Preface

About SunFounder

SunFounder is a company focused on STEAM education with products like open source robots, development boards, STEAM kit, modules, tools and other smart devices distributed globally. In SunFounder, we strive to help elementary and middle school students as well as hobbyists, through STEAM education, strengthen their hands-on practices and problem-solving abilities. In this way, we hope to disseminate knowledge and provide skill training in a full-of-joy way, thus fostering your interest in programming and making, and exposing you to a fascinating world of science and engineering. To embrace the future of artificial intelligence, it is urgent and meaningful to learn abundant STEAM knowledge.

About the PiCar-V

As an upgraded version, PiCar-V (Smart Video Car V2.0 for Raspberry Pi) is also suitable for the Raspberry Pi 3 module B, 3 module B+ and 4 module B, as the V1.0.

The car PiCar-V is equipped with a wide-angle USB webcam, three whole new circuit boards with less but more simple wiring, some acrylic plates with an adjusted structure, and new code suitable for almost all platforms to control the car.

In this book, we will show you how to build the PiCar-V via description, illustrations of physical components, in both hardware and software respects. You will enjoy learning how all this work. You may visit our website www.sunfounder.com to download the related code and view the user manual on **LEARN** -> **Get Tutorials** and watch related videos under **VIDEO**, or clone the code on our page of github.com at https://github.com/sunfounder/SunFounder_PiCar-V.

You are welcomed to post requests and issues on our page on Github.

Free Support

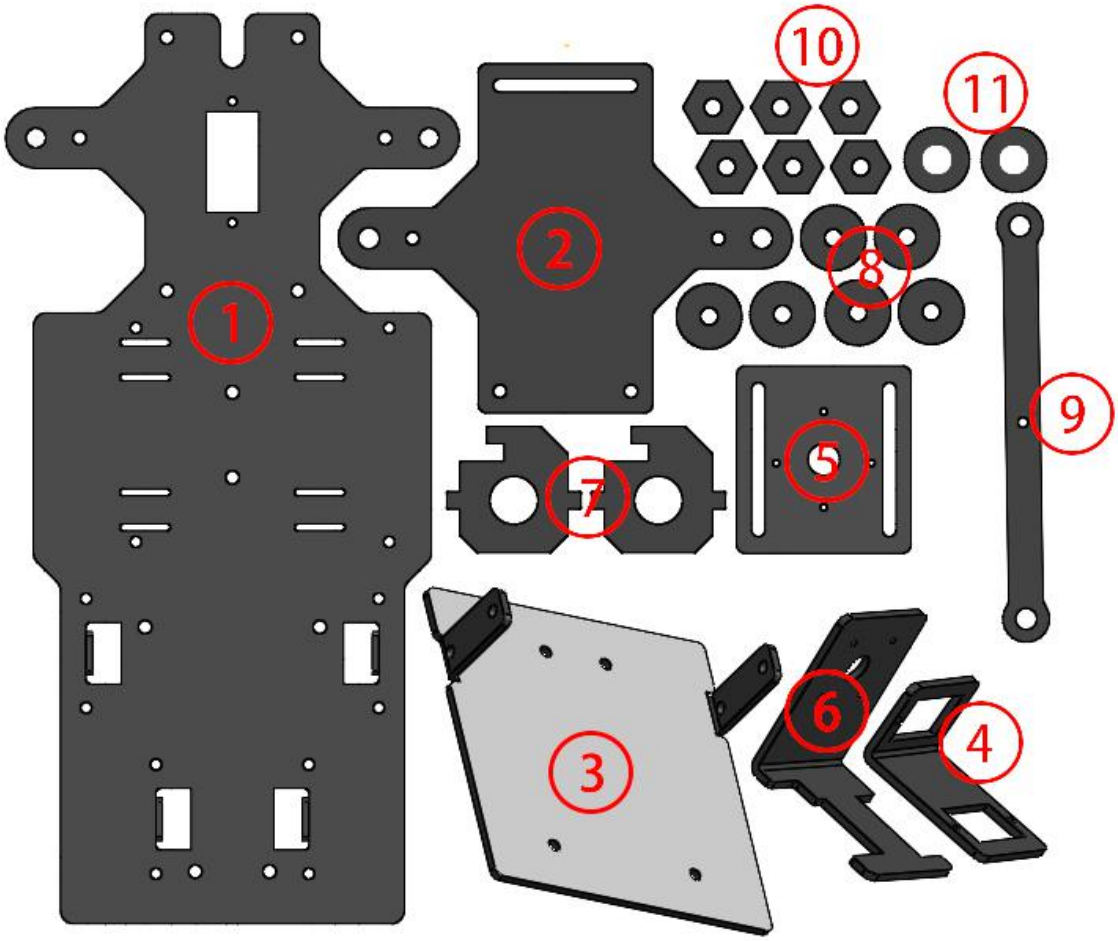
If you have any **TECHNICAL questions**, add a topic under **FORUM** section on our website and we'll reply as soon as possible.

For **NON-TECH questions** like order and shipment issues, please **send an email to service@sunfounder.com**. You're also welcomed to share your projects on FORUM.

Contents

Components List

Plates



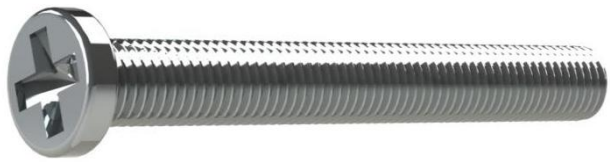
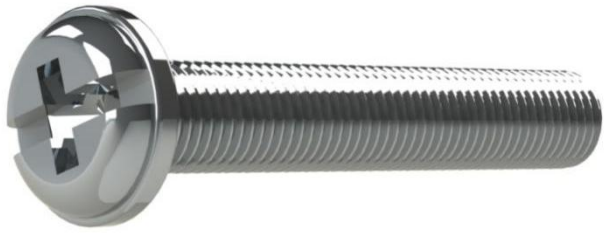
1. Upper Plate x 1
2. Front Half Plate x 1
3. Back Half Plate x 1
4. Pan-and-tilt Plate x 1
5. Pan-and-tilt Base Plate x 1
6. Camera Mount Plate x 1
7. Servo Connector Plate x 2
8. Bearing Shield x 6
9. Servo Linkage Plate x 1
10. Hex Front Wheel Fixing Plate x 6
11. Gasket Plate x 2

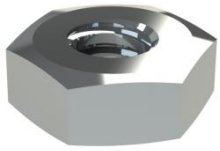
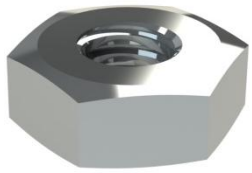
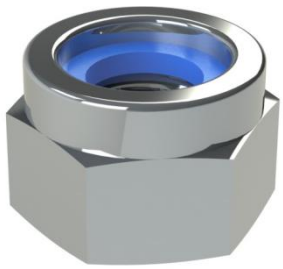
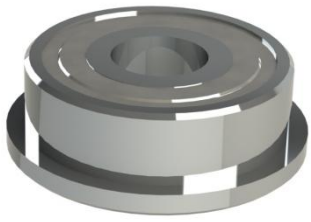
Note: The Bearing Shield is similar to the Gasket Plate, but the aperture of Gasket Plate is large than the Bearing Shield.

Servo x 3

1. Servo
2. 1-arm Rocker Arm
3. arm Rocker Arm
4. 4-arm Rocker Arm
5. Rocker Arm Fixing Screw
6. Rocker Arm Screw

Mechanical Fasteners

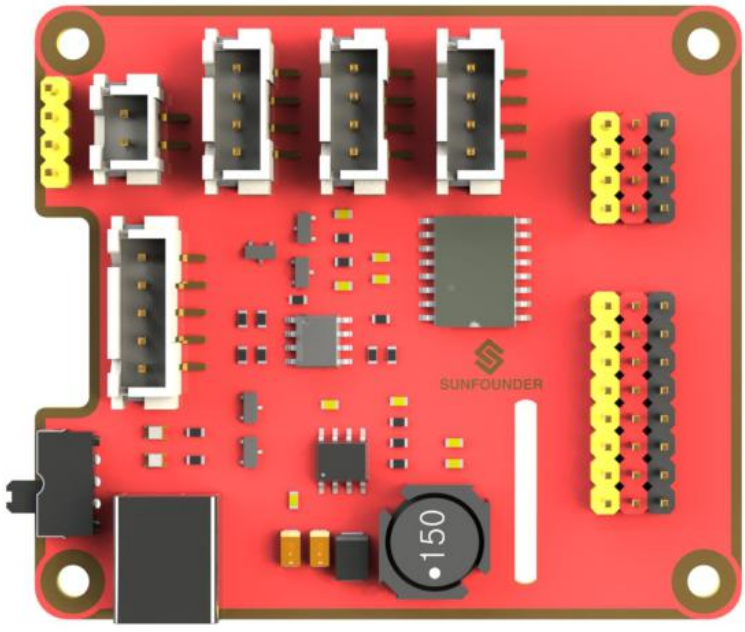
Name	Component	Qty.
M1.6x4 Self-tapping Screw		8
M2x8 Screw		6
M2.5x6 Screw		4
M2.5x12 Screw		8
M3x8 Screw		8
M3x8 Countersunk Screw		2
M3x10 Screw		7
M3x25 Screw		4
M4x25 Screw		2

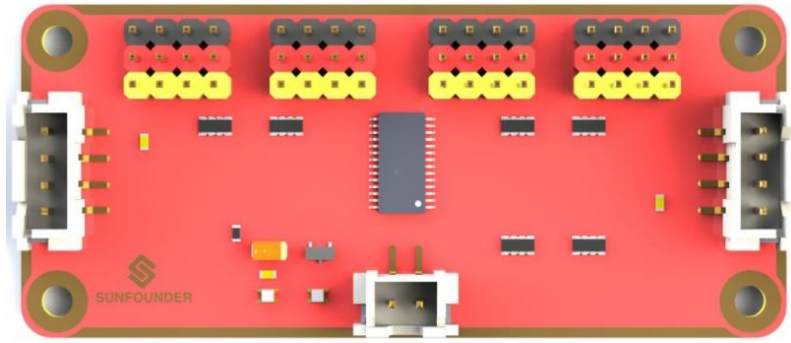
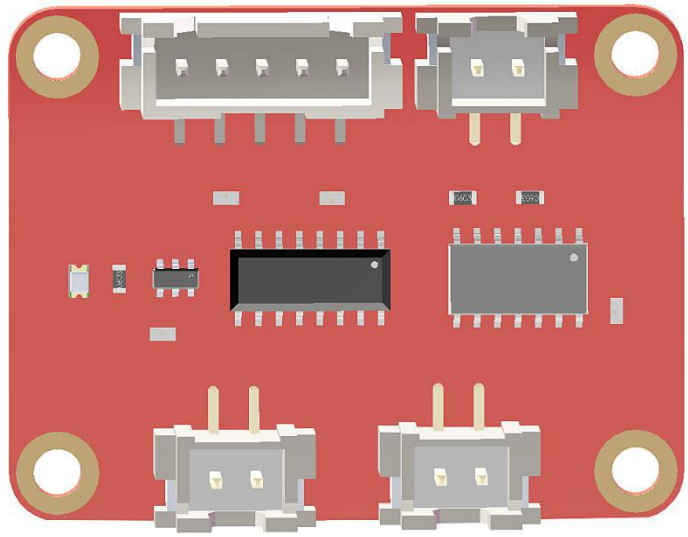
M2 Nut		6
M2.5 Nut		12
M3 Nut		21
M4 Self-locking Nut		2
M2.5x8 Copper Standoff		8
M3x25 Copper Standoff		8
4x11x4 F694ZZ Flange Bearing		2

Wires

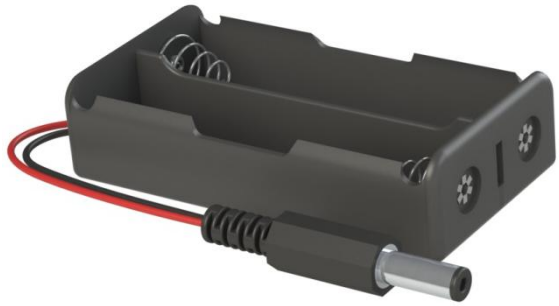
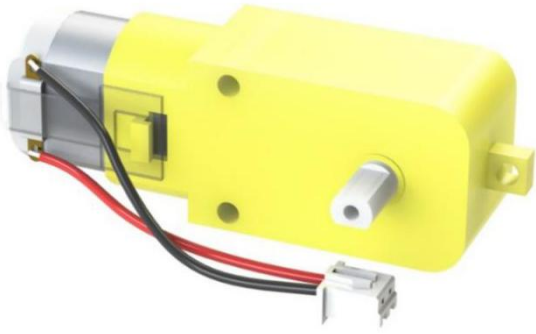
100mm HX2.54 5-Pin Jumper Wire		1
50mm HX-2.54 4-Pin Jumper Wire		1
50mm HX-2.54 2-Pin Jumper Wire		1
100mm HX-2.54 2-Pin Jumper Wire		1

PCB

Robot HATS		1

PCA9685 PWM Driver		1
Motor Driver Module		1

Other Components

2x18650 Battery Holder		1
DC Gear Motor		2

120° Wide-angle USB Camera		1
Rear Wheel		2
Front Wheel		2
Ribbon (30cm)		1

Tools

Cross Screwdriver		1
Cross Socket Wrench		1
M2.5/M4 Small Wrench		1
M2/M3 Small Wrench		1

Introduction

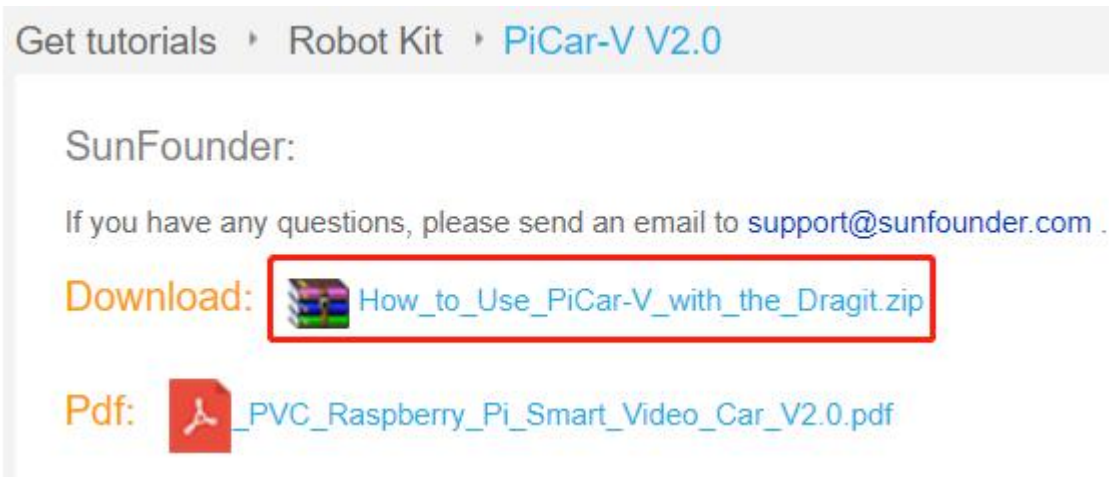
The PiCar-V is developed based on the previous Smart Video Car Kit. It’s equipped with the new PCA9685 PWM driver, smaller and better-performing DC motor driver module, a Robot Hat with power and interface integrated, a wider-angle camera with clearer vision, some better wheels and tires, and so on. Later we'll check more details.

The basic functions of this new version still stay the same, including controlling the PiCar-V remotely and streaming the video data it captures back. Then what's new? Let's unveil the mysteries!

Visual Programming with Dragit

Instead of coding the Sensor Kit directly in Python, you can also do it in a graphical/visual programming software - Dragit, developed based on Snap! (check http://snap.berkeley.edu/ for more). What's good about Dragit is that, the programming process is visible with graphics. So you can just drag and drop the blocks (unit in the software) to make the code in each lesson, no need of the knowledge of any programming languages or syntaxes.

You can go to our official website www.sunfounder.com to download **How to use PiCar-V with the Dragit.zip** by clicking **LEARN** -> **Get Tutorials** ->**Robot Kit** -> **PiCar-V V2.0**



Building the Car

Are you excited when you open the box and see so many components? Keep your patience and take it easy. Please note that some details in the following steps need careful observation. You should double-check your work based on the figures in the manual after finishing each step. let's start!

Front Wheels

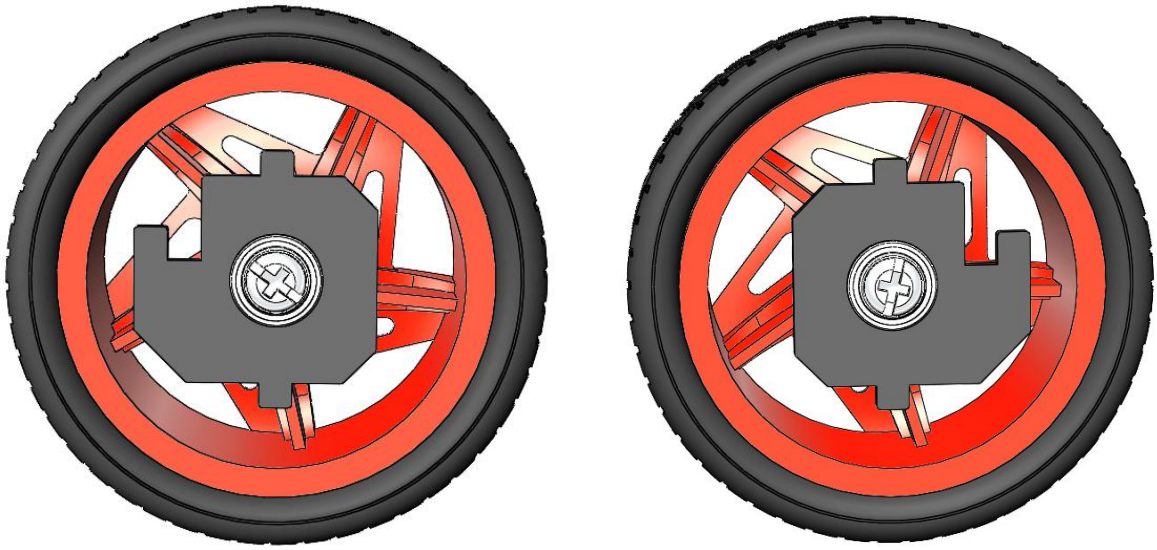
Insert an **M4x25 screw** through a **Flange Bearing** (pay attention to the direction – the flange near the cap of the screw), a **Steering Connector Plate**, 3 **Bearing Shields**, 3 **Hex Front Wheel Fixing Plates**, and a **front wheel**, into an **M4 Self-locking Nut** as shown below:

You can use the Cross Socket Wrench to secure the **M4 Self-locking Nut**, then use the screwdriver to tighten the **M4x25 screw.**

Note:

The Self-locking Nut should be screwed tight enough. It would be better to tighten the screw until the wheel and Steering Connector cannot move first, then loosen the screw a little, so that the Steering Plate can just move. Thus, the wheel can turn flexibly when the connection would not be too loose.

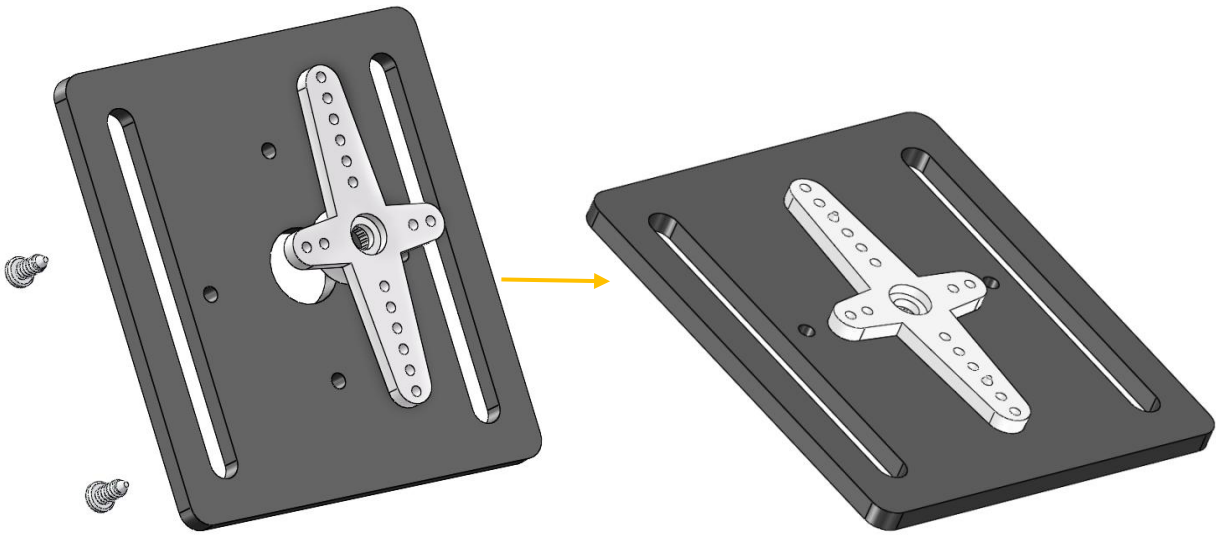
Assemble the other front wheel in the same way, but bear in mind the Steering Connector plate on the wheel should be symmetric with the previous one.



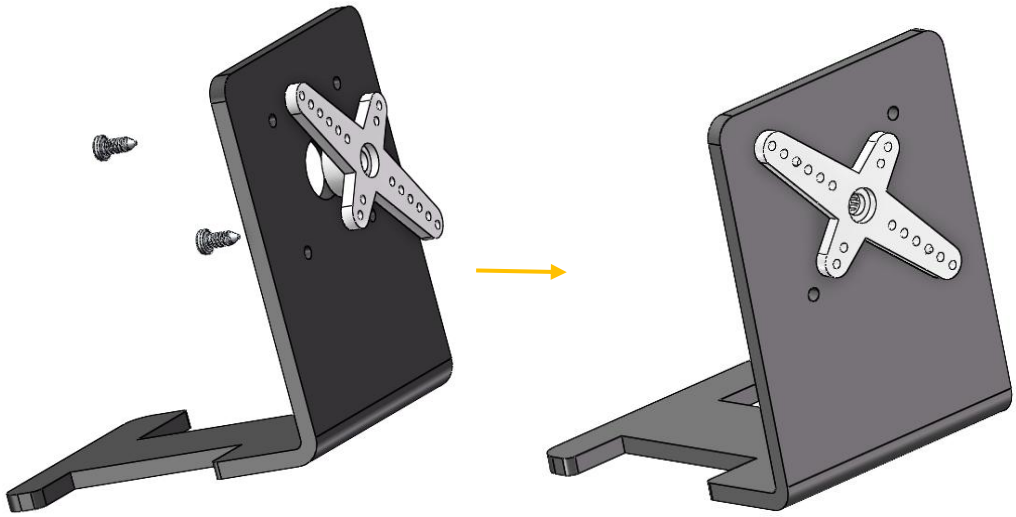
Now two front wheels have finished assembly.

Pan-and-Tilt

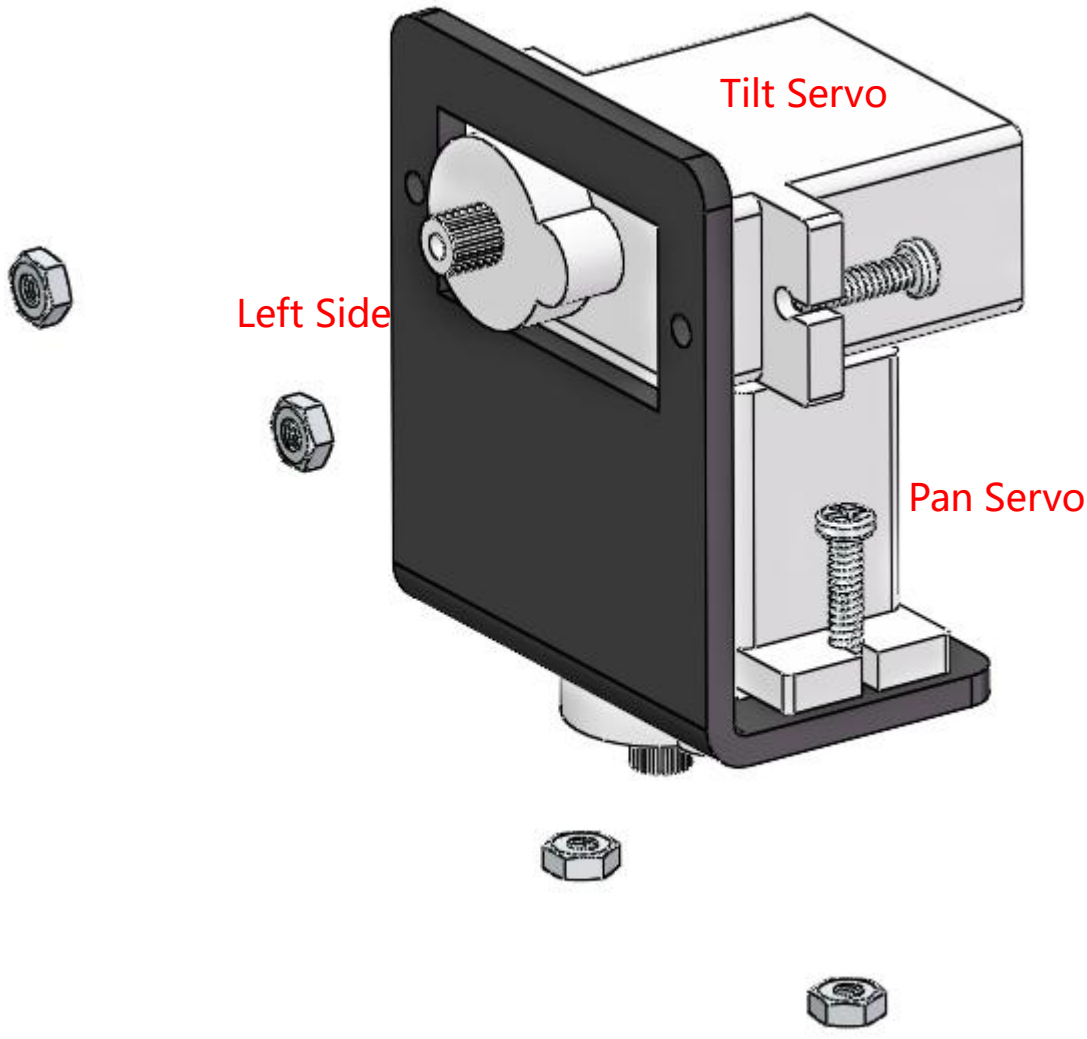
Take out the **Cross rocker arm** and mount it onto the **Pan-and-tilt Base Plate** with four **M1.6x4 screws** (into the hole like this). Pay attention to the holes on the round rocker arm to be fastened.



Do the same operation to the **Camera Mount Plate**.



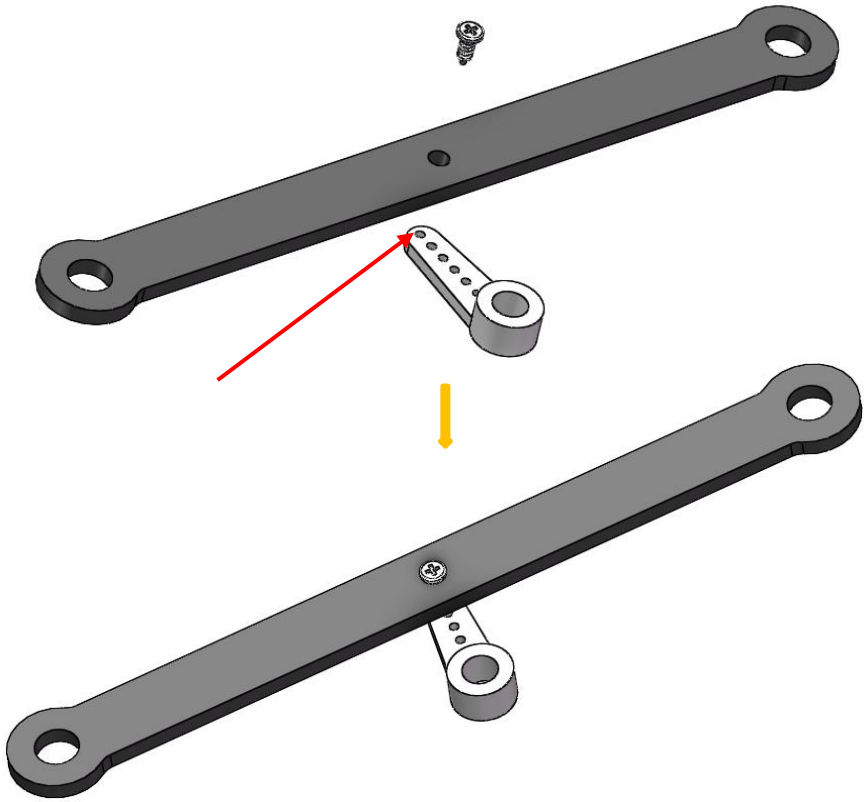
Assemble the two servos to the **Pan-and-tilt Plate** with four **M2x8 screws** and the **M2 nuts** (The 2 servo shaft are close to the left side):



Steering Part

Connect the **Steering Linkage** and the **1-arm Rocker Arm** with the **Rocker Arm Screw** (a longer one in the servo package).

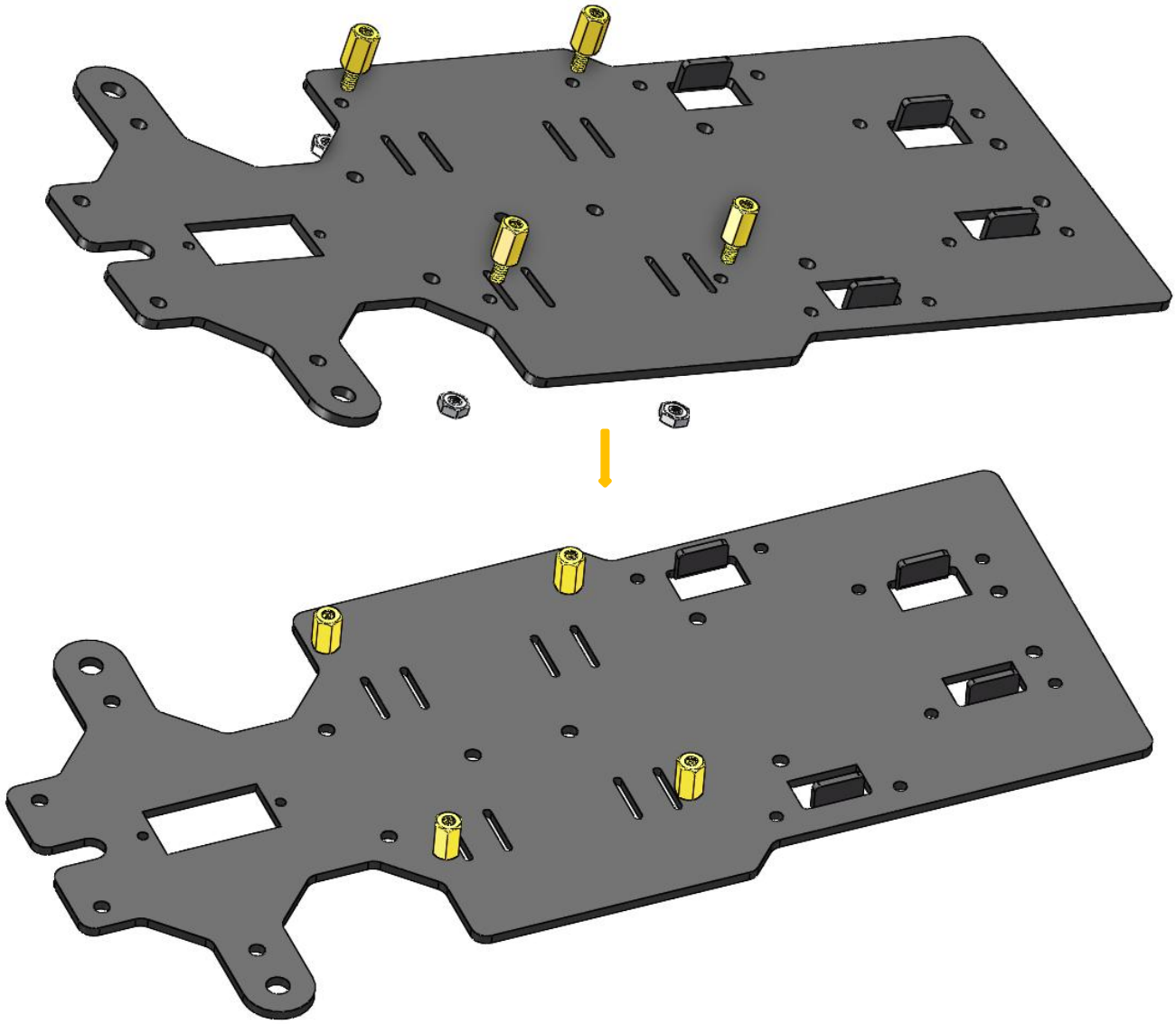
Note: Insert it into the FIRST hole of the arm (as indicated by the arrow below) which is the farthest from the gears. Since the screw is larger than the hole, you should try to screw it hardly so as tight to the arm. Don't worry of the arm which is soft.



