

---

# **SunFounder Zeus Robot Car Kit for Arduino**

***Release 1.0***

**[www.sunfounder.com](http://www.sunfounder.com)**

**Feb 18, 2024**





# CONTENTS

<b>1</b>	<b>Get Started</b>	<b>3</b>
1.1	Assemble the Car . . . . .	3
1.2	Play Mode . . . . .	4
1.3	Programming Mode . . . . .	28
<b>2</b>	<b>Hardware</b>	<b>111</b>
2.1	SunFounder R3 Board . . . . .	111
2.2	Zeus Car Shield . . . . .	113
2.3	ESP32 CAM . . . . .	124
2.4	Camera Adapter Board . . . . .	126
2.5	Omni Grayscale Module . . . . .	132
2.6	Ultrasonic Module . . . . .	138
2.7	IR Obstacle Avoidance Module . . . . .	140
2.8	4 RGB LEDs Strip . . . . .	143
2.9	18650 Battery . . . . .	144
2.10	TT Motor . . . . .	145
2.11	Mecanum Wheel . . . . .	146
<b>3</b>	<b>FAQ</b>	<b>149</b>
3.1	Q1: Compilation error: SoftPWM.h: No such file or directory? . . . . .	149
3.2	Q2: avrdude: stk500_getsync() attempt 10 of 10: not in sync: resp=0x6e? . . . . .	149
3.3	Q3: How can I use the STT feature on my Android device? . . . . .	150



Thanks for choosing our Zeus Car.

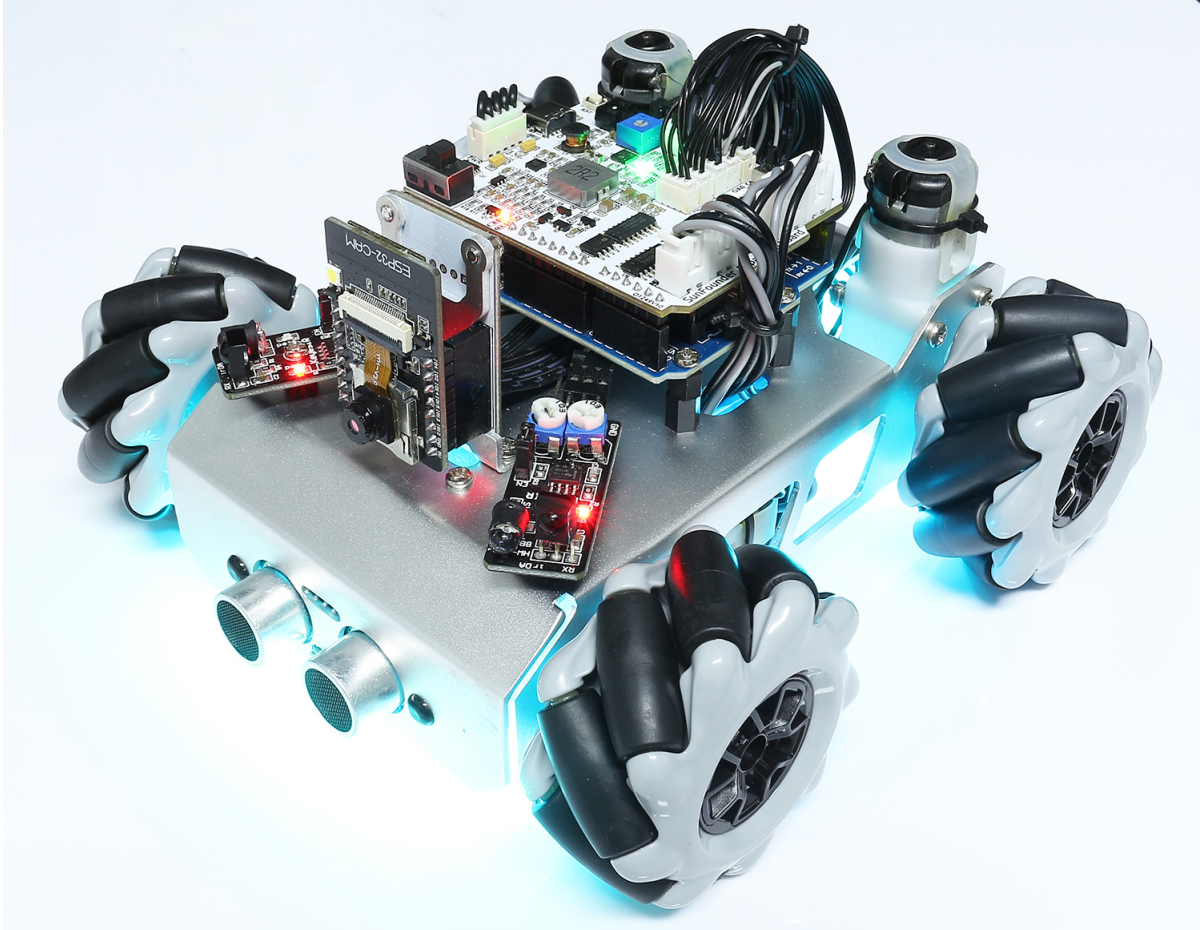
---

**Note:** This document is available in the following languages.

- 
- 
- 

Please click on the respective links to access the document in your preferred language.

---



This is an educational kit for beginning programmers (children) to gain hands-on experience in electronics, robotics and programming.

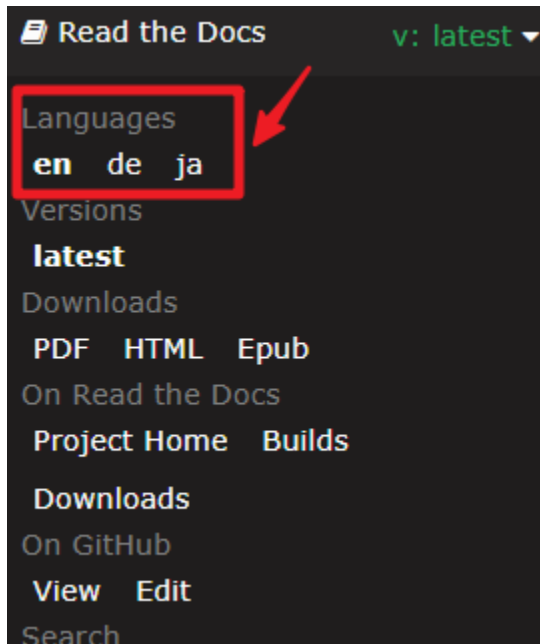
- Using 6cm Mecanum Wheel, easy to achieve 360 ° omnidirectional movement, and can perform difficult movements such as left and right drift.
- Equipped with ESP32 CAM, you can experience first-person driving on the APP.
- With 2 obstacle avoidance modules and an ultrasonic module, it can realize obstacle avoidance and following function.
- The bottom is mounted with an 8-channel circularly placed grayscale module, which allows the car to follow the line in any direction.
- There are also IR remote control and cool lighting effects, etc.

- Module interfaces are XH1.5, easy to assemble and less chance of error.

We set a *Play Mode* for it, so users can directly turn on the car after assembling it and play with its various cool functions with its own remote control or mobile APP. In case you want to learn more about how to implement its various functions, you can refer to the *Programming Mode*.

### About the display language

This document is available in other languages as well. To switch the display language, kindly click on the Read the Docs icon located in the lower left corner of the page.



If you have any questions, please send an email to [service@sunfounder.com](mailto:service@sunfounder.com) and we will respond as soon as possible.

## GET STARTED

We have set two modes for Zeus Car, the hands-on play mode and the programming mode.

- Hands-on *Play Mode* means you just need to turn on the Zeus Car after assembling, and then you can directly control this car with the remote control or APP to make it achieve various cool functions. Because we have pre-uploaded the code to the Arduino board at the factory.
- The *Programming Mode* is that after you have experienced the various cool functions of Zeus Car, you can understand the principle of implementing each function individually according to each project, and then modify it according to your own ideas to achieve the functions you want.

### 1.1 Assemble the Car

Here is the assembly instructions for the Zeus Car, PDF version, which is the same as the foldout you have. If you think there is something on the foldout that you can't read, you can download this PDF.

(PDF)Zeus Car Assemble Instruction

#### Assembly Tutorial Video

This video will walk you through the process of assembling your robot from scratch.

In this tutorial, you will learn:

- **Preparation:** We'll introduce you to all the tools and parts needed, ensuring you're fully equipped before starting the assembly.
- **Assembly Steps:** We'll demonstrate each assembly step in a systematic manner.
- **Tips and Considerations:** Throughout the process, we'll share essential tips and tricks to help you avoid common mistakes and ensure your car operates smoothly.
- **Testing and Calibration:** Once assembly is complete, we'll walk you through a brief test of this car to ensure that your assembly is free of issues.

## 1.2 Play Mode

The play mode is where we pre-upload the code to the Arduino board at the factory, and the Zeus Car can be remotely controlled by APP or remote control once it is turned on.

- *Controlled by APP*: You need to install the SunFounder Controller on your mobile or tablet, which allows you to control the Zeus Car in all directions with a joystick widget, as well as speech control, and most importantly, you can see the real-time video streaming taken by the Zeus Car on the APP.
- *Control by Remote*: We have equipped a simple remote control, just turn on the Zeus Car and you can control it by remote control, which is a good choice for users who want to experience the fun of controlling Zeus Car as soon as possible.

---

**Note:** Here is the code package for this kit.

- SunFounder Zeus Car Kit for Arduino

If you wish to return to **Play Mode** after uploading different code to your Arduino, you first need to *Install the Required Libraries*, then upload the Zeus\_Car.ino file found under the path zeus-car-main\Zeus\_Car to the Arduino board.

---

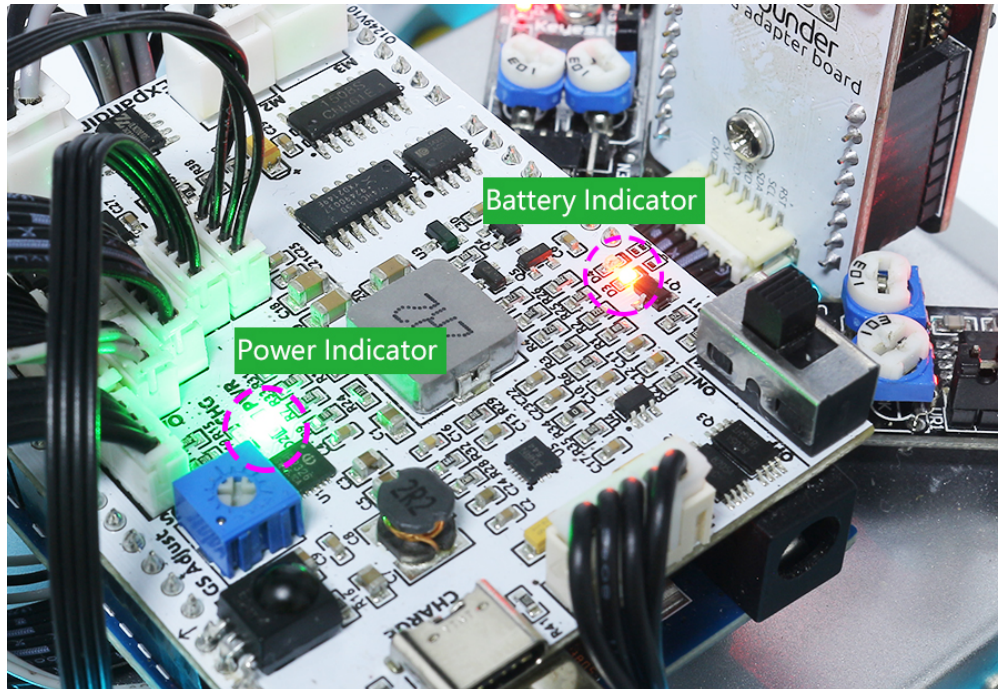
### 1.2.1 Controlled by APP

To control Zeus Car from your mobile device, you must download the APP (SunFounder Controller) on your device, connect to the Zeus Car LAN, and then create your own controller on the APP.

#### Quick Guide

This is a quick tutorial video. Please watch the video first, and then follow the instructions provided below.

1. Let's start the Zeus Car.
  - When first used or when the battery cable is unplugged, Zeus Car Shield will activate its over-discharge protection circuitry.
  - So you'll need to plug in the Type-C cable for about 5 seconds.
  - If the power indicator lights up, it means that the protection status has been released. At this time look at the battery indicators, if both battery indicators are off, please continue to plug in the Type-C cable to charge the battery.



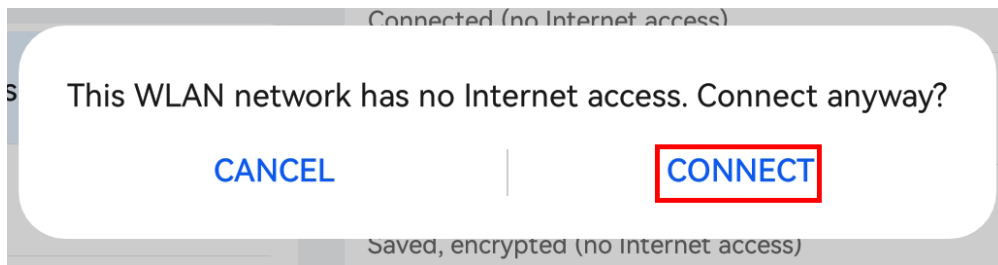
- Now, you'll need to flick the smaller switch to the right to establish communication between the car and the ESP32 CAM. Afterward, press the Reset button to reboot the code. At this point, you'll observe the undercarriage lights transition from orange to a light blue.

2. Install [SunFounder Controller](#) from **APP Store(iOS)** or **Google Play(Android)**.

3. Connect to Zeus\_Car WLAN.

Now, connect your mobile device to the local area network (LAN) broadcast by the Zeus Car. This way, your mobile device and the Zeus Car will be on the same network, which will facilitate communication between the applications on your mobile device and the Zeus Car.

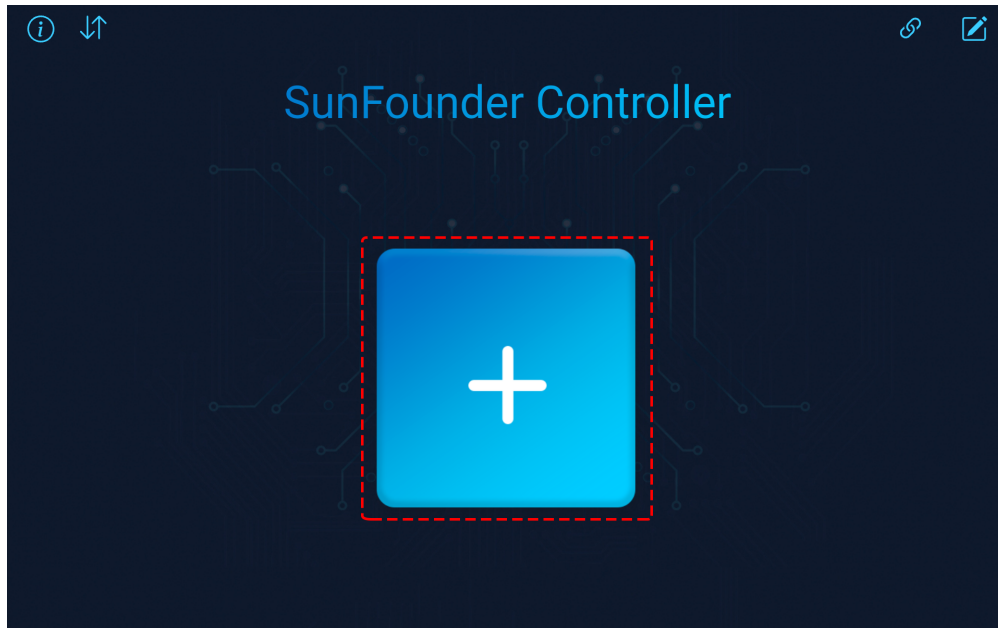
- Find Zeus\_Car on the WLAN of the mobile phone (tablet), enter the password 12345678 and connect to it.
- The default connection mode is AP mode. So after you connect, there will be a prompt telling you that there is no Internet access on this WLAN network, please choose to continue connecting.



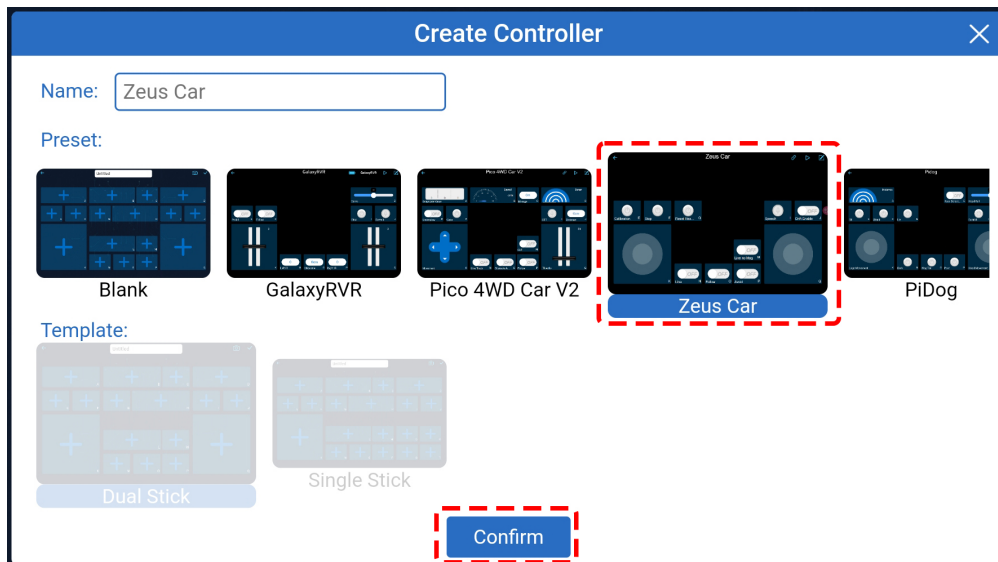
4. Create a controller.

- To add a controller on SunFounder Controller, click the + icon.



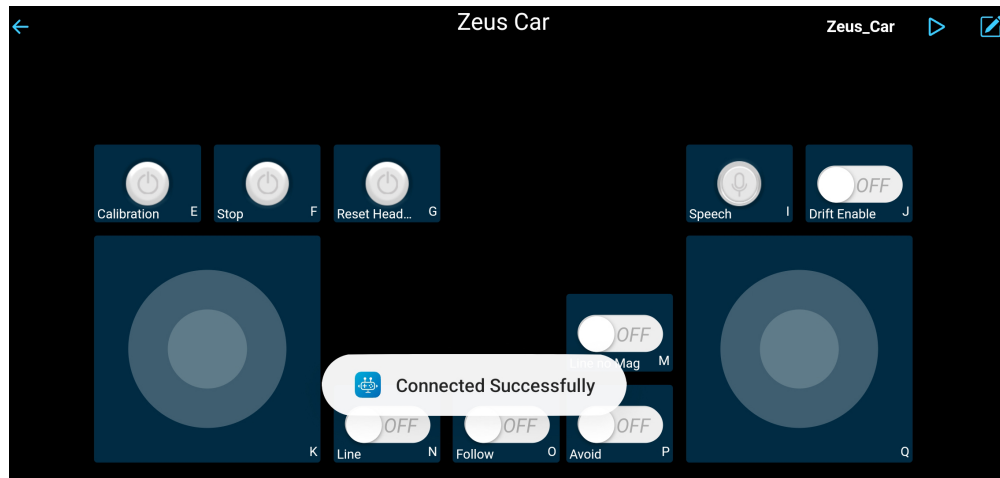



- Preset controllers are available for some products, here we choose **Zeus Car**. Give it a name, or simply tap **Confirm**.

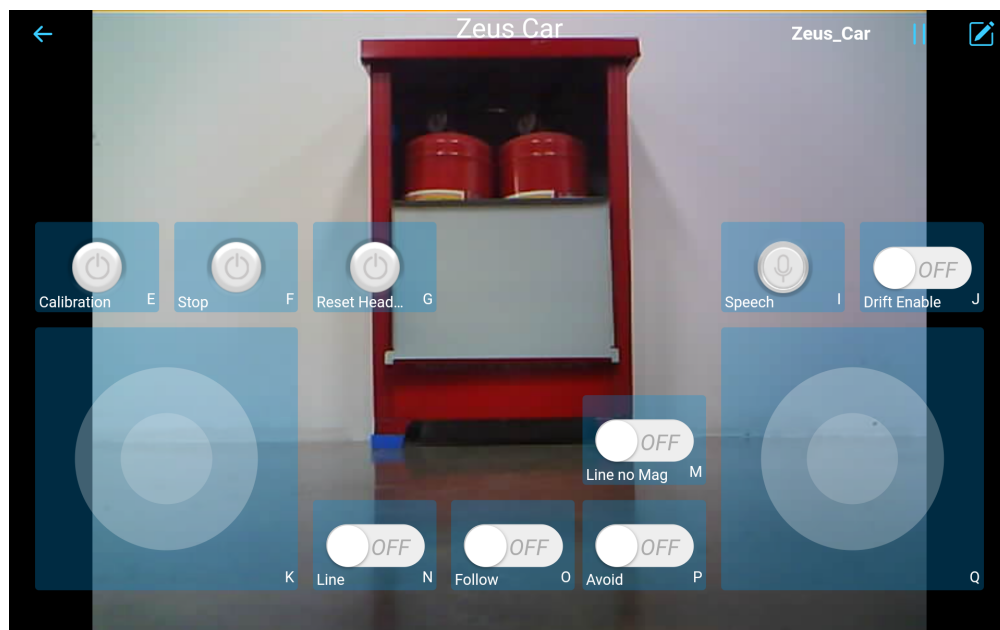


- Once inside, the app will automatically search for the Zeus Car. After a moment, you will see a prompt saying "Connected Successfully."





- Now, tap the  button enables you to view the live video feed from the camera and control the car using the provided widgets.

