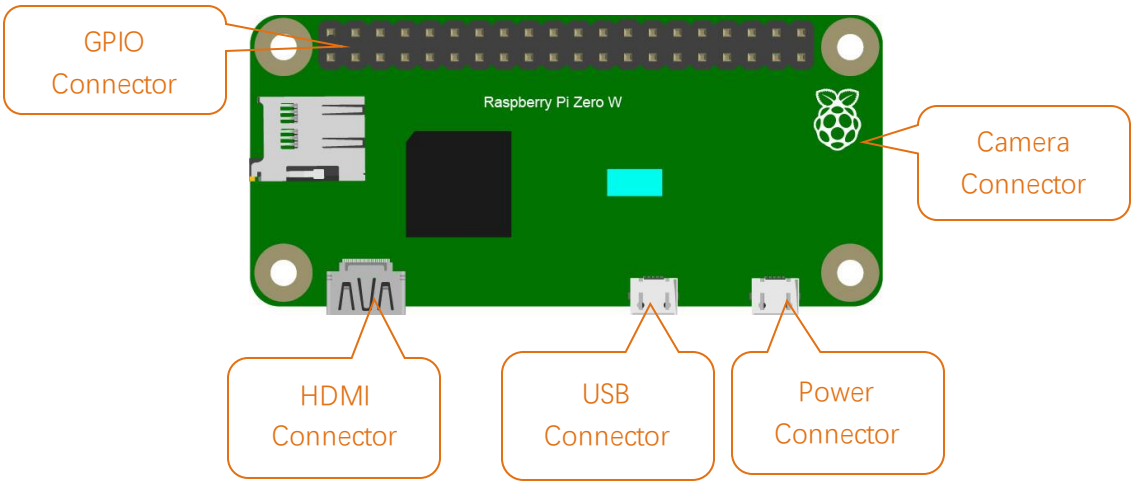
Hardware interface diagram of RPi 3B+/3B/2B/1B+ is shown below:



Hardware interface diagram of RPi A+ is shown below:



Hardware interface diagram of RPi Zero/Zero W is shown below:



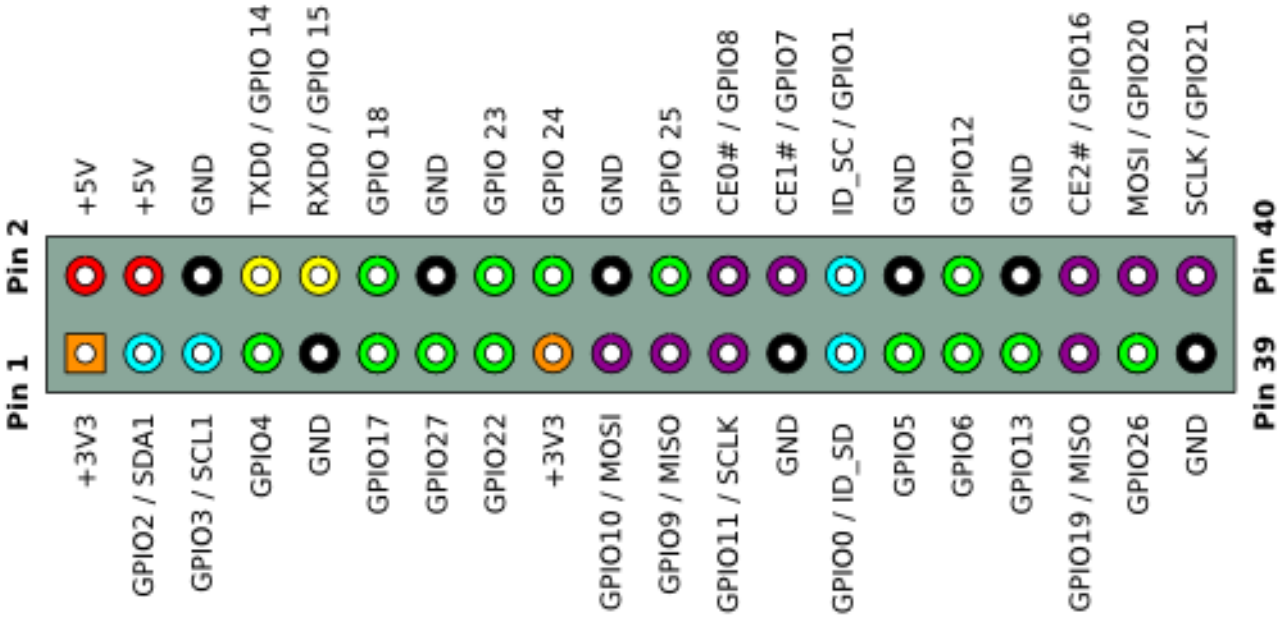
GPIO

General purpose input/output; in this specific case, the pins on the Raspberry Pi and what you can do with them. So called because you can use them for all sorts of purposes; most can be used as either inputs or outputs, depending on your program.
When programming the GPIO pins there are two different ways to refer to them: GPIO numbering and physical numbering.

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 the computer sees them. The numbers don't make any sense to humans, they jump about all over the place, so there is no easy way to remember them. You will need a printed reference or a reference board that fits over the pins.

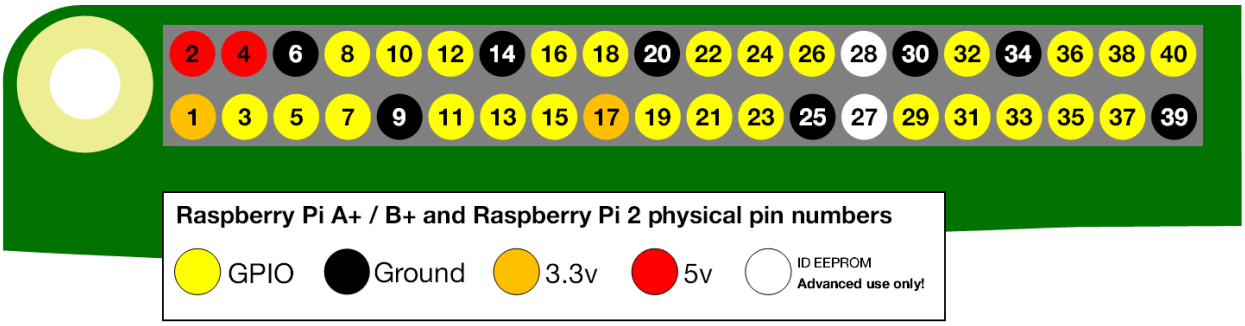
Each pin is defined as below:



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

PHYSICAL Numbering

The other 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' and it looks like this:



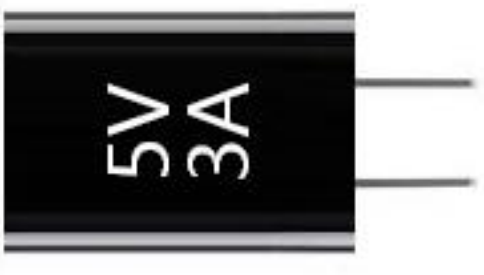
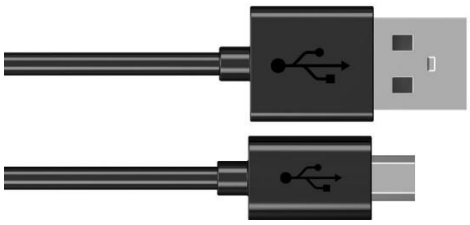
Preparation

Install the System

Firstly, install a system for your RPi.

Component List

Required Components

Raspberry Pi 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.

In this kit, we highly recommend Raspberry Pi 4B / 3B+ / 3A+ / 3B.

Raspberry Pi 2B / B+ is also usable, but an additional USB Wi-Fi module is required.

Raspberry Pi A+ / Zero / Zero W are not recommended.

Power requirement of different versions of Raspberry Pi is shown in following table:

Product	Recommended PSU current capacity	Maximum total USB peripheral current draw	Typical bare-board active current consumption
Raspberry Pi Model A	700mA	500mA	200mA
Raspberry Pi Model B	1.2A	500mA	500mA
Raspberry Pi Model A+	700mA	500mA	180mA
Raspberry Pi Model B+	1.8A	600mA/1.2A (switchable)	330mA
Raspberry Pi 2 Model B	1.8A	600mA/1.2A (switchable)	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 Zero W	1.2A	Limited by PSU, board, and connector ratings only.	150mA
Raspberry Pi Zero	1.2A	Limited by PSU, board, and connector ratings only	100mA

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

In addition, RPi also needs a network cable used to connect it to wide area network.

All of these components are necessary. Among them, the power supply is required at least 5V/2.5A, because lack of power supply will lead to many abnormal problems, even damage to your RPi. So power supply with 5V/2.5A is highly recommend. SD Card Micro (recommended capacity 16GB or more) is a hard drive for RPi, which is used to store the system and personal files. In later projects, the components list with a RPi will contains these required components, using only RPi as a representative rather than presenting details.

Optional Components

Under normal circumstances, there are two ways to login to Raspberry Pi: using independent monitor, or remote desktop to share a monitor with your PC.

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. Mini-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, the optional items are slightly different. But all of their aims are to convert the special interface to standard interface of standard Raspberry Pi.

Item	Pi Zero	Pi Zero W	Pi A+	Pi 3A+	Pi B+/2B	Pi 3B/3B+/4B
Monitor	Yes	Yes	Yes	Yes	Yes	Yes
Mouse	Yes	Yes	Yes	Yes	Yes	Yes
Keyboard	Yes	Yes	Yes	Yes	Yes	Yes
Mini-HDMI to HDMI cable	Yes	Yes	No	No	No	No
Micro-USB to USB-A OTG cable	Yes	Yes	No	No	No	No
USB HUB	Yes	Yes	Yes	Yes	No	No
USB transferring to Ethernet interface	select one from two or select two from two	optional	select one from two or select two from two	optional	Internal Integration	Internal Integration
USB Wi-Fi receiver		Internal Integration		Internal Integration	optional	

Required Accessories for Remote Desktop

If you don't have an independent monitor, or you want to use a remote desktop, first you need to login to Raspberry Pi through SSH, then open the VNC or RDP service. So you need the following accessories.

Item	Pi Zero	Pi Zero W	Pi A+	Pi 3A+	Pi B+/2B	Pi 3B/3B+/4B
Micro-USB to USB-A OTG cable	Yes	Yes	No	NO		
USB transferring to Ethernet interface	Yes	Yes	Yes			

Raspbian System

Official Method

It is recommended to use this method.

You can follow the official method to install the system for raspberry pi

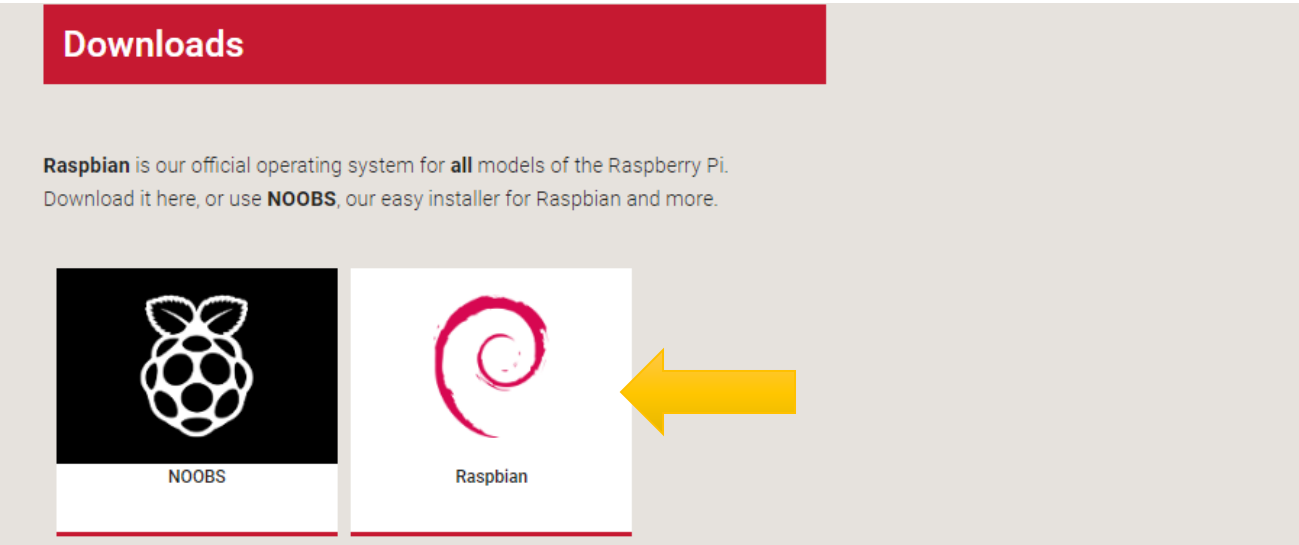
https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/2

In this way, the system will be download **automatically** via the application.

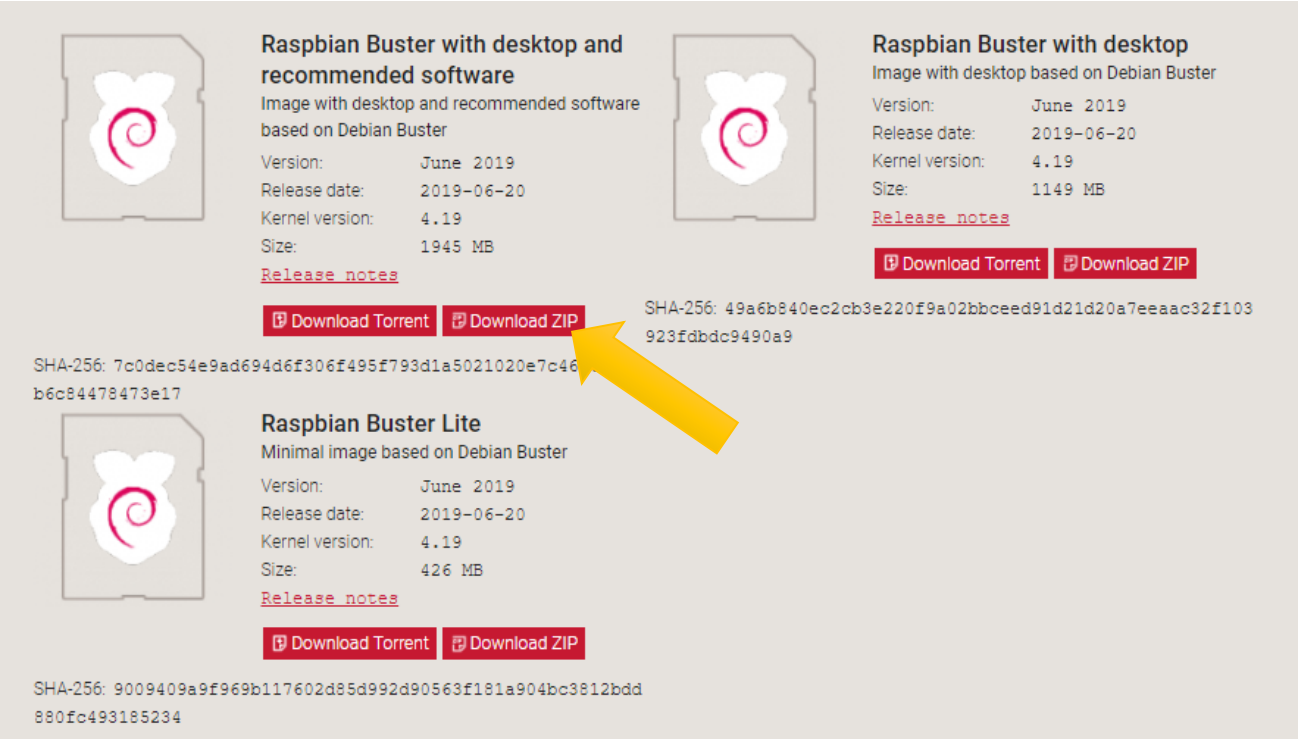
Download system manually (optional)

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

Visit RPi official website (https://www.RaspberryPi.org/), click "Downloads" and choose to download "RASPBIAN". RASPBIAN supported by RPI is an operating system based on Linux, which contains a number of contents required for RPi. We recommended RASPBIAN system to beginners. All projects in this tutorial are operated under the RASPBIAN system.



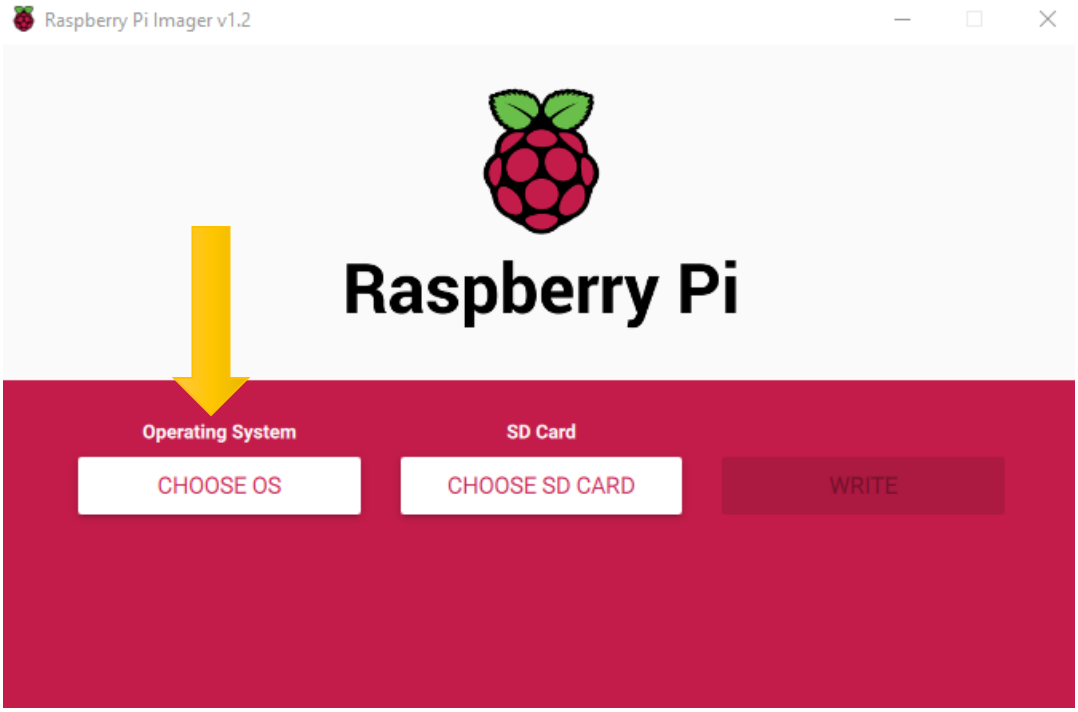
https://www.raspberrypi.org/downloads/raspbian/

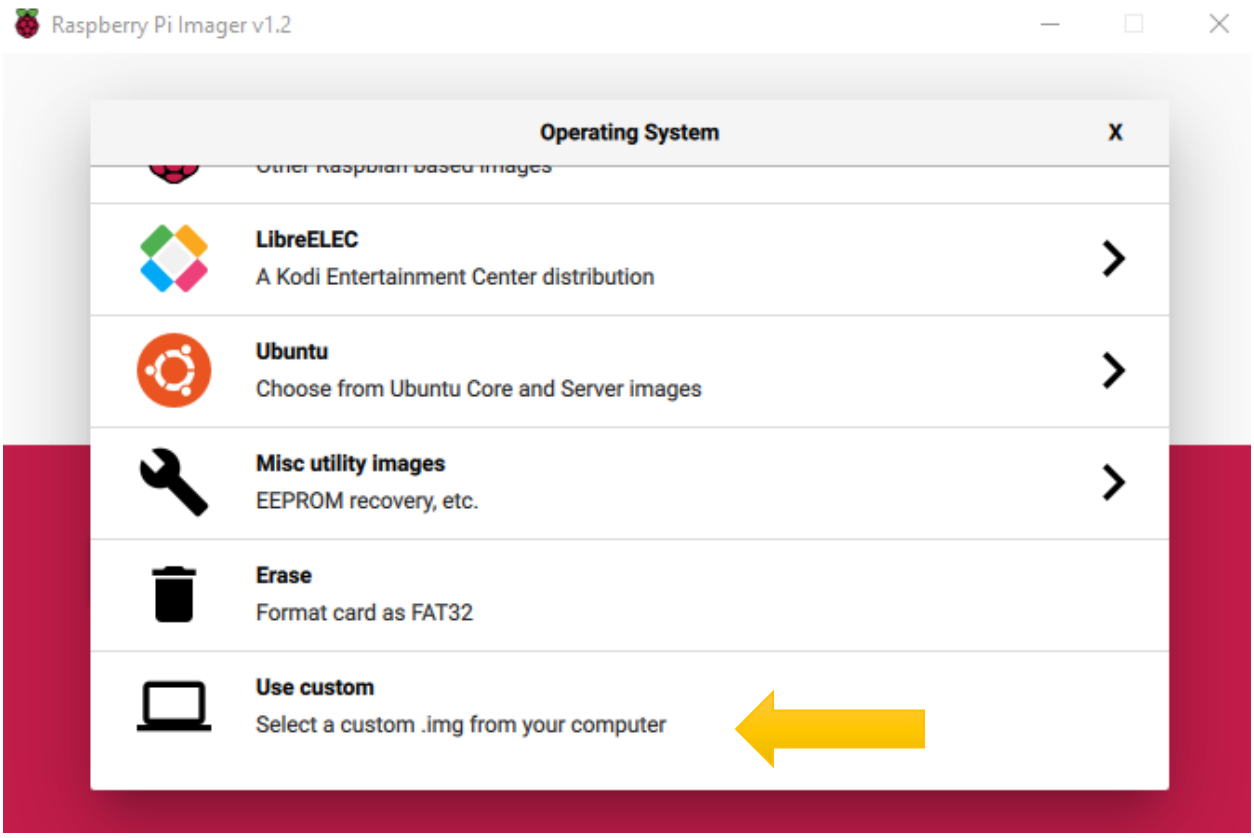


After the zip file is download.

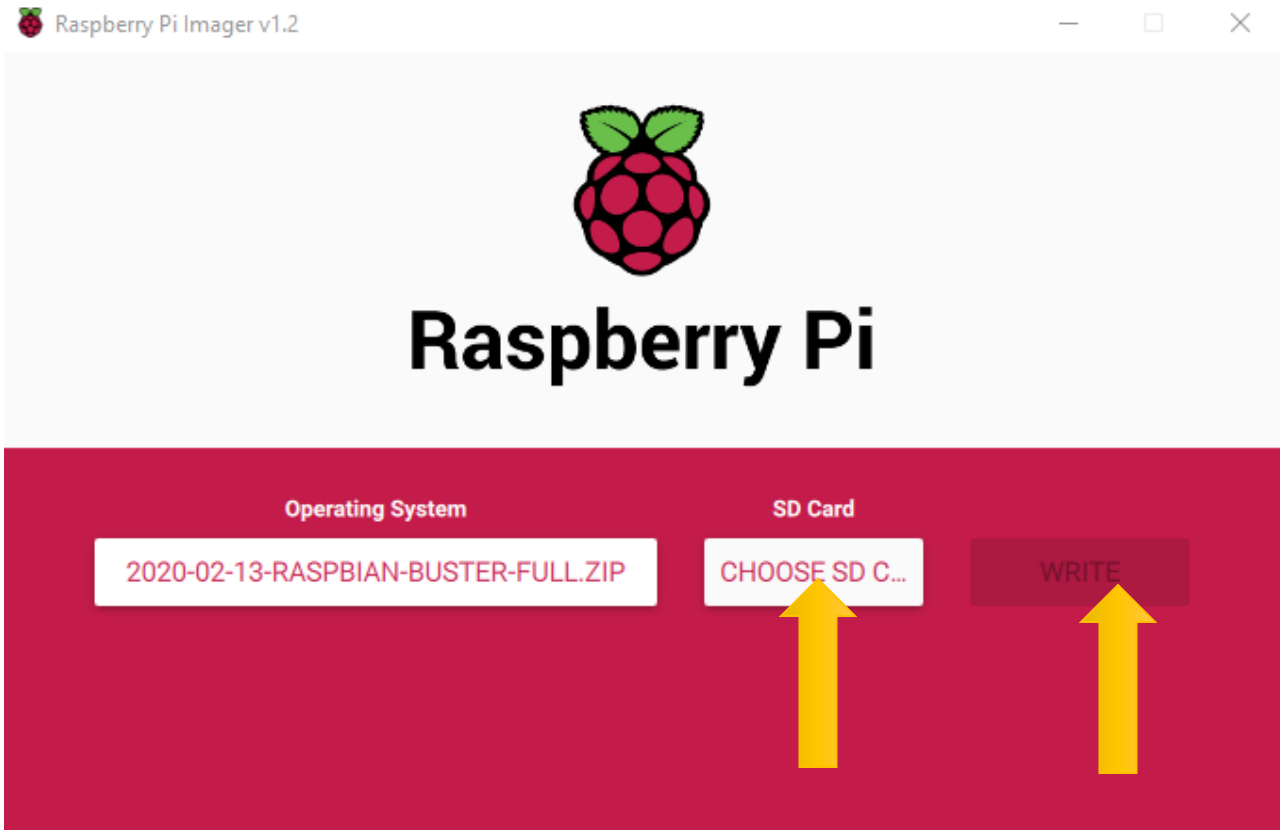
Write System to Micro SD Card

First, put your Micro **SD card** into card reader and connect it to USB port of PC. Then open imager toll, choose Choose system that you just download in Use custom.





Choose the SD card. Then click "WRITE".



Start Raspberry Pi

If you don't have a spare monitor, please jumper to next section. If you have a spare monitor, please follow steps in this section.

After the system is written successfully, take out Micro SD Card and put it into the card slot of RPi. Then connect RPi to screen through the HDMI, to mouse and keyboard through the USB port, to network cable through the network card interface and to the power supply. Then your RPi starts initially. Later, you need to enter the user name and password to login. The default user name: pi; password: raspberry. Enter and login. After login, you can enter the following interface.

Now, you have successfully installed the RASPBIAN operating system for your RPi.

Then you can connect WiFi on the right corner.

Now you can jumper to VNC Viewer.

Remote desktop & VNC

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 remote desktop under the Windows operating system to control RPi.

Under windows, Raspberry Pi can be generally accessed remotely through two applications. The first one is the windows built-in application remote desktop, which corresponds to the Raspberry Pi xrdp service. The second one is the free application VNC Viewer, which corresponds to the VNC interface of Raspberry Pi. Each way has its own advantages. You can choose either one or two.

Windows	Raspberry Pi
Remote Desktop Connection	Xrdp
VNC Viewer	VNC

VNC Viewer can not only run under Windows, but also under system MAC, Linux, IOS, Android and so on.
Some remote connection tools like Xrdp, it does not support opencv and pyqt window display. So it is recommended to use VNC Viewer to connect Raspberry Pi for this robot.

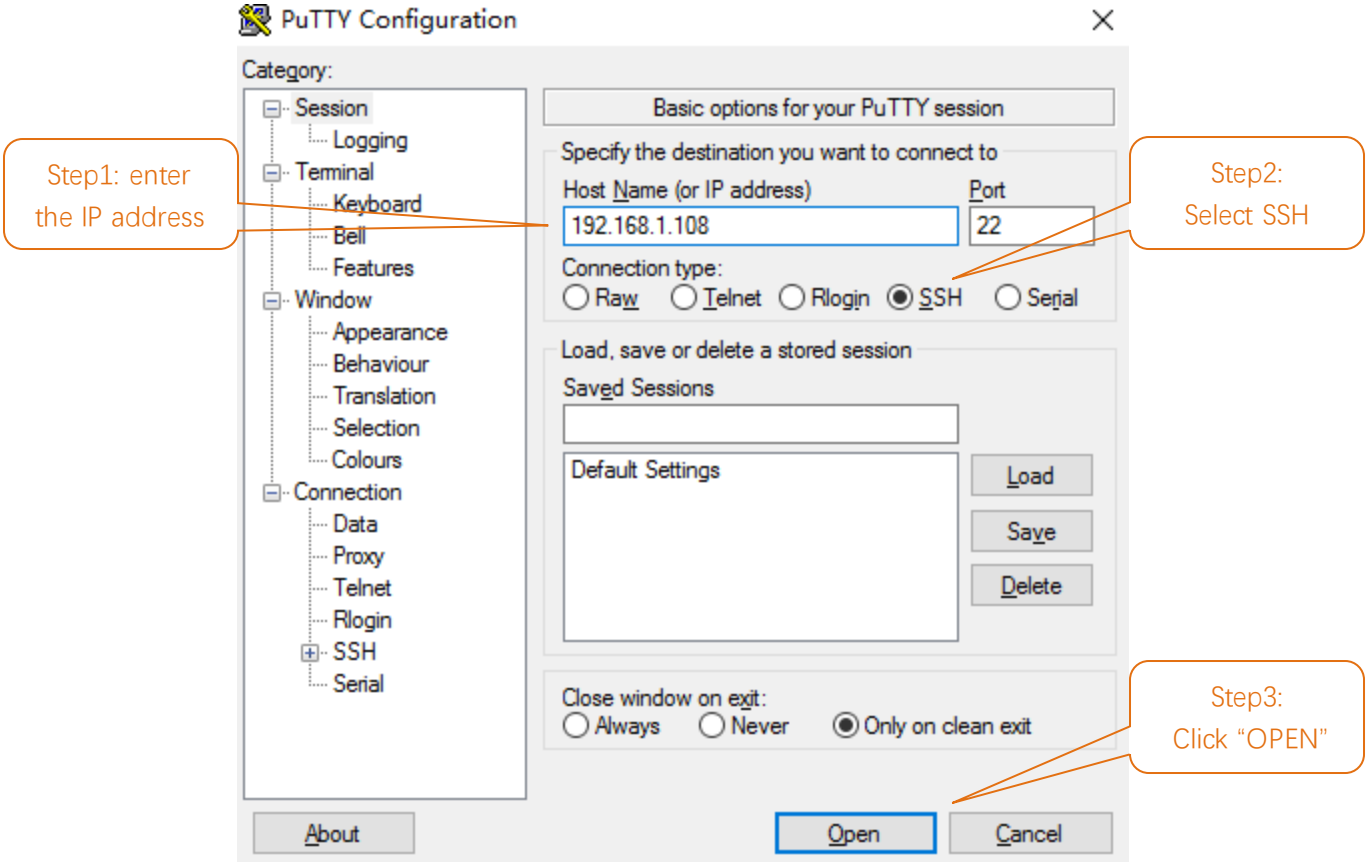
SSH

Under previous Raspbian system, SSH is opened by default. Under the latest version of Raspbian system, it is closed by default. So you need to open it first.

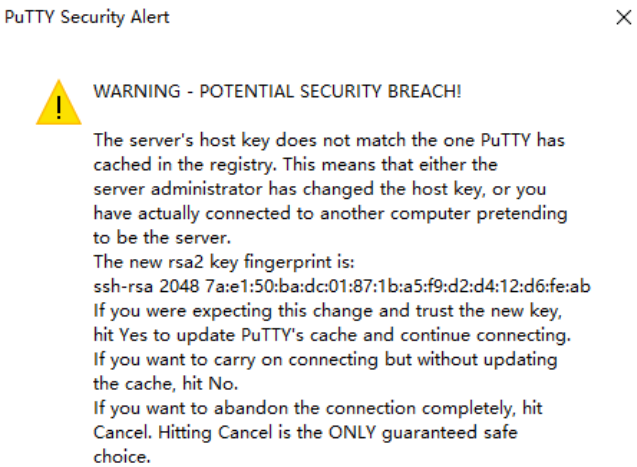
Method: after the system is written. Create a folder named "ssh" under generated boot disk, then the SSH connection will be opened.

And then, download the tool software Putty. Its official address: http://www.putty.org/
Or download it here: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

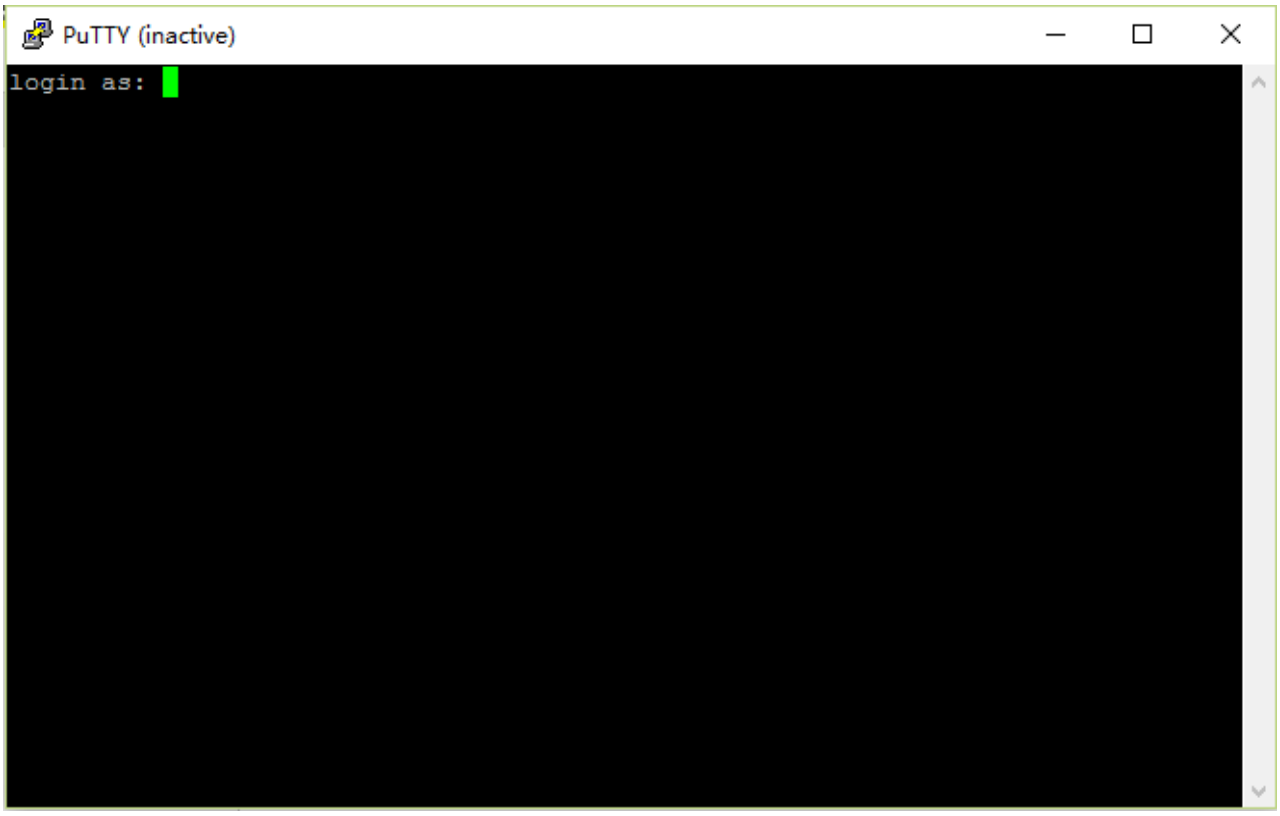
Then use cable to connect your RPi to the routers of your PC LAN, to ensure your PC and your RPi in the same LAN. Then put the system Micro SD Card prepared before into the slot of the RPi and turn on the power supply waiting for starting RPi. Later, enter control terminal of the router to inquiry IP address named "raspberry pi". For example, I have inquired to my RPi IP address, and it is "192.168.1.108". Then open Putty, enter the address, select SSH, and then click "OPEN", as shown below:



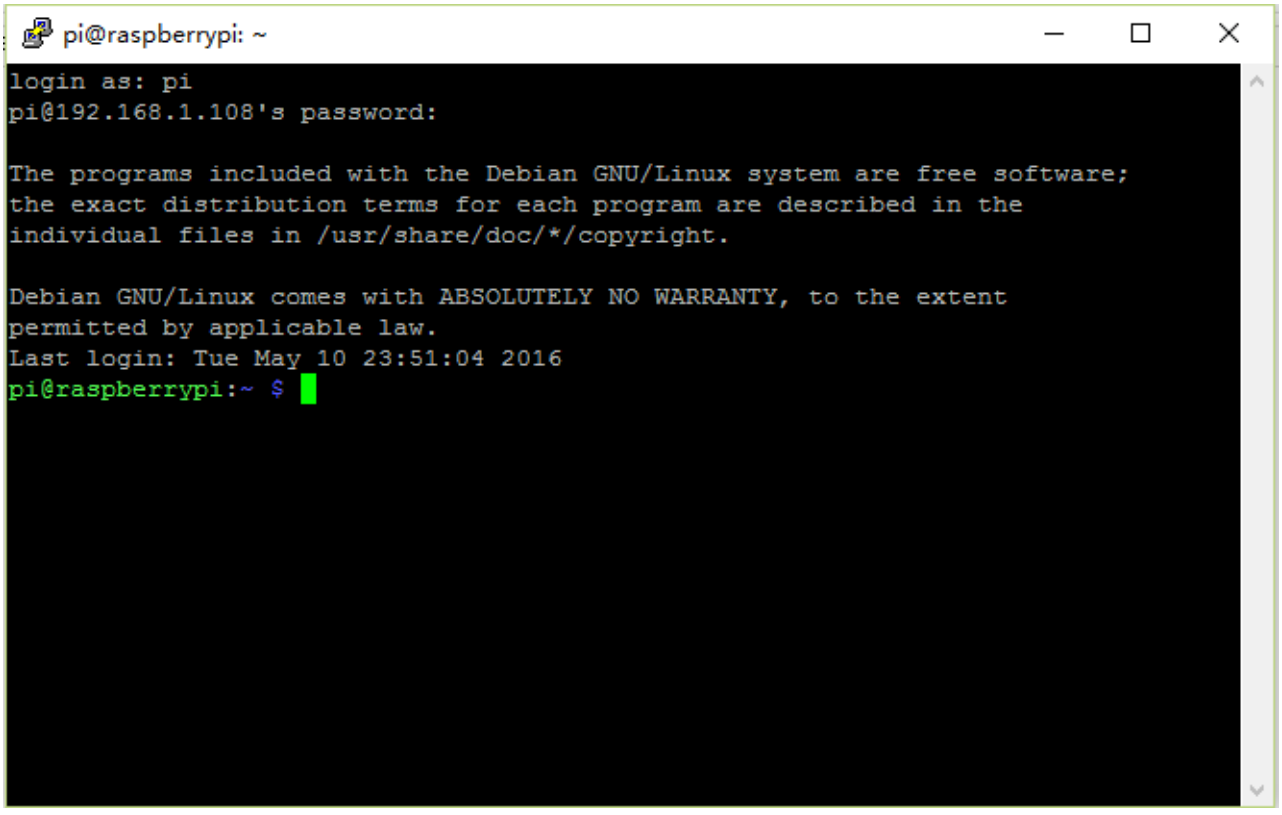
There will appear a security warning at first login. Just click "YES".