And also fasten them as tightly as possible, and then loosen the screw a little so the Steering Linkage can move flexibly.

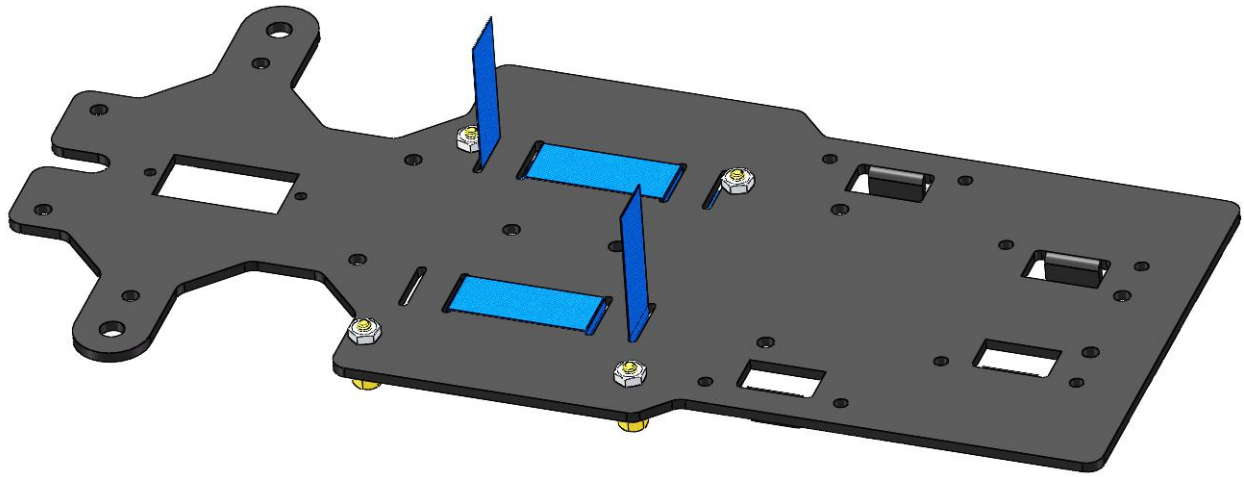
Upper Plate

Mount the **M2.5x8 copper standoffs** and **M2.5 nuts** into the **upper plate** first. Pay attention that the side the protruding prop should face up.

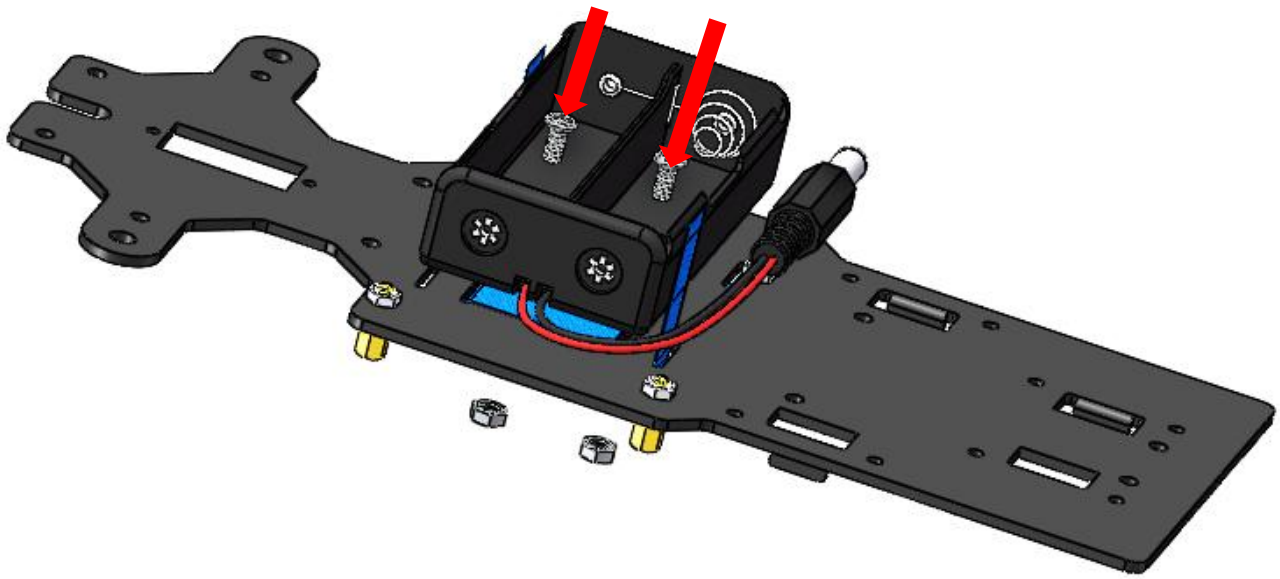


Battery Holder

Turn the Upper Plate upside down. Cut the **ribbon** into two halves. Thread them through the holes on the plate. Pay attention to the direction and leave one end longer out of the plate for each to remove the battery easily later.

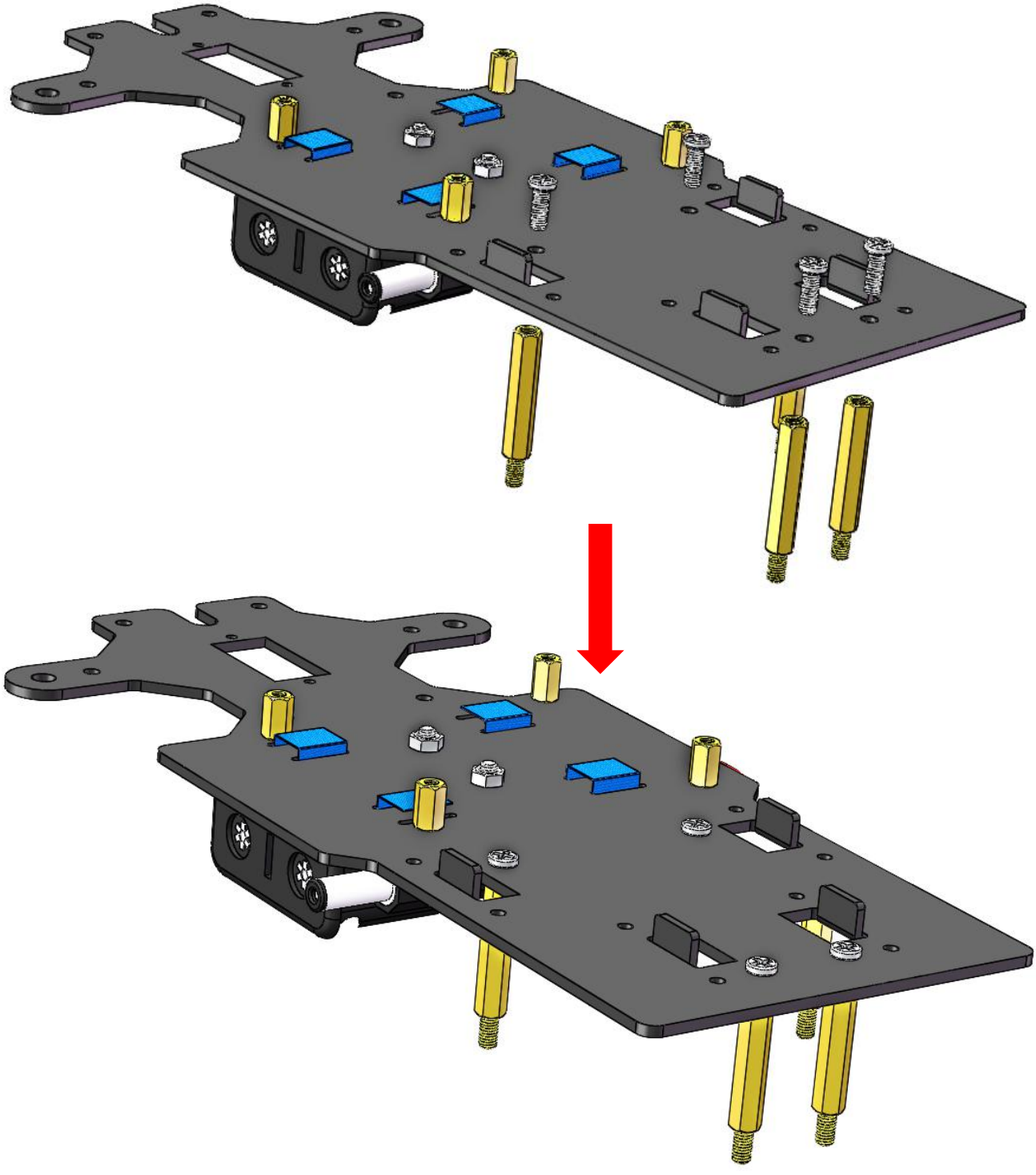


Fasten the battery holder with two **M3x8 countersunk screws** and **M3 nuts**. Pay attention to the direction of battery holder's wire.



Rear Wheels (Screws)

Insert 4 **M3x8 screws** with four **M3x25 copper standoffs**:

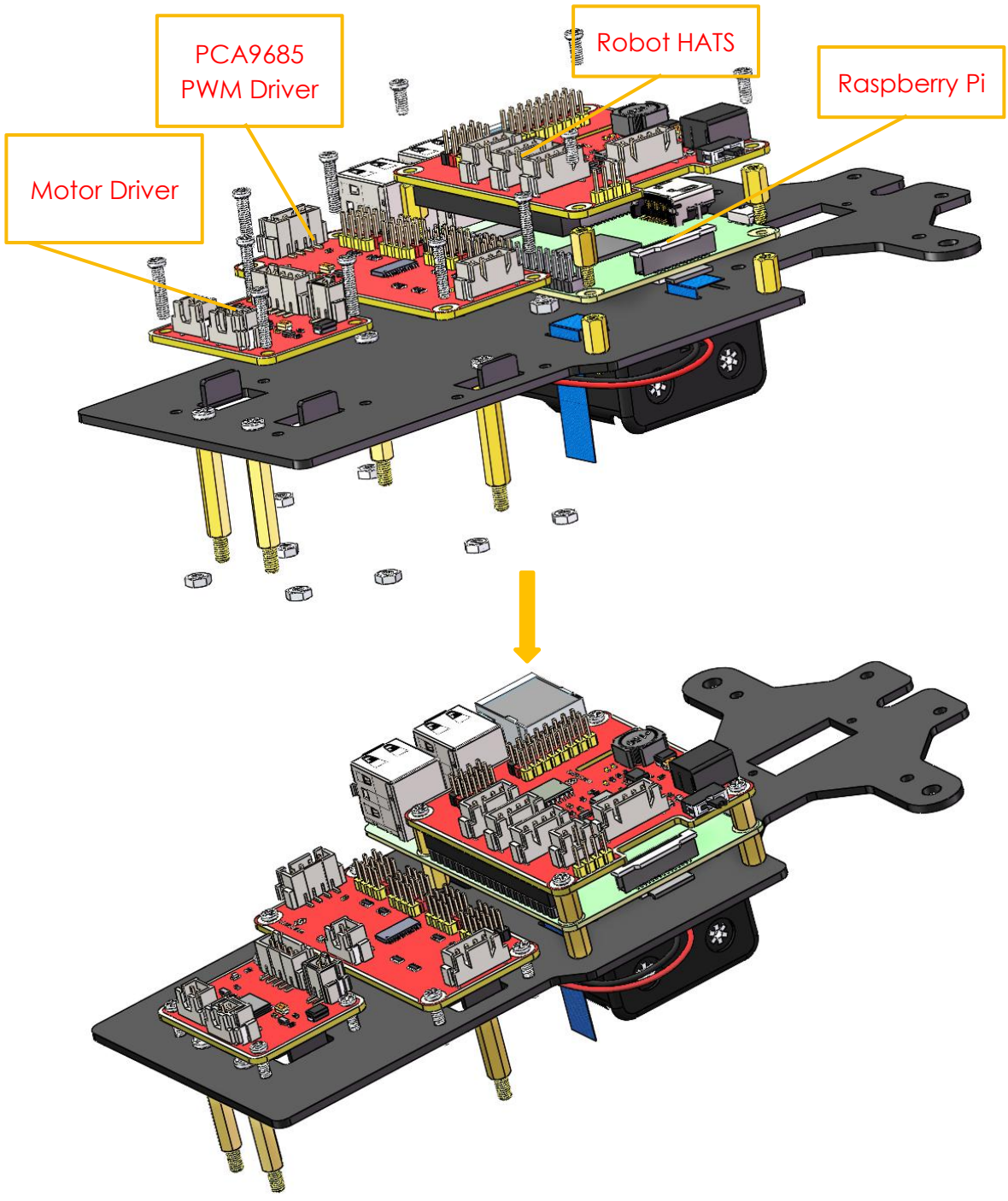


PCB Assembly

1) Assemble the **Raspberry Pi** (TF Card inserted) with 8 **M2.5x8 single pass copper standoffs**, then plug the **Robot HATS** onto it.

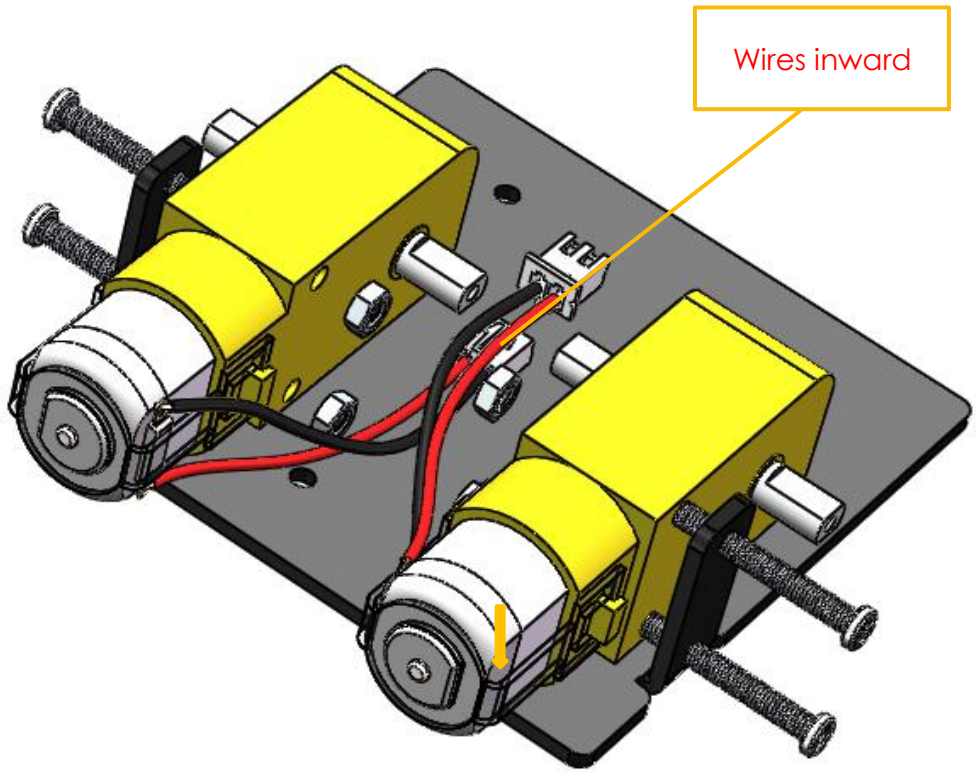
2) Fix the **Robot HATS** with 4 **M2.5x6 screws**.

3) Fix The **PCA9685 PWM Driver** and the **Motor Driver** with 8 **M2.5x12 screws and M2.5 nuts** into the **down plate**:

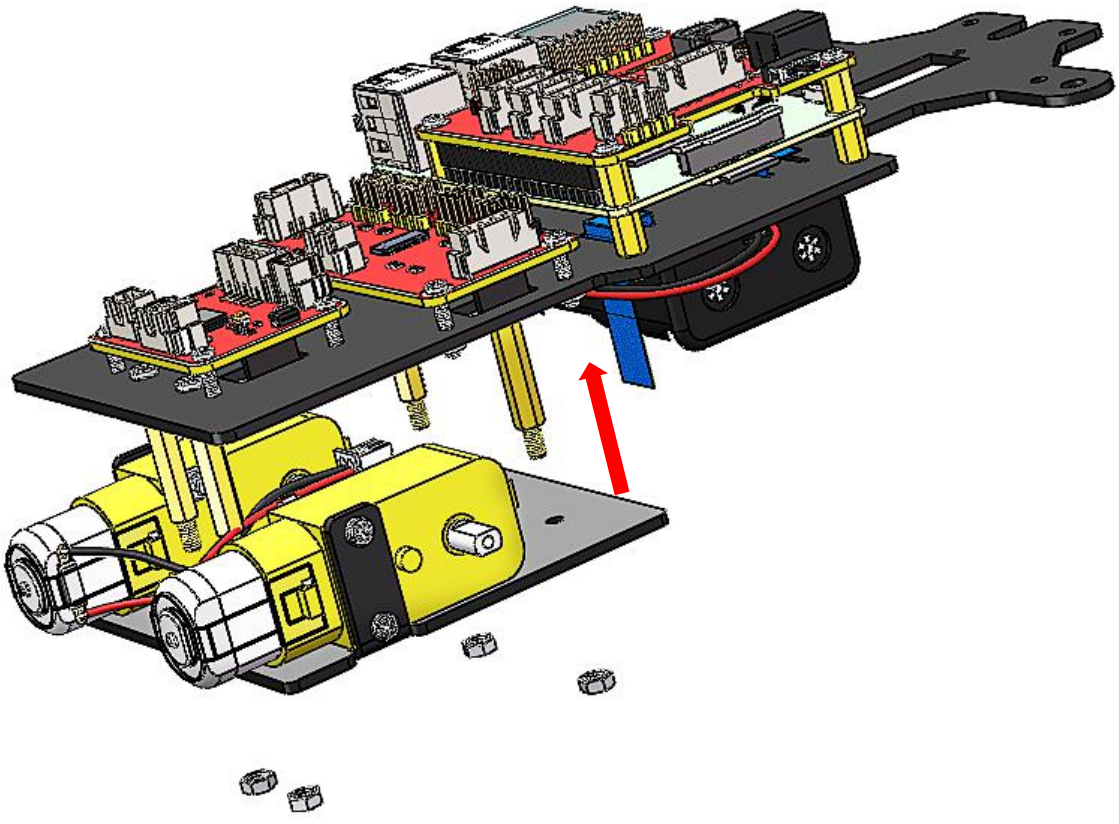


Fixing Rear Wheels

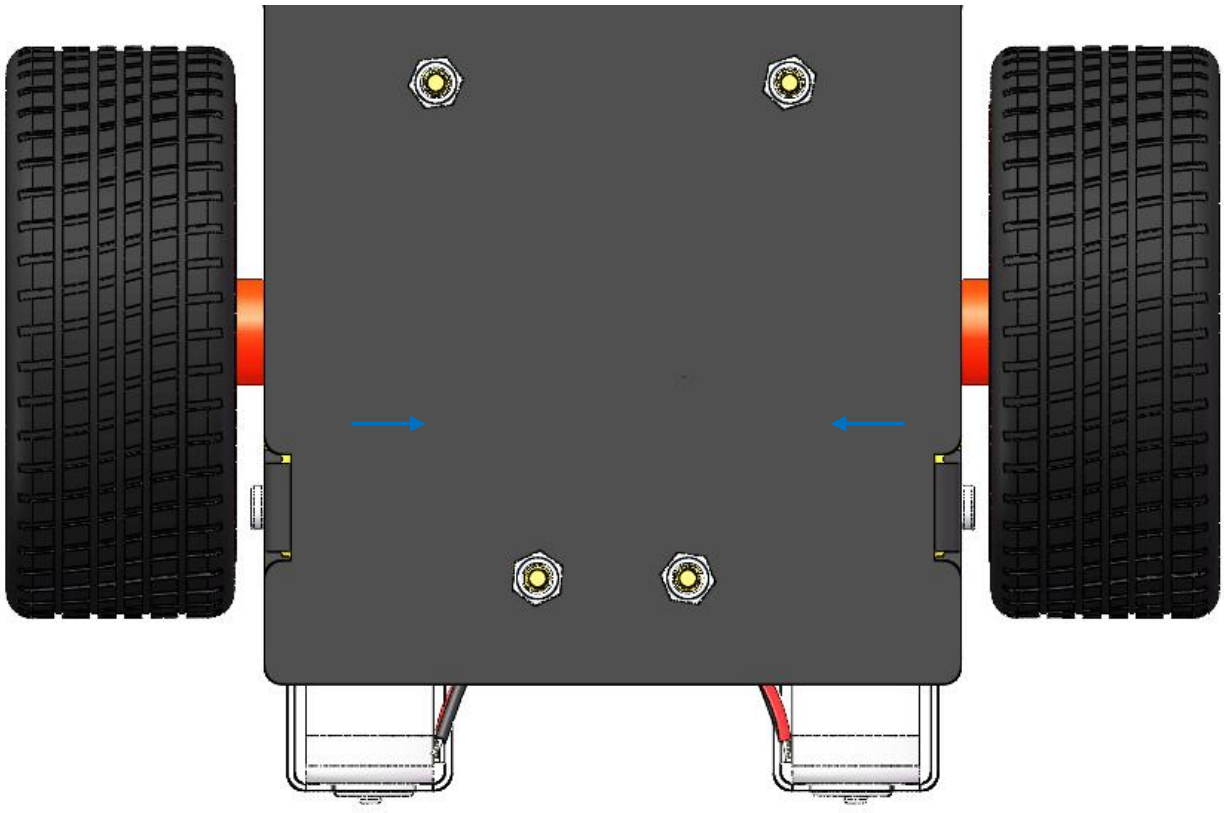
Assemble the two motors with four **M3x25 screws** and **M3 nuts**. Pay attention to place the motors with wires inward, providing convenience for connecting the circuit.



Assemble the rear wheels with 4 M3 nuts.

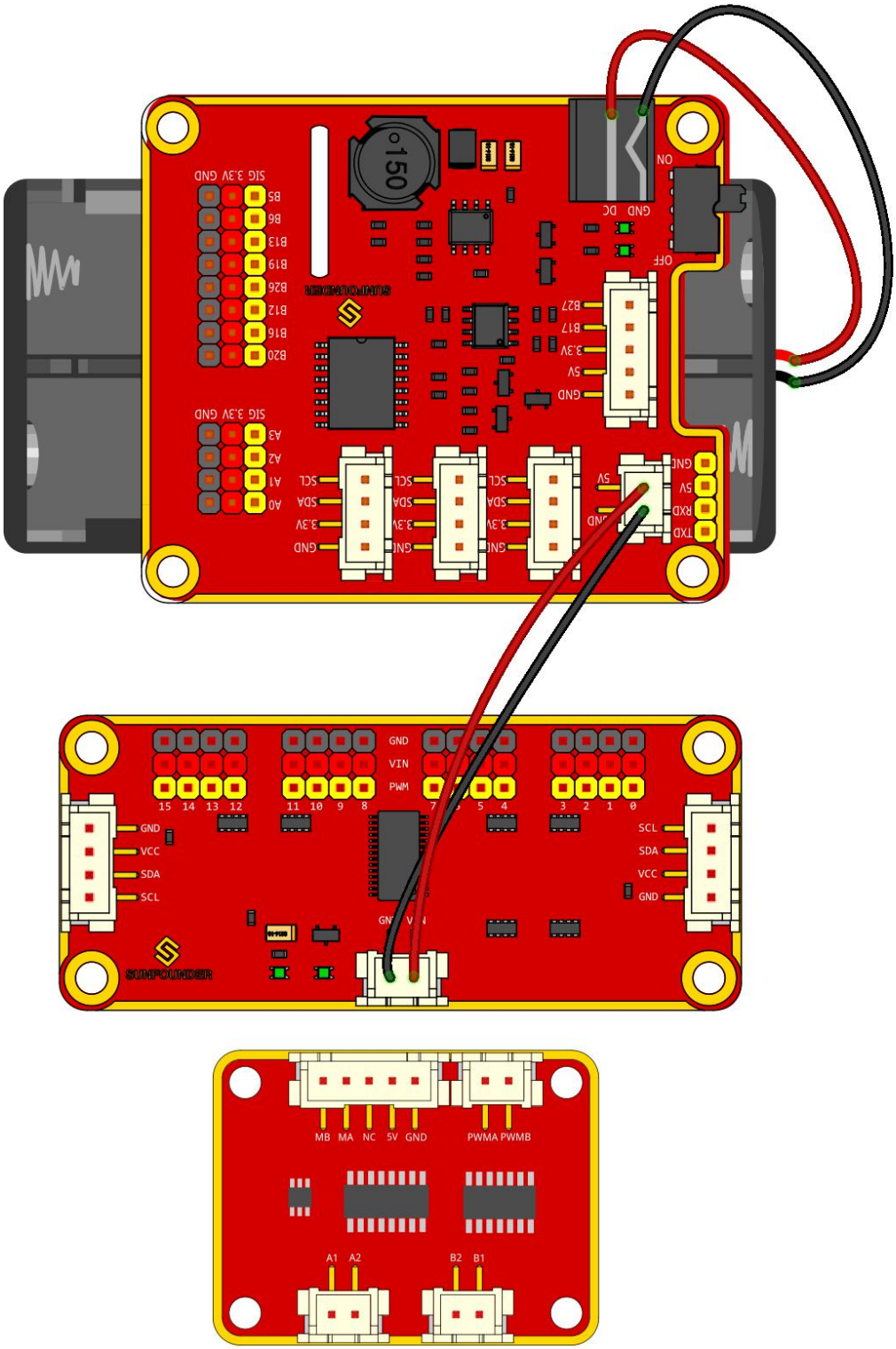


Align the **rear wheels** with the motor shaft, and rotate to insert them gently.