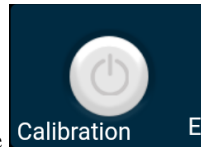


5. Here are the functions of the widgets.

- **Calibration(E):** Turn on compass calibration.
- **Stop(F):** Stop all movements of the car.
- **Reset Heading(G):** After placing the car in one direction with your hand, click on this widget to make this direction as the front of the car movement. This allows you to quickly specify a direction instead of slowly rotating the car to that direction with other widgets.
- **Speech(I):** Switching to speech control mode.
- **Drift Enable(J):** Activate the drift function.
- **Move in All Directions(K):** Control the car to move in all directions.
- **Line Track:** The following two widgets can both switch to line track mode.

- **Line no Mag(M)**: Switch to line track mode, but not affected by the magnetic field. During the line tracking process, the Zeus Car's orientation will continuously change.
- **Line(N)**: Switching to line track mode, due to the presence of the magnetic field, the Zeus Car's orientation during line tracking will be oriented towards a specific direction.
- *Follow(O)*: Switching to follow mode.
- *Avoid(P)*: Switch to obstacle avoidance mode.
- *Control the Direction(Q)*: Used to control the head direction.

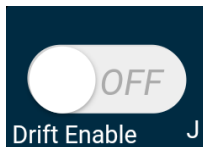
### Calibration(E)



Turn on compass calibration by clicking the Calibration button.

Place the Zeus car on the ground. Upon turning on the compass calibration, the car will start rotating counterclockwise and will stop in about 1 minute. If it rotates longer than 2 minutes, the magnetic field here is complicated. Try changing the location and calibrating again.

### Drift Enable(J)



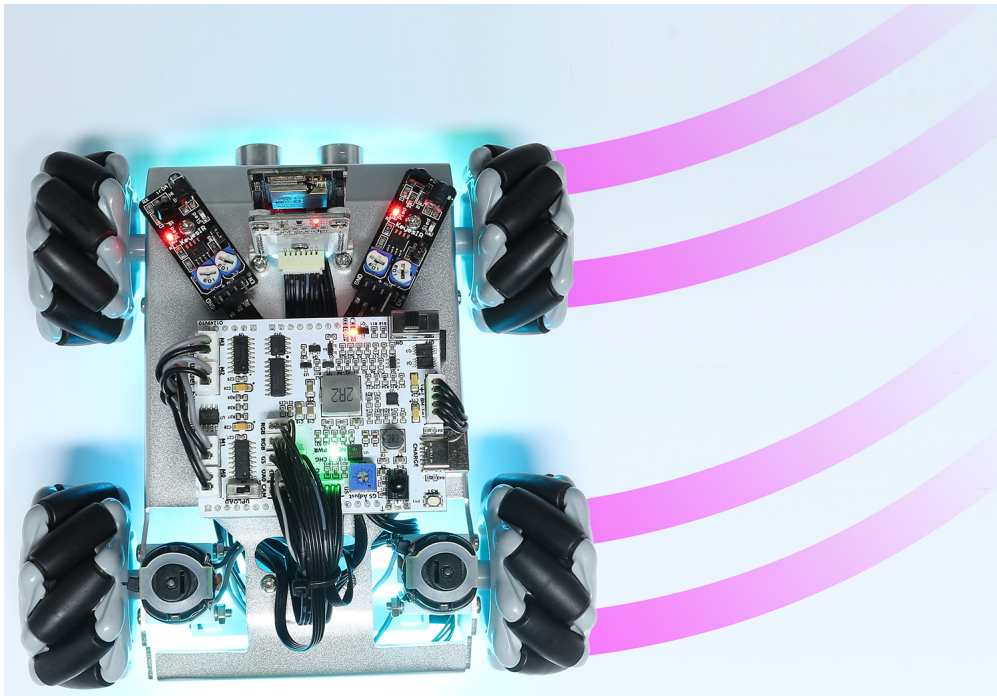
Click the Drift Enable button to enable the drift function.



- By sliding the knob counterclockwise, you will see Zeus Car drift to the right. Upon releasing the hand, the car will stop at its current position.



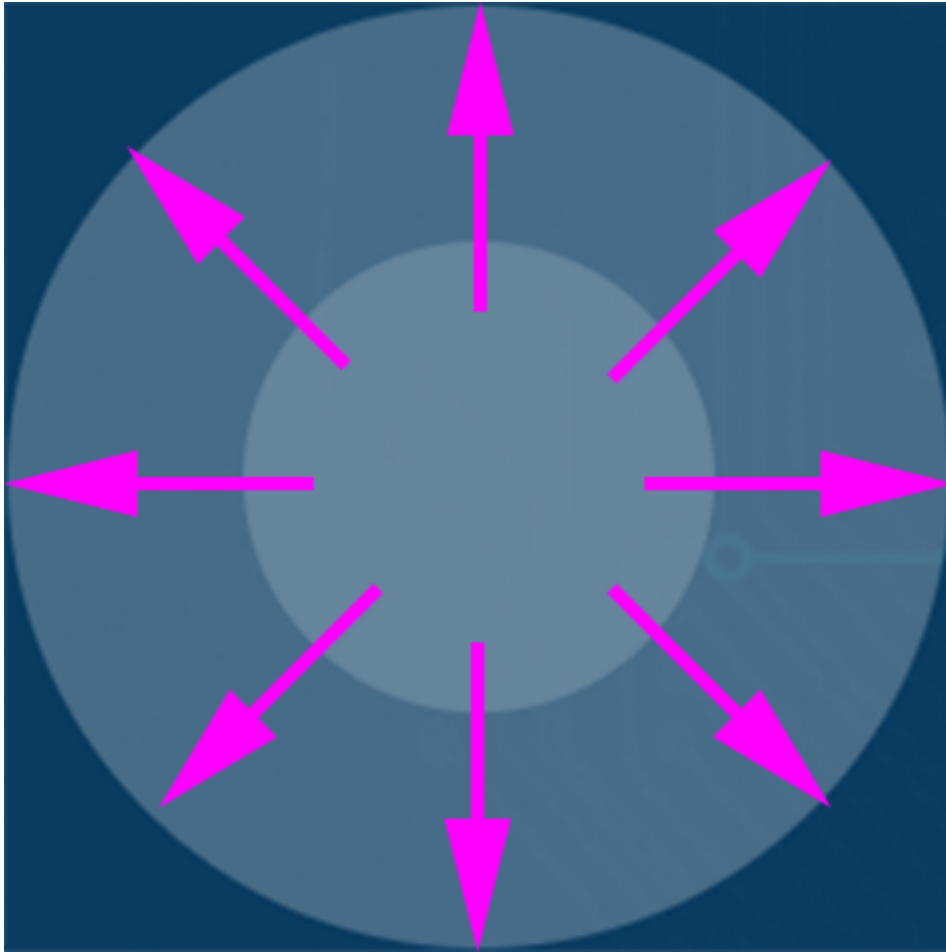
- Similarly, if you slide the **Q** widget clockwise, the Zeus Car will drift to the left and stop in the current position.



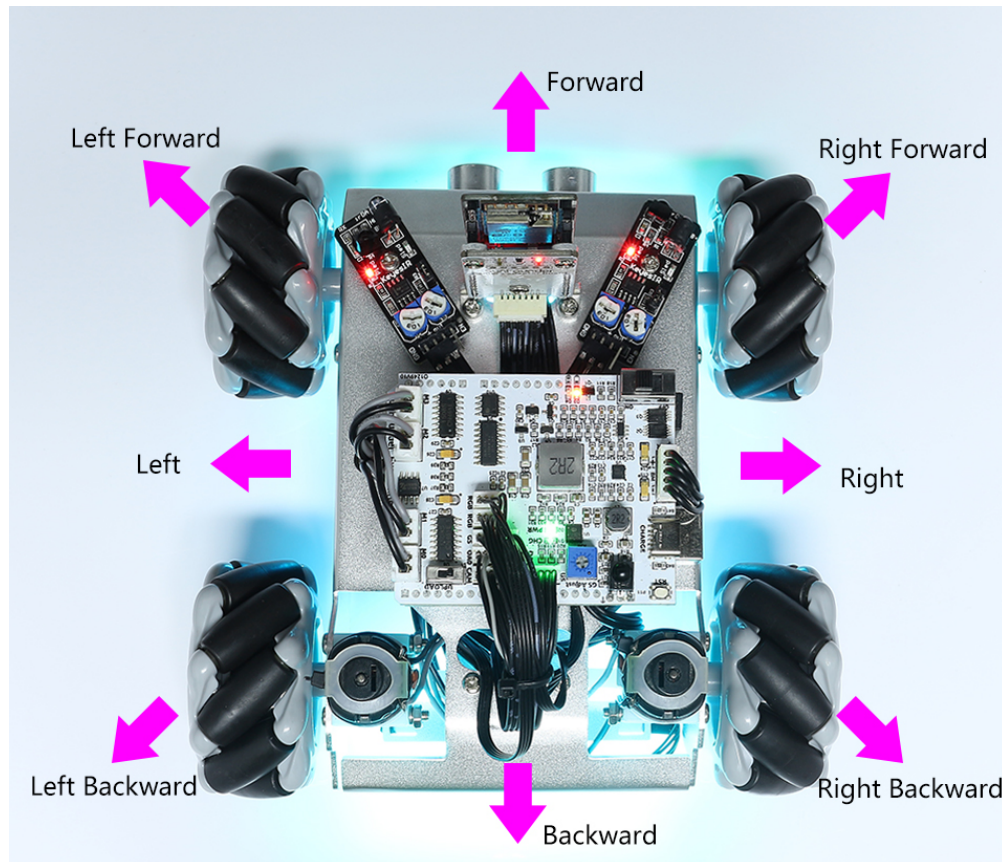
## Move in All Directions(K)



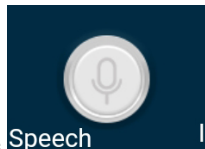
The Zeus Car will move in the appropriate direction when you swipe the **K** widget.



The car moves once every time you slide, so if you don't release your hand all the time, the car keeps moving.



## Speech(I)



By pressing the **Speech(I)** widget, you can activate the STT feature, where STT stands for Speech to Text.

The SunFounder Controller app integrates with your mobile device's voice recognition engine. Hence, when you tap and hold the **Speech(I)** widget on the SunFounder Controller and speak into your mobile device.

Your device will capture your speech, convert it into text, and send it to the Zeus Car. If this text matches the pre-set commands in your code, the Car will carry out the corresponding actions.

The following are the commands currently preset in the code. Speak any of the following commands and observe how the Zeus Car responds.

- stop: All movements of the car can be stopped.
- pasue: The function is basically the same as Stop, but if the head of the car is not facing the direction originally set, it will slowly move to the set direction.
- forward
- backward
- left forward
- left backward



- right forward
- right backward
- move left
- move right

---

**Note:** The STT (Speech to Text) function requires an internet connection and Google services on Android devices. However, this doesn't work with the pre-set AP (Access Point) mode on the Zeus Car.

In AP mode, the Zeus Car creates a local Wi-Fi network that your mobile device can connect to, but it does not provide internet access.

To use the STT function on Android, switch the car's code from AP to STA mode as outlined in [Q3: How can I use the STT feature on my Android device?](#).

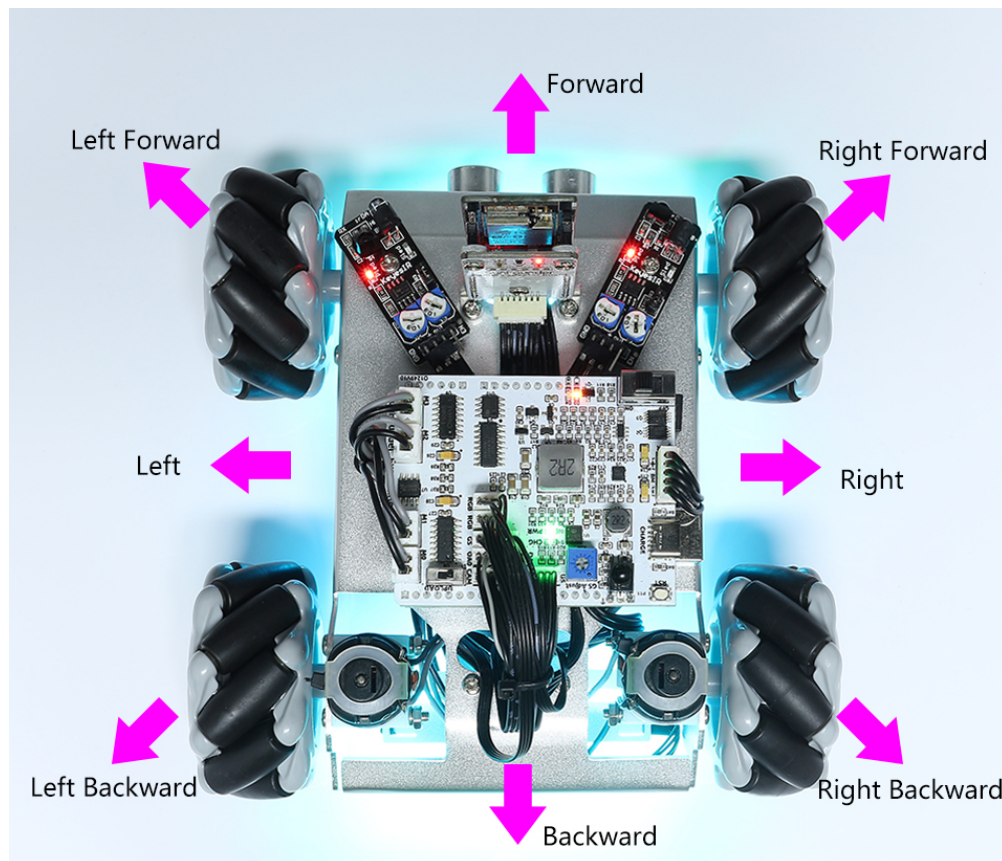
---

---

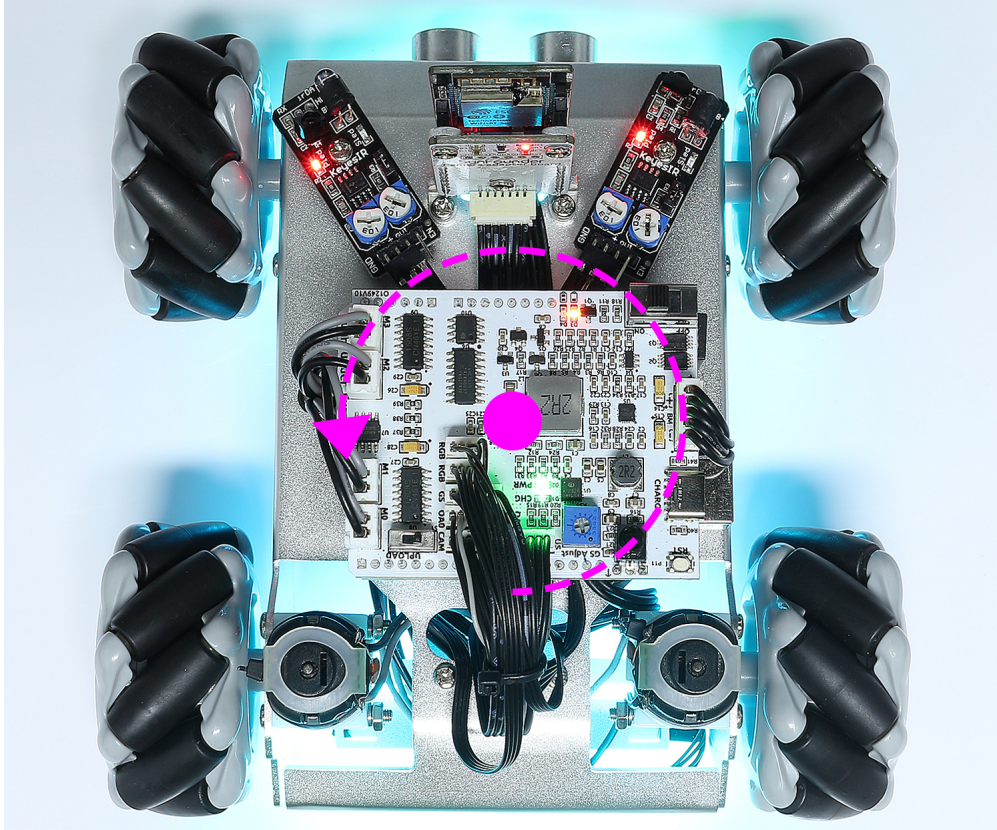
**Note:** iOS devices, using an offline voice recognition engine, work fine in both AP and STA modes.

---

After the car receives the above 8 commands, it will keep moving in the corresponding direction unless it receives stop or pasue commands.

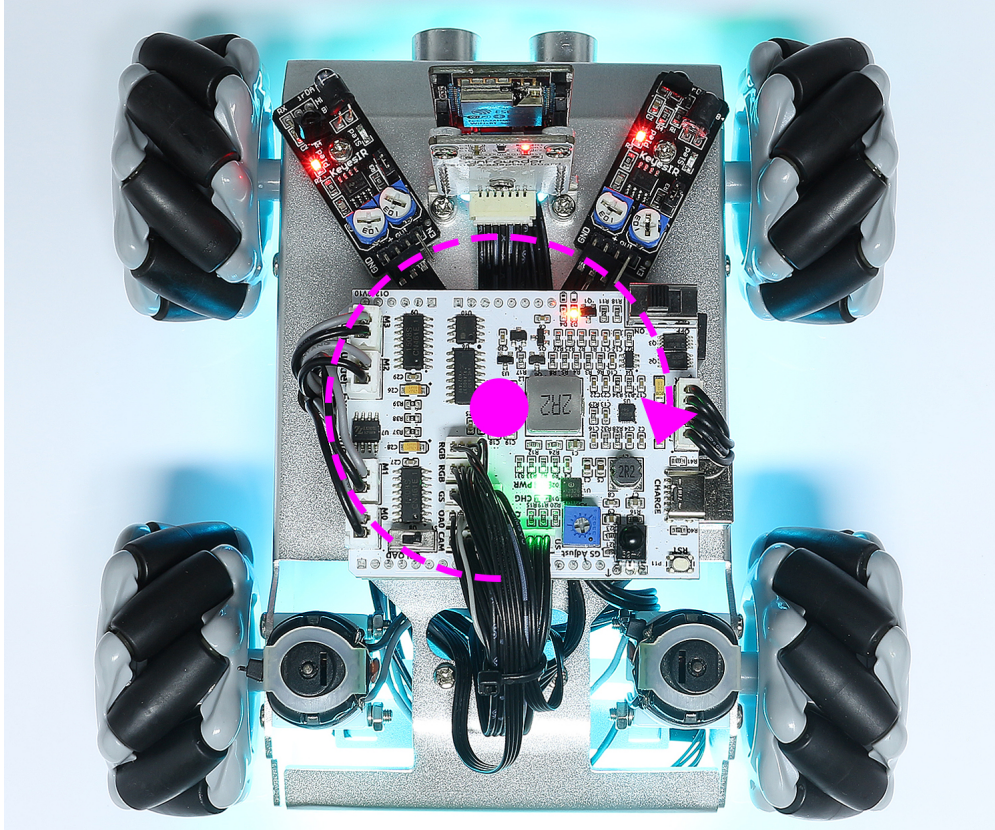


- **turn left:** This command will make the car to turn left 45° with the body as the center, then it will move forward or stop according to the previous state. If the previous state is stop, it will stop after turning left 45°; if it is forward, it will move forward after turning.

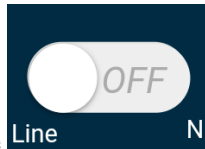


- **turn right:** This command will make the car turn  $45^\circ$  to the right with the body as the center, and again will move forward or stop depending on the previous state.





### Line Track



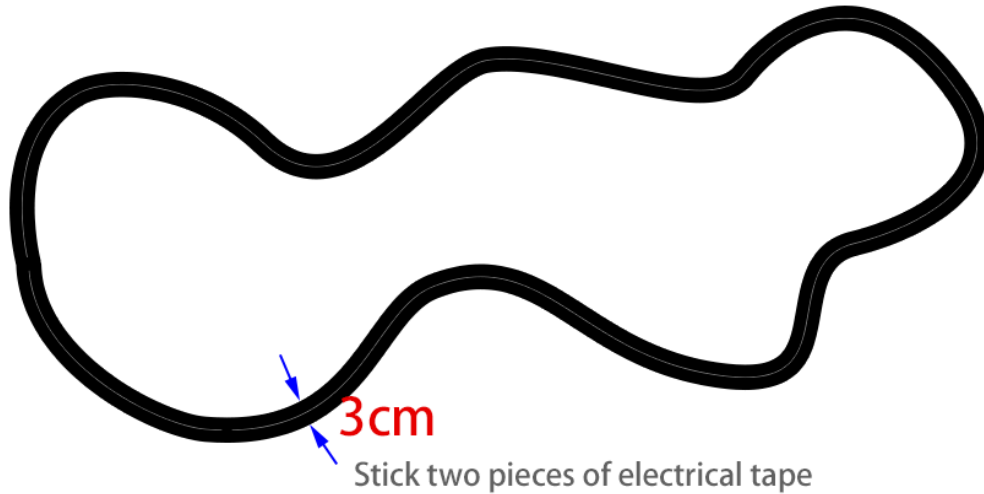
Click on the **Line** widget to switch to the line track mode.

Two modes of line track are available on the Zeus Car, one with its head always facing the direction of movement and one with its head facing a fixed direction. Here, the second mode is selected.

1. Stick a 3cm wide line

There are eight sensors on the Omni grayscale module, and the distance between each sensor is between 2 and 3 cm. There must be two sensors to detect the black line simultaneously. Therefore, the line you stick must be at least 3cm wide and the bend angle should not be less than 90°.