Then there will be a login interface (RPi default user name: **pi**; the password: **raspberry**). When you enter the password, there will be **no display** on the screen. This is normal. After the correct input, press "Enter" to confirm.



Then enter the command line of RPi, which means that you have successfully login to RPi command line mode.



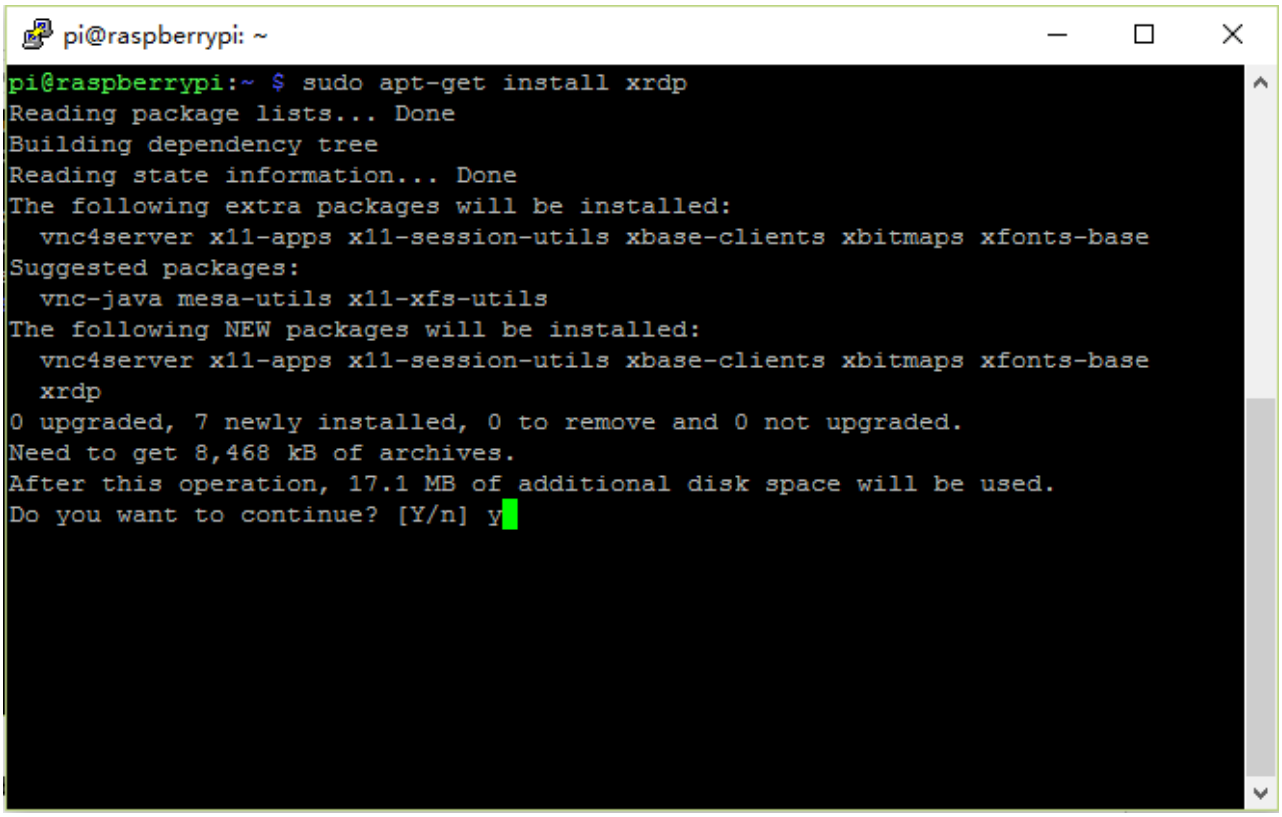
Remote Desktop Connection & xrdp

If you want to use built-in Remote Desktop Connection under Windows, you need install xrdp service on Raspberry Pi.

Next, install a xrdp service, an open source remote desktop protocol(xrdp) server, for RPi. Type the following command, then press enter to confirm:

```
sudo apt-get install xrdp
```

Later, the installation starts.

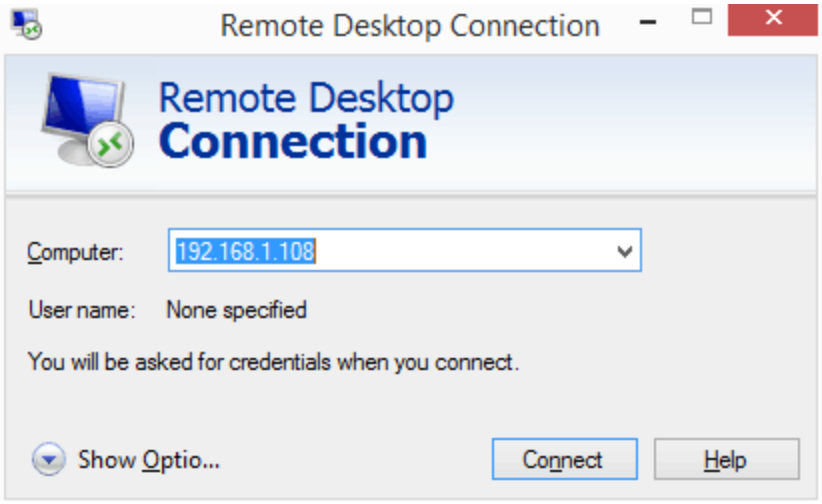


Enter "Y", press key “Enter” to confirm.

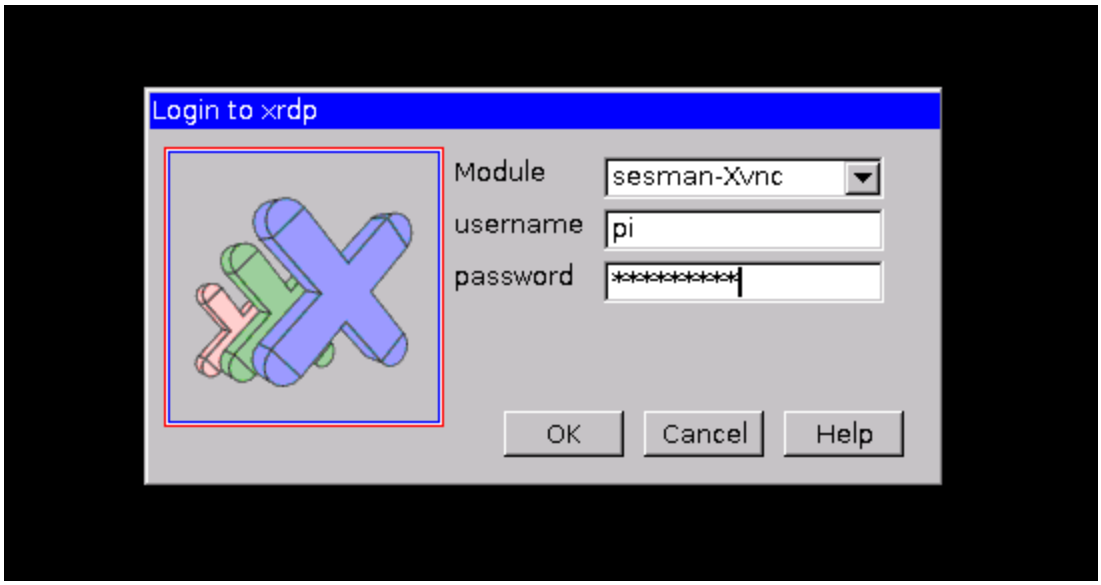
After the installation is completed, you can use Windows remote desktop applications to login to your RPi.

Login to Windows remote desktop

Use "WIN+R" or search function, open the remote desktop application "mstsc.exe" under Windows, enter the IP address of RPi and then click “Connect”.



Later, there will be xrdp login screen. Enter the user name and password of RPi (RPi default user name: pi; password: raspberry) and click “OK”.



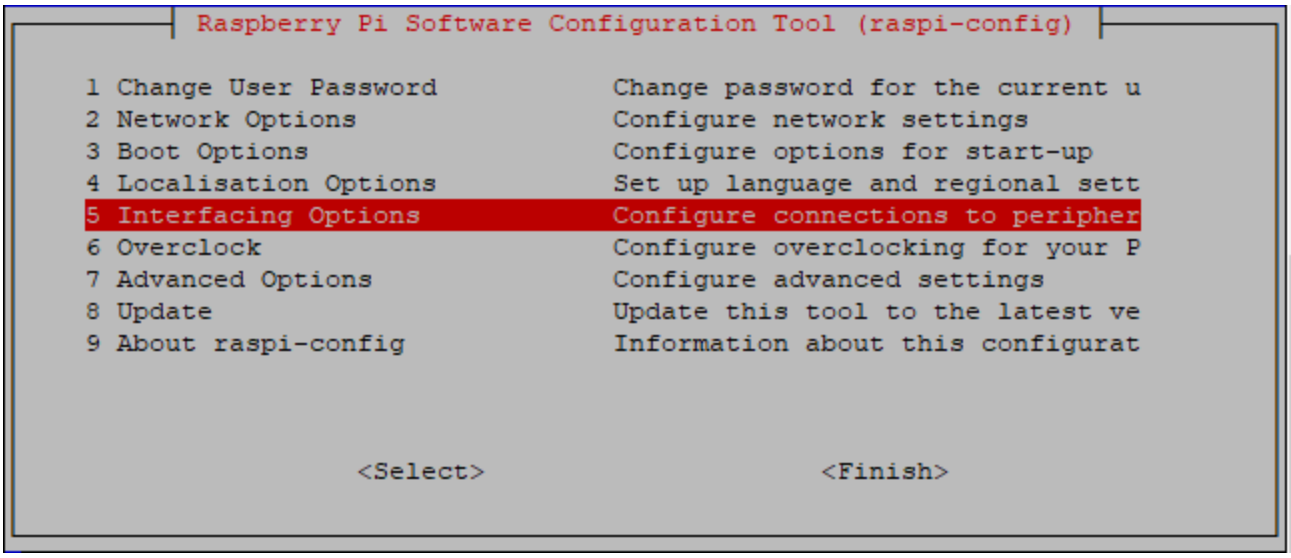
Later, you can enter the RPi desktop system.

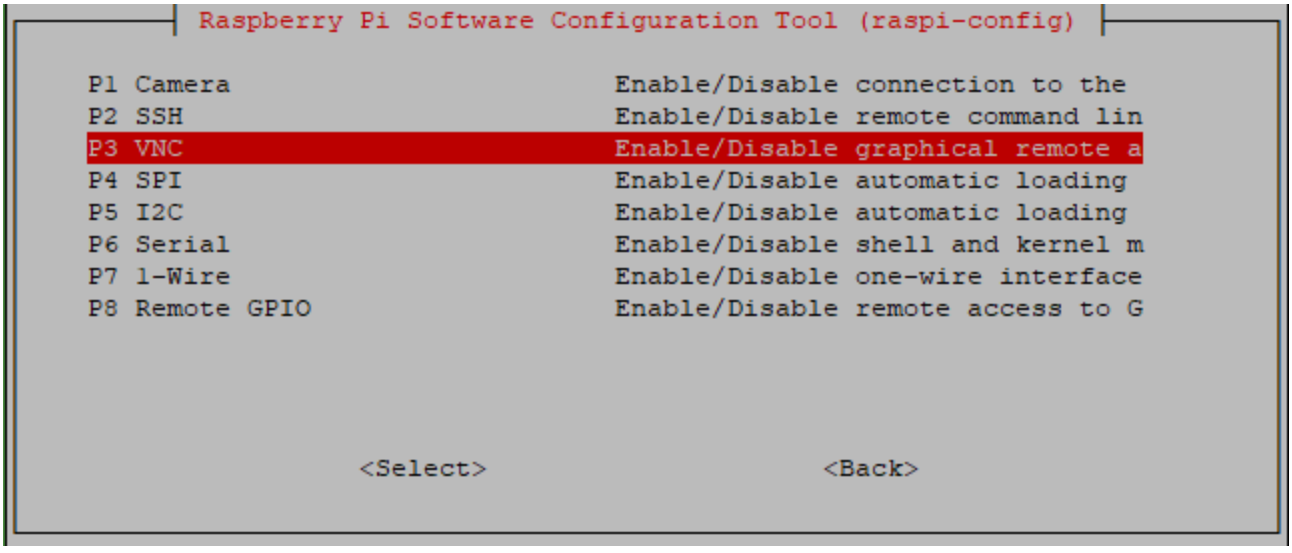
Here, you have successfully used the remote desktop login to RPi.

VNC Viewer & VNC

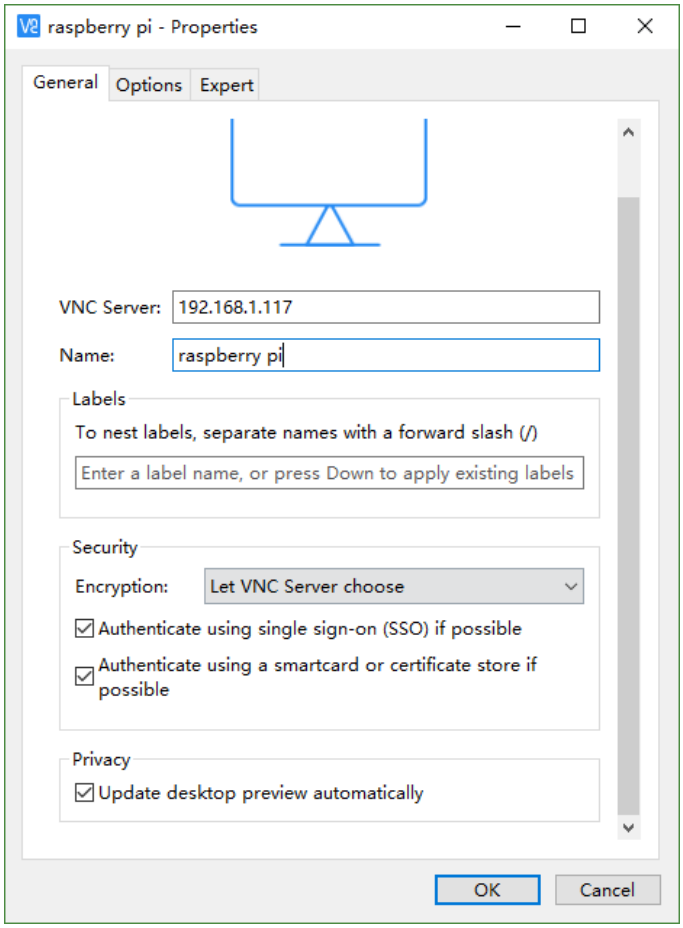
Type the following command. And select 5 Interfacing Options→P3 VNC →Yes→OK→Finish. Here Raspberry Pi may need be restarted, and choose ok. Then open VNC interface.

```
sudo raspi-config
```

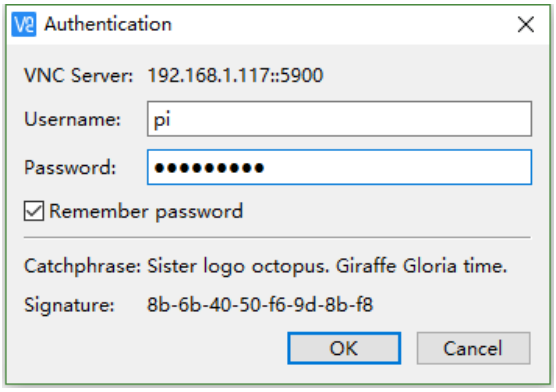




Then download and install VNC Viewer by click following link:
https://www.realvnc.com/en/connect/download/viewer/windows/
After installation is completed, open VNC Viewer. And click File → New Connection. Then the interface is shown below.



Enter ip address of your Raspberry Pi and fill in a Name. And click OK.
Then on the VNC Viewer panel, double-click new connection you just created, and the following dialog box pops up.



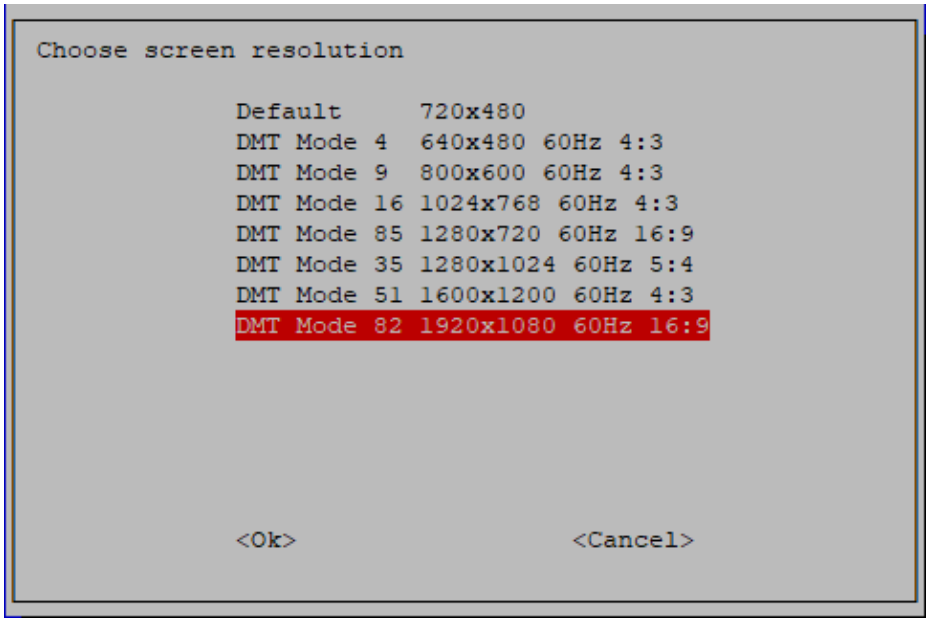
Enter username: **pi** and Password: **raspberry**. And click OK.

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

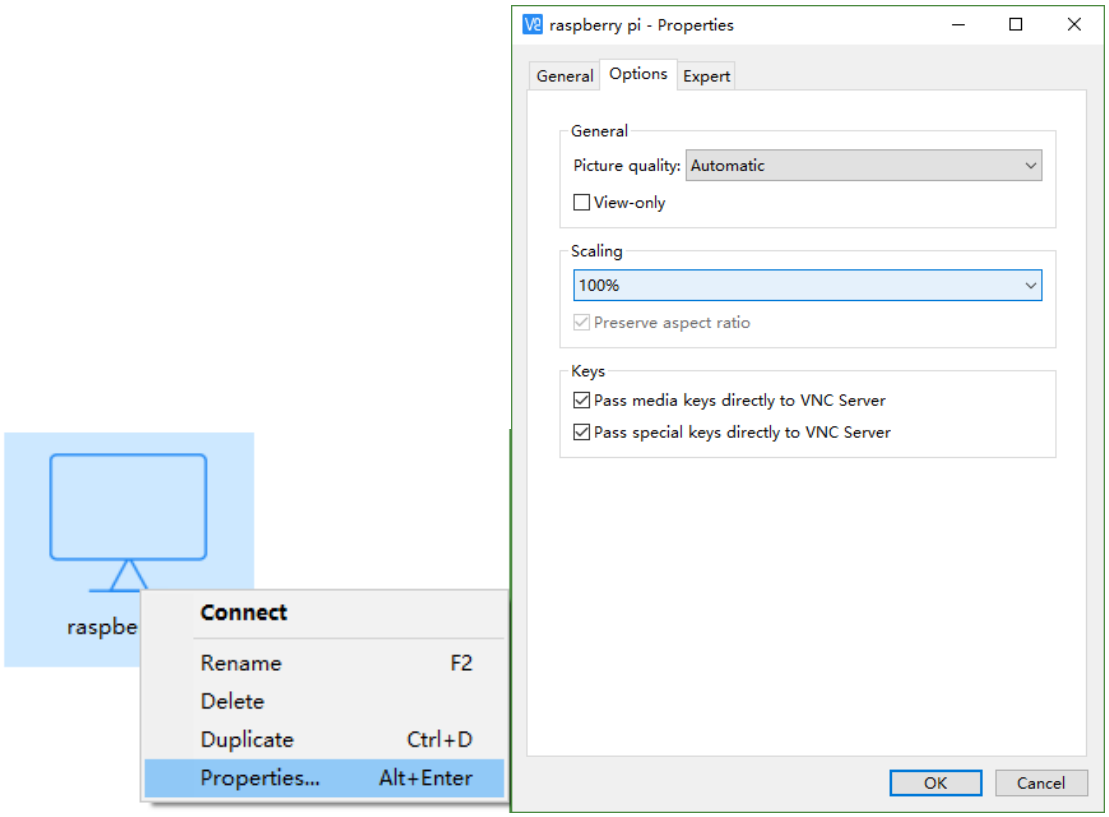
If the resolution ratio is not great or there is just a **black little window**, you can set a proper resolution ratio via steps below.

```
sudo raspi-config
```

Select 7 Advanced Options→A5 Resolution→proper resolution ratio (set by yourself)→OK→Finish. And then reboot Raspberry Pi.



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.



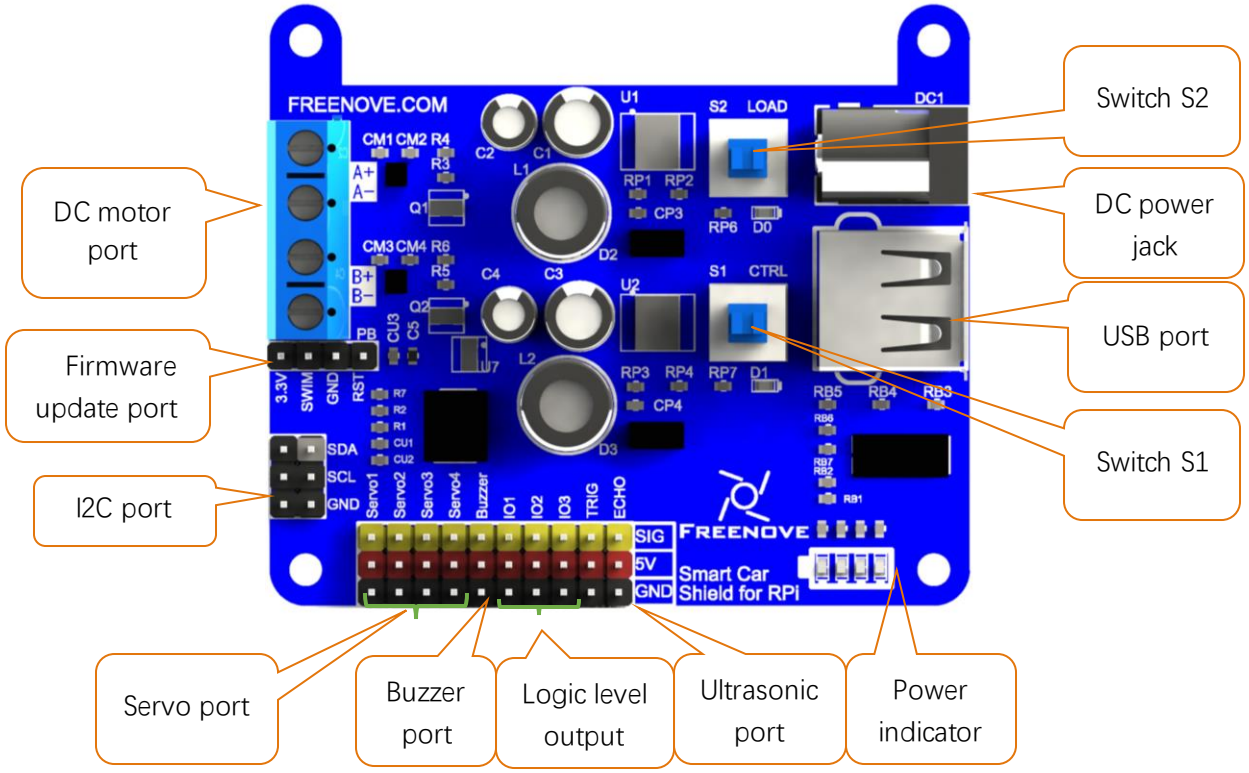
Here, you have logged in to Raspberry Pi successfully by using VNC Viewer and operated proper setting.

Wi-Fi

Raspberry Pi 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.

Smart Car Shield for RPi

Smart Car Shield for RPi (hereinafter called Shield) is designed to extend smart car control board, which is suitable for control board with host computer communication ability of I2C. The size and position of the location holes on the shield is suitable for RPi. Below is the schematic diagram and function description of the Smart Car Shield for RPi:



- DC power jack: this shield uses 5.6~11V DC power supply. If the power supply is beyond this voltage range, it may cause damage to Shield.
- There are two Voltage Regulation System for 5V/3A, which are controlled by switch S1 and S2 respectively.
  a) First 5V/3A power supply is output by the USB port on the right side of Shield, which is used to provide power for RPi. In addition, the motor driver, controller, battery voltage detector and other systems logic power on the Shield are powered by this one. It is controlled by the switch S1:CTRL for on or off. This means that if you want the Shield to work, you have to turn on the switch S1:CTRL.
  b) Second 5V/3A power is output by the red pins on Shield (marked as 5V). Motor, servo, buzzer, LED, ultrasonic and other load are powered by this one. It is controlled by the switch S2:LOAD for on or off. This means that if you want the load connected to the Shield to work, you have to turn on the switch S2:LOAD.
- Power indicator. Four LEDs are used to indicate the power. With the power consumption, LEDs will be turned off one by one.
- I2C communication port. Shield using I2C communication protocol. The default device address for I2C is 0x18, which can be changed to any address through specific command. And the data can be preserved with power off.
- Firmware update port. If there is a new released firmware, the firmware can be updated through this port.
- DC motor interface. This shield can control speed and steering of two motors. Maximum current of each

motor is 1.8A, and voltage is the input voltage of Shield.

- Servo port. This shield provides four servo ports. The servo control accuracy is 1us, which is 0.09 degrees.
- Buzzer port: This shield provides 1 buzzer port. This port can produce PWM with frequency 0-65535Hz, and duty cycle 50%.
- The logic level output: This shield provides 3 common ports which can output logic level (0V/3.3V).
- Ultrasonic port: This shield provides a SR04 ultrasonic port.
- USB power port: This shield provides a USB power port to supply power for RPi.

The communication and command about this shield will be introduced later.

Chapter 0 Software installation and Test

In this chapter, we will do some necessary preparation work: start your Pi Raspberry and install some necessary libraries.
If you are using remote desktop mode to login Raspberry Pi, you need use VNC viewer.

Step 0.1 Obtain the Code

Type the following command in the terminal to obtain the code for the smart car. And place the code in the user directory "Pi". **(Note: Here are two commands. Please execute commands in order.)**

```
cd ~

git clone http://github.com/Freenove/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi.git
```

```
pi@raspberrypi:~ $ git clone http://github.com/freenove/Freenove_Three-wheeled_S
mart_Car_Kit_for_Raspberry_Pi.git
Cloning into 'Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi'...
remote: Counting objects: 163, done.
remote: Compressing objects: 100% (124/124), done.
remote: Total 163 (delta 34), reused 162 (delta 33), pack-reused 0
Receiving objects: 100% (163/163), 674.63 KiB | 5.00 KiB/s, done.
Resolving deltas: 100% (34/34), done.
Checking connectivity... done.
```

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).

Please notice that our code for this smart car is written with Python3.

Set Python3 as default python

First, execute python to check default python on your raspberry Pi. Press Ctrl-Z to exit.

```
pi@raspberrypi:~ $ python
```

If it is python3, you can skip this section.
If it is python2, you need execute following commands to set default python to python3.

1. Enter directory /usr/bin

```
cd /usr/bin
```

2. Delete the old python link.

```
sudo rm python
```

3. Create new python links to python.

```
sudo ln -s python3 python
```

4. Check python. Press Ctrl-Z to exit.

```
python
```

```
pi@raspberrypi:/usr/bin $ sudo rm python
pi@raspberrypi:/usr/bin $ sudo ln -s python3 python
pi@raspberrypi:/usr/bin $ python
Python 3.5.3 (default, Jan 19 2017, 14:11:04)
[GCC 6.3.0 20170124] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

If you want to set python2 as default python in **other projects**.
Just repeat command above and change python3 to python2.

```
pi@raspberrypi:~ $ cd /usr/bin
pi@raspberrypi:/usr/bin $ sudo rm python
pi@raspberrypi:/usr/bin $ sudo ln -s python2 python
pi@raspberrypi:/usr/bin $ python
Python 2.7.16 (default, Apr  6 2019, 01:42:57)
[GCC 8.2.0] on linux2
Type "help", "copyright", "credits" or "license" for more
>>>
```

Shortcut Key

Now, we will introduce several shortcuts that are very **useful** and **commonly used** in terminal.

1. **up and down arrow keys**. History commands can be quickly brought back by using up and down arrow keys, which are very useful when you need to reuse certain commands.

When you need to type command, pressing "↑" will bring back the previous command, and pressing "↓" will bring back the latter command.

2. **Tab key**. The Tab key can automatically complete the command/path you want to type. When there are multiple commands/paths conforming to the already typed letter, pressing Tab key once won't have any result. And pressing Tab key again will list all the eligible options. This command/path will be directly completed when there is only one eligible option.

As shown below, under the '~'directory, enter the Documents directory with the "cd" command. After typing "cd D", press Tab key, then there is no response. Press Tab key again, then all the files/folders that begin with "D" is listed. Continue to type the character "oc", then press the Tab key, and then "Documents" is completed automatically.

```
pi@raspberrypi:~ $ cd D
Desktop/   Documents/ Downloads/
pi@raspberrypi:~ $ cd Doc
```

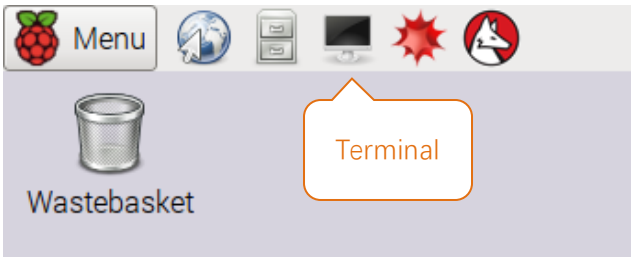
```
pi@raspberrypi:~ $ cd D
Desktop/   Documents/ Downloads/
pi@raspberrypi:~ $ cd Documents/
```

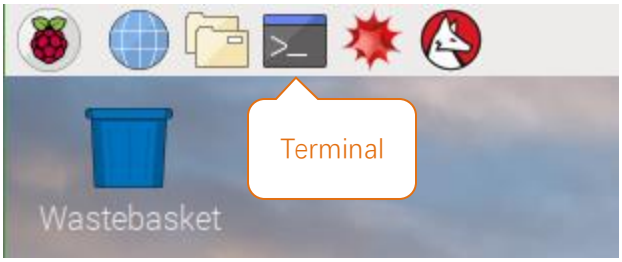
Step 0.2 Configure I2C

Enable I2C

The I2C interface raspberry pi is closed in default. You need to open it manually. You can enable the I2C interface in the following way.

open the terminal:

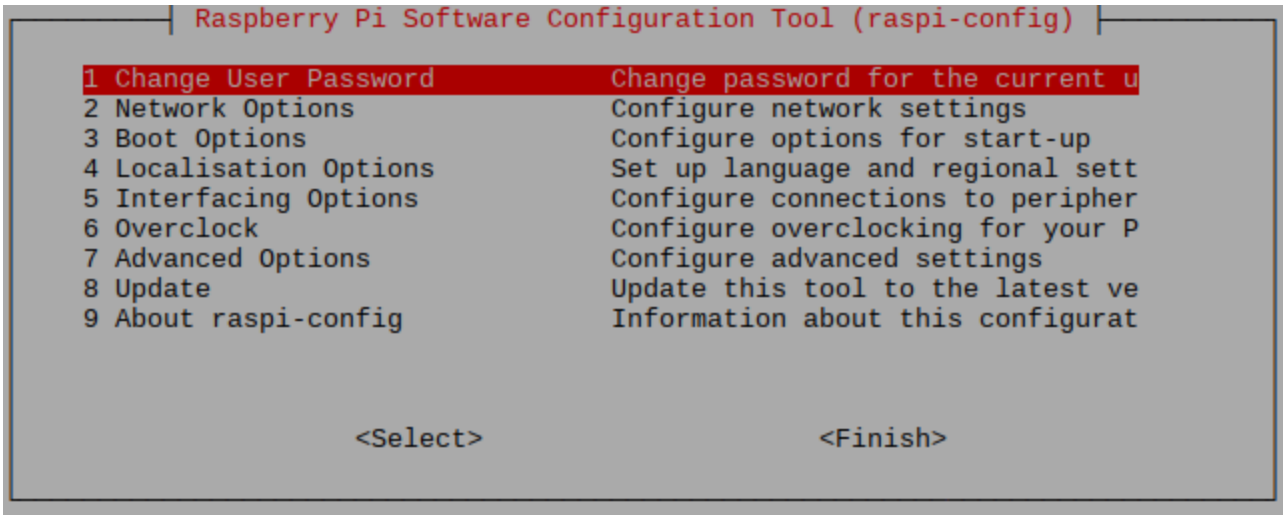




Type command in the terminal:

```
sudo raspi-config
```

Then open the following dialog box:



Choose “5 Interfacing Options”→“P5 I2C”→“Yes”→“Finish” in order and restart your RPi later. Then the I2C module is started.

Type a command to check whether the I2C module is started:

```
lsmod | grep i2c
```

If the I2C module has been started, the following content will be shown:

```
pi@raspberrypi:~ $ lsmod | grep i2c
i2c_bcm2708             4770  0
i2c_dev                 5859  0
pi@raspberrypi:~ $
```

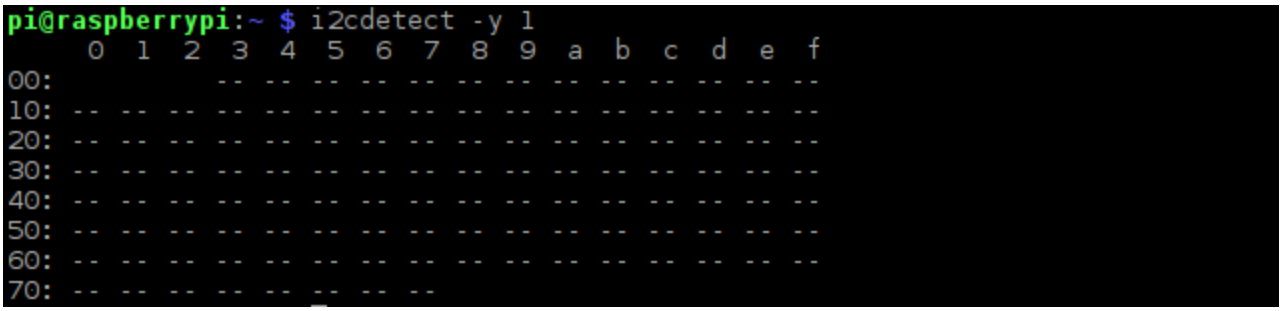
Install I2C-Tools

Type the command to install I2C-Tools.