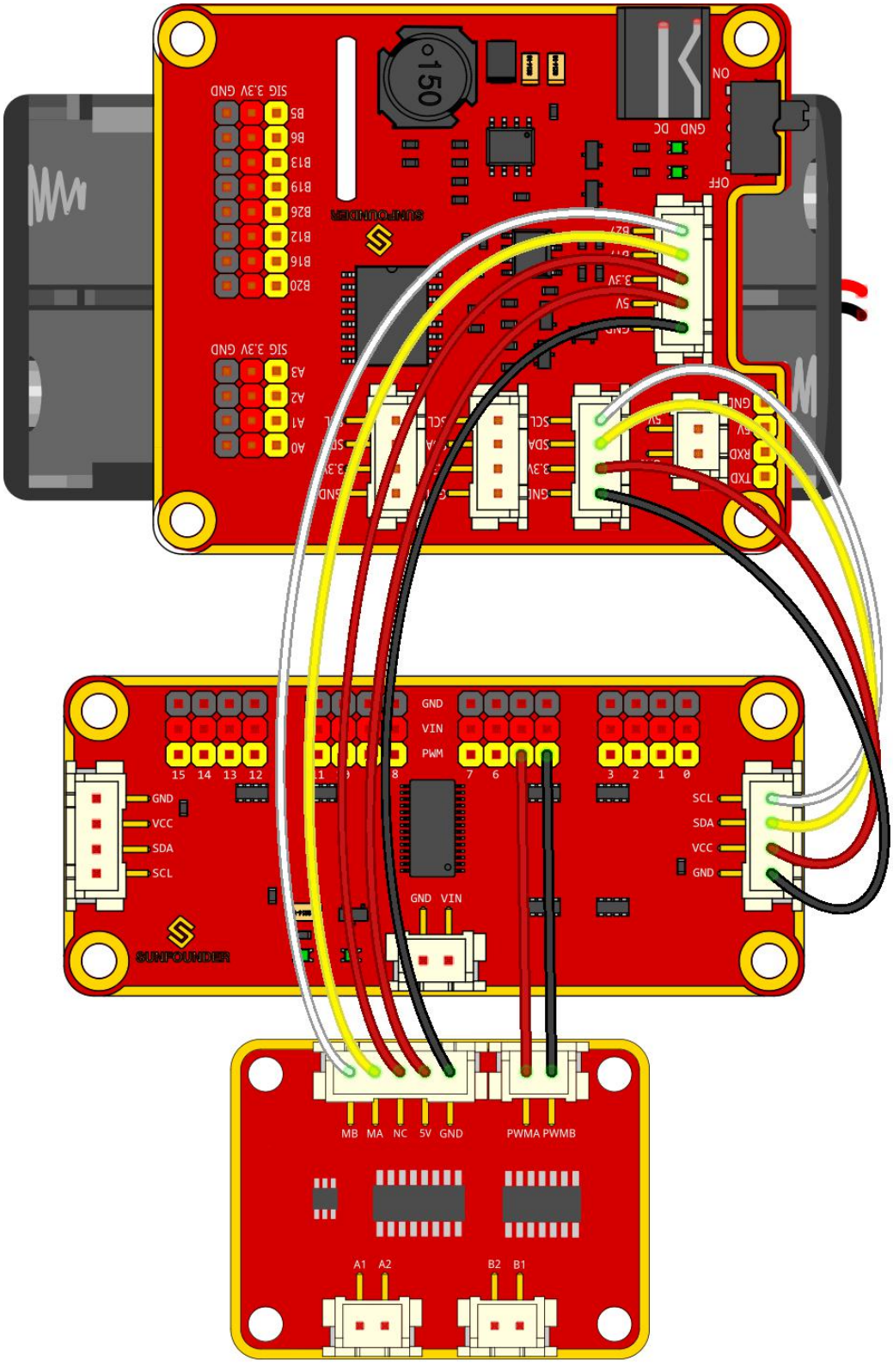


Circuits Building

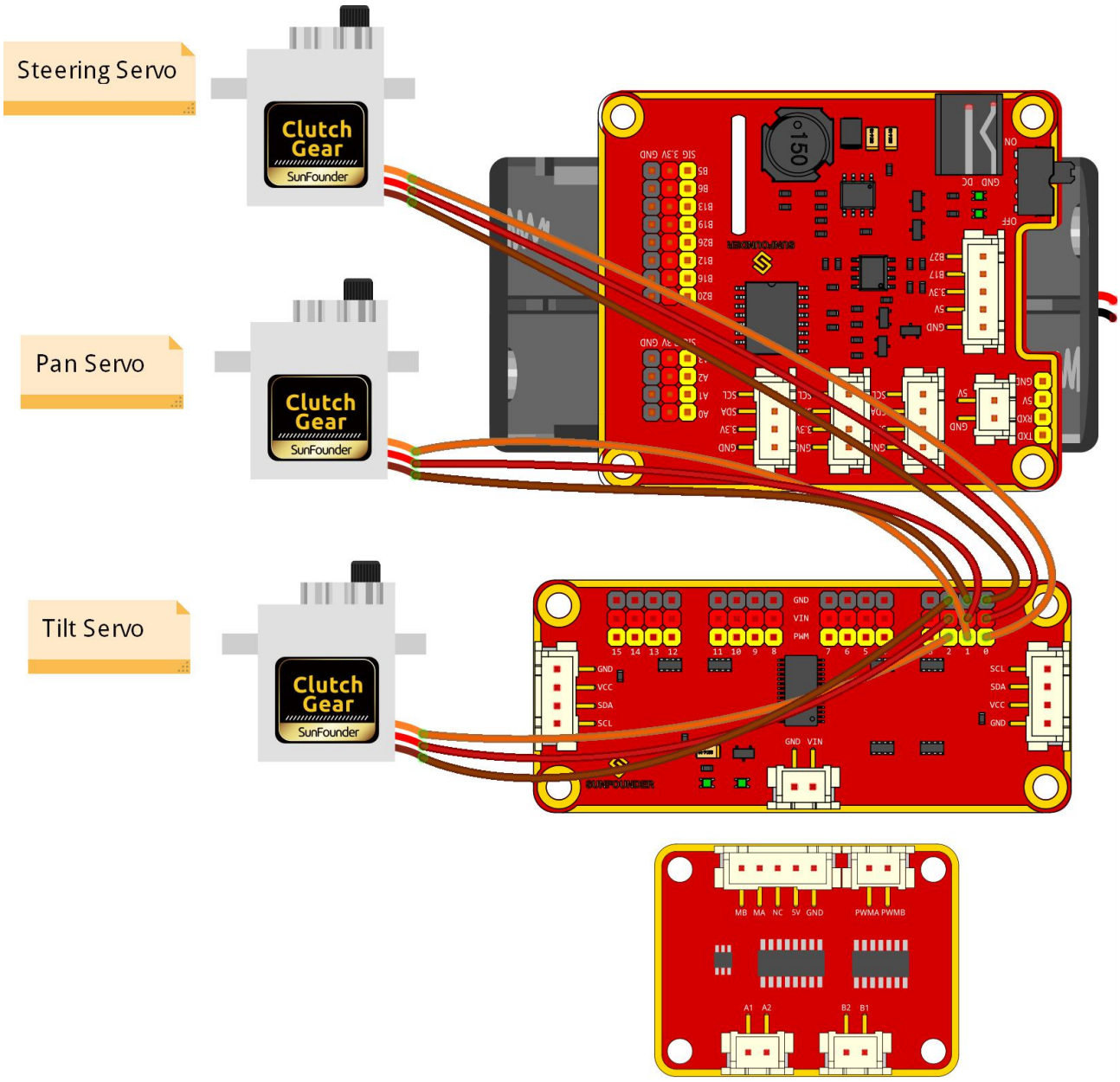
Connect the Power



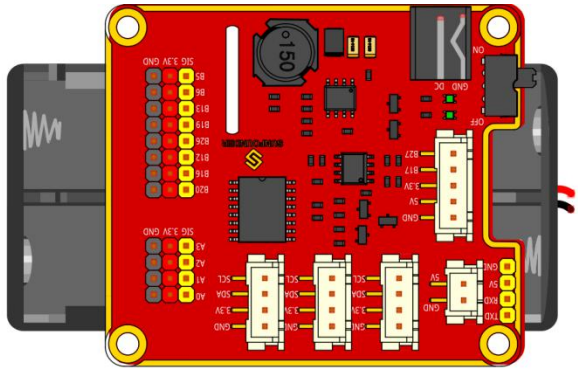
Connect the Modules

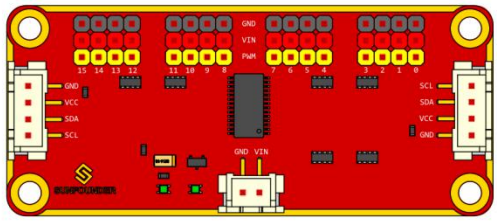


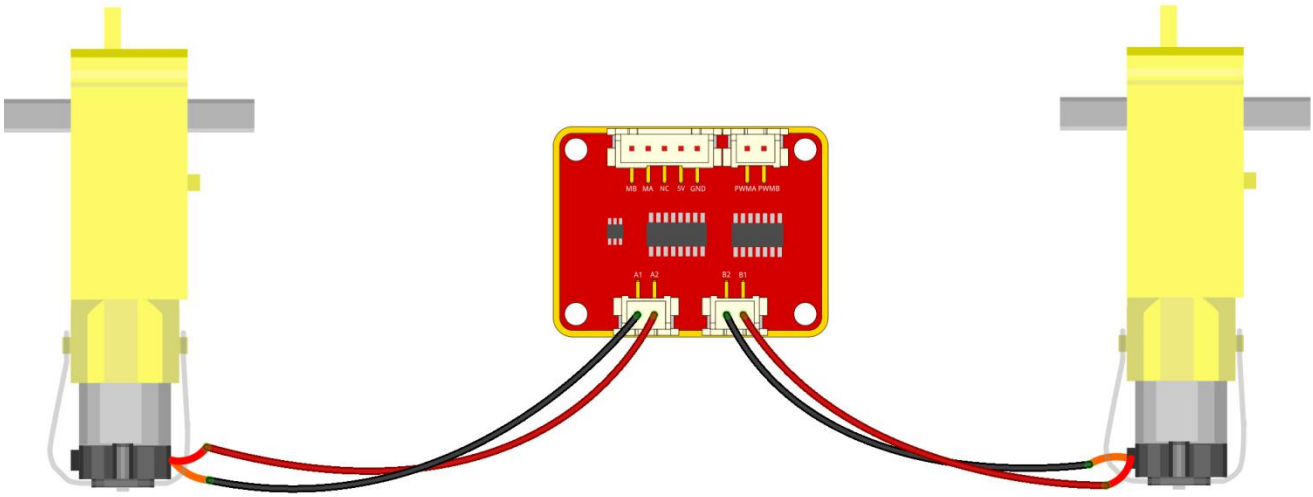
Connect the Servos



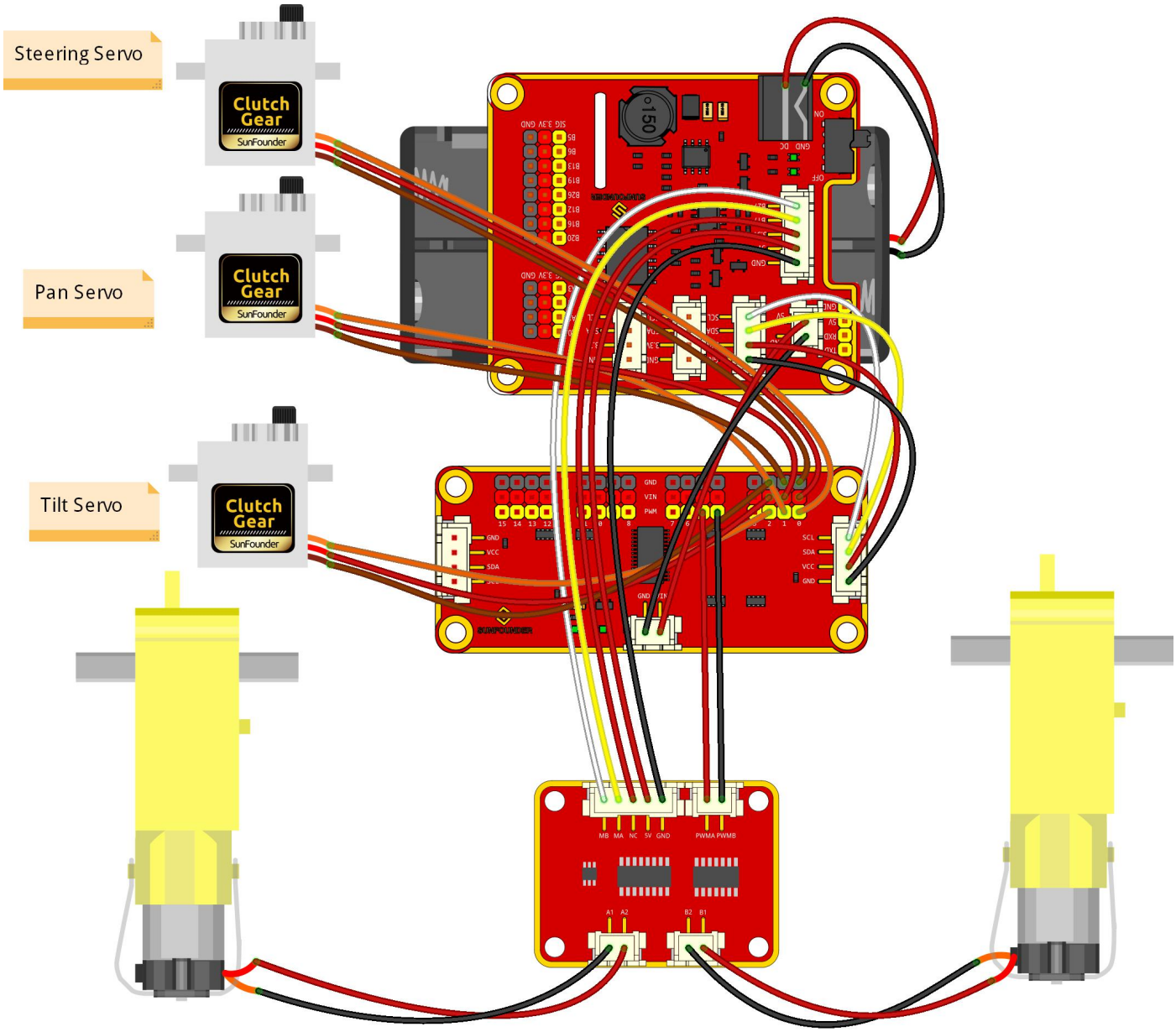
Connect the Motor







The complete connection is shown as follows.



Get Started with Raspberry Pi

In this chapter, we firstly learn to start up Raspberry Pi.

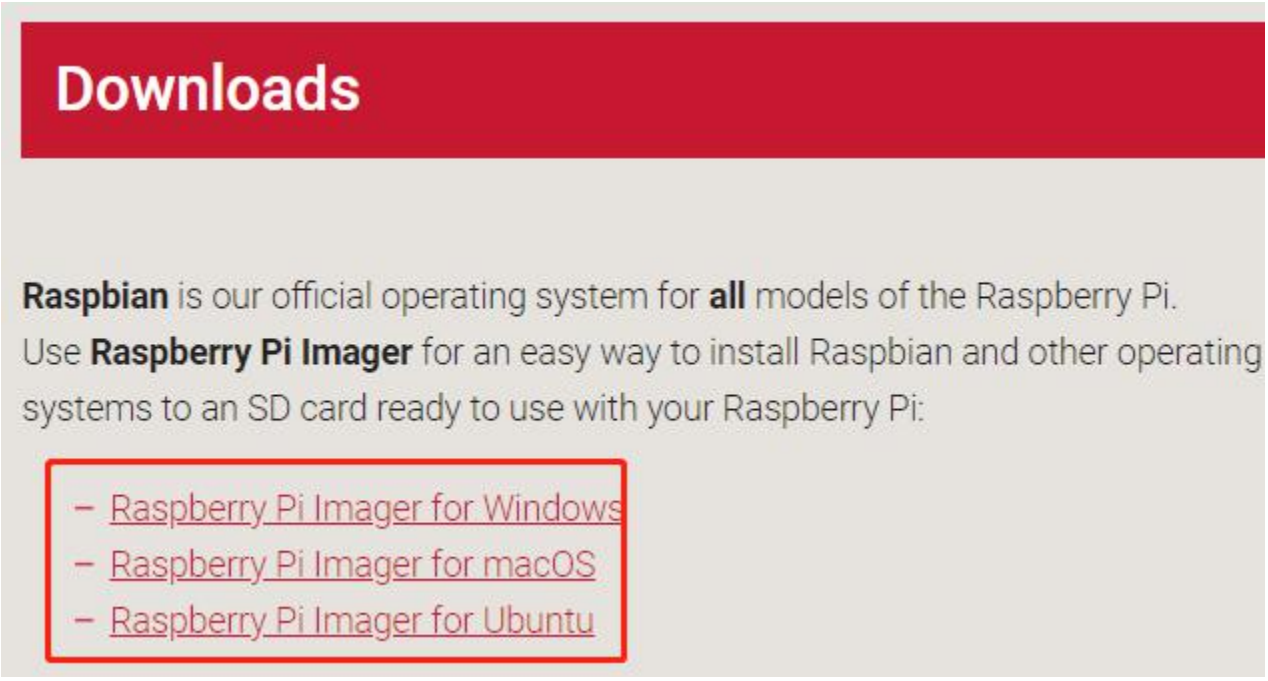
Depending on the different devices you use, you can start up the Raspberry Pi in different methods. We have two situations: having a screen or no screen, and you can refer to relevant tutorials respectively. If your Raspberry Pi is set up, you can skip the part: Get Started with Raspberry Pi and go into the next chapter Servo Configuration.

❖ News

Recently, RPi lauched a new method-Raspberry Pi Imager to install Raspbian and other operating systems to an SD card ready to be used with your Raspberry Pi.

Compared to the previous methods, it ‘s more time-saving for that it processes the flashing and download of the image at the same time. You can have a try.

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



There are Raspberry Pi Imagers of Windows, Mac OS and Ubuntu. Check the link to download, install and open Raspberry Pi Imager. Select Operating System and SD card then click **WRITE**. After flashing, you can start your Raspberry Pi and plug in a screen.

Without a screen, you can refer to Page 35 to configure Wi-Fi and start SSH after flashing.

If You Have A Screen

If you have a screen, you can use the NOOBS (New Out Of Box System) to install the Raspbian system.

- Required Components

Any Raspberry Pi	1 * Power Adapter
1 * Monitor	1 * Monitor Power Adapter
1 * HDMI cable	1 * Micro SD card
1 * Mouse	1 * Keyboard
1 * Personal Computer	

- Procedures

Step 1

To download NOOBS from your PC, you can choose **NOOBS** or **NOOBS LITE** - the only difference is that there is a built-in offline Raspbian installer in **NOOBS**, while

the **NOOBS LITE** can only be operated online. Here, you are suggested to use the former. Here is the download address of NOOBS:

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

Step 2

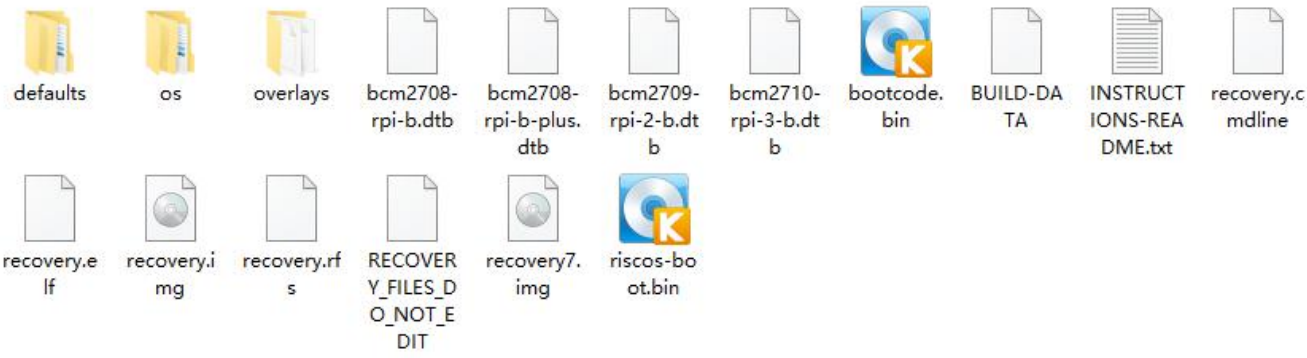
Plug in the Micro SD reader and format the Micro SD card with the **SD Formatter** (https://www.sdcard.org/downloads/formatter/index.html). If there are some important files in the Micro SD card, please backup them first.

Step 3

Next, you will need to extract the files from the NOOBS zip archive you downloaded from the Raspberry Pi website.

- Find the downloaded archive — by default, it should be in your Downloads folder.
- Double-click on it to extract the files, and keep the resulting Explorer/Finder window open.

Finally Select all the files in the NOOBS folder and copy them to the Micro SD card.



Step 4

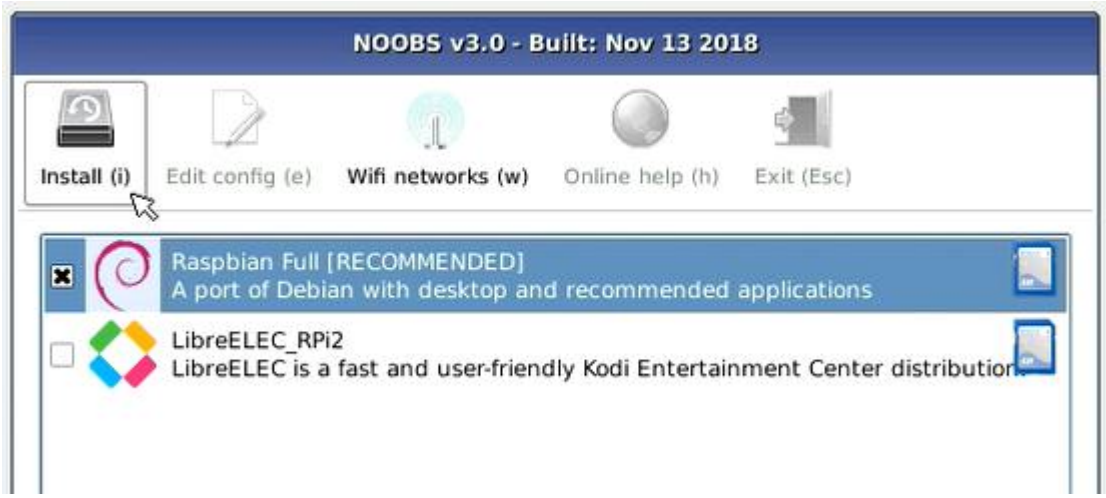
All the files transferred, the Micro SD card pops up.

Step 5

Insert the Micro SD card into the Raspberry Pi. In addition, connect the screen, keyboard and mouse to it. Finally you are also recommended to use the power adapter of Raspberry Pi to power your car for that the first test will take a long time.

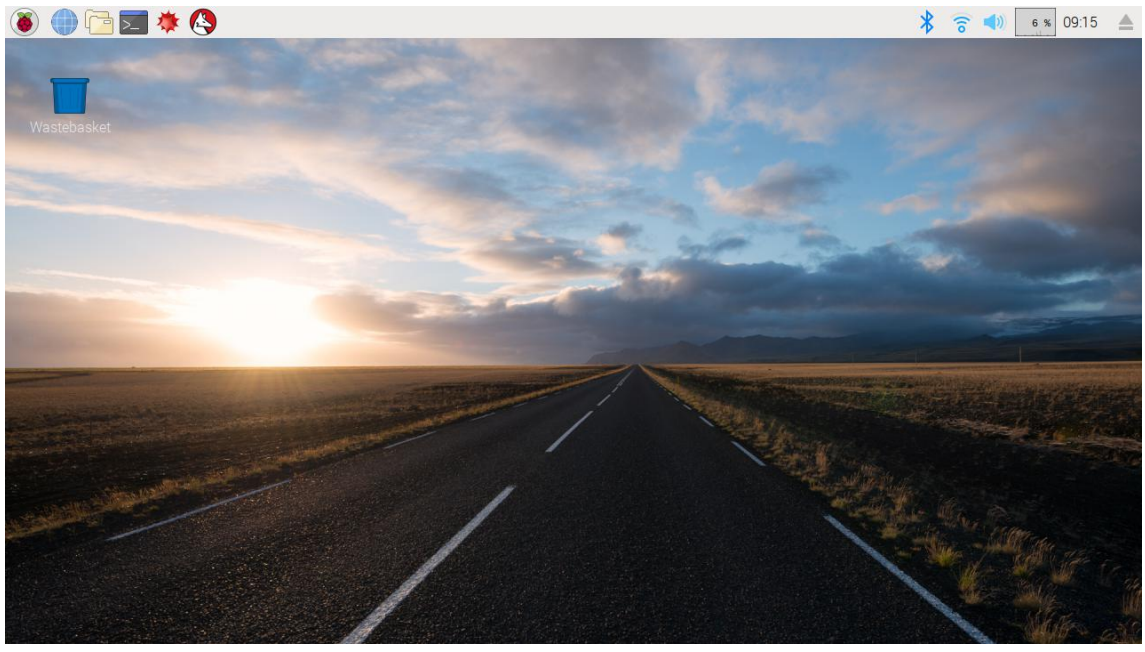
Step 6

It will go to the NOOBS interface after starting up. If you use NOOBS LITE, you need to select Wi-Fi networks (w) first. Tick the checkbox of the Raspbian and click Install in the top left corner. The NOOBS will help to conduct the installation automatically. This process will take a few minutes.



Step 7

When the installation is done, the system will restart automatically and the desktop of the system will appear.



Step 8

If you run Raspberry Pi for the first time, the application of "Welcome to Raspberry Pi" pops up and guides you to perform the initial setup.

Step 9

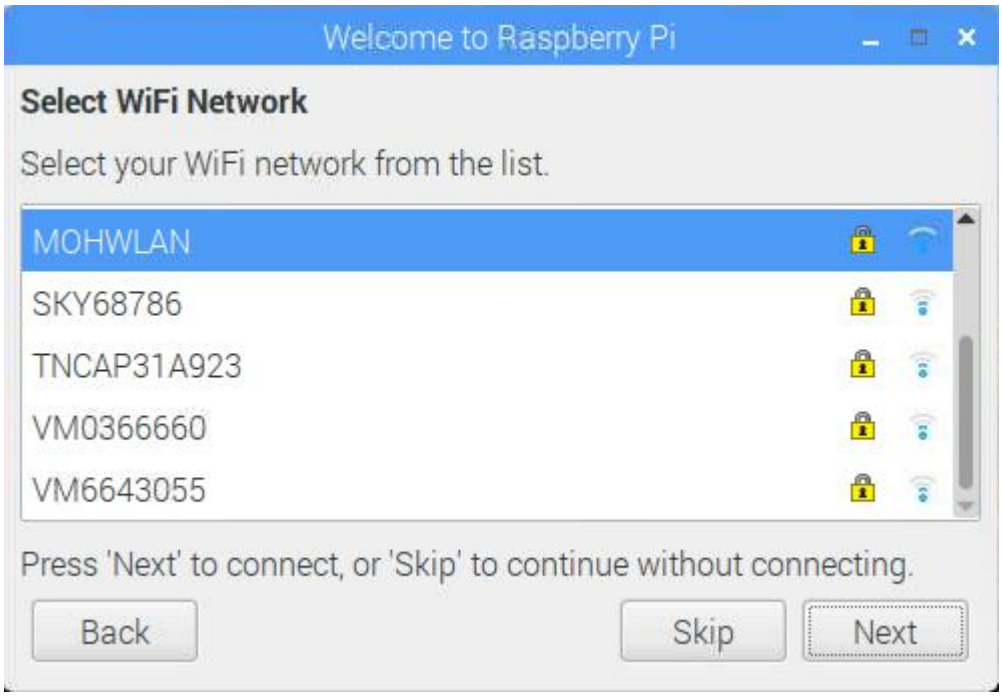
Set country/region, language and time zone, and then click "next" again.

Step 10

Input the new password of Raspberry Pi and click “Next”.

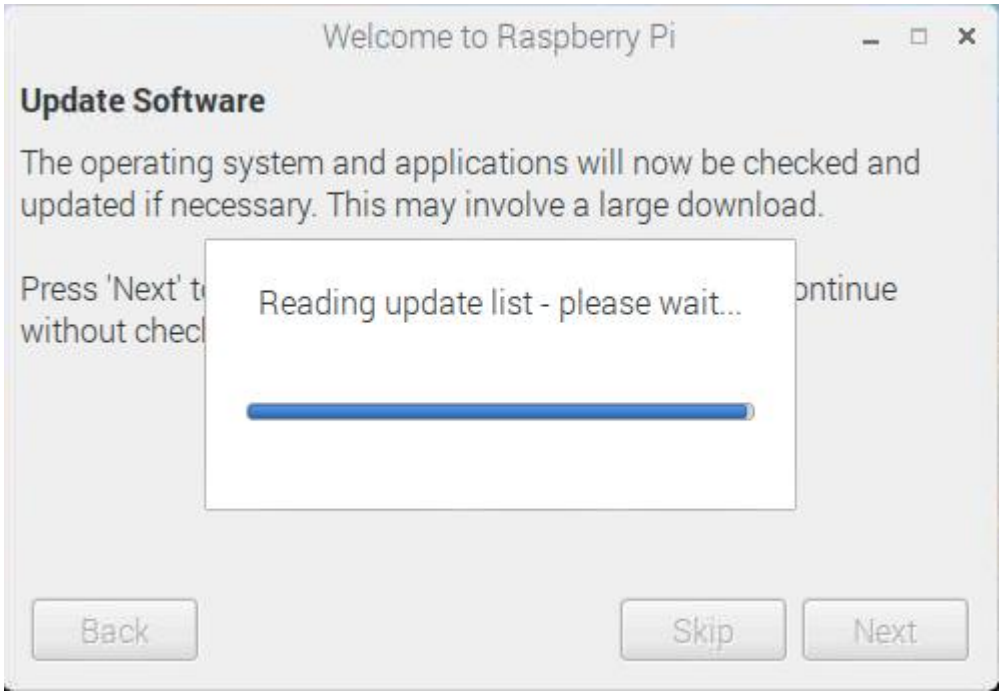
Step 11

Connect the Raspberry Pi to WIFI and click "Next".



Step 12

Retrieve update.



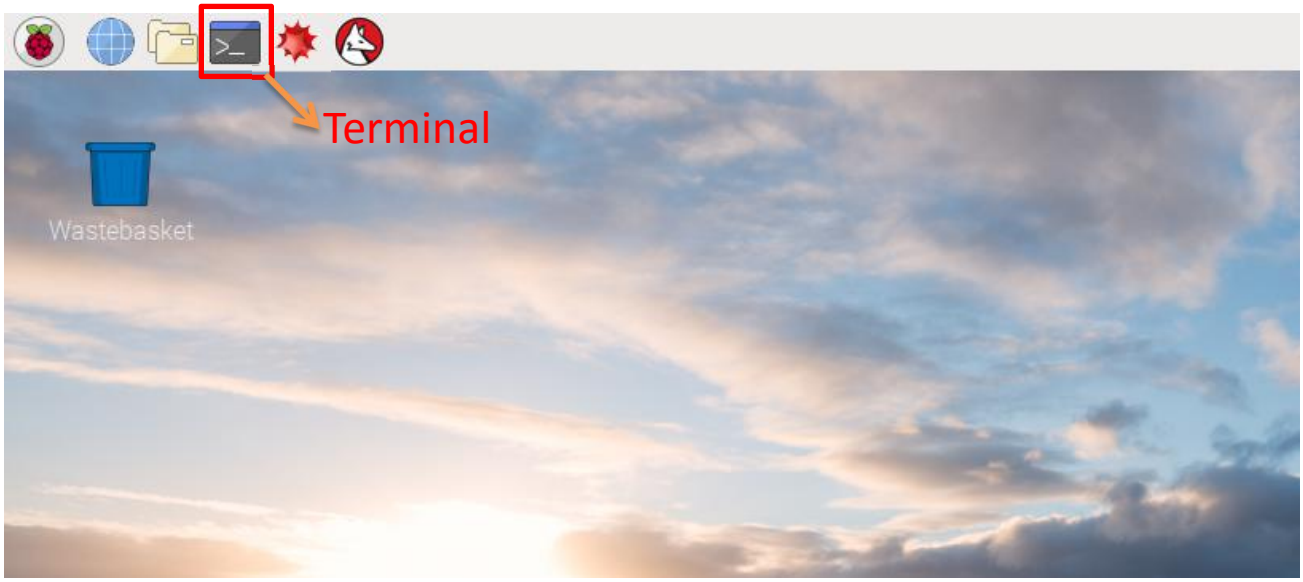
Step 13

Click "Done" to complete the Settings.

Now we can run the Raspberry Pi.

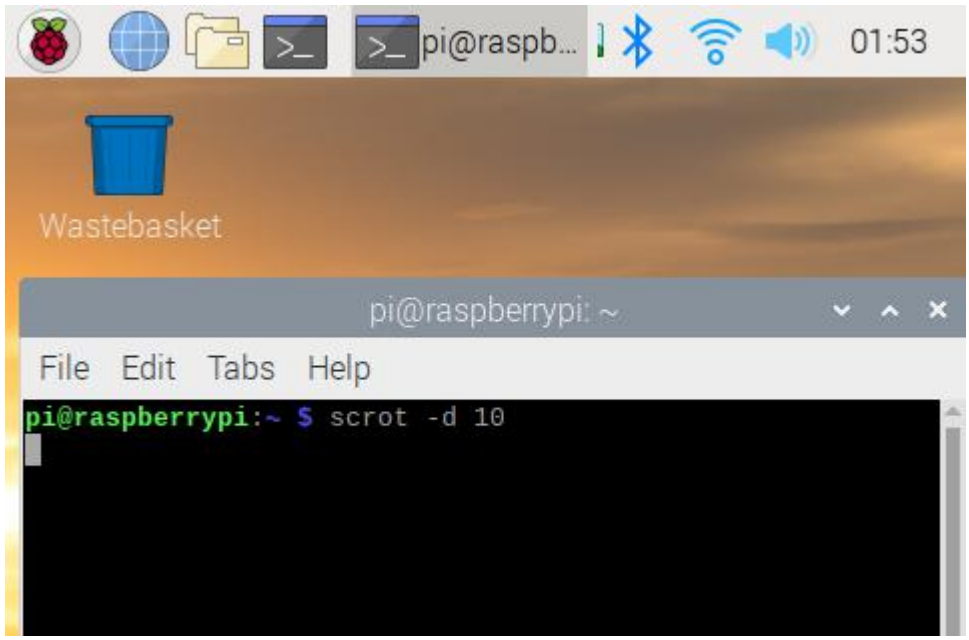
Step 14

Click the Terminal icon on the top left corner.



Step 15

Then you can input the commands on the Terminal.



Note: You can check the complete tutorial of NOOBS on the official website of the Raspberry Pi: https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

If You Have No Screen

If we don't have a screen, we can directly write the Raspbian system to the Micro SD card and we can control the Raspberry Pi on PC remotely by directly modifying the configuration file of the network settings in the Micro SD card.

- Burn System

Step 1

Prepare the tool of image burning. Here we use the **balenaEtcher**. You can download the software from the link: *https://www.balena.io/etcher/*

Step 2

Download the complete image on the official website by clicking this link: *https://www.raspberrypi.org/downloads/raspbian/*. There are three different kinds of Raspbian system available, You are recommend to install the version: **Raspbian Buster with desktop and recommended software**.

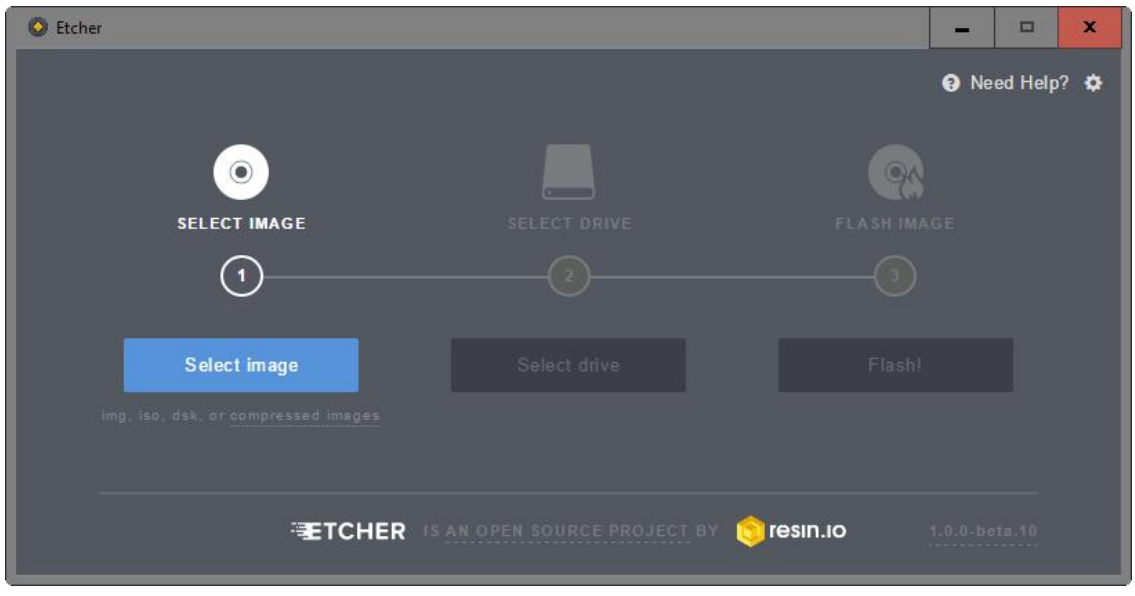
Step 3

Unzip the package downloaded and you will see the *.img* file inside.

> **Note**: DO NOT extract the file.

Step 4

Plug the USB Card Reader into the computer, then you can burn the image file with the Etcher.



Step 5

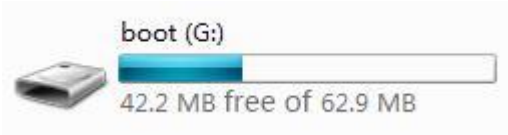
At this point, Raspbian is installed. **Keep the USB card reader plug in your computer**. If you want to apply it, next you need to complete the settings accordingly.

- Connect the Raspberry Pi to the Internet

There are two methods to help get the Raspberry Pi connected to the network: the first one is using a network cable, the other way is using WIFI. We will talk in detail about how to connect via WIFI as below.

If you want to use the WIFI function, you need to modify a WIFI configuration file wpa_supplicant.conf in the Micro SD card by your PC that is located in the directory **/etc/wpa_supplicant/**.

If your personal computer is working on a linux system, you can access the directory directly to modify the configuration file; however, if your PC use Windows system, then you can't access the directory and what you need next is to go to the directory, */boot/* to create a new file with the same name, **wpa_supplicant.conf**.