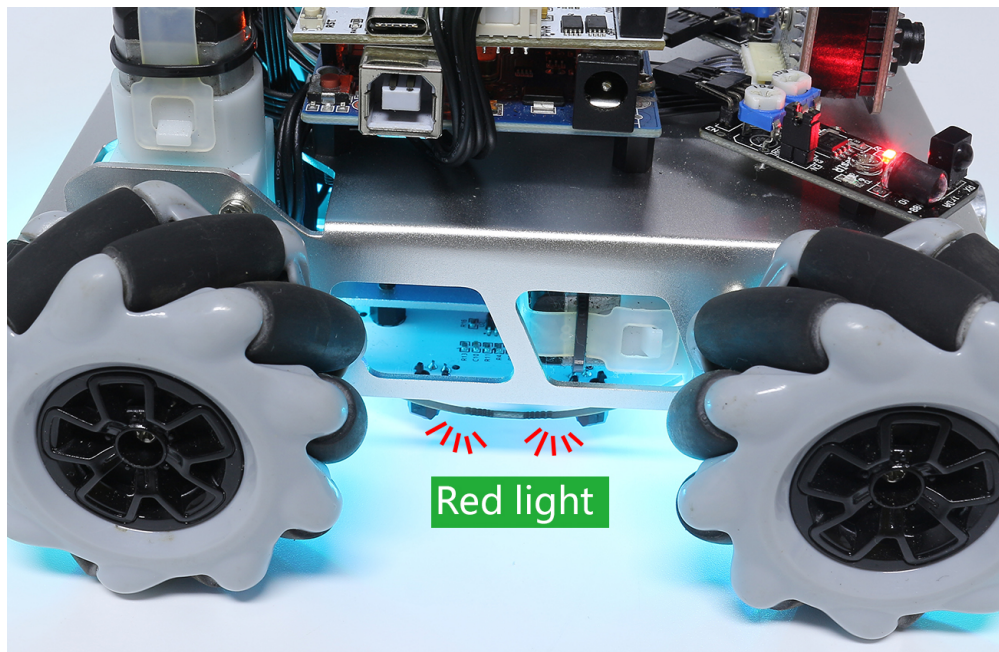




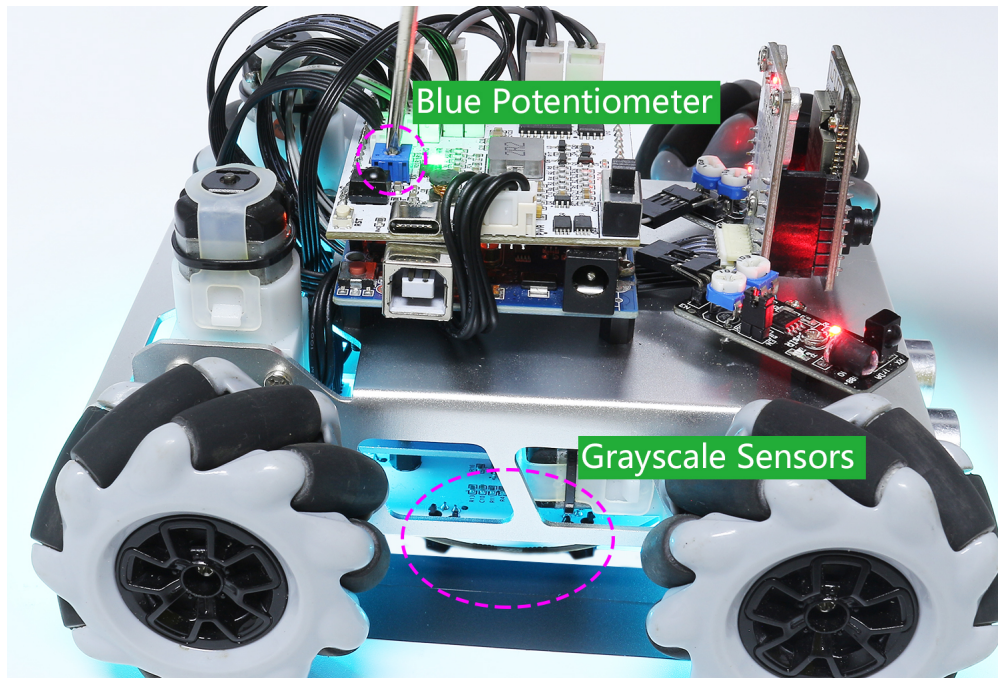
2. Calibrate the Omni Grayscale module.

Since each subfloor has different grayscale values, the factory-set grayscale threshold may not be appropriate for your current environment, so you will need to calibrate this module before use. It is recommended that you need to calibrate it whenever the floor color changes a lot.

- Place the Zeus Car on white surface and turn the potentiometer until the gray sensor light is just illuminated.



- Now let the two greyscale sensors on the side be located just between the black line and white surface, and slowly turn the potentiometer until the signal indicator just goes off.

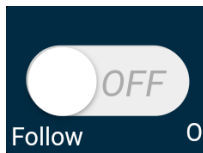


- You can move repeatedly over the the black line and white surface to make sure that the lights of the greyscale sensor are off when they are between the the black line and white surface and on when they are on the white surface, indicating that the module is successfully calibrated.



3. Place the Zeus Car on your stickied line, click the **Line** widget, and it will track the line.
4. Due to the high environmental requirements of the Omni grayscale module, it is recommended to calibrate it a few more times if the tracking effect is not satisfactory (off-track).

## Follow(O)

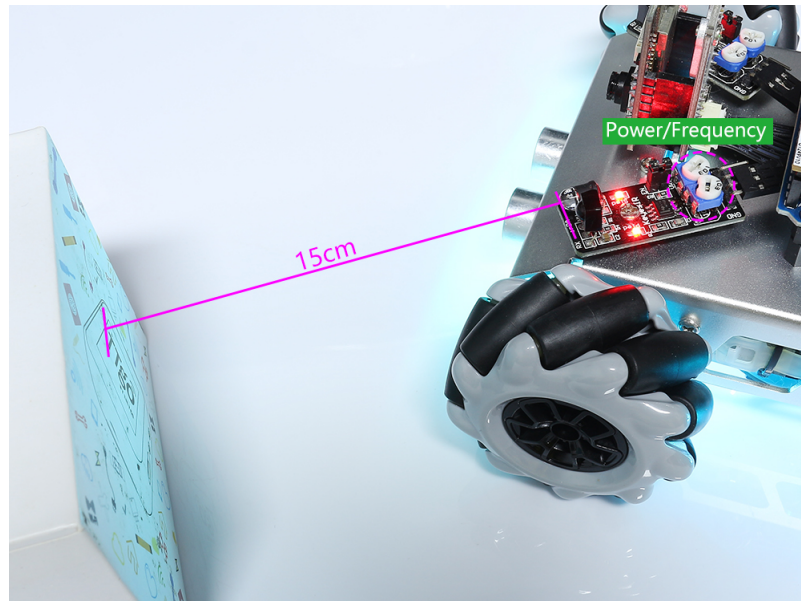


Click the **Follow** widget to switch to follow mode.

The ultrasonic sensor detects obstacles in front (20 cm) and follows them forward. These two obstacle avoidance modules allow the car to follow left or right, but they need to be calibrated (15cm) before use.

1. Calibrate the IR obstacle avoidance module.
  - Start by adjusting the right obstacle avoidance module. During transportation, collisions may cause the transmitter and receiver on the infrared module to tilt. Therefore, you need to manually straighten them.
  - Place an obstacle about 15cm away from the IR obstacle avoidance module.
  - On the module are two potentiometers, one to adjust the sending power and one to adjust the sending frequency. By adjusting these two potentiometers, you can adjust the detection distance.

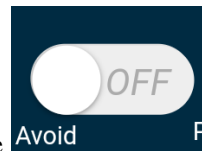
- Then you can adjust a potentiometer, and if at 15cm, the signal light on the module illuminates, the adjustment is successful; if it doesn't, adjust another potentiometer.



- Calibrate the other obstacle avoidance module in the same way.

2. Place Zeus car on a table or the ground and let it follow your hand or other obstacles.

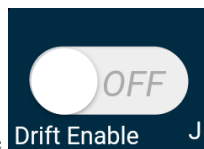
## Avoid(P)



When you want to go into obstacle avoidance mode, click the **Avoid P** widget, but first reference the *Follow(O)* to calibrate the two obstacle avoidance modules.

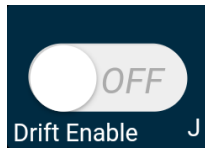
- Zeus car will move forward.
- An ultrasonic module detects obstacles in front, if detected, the car turns left.
- When the left obstacle avoidance module detects an obstacle, the car turns right, and when the right obstacle avoidance module detects an obstacle, the car turns left.

## Control the Drection(Q)



- When the **Drift Enable J** button is on, the **Q** widget is used to make the Zeus Car drift left and right.

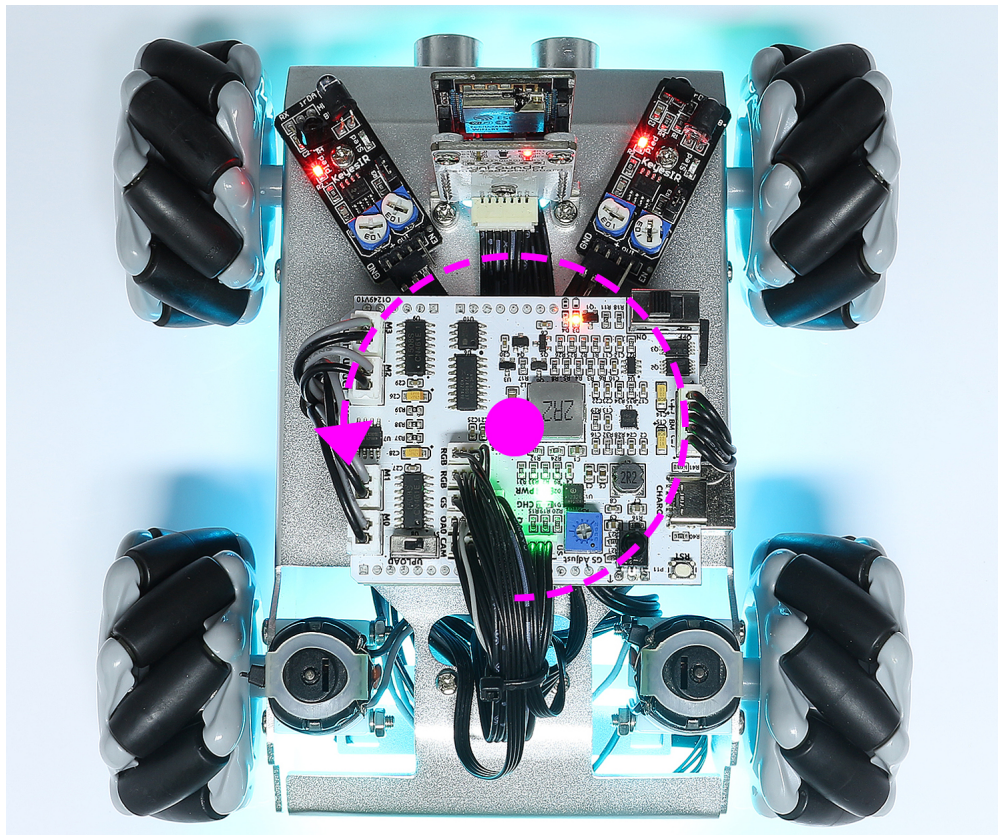




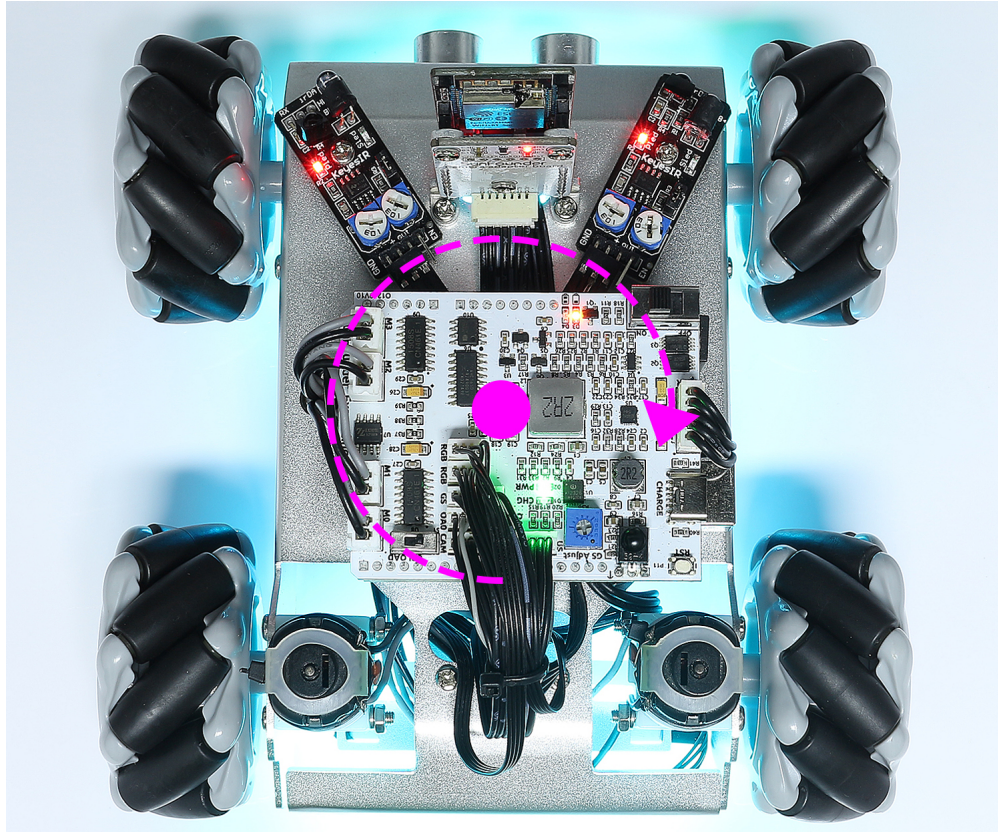
- When the Drift Enable widget is off, the



- By sliding the widget counterclockwise, the car will also rotate counterclockwise. Upon releasing the hand, the head of the car will back to the original direction.



- Similarly the car will rotate clockwise with the widget and return to the original direction when released.



## 1.2.2 Control by Remote

Zeus Car can be controlled with a remote control by turning the car on then pressing the buttons on the remote control directly. This method is stable and less susceptible to outside interference, but the control may not be as flexible.

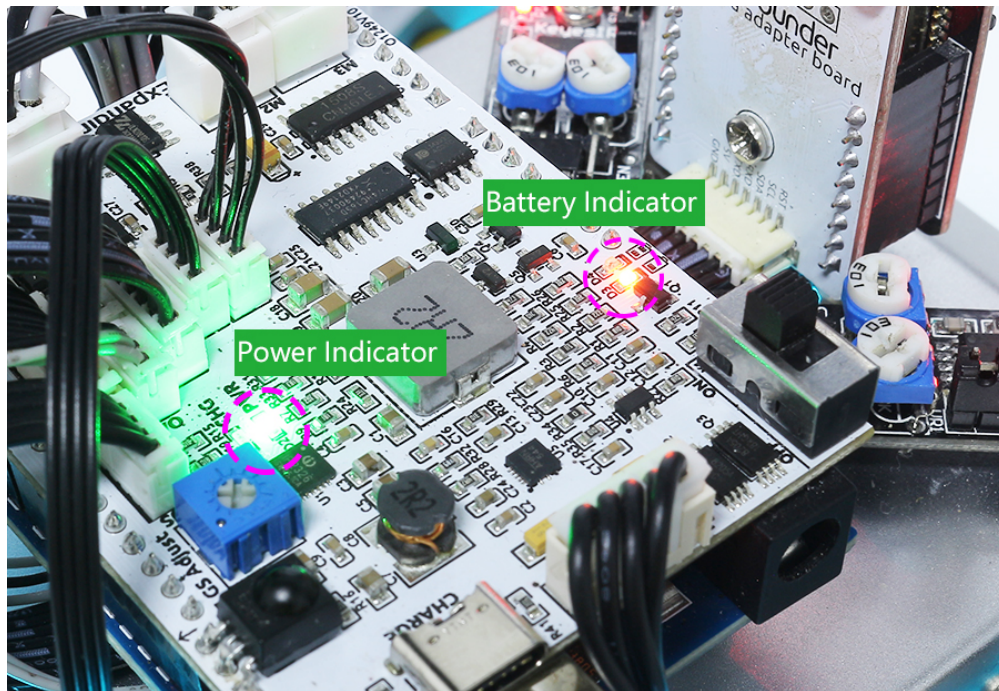
### Quick Guide

This is a quick tutorial video. Please watch the video first, and then follow the instructions provided below.

1. Let's start the Zeus Car.

- When first used or when the battery cable is unplugged, Zeus Car Shield will activate its over-discharge protection circuitry.
- So you'll need to plug in the Type-C cable for about 5 seconds.
- If the power indicator lights up, it means that the protection status has been released. At this time look at the battery indicators, if both battery indicators are off, please continue to plug in the Type-C cable to charge the battery.





- Now, you'll need to flick the smaller switch to the right to establish communication between the car and the ESP32 CAM. Afterward, press the Reset button to reboot the code. At this point, you'll observe the undercarriage lights transition from orange to a light blue.
2. Press the different keys on the remote control to control the car.



- **Stop:** Stop all movements of the car.
- *Compass Calibration:* Turn on compass calibration.
- *Line Track:* Switching to line track mode.
- *Follow:* Switching to follow mode.
- *Obstacle Avoidance:* Switch to obstacle avoidance mode.
- *Drift Left/Right:* Drift to the left/right.
- **Set Heading:** After placing the car in one direction with your hand, click on this key to make this direction as the front of the car movement. This allows you to quickly specify a direction instead of slowly rotating the car to that direction with other keys.
- *Rotate Left/Right(Cycle/USD Key):* Turn left/right
- *Move in All Directions(1 ~ 9):* Control the car to move in all directions.
- **Pause:** The function is basically the same as Stop, but if the head of the car is not facing the direction originally set, it will slowly move to the set direction.

## Compass Calibration



Turn on compass calibration by pressing the key.

Place the Zeus car on the ground. Upon turning on the compass calibration, the car will start rotating counterclockwise and will stop in about 1 minute. If it rotates longer than 2 minutes, the magnetic field here is complicated. Try changing the location and calibrating again.

## Line Track



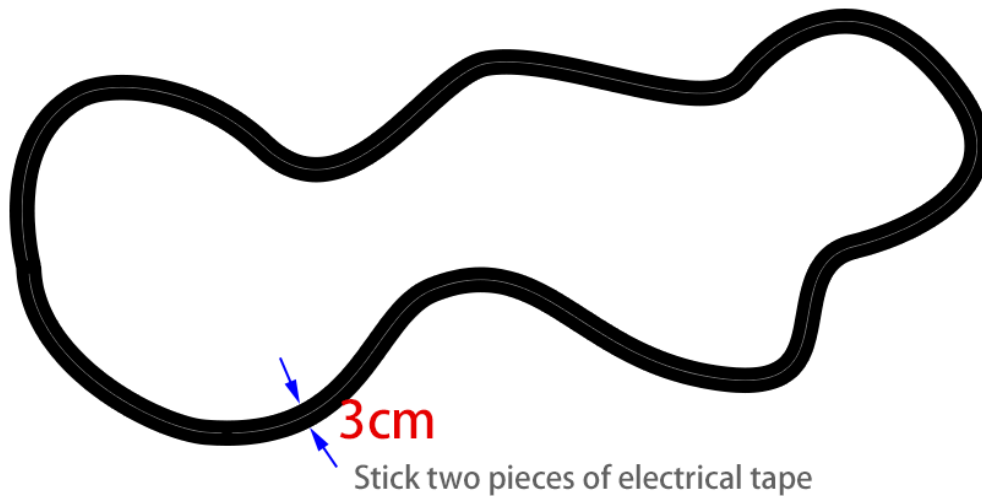
Press the key to switch to the line track mode.

Two modes of line track are available on the Zeus Car, one with its head always facing the direction of movement and one with its head facing a fixed direction. Here, the second mode is selected.

But before you can get it to follow the line, you need to calibrate the Omni Grayscale module and stick the line, as follows.

### 1. Stick a 3cm wide line

There are eight sensors on the Omni grayscale module, and the distance between each sensor is between 2 and 3 cm. There must be two sensors to detect the black line simultaneously. Therefore, the line you stick must be at least 3cm wide and the bend angle should not be less than 90°.