```
sudo apt-get install i2c-tools
```

I2C device address detection:

```
i2cdetect –y 1
```



If there are I2C devices connected to your RPi, here will display their I2C device address.

Install python-smbus

Python-smbus is a module of the program Python, which contains some classes and methods to operate I2C.

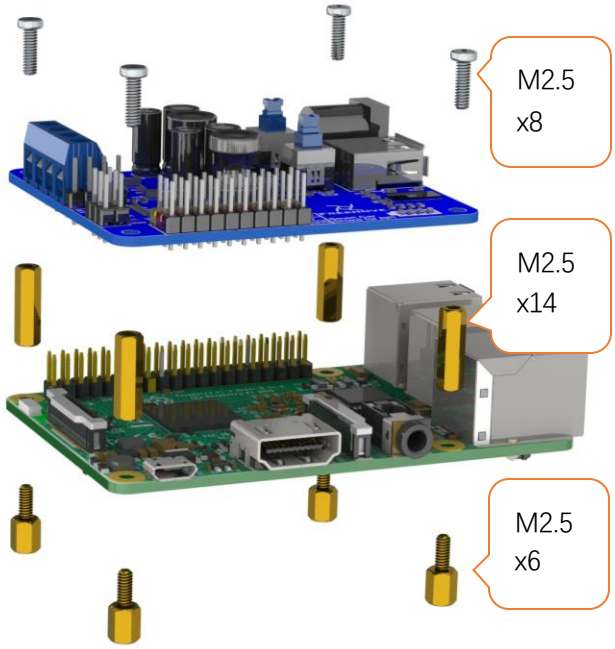
Type the following command to install python-smbus:

```
sudo apt-get install python-smbus
```

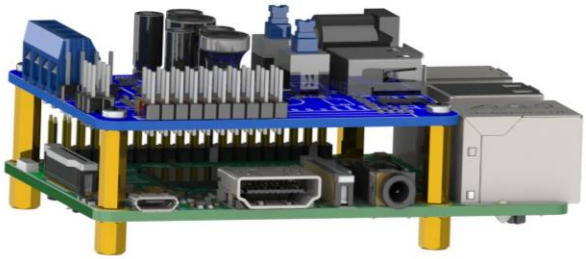
Communication test

Follow the steps below to connect the Shield with the RPi.

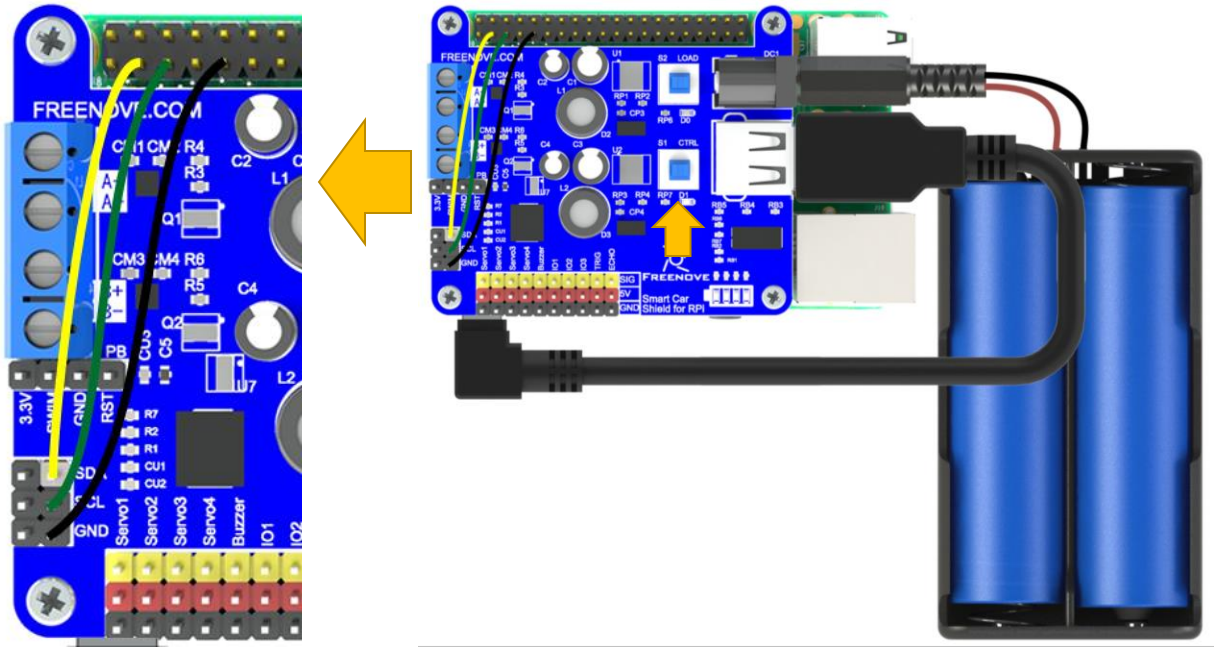
1.Prepare the following boards and parts.



2. Assembly



3.Use Jumper Wire F-F to connect the Shield with I2C port of RPi. Use the battery box to supply power for the Shield, and open the switch S1. RPi can be powered by USB power port, or external power supply adapter.



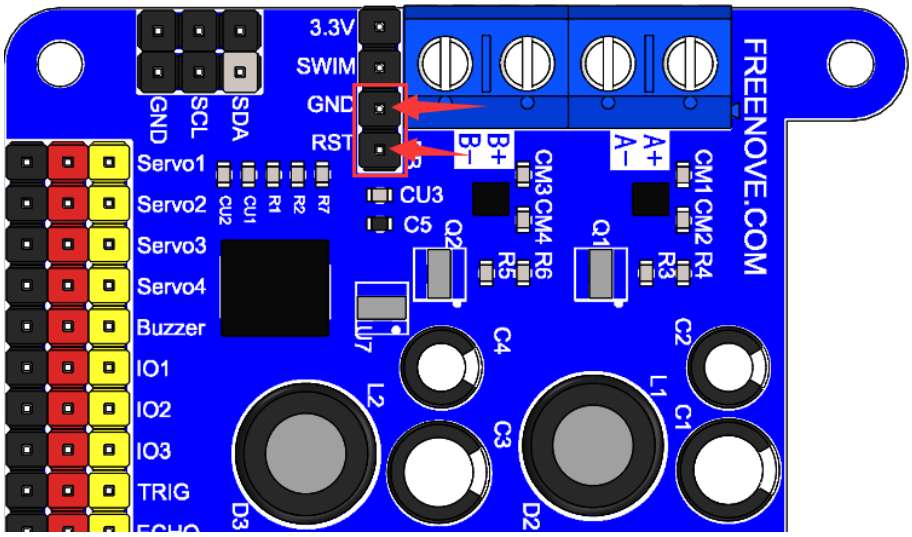
Default I2C address of the Shield is 0x18. Execute command i2cdetect -y1 again to detect whether the shield is connected to RPi successfully.

```
i2cdetect –y 1
```

```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- 18 -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

If you cannot detect i2c. Please check to S1 and S2 is press, Or try following method to reset.

Keep the power on and connect GND and RST with one F/F jumper. Then disconnect GND and RST. Then detect i2c again to try.



If it does not work, please contact us at support@freenove.com

Step 0.3 Install mjpg-streamer

Camera is drived by mjpg-streamer. So you need to install mjpg-streamer.

Install

Open the terminal and execute the following command to install.

1. Install the relay for mjpg-streamer:

```
sudo apt-get install libv4l-dev libjpeg8-dev imagemagick
```

2. You may also need to install SVN if it is not installed in your raspberry pi。

```
sudo apt-get install subversion
```

3. Generate executable file mjpg-streamer: **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/mjpg-streamer
make USE_LIBV4L2=true clean all
```

Test mjpg-streamer

Connect the camera to any one of the USB ports on the RPi. And execute the following command to verify that the camera is successfully connected to RPi.

```
ls /dev/video*
```

If the results list the video0, the camera is connected successfully.

```
pi@raspberrypi:~ $ ls /dev/video*
/dev/video0
```

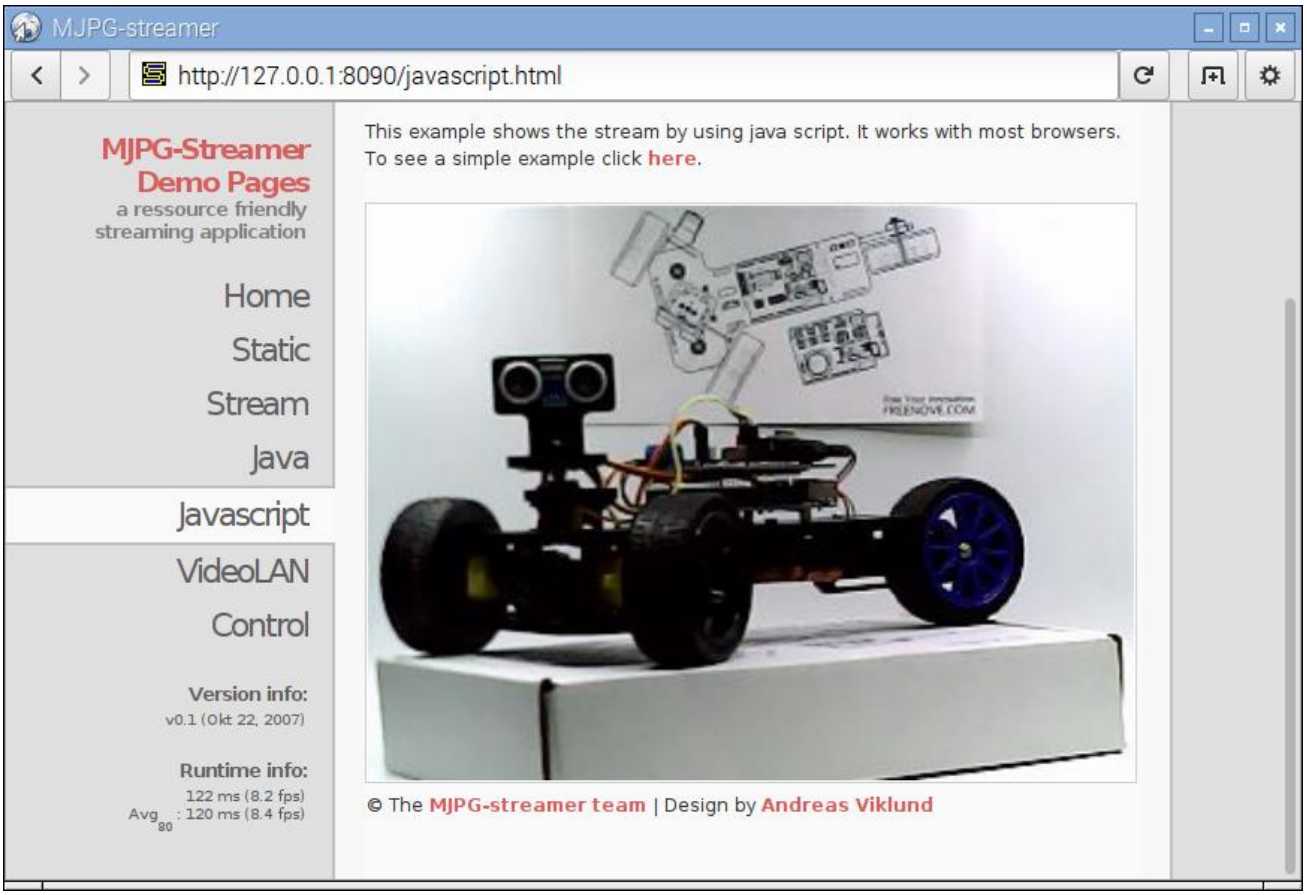
Under the mjpg-streamer directory, execute the following command to open the mjpg-streamer service.

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/mjpg-streamer
sh Start_mjpg_Streamer.sh
```

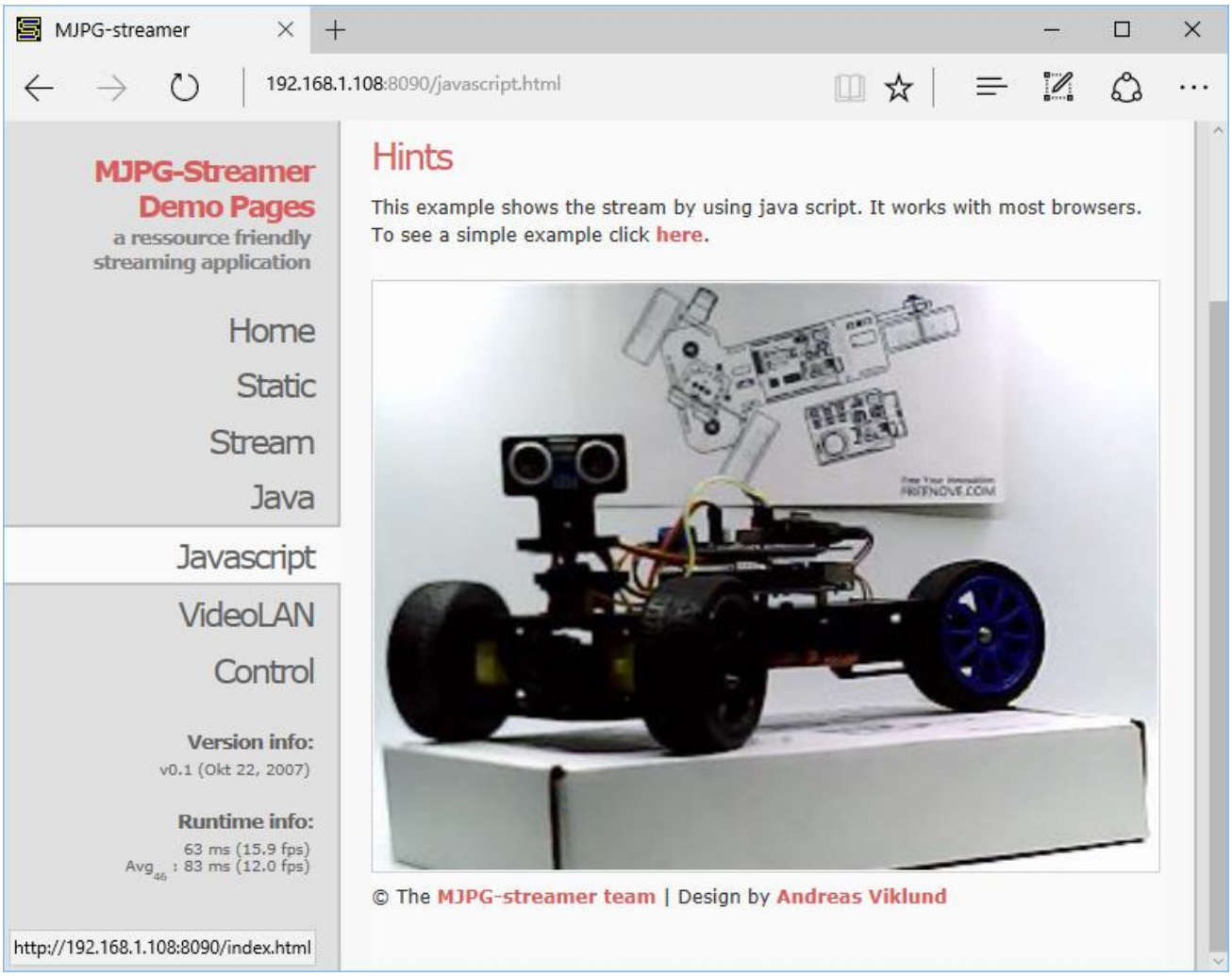
```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/mjpg-stre
amer $ sh Start_mjpg_Streamer.sh
MJPG Streamer Version: svn rev: Unversioned directory
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 320 x 240
 i: Frames Per Second.: 30
 i: Format............: YUV
 i: JPEG Quality......: 80
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8090
 o: username:password.: disabled
 o: commands.........: enabled
```

Open Web browser of RPi, access to http://127.0.0.1:8090/ or http://localhost:8090/. Then the following picture appears.

Click the "Javascript" tab on the left of navigation bar, then you will be see the real-time picture of the camera. If the picture is not clear enough, you can rotate the lens to adjust the focus.



You can access this page through accessing http://xxx.xxx.xxx.xxx:8090/ with your Web browser of your PC or mobile phone. This requires your PC or mobile phone to be in the same local area network with your RPi, where xxx.xxx.xxx.xxx is IP address of RPi. For example, my RPi IP address is 192.168.1.108. In the Windows 10, access to http://192.168.1.108:8090/ through the browser, as is shown below.



Step 0.4 Install PyQt5

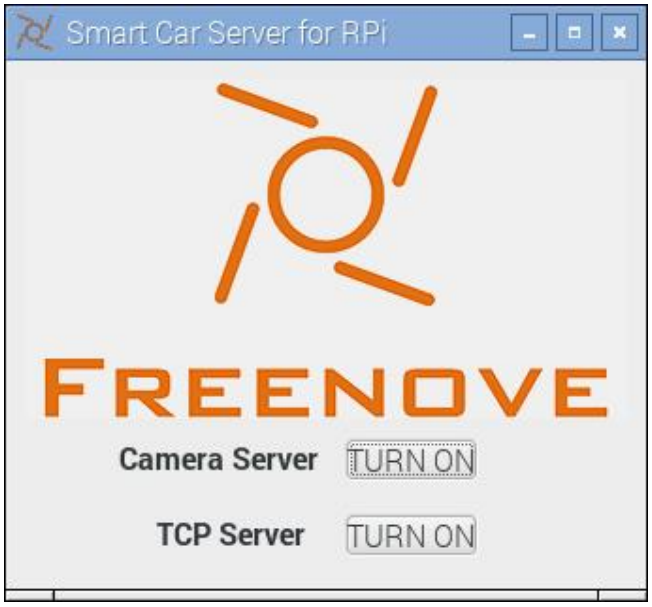
The project code is based on PyQt5. So operation of the program requires the support of PyQt5.
Open the terminal and execute the following command to install PyQt5. **(Note: Here are four commands. Please excute commands in order.)**

```
sudo apt-get update

sudo apt-get install python3-pyqt5

sudo apt-get install python3-pyqt5.qtwebkit

sudo apt-get install python3-dev
```

After the installation completed, type the following command to test whether PyQt5 is installed successfully.
(Note: Here are two commands. Please excute commands in order.)

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python Main.py
```

If you can run it successfully, and the following picture appears, it means that PyQt5 has been successfully installed. Then click on the top right corner to close the program.



About how to open the client in raspberry pi, please refer to the document "Patch for Stretch. pdf" under the unzipped folder.

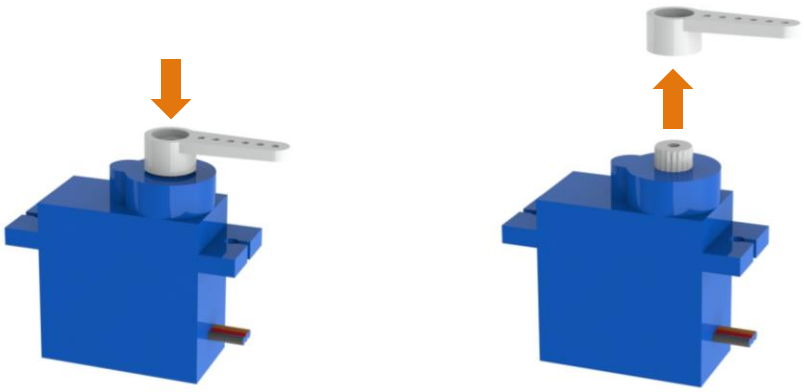
Step 0.5 Test

Next, test the servo, motor, buzzer, RGBLED module and so on.

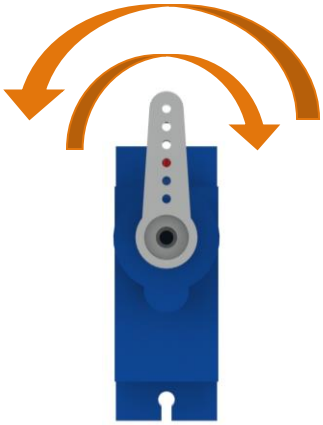
Servo

The servo can be connected with rocker arm to drive other parts to move. There are 3 kinds of rocker arm, and 3 screws for the servo. The smaller screw is used to fix the rocker arm onto servo.

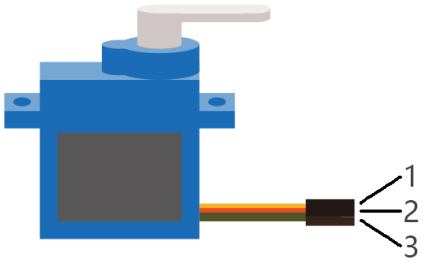
You can install or remove the rocker arm as below.
Don't install the screw first.

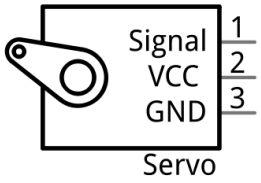


You can turn the rocker arm to rotate in the range from 0 to 180 degrees with hand:

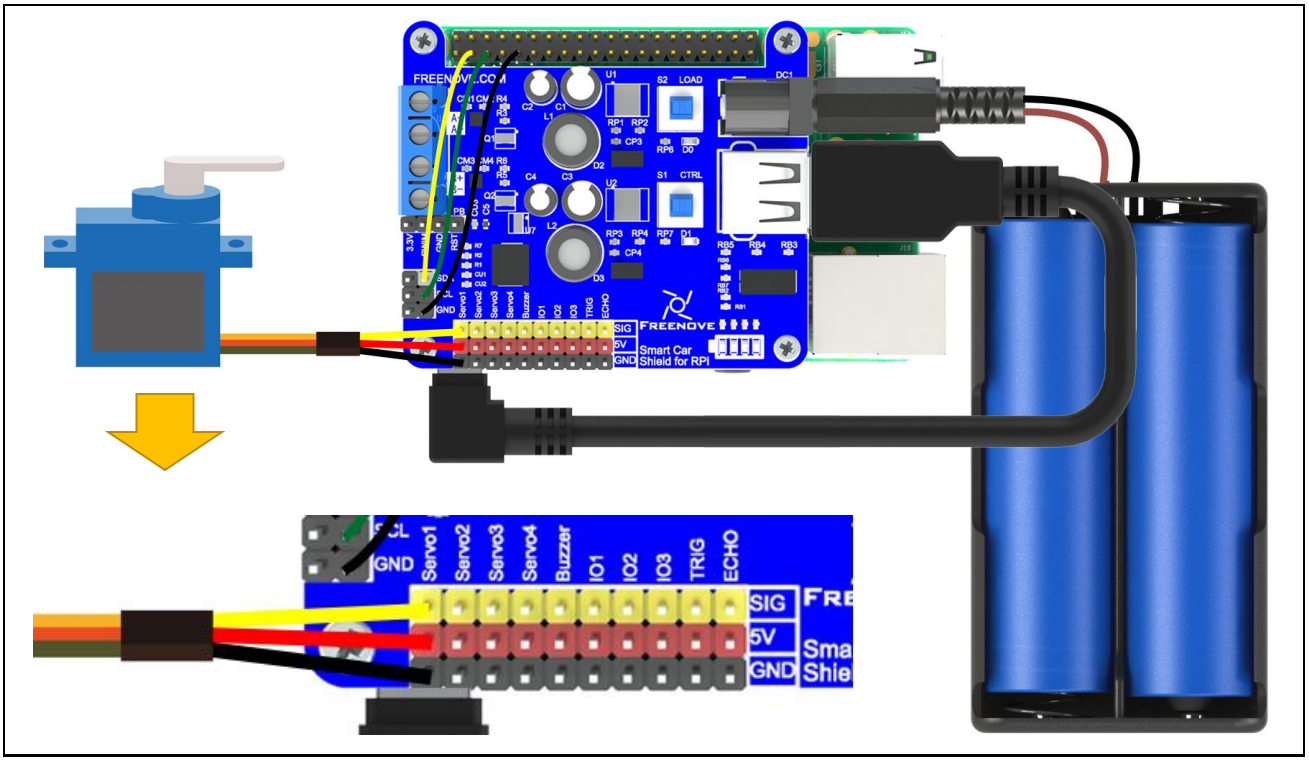


Servo has three lines for controlling. The orange one is for the signal line, the red line for the power VCC, the black one for the power GND.





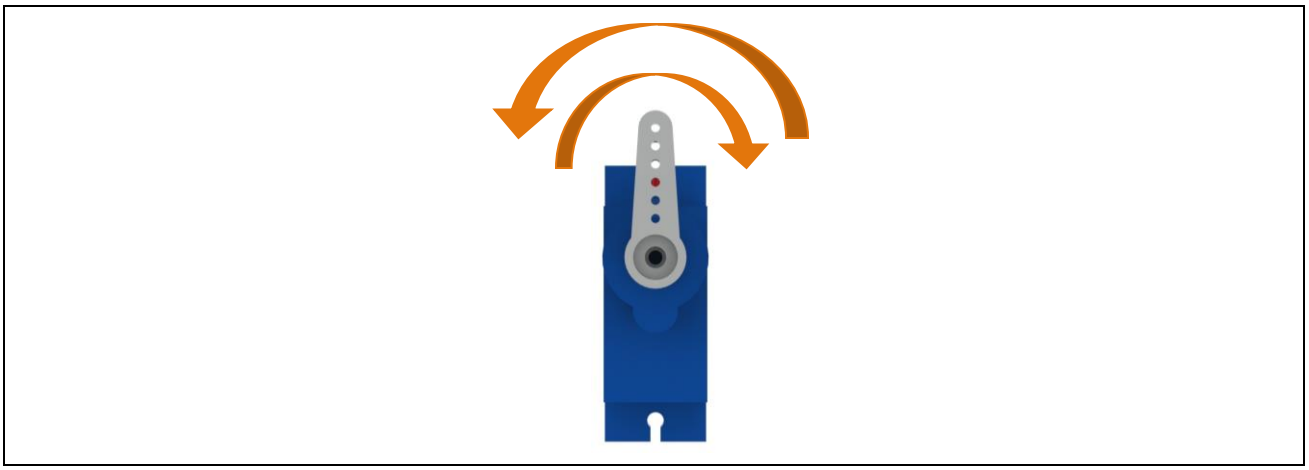
According to the following steps, connect any one of servos to the Servo1 port of the Shield. Open the switch S2, then the servo will rotate to 90 degrees automatically.



Type the following command in the terminal to test the servo: **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python mDev.py servo
```

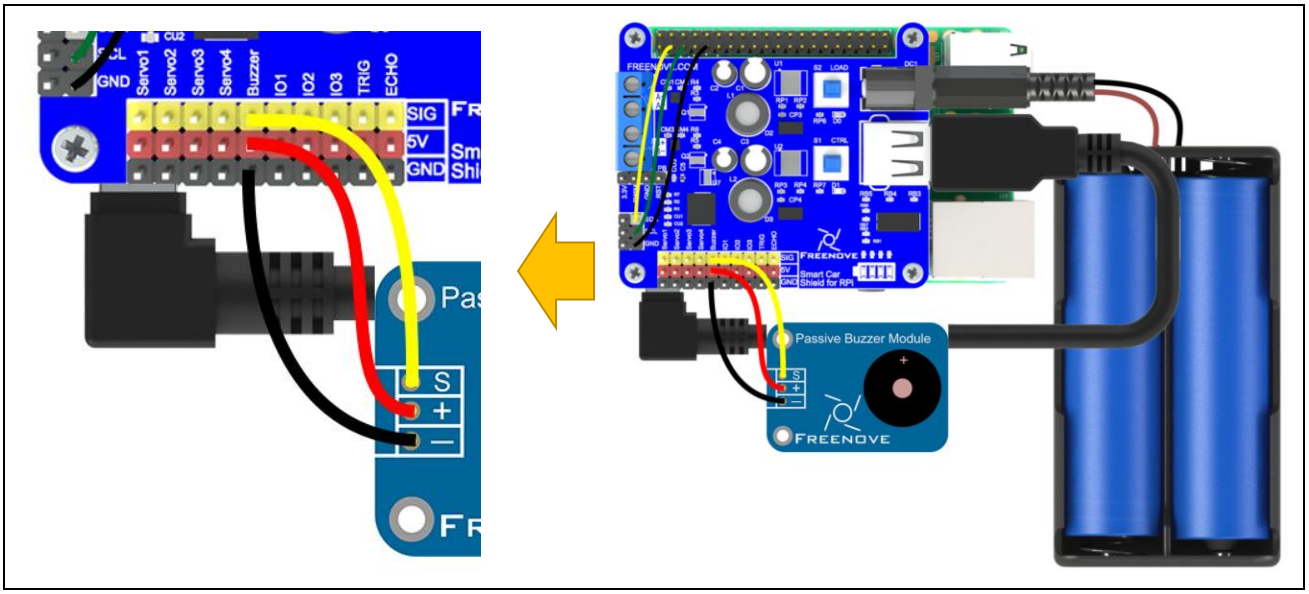
Then, the servo will rotate back and forth in a certain range.



After the test is completed. The servo will stop at 90 degrees. Then remove the rocker arm, and then pull off the servo line. After that, do not rotate the servo manually not to affect the following installation. Make all other servos to rotate to 90 degrees according to this method.

Buzzer

Connect the buzzer module to the buzzer port in the Shield.

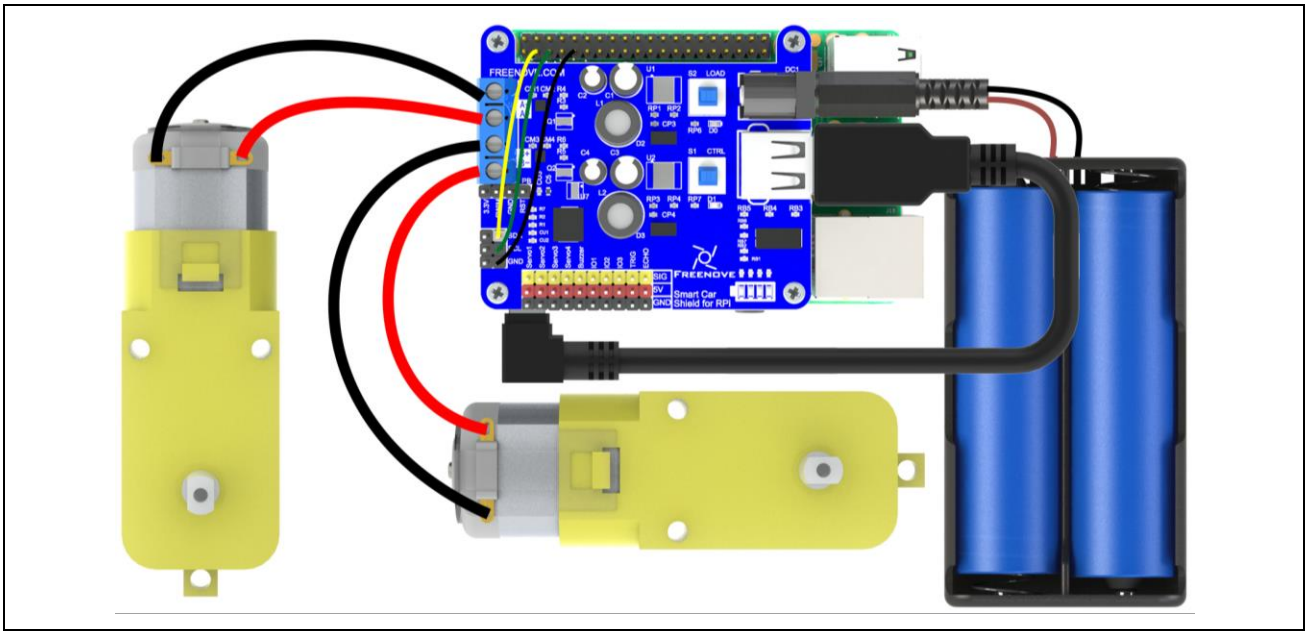


Type the following command. Then the buzzer will start to sound. The program will exit 3 seconds later, then the buzzer stop sounding. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python mDev.py buzzer
```

Motor

Connect the motor module to the motor port in the Shield.

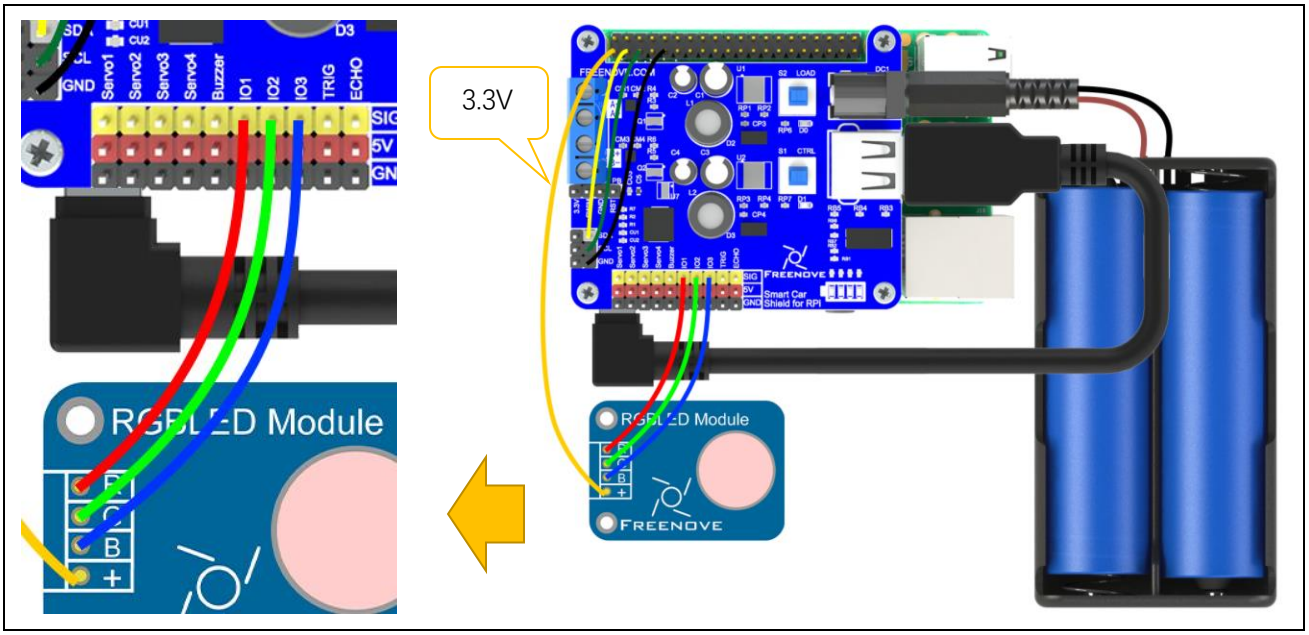


Type the following command, then the motor will rotate back and forth. Then the program will be terminated and the motor will stop running.. **(Note: Here are two commands. Please excute commands in order.)**
If the path is already ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server, you don't need to enter the first command.

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python mDev.py motor
```

RGBLED Module

Connect pin R, G, B of RGBLEG Module to port IO1, IO2, IO3 of the Shield respectively. Connect "+" to 3.3V of RPi.

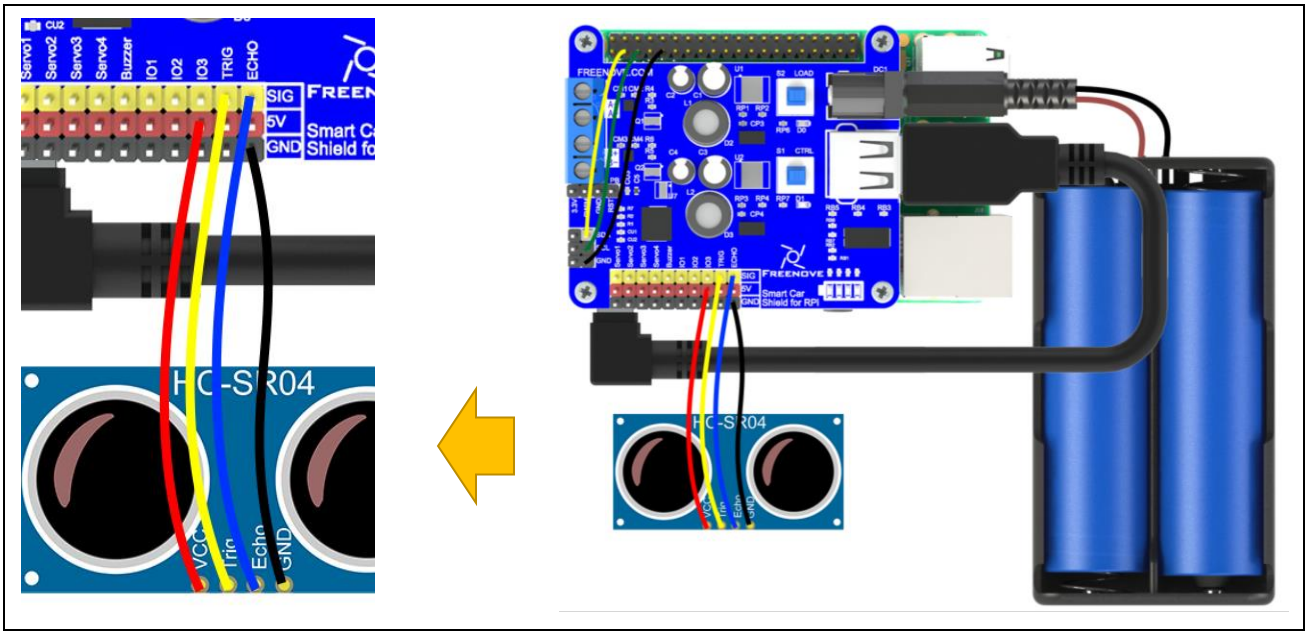


Type the following command, RGBLED will emit red, green, and blue light several times circularly. Then the program exits, the RGBLED is turned off. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python mDev.py RGBLED
```

Ultrasonic Module

Connect pin VCC, GND, TRIG, ECHO of Ultrasonic Module to port 5V, GND, TRIG, ECHO of the Shield respectively:



Type the following command to print out the ultrasonic measurement data in the terminal. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python mDev.py ultrasonic
```

Step 0.6 Client

The entire project code contains two parts: the server and the client. The server that runs on RPi will be introduced later.

The client runs on other devices, and of course, it can run on the same RPi with the server. The device that the client runs on requires the installation of Python3 and PyQt5. If you need to run the client on Linux system, you need install PyQt5 according to Step 0.4. If you want to run the client on windows, you need install the program and service according to the following steps.

Run the client on Windows OS

Install python3

The client can only work under Python before Python 3.7.4 now, because some libraries don't support latest Python now. So it is recommended to install Python 3.7.4 to run the client.