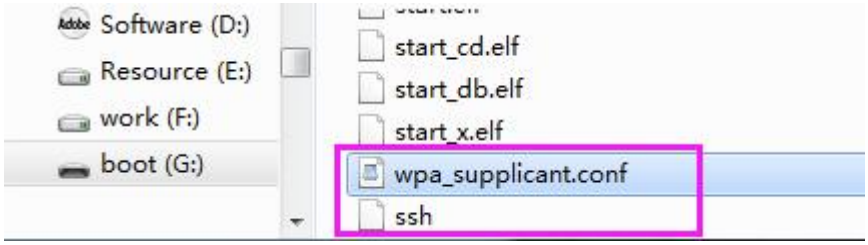
Input the following content in the file.

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB
network={
ssid="WiFi-A"
psk="Sunfounder"
key_mgmt=WPA-PSK
priority=1
}
```

You need to replace "**WiFi-A**" with your custom name of WiFi and "**Sunfounder**" with your password. By doing these, the Raspbian system will move this file to the target directory automatically to overwrite the original WIFI configuration file when it runs next time. After doing this step, you also need to **keep the USB card reader plug in your computer.**

- Start SSH

To use the function of remote control of the Raspberry Pi, you need to start SSH firstly that is a more reliable protocol providing security for remote login sessions and other network services. Generally, SSH of Raspberry Pi is in a disabled state. Additionally, if you want to run it, you need to create a file named SSH under directory /boot/.

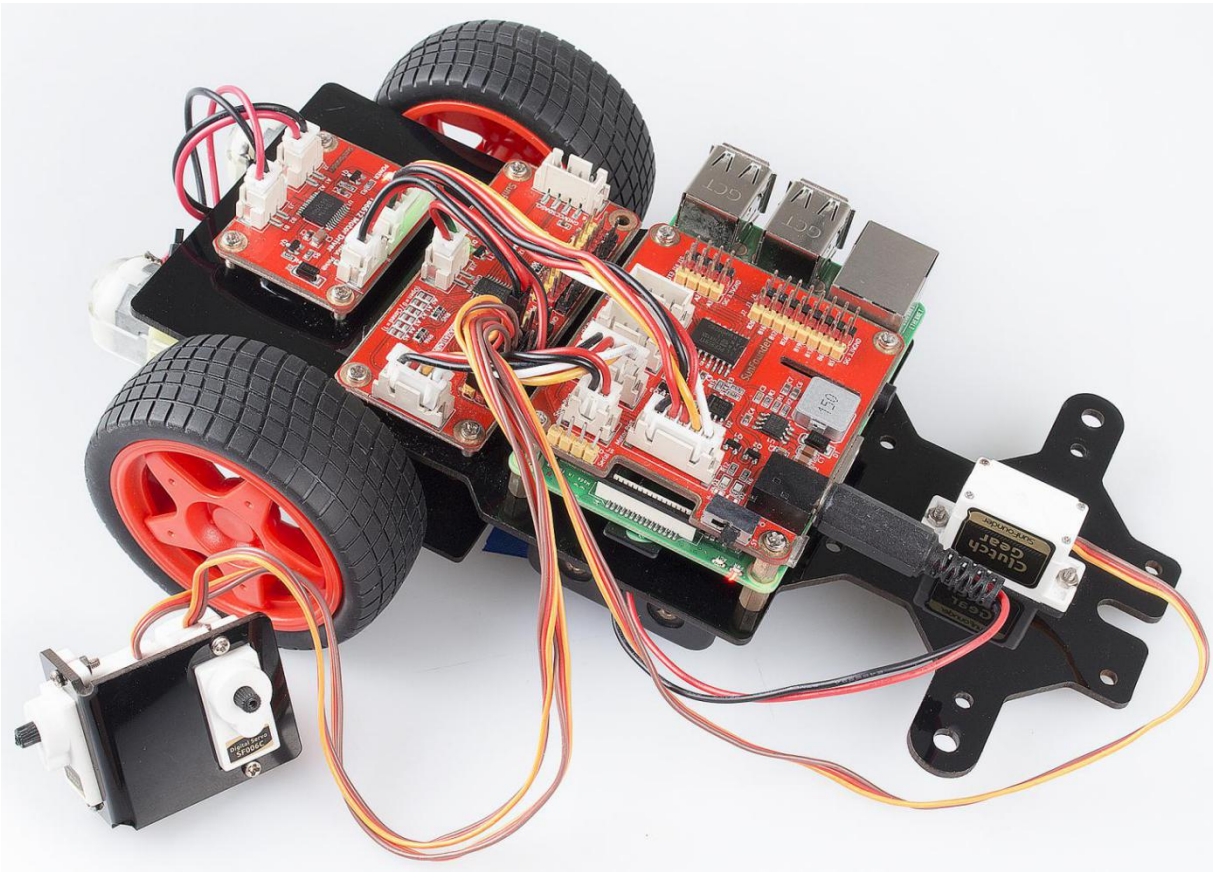


Now, the Raspbian system is configured.

- Power on the Raspberry Pi

You can plug out the USB card reader and then plug the Micro SD card into the Raspberry Pi.

Put two 18650 fully charged batteries in the holder, plug the wires from the battery holder into the development board then toggle the switch from off to on.



- Get the IP Address

After the Raspberry Pi is powered on with a power adapter, we need to get the IP address of it. There are many ways to know the IP address, and two of them are listed as follows.

1. Checking via the router

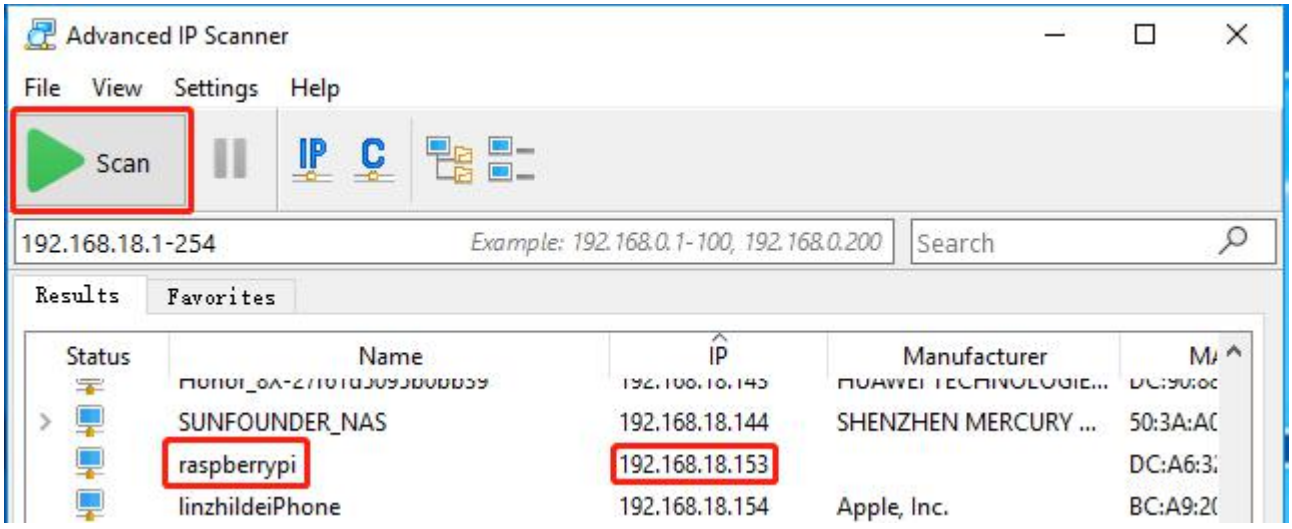
If you have permission to log in the router(such as a home network), you can check the addresses assigned to Raspberry Pi on the admin interface of router.

The default hostname of the system, Raspbian is **raspberrypi**, and you need to find it. (If you are using ArchLinuxARM system, please find alarmpi.)

2. Network Segment Scanning

You can also use network scanning to look up the IP address of Raspberry Pi. You can apply the software, **Advanced IP scanner(download from Google)**.

Click **Scan** and the name of all connected devices will be displayed. Similarly, the default hostname of the Raspbian system is **raspberrypi**, now you need to find the hostname and its IP.



- Use the SSH Remote Control

We can open the Bash Shell of Raspberry Pi by applying SSH. Bash is the standard default shell of Linux. The Shell itself is a program written in C that is the bridge linking the customers and Unix/Linux. Moreover, it can help to complete most of the work needed.

➢ For Linux or/Mac OS X Users

Step 1

Go to **Applications**->**Utilities**, find the **Terminal**, and open it.

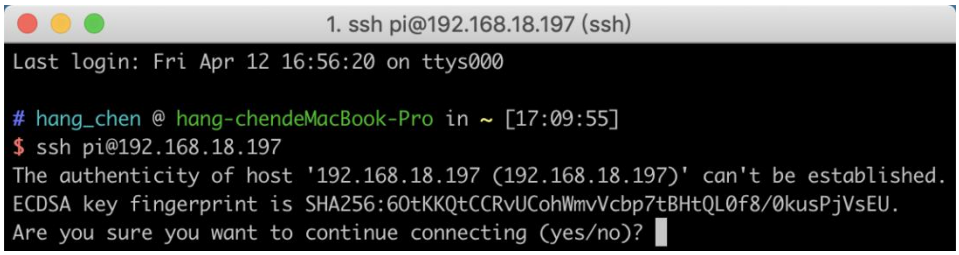
Step 2

Type in **ssh pi@ip_address** . “pi”is your username and “ip_address” is your IP address. For example:

```
ssh pi@192.168.18.197
```

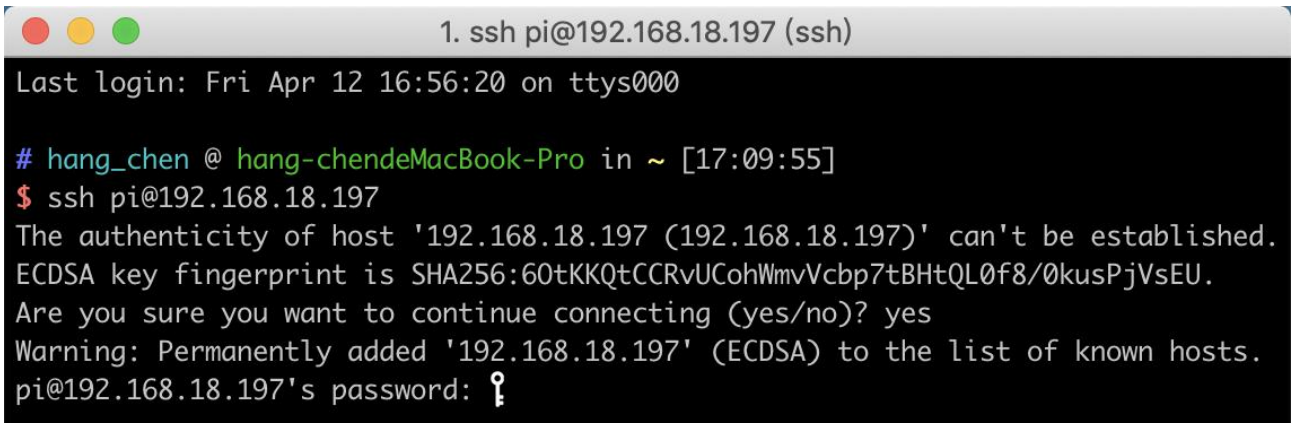
Step 3

Input“yes”.



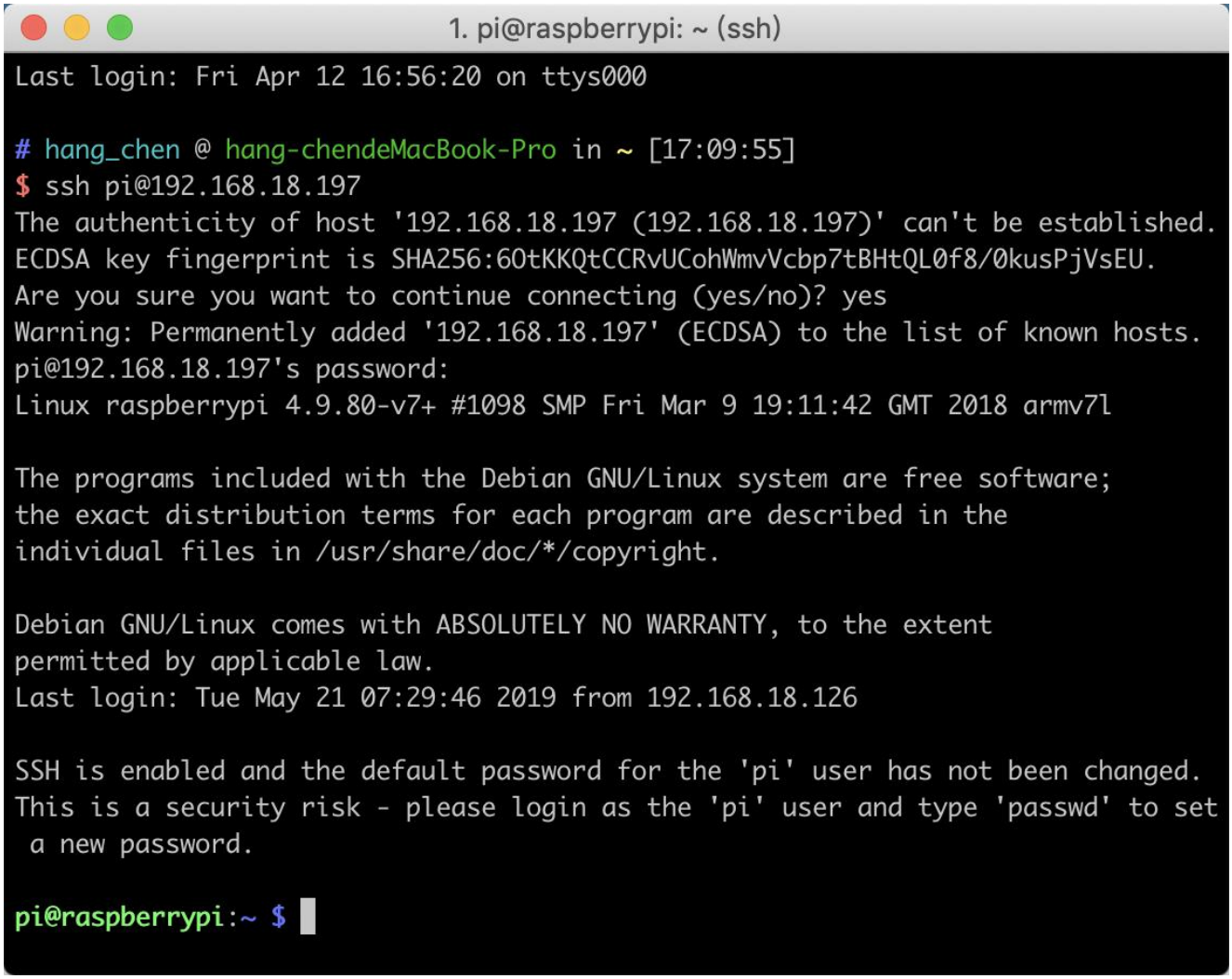
Step 4

Input the passcode and the default password is **raspberry**.



Step 5

We now get the Raspberry Pi connected and are ready to go to the next step.



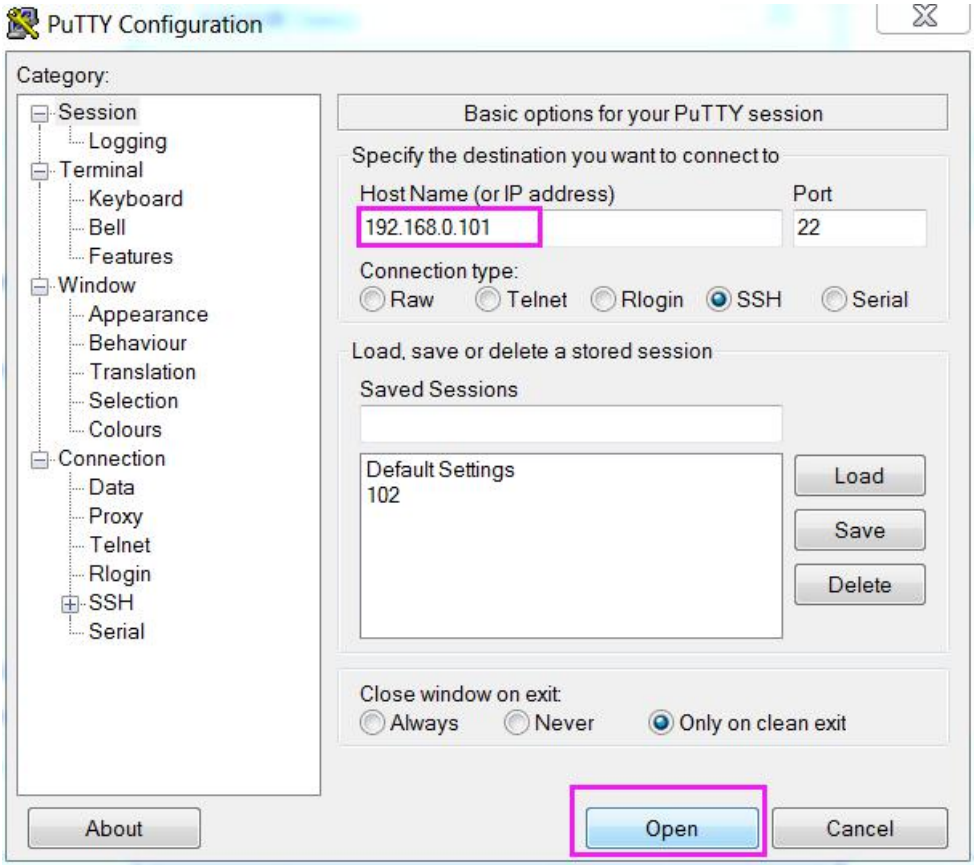
Note: When you input the password, the characters do not display on window accordingly, which is normal. What you need is to input the correct passcode.

➢ For Windows Users

If you're a Windows user, you can use SSH with the application of some software. Here, we recommend **PuTTY**(You can download from Google).

Step 1

Download PuTTY. Open PuTTY and click **Session** on the left tree-alike structure. Enter the IP address of the RPi in the text box under **Host Name (or IP address)** and **22** under **Port** (by default it is 22).



Step 2

Click **Open**. Note that when you first log in to the Raspberry Pi with the IP address, there prompts a security reminder. Just click **Yes**.

Step 3

When the PuTTY window prompts "**login as:**", type in "**pi**"(the user name of the RPi), and **password:** "raspberry" (the default one, if you haven't changed it).

Note: When you input the password, the characters do not display on window accordingly, which is normal. What you need is to input the correct password.

Here, we get the Raspberry Pi connected and it is time to conduct the next steps.

Servo Configuration

And since the servos used in this kit are adjusted by software and there's no such physical sticking point as other servos, here we need to configure the servo via software. First you need to finish some software installation before the configuration.

Note: Please do forget to put in the battery and slide the power switch to ON in this chapters.

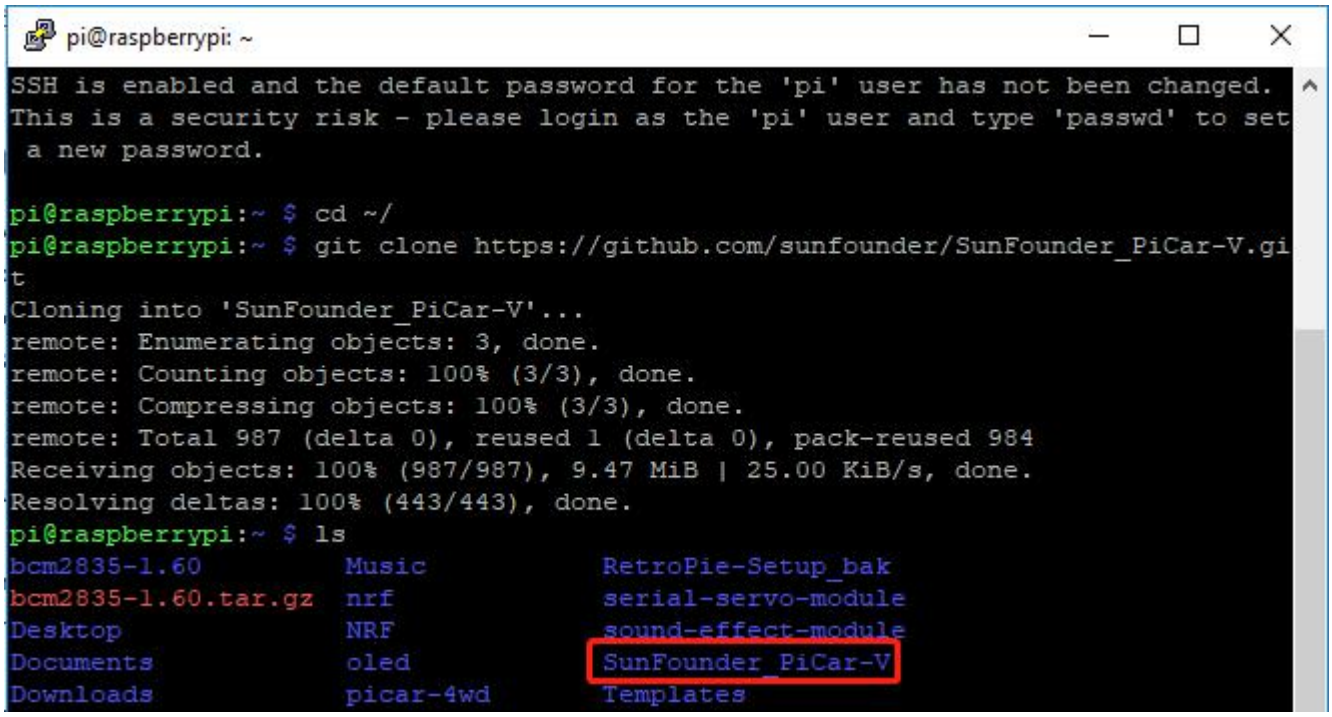
Get Source Code

You can find the source code in our Github repositories. Download the source code by *git clone*:

```
cd ~/
git clone https://github.com/sunfounder/SunFounder_PiCar-V.git
```

Note: Please pay attention to your typing – if you get the prompt of entering your user name and password, you may have typed wrong. If unluckily you did so, press Ctrl + C to exit and try again.

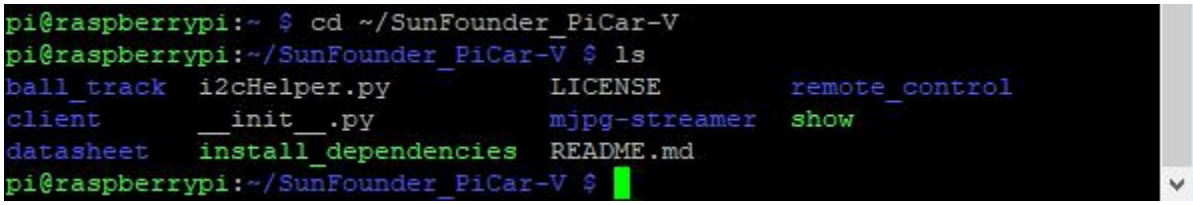
Check by the **ls** command, then you can see the code directory *SunFounder_PiCar-V*:



Go to the Code Directory

```
cd ~/SunFounder_PiCar-V
```

Enter the code directory and you can see the installation script:



Install the Environment via the Script

You can get all the required software and configuration done with the install_dependencies script. If you want to do step by step instead, please follow the instructions in the **Appendix 1: Function of the Server Installation Scripts.**

```
sudo ./install_dependencies
```

Notes:

1. The installation script will install the required components and configure for the running environment. Make sure your Raspberry is connected to the Internet during the installation, or it would fail.
2. The Raspberry Pi will prompt you to reboot after the installation. You're recommended to type in **yes** to reboot.

Configure the Servo to 90 degrees

After reboot, run the picar tool:

```
cd ~/SunFounder_PiCar-V
picar servo-install
```

```
pi@raspberrypi:~/SunFounder_PiCar-V $ picar servo-install
```

Notes:

If the "OSError: [Errno 121] Remote I/O error" error message appears, open raspi-config:

```
sudo raspi-config
```

Then choose 5 Interfacing Options → P5 I2C → <YES> to enable I2C service.

After the code is running, insert the rocker arm into the servo. You will see the rocker arm is rotate in clockwise and counterclockwise, then stop at a specific location. It means the servo is good. If the any of the conditions below happened to your servo, your servo is bad:

1) Noisy, hot.
2) If unplug the servo line and rotate the rocker arm, it sounds like "ka" "ka" "ka" or there has no sounds of gear driving.
3) Rotate slowly but continuously.

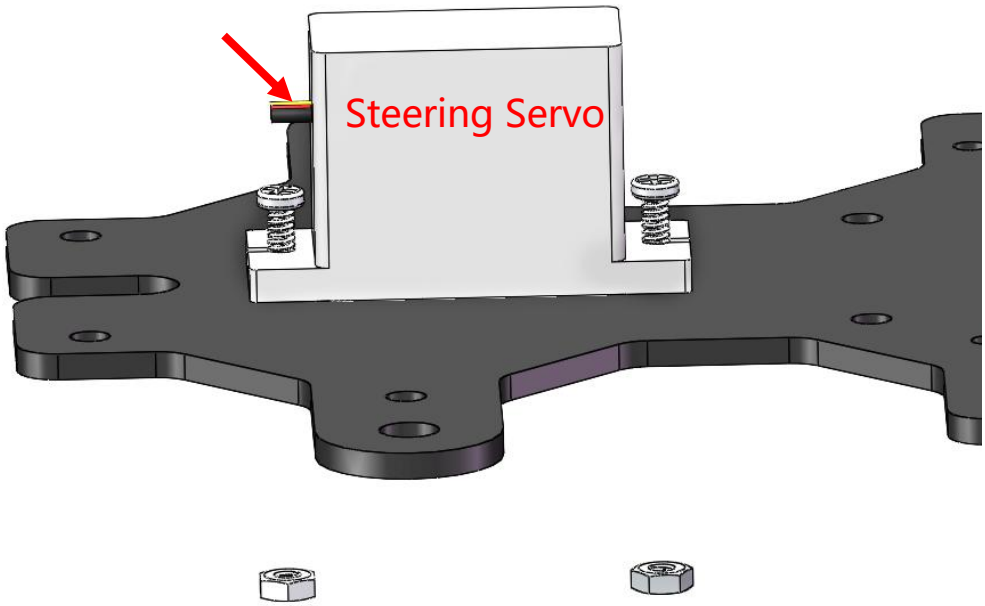
If you find one of the conditions above, please send e-mail to **service@sunfounder.com** . We will change a new one to you. If it is broken in the process of using or assembling, you should go to the official website www.sunfounder.com to buy.

Continue to Assemble

Please do forget to put in the battery and slide the power switch to ON, then keep servo-install running in the whole process of assembly.

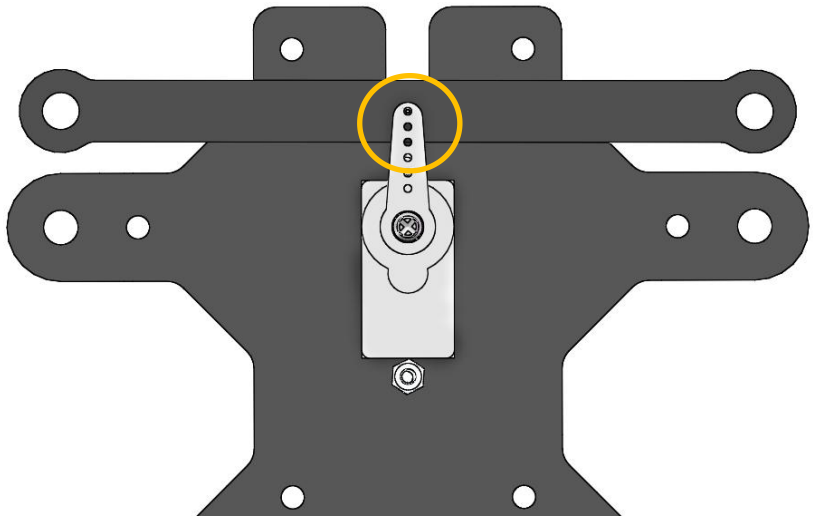
Assemble the Steering Servo

Mount the **Steering Servo** to the Upper Plate with two **M2x8 Screws** and **M2 nuts** (pay attention to the direction of the servo wires):

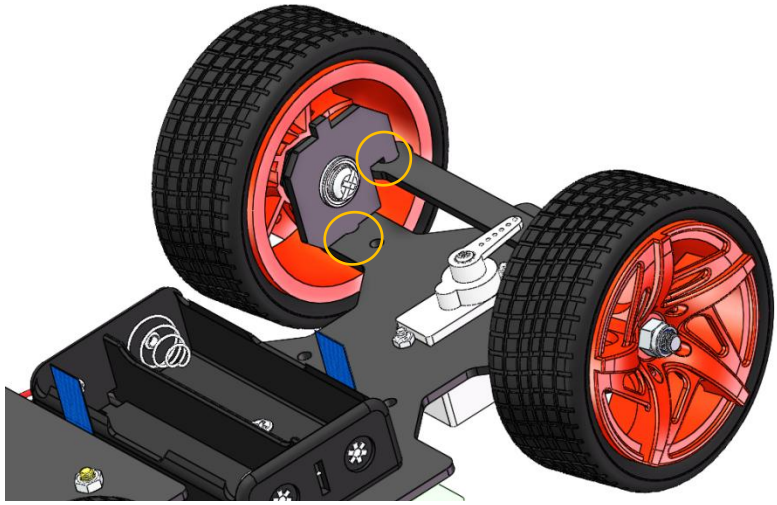


Front Half Chassis

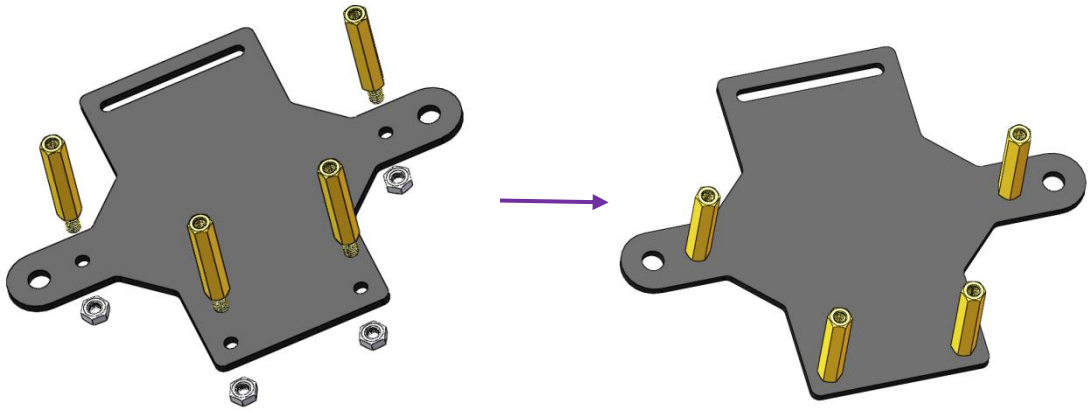
Connect the **Steering Linkage** and the **Rocker Arm** with **Rocker Arm Fixing Screw** (the shortest).



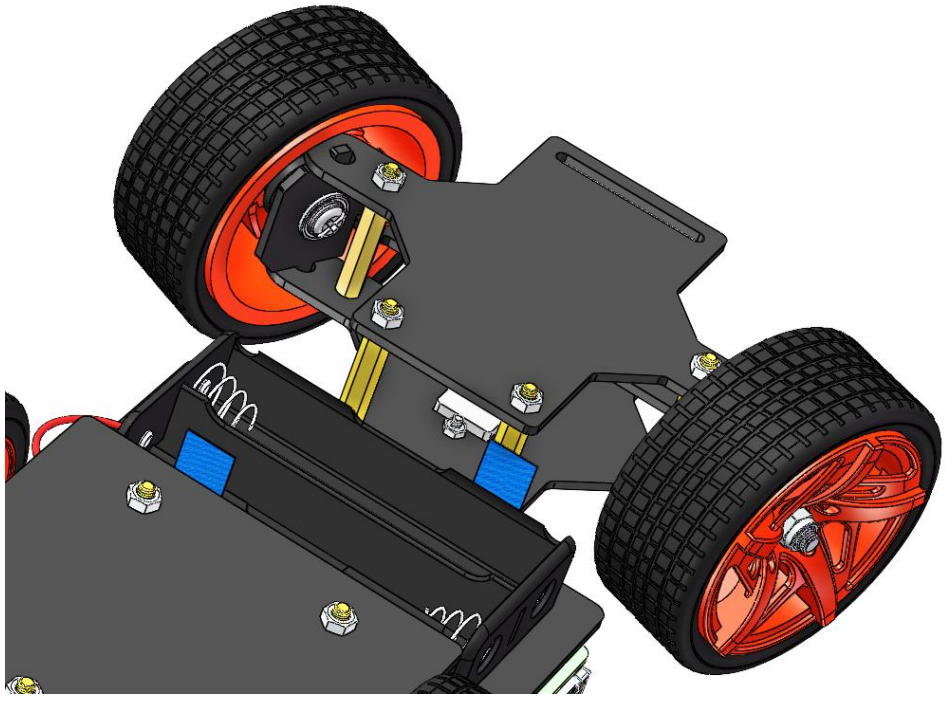
Mount the wheels onto the Upper Plate carefully.



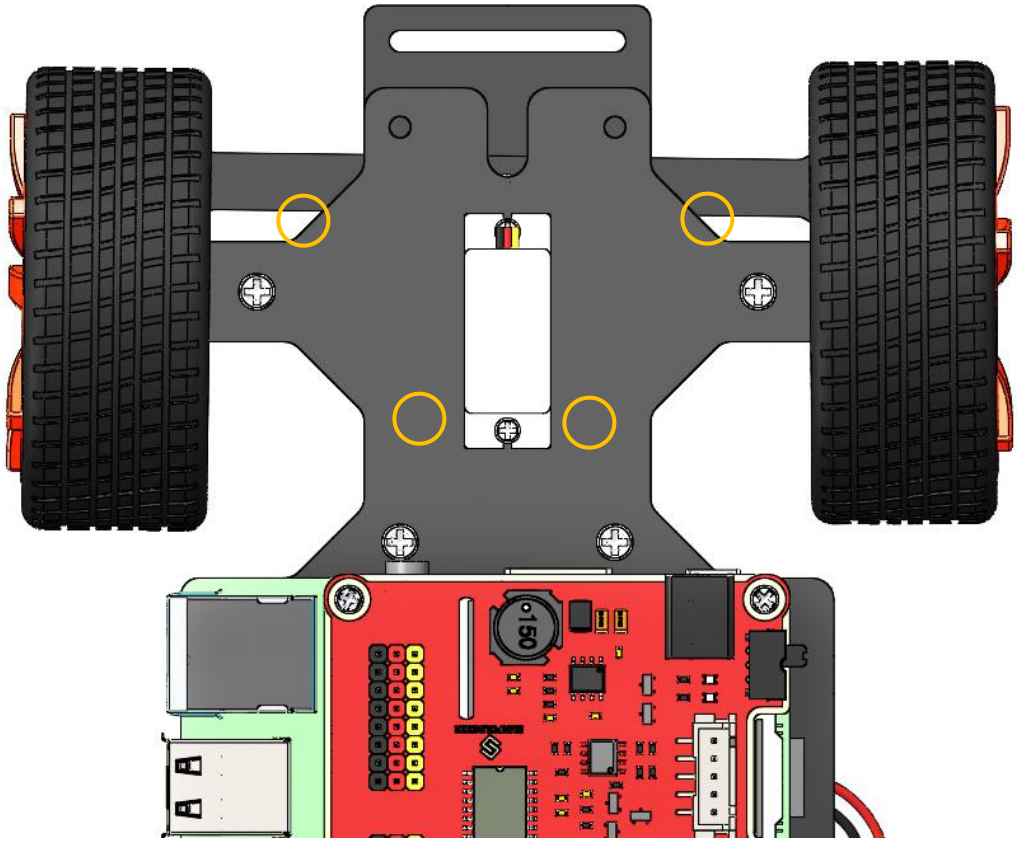
Assemble the **Front Half Chassis** with four **M3x25 copper standoffs** and **M3 nuts**.



Then put the assembled Front Half Chassis onto the Upper Plate with standoffs aligned with the holes.

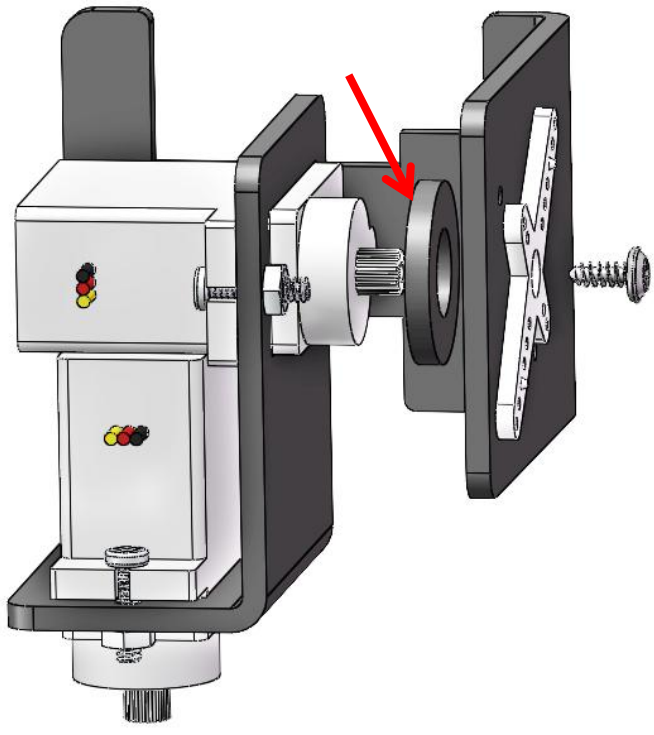


Hold them carefully, turn upside down, and fasten the standoffs and Upper Plate with four **M3x8 screws.**

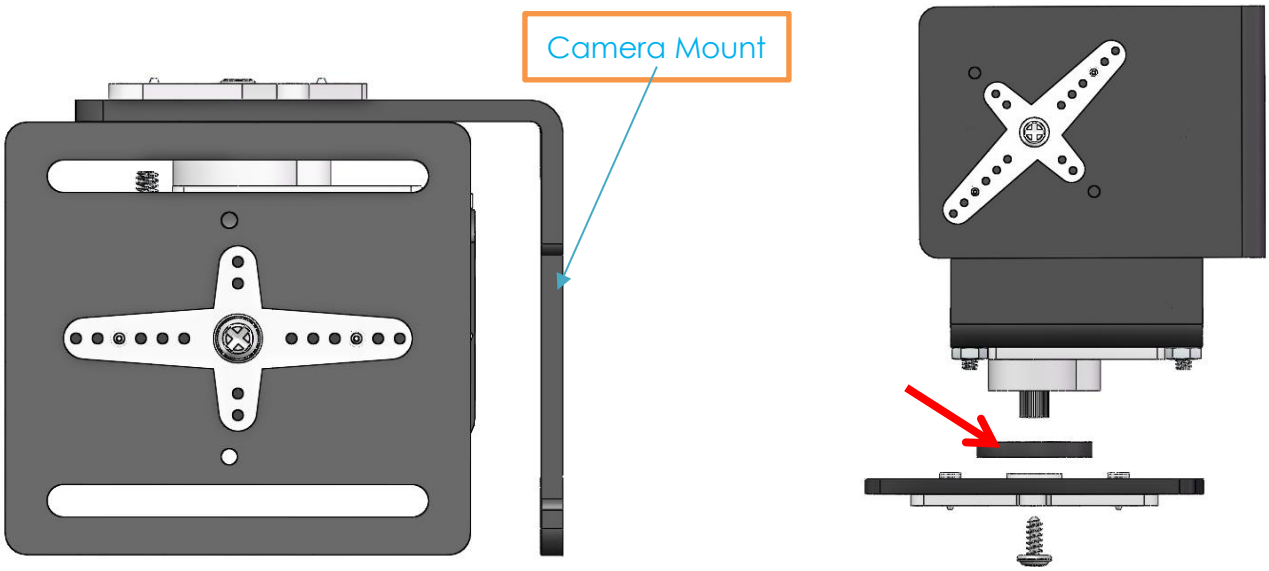


Assemble the assembled **Pan-and-tilt Plate, Gasket Plate** and **Camera Mount Plate** with the **Rocker Arm Fixing Screw** in a fixed angle as shown below:

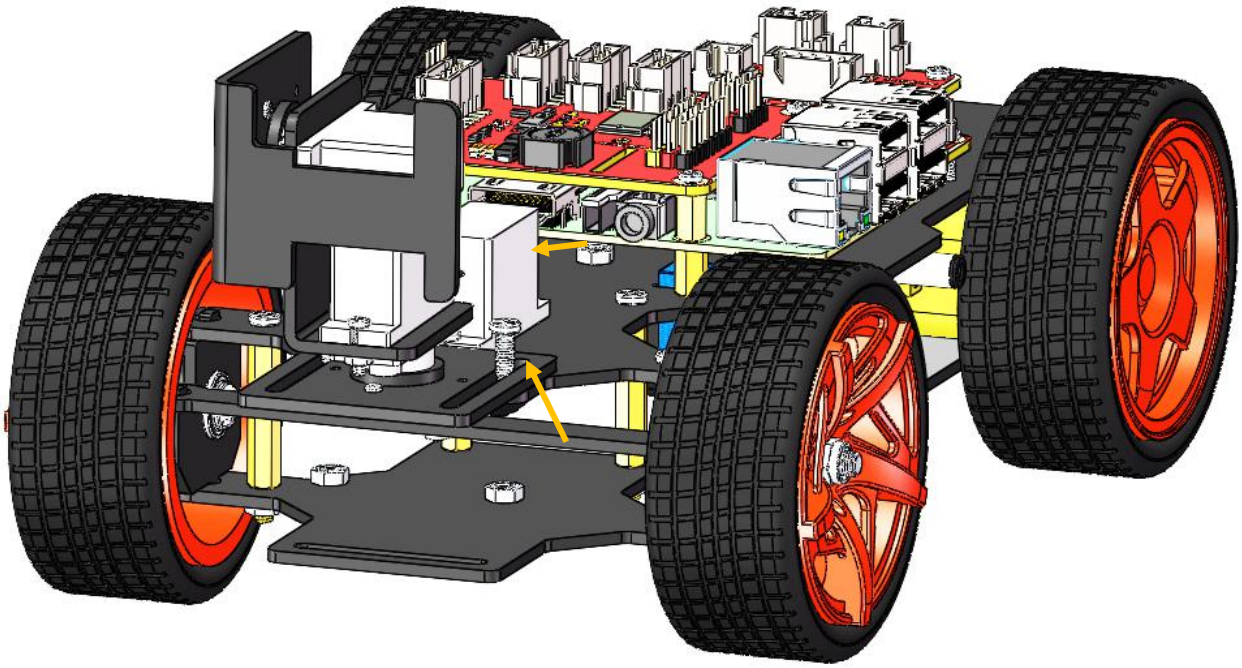
Note: Do not rotate the servo by hands in case of breaking the servo down.



Similarly check that the servo shaft has spun to 90 degrees. Then assemble the **Pan-and-tilt Base Plate** with the **Rocker Arm Fixing Screw** and the **Gasket Plate** in the angle as shown below.



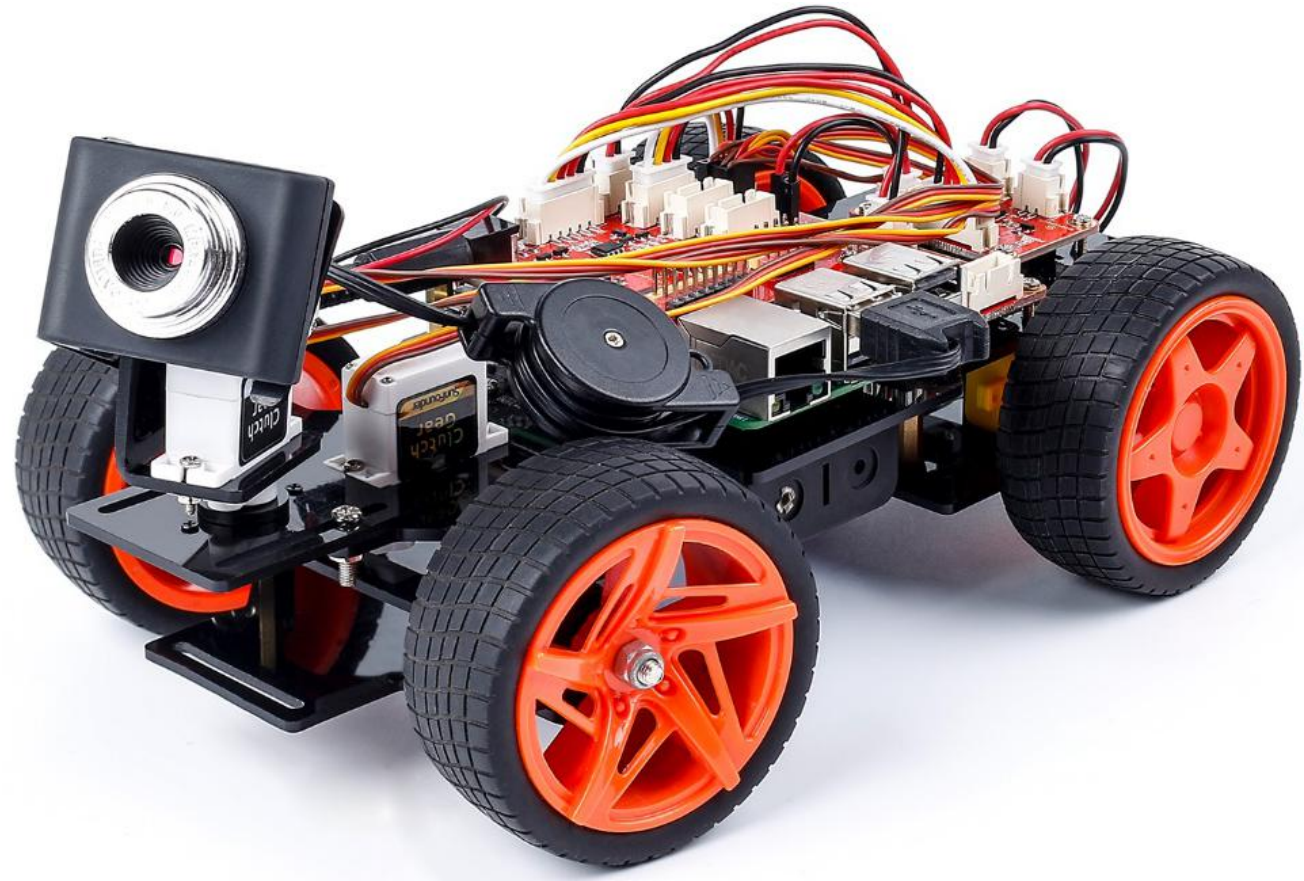
Assemble the **Pan-and-tilt Plate** to the car with two **M3x10 screws** and the **M3 nuts.**



Assemble the Camera

Take out the camera and nip it to the Camera Mount. Connect its USB cable to the USB port on the Raspberry Pi.

So now, the whole assembly is DONE! Congratulations! You can power off the car now! Don’t forget to charge your battery.



Installing the Client (Operation on PC)

Since you've finished the car building, now it's time to configure the environment for the client. Or you may skip this part and go to the subsequent Web Client for controlling on the web page.

The client is written by Python 3 + PyQt5, so it's necessary to install the two in your PC.

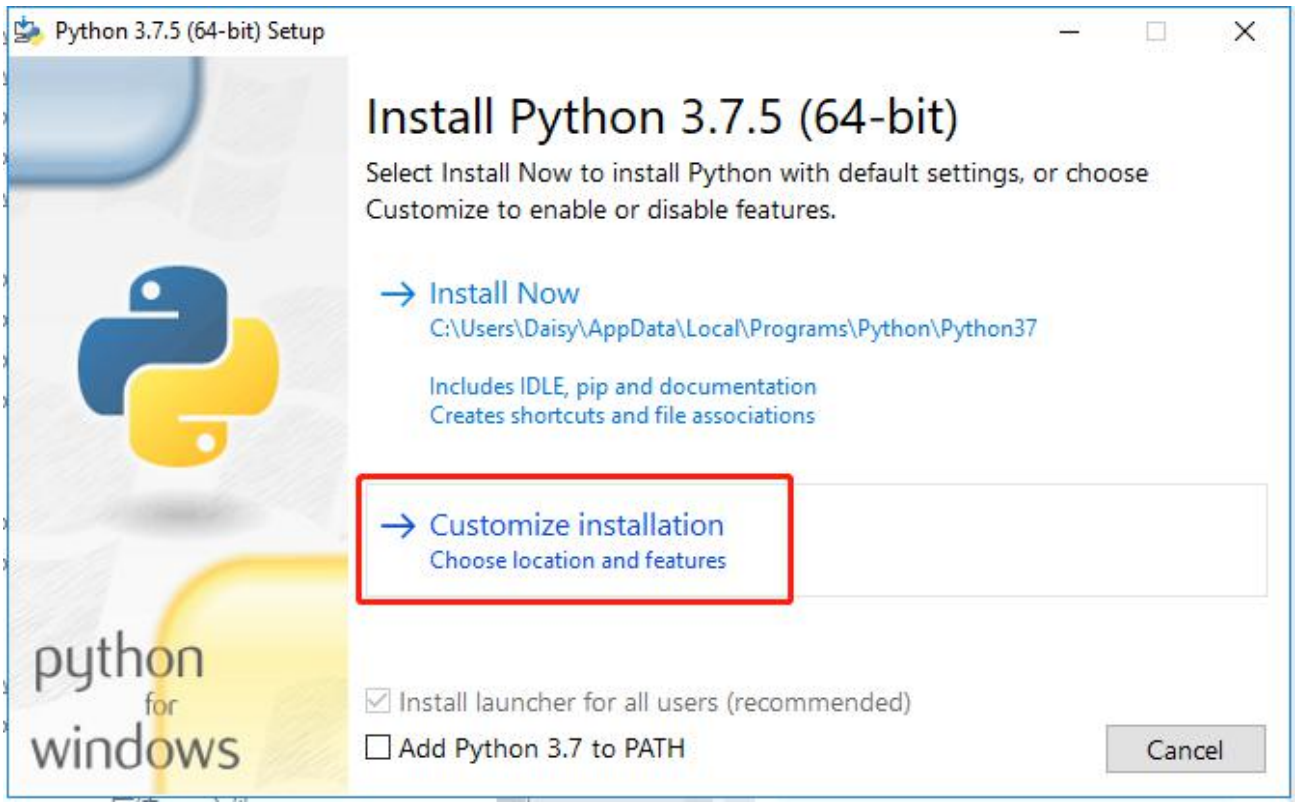
Install Python3

Python 3 website: https://www.python.org/downloads/

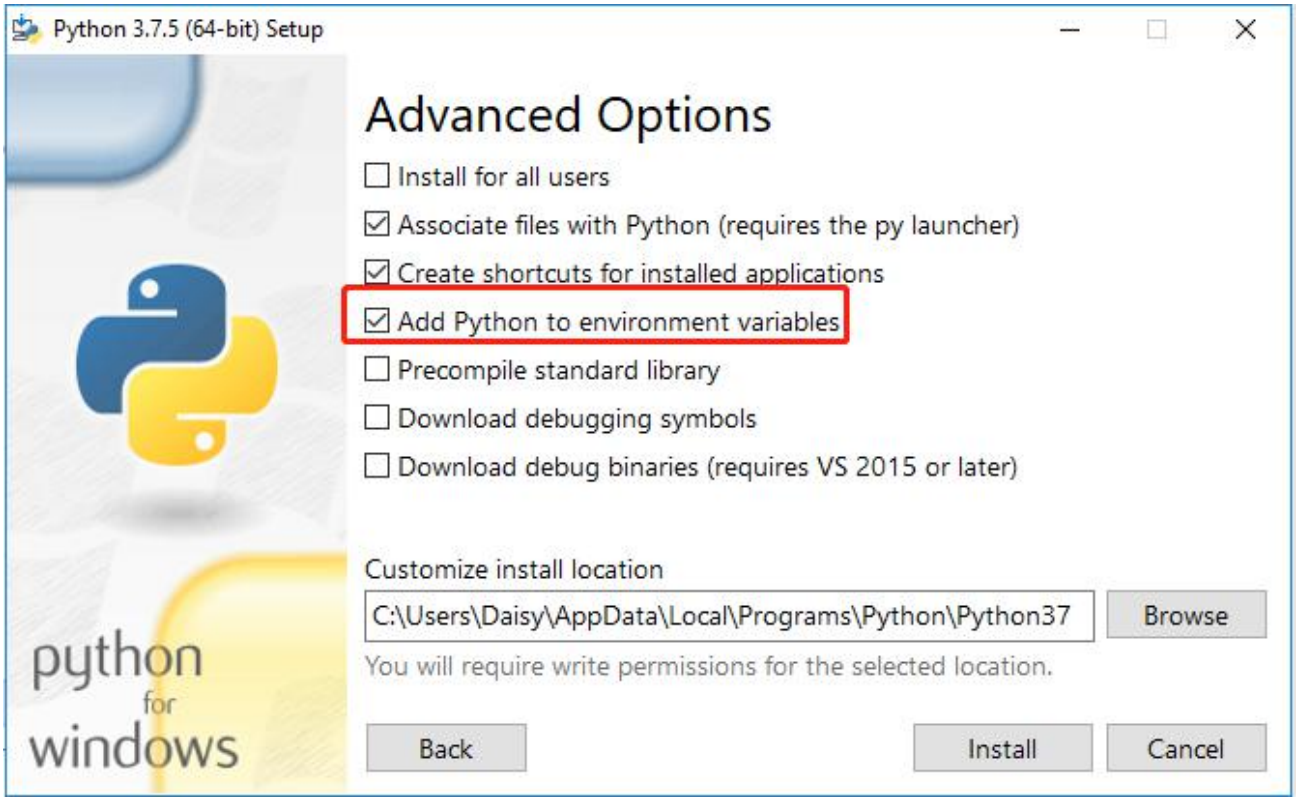
Select the proper version of Python 3 for your PC. Many Linux and Mac OS X computers should have Pyhon3 installed automatically.

➢ For Windows Users

Choose **Customize installation** when you complete installation.



Then click **Next**, check "Add Python to environment variables" in Advanced Options.



Install PyQt5

PyQt5 website: https://www.riverbankcomputing.com/software/pyqt/download5/

➢ For Windows Users

Type in cmd in search box then find the **Command prompt**. Right-click **Run as administrator** and input the following command.

```
pip install pyqt5
```

For the well working of Client, you need to install a requests library.

```
pip install requests
```

➢ For Linux or/Mac OS X Users

The installation method is provided: http://pyqt.sourceforge.net/Docs/PyQt5/installation.html, and the preferred way is installing by pip:

```
pip3 install pyqt5
```

For the well working of Client, you need to install a requests library.

```
pip3 install requests
```

Notes:

1. You may need to install python3-pip first. If you get the prompt of "pip3 not found".

```
apt-get install python3-pip
```

2. The method of pip3 install pyqt5 only support 64-bit **Mac OS X** and **Linux**.

If your PC is running on the 32-bit Linux or other systems, you will be prompted that the corresponding version file cannot be found. In this case, please take the method show on the link:

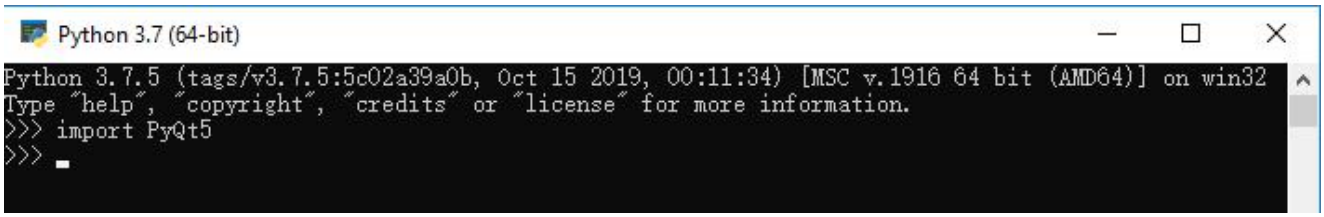
https://www.riverbankcomputing.com/static/Docs/PyQt5/installation.html#building-and-installing-from-source

Check the Installation

Open the Python 3 command line/Python 3 shell after installation, and type in import PyQt5. If there is no error message, then everything is OK:

➢ For Windows users

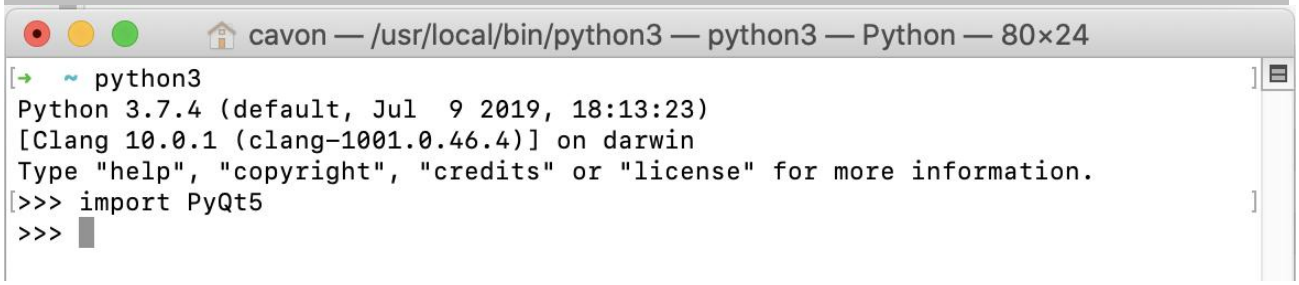
Double click on the Python software, type in *import PyQt5*, and press **Enter** to run:



➢ For Linux or/Mac OS X Users

On **Linux** and **Mac OS X**, type in command *python3* in the terminal, and *import PyQt5* and run.

```
python3
import PyQt5
```



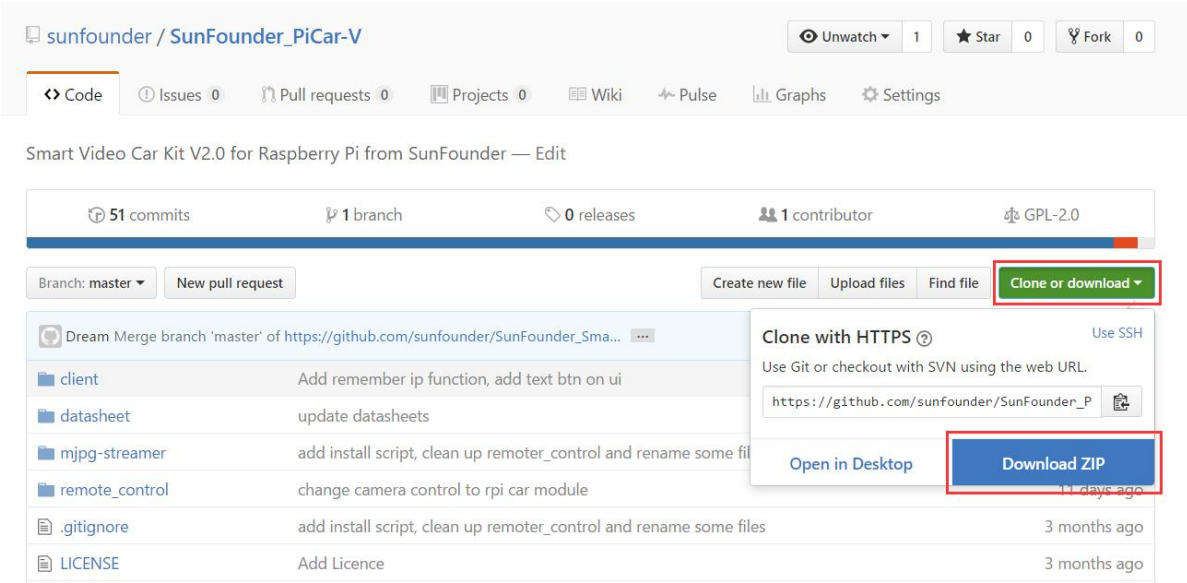
Code Package Download

➢ For Windows users

Open a web browser and go to the Github page of the PiCar-V:

https://github.com/sunfounder/SunFounder_PiCar-V

Click **Clone or download** on the page, and click **Download ZIP** as shown below.



After download, unzip the file.

➢ For Linux or/Mac OS X Users

Or in **Linux**, you can download the code package by git clone (recommended):

Install git with software manager:

Ubuntu/Debian:

```
sudo apt-get install git
```

Fedora/Red Hat:

```
sudo yum install git
```

Then clone the code:

```
git clone https://github.com/sunfounder/SunFounder_PiCar-V.git
```

Note: Please pay attention to your typing – if you get the prompt of entering your user name and password, you may have typed wrong. If unluckily you did so, press Ctrl + C to exit and try again.

Getting on the Road!

Make sure all the procedures have been finished, and check there is no problem about the mechanical assembly and software installation.

Here is what we're going to do:

Use the Raspberry Pi as the **server**. Run a web server with an API for controlling the car and transmitting images captured by the camera.

Then take a PC, cell phone, or tablet as the **client**, and acquire the images from the camera and control the car by calling the API of the web server.

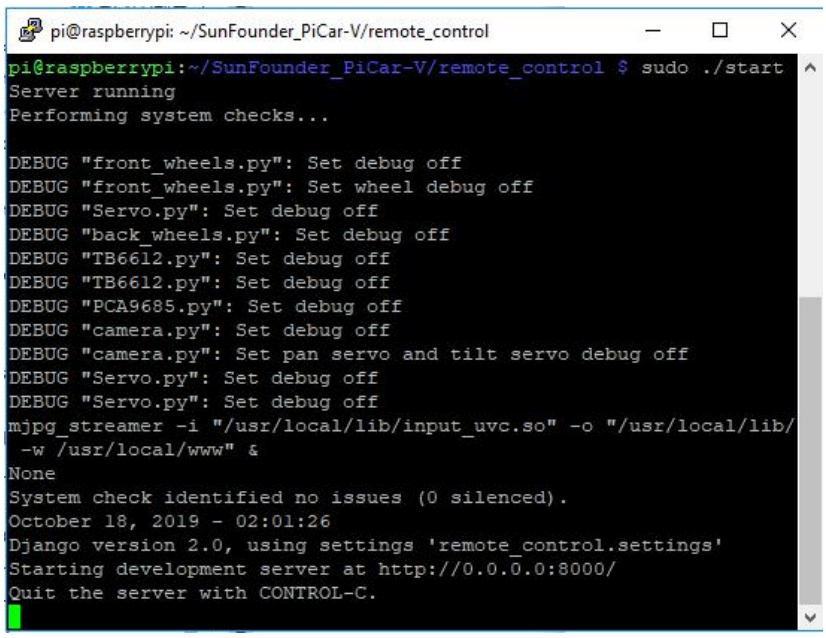
Then you can power on the car again. You are also recommended to use the power adapter of Raspberry Pi to power your car for that the first test will take a long time.

Run the Server(Operation on Raspberry Pi)

Remotely log into the Raspberry Pi. Run the startup script ***start*** under the ***remote_control*** directory to start the web service.