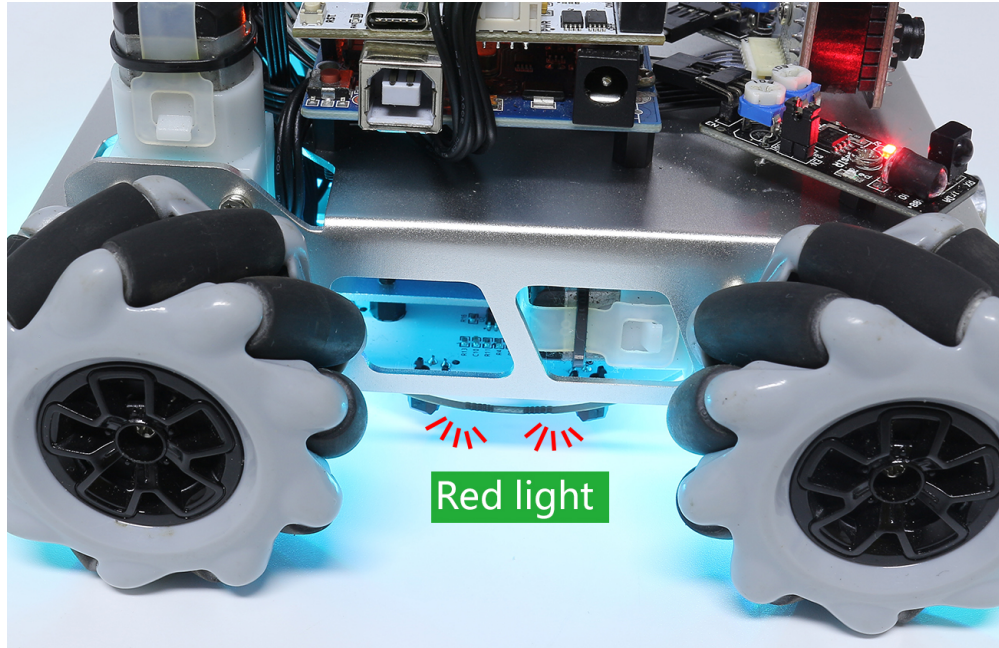


### 2. Calibrate the Omni Grayscale module.

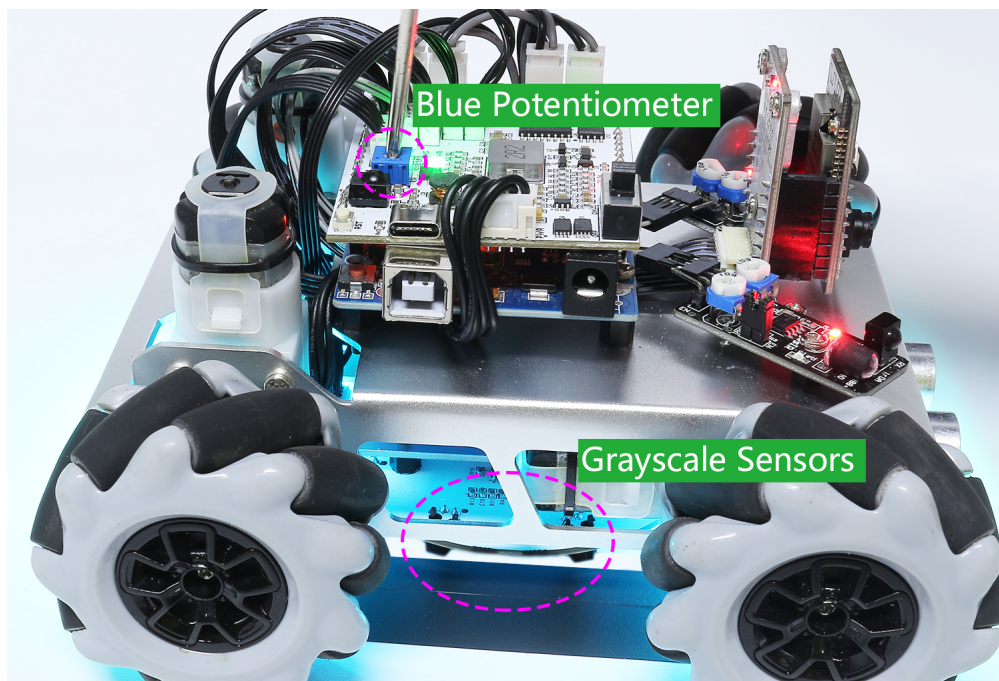
Since each subfloor has different grayscale values, the factory-set grayscale threshold may not be appropriate for your current environment, so you will need to calibrate this module before use. It is recommended that you need to calibrate it whenever the floor color changes a lot.

- Place the Zeus Car on white surface and turn the potentiometer until the gray sensor light is just illuminated.






- Now let the two greyscale sensors on the side be located just between the black line and white surface, and slowly turn the potentiometer until the signal indicator just goes off.




- You can move repeatedly over the the black line and white surface to make sure that the lights of the greyscale sensor are off when they are between the the black line and white surface and on when they are on the white surface, indicating that the module is successfully calibrated.



3. Place the Zeus Car on your stickied line, press the  key, and it will track the line.
4. Due to the high environmental requirements of the Omni grayscale module, it is recommended to calibrate it a few more times if the tracking effect is not satisfactory (off-track).

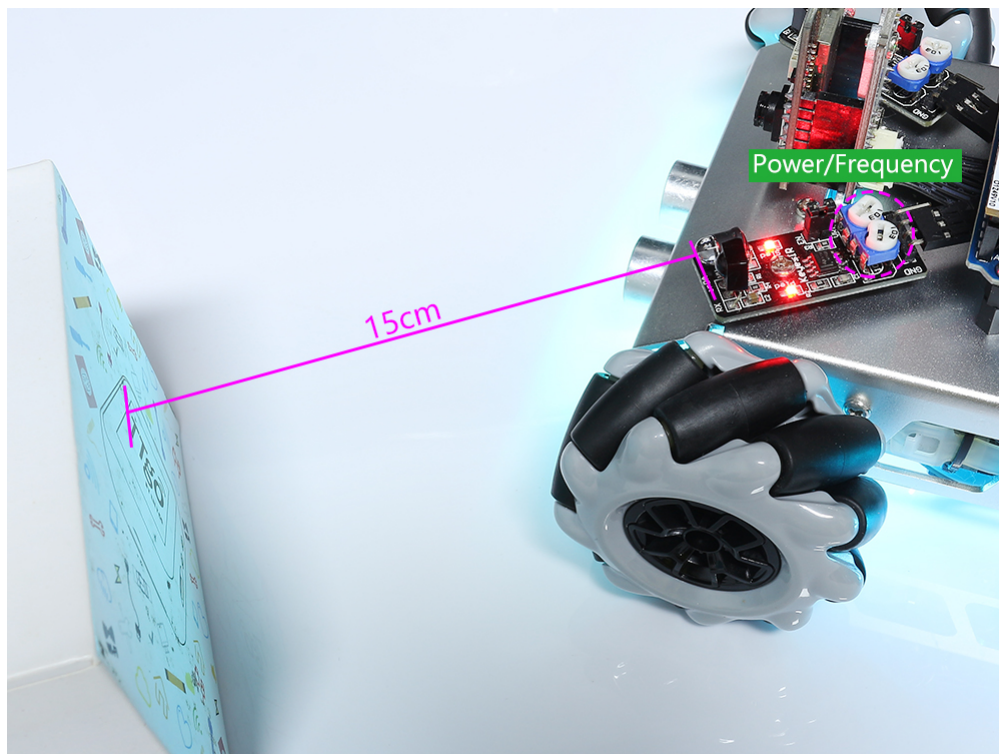
### Follow

Press the  key to switch to follow mode.

The ultrasonic sensor detects obstacles in front (20 cm) and follows them forward. These two obstacle avoidance modules allow the car to follow left or right, but they need to be calibrated (15cm) before use.

#### 1. Calibrate the IR obstacle avoidance module.


- Start by adjusting the right obstacle avoidance module. During transportation, collisions may cause the transmitter and receiver on the infrared module to tilt. Therefore, you need to manually straighten them.
- Place an obstacle about 15cm away from the IR obstacle avoidance module.
- On the module are two potentiometers, one to adjust the sending power and one to adjust the sending frequency. By adjusting these two potentiometers, you can adjust the detection distance.
- Then you can adjust a potentiometer, and if at 15cm, the signal light on the module illuminates, the adjustment is successful; if it doesn't, adjust another potentiometer.



- Calibrate the other obstacle avoidance module in the same way.

#### 2. Place Zeus car on a table or the ground and let it follow your hand or other obstacles.


## Obstacle Avoidance

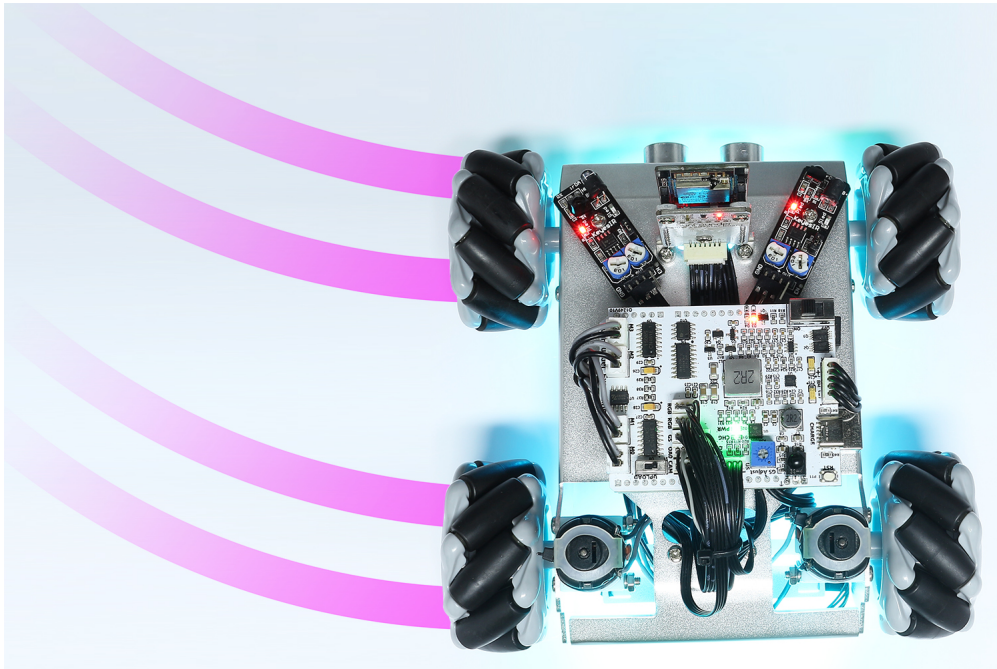
When you want to go into obstacle avoidance mode, press the  key, but first reference the *Follow* to calibrate the two obstacle avoidance modules.


- Zeus car will move forward.
- An ultrasonic module detects obstacles in front, if detected, the car turns left.
- When the left obstacle avoidance module detects an obstacle, the car turns right, and when the right obstacle avoidance module detects an obstacle, the car turns left.

## Drift Left/Right

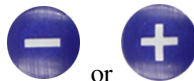
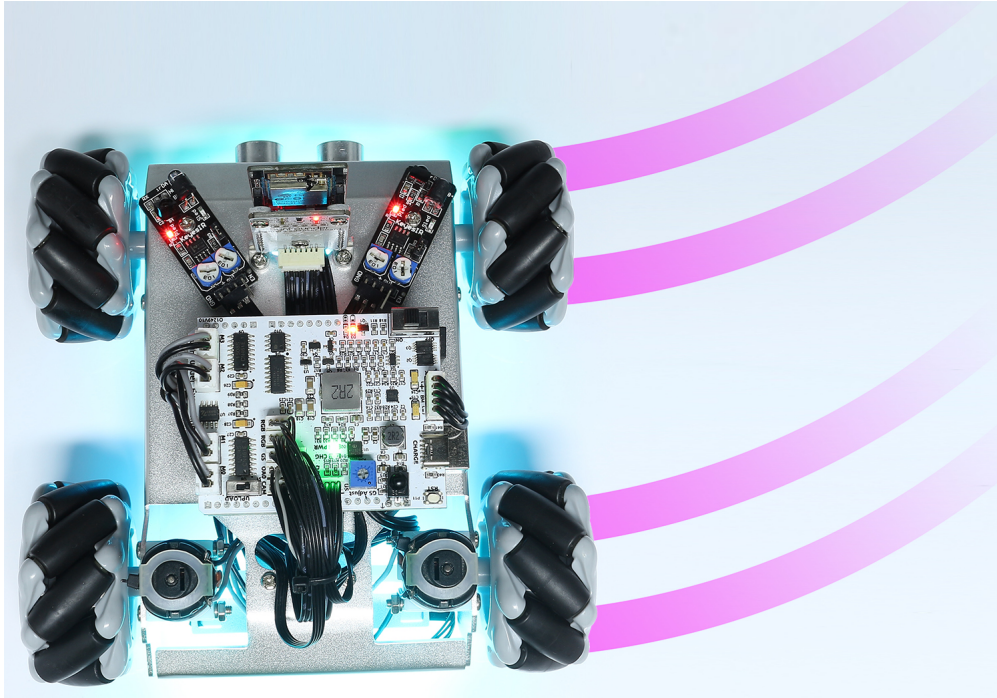
Use the  and  keys to drift the car to the left/right.



- Press the  key, the car will drift to the left.



- Press the  key, the car will drift to the right.

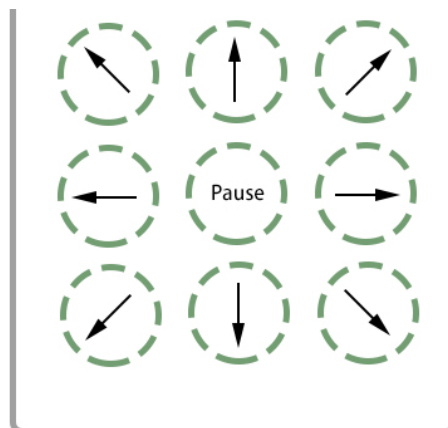





- If it was stopped before drifting, pressing the  or  key once will cause the Zeus to drift 90° to the left or right.
- If it was moving before drifting, after pressing the key, the Zeus Car will drift 90° to the left or right, then pan until you press another key.

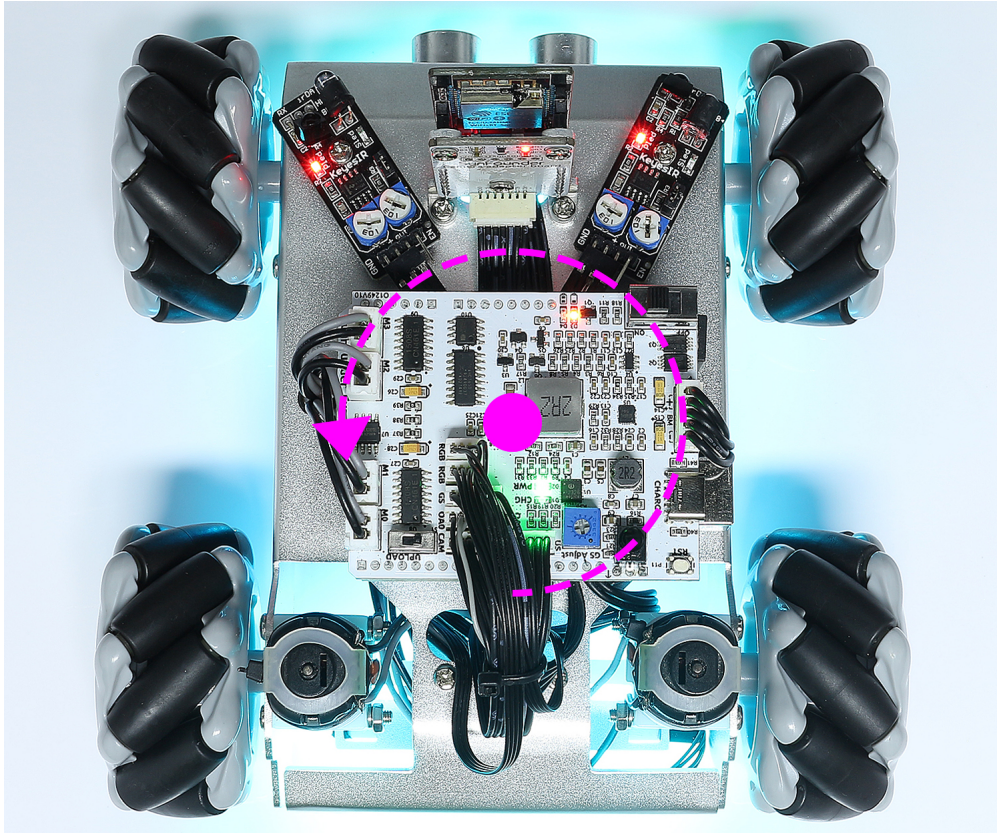
### Move in All Directions(1 ~ 9)


Use the number keys 1~9 to make the Zeus Car move in 8 directions.



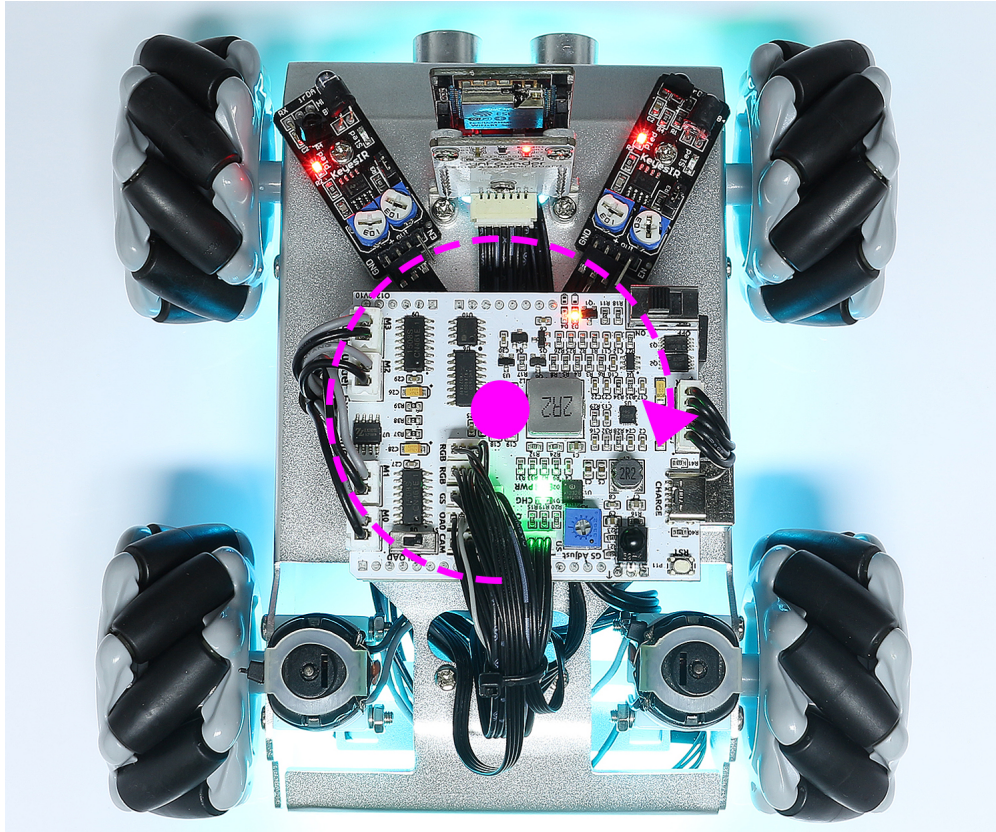
## Rotate Left/Right(Cycle/USD Key)

- The car will turn 45° left with the body as the center if you press  once. Depending on the previous state, it will move forward or stop. If the previous state is stop, it will stop after 45° left turn; if it is forward, it will move forward after the left turn.



- Similarly, pressing  once will make the car turn 45° to the right, and then it will move forward or stop depending on the previous state.





## 1.3 Programming Mode

After you have experienced *Play Mode*, if you want to understand how each function is implemented, and then modify the effect according to your own ideas, you can come to read this programming mode carefully.

Every function in the programming mode chapter is made into a separate project, there are 17 of them. Each project has corresponding code and explanation, so you can see how each function works.

This project uses the Arduino programming language, so you'll need to install the Arduino IDE and related libraries before you can begin.

Now let's start the journey of discovery!

### 1.3.1 Download the Code

Download the relevant code from the link below.

- SunFounder Zeus Car Kit for Arduino

Or check out the code at .

### 1.3.2 Download and Install Arduino IDE 2.0

The Arduino IDE, known as Arduino Integrated Development Environment, provides all the software support needed to complete an Arduino project. It is a programming software specifically designed for Arduino, provided by the Arduino team, that allows us to write programs and upload them to the Arduino board.

The Arduino IDE 2.0 is an open-source project. It is a big step from its sturdy predecessor, Arduino IDE 1.x, and comes with revamped UI, improved board & library manager, debugger, autocomplete feature and much more.

In this tutorial, we will show how to download and install the Arduino IDE 2.0 on your Windows, Mac, or Linux computer.

#### Requirements

- Windows - Win 10 and newer, 64 bits
- Linux - 64 bits
- Mac OS X - Version 10.14: “Mojave” or newer, 64 bits

#### Download the Arduino IDE 2.0

1. Visit .
2. Download the IDE for your OS version.



**Arduino IDE 2.0.0**

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through the section below.