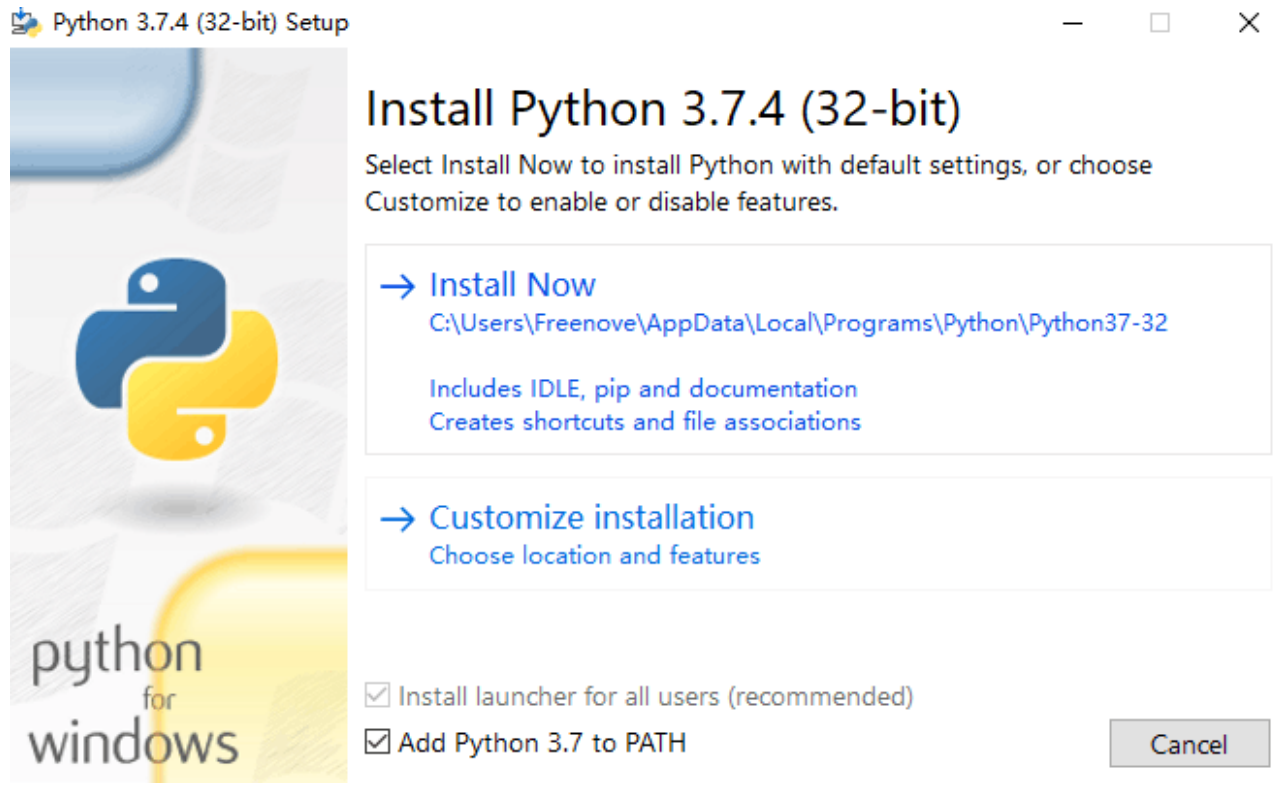
Download the installation file via link below:
https://www.python.org/downloads/release/python-374/

On the page bottom, choose "Windows x86 executable installer"

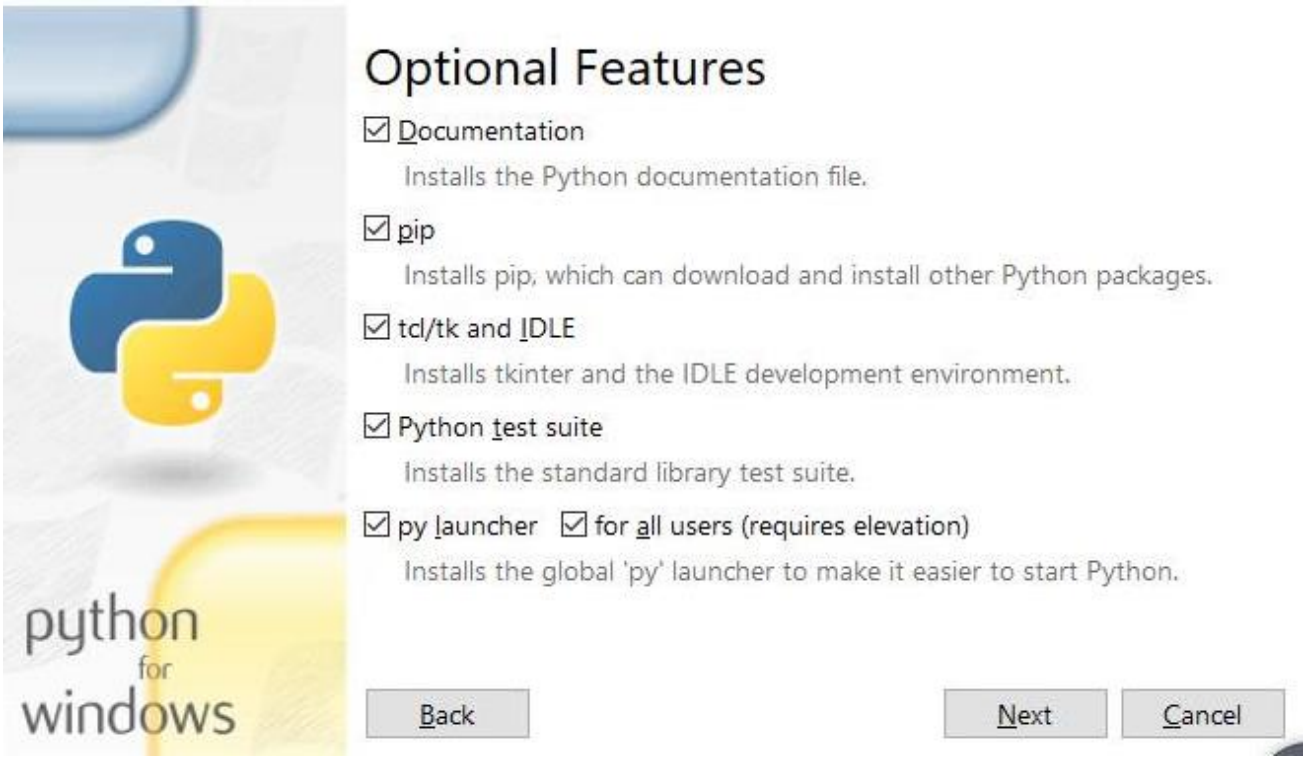
Files

Version	Operating System	Description	MD5 Sum	File Size
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda9413357413 9d8	6741626
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608

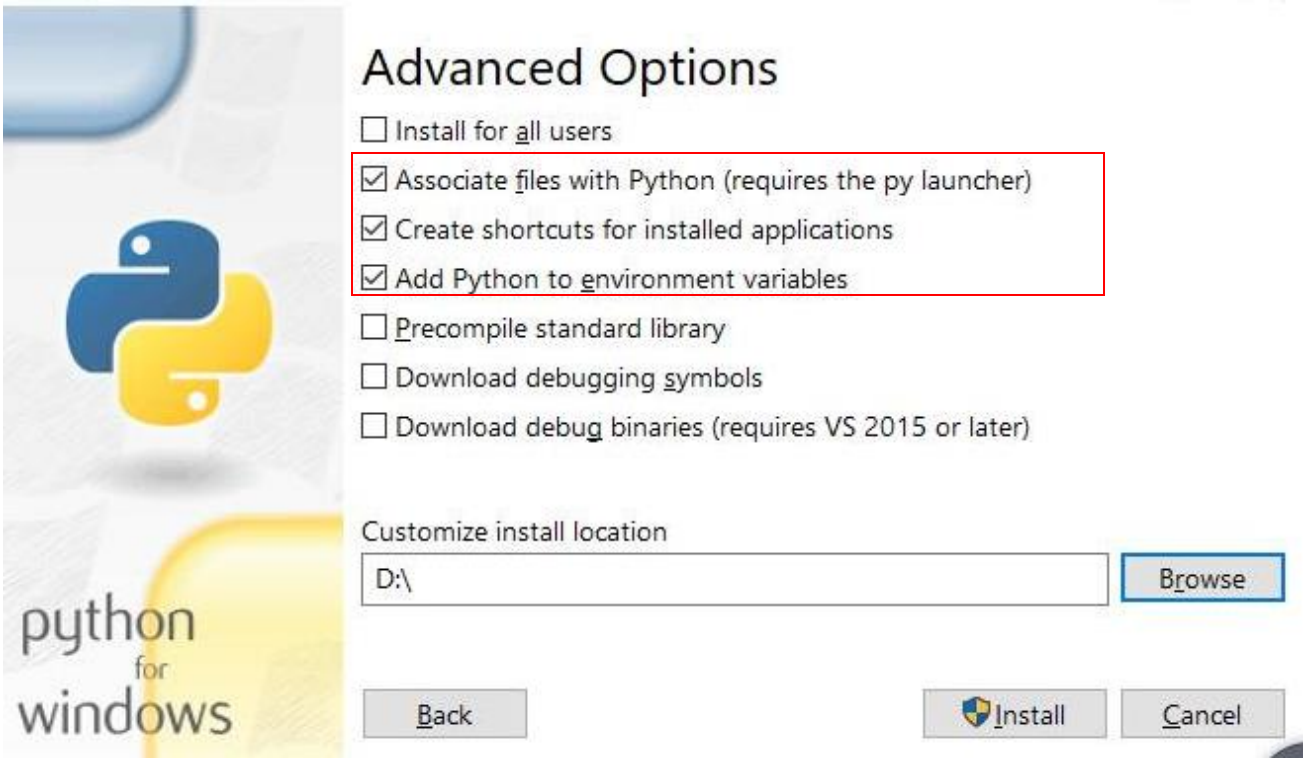
Choose "Windows x86 executable installer" to download and install.



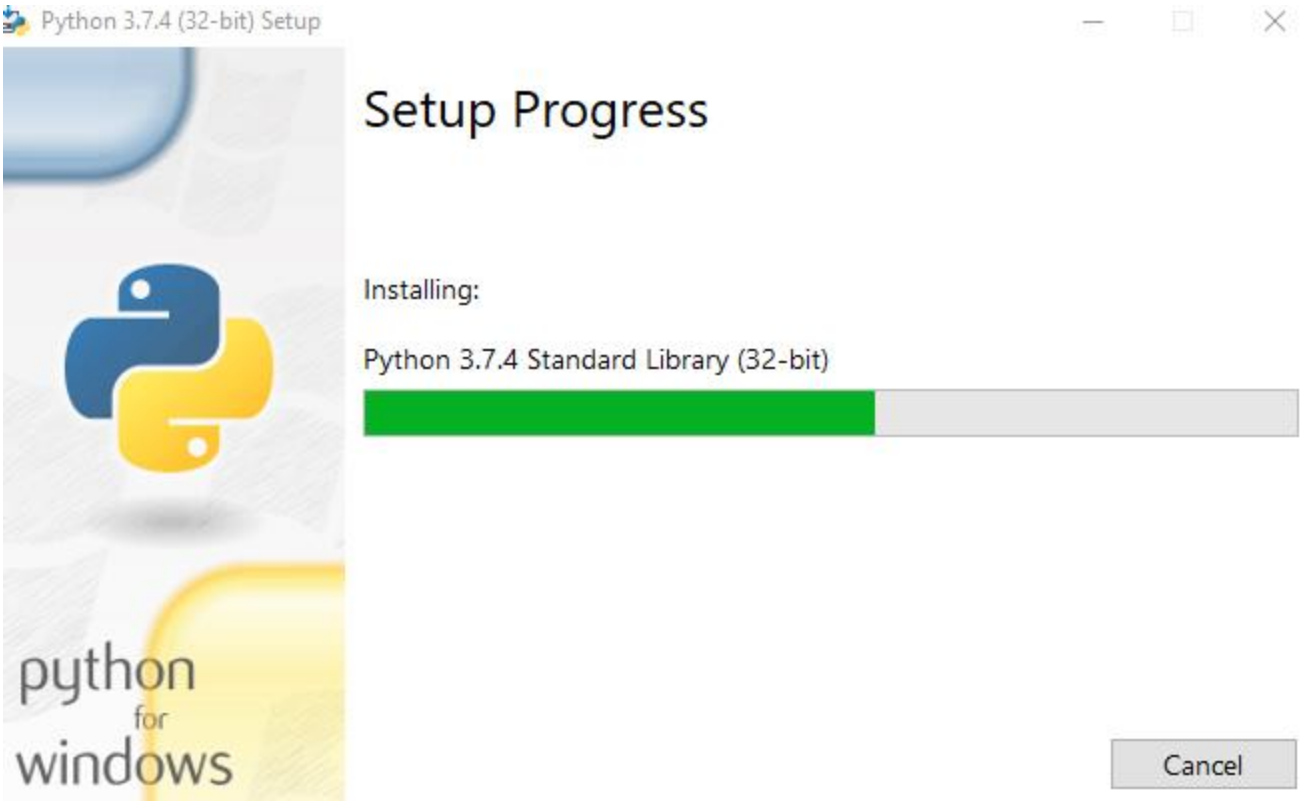
Select "Add Python 3.7 to PATH". And choose Customize installation.



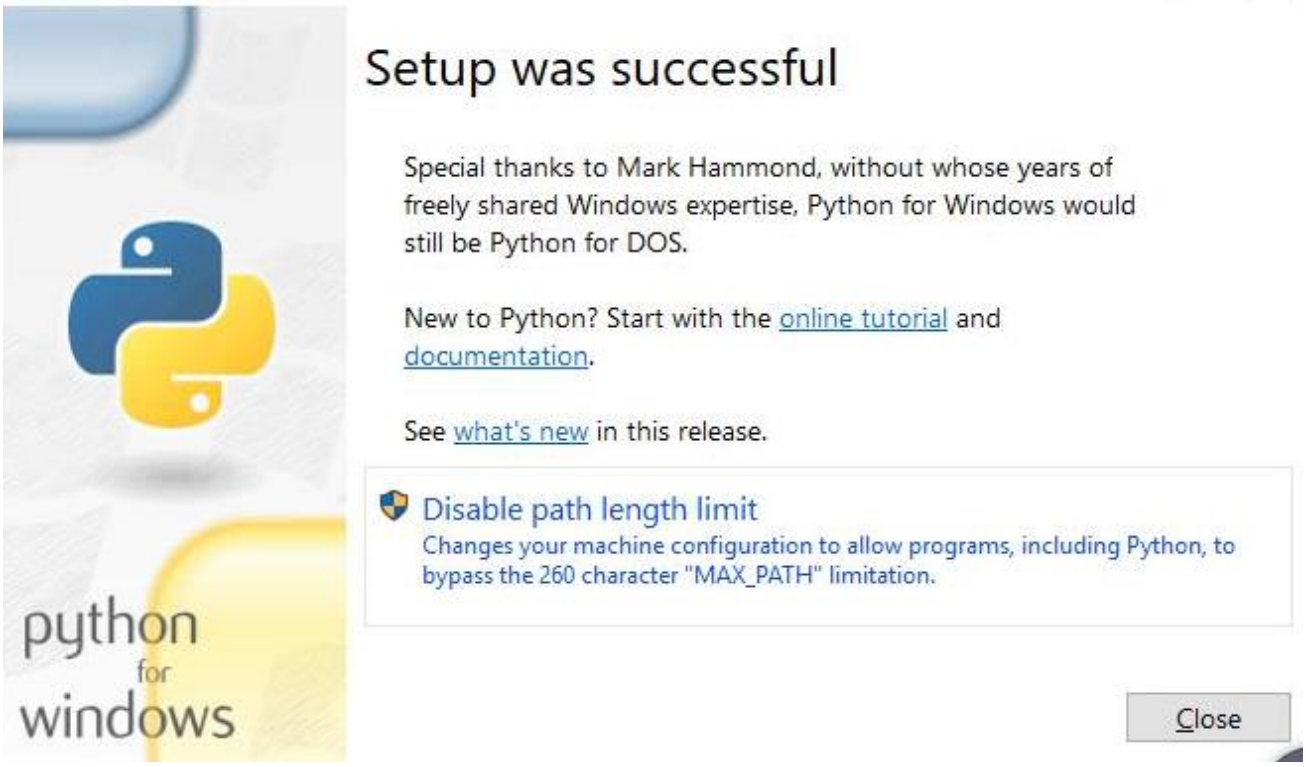
Select all options and click Next.



Here python is installed into D disk as an example (You can choose your own installation path). Click Install.



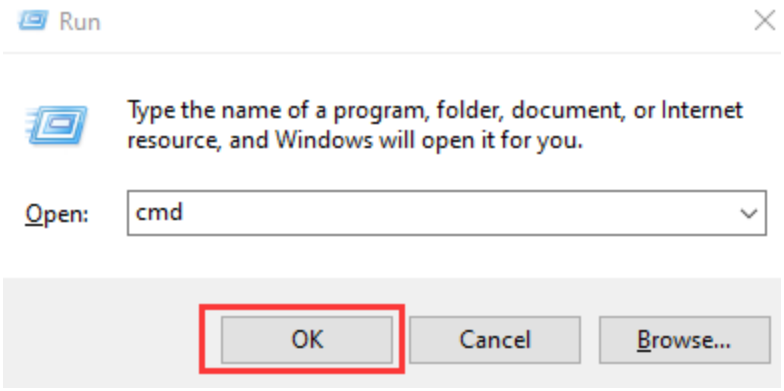
Wait for installation.



Now python installation is successful.

Install PyQt5

Press "Win+R" to open Run and type cmd to open windows terminal. Type following commands one by one, four in total.




```
python -m pip install --upgrade pip

pip3 install PyQt5==5.13.2

pip3 install PyQtWebKit

pip3 install pyqt5-tools
```

After installation is successful, type command below to check.

```
pip3 list
```

Android 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).

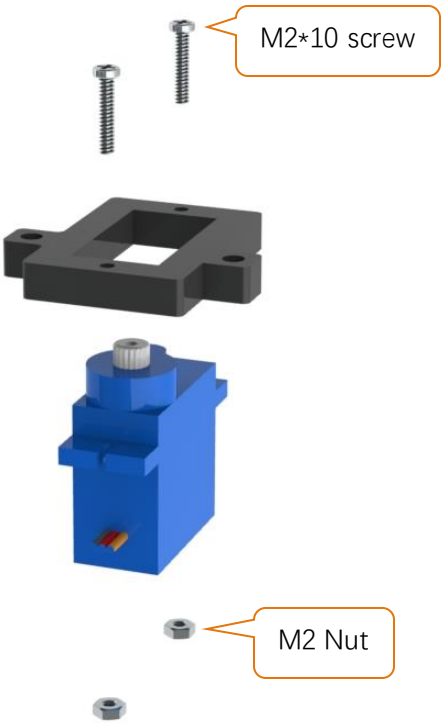
Chapter 1 Smart video car

First, let's make a smart video car.

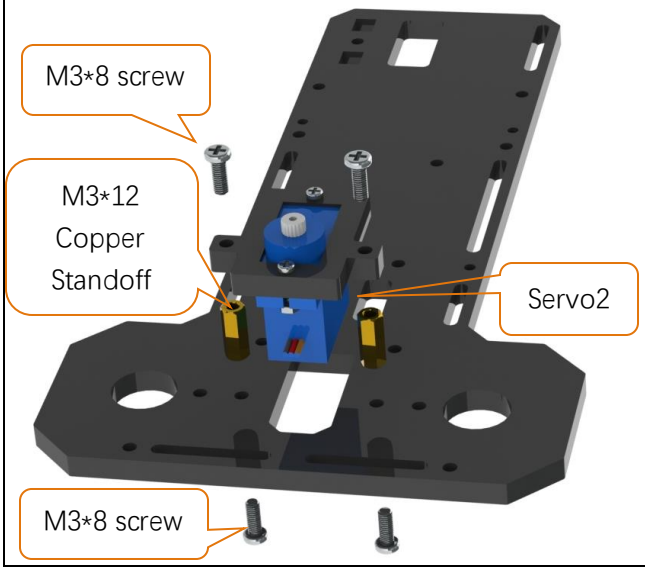
Assembly

Pan-tilt

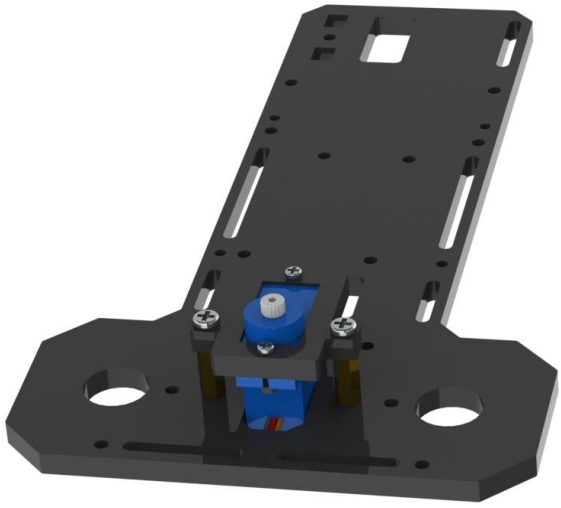
Assemble servo2

1.Assemble the following components	After assembling
	

2.Assemble the following components

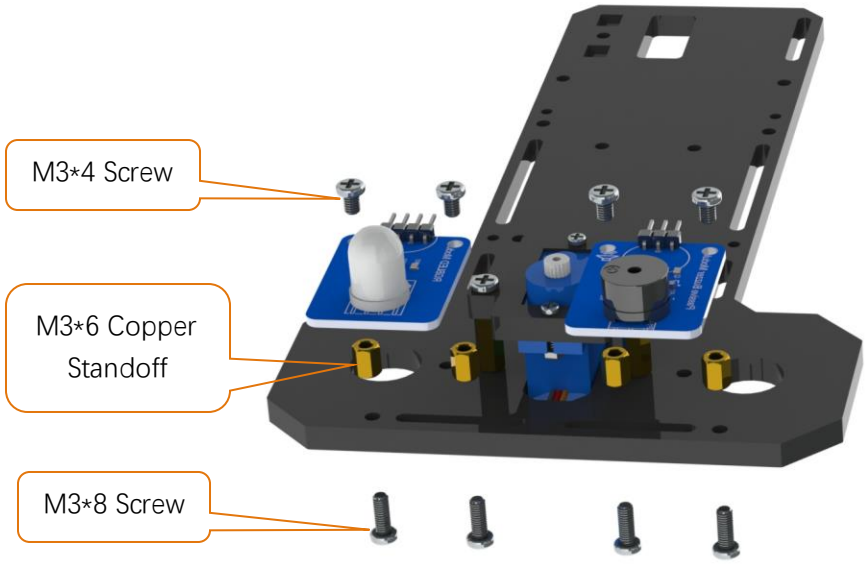


After assembling

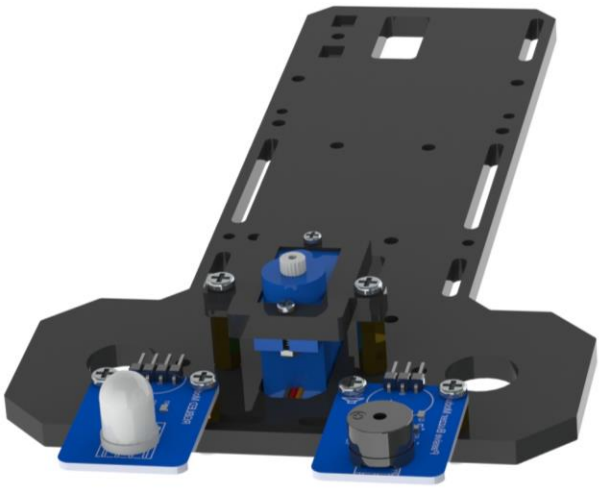


Assemble RGBLED Module and Passive Buzzer Module

Assemble the following components

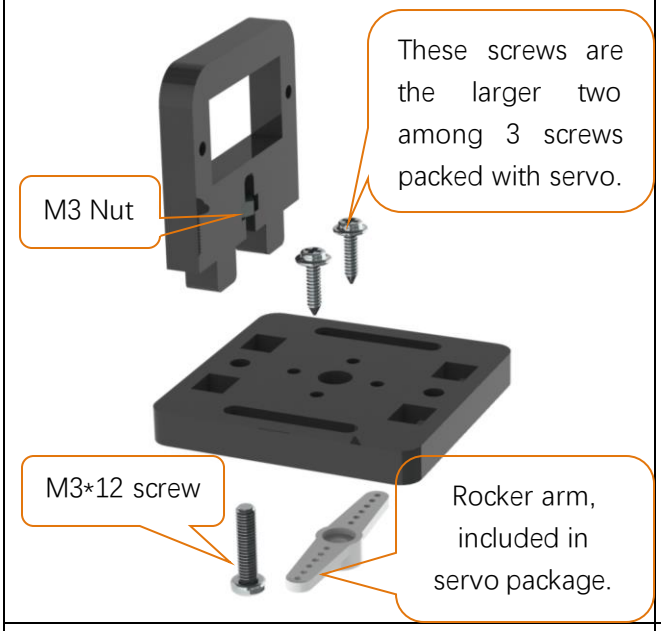


After assembling



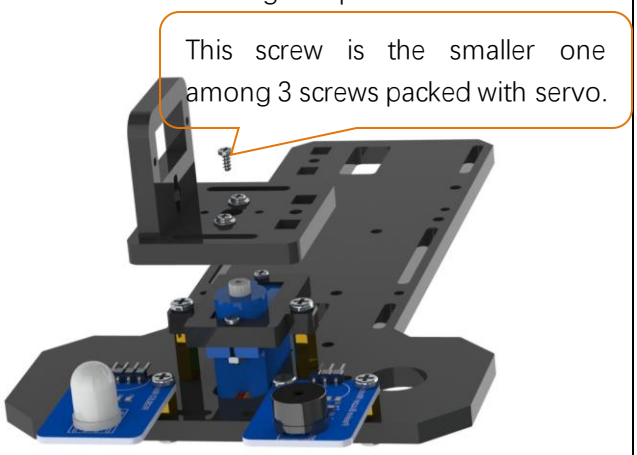
Assemble servo3 support

Assemble the following components

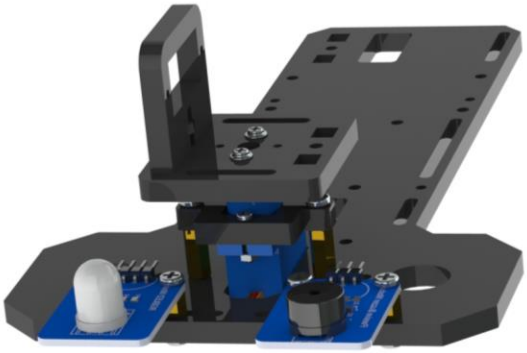


After assembling

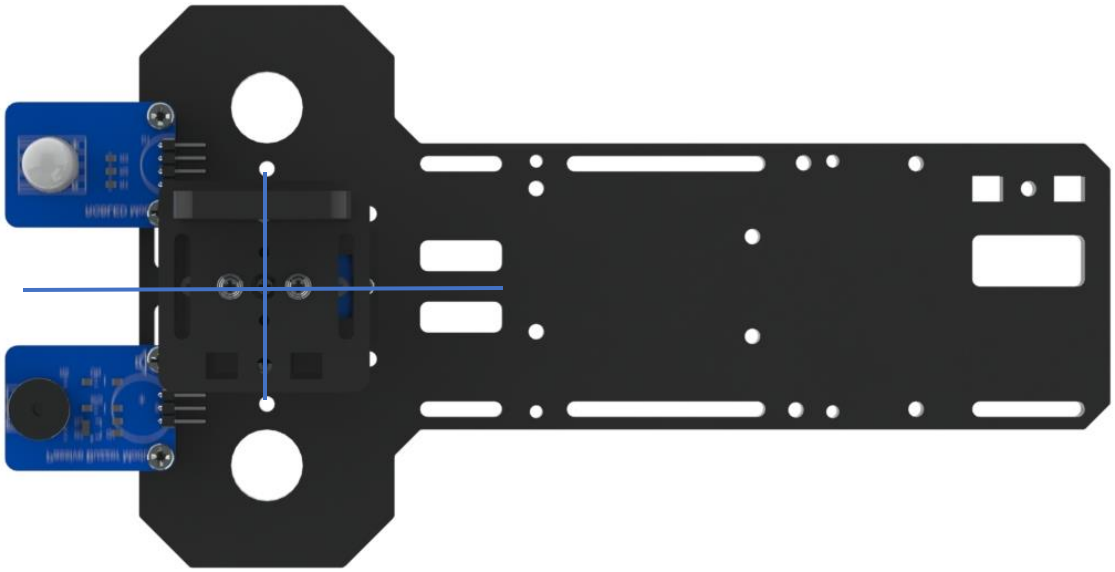
Assemble the following components



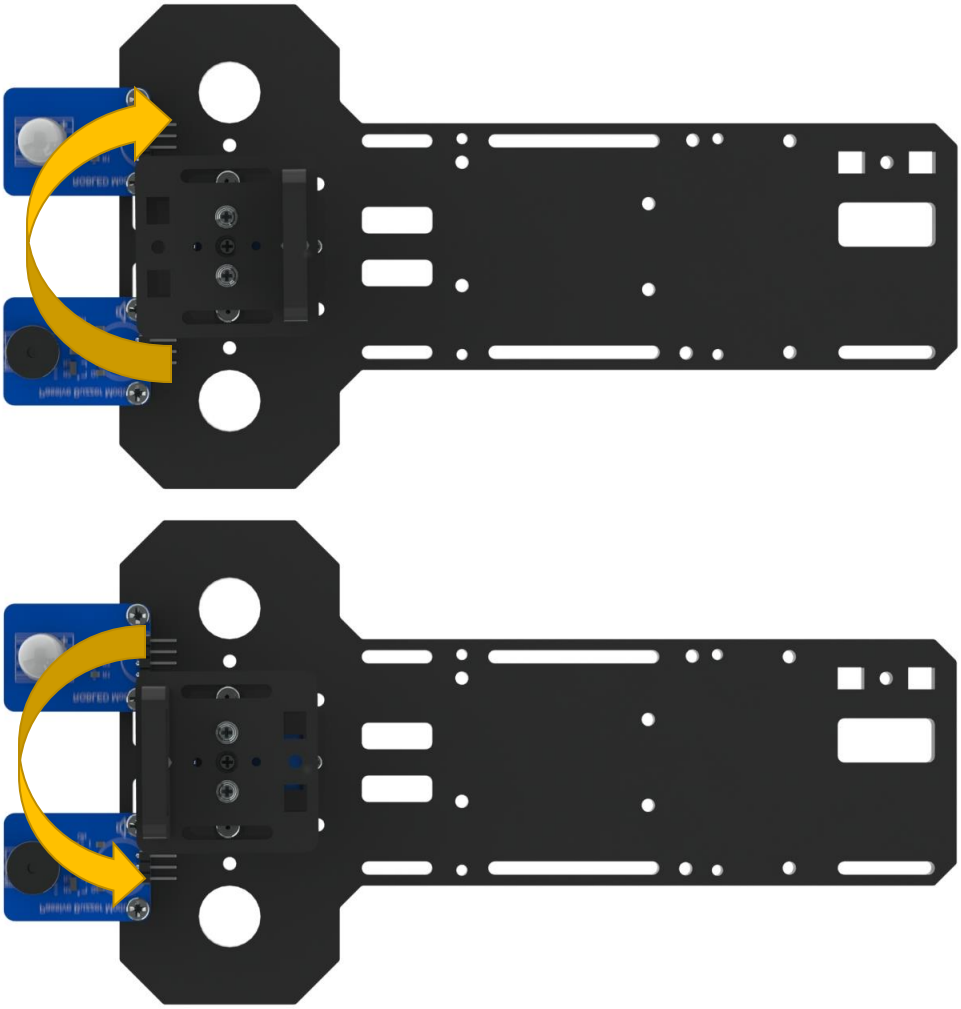
After assembling



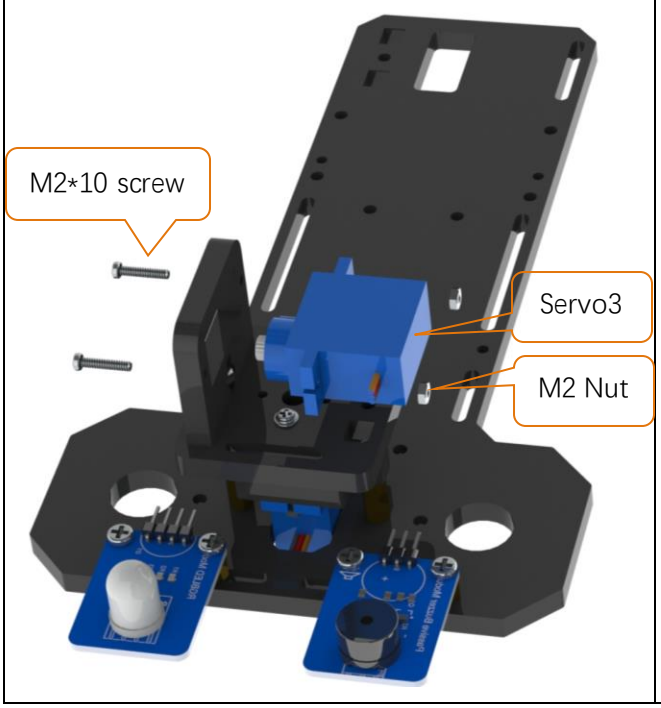
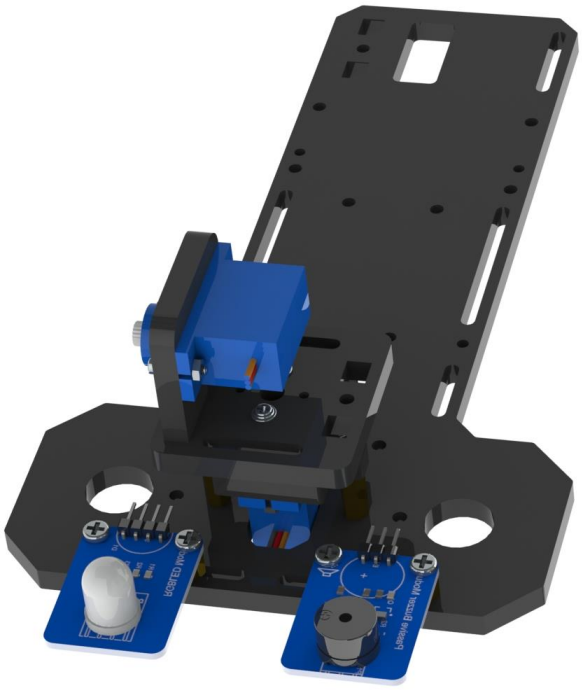
Keep the servo2 rotating to 90 degrees. If the drection is changed before installation, please make it rotate to the 90 degrees with the previous method. Complete the installation according to the direction below.



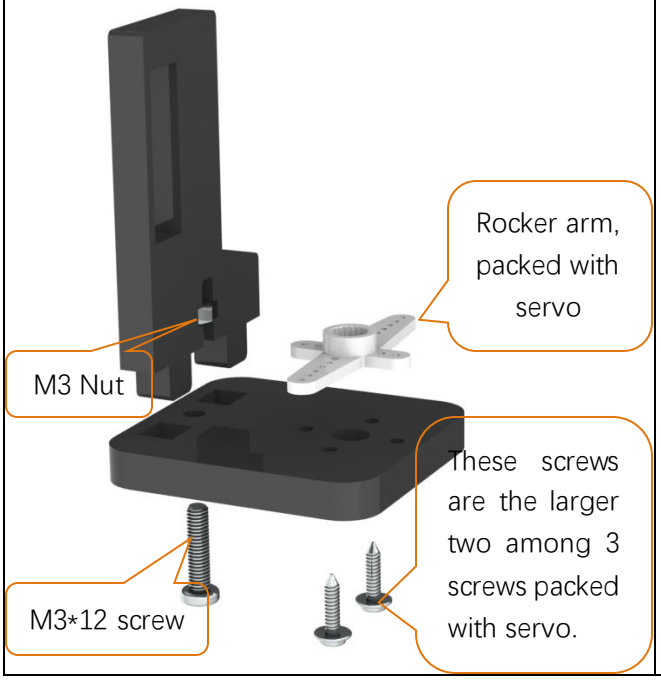
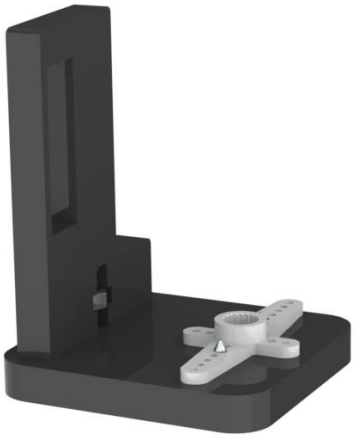
After correct assembly, the bracket can rotate 90 degrees to left or right.