```
cd ~/SunFounder_PiCar-V/remote_control
sudo ./start
```

The script will enable the service and the corresponding data will appear. The hardware is initialized at the same time, so the servos connected to the front wheels and the pan-and-tilt will turn, indicating the hardware initialization is done.



If you get the result similar as shown above, the server is ready. Now move on to start the client.

Keep the server being running all the time until you stop to run the Client.

Run the Client (Operation on PC)

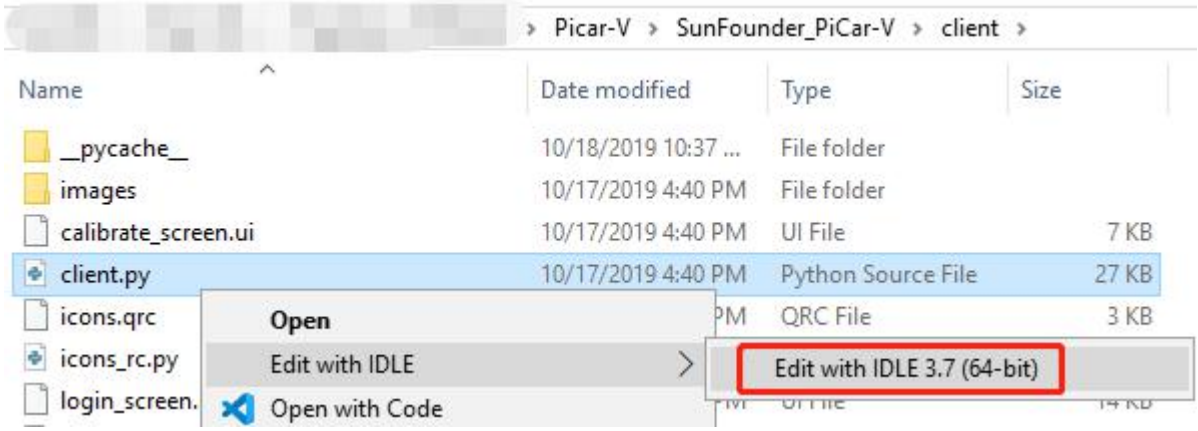
There are **TWO** methods to run the client.

- Method A

➢ For Windows users

If Python 3 has been installed previously, you can right-click **client.py** on the path of SunFounder_PiCar-V\client and then choose **Edit with IDLE 3.7 (Python 3.7)**.

Make sure it runs under Python 3 not Python 2, if both are installed in your PC.



Click **Run -> Run Module F5**, and you will see login interface of the client.

➢ For Linux or/Mac OS X Users

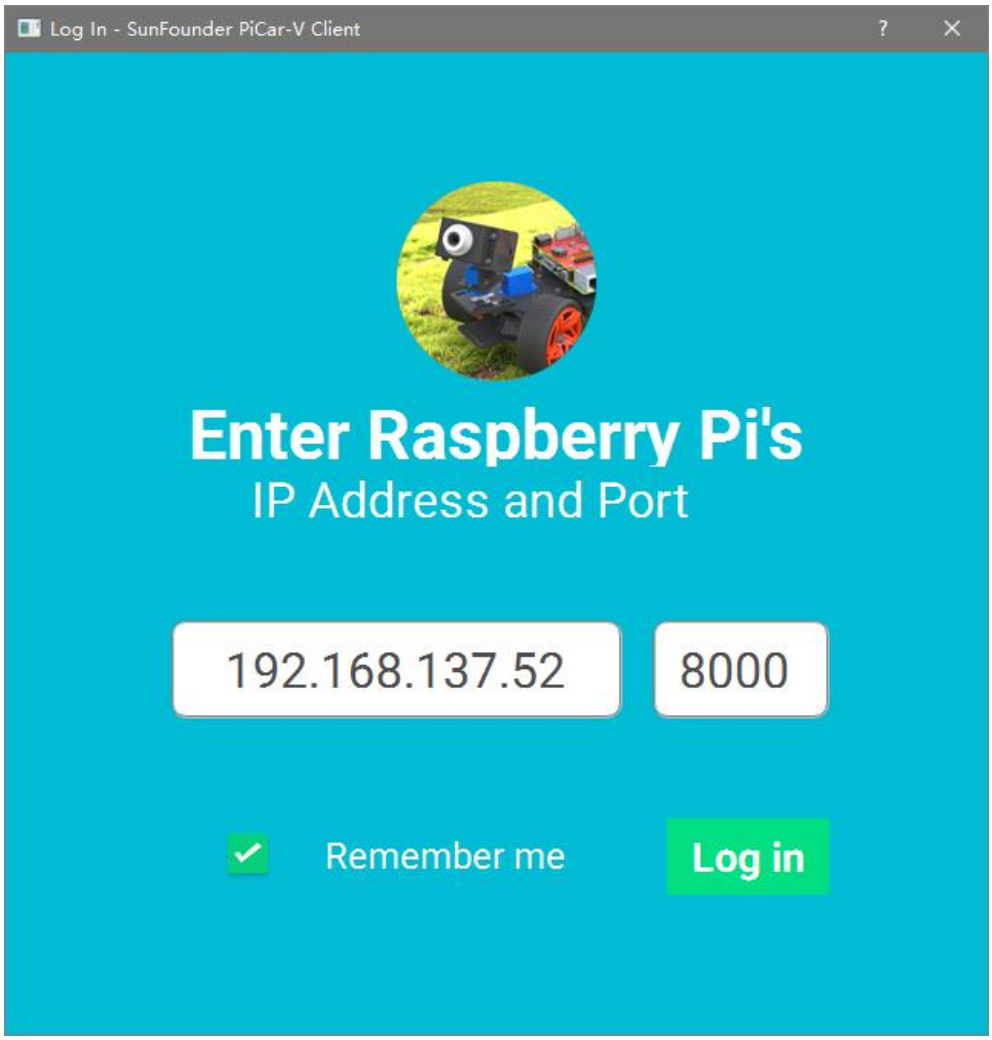
If yours PC runs on **Linux** and **Mac OS X** , you can run python 3 client.py in the terminal.

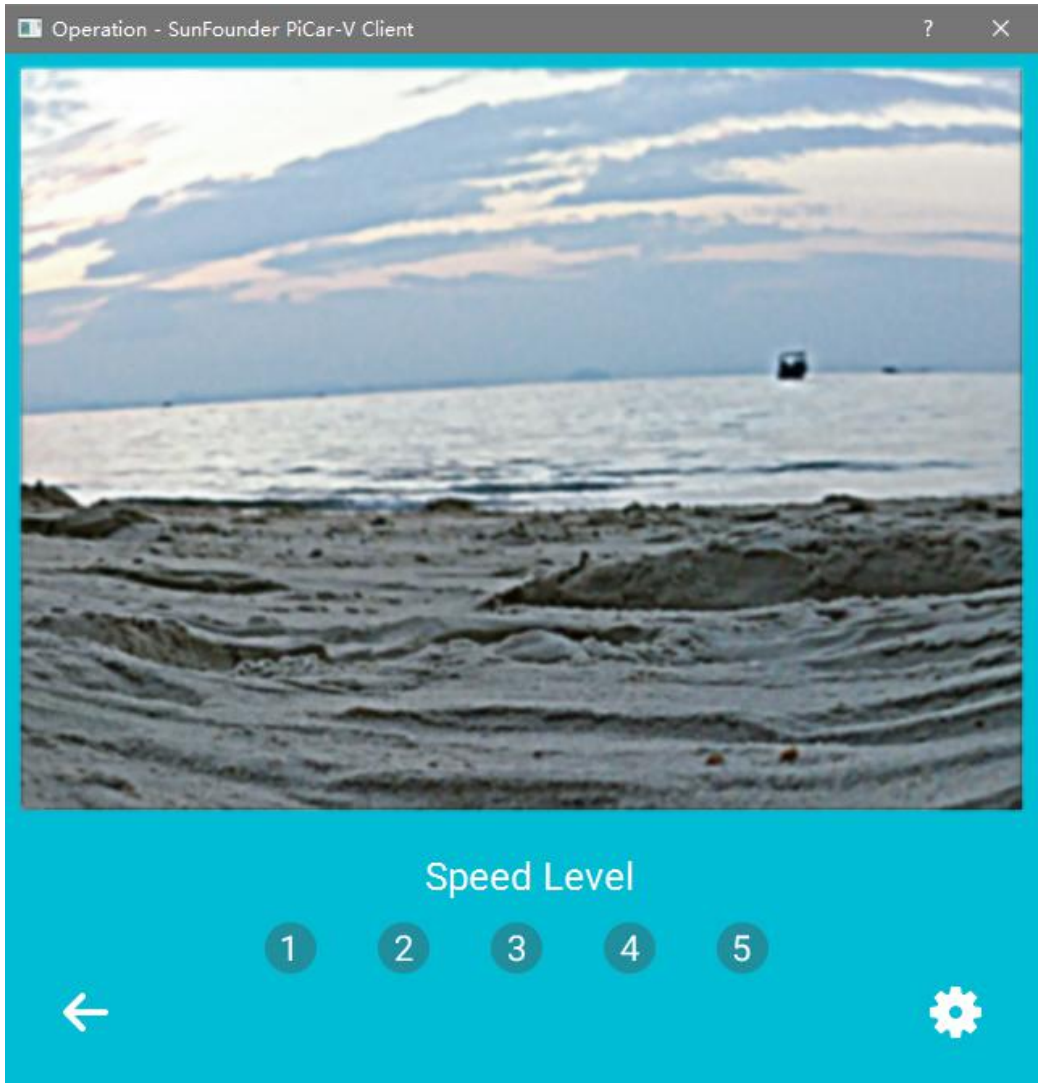
```
python client.py     # If Python 3 is the default or only python
python3 client.py    # Or you have both Python 2 and Python 3, and the default is Python 2
```

Introduction of Client

Run the client and you will see the login interface of the client.

Enter your Raspberry Pi's IP address and the port, 8000 by default, and then click **Log in** to connect to the Raspberry Pi. It may take a while to establish connection, sometimes may not respond at all, so just wait several seconds. If it's successful, you'll be on the operation interface. If not, double check your Raspberry Pi's IP address and make sure the server script is run without error prompts.





The interface will show the view captured by the camera in a real-time manner.

There are some number buttons at the bottom to adjust the car's speed, ascending from level 1 to 5. You can also press the key 1-5 on the keyboard. The rest control movements of the car are implemented by keyboard.

On keyboard:

W, **S**: to control the rear wheels to move forward/backward;

A, **D**: to control the front wheels to turn left/right;

↑, ↓: to control the camera's rotating up/down;

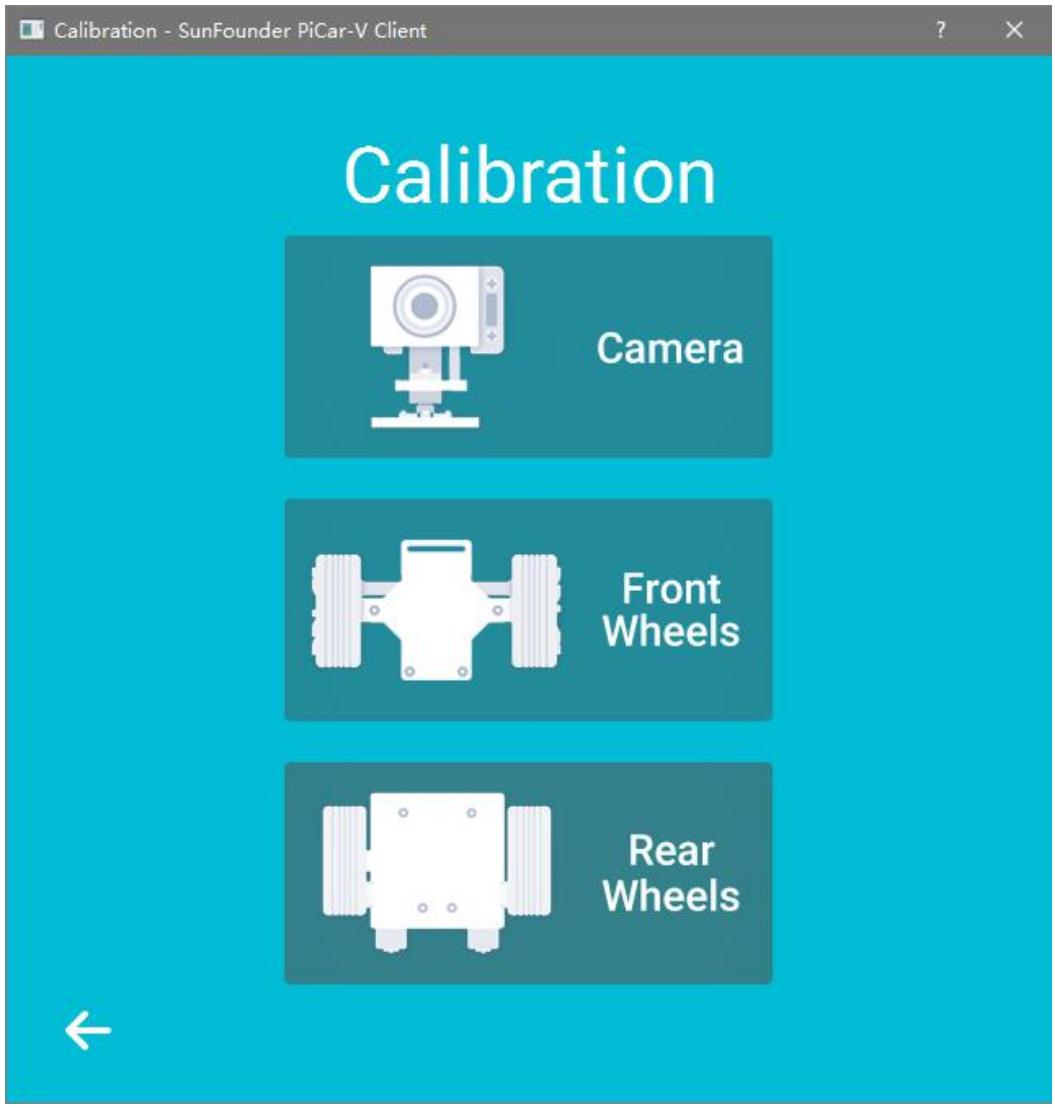
←, →: to control the camera's rotating left/right.

Click the left arrow button at the bottom left or press **Alt** + ← to return to the previous page, that is, the login.

For better application, you're recommended to calibrate the assembled car first.

Calibration

Click the setting button at the bottom right or press **Alt** + **S**, and you will see the **Calibration** page as below (click the three larger icons for the corresponding calibration):



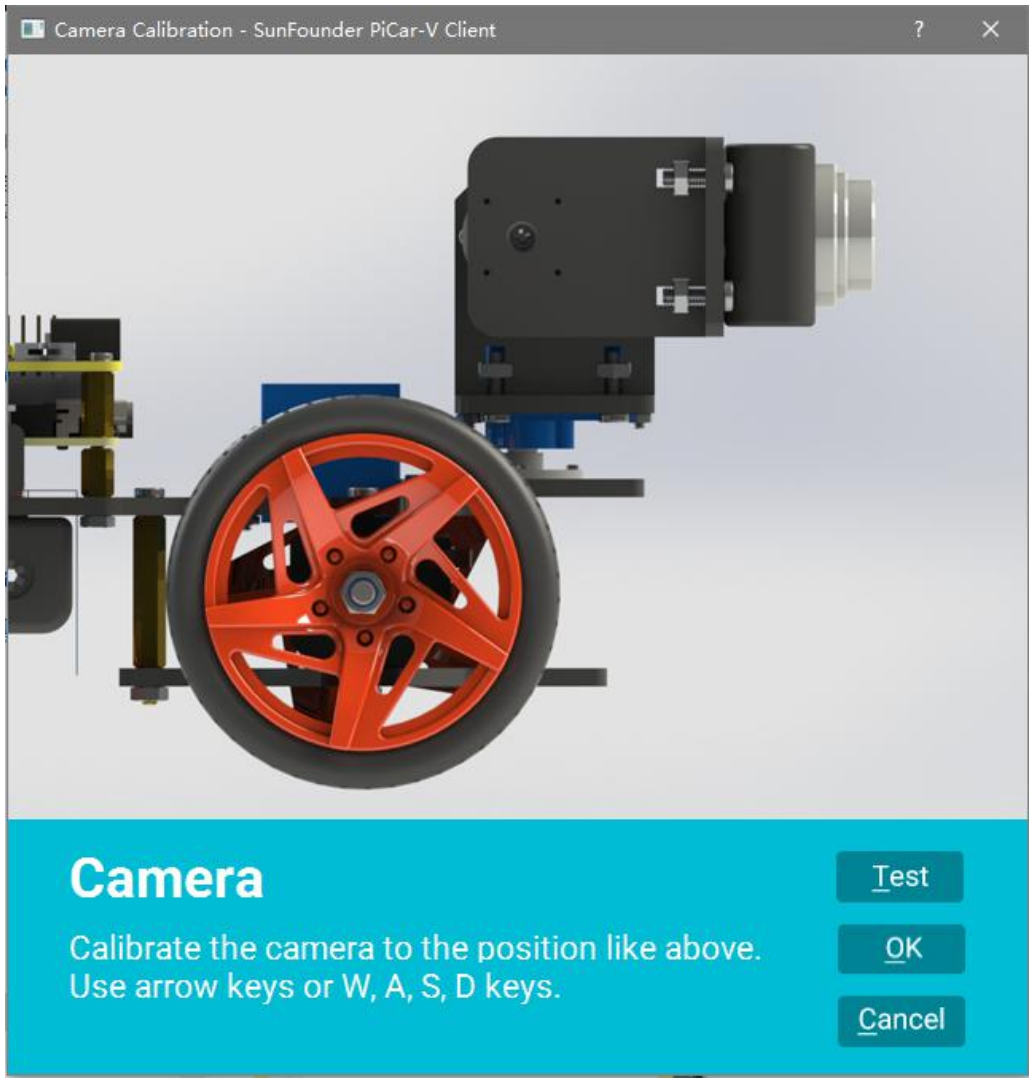
Camera – to calibrate the camera movement. Short cut: **Alt** + **C**.

Front Wheels – to calibrate the front wheels. Short cut: **Alt** + **F**.

Rear Wheels – to calibrate the rear wheels. Short cut: **Alt** + **R**.

Click the arrow return button or **Alt** + ← and you will go back to the operation interface.

Calibration: Camera.



Click Test button，the camera will rotate. You can use the the arrow keys ↑, ←, ↓, →, or **W**, **A**, **S**, **D** to adjust the angle of the camera.

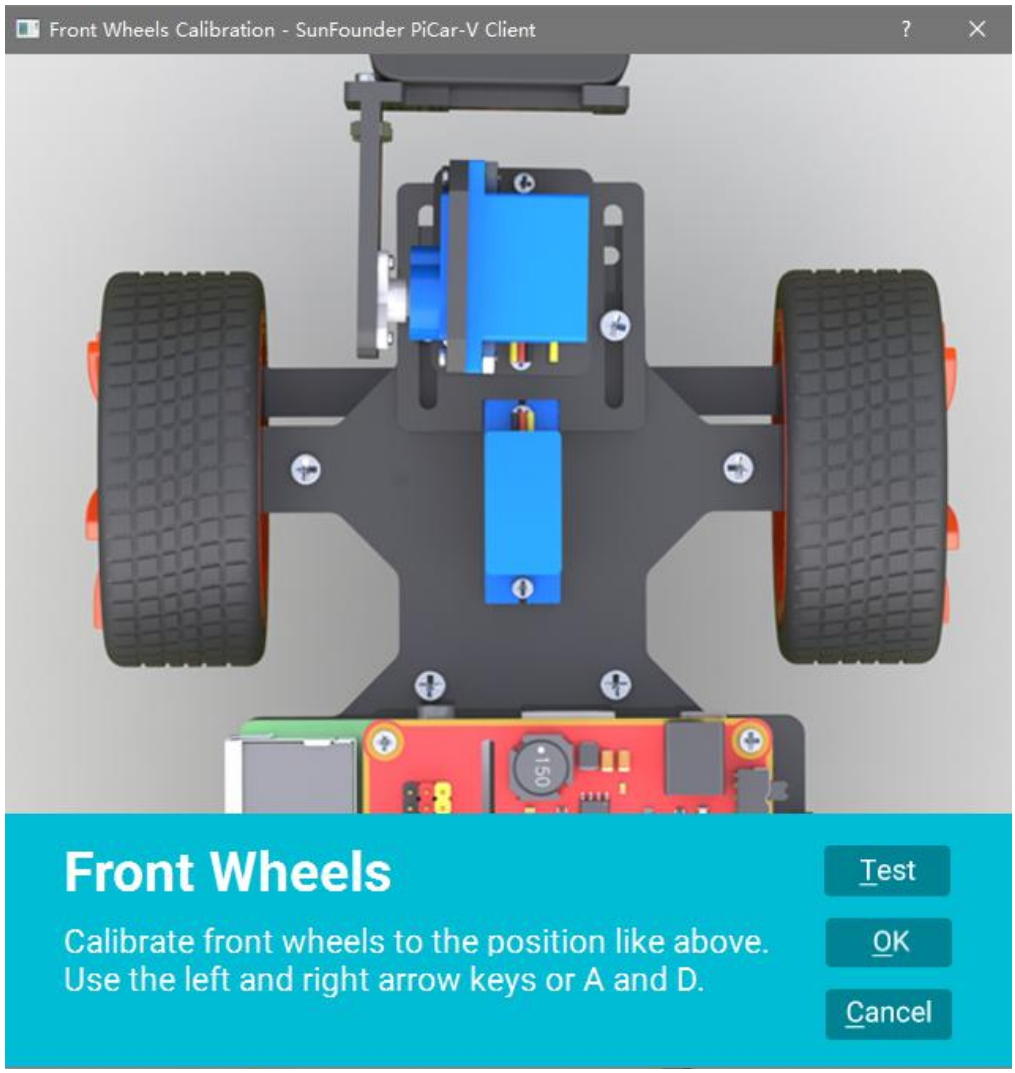
Click **OK** or press **Alt** + **O** to save the calibration result; click **Cancel** or press **Alt** + **C** to cancel the calibration and exit.

Notes:

1）Every time you press, you can get a small angular adjustment; for a large one, you need to long press the button.

2）During the calibration, the servo may get stuck with an abnormal sound, and heated abnormally after a while if the servo shaft is not adjusted to 90° in the previous mechanical assembly. You should at once disconnect the wires of the servo and remove the rocker arm fixing screw and then install the servo according to the above picture.

Calibration: Front Wheels.



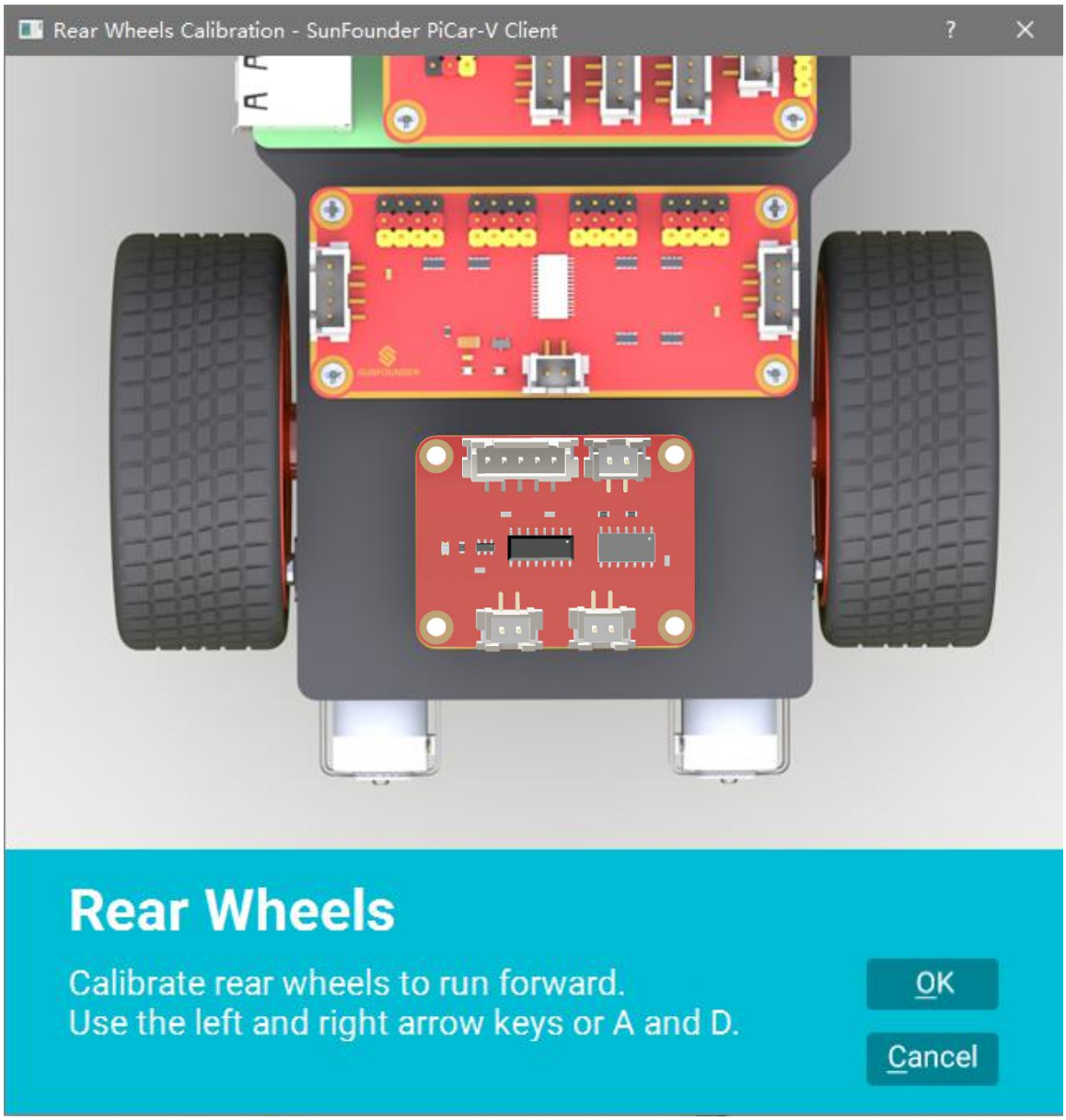
Click Test button, the front wheels will turn left and turn right if there is any direction deviation over the front wheel.

you can use the keyboard to adjust, that is, ←, → or **A**, **D**.

Click **OK** or press **Alt** + **O** to save the calibration result; click **Cancel** or press **Alt** + **C** to cancel the calibration and exit.

If the servo gets stuck, remove the rocker arm and readjust again (refer to the **Servo Configuration**).

Calibration: Rear Wheels

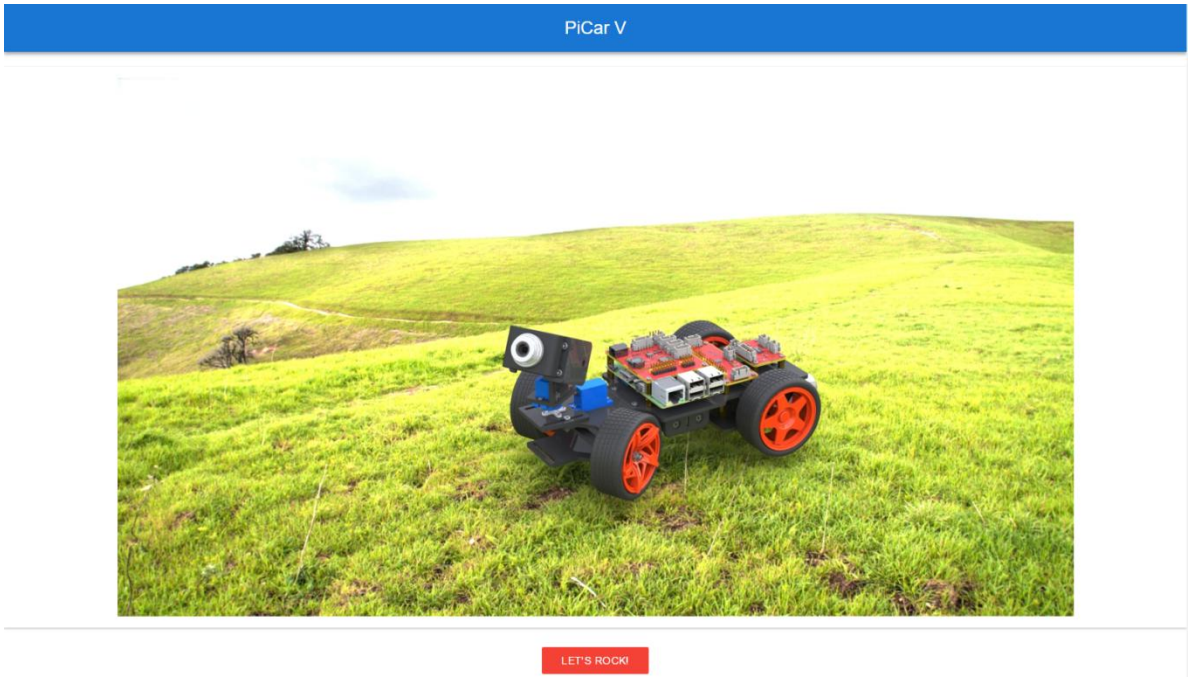


When you enter the page, the car goes forward. If not, you can adjust the rotation of the left and right wheels by using ← **or A** and → or D separately.

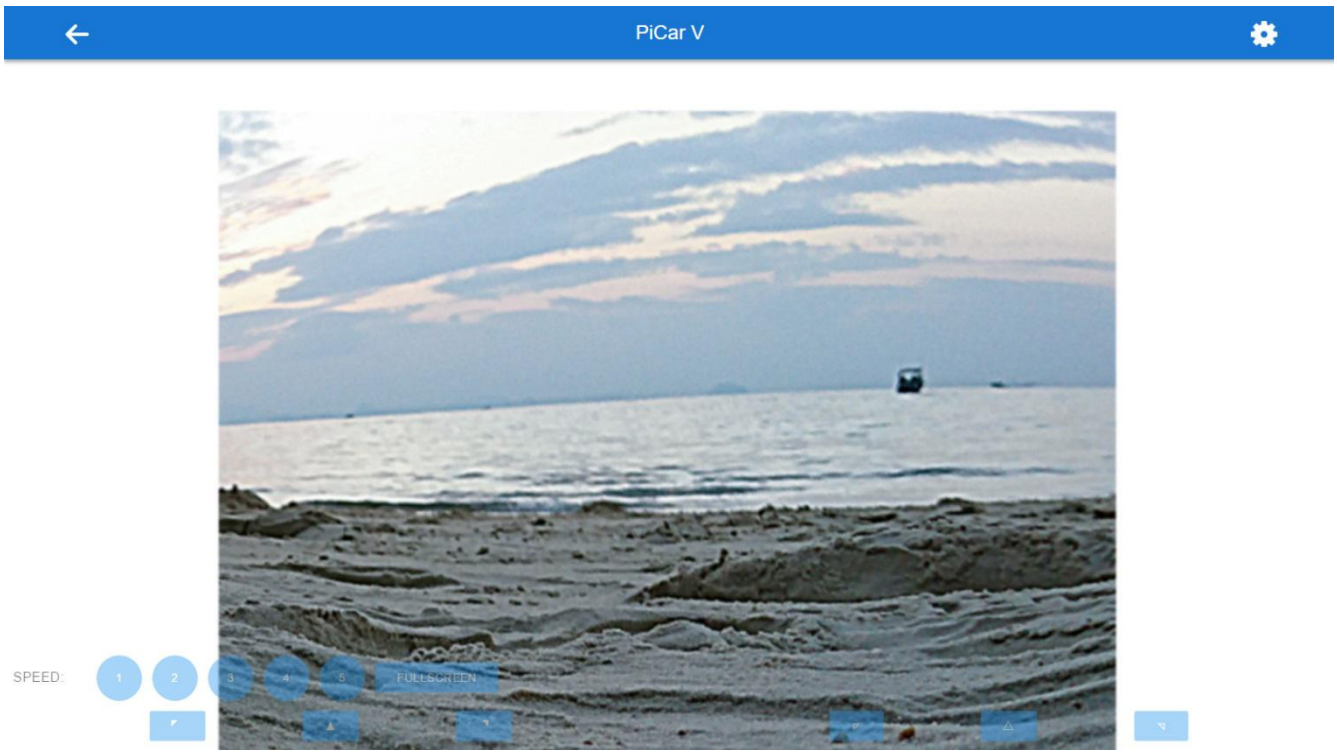
After all the calibration is done, you can return to the operation interface. Make sure both LED indicators on the Robot HATS board light up. Now you can unplug the power adapter and put the car on the ground. Let's start the journey!

- Method B

Visit the server of the car at http://<RPi_IP_address>:8000/. You will see a welcome page:



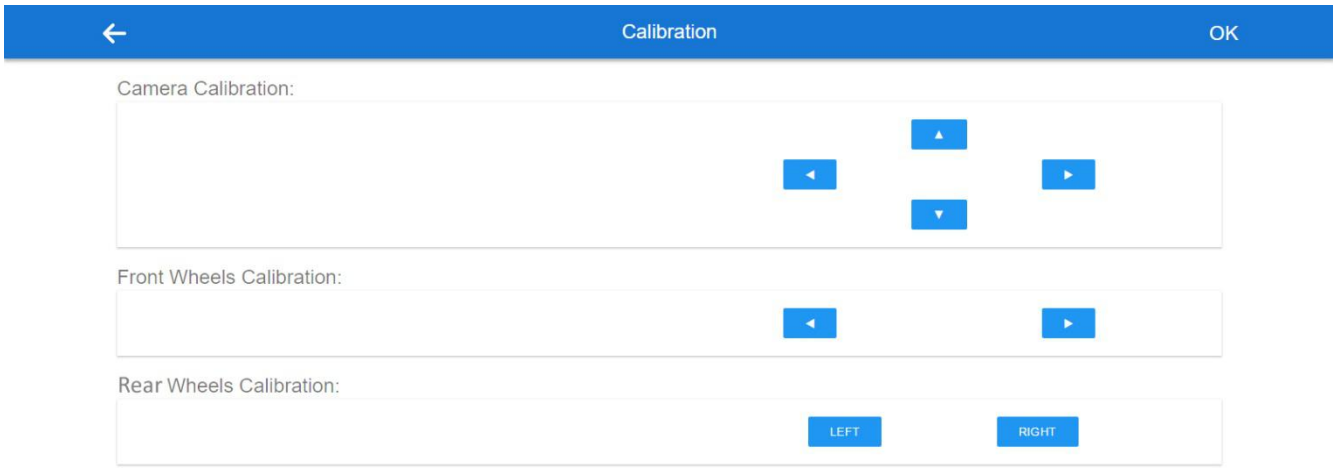
Click **LET'S ROCK** to go to the operation interface:



On this page, you can press the keys **W**, **A**, **S**, and **D** on the keyboard to control the car to move **forward**, **backward**, **turn left**, and **turn right**, press the arrow keys to control the camera's movement, and number **1~5** to change the speed level.

Calibration

Click **FULLSCREEN** again to bring back the title bar. And then, tap the setting button at the top-right corner of the page to go to the calibration page:



There are three calibration parts: **Camera Calibration**, **Front Wheels Calibration** and **Rear Wheels Calibration**.

When you enter this page, the car will go forward; if not, click **Left** and **Right** in **Rear Wheels Calibration** to adjust the angle of the wheel.

Notes:

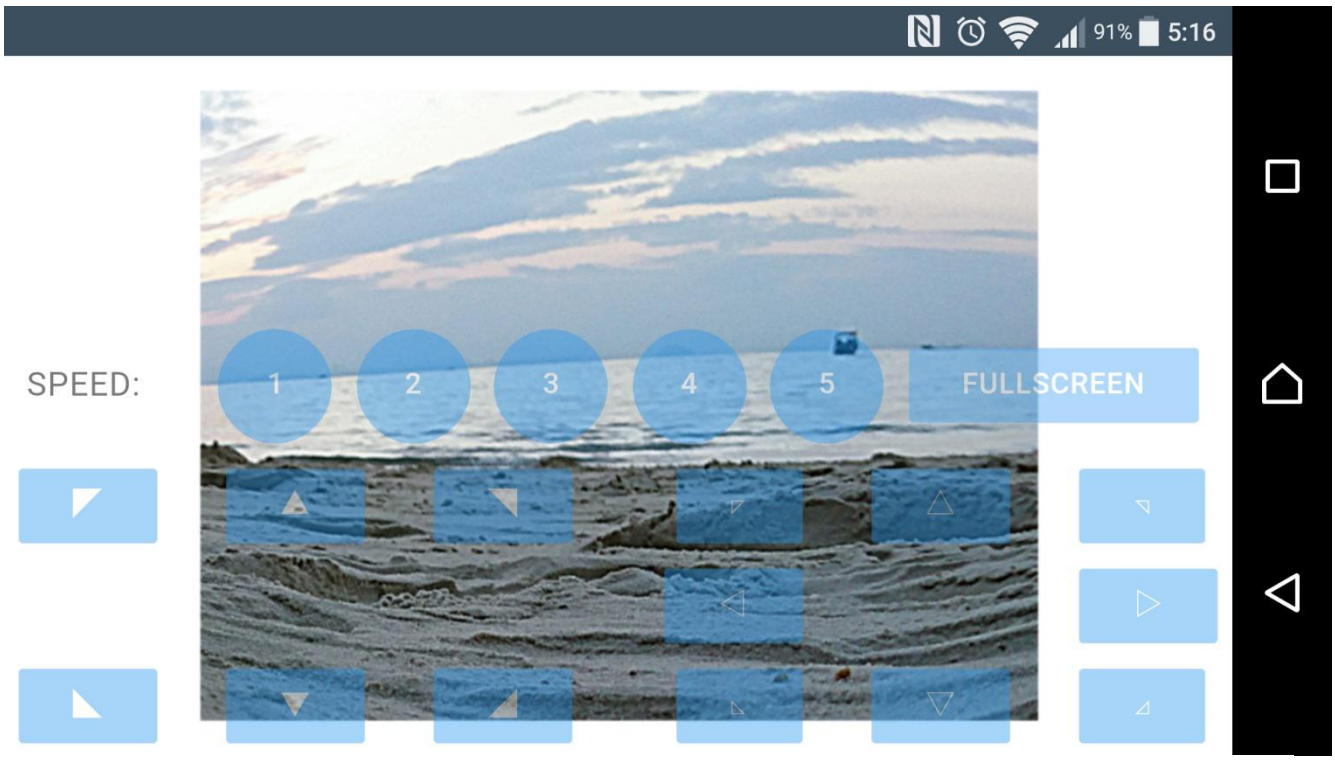
Every time you press a button, the angle will be changed slightly. For your larger change per time, you need to long press the button.

Click **OK** to save the result after all the calibration is done.

For Mobile Phones

Also for mobile phones, tap the **FULLSCREEN** button to have a better view and performance. Then, tap the buttons of 5 speed levels on the page to control the speed, and the arrow buttons to control the direction of the car and the pan-and-tilt. But you can only tab one touch point at **one** time.

Take this screenshot from an Android phone:



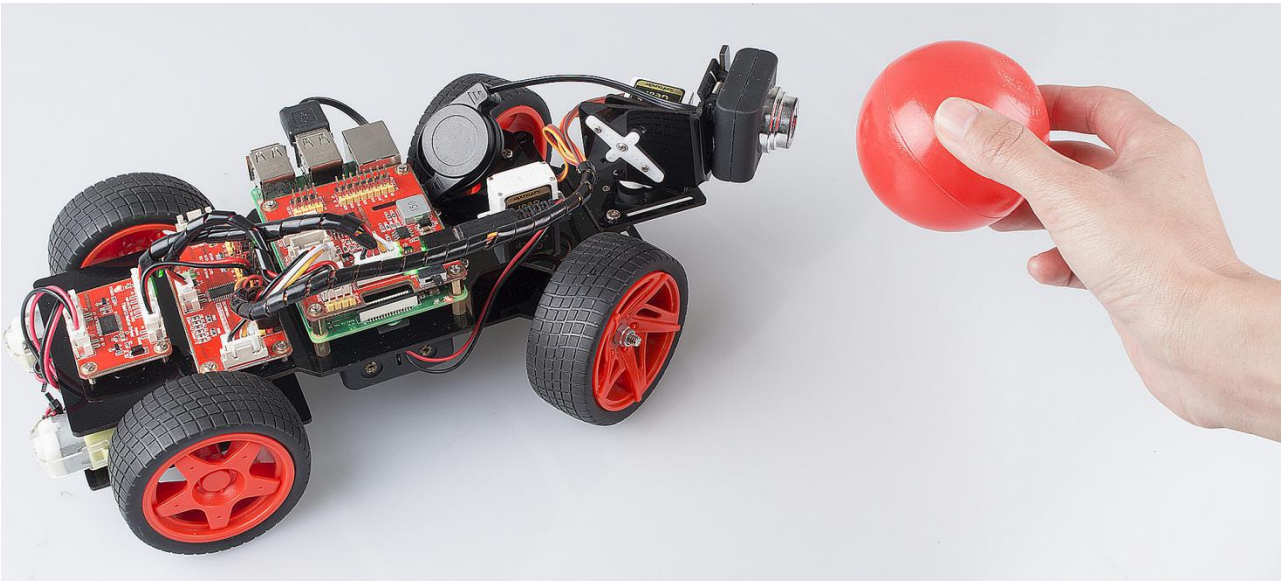
Though the appearance may not be as good as on the PC. Later updates may be released irregularly on **Github**. You are welcome to fork our repository and submit a Pull request with your changes. If there is no problem after testing, we are more than pleased to merge your request.

ball_tracker

In addition to using the keyboard to control the status of the car, we also wrote a ball tracking code.