**SOURCE CODE**

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

**DOWNLOAD OPTIONS**

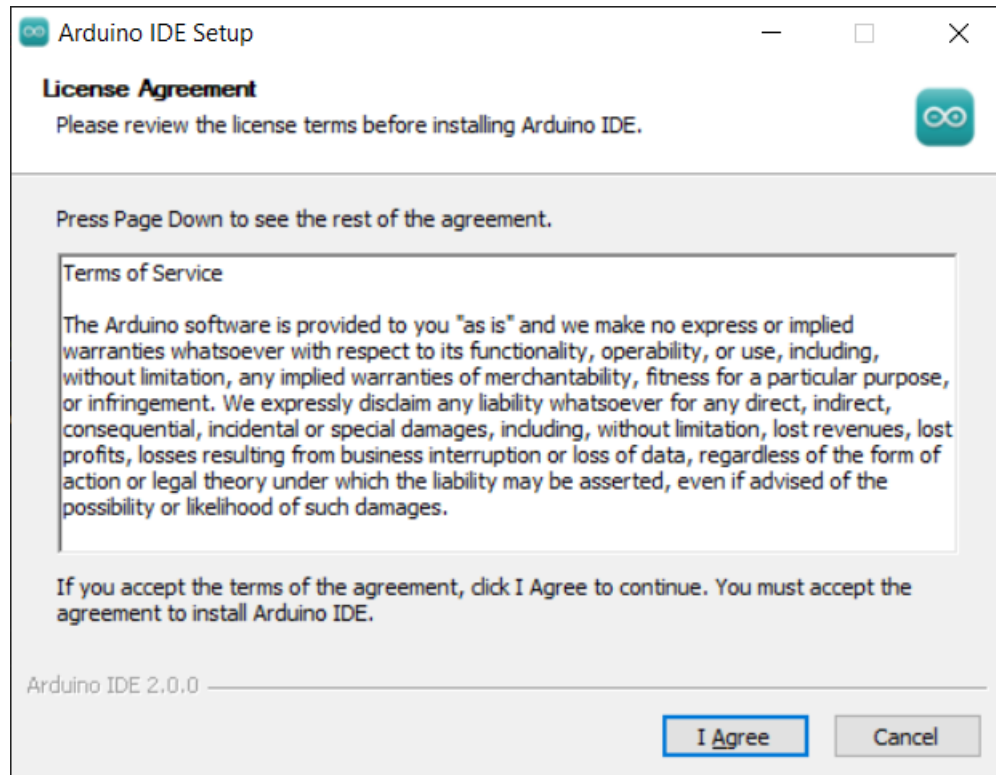
- Windows** Win 10 and newer, 64 bits
- Windows** MSI installer
- Windows** ZIP file
- Linux** AppImage 64 bits (X86-64)
- Linux** ZIP file 64 bits (X86-64)
- macOS** 10.14: “Mojave” or newer, 64 bits

#### Installation

- *Windows*
- *macOS*
- *Linux*

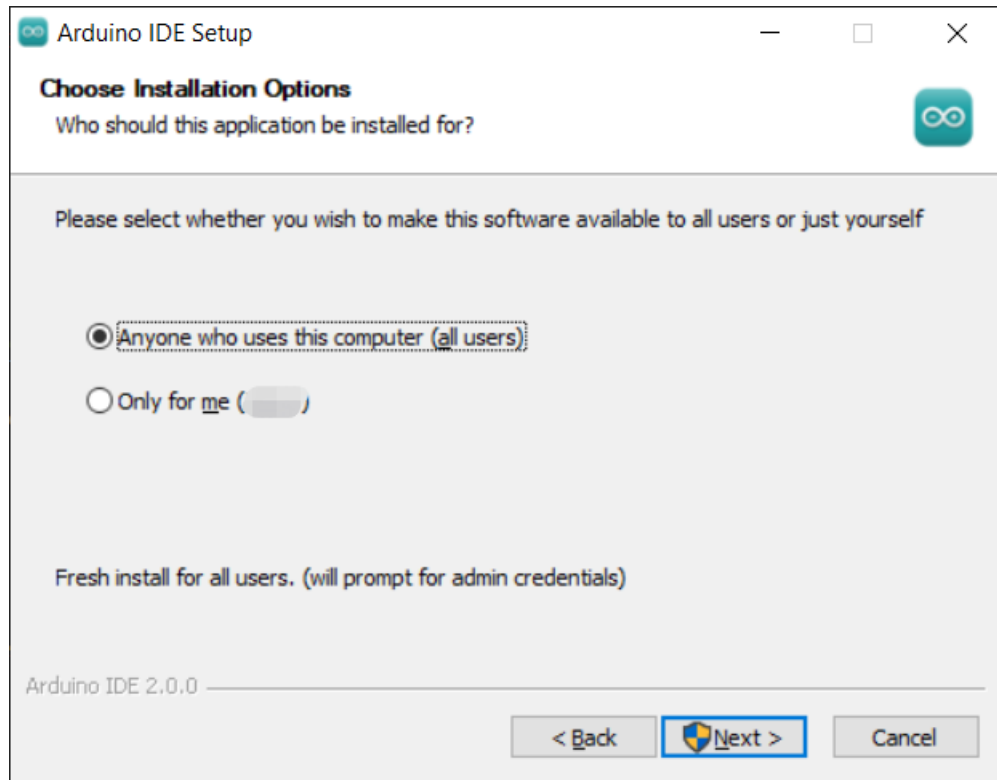
### Windows

1. Double click the `arduino-ide_xxxx.exe` file to run the downloaded file.
2. Read the License Agreement and agree it.

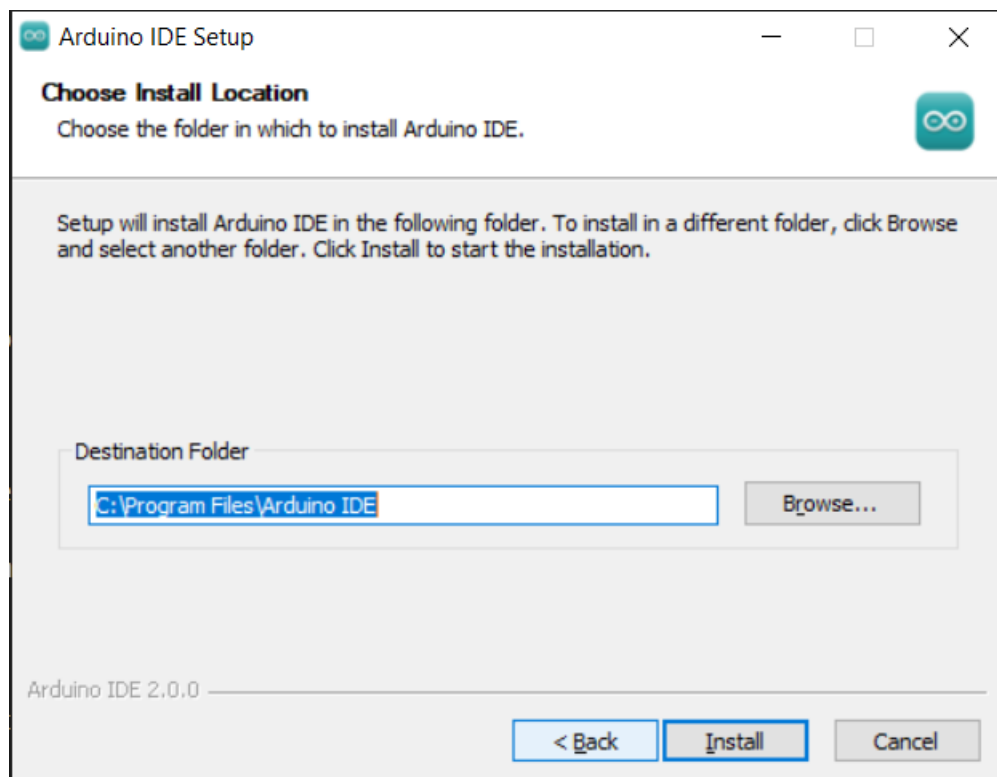


3. Choose installation options.

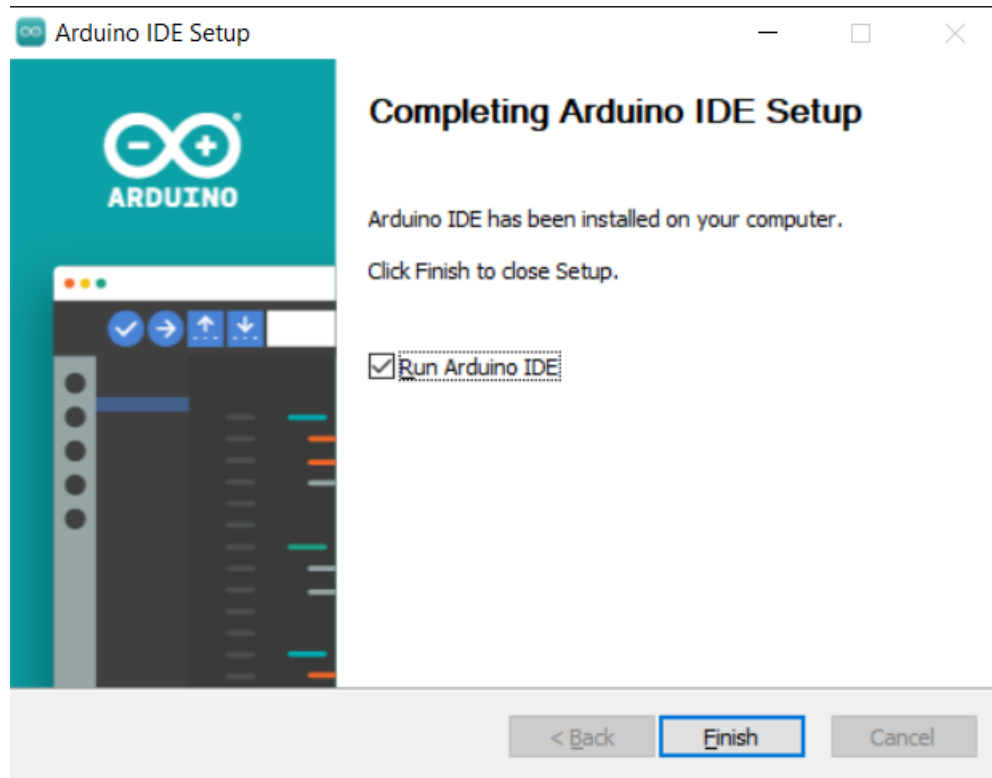




4. Choose install location. It is recommended that the software be installed on a drive other than the system drive.

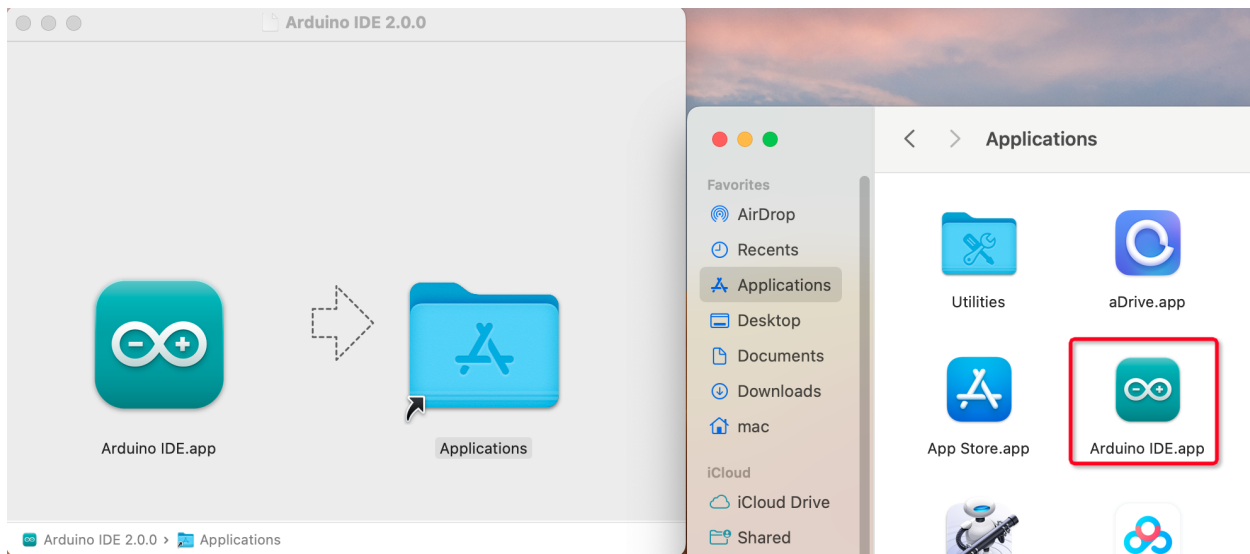


5. Then Finish.



## macOS

Double click on the downloaded `arduino_ide_xxxx.dmg` file and follow the instructions to copy the **Arduino IDE.app** to the **Applications** folder, you will see the Arduino IDE installed successfully after a few seconds.

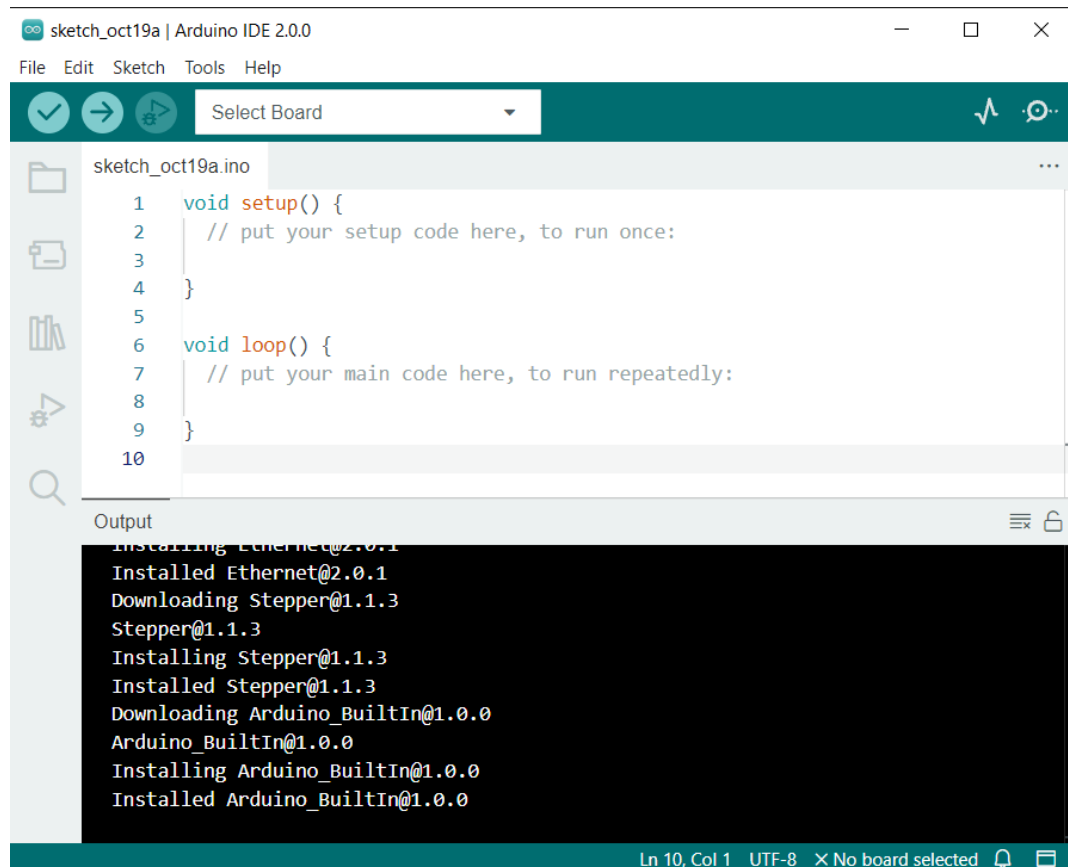


## Linux

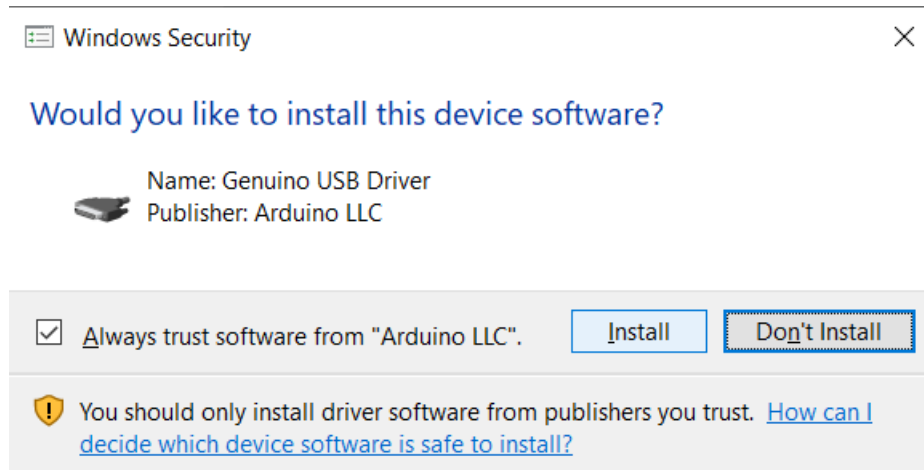
For the tutorial on installing the Arduino IDE 2.0 on a Linux system, please refer to: <https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing#linux>

### Open the IDE

1. When you first open Arduino IDE 2.0, it automatically installs the Arduino AVR Boards, built-in libraries, and other required files.



2. In addition, your firewall or security center may pop up a few times asking you if you want to install some device driver. Please install all of them.



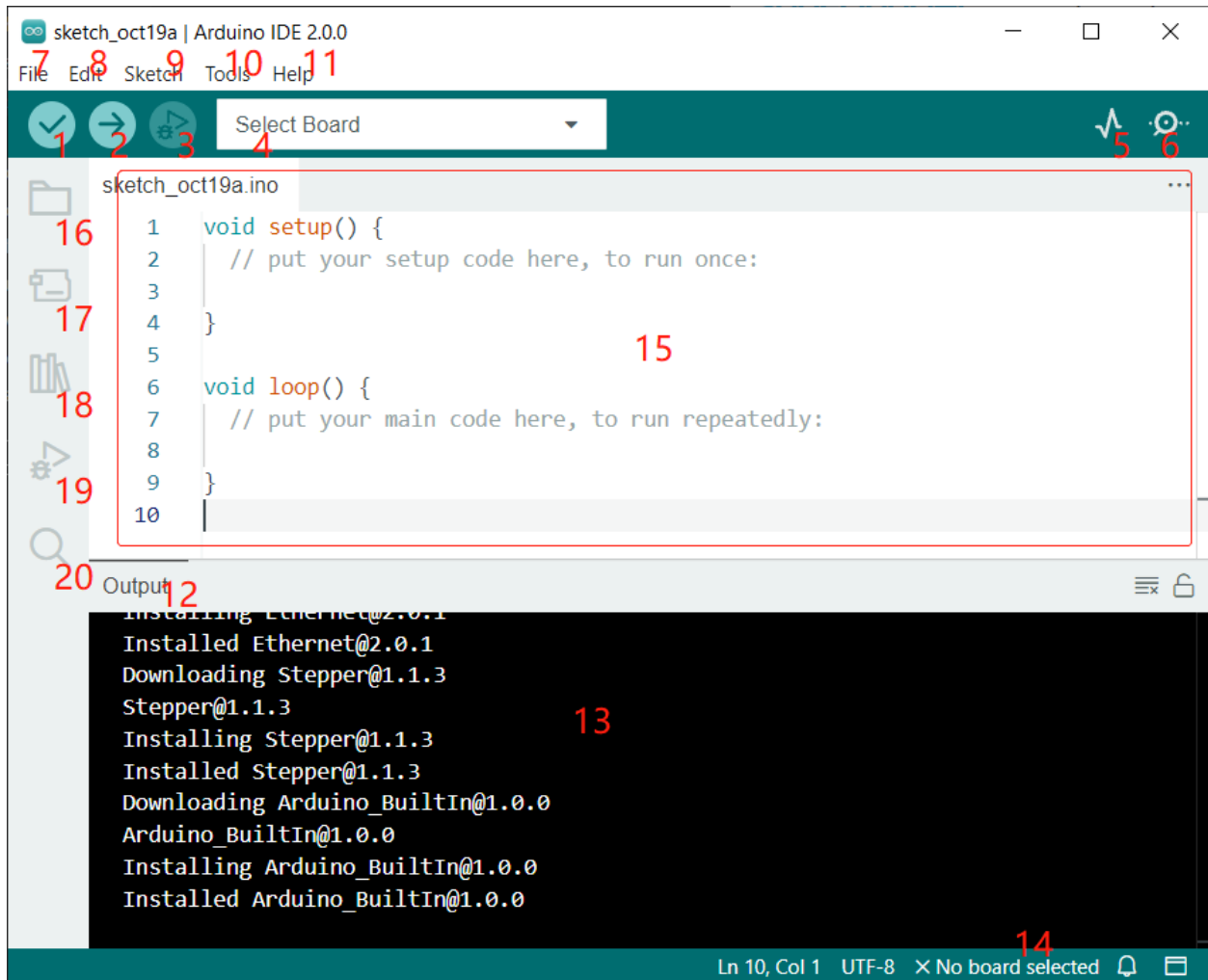
3. Now your Arduino IDE is ready!

---

**Note:** In the event that some installations didn't work due to network issues or other reasons, you can reopen the Arduino IDE and it will finish the rest of the installation. The Output window will not automatically open after all installations are complete unless you click Verify or Upload.

---

### 1.3.3 Introduce of Arduino IDE



1. **Verify:** Compile your code. Any syntax problem will be prompted with errors.
2. **Upload:** Upload the code to your board. When you click the button, the RX and TX LEDs on the board will flicker fast and won't stop until the upload is done.
3. **Debug:** For line-by-line error checking.
4. **Select Board:** Quick setup board and port.
5. **Serial Plotter:** Check the change of reading value.
6. **Serial Monitor:** Click the button and a window will appear. It receives the data sent from your control board. It is very useful for debugging.
7. **File:** Click the menu and a drop-down list will appear, including file creating, opening, saving, closing, some parameter configuring, etc.
8. **Edit:** Click the menu. On the drop-down list, there are some editing operations like **Cut**, **Copy**, **Paste**, **Find**, and so on, with their corresponding shortcuts.
9. **Sketch:** Includes operations like **Verify**, **Upload**, **Add** files, etc. More important function is **Include Library** – where you can add libraries.

10. **Tool:** Includes some tools – the most frequently used Board (the board you use) and Port (the port your board is at). Every time you want to upload the code, you need to select or check them.
11. **Help:** If you're a beginner, you may check the options under the menu and get the help you need, including operations in IDE, introduction information, troubleshooting, code explanation, etc.
12. **Output Bar:** Switch the output tab here.
13. **Output Window:** Print information.
14. **Board and Port:** Here you can preview the board and port selected for code upload. You can select them again by **Tools -> Board / Port** if any is incorrect.
15. The editing area of the IDE. You can write code here.
16. **Sketchbook:** For managing sketch files.
17. **Board Manager:** For managing board driver.
18. **Library Manager:** For managing your library files.
19. **Debug:** Help debugging code.
20. **Search:** Search the codes from your sketches.

### 1.3.4 Install the Required Libraries

#### What is library?

A library, gathering some function definitions and header files, usually contains two files: .h (header file, including function statement, Macro definition, constructor definition, etc.) and .cpp (execution file, with function implementation, variable definition, and so on).

When you need to use a function in some library, you just need to add a header file (e.g. `#include <dht.h>`), and then call that function. This can make your code more concise.

If you don't want to use the library, you can also write that function definition directly. Though as a result, the code will be long and inconvenient to read.

Some libraries are already built in the Arduino IDE, when some others may need to be installed. So now let's see how to install one.