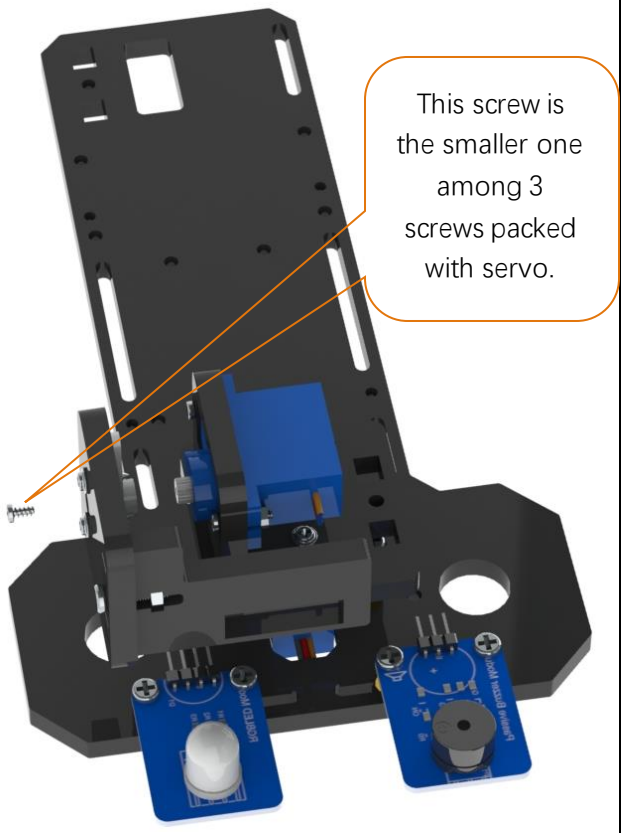
Assemble servo3

Assemble the following components	After assembling
	

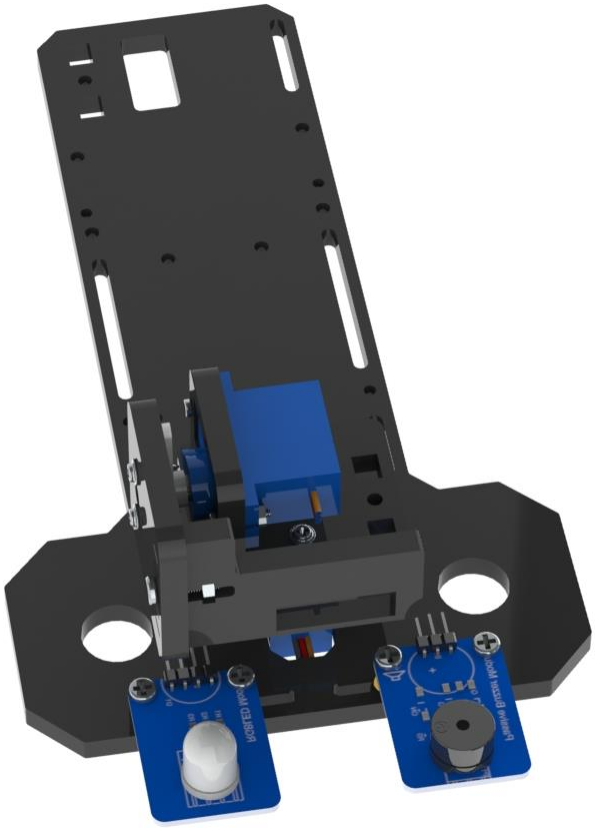
Assemble camera support

Assemble the following components	After assembling
	

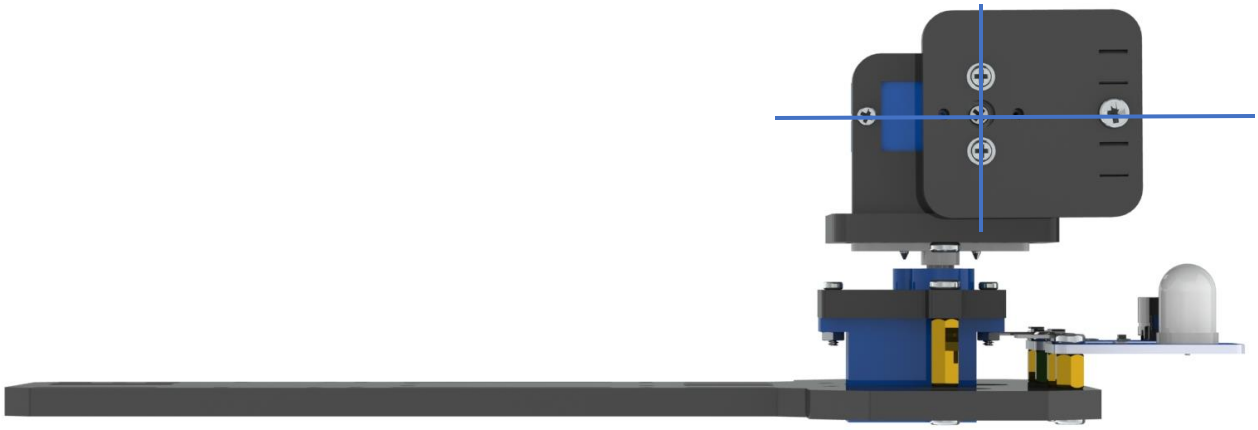
Assemble the following components



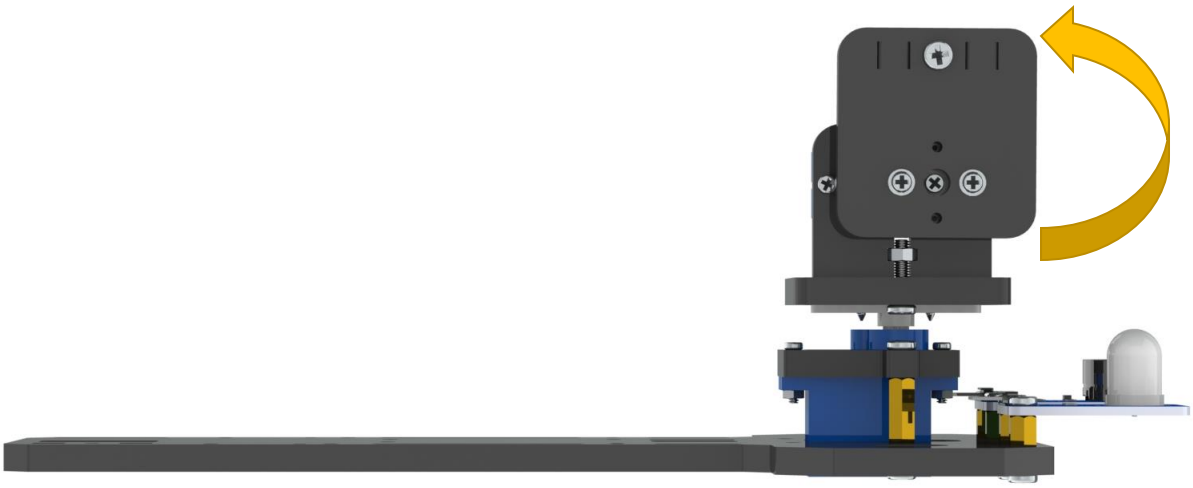
After assembling



Keep the servo3 rotating to 90 degrees. If the drection is changed before installation, please make it rotate to the 90 degrees with the previous method. Complete the installation according to the direction below.



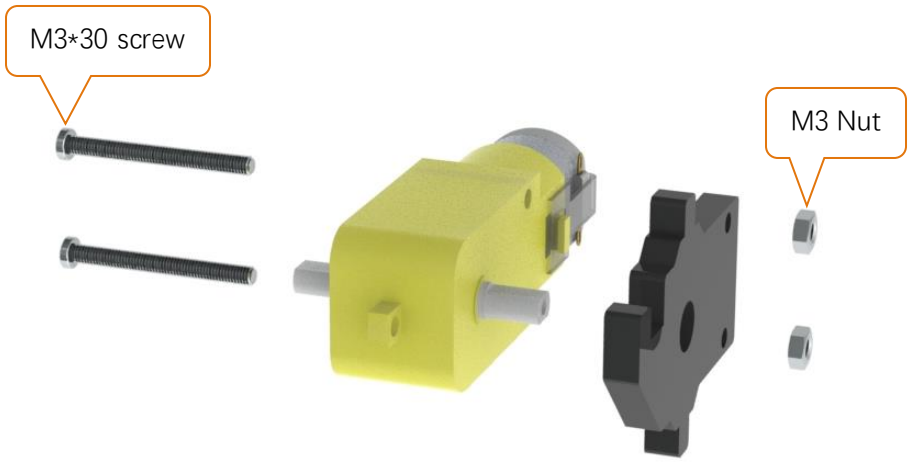
After the correct assembly, the support can rotate up to 90 degrees, as shown below.



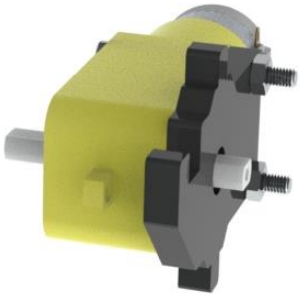
Car body

Driving wheel and motor

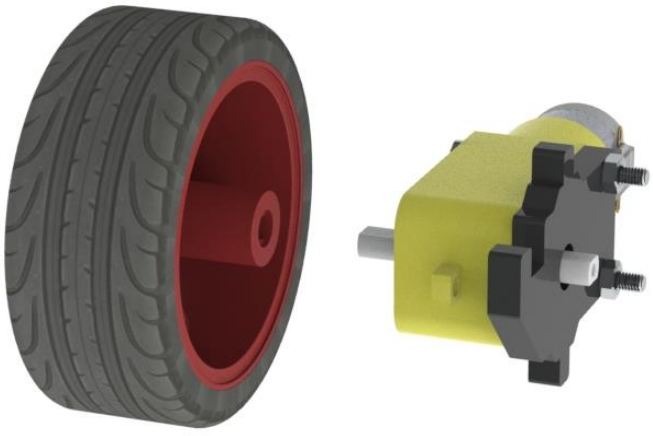
1.Assemble the following components



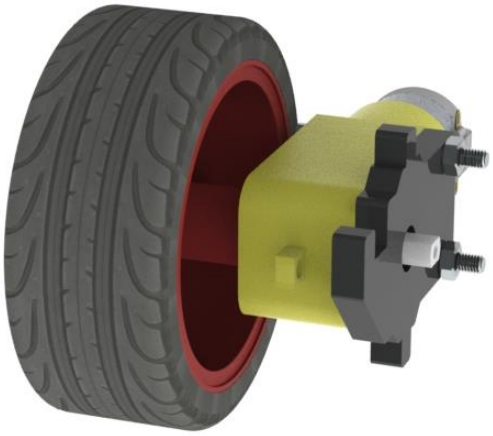
After assembling



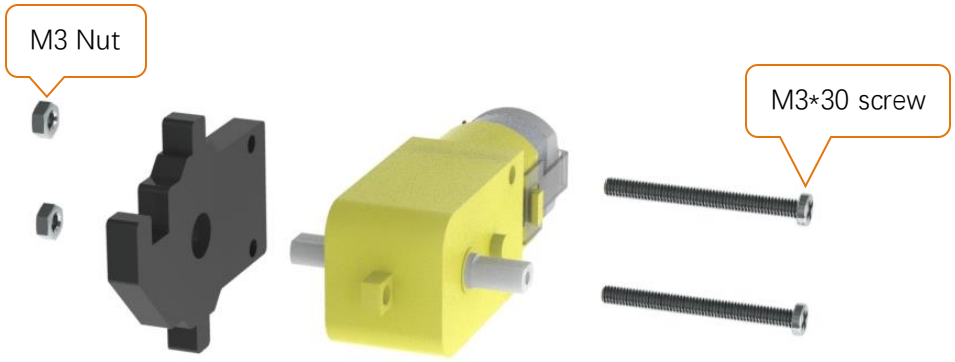
2.Install the wheel(Left)



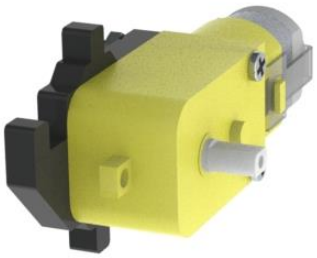
After assembling



3.Assemble the following components



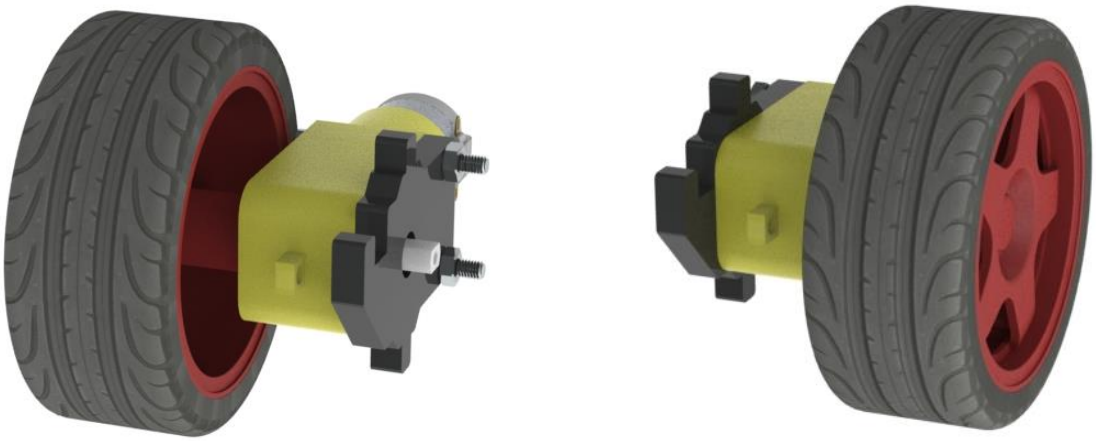
After assembling



4.Install the wheel(Right)

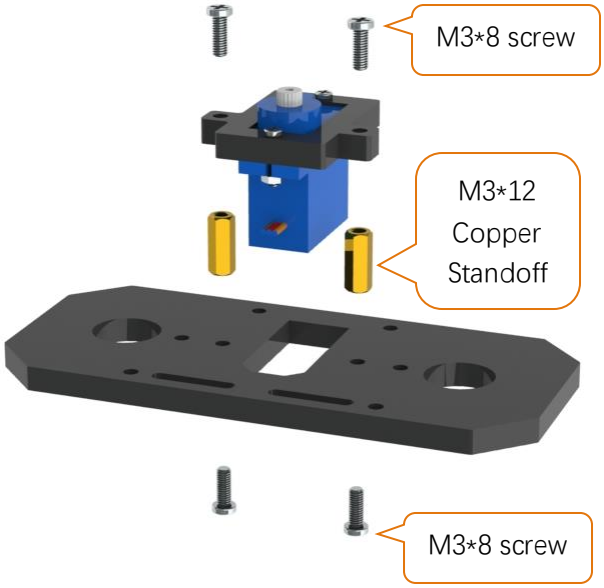
After assembling

After finishing the previous installing, you'll get two parts as follows:

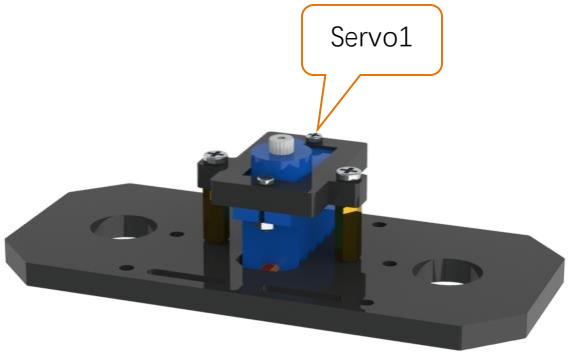


Assemble servo1

Assemble the following components

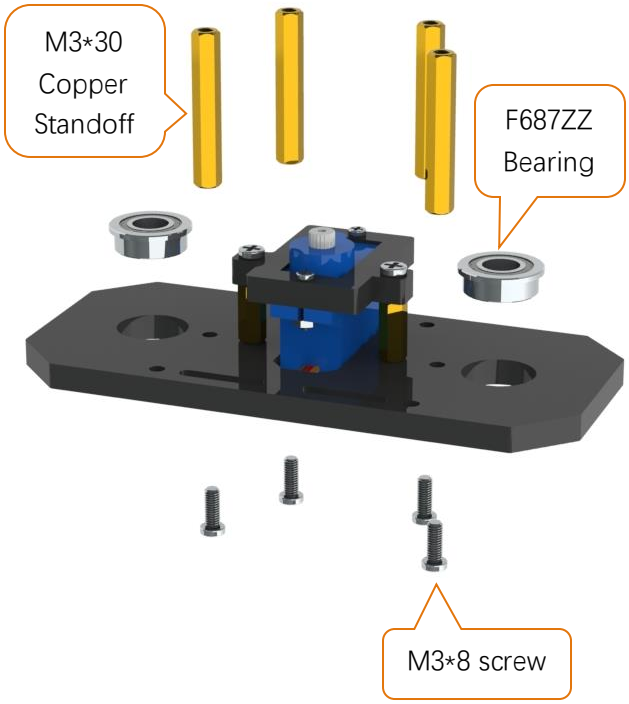


After assembling

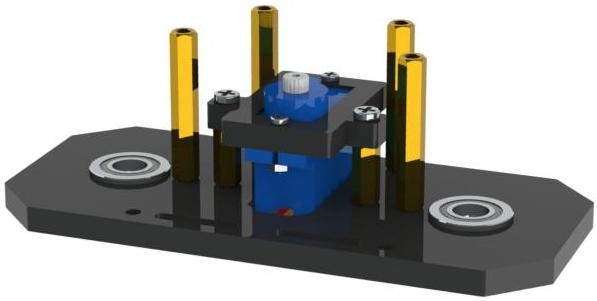


Assemble Copper Standoff

Assemble the following components

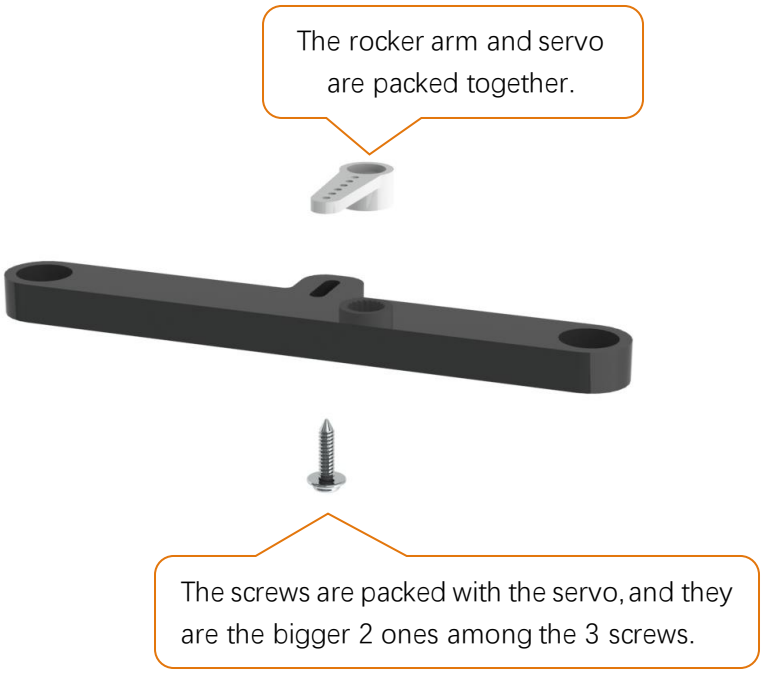


After assembling

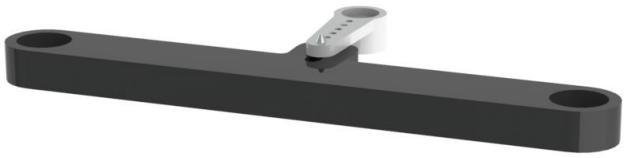


Assemble Direction stick

Assemble the following components



After assembling

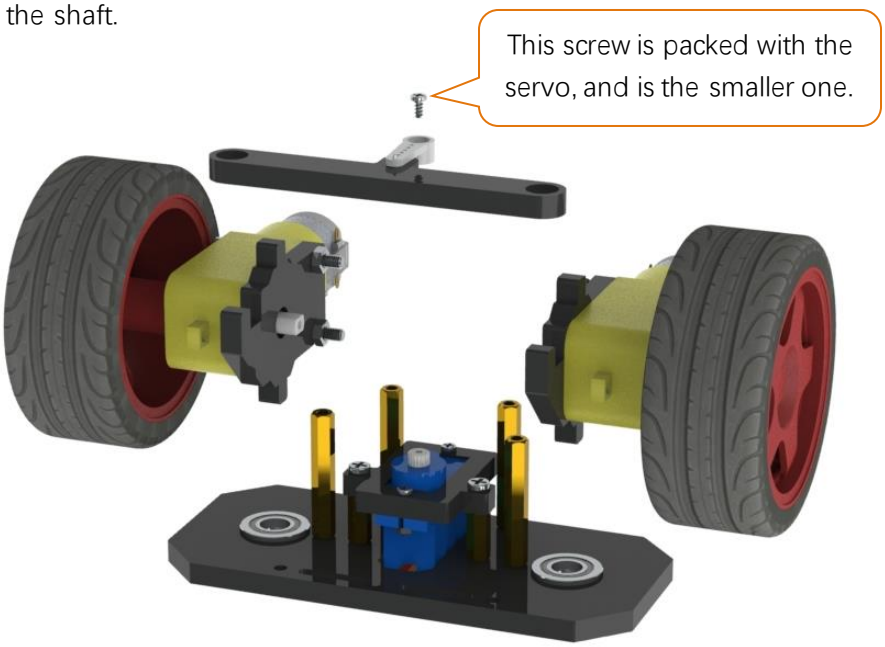


Pay attention not to tighten the screw, otherwise the acrylic plate can't move freely.

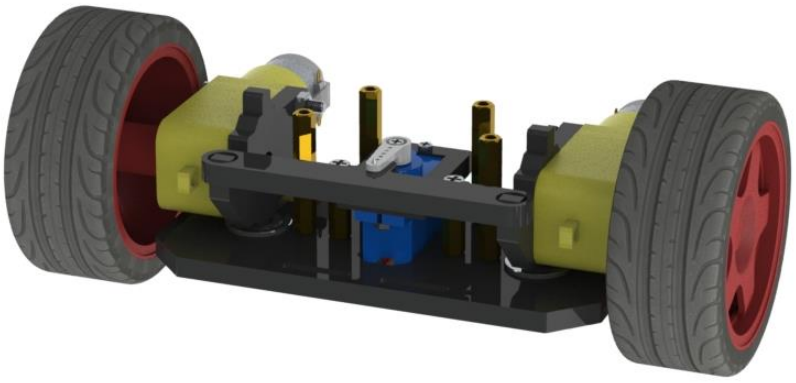
Assembly car head

Assemble the following components

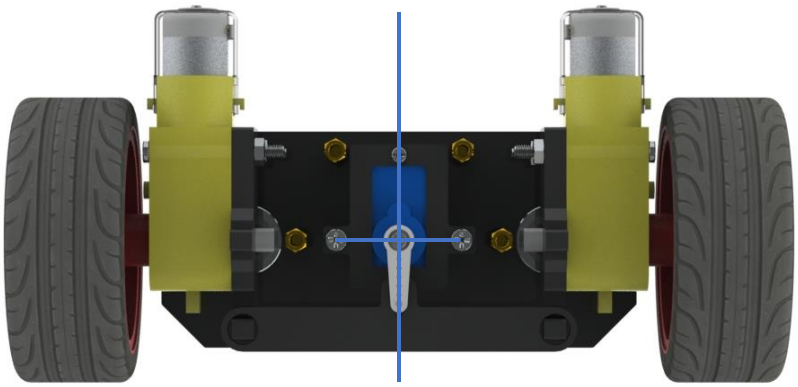
The rocker arm should be installed in the middle position between its rotation range. A little deviation is acceptable. If it is not installed in the middle position, you should remove the rocker arm and install it again instead of turning the shaft.



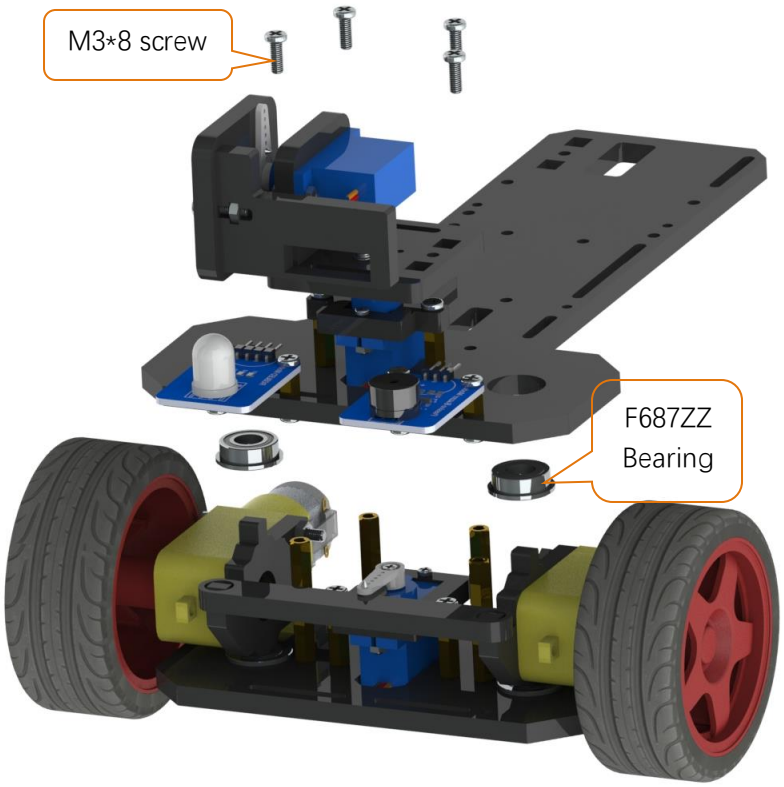
After assembling



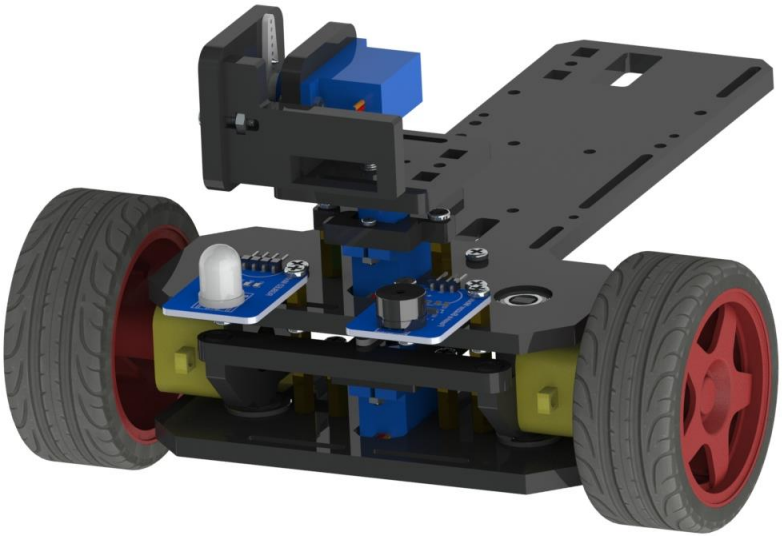
Keep the servo1 rotating to 90 degrees. If the drection is changed before installation, please make it rotate to the 90 degrees with the previous method. Complete the installation according to the direction below.



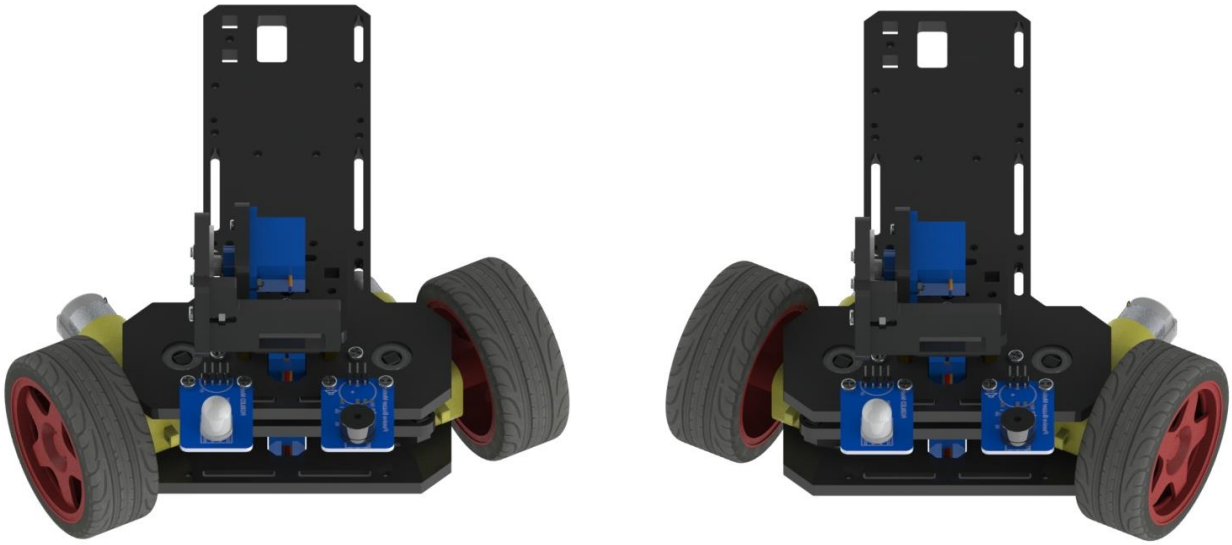
Assemble the following components



After assembling

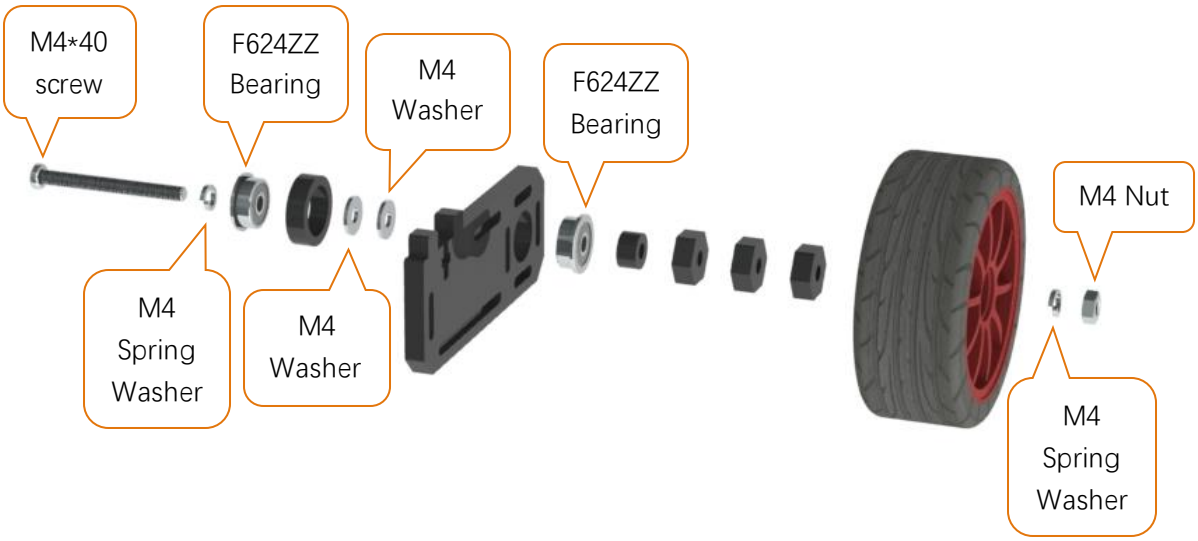


The two wheels can turn left and right



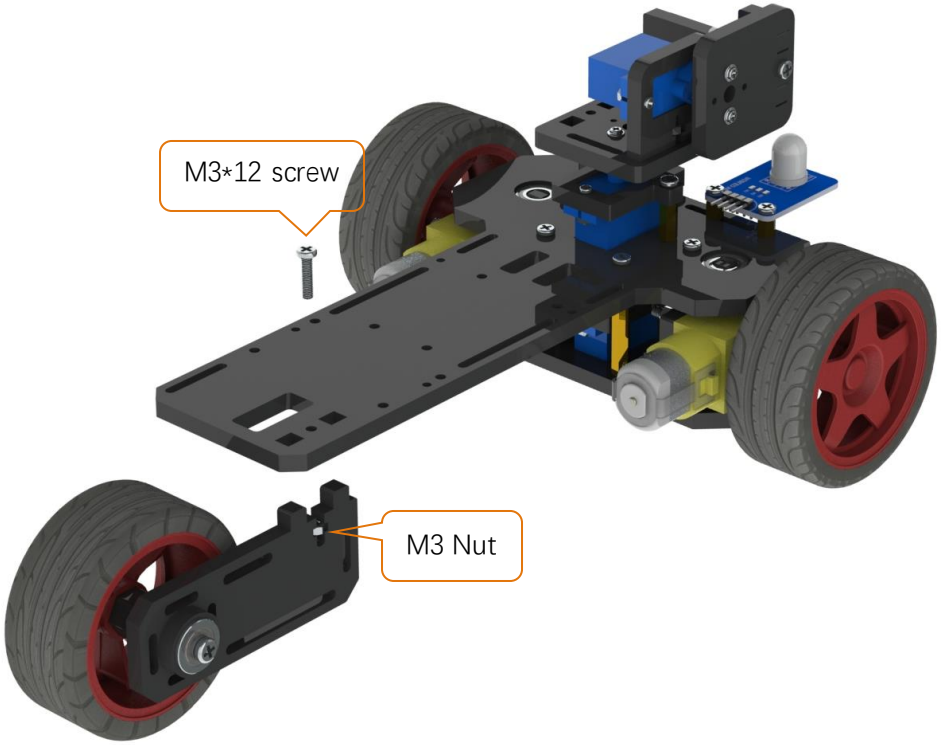
Assemble driven wheel

Assemble the following components

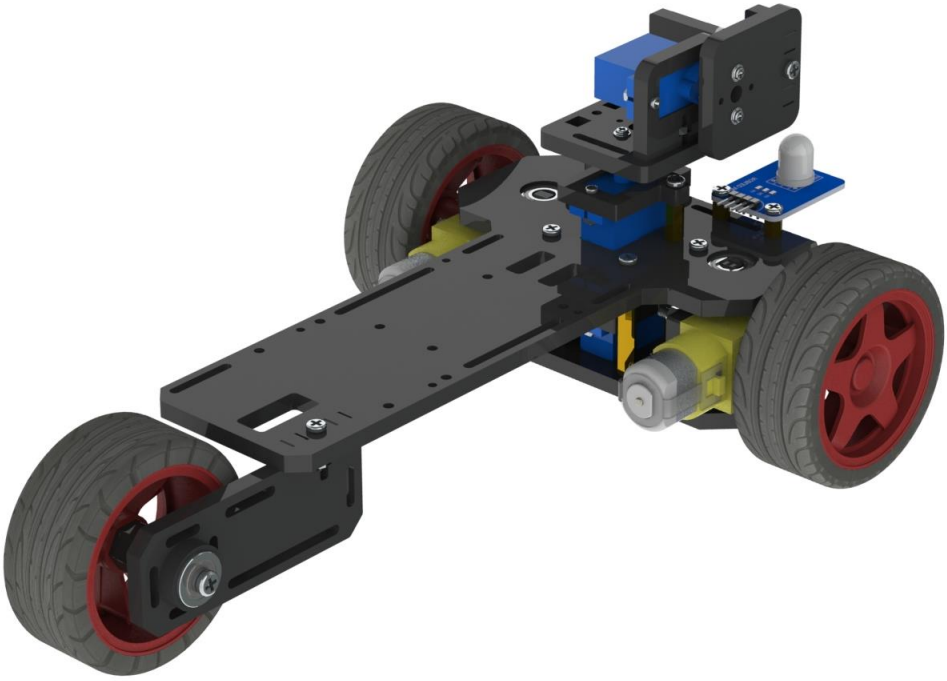


After assembling

Assemble the following components



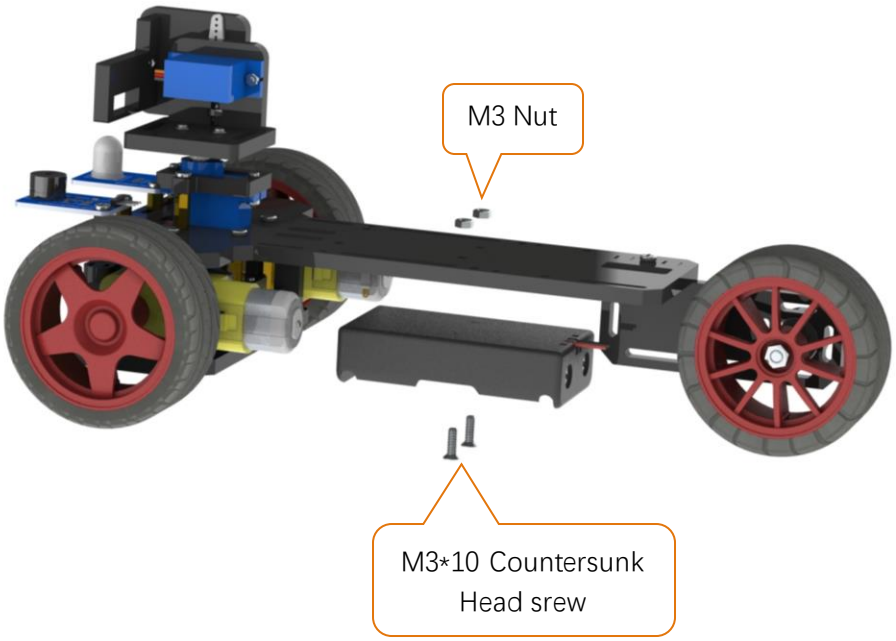
After assembling



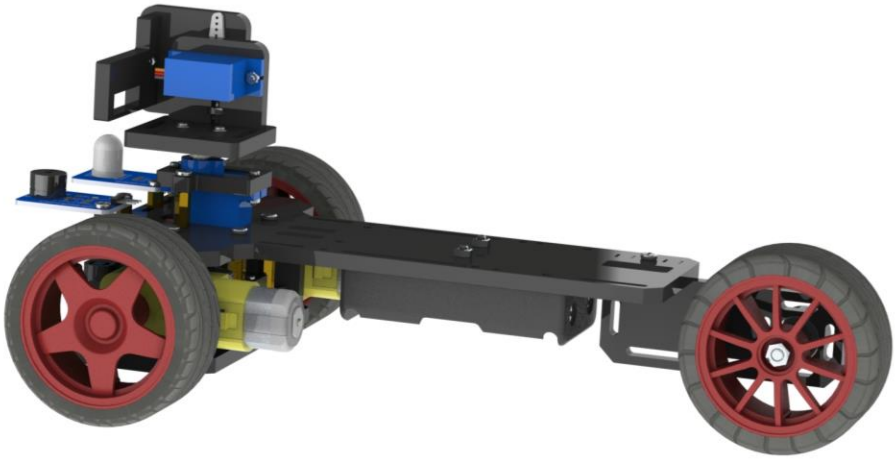
Electronic device

Assemble battery box

Assemble the following components

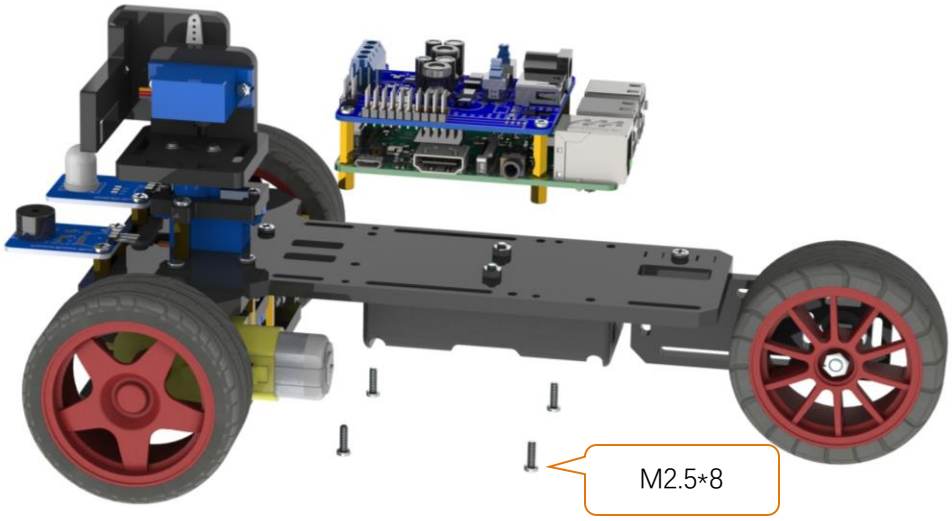


After assembling

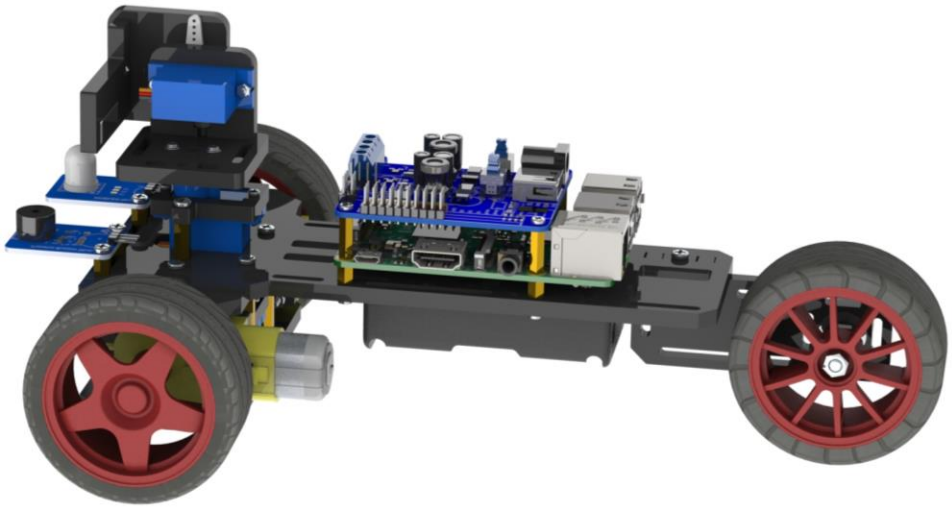


RPi and camera

Assemble the following components



After assembly

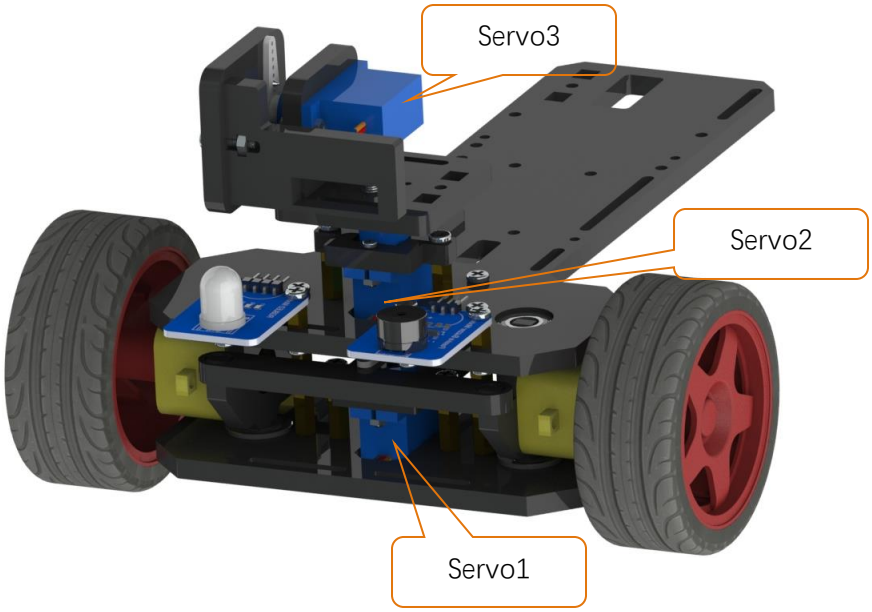


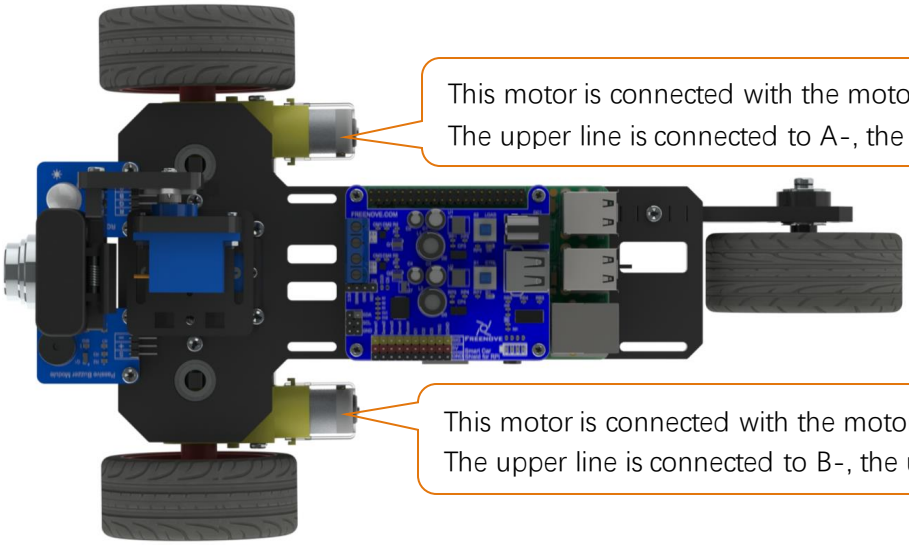
Connection: please refer to Charpter 0, Step 0.5 test.