```
sudo apt-get install libatlas-base-dev libjasper-dev libqt4-test libwebp6 libtiff5 libopenexr23 libgstreamer1.0-0 libavcodec-dev libavformat-dev libswscale-dev libqtgui4 -y
sudo pip3 install opencv-python==3.4.6.27
cd ~/SunFounder_PiCar-V/ball_track
python3 ball_tracker.py
```

After the code runs, find a red ball (at least larger than the diameter of the camera) and put it within 10-50cm range in front of the camera to get the car to follow your ball.

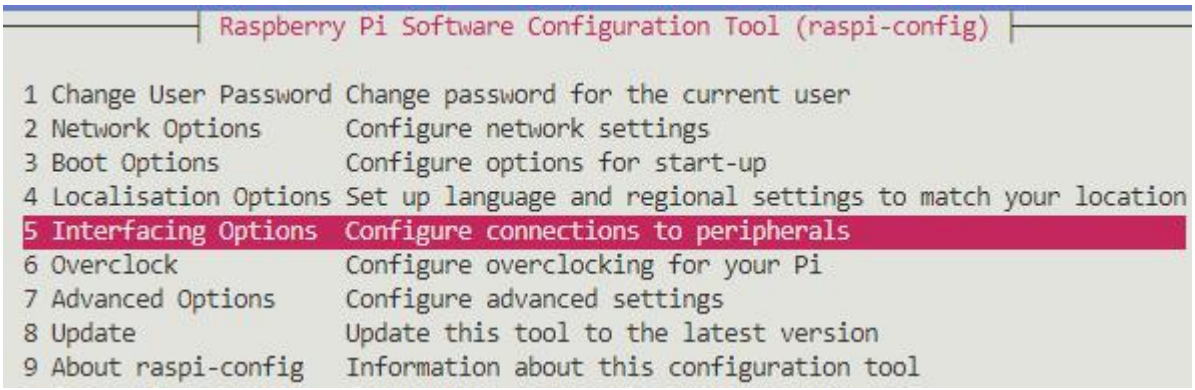


If you want to know whether the car has catched your ball, you can do as follows:

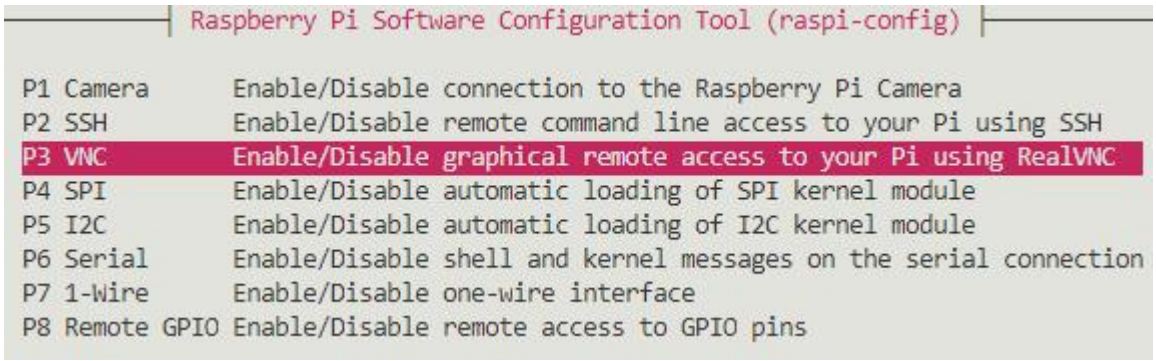
Step 1: Enable the VNC on the Raspberry Pi.

```
sudo raspi-config
```

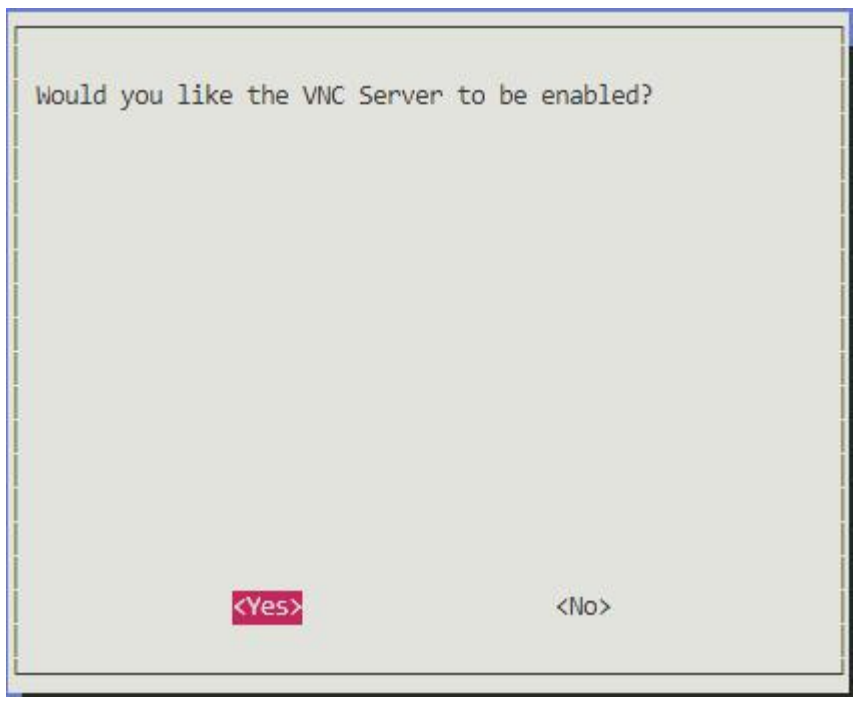
Choose 5 Interfacing Options.



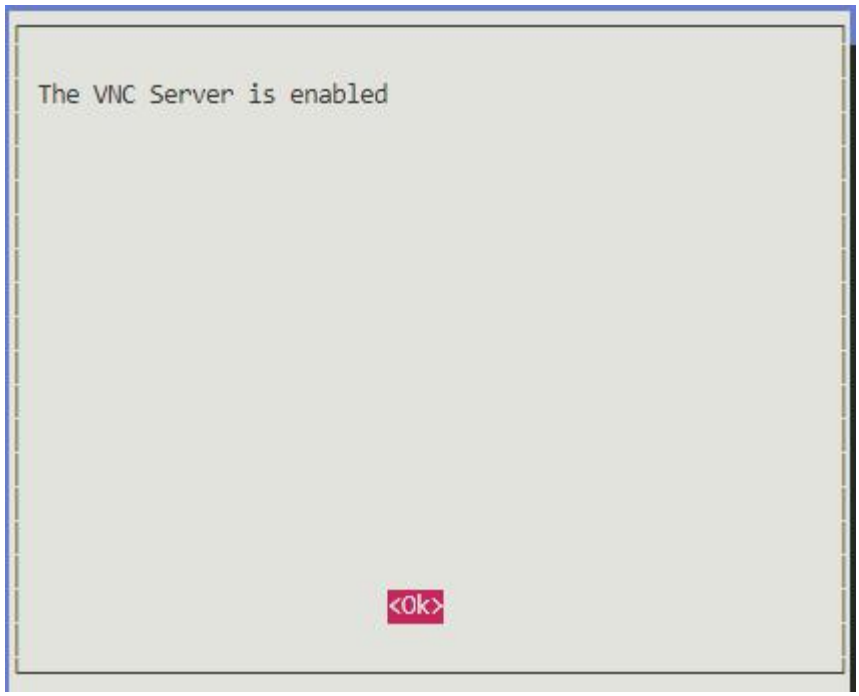
P3 VNC



Yes



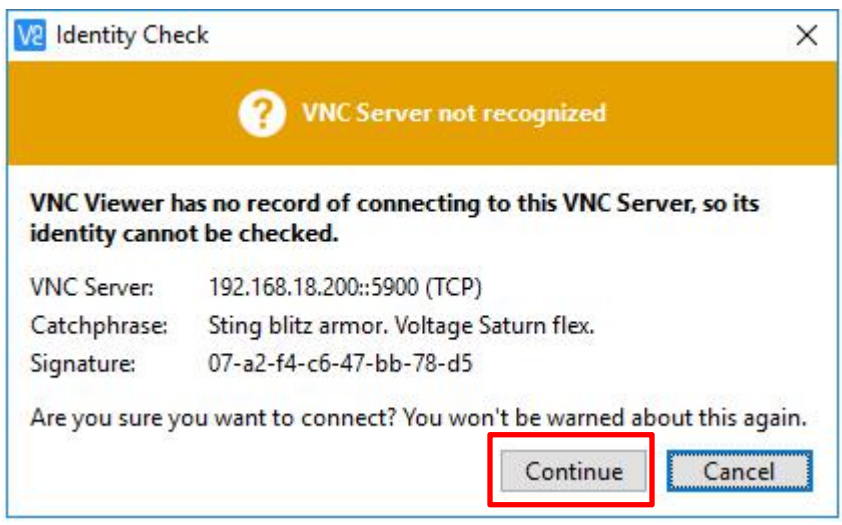
Finally select **OK**->**Finish** to exit the config.



Step 2: Download the VNC Viewer for your computer and install it.

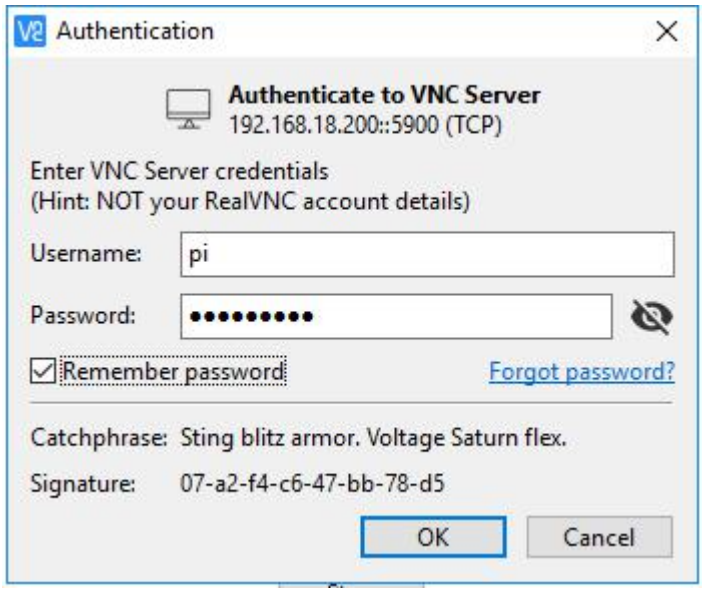
https://www.realvnc.com/en/connect/download/viewer/

Note: During your installation on your Windows, if you encounter the following prompt, please click **Comtinue** to continue your installation.



Step 3: Open VNC Viewer you have installed, type in your Raspberry Pi IP address, and press the Enter button.

Step 4: Enter the username and password of your Raspberry Pi on this page, by default, they are pi and raspberry; then click OK.



Step 5: Now, we enter the Raspberry Pi desktop, click Terminal and type the following commands to open the code ball_tracker.py.

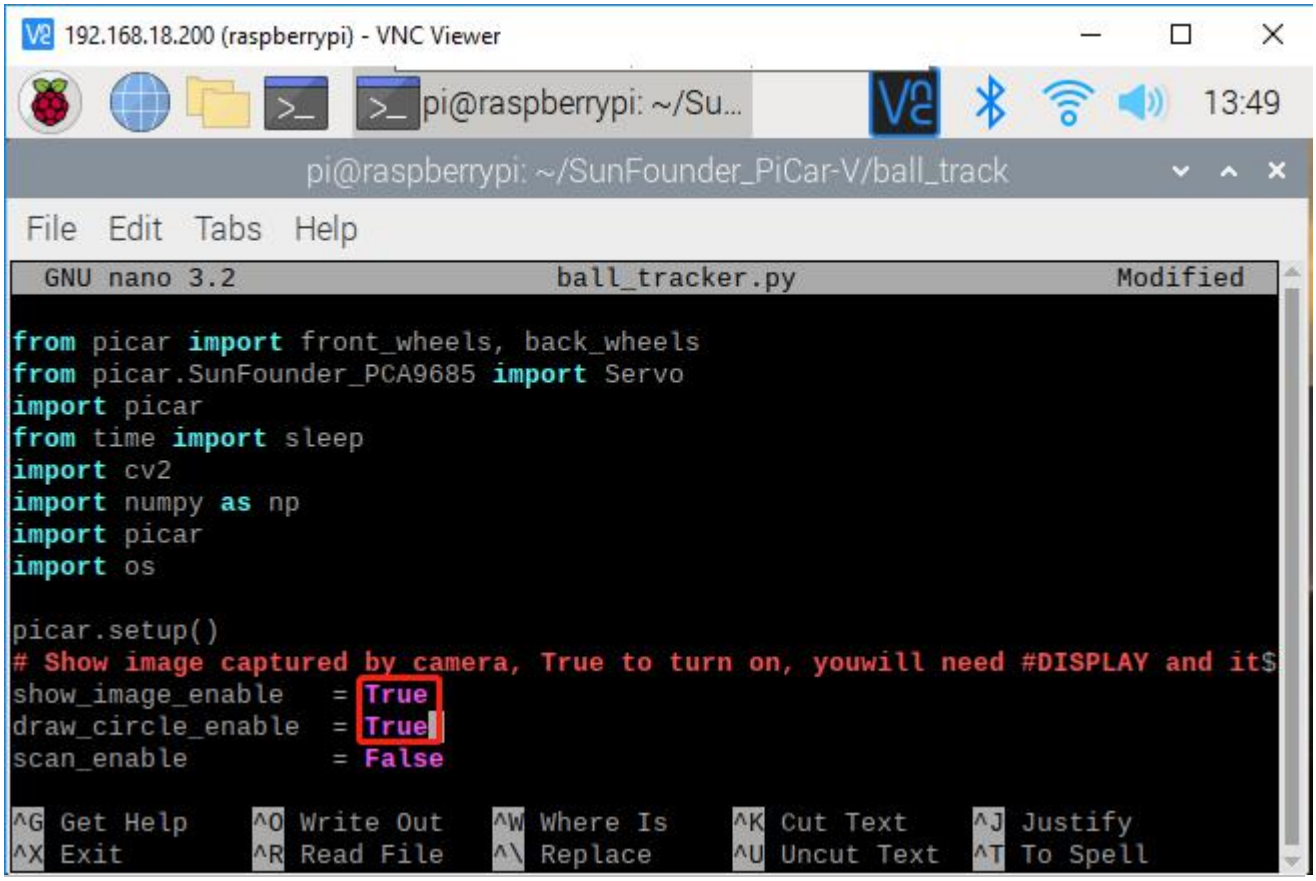
```
cd ~/SunFounder_PiCar-V/ball_track
sudo nano ball_tracker.py
```

Then modify the code as follows:

```
Show_image_enable = True

Draw_circle_enble = True
```

Press Ctrl+X and Y to save the change.

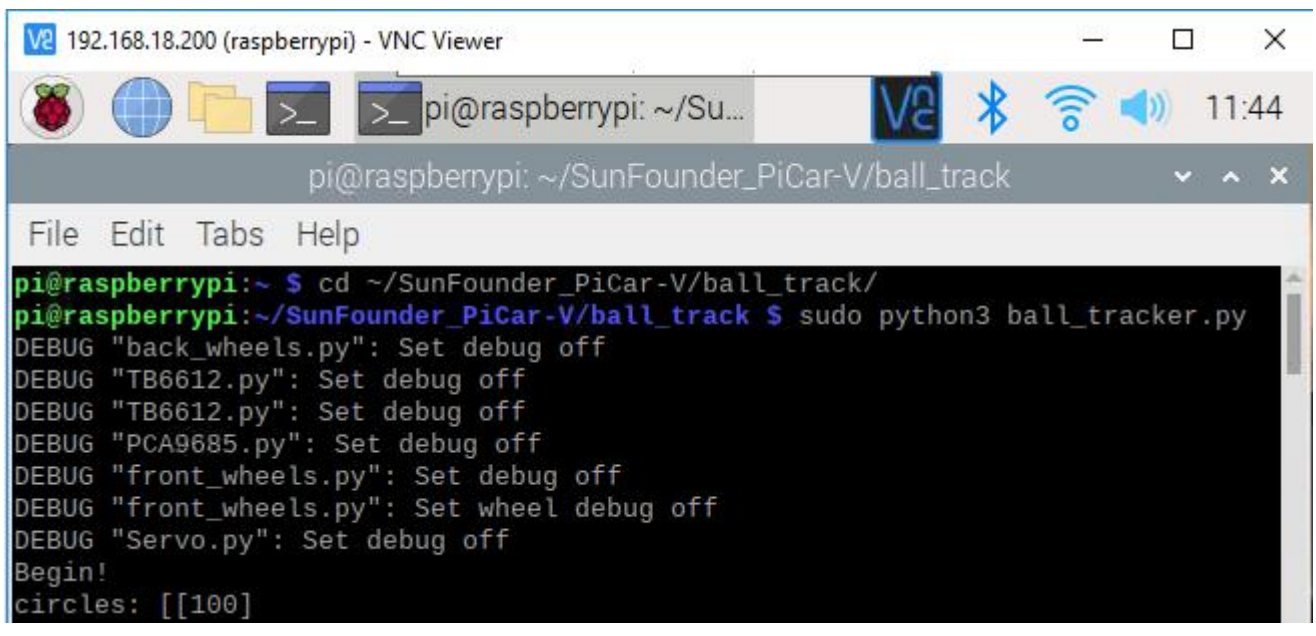


Note: After you finish doing this step, you must log in the Raspberry Pi via VNC, or you can use a monitor; if not, the warning is as follows:

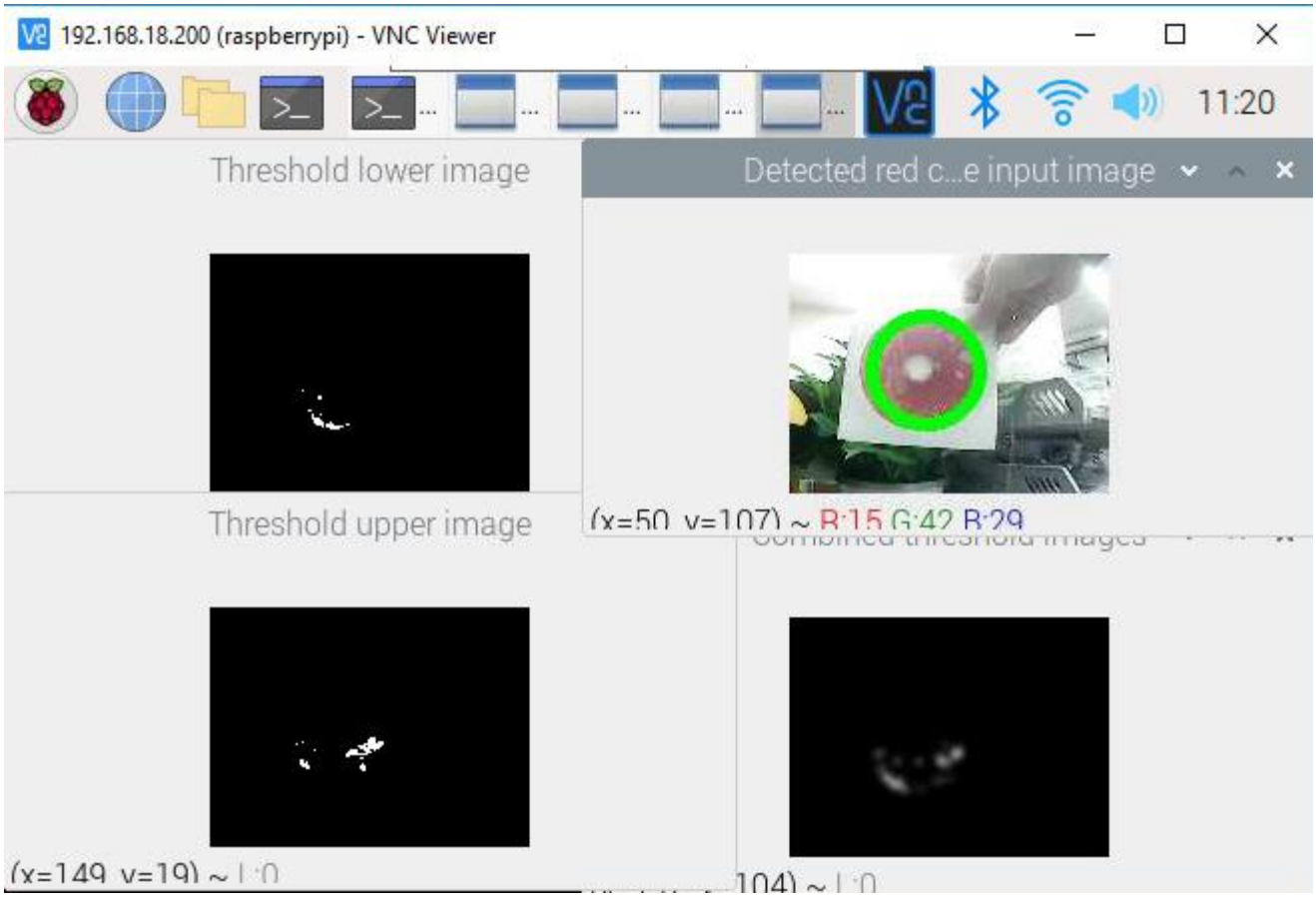
```
pi@raspberrypi:~/SunFounder_PiCar-V/ball_track $ python3 ball_tracker.py
Warning: Display not found, turn off "show_image_enable" and "draw_circle_enable"
```

Step 6: Run the ball_tracker.py.