#### How to install?

---

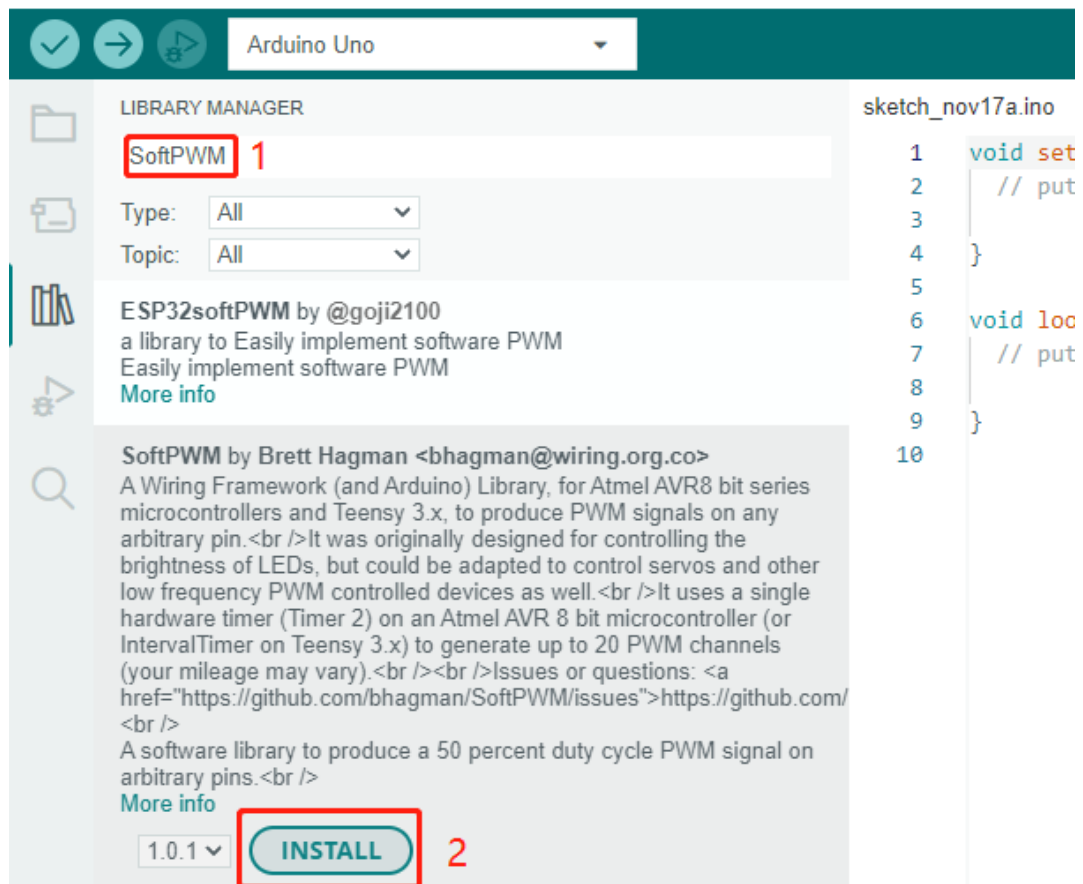
**Note:** I am using Arduino IDE 2.0, if you are using Arduino IDE 1.x, you can refer to .

---

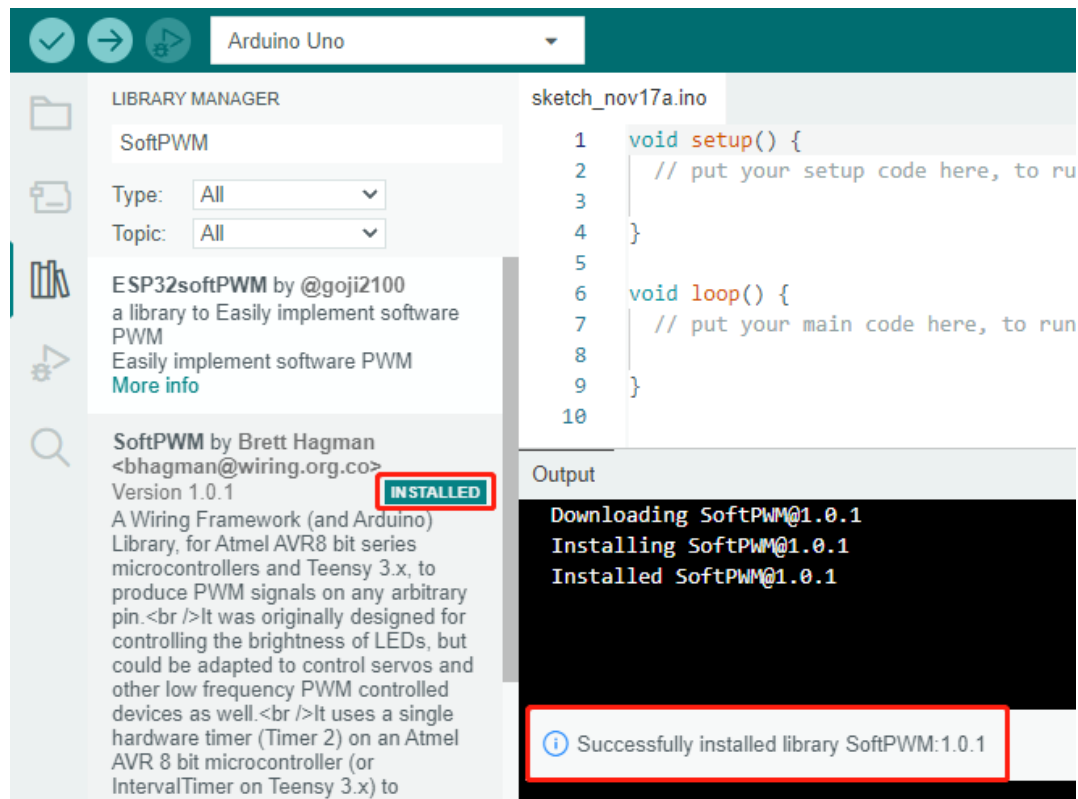
1. To install a new library into your Arduino IDE you can use the **Library Manager** which can open from the left column.



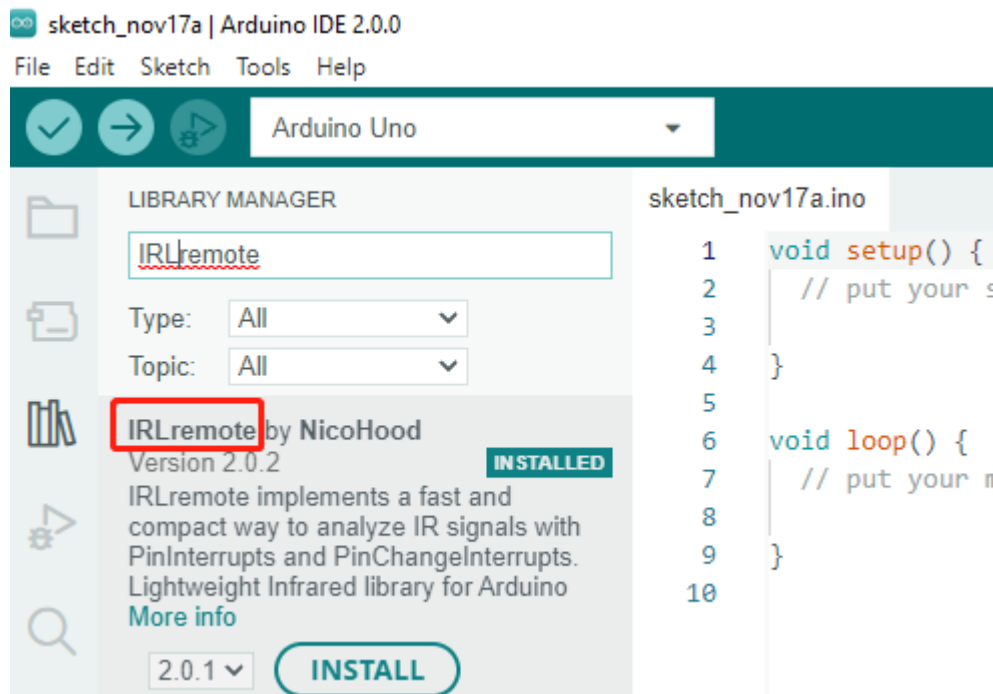
2. A list will now appear of all available libraries, where we can also search for the library we want to use. Here, we are going to install the SoftPWM library. Click on the **INSTALL** button to install the library.



3. This process should not take too long, but allow up to a minute to install it. When it is finished, we can take a look at the library in the library manager column, where it should say **INSTALLED**.

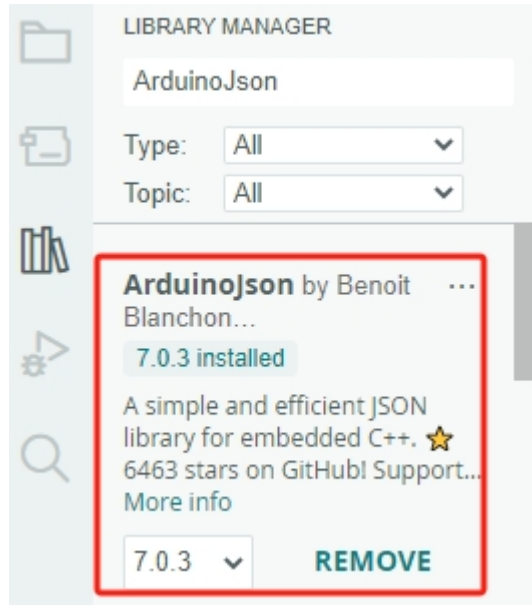


4. Install the IRLremote library in the same way, but be careful not to misspell the name.



5. Finally, install the ArduinoJson library.





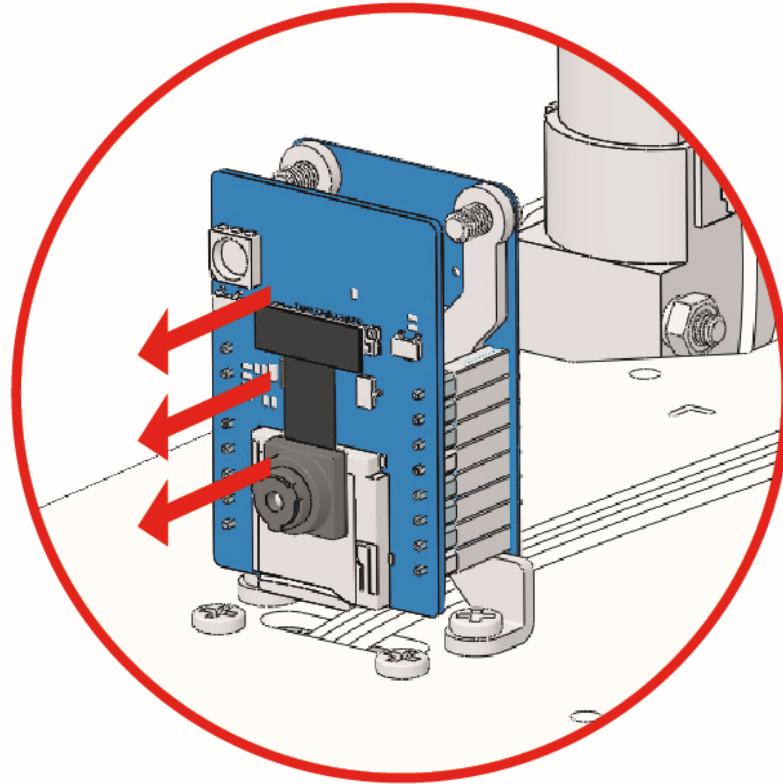
## 1.3.5 Funny Projects

### 1. Basic Movements

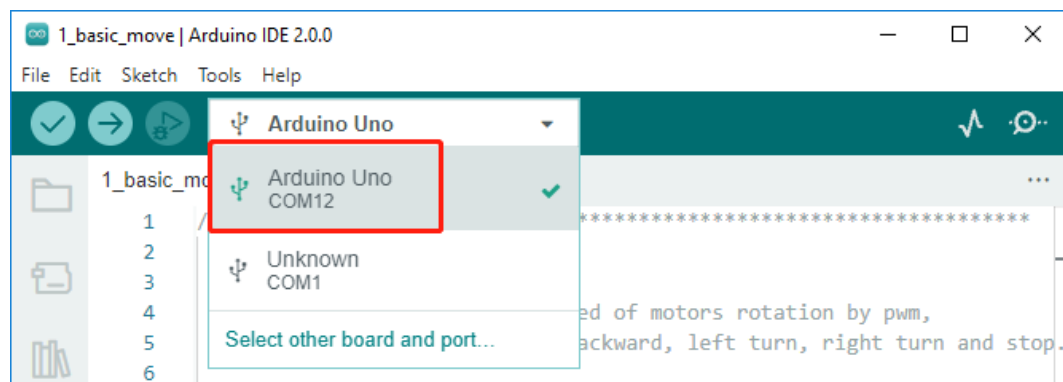
You will learn how to make the Zeus Car move in all directions in this project.

#### How to do?

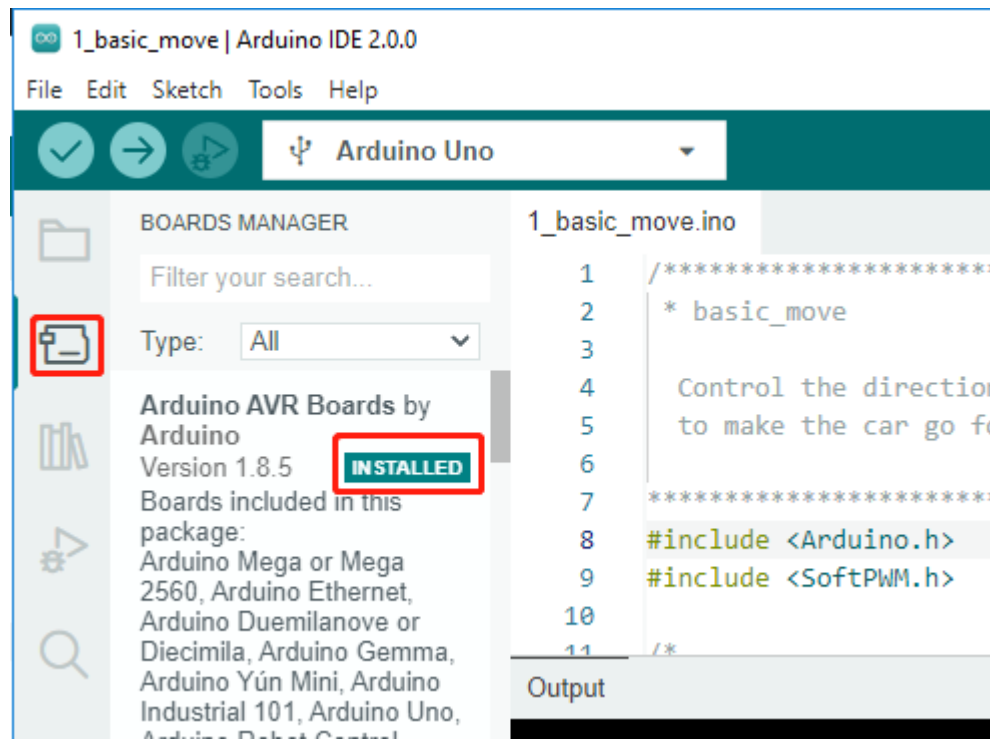
1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Connect the Zeus Car to your computer with the blue USB cable. When you plug in your Arduino board, the computer automatically recognizes it and assigns it a COM port, which you can view in Device Manager.
3. Open the `1_basic_move.ino` file under the path of `zeus-car-main\examples\1_basic_move`.
4. You can quickly select the board and port from this place.



**Note:** In case COMxx doesn't appear, open the Board Manager from the left and make sure the "Arduino AVR Boards" core is installed.

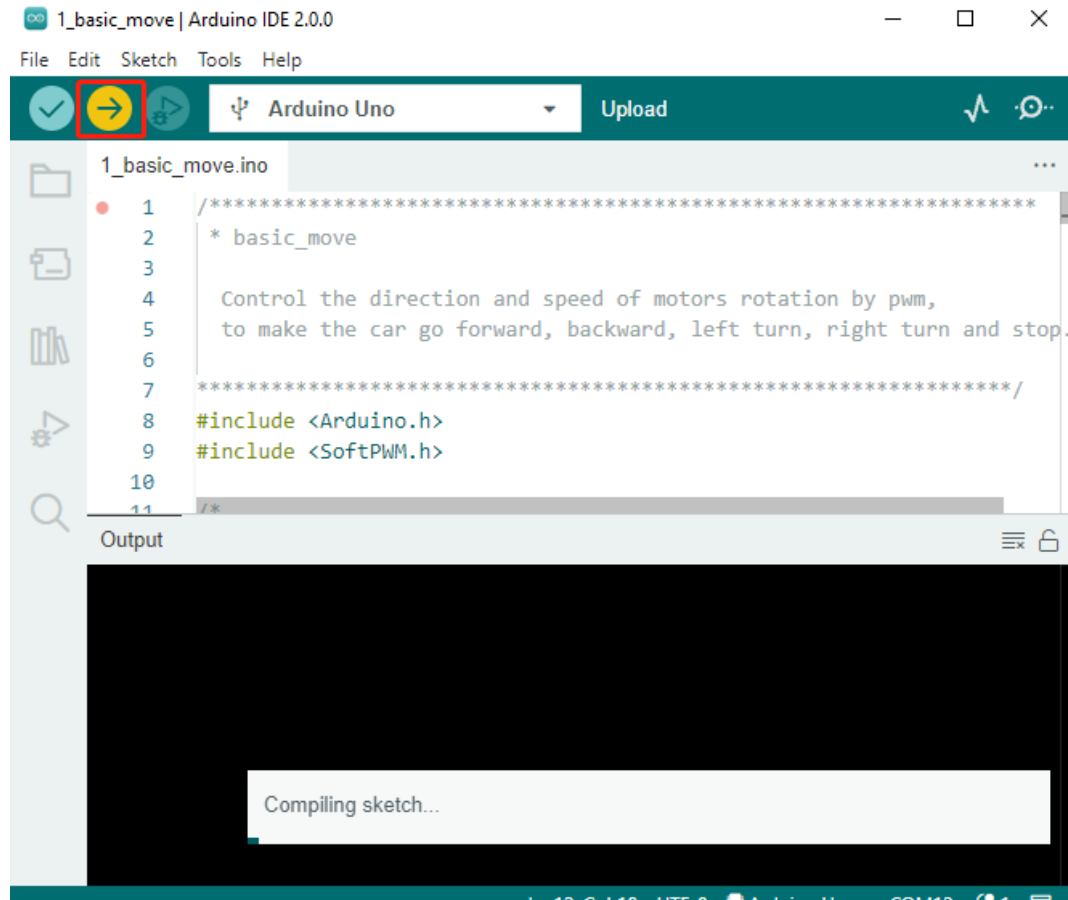


5. Now click the **Upload** button to upload the code to the Arduino board. When it is finished, a notification pops up in the bottom right of your IDE window. Of course, sometimes there are some complications when uploading, and these errors will be listed here as well.

---

**Note:**

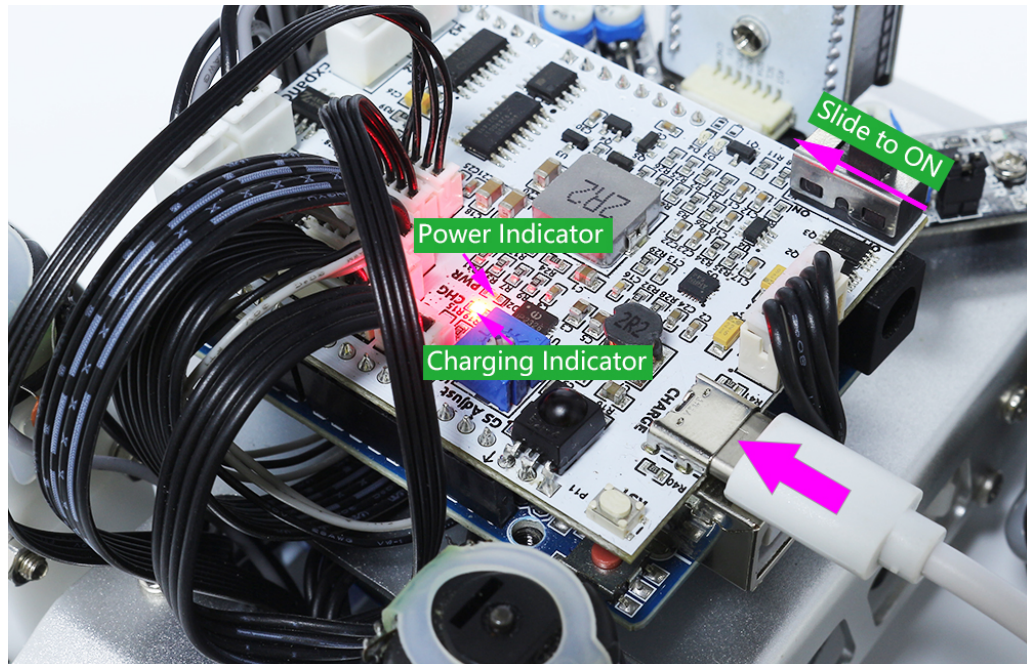
- If you get a “Compilation error: SoftPWM.h: No such file or directory” prompt, it means you don’t have the SoftPWM library installed.
  - Please refer to [Install the Required Libraries](#) to install the two required libraries SoftPWM and IRLremote.
-



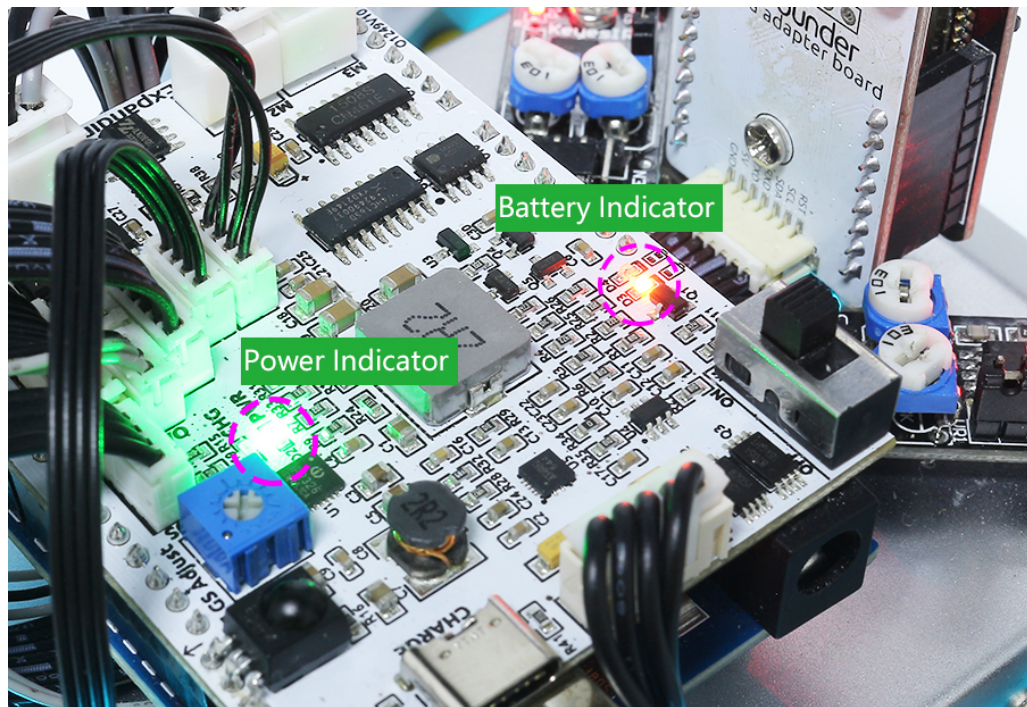
6. Let's start the Zeus Car.