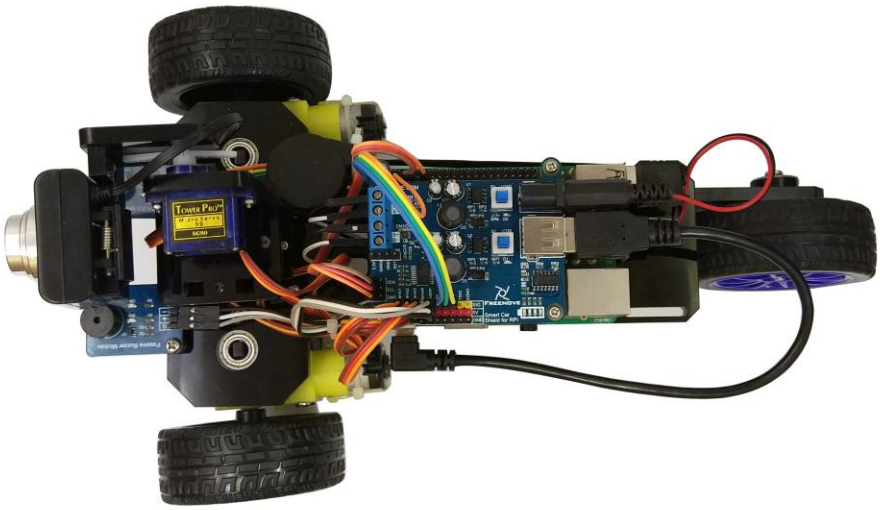
Servo1, Servo2, Servo3 are connected to Servo1, Servo2, Servo3 port of the Shield respectively.

RGBLED Module, Buzzer Module, and other loads are connected to the Shield in the same connection mode with front "test" section.

The motor is connected as below, in which, if you find motor steering error, exchange connection of two lines to Shield. The camera is connected to any USB port on the RPi. Using the matching Micro USB Cable to connect Shield USB port with RPi power supply port. Assemble two 18650 batteries in the battery box, and connect the interface of battery box to DC power jack of the Shiel


Servo3
Servo2
Servo1


This motor is connected with the motor A port of Shield.
The upper line is connected to A-, the under line A+.
This motor is connected with the motor B port of Shield.
The upper line is connected to B-, the under line B+.



Run the code

After the assembly is completed, run the code according to the following way. Then you can let the video car run up.

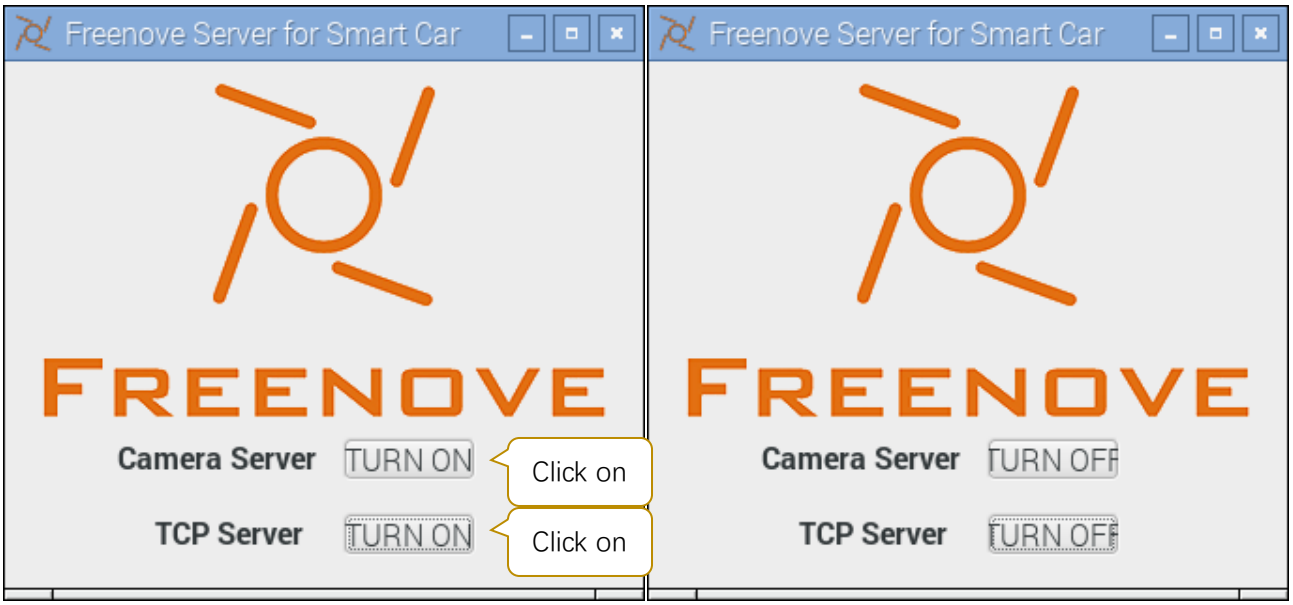
Open the server

Windowed server

Open the switch S1 and S2 on the Shield. After the RPi starts, use the remote desktop to connect RPi. Then open the terminal, and execute the following command to open the server. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python Main.py
```

Later, the following window interface appears. Click on the two buttons in the window to open the camera service and TCP communication service, respectively.



If the terminal shows information below, it indicates that the camera service and TCP communication service have been opened.

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
python Main.py
btn_CameraServer Clicked!
.............Camera server starting ......
MJPG Streamer Version: svn rev: Unversioned directory
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 320 x 240
 i: Frames Per Second.: 30
 i: Format............: YUV
 i: JPEG Quality......: 80
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8090
 o: username:password.: disabled
 o: commands..........: enabled
TCP Server Thread Starting ...
Wating for connect ...
```

When you want to close them, first click on two TURN OFF button to close the services, and then click on the close button on the top right corner of the window to terminate the program.

Server in command line mode

If you do not like the windowed server, you can open the camera and TCP communication services directly through the commands. Open the switch S1 and S2 on the Shield. After the RPi starts, use the remote desktop connect RPi. Then open the terminal and execute the following command. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server
python Main.py –mnt
```

or

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server
python Main.py –m –t -n
```

Parameter "-m" is mjpg-streamer, which means to open the camera service. "-t" means to open the tcp service. "-n" means not to use the visual window interface.

Later, if the following contents appears, it indicates that the camera and tcp services have been opened.

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
python Main.py -mnt
.............Camera server starting ......
TCP Server Thread Starting ...
Wating for connect ...
MJPG Streamer Version: svn rev: Unversioned directory
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 320 x 240
 i: Frames Per Second.: 30
 i: Format............: YUV
 i: JPEG Quality......: 80
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8090
 o: username:password.: disabled
 o: commands..........: enabled
```

Press twice Ctrl-C or Ctrl-\ to terminate the program.

Open the client

The client can run under any operating system in which Python3 and PyQt5 is installed. For example, Windows OS, Linux OS.

Client under Windows OS

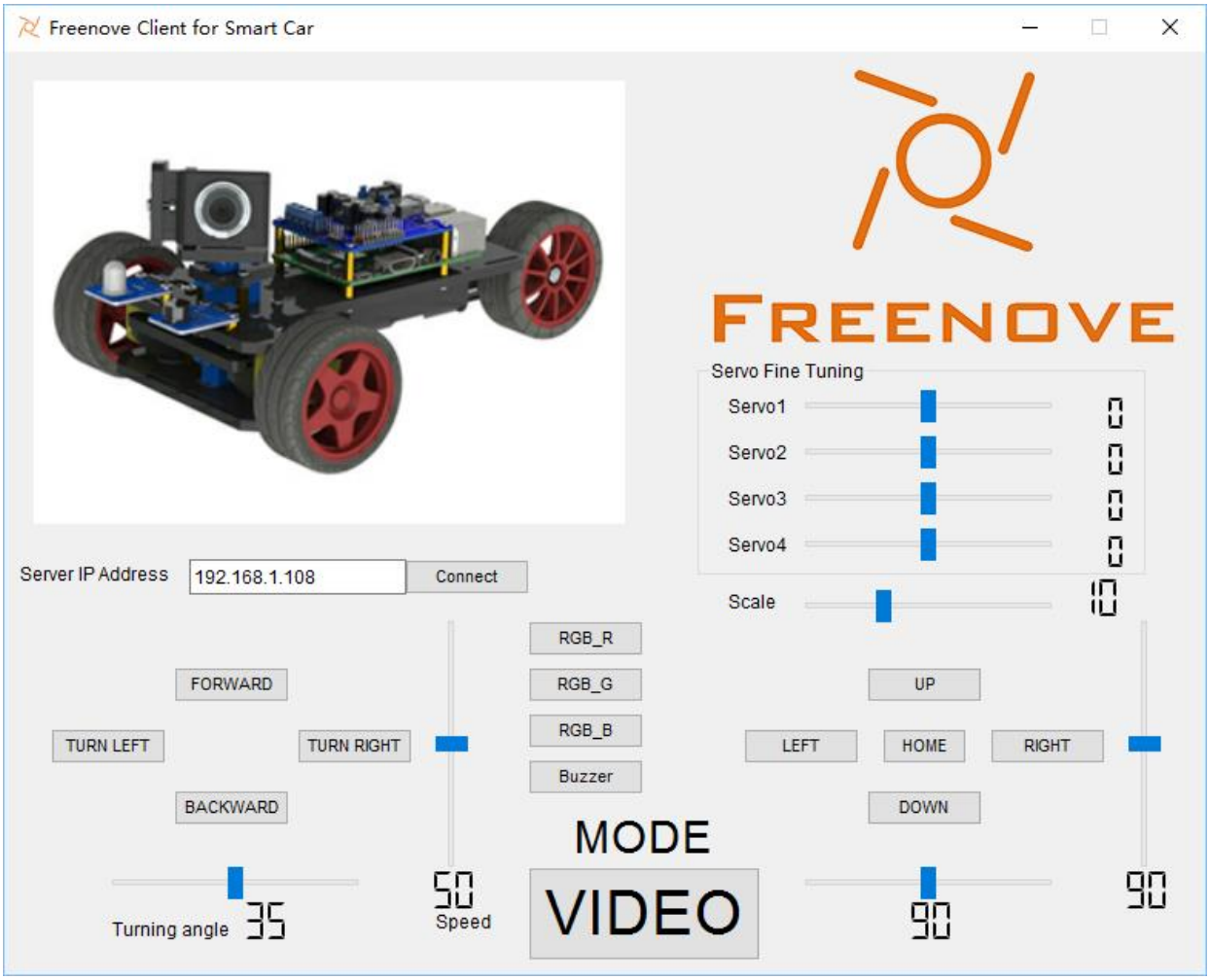
Download the code under windows. Unzip it and place it in the D disk root directory. You can also place it into other disks (like E), but the path in following command should be modified accordingly (replace D: by E:). Click on the link below to download the file directly. You need to delete "-master" to rename the unzipped file to "Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi".

https://github.com/Freenove/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/archive/master.zip

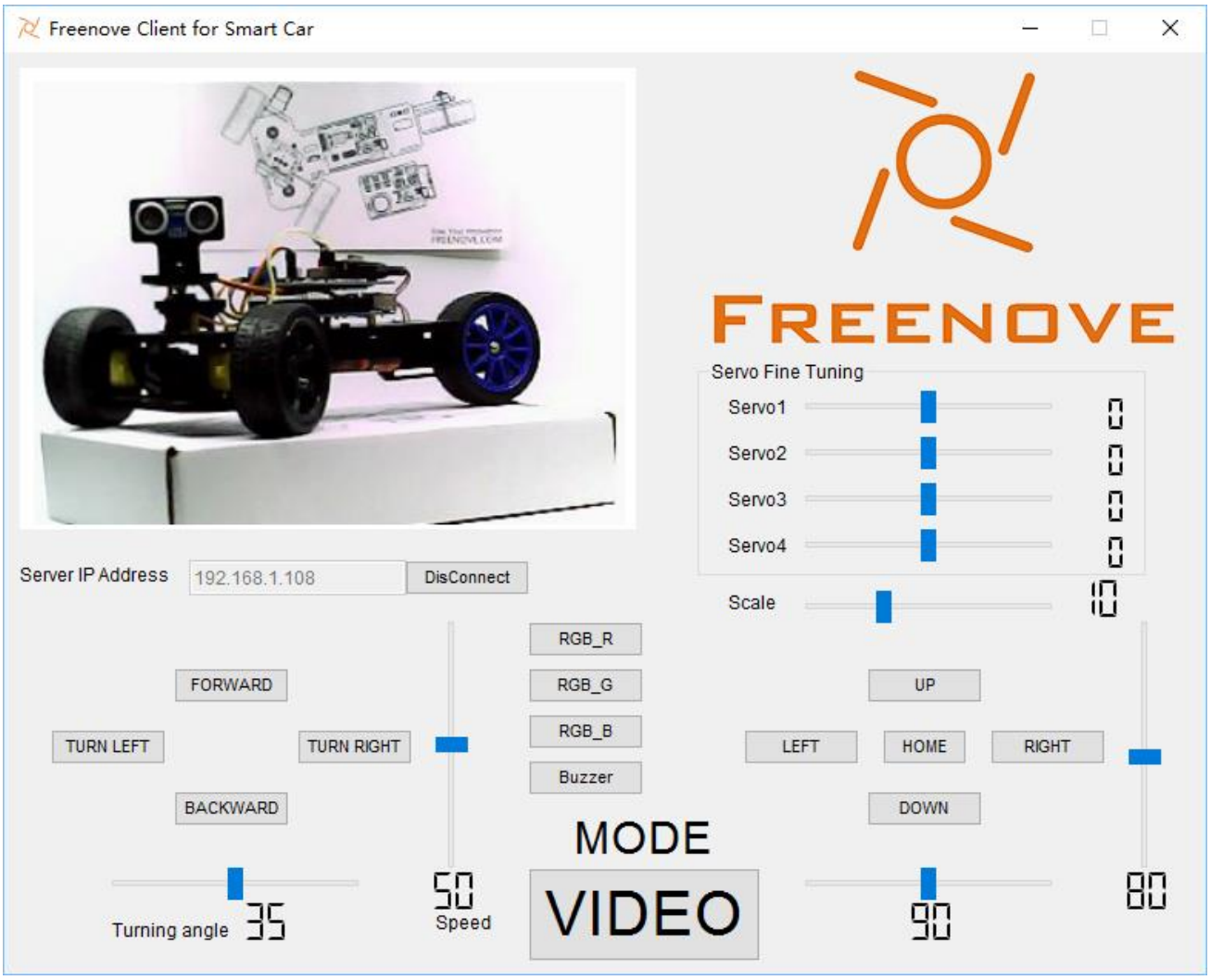
Press WIN+R, and type cmd to open the command line window. Then type the following command to open the client. **(Note: Here are three commands. Please excute commands in order.)**

```
D:
cd \Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi\Client\
python main.py
```

Or enter path "D:\Freenove_Three_wheeled_Smart_Car_for_Raspberry_Pi\Client\" and double-click main.py with open way of Python3.exe. Then the following window interface appears.



In the edit box Server IP Address, input IP address of video car RPi and click Connect. Make sure that server of your RPi have been opened already, and the switch S1 and S2 on the Shield have also been opened.



After the connection is successful, you can control the video car.

Control mode

Besides using the mouse to click on the button in the window to control the video car, you can also use the highlight keys of keyboard to control the video car, as shown below.

The following is the corresponding action of Button/Key.

Button	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	H	Turn camera back Home
RGB_R	R	On/off red LED of RGBLED
RGB_G	G	On/off green LED of RGBLED
RGB_B	B	On/off blue LED of RGBLED
Buzzer	V	On/off Buzzer

The function of the SiliderBar with name in the window is shown below.

SliderBar	Function
Speed	Control the forward / backward speed of the car.
Turning angle	Control turning angle.
Scale	The minimum angle of each rotation of the camera.
Servo Fine Turning 1,2,3,4	Servo1,2,3,4 is for angle fine tuning settings. If the servo is not completely centered in installation, you can make a fine tuning by the SliderBar

Other control:

Control	Function
Edit box Server IP Address	IP address of Server
Button “Connect/DisConnect”	Connect or DisConnect Server
Mode Button	Switch to Video/Radar Mode

Client under Linux OS

Similarly, using python to execute code Client.py and type the following command to open the client. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi /Client

python main.py
```

Later, the same client window interface appears. And the control mode and operation mode are the same with Windows OS.

Other operating systems

Under operating system in which Python3 and PyQt5 is installed, you can run the client application through Python. Because their code is same, and with same control method.

Servo Reversed

Note:

When you run client to control servos, if some servo direction is reversed. You need modify **mTCPServer.py** file. Under following path in your **Raspberry Pi**.

Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server/mTCPServer.py

When a servo is reversed, just modify corresponding False to True.

```
##############################################################
#If your servo rotation is inverted, please set the corresponding value to True
servo1_reversed = False         # True or False
servo2_reversed = False         # True or False
servo3_reversed = False         # True or False
##############################################################
```

For example, servo1 rotation is reversed, modify the code.

```
##############################################################
#If your servo rotation is inverted, please set the corresponding value to True
servo1_reversed = True          # True or False
servo2_reversed = False         # True or False
servo3_reversed = False         # True or False
##############################################################
```

Then save the modification.

Automatic Start

After you understand how the video car works, the server can be set to start automatically once Raspberry Pi is turned on. So when you start Raspberry Pi, you can directly connect the car to client, without opening and configuring the server remotely. There are many ways to set automatic start. Here we introduce a simple and easy way.

Configuration for automatic start

The file "Raspberry PiCar.desktop" under "Server" directory is the start configuration file of Server program. The name of it shown in file manager is "RaspberryPiCar". First, use following command to see if the file is executable.

```
ls -l | grep "RaspberryPi"
```

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
ls -l | grep "RaspberryPi"
-rw-r--r-- 1 pi pi    234 Nov 16 02:20 RaspberryPiCar.desktop
```

As shown above, if the file is not executable, use following command to add executable permissions.

```
chmod +x RaspberryPiCar.desktop
```

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
chmod +x RaspberryPiCar.desktop
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
ls -l | grep "RaspberryPi"
-rwxr-xr-x 1 pi pi    234 Nov 16 02:20 RaspberryPiCar.desktop
```

If you haven't changed the file "Raspberry PiCar.desktop" name or path, you don't need to modify. Otherwise you must modify it to your own path. Open the file with file editor. **Exec = XXX** indicates that the command xxx which will be executed. Make sure that the command and path are correct.

```
geany RaspberryPiCar.desktop
```

RaspberryPiCar.desktop

```
1  [Desktop Entry]
2  Type=Application
3  Terminal=true
4  Name=RaspberryPiCar
5  Comment=Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi.Server.Autostart
6  Exec=sudo sh /home/pi/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server/Startup.sh
7  Icon=None
8  MultipleArgs=false
9  Categories=Application
10 StartupNotify=true
11 Hidden=false
12 NoDisplay=false
13
```

Finally, we will guide you to copy the file "Raspberry PiCar.desktop" to directory "/home/pi/.config/autostart/". If the directory does not exist, the directory need to be created first.

Check if directory exists.

```
ls ~/.config/
```

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
ls ~/.config/
chromium  gtk-2.0  lxpanel      openbox   QtProject.conf    vlc
dconf     gtk-3.0  lxsession    pcmanfm   Trolltech.conf
geany     leafpad  lxtask.conf  qpdfview  user-dirs.dirs
gpicview  libfm    lxterminal   qt5ct     user-dirs.locale
```

If the directory does not exist, create it by using following command. If the directory exists, skip the command.

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

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
mkdir ~/.config/autostart/
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
ls ~/.config/
autostart  gpicview  libfm        lxterminal  qt5ct          user-dirs.locale
chromium   gtk-2.0   lxpanel      openbox     QtProject.conf  vlc
dconf      gtk-3.0   lxsession    pcmanfm     Trolltech.conf
geany      leafpad   lxtask.conf  qpdfview    user-dirs.dirs
```

Use following command to copy the file "RaspberryPiCar.desktop" to folder "autostart".

```
cp RaspberryPiCar.desktop ~/.config/autostart
```

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
cp RaspberryPiCar.desktop ~/.config/autostart/
```

Execute the "RaspberryPiCar.desktop" in folder "autostart" to test whether it is workable.

```
sudo ~/.config/autostart/RaspberryPiCar.desktop
```

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
sudo ~/.config/autostart/RaspberryPiCar.desktop
/home/pi/.config/autostart/RaspberryPiCar.desktop: 1: /home/pi/.config/autostart
/RaspberryPiCar.desktop: [Desktop: not found
/home/pi/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server
libpng warning: iCCP: known incorrect sRGB profile
libpng warning: iCCP: known incorrect sRGB profile
.............Camera server starting ......
TCP Server Thread Starting ...
Wating for connect ...
/home/pi/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/mjpg-streamer
MJPG Streamer Version: svn rev: Unversioned directory
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 320 x 240
 i: Frames Per Second.: 30
 i: Format............: YUV
 i: JPEG Quality......: 80
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8090
 o: username:password.: disabled
 o: commands..........: enabled
```

The above results show that the Server program has been successfully started. Then restart your raspberry pi to test whether automatic start is successfully.

```
sudo reboot
```

After restarting, you can connect raspberry pi with client. You can check whether the Server terminal works normal with the following commands.

```
ps aux | grep -E "./mjpg_streamer|Main.py"
```

```
pi@raspberrypi:~ $ ps aux | grep -E "./mjpg_streamer|Main.py"
root      3641  0.1  3.5  95736 31388 pts/0    Sl+  06:49   0:00 python2 Main.py -mnt
root      3646 29.4  0.2  23388  2656 pts/0    Sl+  06:49   3:05 ./mjpg_streamer -i ./input_uvc.so
 -y -d /dev/video0 -n -r 320*240 -f 30 -o ./output_http.so -p 8090 -w ./www
```

Exit Server program

Executing the program CloseServer.sh under Server can terminate the Server program.

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server
sh CloseServer.sh
```

```
pi@raspberrypi:~ $ cd Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server/
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $ sh CloseServer.sh
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $
```

Cancel automatic start

If you don't need automatic start for the Server program, just use following command to delete the file "~/.config/autostart/RaspberryPiCar.desktop".