```
python3 ball_tracker.py
```



Step 7: After running the code ball_tracker.py, these four images will appear. If you place a red ball in front of the camera, you can see a green circle which represents that the car has catched your red ball.

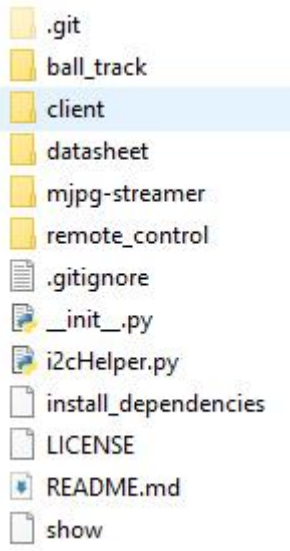


File Analysis

There are two folders under /home/pi: **SunFounder_PiCar** for controlling motors and steering and **SunFounder_PiCar-V** for pan/tilt and wireless control. Here look into the code of PiCar-V.

Since too many contents and instructions are involved in the code, we will not cover every detail. For more knowledge about Python 2, Python 3, PyQt, Django, HTML, and CSS in the programs, you can visit related websites or buy books to learn by yourself. In the following part, we will go through the overall structure and the process in brief.

First, let's check the files in the code folder:



- .git is a hidden directory to store all the information of a Git repository; it's generated automatically after you create the repository.
- Ball_track is used to let the car follow the red ball.
- client is to store the code of the PC client.
- datasheet stores the manual of some chips used in the hardware modules.
- mjpg-streamer is an open source streaming media library, through which the data of the camera is transferred.
- Remote_control is to store the code of the web server; it controls the motor and servo of the car based on the API request.

- .gitignore records the requests of file types to be ignored when the Github repository is synchronized.
- _init_.py is automatically generated when you create a project which is a standard necessary document of python package, just leave it there.
- i2cHelper.py is a Python script written by Python 2 to configure and detect the I2C connection.
- install_dependencies, an executable bash script for simple installation and environment configuration.
- LICENSE, as the name suggests, is a text file of GNU V2 license.
- README.md and show record some information normally for statement and prompts.

Server Code

The server code is based on Django 1.10 (adapt to the latest release if needed). If you are interested in this, you can visit the Django website https://www.djangoproject.com/, or read *The Django Book* to learn more at http://gsl.mit.edu/media/programs/south-africa-summer-2015/materials/djangobook.pdf. Here we will just learn how the web server works.

Note:

The code may be updated irregularly on Github to fix bugs and release some functions update. So the code here is the initial version. You can view the updates in the Github repository at:

https://github.com/sunfounder/SunFounder_PiCar-V.

Open the code directory and check the file by **ls**:

```
pi@raspberry:~ $ cd SunFounder_PiCar-V/
pi@raspberry:~/SunFounder_PiCar-V $ ls
datasheet       install_dependencies      mjpg-streamer   remote_control
client          i2cHelper.py   LICENSE    README.md       show
```

remote_control is the main code directory of the web server. Open the code directory by cd remote_control, and check the file by ls:

```
pi@raspberry:~/SunFounder_PiCar-V $ cd remote_control
pi@raspberry:~/SunFounder_PiCar-V/remote_control $ ls
db.sqlite3  manage.py  remote_control  start  static
```

- db.sqlite3 is generated when you create a Django project.
- manage.py is the main program of the Django project and is generated when the project is created. It normally does not need to be modified.
- remote_control includes the main code files.
- start is a small script written to run sudo python manage.py runserver 0.0.0.0:8000, and servo installation with attribute install just for convenience.
- static is to store some static pictures on the web.

The Django web server normally runs sudo python manage.py runserver to start. The address 0.0.0.0:8000 means the listening address covers all the addresses on the LAN, and the port number is 8000. Here we will just focus on the code in remote_control folder. Go to the directory via cd remote_control:

```
pi@raspberry:~/SunFounder_PiCar-V/remote_control $ cd remote_control

pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control $ ls
driver  __init__.py  settings.py  templates  urls.py  views.py  wsgi.py
```

- driver stores the drivers of the car.
- __init__.py is automatically generated when you create a Django project which is a standard necessary document of python package, just leave it there.
- settings.py is automatically generated and stores the related settings.
- templates is a Django app for storing the webs in the html format.
- urls.py is generated automatically to configure the URL to associate with the code.
- views.py is the code for page control which is associated by the URL. It calls the templates to show the page and the driver to control the car.
- wsgi.py is generated automatically and does not need changes. For more, please visit the official website of Django.

So here is how the code works: Run the main program manage.py which will be associated with urls.py automatically, to respond to the URL. When you run the web browser like the Chrome to visit http://<rpi_ip_address>:<port> or visit the configured API via the client, the manage.py will turn to views.py due to the association of the urls.py. Then the views.py processes this and returns the templates to the browser. In addition, it will call the driver package on the basis of the parameters set in the browser to control the car.

Now open the folder driver and check:

```
pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control $ cd
driver/

pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control/driver $ ls

camera.py   config   __init__.py   stream.py
```

The driver folder mainly includes the driver modules for controlling the pan and tilt and camera streamer.

camera.py for controlling the pan-and-tilt.

config stores the calibration data.

__init__.py is the essential file of the package and you can just leave it alone.

stream.py is a video streaming service based on the MJPG-streamer.

Exit and open the folder templates to view:

```
pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control/driver
$ cd ../
pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control $ cd
templates/
admin.py   __init__.py   models.py   tests.py
apps.py    migrations    templates    views.py
```

This folder is created by the manage.py startapp just for calling the templates conveniently. Therefore, the files have not been changed except for the templates. Open the templates again:

```
pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control/templates
$ cd templates

pi@raspberry:~/SunFounder_PiCar-V/remote_control/remote_control/templates/
templates $ ls

base.html    cali.html    run.html
```

There are three HTML files that also consist of layers. The low level `base.html` stores contents contained on each page such as the `<head>` of HTML, the overall layout, and contents of the home page by default. The surface layer: `cali.html` for calibration and `run.html` to control the car.

So that's the code of the server. Next we will come to the code of the client written by PyQt.

Client Code

In **Windows**, you can check the contents of client directory by grouping files by file type:

- `client.py`, is the main part of the code of the client.
- `icons_rc.py` is the python code converted from the UI resource file of Qt, which is the `icons.qrc` file beside, by `pyrcc5 –o icons_rc.py icons.qrc`.
- `icons.qrc` is the Qt resource file and restores the location of the resource pictures.
- `.ui` is the ui file, or the graphical user interface (GUI) file designed by the Qt tool. Open it by a text editor and you will see the code is written similar to an xml file.
- `images` stores the picture resources used in designing the UI (user interface). Do not rename or move the folder and the files in it, or else the corresponding content on the UI will be missing because it fails to find the graphic resource.

- `_pycache_` not shown in the figure above, is a cache folder automatically generated when you run a `.py` file of the client. If you've just downloaded it and not run yet, it's not generated.

The code of the client is written in Python 3, and the GUI is designed by the Qt tool. The open source library PyQt is used in the client.py to load the .ui file written by Qt into the python code.

For more information about PyQt, please visit the website:

https://riverbankcomputing.com/software/pyqt/intro

The interfaces go one by one like this:

1. The **login** page: enter the Raspberry Pi's IP address and click **Log in** go to the operation interface.

2. The **operation** interface will show the view captured by the camera. You can control the movement of the car on this page. You can click the return button, to go back to login, and the settings button to go to calibration interface.

3. The **calibration** interface: you can control and adjust the car by keyboard. Click **OK** to save the calibration data, and **Cancel** to return to the upper level interface.

Now after going through the code, you may wonder actually how to design the graphical interface and use PyQt. So let's take a closer look!

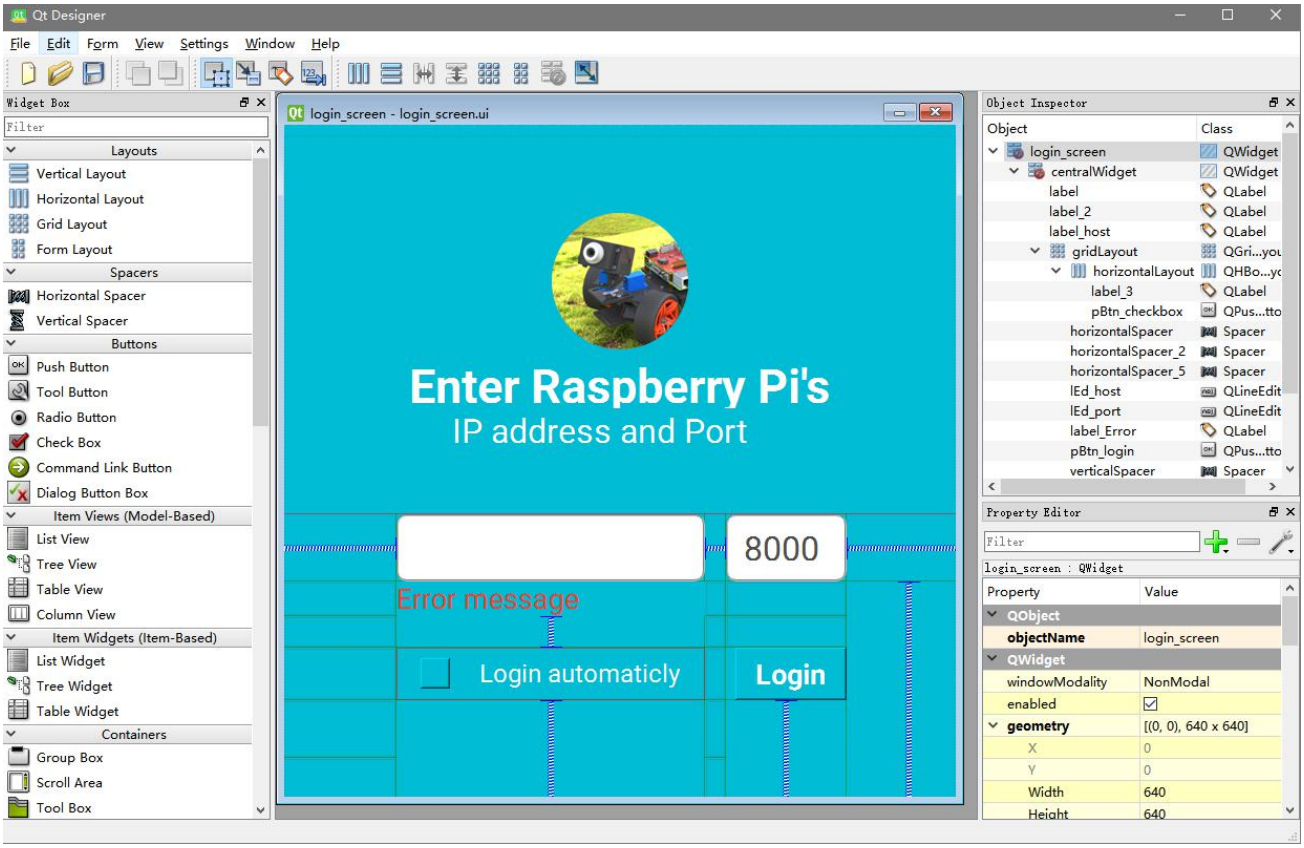
- Designing the Interface

Let's start with the design interface.

In **Windows**, if you install the PyQt by the .exe package, select the option to install all, so the GUI design tools like the Qt Designer will be installed automatically. Or you can install the entire kit of Qt by yourself. Download the package (may need some setting and registration):

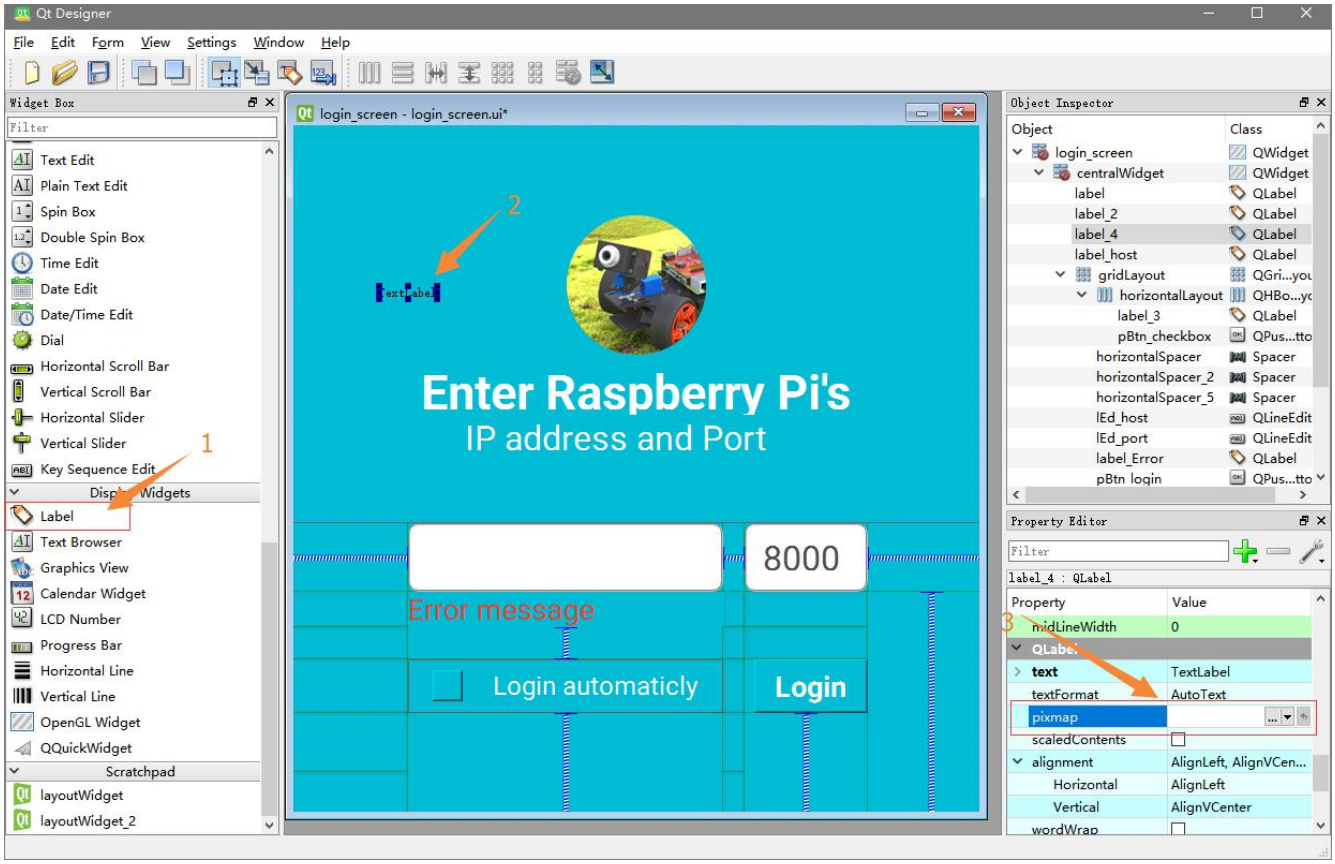
https://www.qt.io/developers/

Open the .ui file with a Qt tool after the installation is completed. The Qt designer in Qt tools is a GUI design tool.

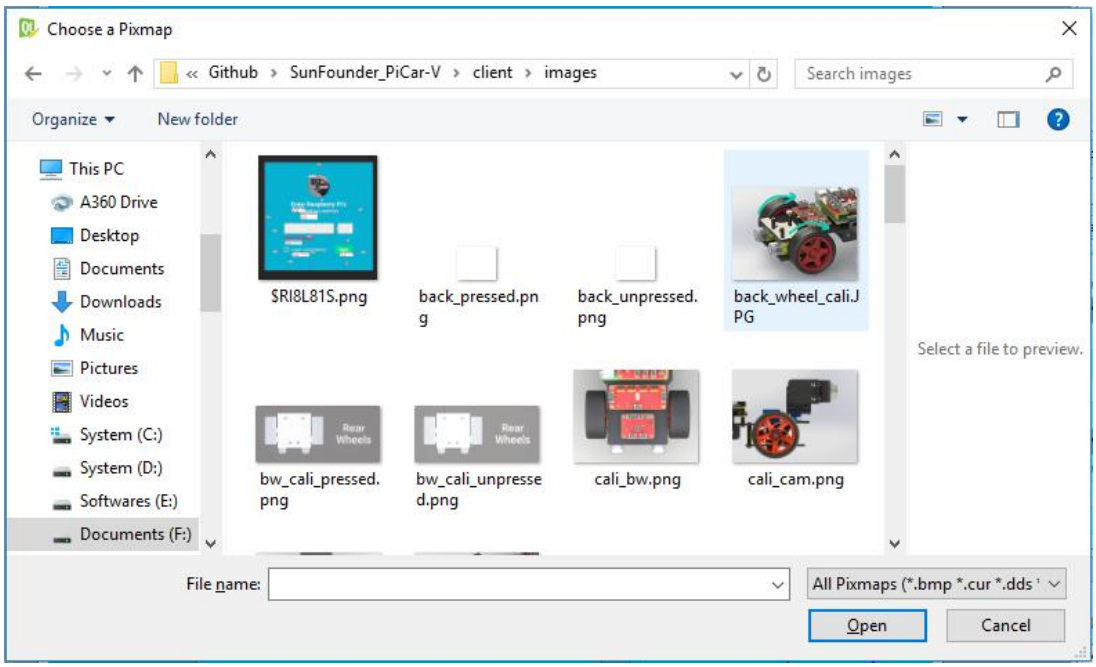


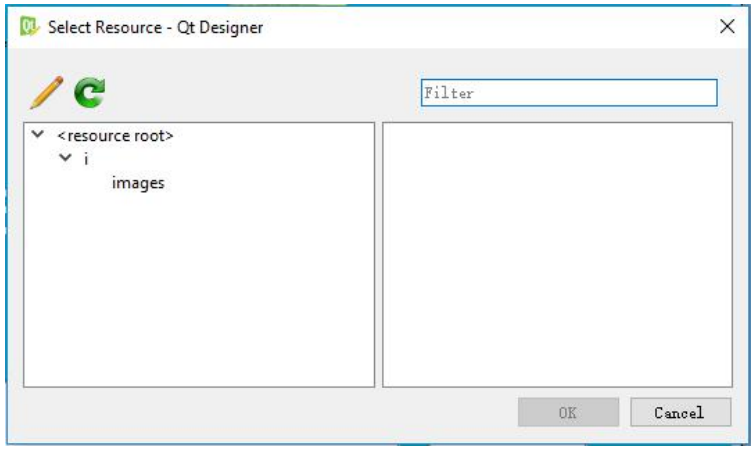
Design the first interface (login), with the GUI design tool. In the left column is the **Widget Box** section. You can just drag the components available here to the edit window in the center to see the effect directly. In the right column, on top is **Object Inspector**, where you can check the association and category of the added objects. Under this section, the property of the object is shown. You can set the attributes in this part, to differentiate all the widget objects.

Let's take the picture of the login interface as an example. Drag a **Label** from **Widget Box** to the picture, then click on the label and there is a square around, by which you can change its size and position. Then in the property part, select **pixmap** under **text** and click the box next to it, and on the pop-up drop-down list, select to read the picture from resource or from file.



If you choose to read from file, find the picture to be used, select it and click **OK**; to read from resource, create the resource file first. It is very simple – by the pencil icon (edit button) in the window of creating from resource. Click the icon, and select to create a resource and load it.





The setting of other components such as button, line edit and on, is just similar, though the properties are different. The name of the objects whose property has been changed will be marked bold automatically for convenience in next check. You can open those .ui files to see the property details that can be changed, or the widget type by objects in **Object Inspector**. After editing, you can select **Window** -> **Preview** in the Qt Designer to see the effect, or by the shortcut **Ctrl** + **R**.

Don't need to worry if you are not familiar with Qt because great details are given in the Qt document. If you don't know which property is for what use, you can check **Help** or the Qt assistant. And those sample sketches of Qt will help a lot for you to learn about Qt.

- Programing the Functions

Now the interface design is finished. However, the interface files are just some static images and cannot be put into use. So the next step, write the client.py to implement all the interfaces with the corresponding functions as well as redirections. The PyQt is like this:

```
1
2  import sys
3  from PyQt5 import QtCore, QtWidgets, uic
4
5  qtCreatorFile = "" # Enter file here.
6
7  Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
8
9  class MyApp(QtGui.QMainWindow, Ui_MainWindow):
10     def __init__(self):
11         QtGui.QMainWindow.__init__(self)
12         Ui_MainWindow.__init__(self)
13         self.setupUi(self)
14
15 if __name__ == "__main__":
16     app = QtGui.QApplication(sys.argv)
17     window = MyApp()
18     window.show()
19     sys.exit(app.exec_())
```

In Line 5, enter the name of the ui file you designed in the quotation marks.

In Lin 7e, "uic" is the contents in the previously imported PyQt; the built-in function loads the ui files.

In Line 9-13, the class defined is just your GUI window. Define (*def*) the corresponding functions in the class and write the functions in them. The first *def* __init __ (self) is the initialization function which is a constructor in C++. It means when the instance is created, the program will run the code and initialize the instance.

Line 15-19 is the entry of the __main__ function, where it starts to execute the whole program.

So here's how the program goes:

1. Run the __main__ function and create an instance *app* of QApplication, which is the GUI application.

2. Create an instance *window* of the MyApp class type. After the creation is done, run the __init__ initialization function of the class.

3. The window instance is displayed.

4. Wait for a while. If the instance *app* of the GUI application sends an exit signal *app.exec_()*, then enter sys.exit() to exit Python.

So that's all for the code. Here we've only looked at the entry-level usage. For more detailed code explanation, go to the *client.py* file. There you can see very detailed comments.

Advanced

Now the car is running! You may try control it or view the image captured in Windows, Linux, or Mac OS X.

So what to do next?

Yes, sensors! There are 8 digital and 4 analog channels on the Robot HATS, as well as two I2C ports. Plus the 3mm slot on the acrylic plate in the car head, it enables the connection of various sensors, making the car smarter and more intriguing as you want!

And, coding! If you know programming and are capable of compiling code, you may change the code provided for the kit to make the car more fabulous. For instance, you can find the code that controls the front and back wheels in the folder SunFounder_PiCar, downloaded to the directory /home/pi if you've installed the program by install-dependences.

You may check what's in the file:

```
cd ~/SunFounder_PiCar
ls
```

After you've changed the code, you can go to the page of the Github repository for the car, and fork it to help improve it! Also you're quite welcome to post issues and request a pull on our Github page:

PiCar-V: https://github.com/sunfounder/SunFouner_PiCar-V

PiCar driver:

https://github.com/sunfounder/SunFounder_PiCar

And of course, if you have any questions or ideas, please feel free to email us at service@sunfounder.com.

Appendix 1: Function of the Server Installation Scripts

You may have a question: What do the installation scripts do when we install the server on the Raspberry Pi by them? So here let's check the detailed steps.

1. Install pip.

```
apt-get install python-pip
```

2. Use pip to install django.

```
sudo pip install django
```

3. Install i2c-tools and python-smbus.

```
sudo apt-get install i2c-tools python-smbus –y
```

4. Install PiCar drivers.

```
cd ~/
git clone -–recursive https://github.com/sunfounder/SunFounder_PiCar.git
cd SunFounder_PiCar
sudo python setup.py install
```

5. Download source code.

```
cd ~/
git clone https://github.com/sunfounder/SunFounder_PiCar-V.git
```

6. Copy the MJPG-Streamer file to system directory.

```
cd ~/SunFounder_PiCar-V
sudo cp mjpg-streamer/mjpg_streamer /usr/local/bin
sudo cp mjpg-streamer/output_http.so /usr/local/lib/
sudo cp mjpg-streamer/input_file.so /usr/local/lib/
sudo cp mjpg-streamer/input_uvc.so /usr/local/lib/
sudo cp -R mjpg-streamer/www /usr/local/www
```

7. Export paths.

```
export LD_LIBRARY_PATH=/usr/local/lib/ >> ~/.bashrc
export LD_LIBRARY_PATH=/usr/local/lib/ >> ~/.profile
source ~/.bashrc
```

8. Enable I2C1.

Edit the file /boot/config.txt:

```
sudo nano /boot/config.txt
```

Add the line in the end:

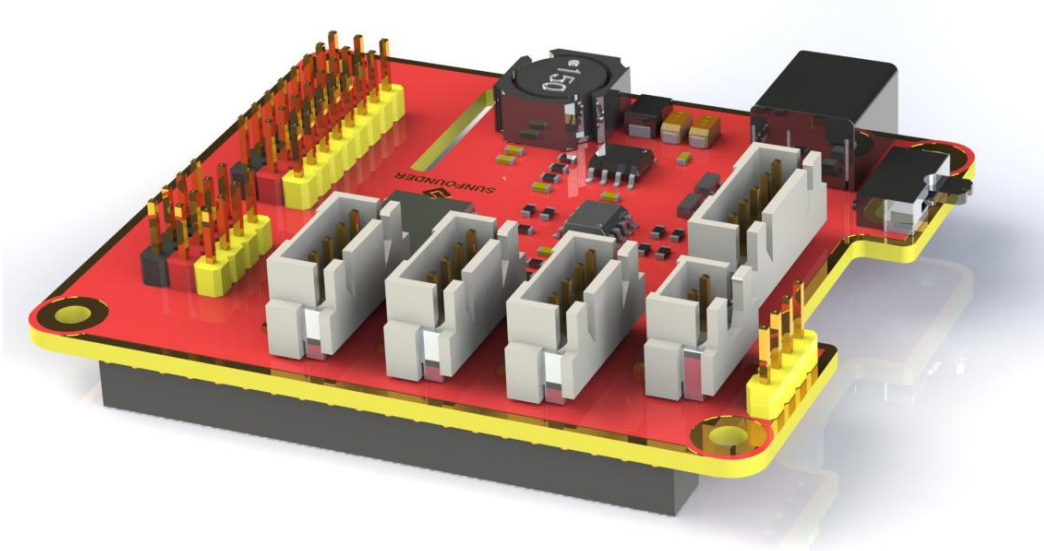
```
dtparam=i2c_arm=ons
```

9. Reboot.

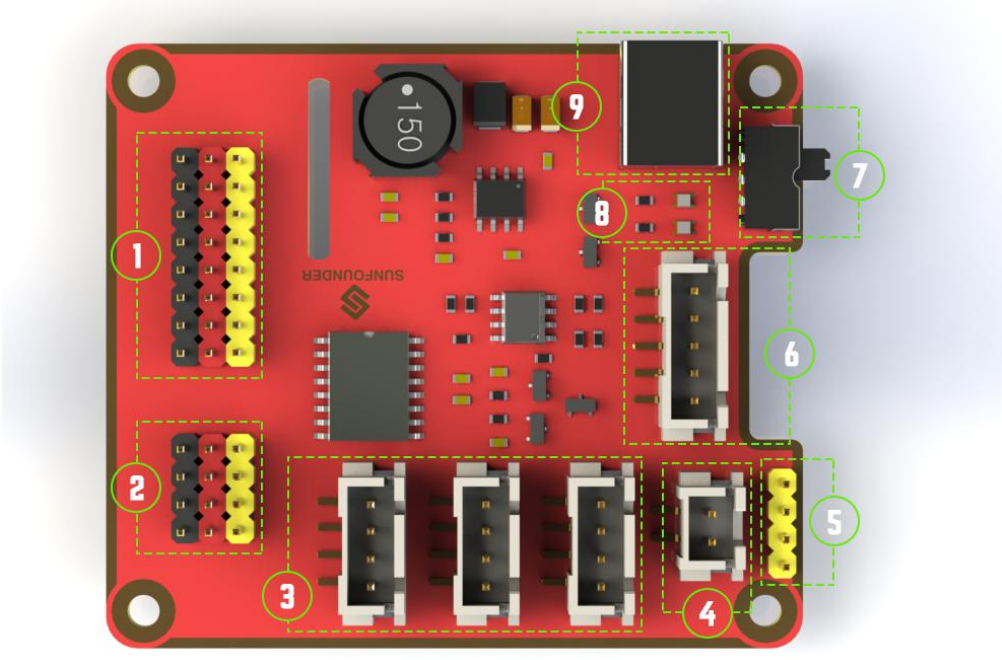
```
sudo reboot
```

Appendix 2: Components

Robot HATS



Robot HATS is a specially-designed HAT for a 40-pin Raspberry Pi and can work with Raspberry Pi model B+, 2 model B, 3 model B, 3 model B+, and 4 model B. It supplies power to the Raspberry Pi from the GPIO ports. Thanks to the design of the ideal diode based on the rules of HATS, it can supply the Raspberry Pi via both the USB cable and the DC port thus protecting it from damaging the TF card caused by batteries running out of power. The PCF8591 is used as the ADC chip, with I2C communication, and the address 0x48.



1. **Digital ports**: 3-wire digital sensor ports, signal voltage: 3.3V, VCC voltage: 3.3V.

2. **Analog ports**: 3-wire 4-channel 8-bit ADC sensor port, reference voltage: 3.3V, VCC voltage: 3.3V.

3. **I2C ports**: 3.3V I2C bus ports

4. **5V power output**: 5V power output to PWM driver.

5. **UART port**: 4-wire UART port, 5V VCC, perfectly working with SunFounder FTDI Serial to USB.

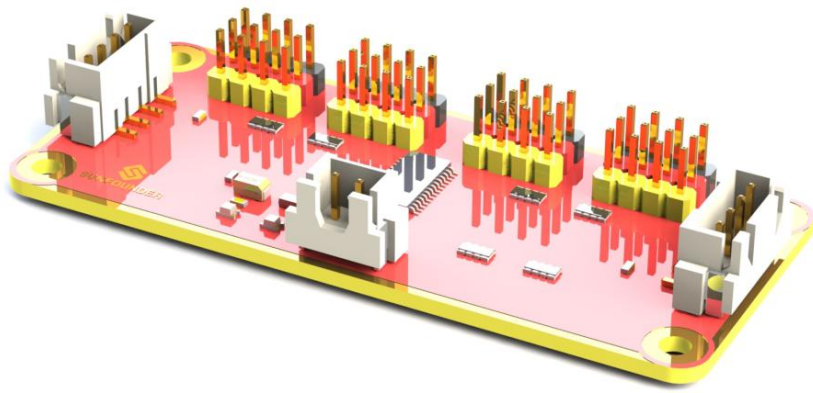
6. **Motor control ports**: 5V for motors, direction control of motors MA and MB and a floating pin NC ; working with SunFounder motor driver module.

7. **Switch**: power switch

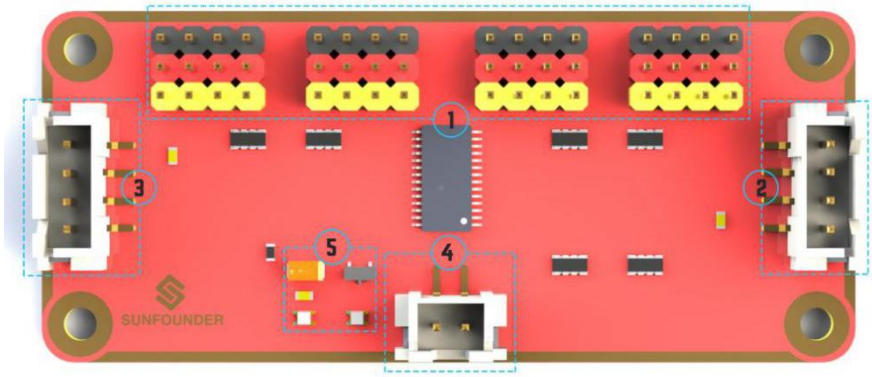
8. **Power indicators**: indicating the voltage – 2 indicators on: >7.9V; 1 indicator on: 7.9V~7.4V; no indicator on: <7.4V. To protect the batteries, you're recommended to take them out for charge when there is no indicator on. The power indicators depend on the voltage measured by the simple comparator circuit; the detected voltage may be lower than normal depending on loads, so it is just for reference.

9. **Power port**: 5.5/2.1mm standard DC port, input voltage: 8.4~7.4V (limited operating voltage: 12V~6V).

PCA9865



PCA9685 16-channel 12-bit I2C Bus PWM driver. It supports independent PWM output power and is easy to use 4-wire I2C port for connection in parallel, distinguished 3-color ports for PWM output.



1. **PWM output ports**: 3-color ports, independent power PWM output port, connect to the servo directly.

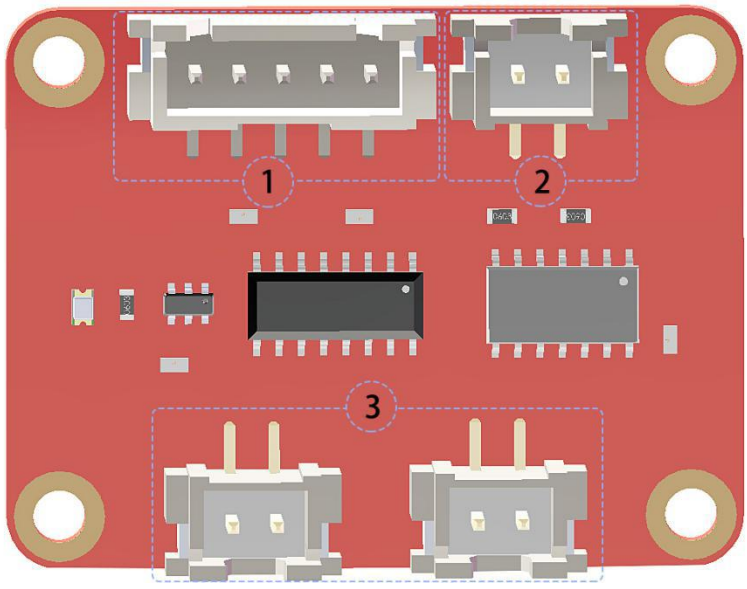
2 & 3. **I2C port**: 4-wire I2C port, can be used in parallel. Compatible with 3.3V/5.5V

4. **PWM power input**: 12V max.

5. **LED**: power indicator for the chip and for the PWM power input.

Motor Driver Module

The motor driver module is a low heat generation one and small packaged motor drive.



1. **Power and motor control port**: includes pins for supplying the chip and the motors and controlling the motors' direction.
2. **PWM input for the motors**: PWM signal input for adjusting the speed of the two motors.
3. **Motor output port**: output port for two motors.

USB Webcam

This camera supports a wide angle of 120°, which provides a wide and clear vision, thus giving better experience when you're using it on the PiCar-V.

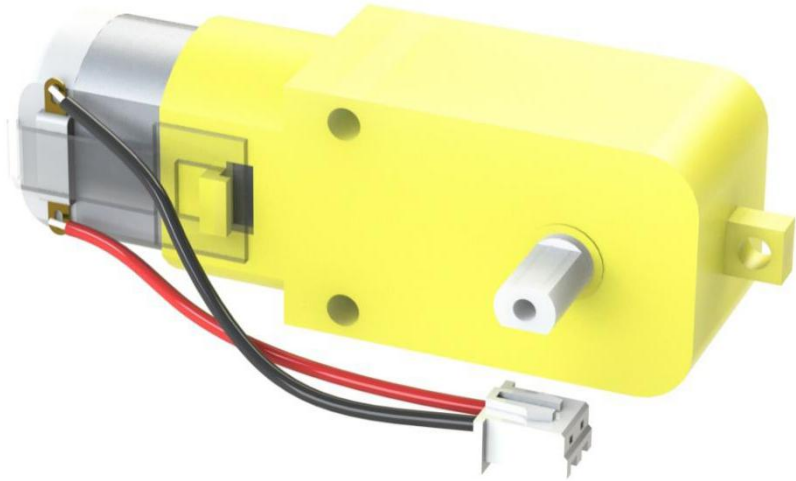
SunFounder SF006C Servo

Clutch gear digital servo with a DC core motor inside, After a certain load, the steering gear reducer will automatically clutch and protect the product from damage and normal load.

Function of the Performance:

Item	V = 4.8V	V = 6.0V
Consumption Current* (No Load)	≦50mA	≦60mA
Stall Current	≦550mA	≦650mA
Rated Torque	≥0.6 kgf·cm	≥0.7 kgf·cm
Max. Torque	≥1.4 kgf.cm	≥1.6 kgf.cm
No Load Speed	≦0.14sec/60°	≦0.12sec/60°

DC Gear Motor



It's a DC motor with a speed reducing gear train. See the parameters below:

Motor	Model	F130SA-11200-38V
	Rated Voltage	4.5V-6V
	No-load Current	≤80mA
	No-load Speed	10000±10%
Gear Reducer	Gear Ratio	1:48
	Speed (no-load)	≈200rpm (≈180rmp in test)
	Current	≤120mA

Copyright Notice