- When first used or when the battery cable is unplugged, Zeus Car Shield will activate its over-discharge protection circuitry.
- So you'll need to plug in the Type-C cable for about 5 seconds.

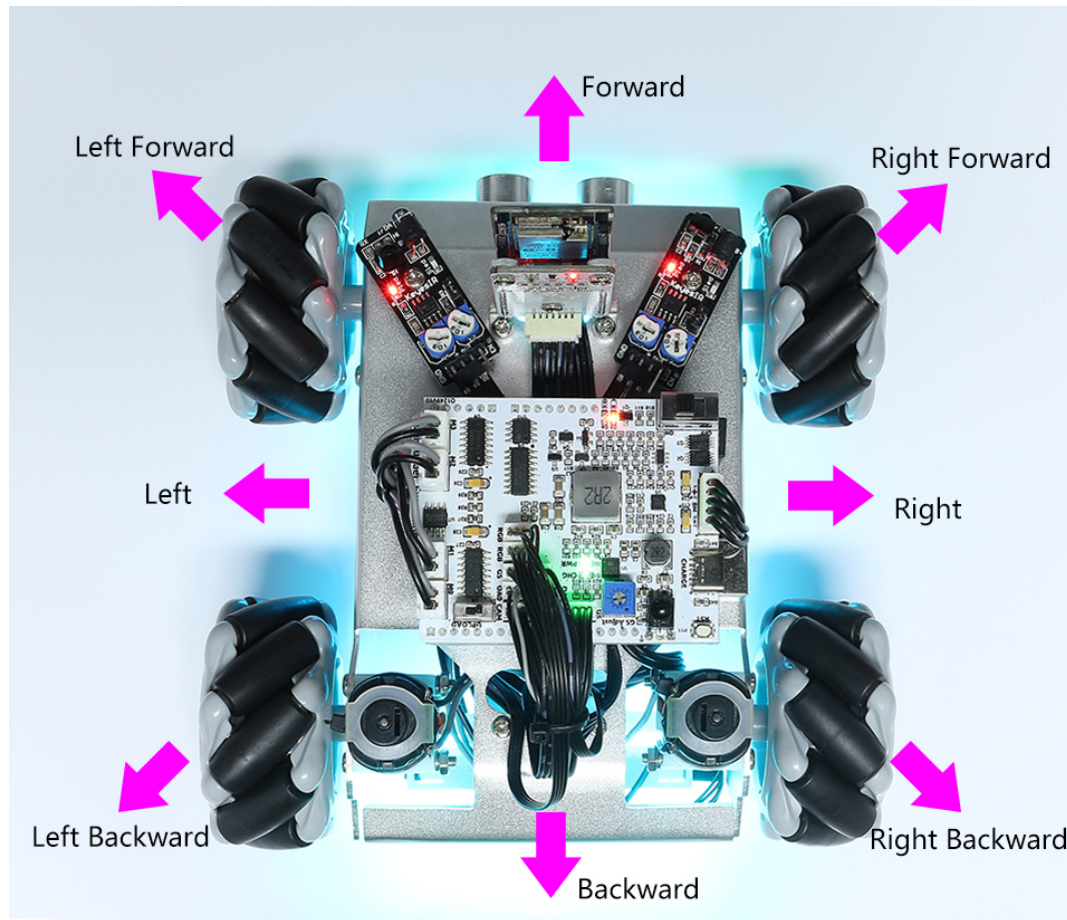




- If the power indicator lights up, it means that the protection status has been released. At this time look at the battery indicators, if both battery indicators are off, please continue to plug in the Type-C cable to charge the battery.



7. You will now see the Zeus Car move for one second in each direction.

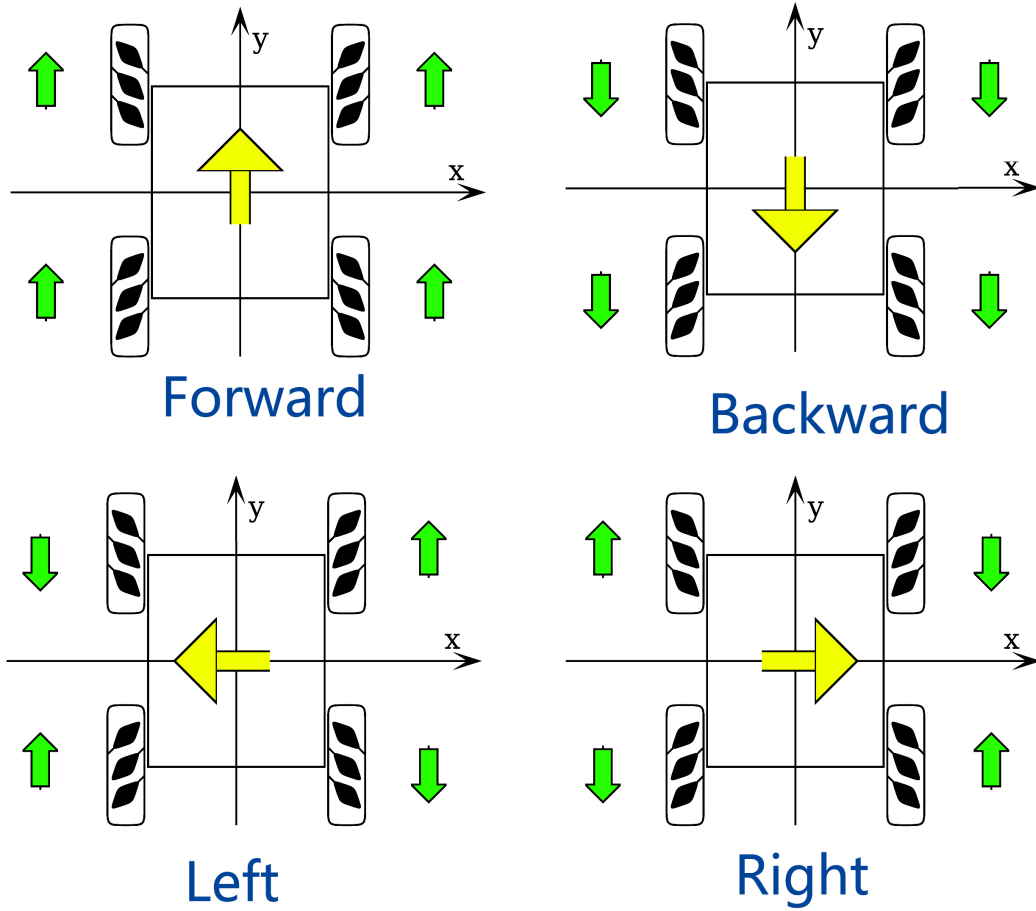


### How it works?

These movements are obtained by the 4 Mecanum Wheels cooperating together, for example, four wheels forward or backward at the same time can make the car also forward or backward, and two wheels forward and two wheels backward can cause the car to pan or rotate to the left or right.

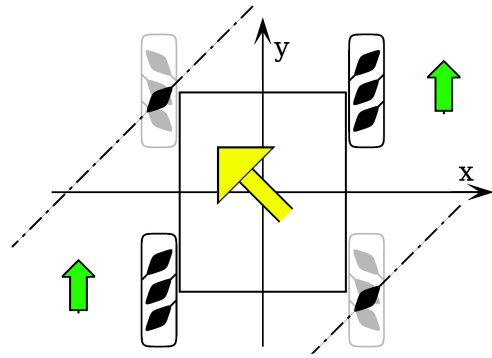
Let's look at how these movements are specifically achieved.

- Moving forward and backward, panning left and right.

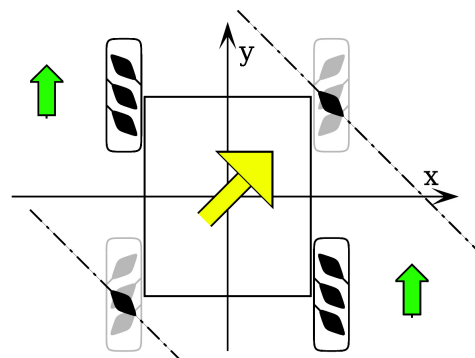


- Diagonal panning

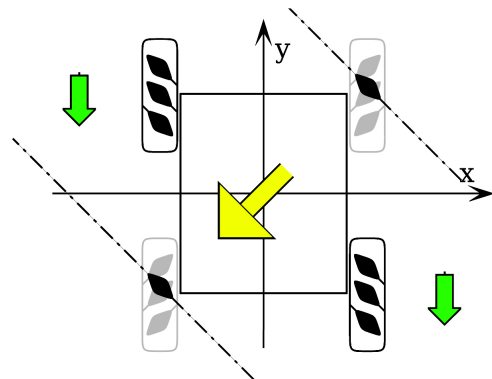




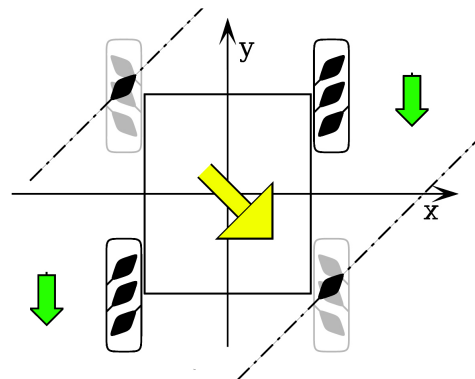
LeftForward



RightForward

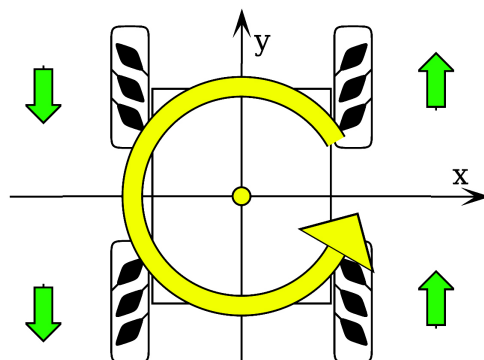


LeftBackward

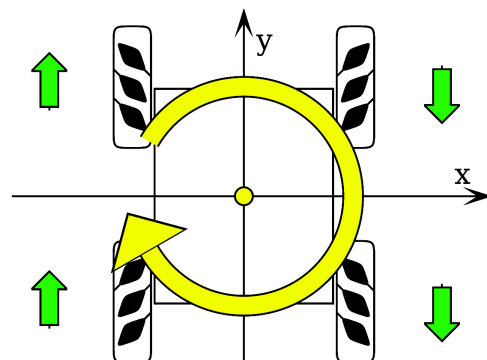


RightBackward

- Rotate left and right



Turn Left



TurnRight



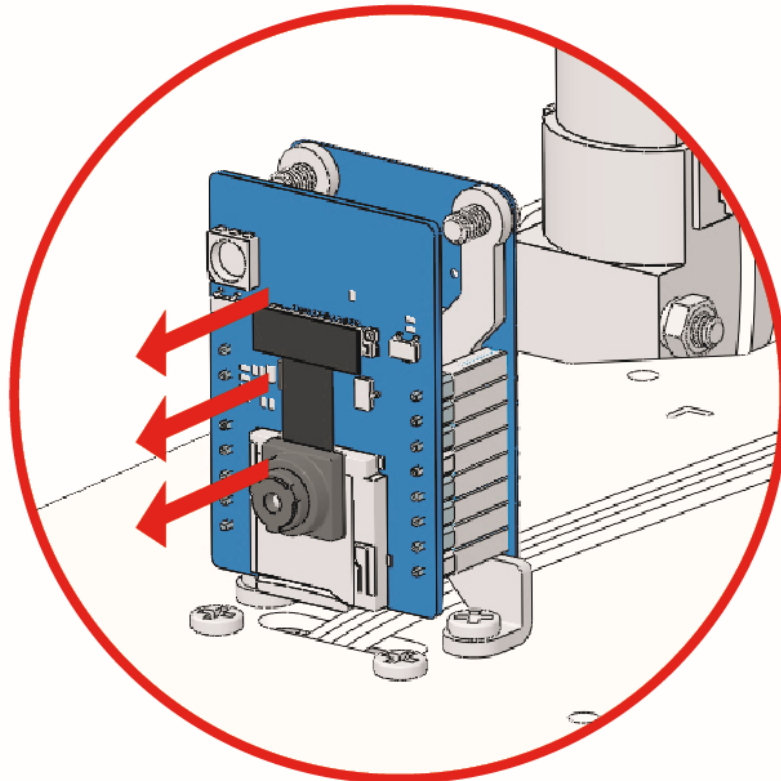
## 2. Omni Move

In this project, you will learn how to move the Zeus Car to 0, 45, 90, 135, 180, 225, 270, and 315 degrees.

If you ignore the friction on the ground and the structural tolerances, its path should be an octagon and eventually return to the origin.

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.

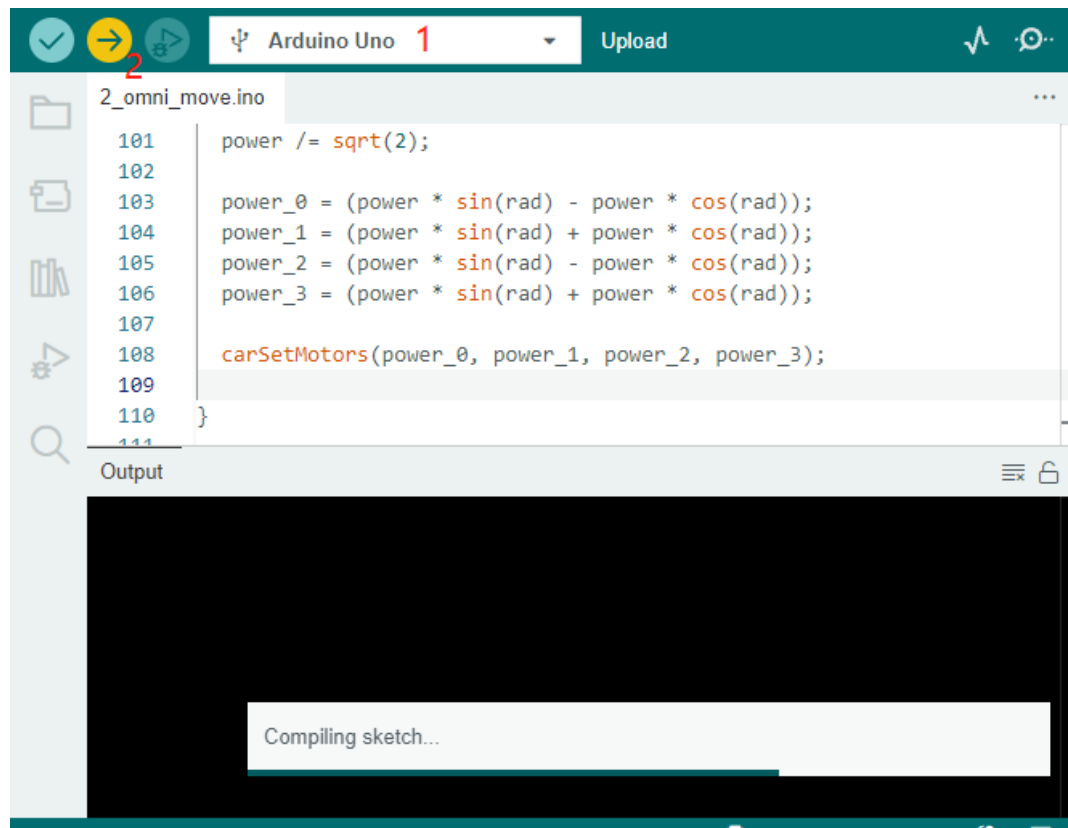


2. Open the `2_omni_move.ino` file under the path of `zeus-car-main\examples\2_omni_move`.
3. Select the correct board and port, then click the **Upload** button.

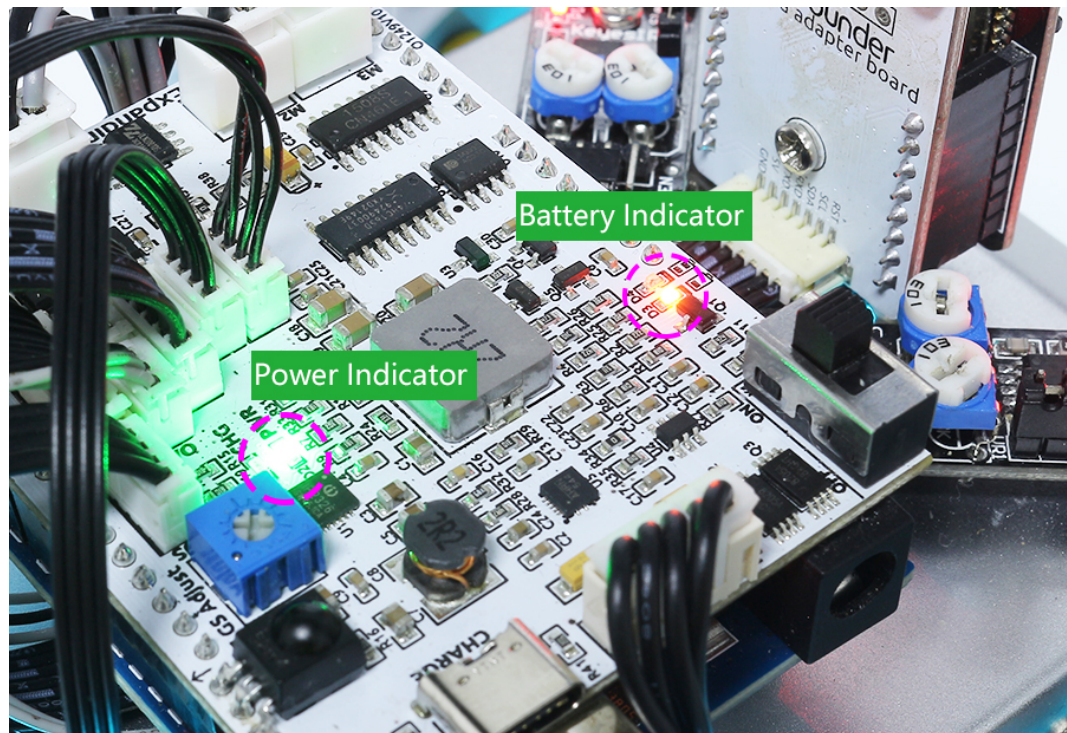
---

#### Note:

- If you get a “Compilation error: SoftPWM.h: No such file or directory” prompt, it means you don't have the SoftPWM library installed.
  - Please refer to [Install the Required Libraries](#) to install the two required libraries SoftPWM and IRLremote.
-



4. Turn the power switch to ON to start the Zeus Car.



5. At this point the Zeus Car will move to 0, 45, 90, 135, 180, 225, 270, and 315 degrees.

How it works?

The movement of the Zeus Car is mainly implemented by this function.

```
void carMove(int16_t angle, int8_t power)
```

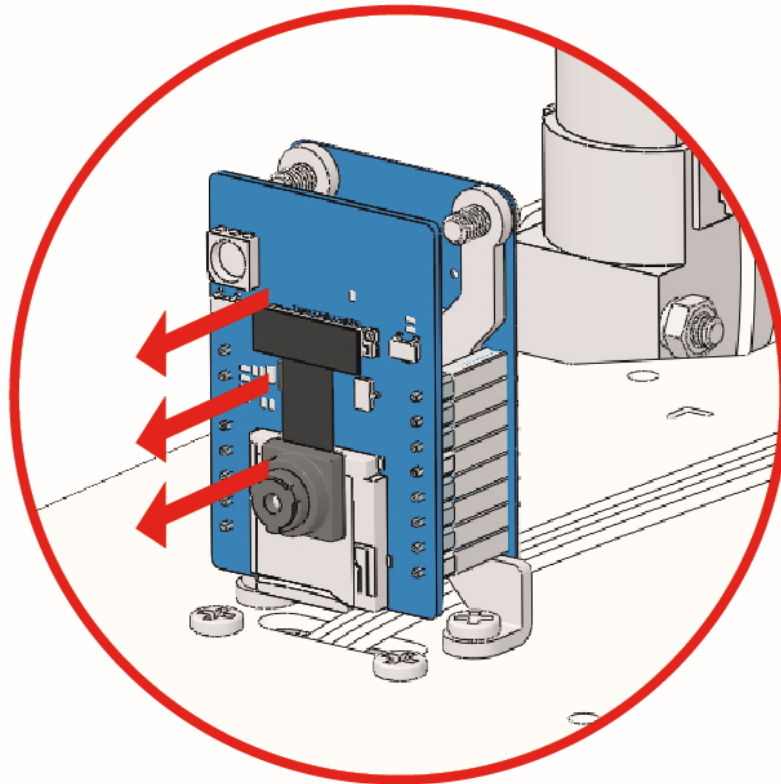
- **angle:** The direction you want the car to move. use the head of the car as the 0 degree and increase the angle in counterclockwise direction.
- **power:** The moving power, the range is -100% ~ 100%. When power is positive, the car moves forward, and vice versa, it moves backward.