```
rm -f ~/.config/autostart/RaspberryPiCar.desktop
```

```
pi@raspberrypi:~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server $ rm -f ~/.config/au
tostart/RaspberryPiCar.desktop
```

Android App

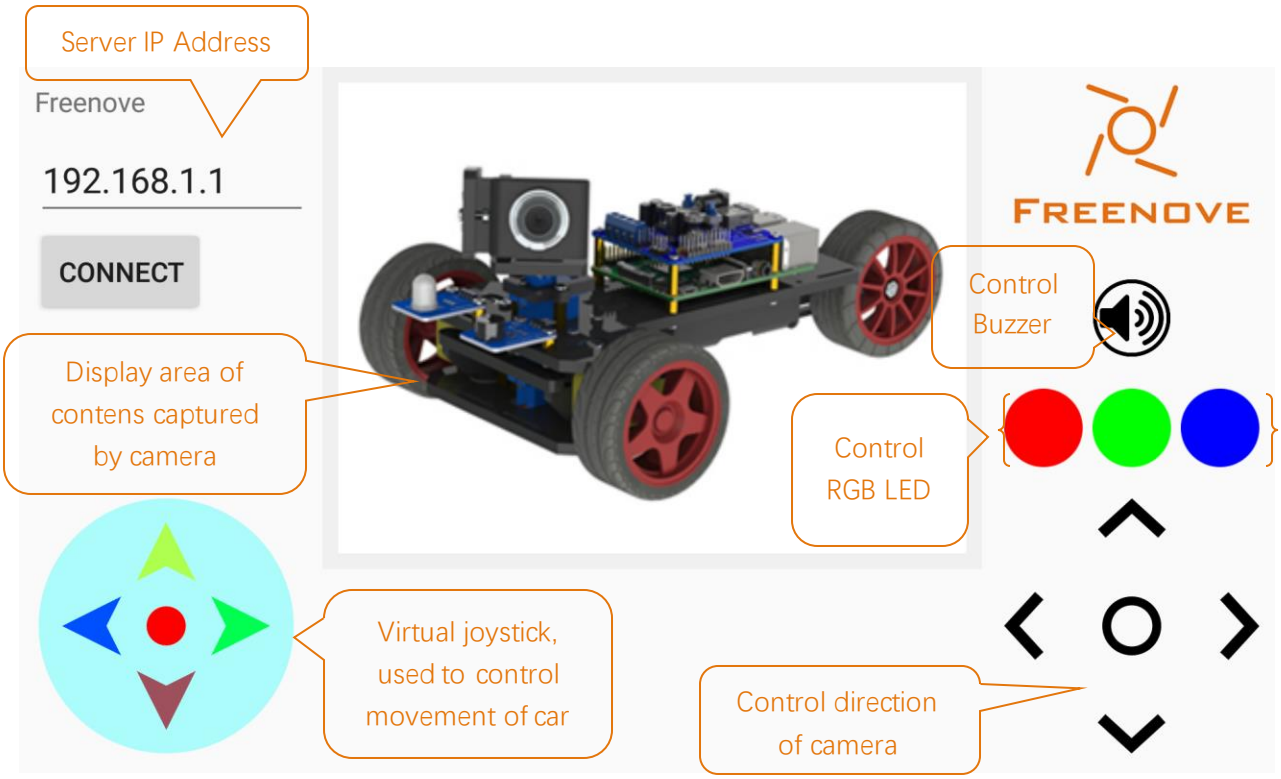
We provide an android app for this kit. You can use Android phone or tablet to control the car.

First, please install Freenove APP for your Android device. You can install the app in any of the following ways:

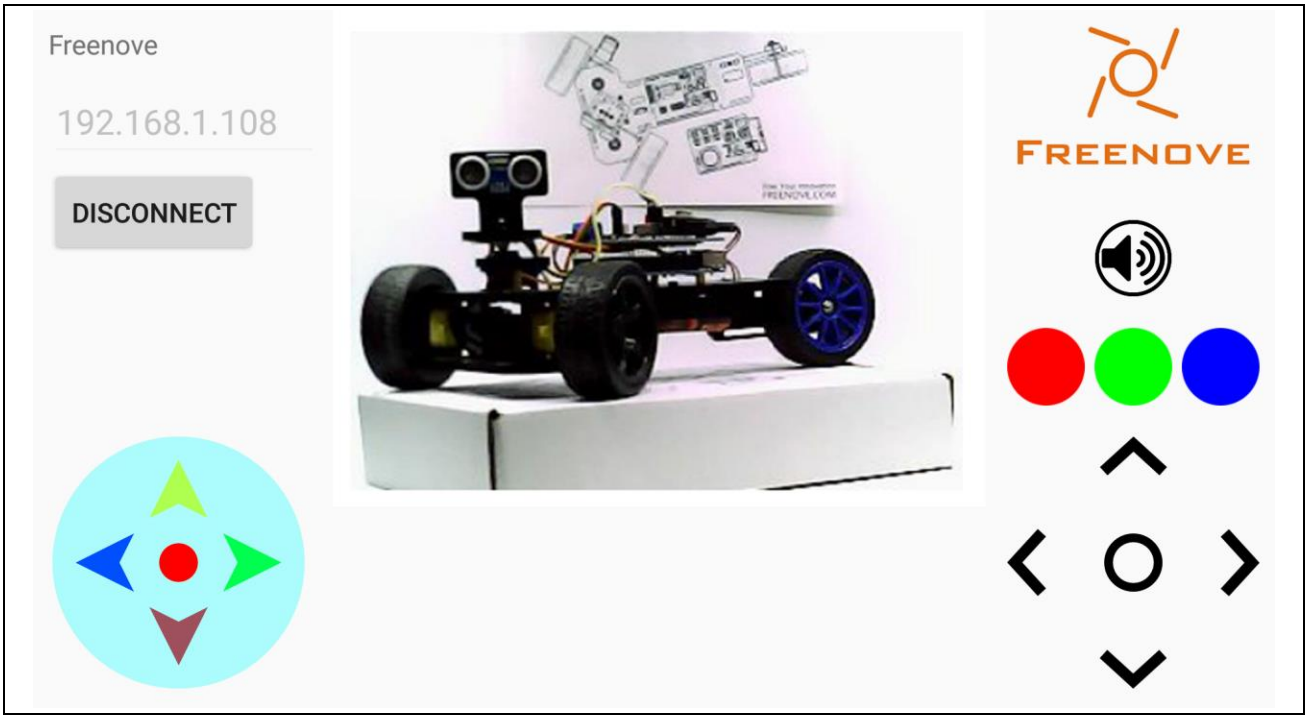
- View or download on Google Play:
  https://play.google.com/store/apps/details?id=com.freenove.suhayl.Freenove
- Download APK file directly:
  https://github.com/Freenove/Freenove_App_for_Android/raw/master/freenove.apk

More information about the app can be viewed and downloaded here:
https://github.com/Freenove/Freenove_App_for_Android/

After the installation is completed, open the Freenove APP and select "Smart Car for Raspberry Pi". As shown below:



Make sure camera and TCP services of the RPi have been opened. Then enter your RPi IP address in the column Server IP Address, click the button CONNECT. Then the connection succeeds later. RPi IP address is 192.168.1.108. after a successful connection, the interface is shown below.



The IP address will be stored after correct connection, so that it can be used multiple times without having to output the IP address every time. Then you can control the car.

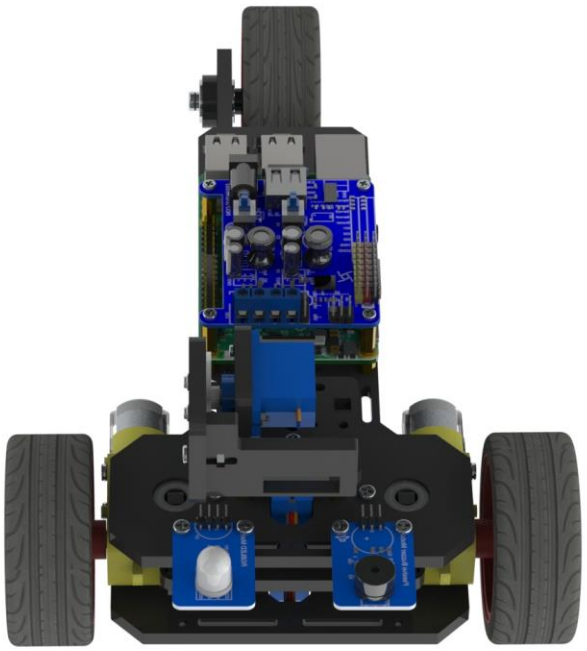
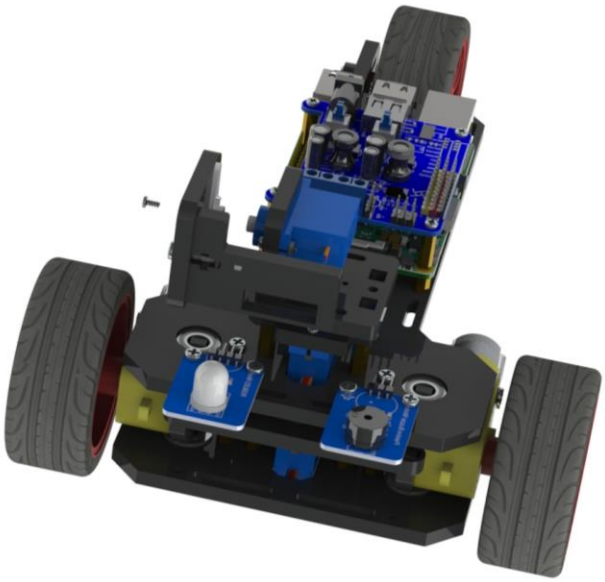
Chapter 2 Ultrasonic Radar Car

Assembly

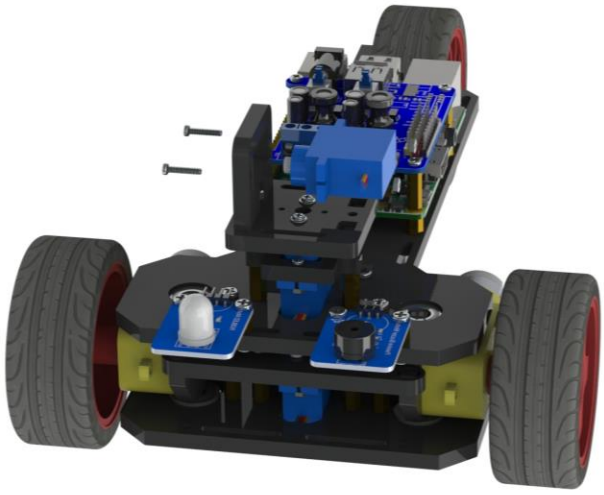
On the basis of the front video car, remove the camera, camera support, servo3, and servo3 support. Then assemble the support for Ultrasonic Module.

Replace pan-tilt

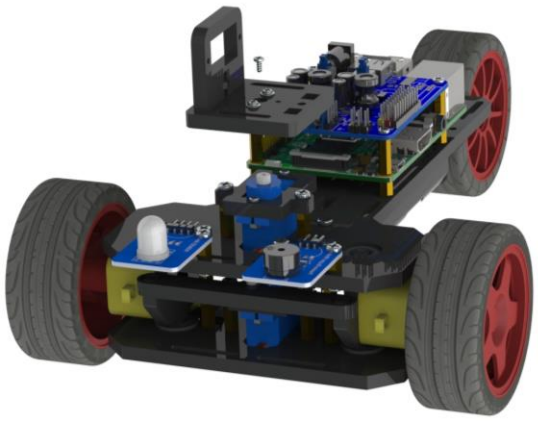
Remove camera part

Remove camera	Remove camera support
	

Remove servo3



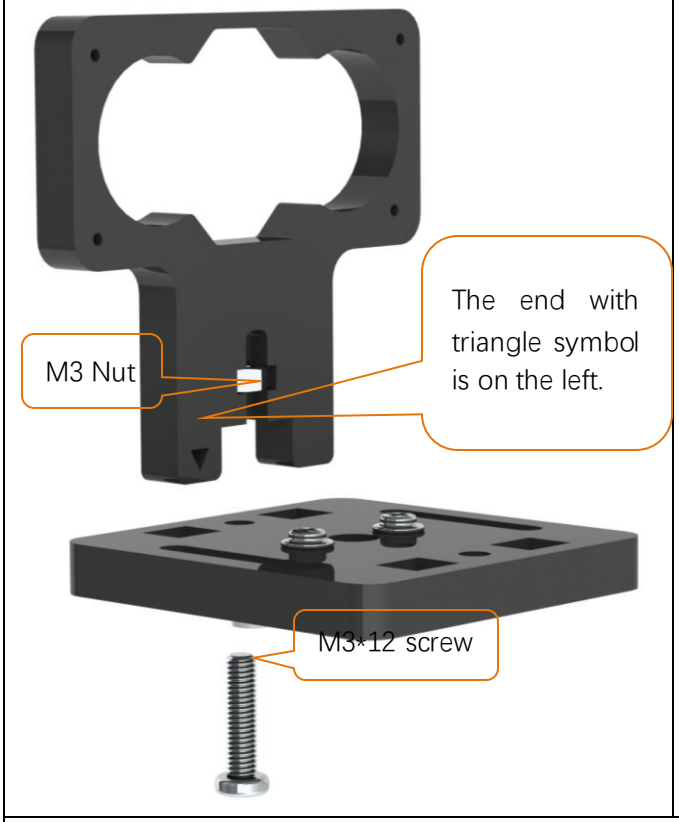
Remove servo3 support



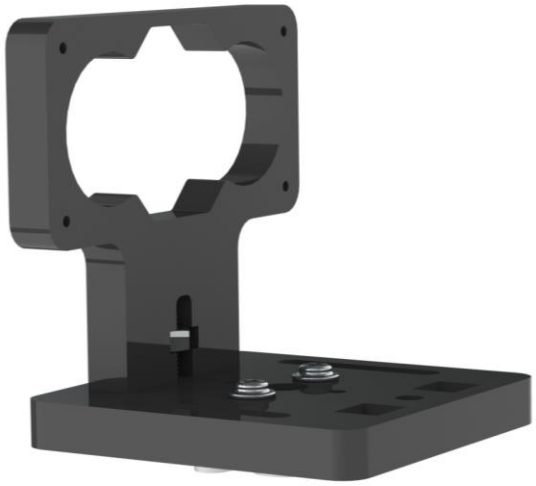
Separate Servo3 support

Assembly ultrasonic module support

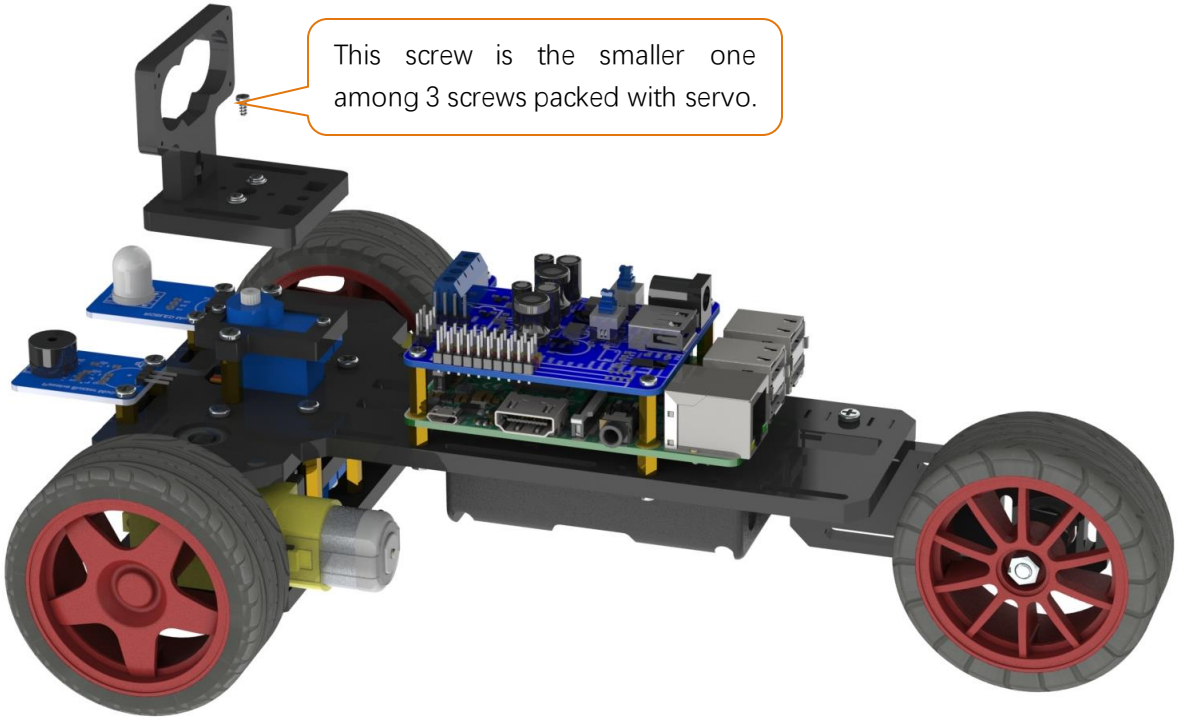
Assemble the following components



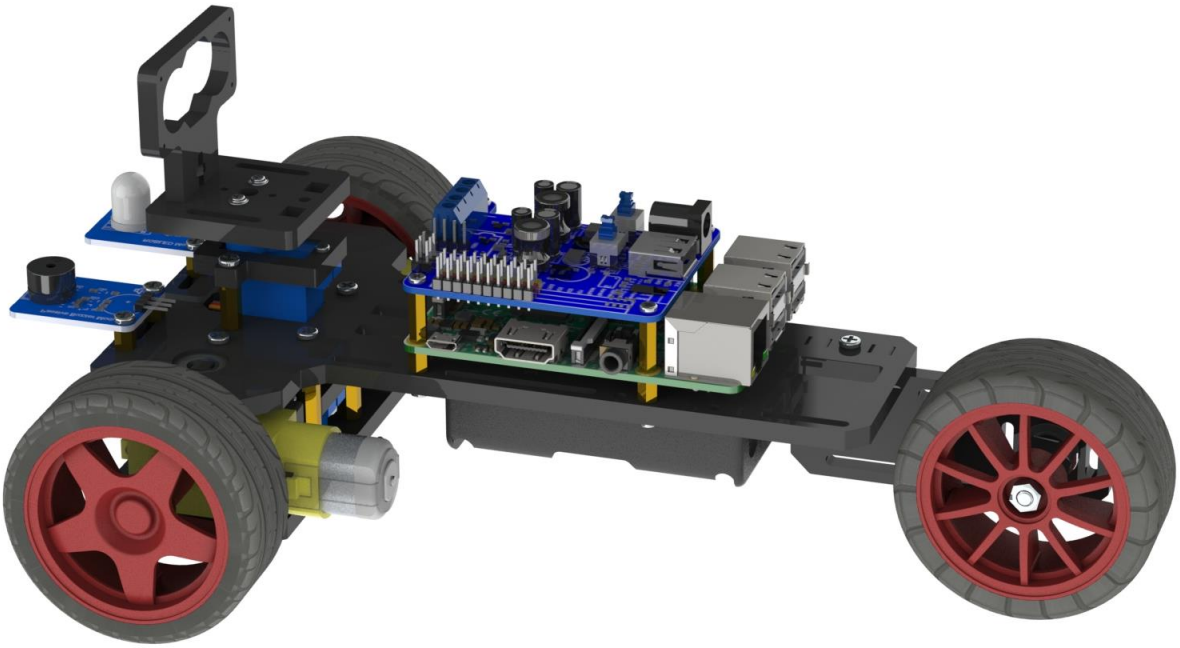
After assembling



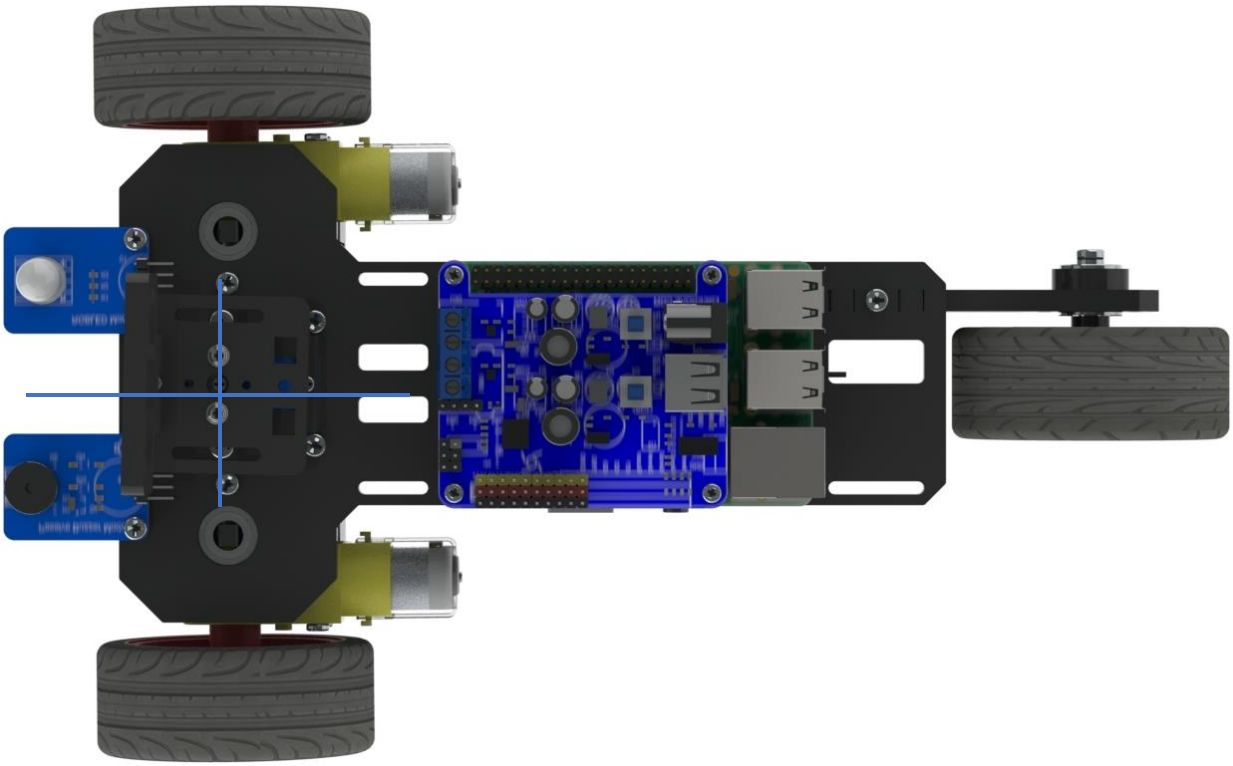
Assemble the following components



After assembling

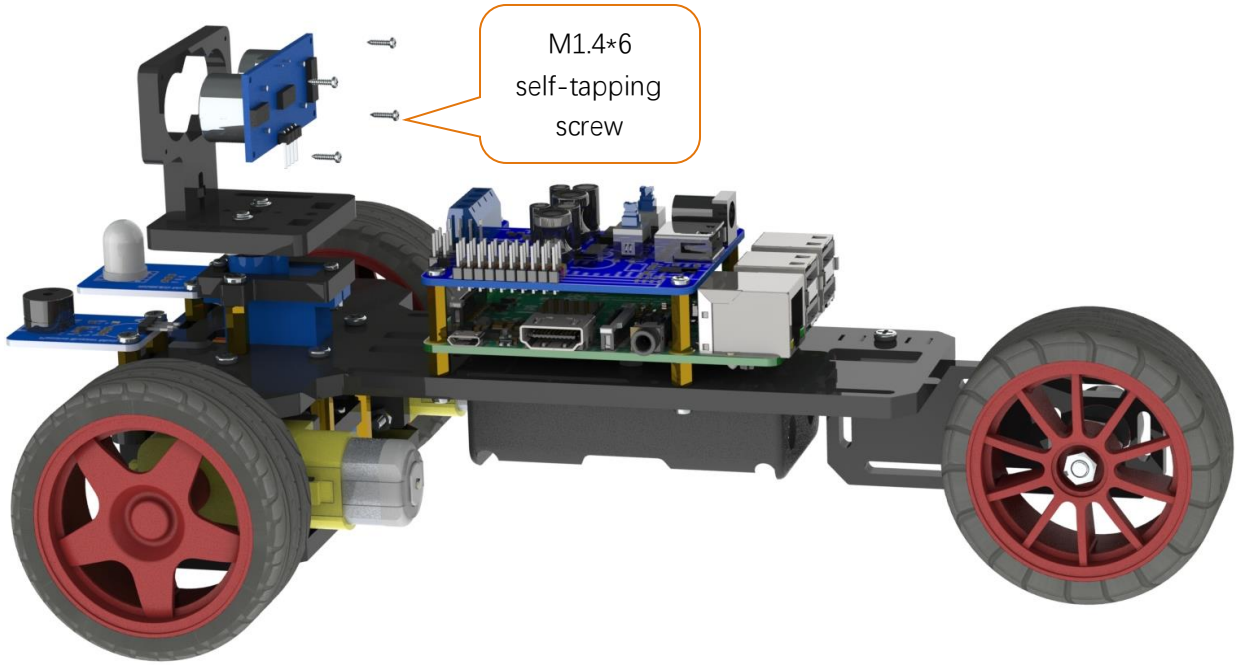


Keep the servo2 rotating to 90 degrees. If the drection is changed before installation, please make it rotate to the 90 degrees with the previous method. Complete the installation according to the direction below.

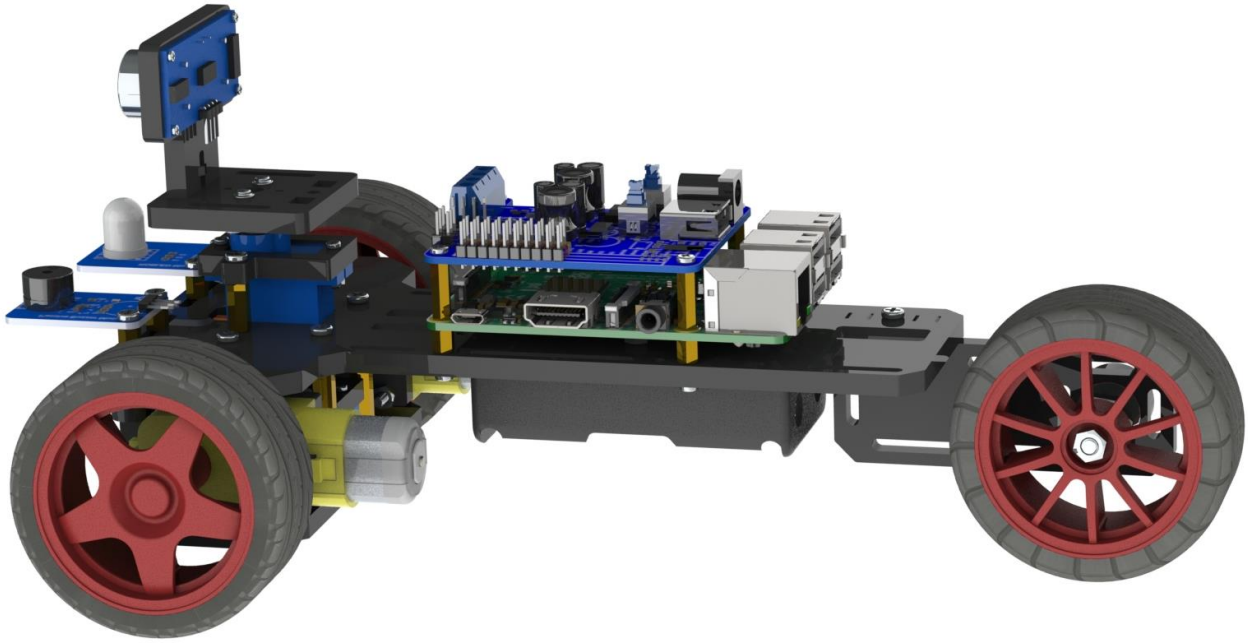


Ultrasonic Module

Use M-M Jumper to connect the Ultrasonic Module with the Shield. And then use TWO M1.4 * 6 tapping screws to assemble the Ultrasonic Module. As follows:

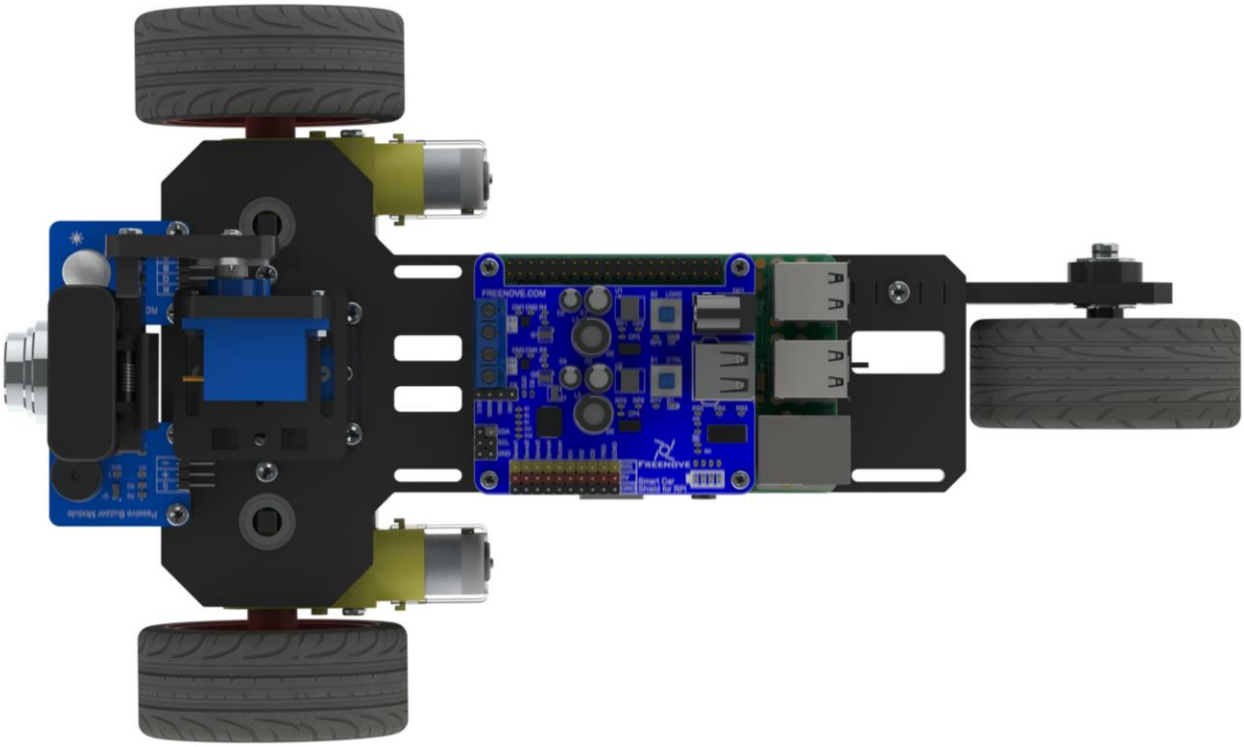


After assembling



Connection: please refer to Charpter 0, Step 0.5 test.

Connection mode between Shield and Ultrasonic Module, is the same as front "Test" section. Connection mode between other devices and Shield is the same with front “Smart video car”.



Run the Code

Open the server

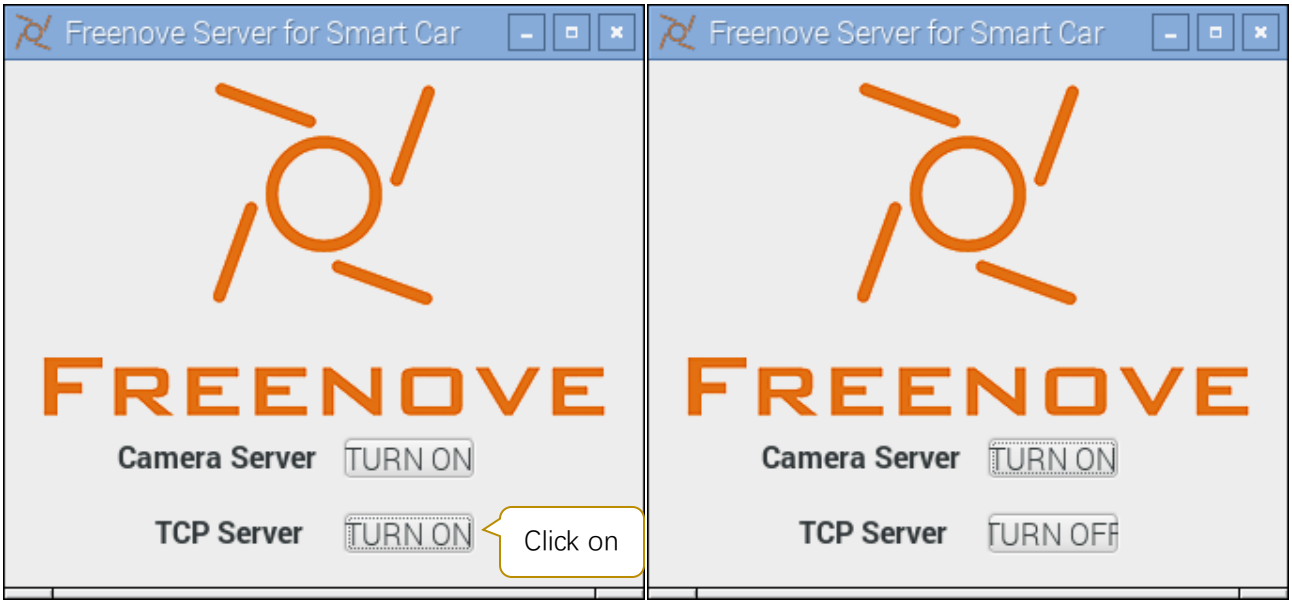
The method of opening the server is the same as the video car. Since there is no Camera device now, please do not open the Camera Server in the server window or use the "-m" command option.

Windowed server

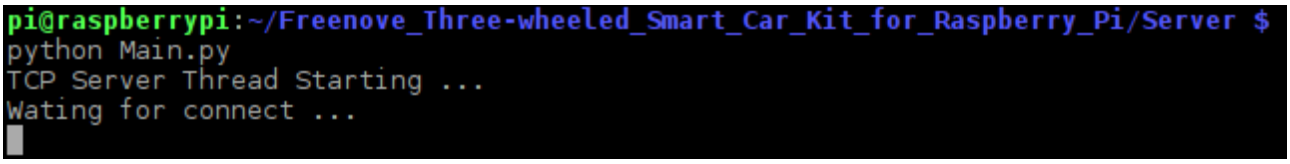
Open the switch S1 and S2 on the Shield. After the RPi starts, use the remote desktop to connect RPi. Then open the terminal, and execute the following command to open the server. **(Note: Here are two commands. Please excute commands in order.)**

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python Main.py
```

Later, the following window interface appears. Click on the corresponding button of TCP Server to open TCP communication Server.



If the terminal shows information below, it indicates that the camera service and TCP communication service have been opened.



When you want to close them, first click on two TURN OFF button to close the services, and then click on the close button on the top right corner of the window to terminate the program.

Server in command line mode

Type the following command to open the TCP service. **(Note: Here are two commands. Please excute commands in order.)**

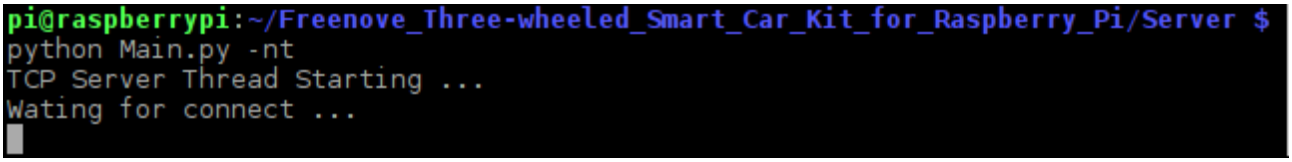
```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python Main.py –nt
```

or

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server

python Main.py –t -n
```

Parameter "-t" is means to open the tcp service. "-n" means not to use the visual window interface.

Later, if the following contents appears, it indicates that the tcp service has been opened.



Press Ctrl-C or Ctrl-\ to terminate the program.

Open the client

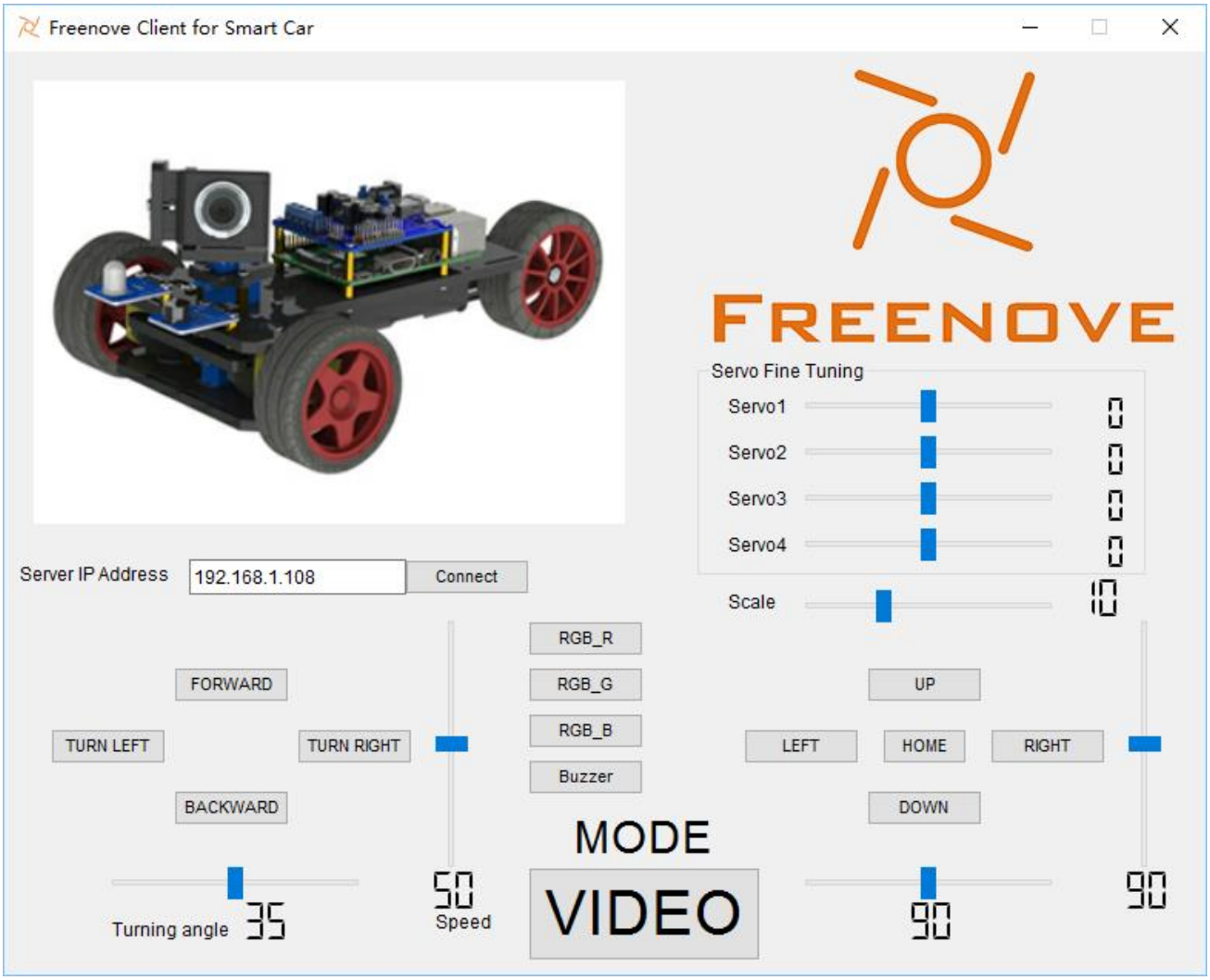
The way to open the client is the same as “Smart video car”

Client under Windows OS

Press WIN+R, and type cmd to open the command line window. Then type the following command to open the client. **(Note: Here are three commands. Please excute commands in order.)**

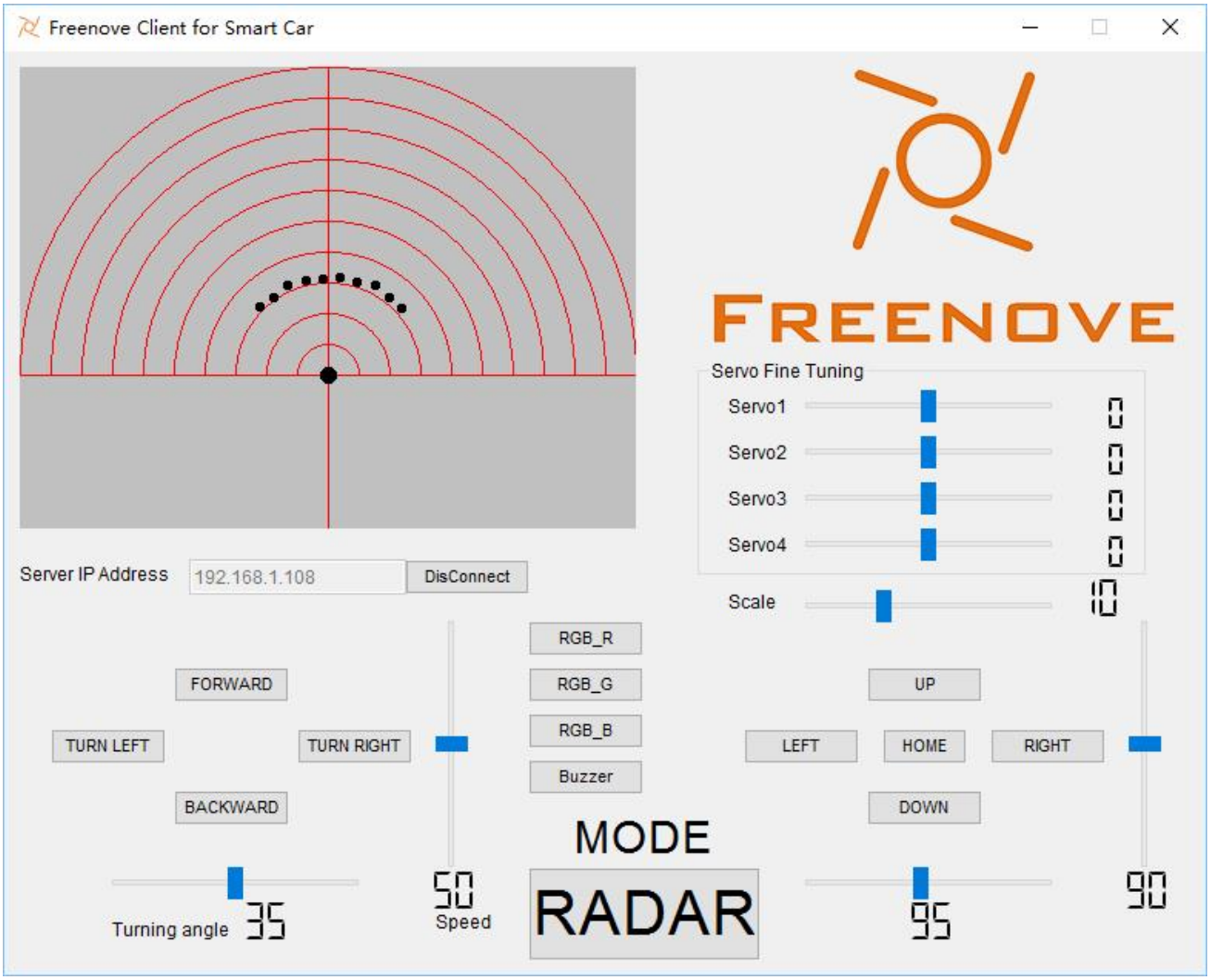
```
D:

cd \Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi\Client\

python main.py
```

Or enter path “D:\Freenove_Three_wheeled_Smart_Car_for_Raspberry_Pi\Client\” and double-click main.py with open way of Python3.exe. Then the following window interface appears.



In the edit box Server IP Address, input IP address of your RPi and click Connect. Make sure that server of your RPi have been opened already, and the switch S1 and Shield S2 have also been opened.

Then click on the MODE Button, the model will be switched to RADAR. Then the car enters ultrasonic radar mode, and constantly scan the distance, and the results will be displayed in the window. In this process, you can still control the car to move, and control the RGBLED module or Passive Buzzer module on the car.



Chapter 3 Part of the Code

In this chapter, we will explain the working principle and the key code of the whole project.

Server and Client

Server

The server is working on the RPi, and provides mjpg-streamer service and TCP service. And mjpg-streamer is for the transmission of camera data, TCP service for the transfer of the command.

mjpg-streamer

In the Camera_Server.py under "Server" folder, define a class Camera_Server, which inherits from the class threading.Thread. So when method .start () is called, Shell script file "Start_mjpg_Streamer.sh" is called, then the mjpg-streamer service is opened in a new thread. After mjpg-streamer starts, when the end signal (Ctrl-C, Ctrl-\, etc.) is received, it will exit, the thread will end.

Class Stop_Camera_Server is used to close the mjpg-streamer service. The class calls the Shell script file "Stop_mjpg_Streamer.sh" in a new thread, obtains the pid of the mjpg-streamer process, and then sends the end signal to end the service. And then the two threads will end.

Camera_Server.py

```
import threading
import os

class Camera_Server(threading.Thread):
    def camera_Http_Server(self):
        os.system("sudo /home/pi/Program/mjpg-streamer/Start_CMD_Suhayl.sh")

    def run(self):
        print "..............Camera server starting ......"
        self.camera_Http_Server()
        print "..............Camera server stop..........."
    def stop(self):
        os.system("sudo ./Stop_mjpg_Streamer.sh")

class Stop_Camera_Server(threading.Thread):
    def run(self):
        os.system("sudo ./Stop_mjpg_Streamer.sh")
```

About the command for opening “mjpg-streamer”, you can open the file “Start_mjpg_Streamer.sh” to view.

```
#!/bin/sh
curr_file=$0
cd "$(dirname "$curr_file")"
./mjpg_streamer -i "./input_uvc.so -y -d /dev/video0 -n -r 320*240 -f 30" -o
"./output_http.so -p 8090 -w ./www"
```

The meaning of the option parameters is as follows:

"-i", means the video input method, where the "input_uvc.so" is the input mode of USB Video Class (uvc).

The parameters can be specified as below:

```
 [-d | --device ].......: video device to open (your camera)
 [-r | --resolution ]...: the resolution of the video device,
                          can be one of the following strings:
                          QSIF QCIF CGA QVGA CIF VGA
                          SVGA XGA SXGA
                          or a custom value like the following
                          example: 640x480
 [-f | --fps ].........: frames per second
 [-y | --yuv ].........: enable YUYV format and disable MJPEG mode
 [-q | --quality ].....: JPEG compression quality in percent
                          (activates YUYV format, disables MJPEG)
 [-m | --minimum_size ].: drop frames smaller then this limit, useful
                          if the webcam produces small-sized garbage frames
                          may happen under low light conditions
 [-n | --no_dynctrl ]...: do not initalize dynctrls of Linux-UVC driver
 [-l | --led ].........: switch the LED "on", "off", let it "blink" or leave
                          it up to the driver using the value "auto"
```

"-o" means the video output method, where the "output_http.so" is used. The parameters can be specified as below:

```
 [-w | --www ].........: folder that contains webpages in
                          flat hierarchy (no subfolders)
 [-p | --port ].........: TCP port for this HTTP server
 [-c | --credentials ]...: ask for "username:password" on connect
 [-n | --nocommands ]....: disable execution of commands
```

The following is the file "Stop_mjpg_Streamer.sh", in which the pid of "mjpg_streamer" process is obtained, and then the kill command is used to end the process.

```
#!/bin/sh
pid="$(pgrep mjpg_streamer)"
echo "mjpg_streamer pid: "$pid
kill -9 $pid
```

TCP

In the mTCPServer.py under folder "Server", define a class mTCPServer, which is used to create a TCP monitor port, and, to forward the commands to Shield after receiving request from the client. Because the monitor port function is also blocked, it will be created in a new thread. You can open file mTCPServer.py to view the details.

mDEV

In the mDev.py under folder "Server", define class mDEV, which contains the control commands and methods for Shield. The following is the content of class mDEV.

```
import smbus
import time

def numMap(value,fromLow,fromHigh,toLow,toHigh):
    return (toHigh-toLow)*(value-fromLow) / (fromHigh-fromLow) + toLow

class mDEV:
    CMD_SERVO1      =   0
    CMD_SERVO2      =   1
    CMD_SERVO3      =   2
    CMD_SERVO4      =   3
    CMD_PWM1        =   4
    CMD_PWM2        =   5
    CMD_DIR1        =   6
    CMD_DIR2        =   7
    CMD_BUZZER      =   8
    CMD_IO1         =   9
    CMD_IO2         =   10
    CMD_IO3         =   11
    CMD_SONIC       =   12
    SERVO_MAX_PULSE_WIDTH = 2500
    SERVO_MIN_PULSE_WIDTH = 500
    Is_IO1_State_True = False
    Is_IO2_State_True = False
    Is_IO3_State_True = False
    Is_Buzzer_State_True = False
    def __init__(self,addr=0x18):
        self.address = addr #default address of mDEV
        self.bus=smbus.SMBus(1)
        self.bus.open(1)
    def i2cRead(self,reg):
        self.bus.read_byte_data(self.address,reg)

    def i2cWrite1(self,cmd,value):
        self.bus.write_byte_data(self.address,cmd,value)
```

```
    def i2cWrite2(self,value):
        self.bus.write_byte(self.address,value)

    def writeReg(self,cmd,value):
        try:
            self.bus.write_i2c_block_data(self.address,cmd,[value>>8,value&0xff])
        except Exception,e:
            print Exception,"I2C Error :",e

    def readReg(self,cmd):
        [a,b]=self.bus.read_i2c_block_data(self.address,cmd,2)
        return a<<8 | b

    def getSonicEchoTime():
        SonicEchoTime = mdev.readReg(mdev.CMD_SONIC)
        return SonicEchoTime

    def getSonic(self):
        SonicEchoTime = mdev.readReg(mdev.CMD_SONIC)
        distance = SonicEchoTime * 17.0 / 1000.0
        return distance
    def setShieldI2cAddress(self,addr):#addr: 7bit I2C Device Address
        if (addr<0x03) or (addr > 0x77) :
            return
        else :
            mdev.writeReg(0xaa, (0xbb<<8)|(addr<<1))
```

The following table is corresponding Shield interface of the command. You can operate them through the member function writeReg and readReg.

Command	Interface	Read/Write	Valid value
CMD_SERVO1	Servo1	W	0-20000
CMD_SERVO2	Servo2	W	0-20000
CMD_SERVO3	Servo3	W	0-20000
CMD_SERVO4	Servo4	W	0-20000
CMD_PWM1	Motor1 Speed	W	0-1000
CMD_PWM2	Motor2 Speed	W	0-1000
CMD_DIR1	Motor1 Steering	W	0 or non-0
CMD_DIR2	Motor2 Steering	W	0 or non-0
CMD_BUZZER	Buzzer	W	0-65535
CMD_IO1	IO1	W	0 or non-0
CMD_IO2	IO2	W	0 or non-0
CMD_IO3	IO3	W	0 or non-0
CMD_SONIC	TRIG/ECHO	R	

Default I2C address of the Shield is 0x18 (7bit). Member function setShieldI2cAddress (self, addr) is used to modify the I2C address of Shield. Don't change its address unless it conflicts with the I2C address of your other device. Because detection range of the I2CTool is 0x03-0x77, so the range of parameter "addr" is limited to 0x03-0x77. It is invalid when beyond this range. When the I2C address is modified, the Shield need to be restarted. when there is a need, change the parameter addr of the class constructor to the new I2C address to facilitate use later. Please use this function with caution.

```
    def __init__(self,addr=0x18):
    ......
    def setShieldI2cAddress(self,addr): #addr: 7bit I2C Device Address
        if (addr<0x03) or (addr > 0x77) :
            return
        else :
            mdev.writeReg(0xaa, (0xbb<<8)|(addr<<1))
```

Client

Connect the Client to the server through the TCP port. Then define class TCPClient in the TCPClient.py under "Client" folder, which contains the method for connecting to server and sending data.

```
from socket import *

class TCPClient:
    #HOST = '127.0.0.1'
    HOST = '192.168.1.108'
    PORT = 12345
    BUFSIZ = 1024
    ADDR = (HOST, PORT)

    def __init__(self):
        #self.client = socket(AF_INET, SOCK_STREAM)
        pass

    def connectToServer(self, address = ADDR):
        self.client = socket(AF_INET, SOCK_STREAM)
        self.client.connect(address)

    def disConnect(self):
        try:
            self.client.close()
        except Exception, e:
            print Exception, "Disconnect error:", e

    def sendData(self, data):
```

```
        try:
            self.client.send(data)
        except Exception, e:
            print Exception, "Send TCP Data error:", e

    def recvData(self):
        try:
            return self.client.recv(self.BUFSIZ)
        except Exception, e:
            print Exception, "Recv TCP Data error:", e
```

In "main.py", monitor the action of the control and keyboard, and send a command to the server. View "main.py" for detailed information.

In "Config.txt", preserve the IP address of successful connection to the server last time, and load it the next time start the client. This enables you to connect to the server quickly without entering the IP address of the server in the edit box each time.

Chapter 4 Contributions of other developers

We are very grateful to all developers for their contributions to this product. We are very welcome and highly praised for such work. We will include the excellent contents from developers in main branch of this tutorial.

Extensions for Scratch 2

This is an extension for Scratch2, with which you can operate this car on Scratch2.
Author: Ruairi

How to use?

How to install the extension into Scratch 2:

1: Extract the Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi-master.zip file
2: cd into the Folder you extracted it to
3: run the following commands:

```
sudo cp ./Client/Scratch/piCar.js /usr/lib/scratch2/scratch_extensions
sudo cp ./Client/html/imgs/car_photo.jpg /usr/lib/scratch2/medialibrarythumbnails
```

4: NOTE: **only run the following command if you have only the default extensions already loaded in Scratch**
If you have extensions already loaded, then add the last line of this extensions.json file into the one on your pi before the last "]"

```
sudo cp ./Client/Scratch/extensions.json /usr/lib/scratch2/scratch_extensions
```

5: cd into the Client folder and run the following command:

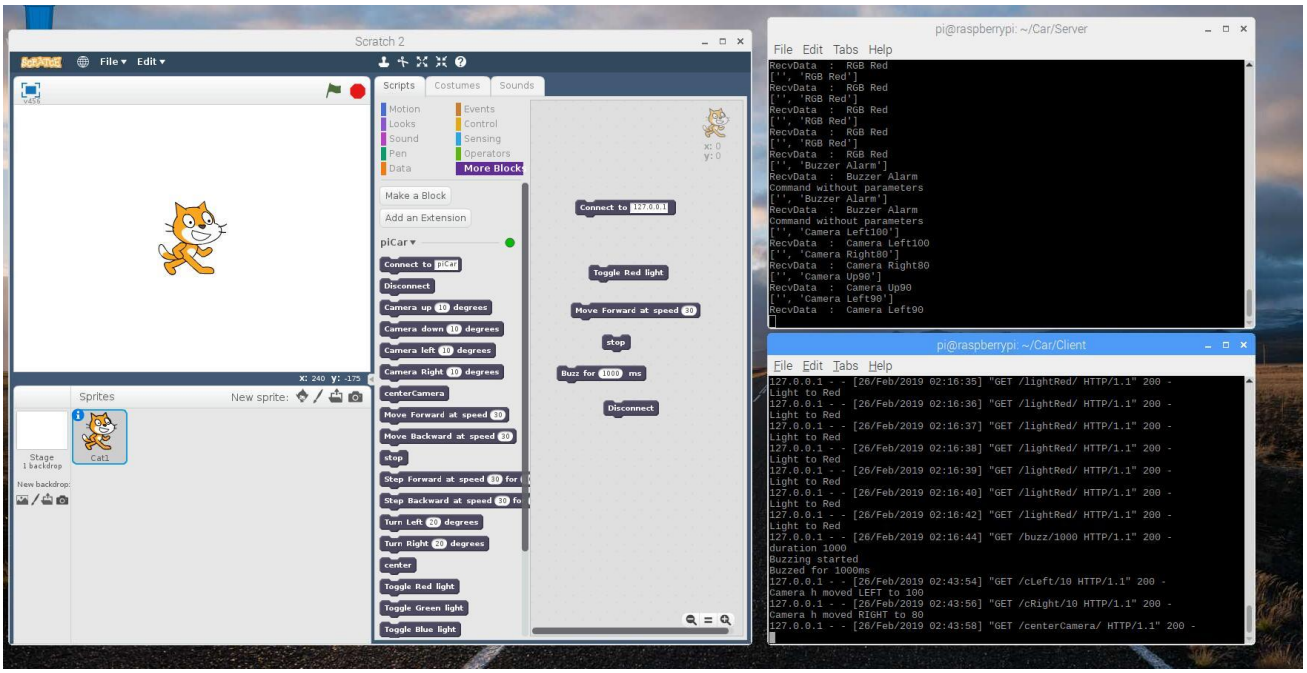
```
python ScratchServer.py
```

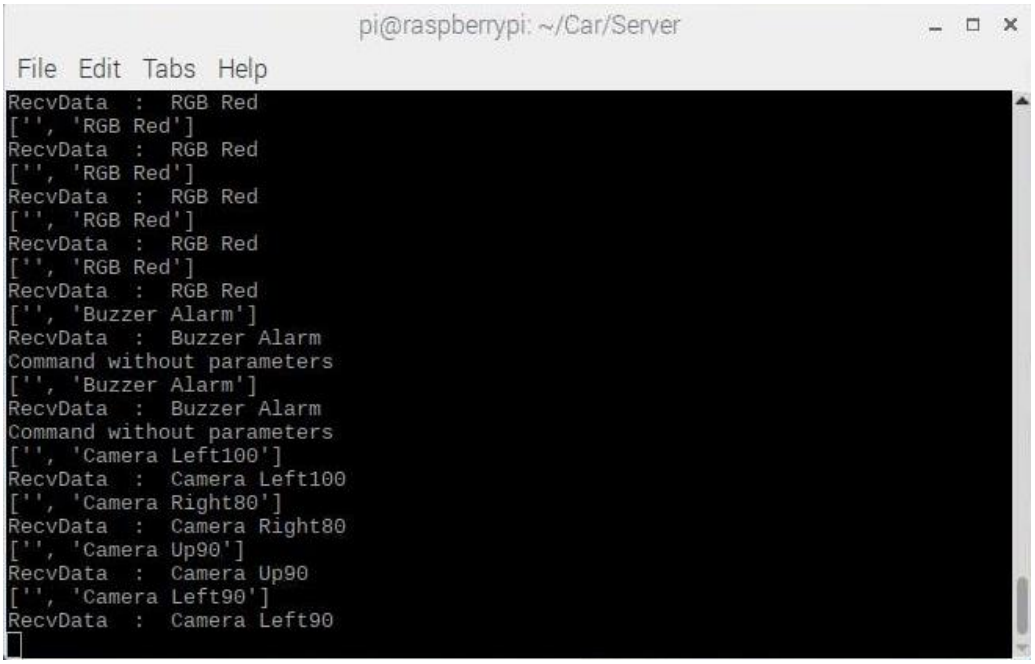
6: Run Scratch 2
7: Click "More Blocks"
8: Click "Add an extension"
9: You should now see the "Freenove PiCar" in the list of extensions, double click it to add them.

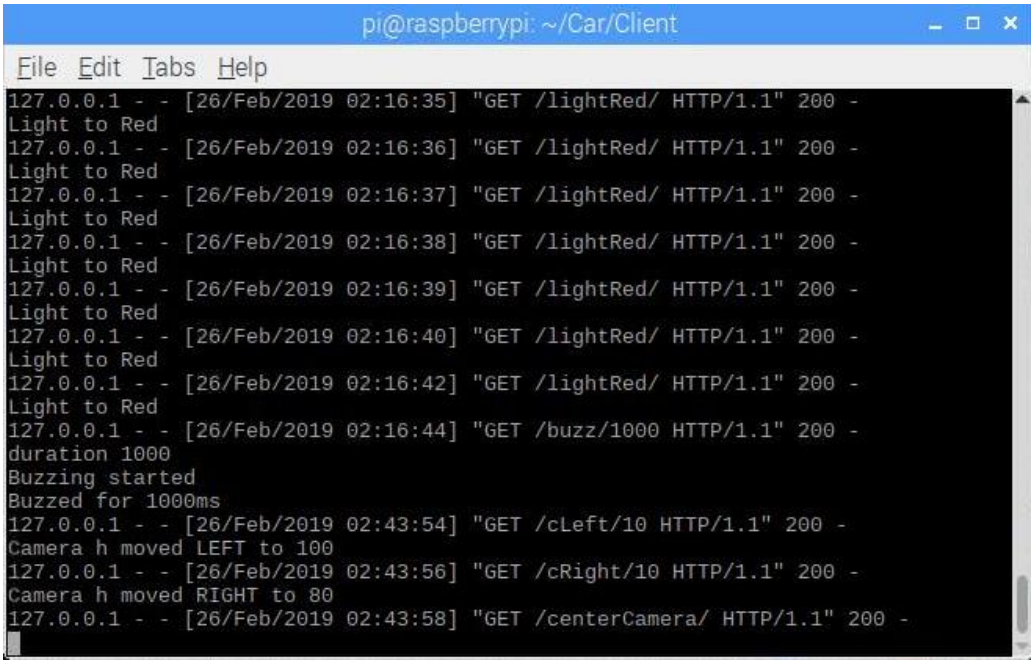
To use the blocks, always remember to run the "connect to" block first and finish off with a disconnect block.

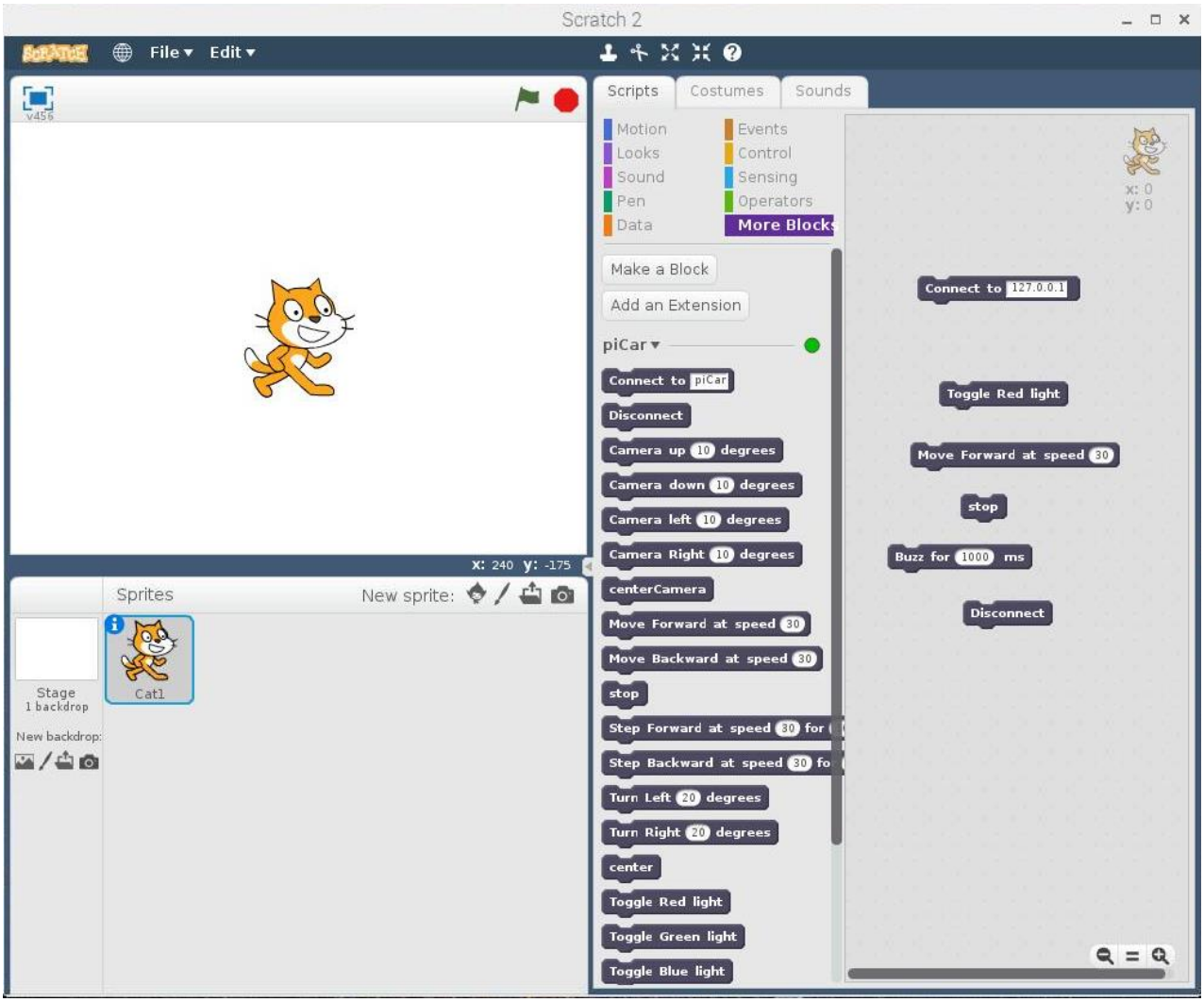
If you don't disconnect you may not be able to connect to you car from another app (e.g. the android app)
Enjoy!

Screenshot of running






pi@raspberrypi: ~/Car/Client
File Edit Tabs Help
127.0.0.1 - - [26/Feb/2019 02:16:35] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:36] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:37] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:38] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:39] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:40] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:42] "GET /lightRed/ HTTP/1.1" 200 -
Light to Red
127.0.0.1 - - [26/Feb/2019 02:16:44] "GET /buzz/1000 HTTP/1.1" 200 -
duration 1000
Buzzing started
Buzzed for 1000ms
127.0.0.1 - - [26/Feb/2019 02:43:54] "GET /cLeft/10 HTTP/1.1" 200 -
Camera h moved LEFT to 100
127.0.0.1 - - [26/Feb/2019 02:43:56] "GET /cRight/10 HTTP/1.1" 200 -
Camera h moved RIGHT to 80
127.0.0.1 - - [26/Feb/2019 02:43:58] "GET /centerCamera/ HTTP/1.1" 200 -


Scratch 2
File Edit
Scripts
Costumes
Sounds
Motion
Looks
Sound
Pen
Data
Events
Control
Sensing
Operators
More Blocks
Make a Block
Add an Extension
piCar
Connect to piCar
Disconnect
Camera up 10 degrees
Camera down 10 degrees
Camera left 10 degrees
Camera Right 10 degrees
centerCamera
Move Forward at speed 30
Move Backward at speed 30
stop
Step Forward at speed 30 for
Step Backward at speed 30 fo
Turn Left 20 degrees
Turn Right 20 degrees
center
Toggle Red light
Toggle Green light
Toggle Blue light
Connect to 127.0.0.1
Toggle Red light
Move Forward at speed 30
stop
Buzz for 1000 ms
Disconnect
x: 240 y: -175
Sprites
New sprite:
Stage
1 backdrop
New backdrop:
Cat1

Chapter 5 Free your 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 this car, you can follow this chapter.

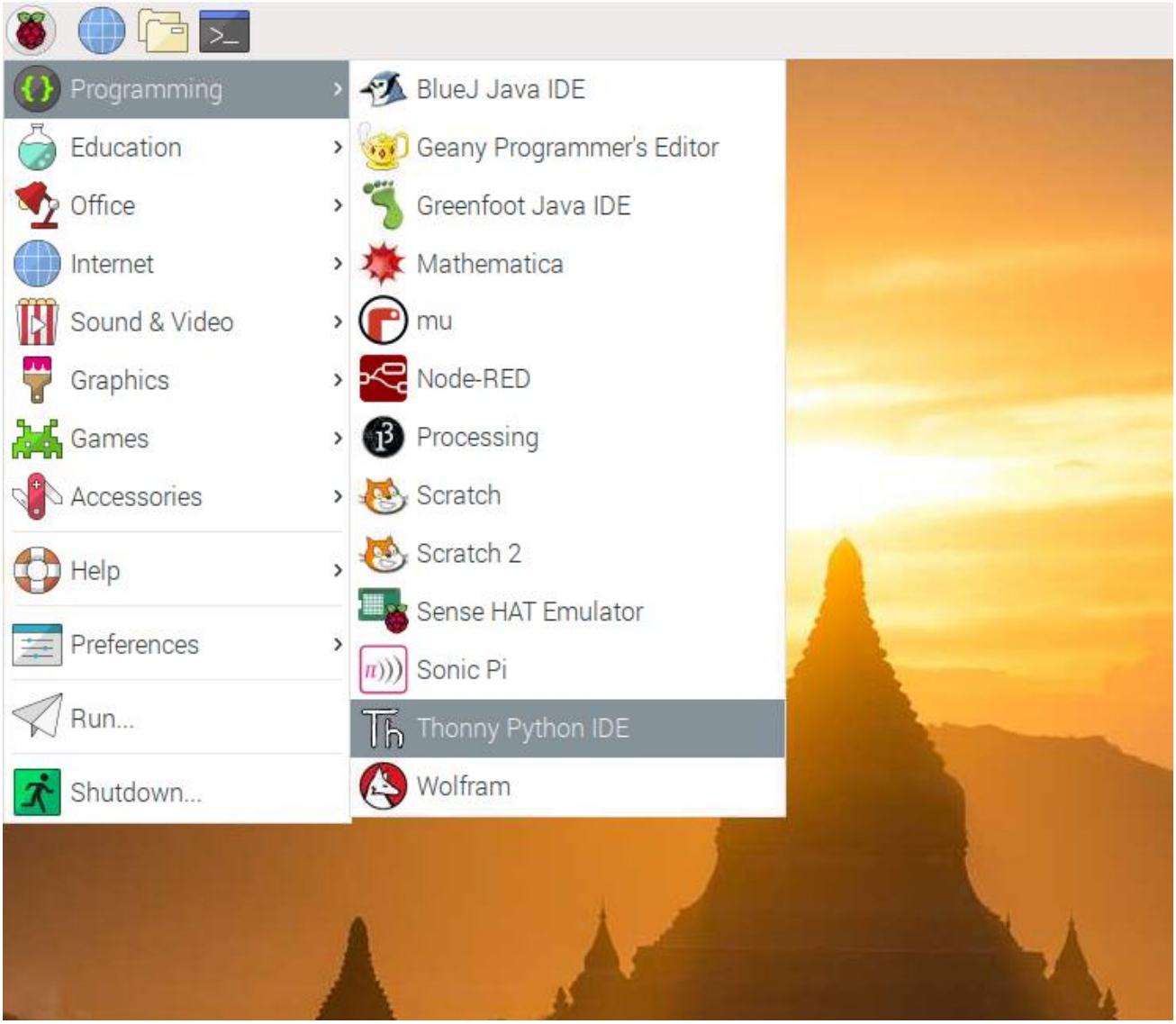
The car program is based on python3. If your default python is python2, please set python3 as default python.

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

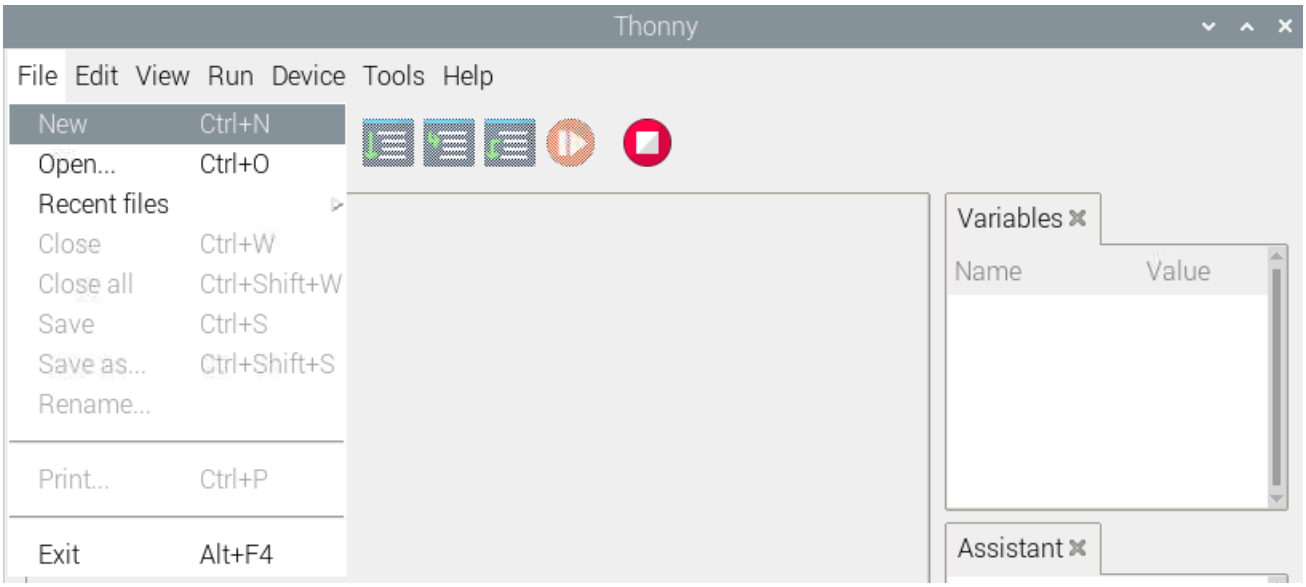
https://python.swaroopch.com/basics.html

Program

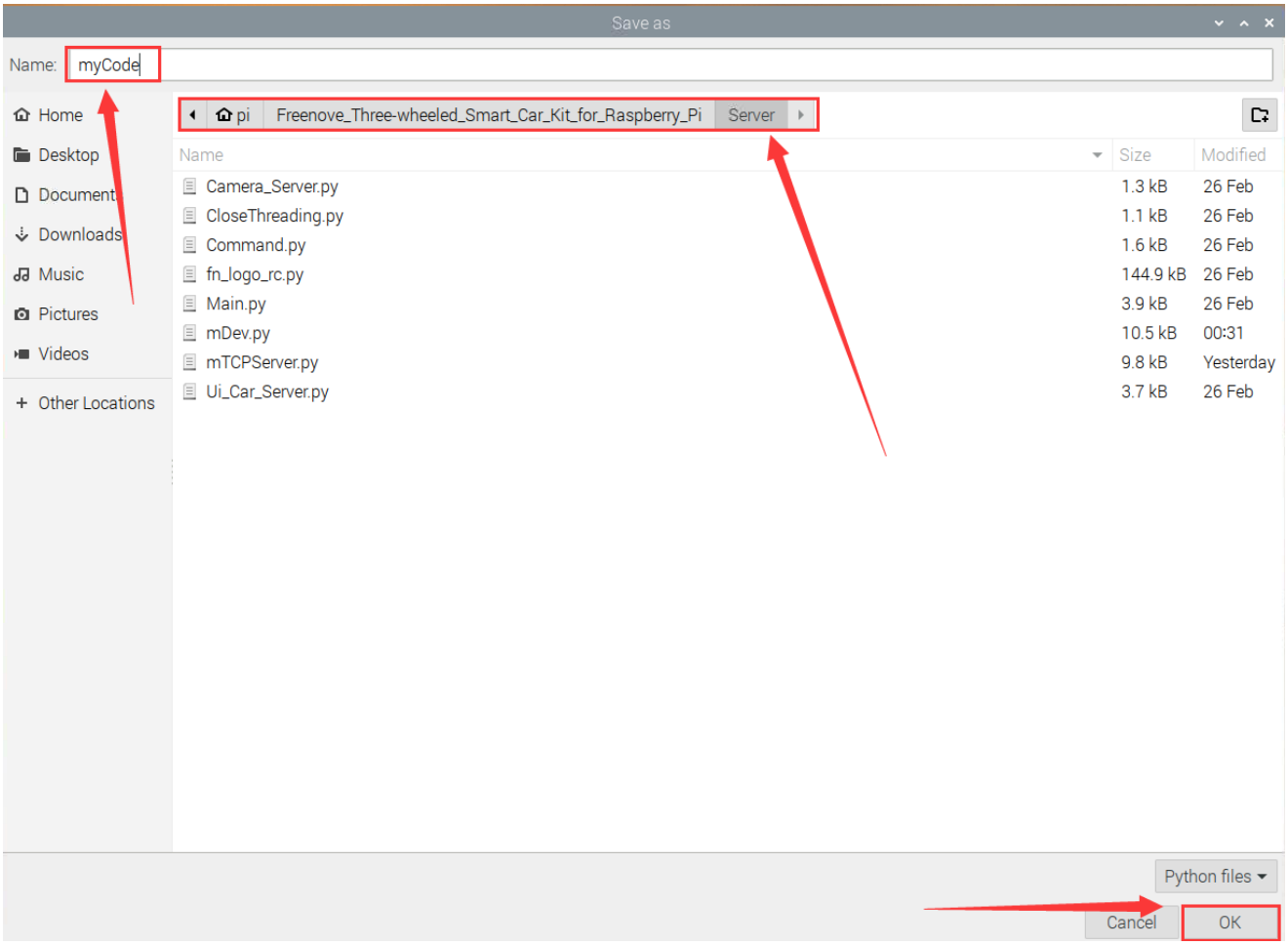
First, open Thonny Python IDE. Thonny is a software easy to use for beginners.



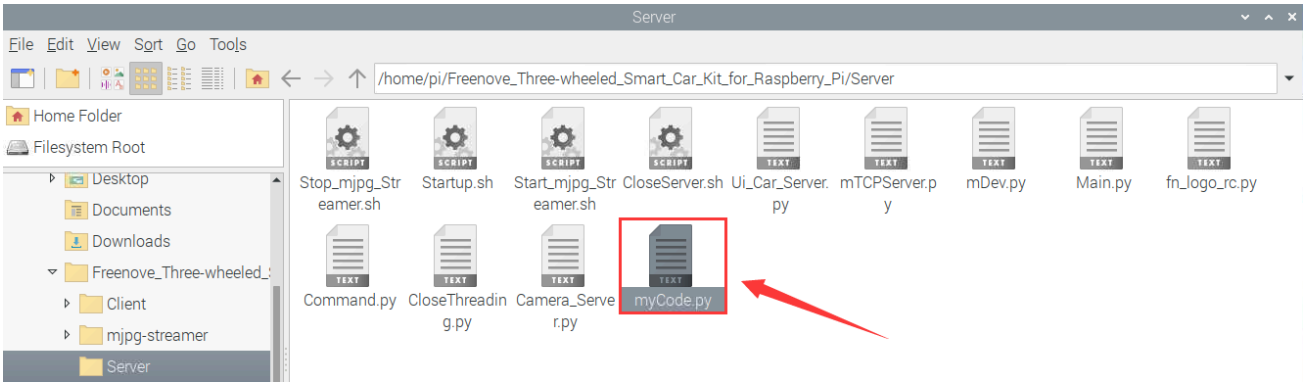
Create a new file.



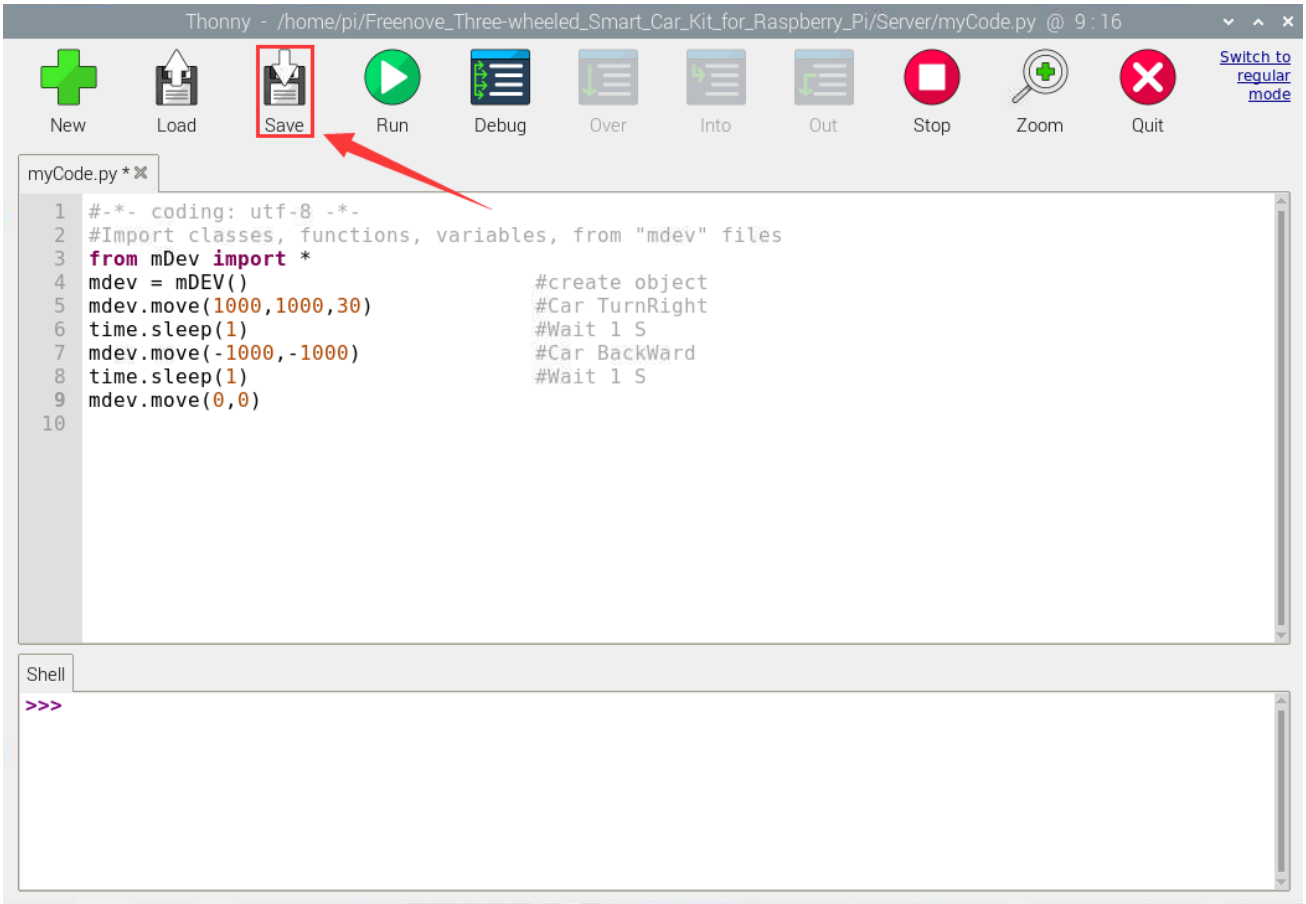
Name the file "myCode" as an example. And save it in Server folder of the car code in Raspberry Pi.
Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server



Open the Server folder of the car code. You will see the code you created.



Write your own code in myCode.py like below as an example. Then save as below:

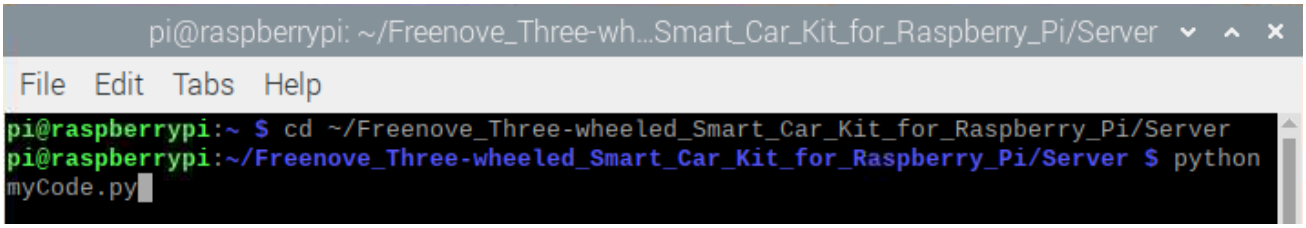


Type following commands to enter folder of myCode.py.

```
cd ~/Freenove_Three-wheeled_Smart_Car_Kit_for_Raspberry_Pi/Server
```

Run myCode.py

```
python myCode.py
```



Result:

The car will turn right with 30° for 1s. Then move back for 1s. Then stop.

Code Example

Motor

```
#-*- coding: utf-8 -*-
#Import classes, functions, variables, from "mdev" files
from mDev import *
mdev = mDEV()                          #create object
mdev.move(1000,1000,30)                #Car TurnRight
time.sleep(1)                          #Wait 1 S
mdev.move(-1000,-1000)                 #Car BackWard
time.sleep(1)                          #Wait 1 S
mdev.move(0,0)
```

Ultrasonic module

```
#-*- coding: utf-8 -*-
#Import classes, functions, variables, from "mdev" files
from mDev import *
mdev = mDEV()                          #create object
#Obtain the distance from the obstacle in front to the ultrasound
distance=mdev.getSonic()
print("Sonic: %.2f cm"%distance)     #Print once
```

Buzzer

```
#-*- coding: utf-8 -*-
#Import classes, functions, variables, from "mdev" files
from mDev import *
mdev = mDEV()                          #create object
#Buzzer sounds for 2 seconds
mdev.setBuzzer(2000)
time.sleep(2)                          #Wait 2 S
mdev.setBuzzer(0)
```

Led

```
#-*- coding: utf-8 -*-
#Import classes, functions, variables, from "mdev" files
from mDev import *
mdev = mDEV()                          #create object
#The order of LED lighting is red-green-blue-white
mdev.setLed(1,0,0)
time.sleep(1)                          #Wait 1 S
mdev.setLed(0,1,0)
time.sleep(1)
mdev.setLed(0,0,1)
time.sleep(1)
mdev.setLed(1,1,1)
time.sleep(1)
mdev.setLed(0,0,0)
```

Servo

```
#-*- coding: utf-8 -*-
#Import classes, functions, variables, from "mdev" files
from mDev import *
mdev = mDEV()                          #create object
#The servo2  rotates back and forth between 40 and 140 degrees
while True:
    for i in range(40,140,1):
        mdev.setServo('2',i)
        time.sleep(0.005)
    for i in range(140,40,-1):
        mdev.setServo('2',i)
        time.sleep(0.005)
```

Related Functions

The functions listed below are included the "mDev.py" file. If you want to see more detailed code content, please open the mDev.py file to check.

Function	Description
move(left_pwm, right_pwm, steering_angle)	This function is used to control the movement of the car and has three input parameters. The first parameter is "left_pwm". Control the PWM of the left wheel of the car. The range is -1000~1000. The positive value makes the wheels rotates positively, and the negative value make the wheels reverse. The larger the absolute value of the number, the faster the speed. The second parameter is "right_pwm". Control the PWM of the right wheel of the tricycle, the range is -1000~1000. The positive value makes the wheels rotates positively, and the negative value make the wheels reverse. The larger the absolute value of the number, the faster the speed. The third parameter is "steering_angle". Controls the turning angle of the car, the range is 0~180. The default is 90 degrees and keep the car going straight.
getSonic()	Returns the distance from the obstacle in front to the ultrasonic module, unit CM.
setBuzzer(pwm)	This function is used to control the buzzer and has one input parameter. The parameter "pwm" controls the PWM signal output by the buzzer, the range is 0-65535. The larger the "pwm", the lager the tone.
setLed(R,G,B)	This function is used to control the color of Led. It has three input parameters. When the parameter "R" is 1, the red channel of the Led is on. When it is 0, the red channel is off. When the parameter "G" is 1, the green channel of the Led is on. When it is 0, the green channel is off. When the parameter "B" is 1, the blue channel of the Led is on. When it is 0, the blue channel is off.
setServo(index,angle)	This function is used to control the servo and has two input parameters. The first parameter "index" selects the servo number to be controlled, which are "1", "2", "3", "4". The second parameter "angle" sets the angle of the servo, the range is 0~180°.

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 so on, please feel free to contact us at support@freenove.com 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 other interesting products in science and technology, please continue to focus on our website (www.freenove.com). We will continue to launch cost-effective, innovative and exciting products.

Thank you again for choosing Freenove products.