### 3. Move and Rotate

In this project, the Zeus Car can increase the parameters of rotation so that it can spin in place or go out of arc.

#### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Open the `3_rotate_and_move.ino` file under the path of `zeus-car-main\examples\3_rotate_and_move`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car. The Zeus Car will turn left and then right to return to its original position. Then the car will turn slowly in an outward arc and the turning angle will slowly decrease until it finally rotates on its own.

#### How it works?

Here is the addition of a rotational power parameter `rot` to the Zeus Car's move function `carMove()`.

```
void carMove(int16_t angle, int8_t power, int8_t rot)
```

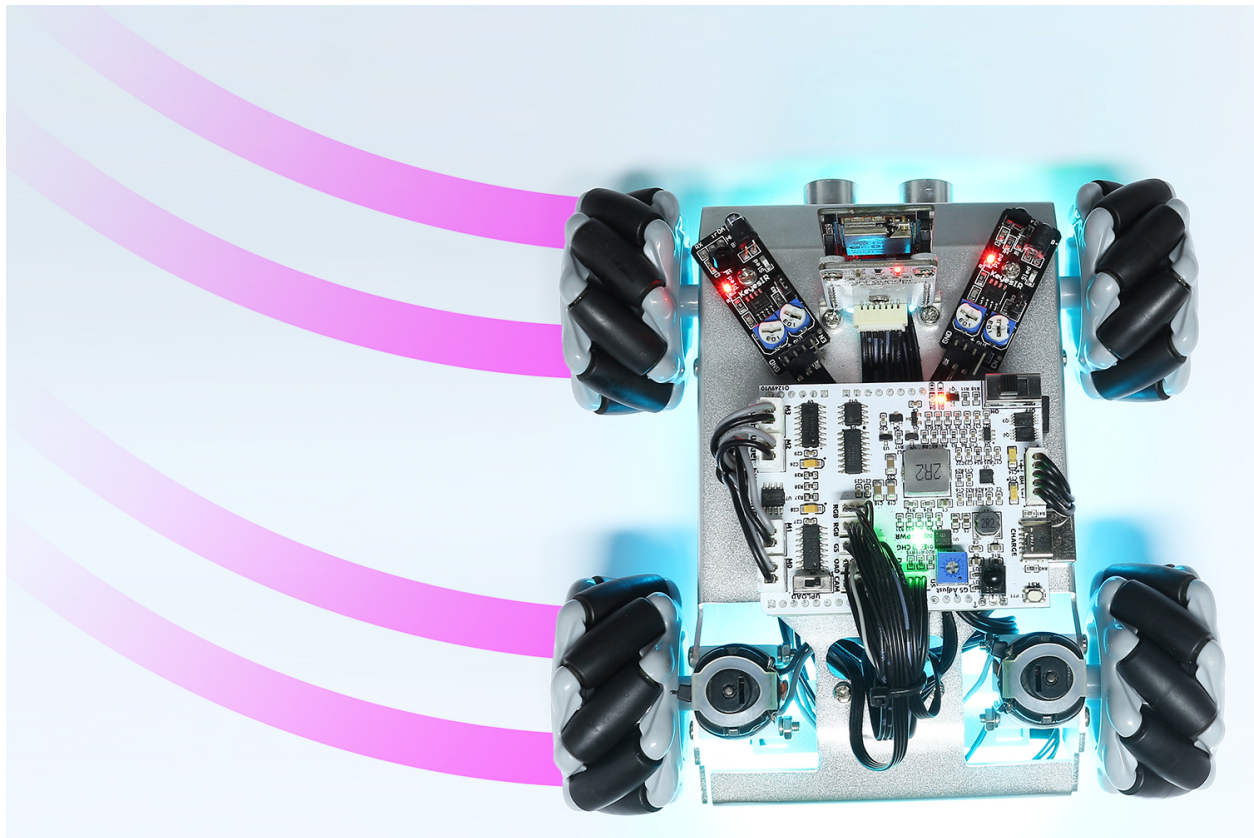
- **angle**: The direction you want the car to move. use the head of the car as the 0 degree and increase the angle in counterclockwise direction.
- **power**: The moving power, the range is -100% ~ 100%. When power is positive, the car moves forward, and vice versa, it moves backward.
- **rot**: Rotation power, the range is -100% ~ 100%. When rot is positive, the car turn counterclockwise, and vice versa.

If power is 0 and rot is not 0, the Zeus Car will spin in place. The higher the rotational power, the faster the rotation speed. When rot is positive, the car will rotate counterclockwise, and vice versa.

If power is not 0, you will find that the Zeus Car will go out in an arc. The turning angle will increase as rot increases, and the Zeus Car will rotate itself when the value is large enough.

### 4. Drift

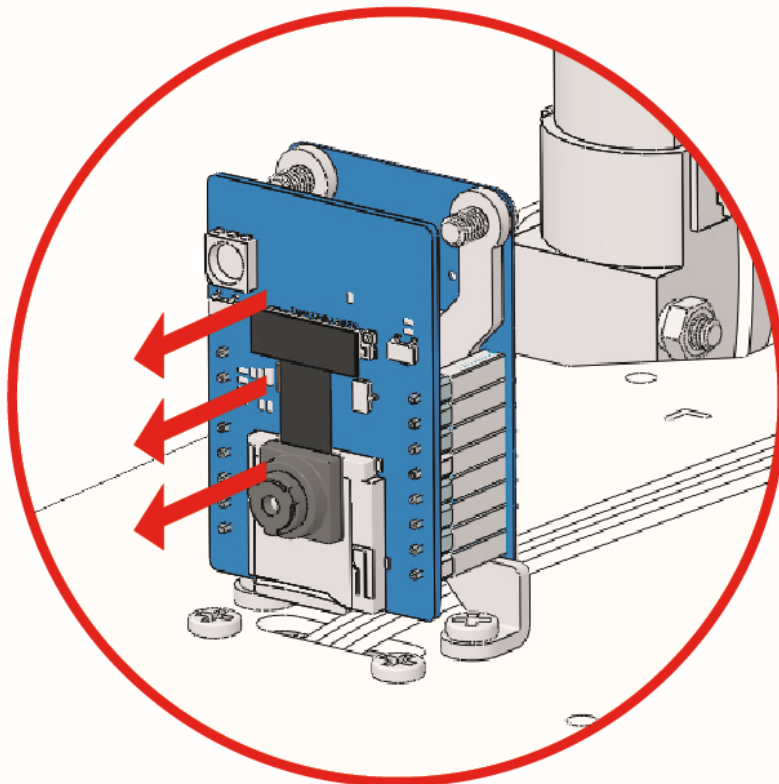
In this project, you will learn how to make the Zeus Car drift.



#### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.





2. Open the `4_drift.ino` file under the path of `zeus-car-main\examples\4_drift`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.
4. Zeus Car will now move forward for 0.5 seconds, then drift 90 degrees left, and so forth. You can try changing 45 (rot) to another value or a negative number to see how the drift angle and direction change.

#### How it works?

The drift of the Zeus Car is achieved by adding the drift parameter `drift` to the move function `carMove()`.

```
void carMove(int16_t angle, int8_t power, int8_t rot, bool drift)
```

- **angle**: The direction you want the car to move. use the head of the car as the 0 degree and increase the angle in counterclockwise direction.
- **power**: The moving power, the range is -100% ~ 100%. When power is positive, the car moves forward, and vice versa, it moves backward.
- **rot**: Rotation power, the range is -100% ~ 100%. When rot is positive, the car turn counterclockwise, and vice versa.
- **drift**: Default is false, when it is true, drift mode is enabled.

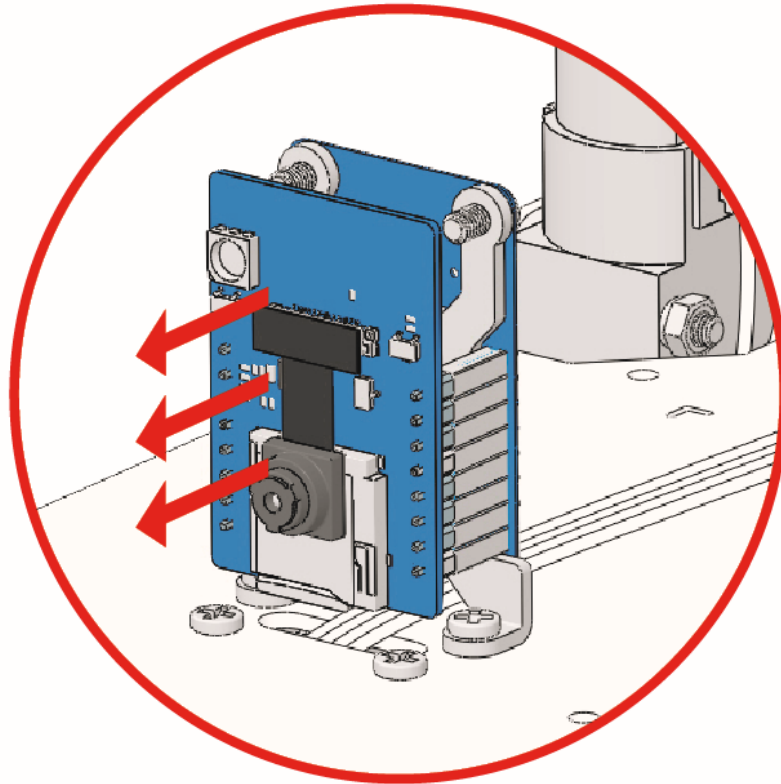
So the different power and rot values will change the drift angle and direction of the Zeus Car.

## 5. Remote Control

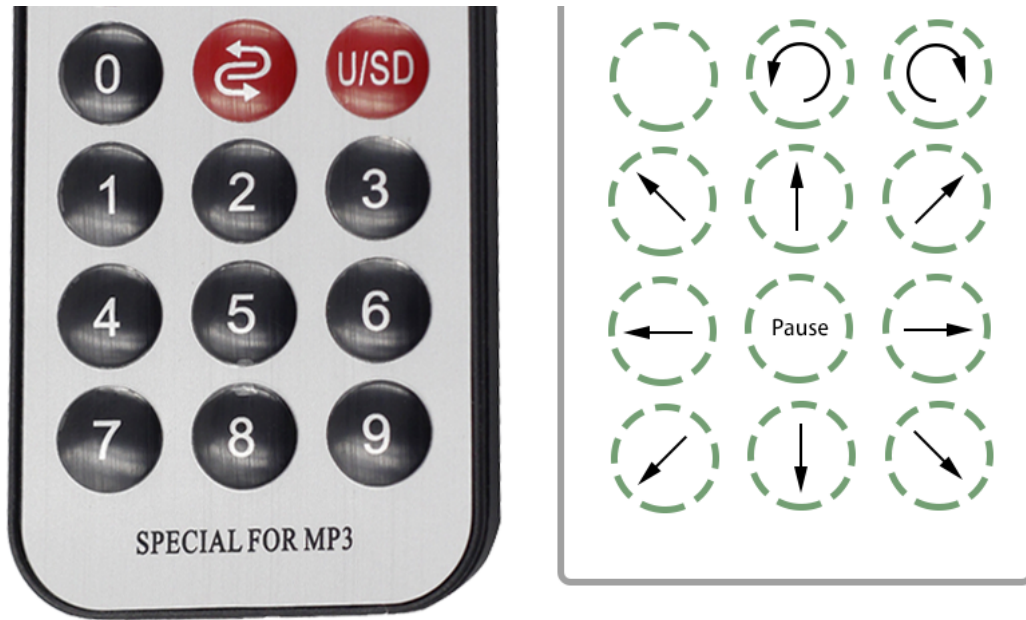
In this project, you will learn how to control the Zeus Car with the remote control.


### How to do?

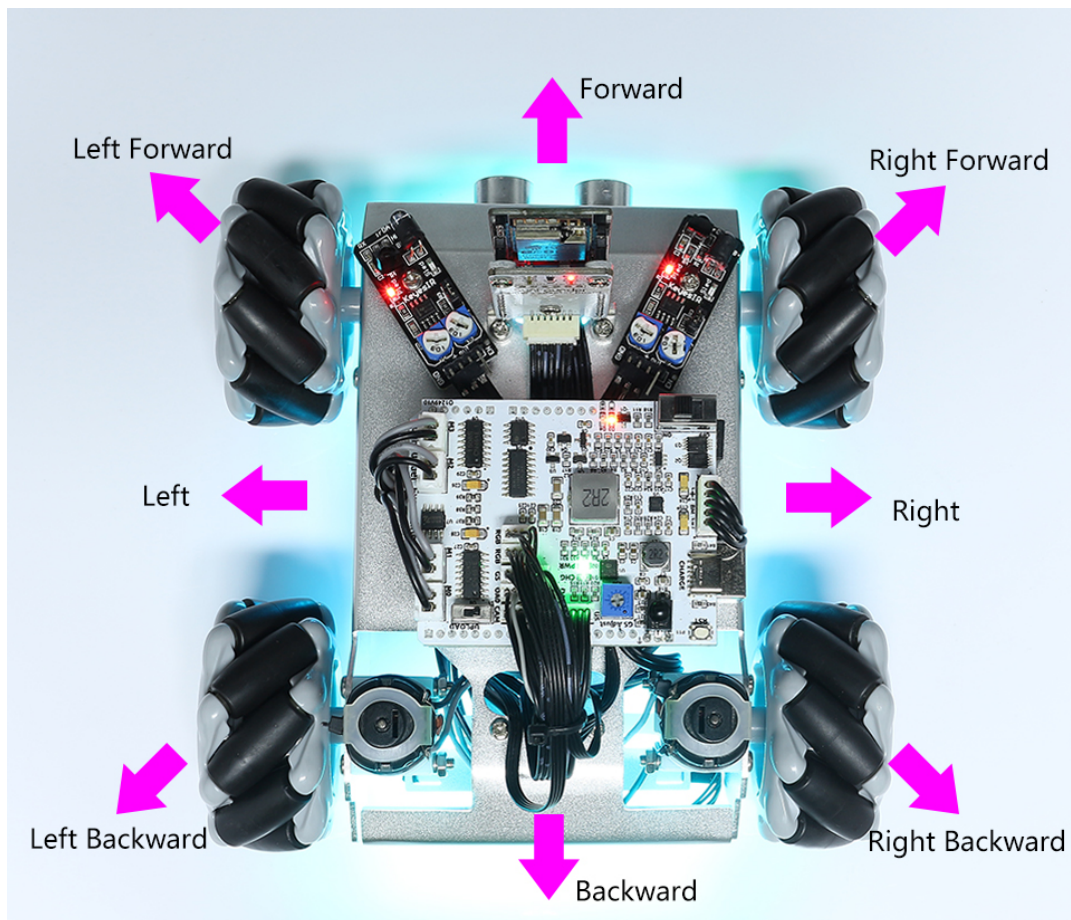
1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.





2. Open the `5_remote_control.ino` file under the path of `zeus-car-main\examples\5_remote_control`.
3. After the code is uploaded successfully, turn the power switch to ON to start the Zeus Car.
4. Then use the 1~9 on the remote control to control the car in 8 directions.

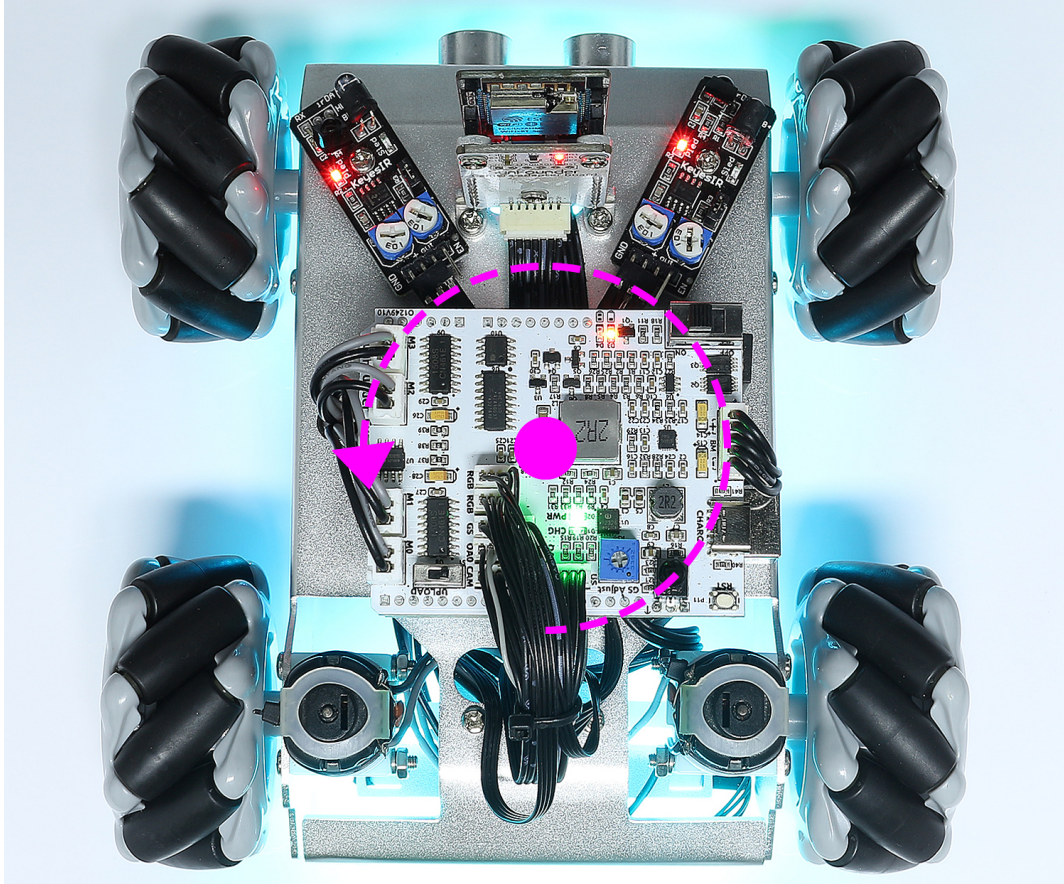




5. After pressing a key, the Zeus Car will keep moving until you press  or the number key 5.



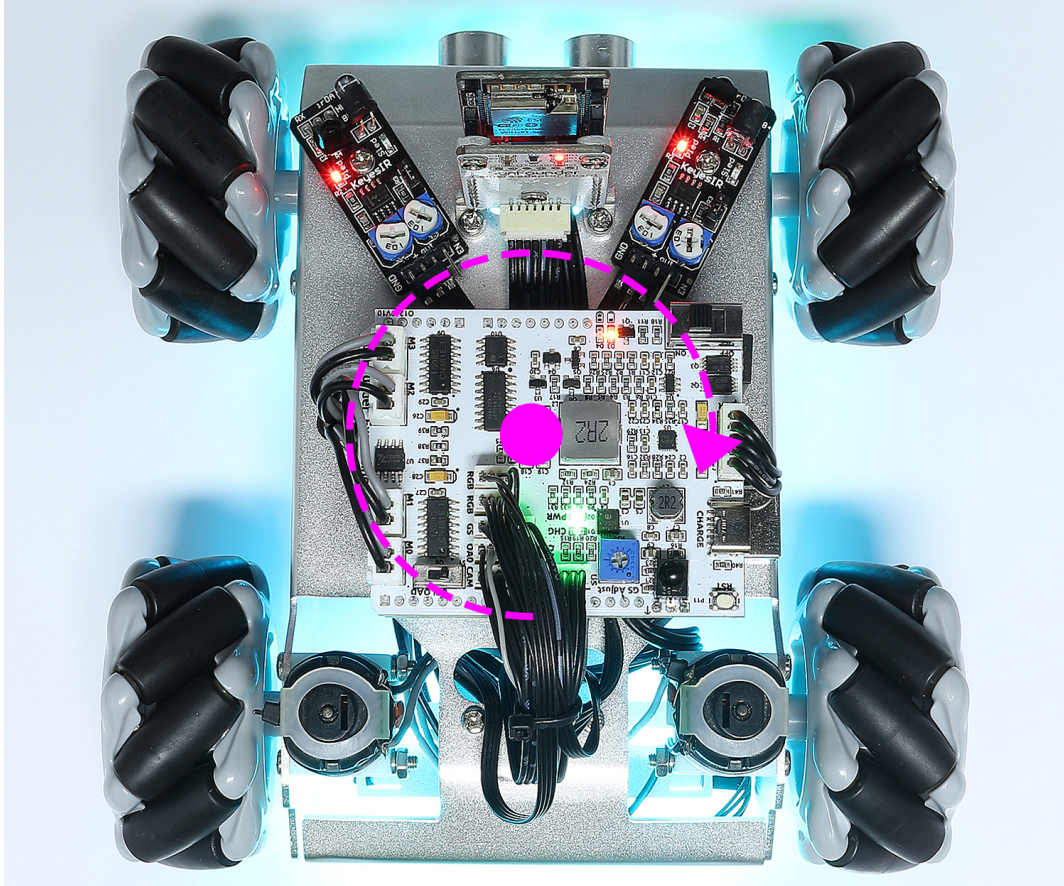


6. When you press  once, the car will rotate counterclockwise with the body as the center and will stop until you press  or the number key 5.



7. Similarly, pressing  once will make the car rotate clockwise, and then it will stop until you press  or the number key 5.



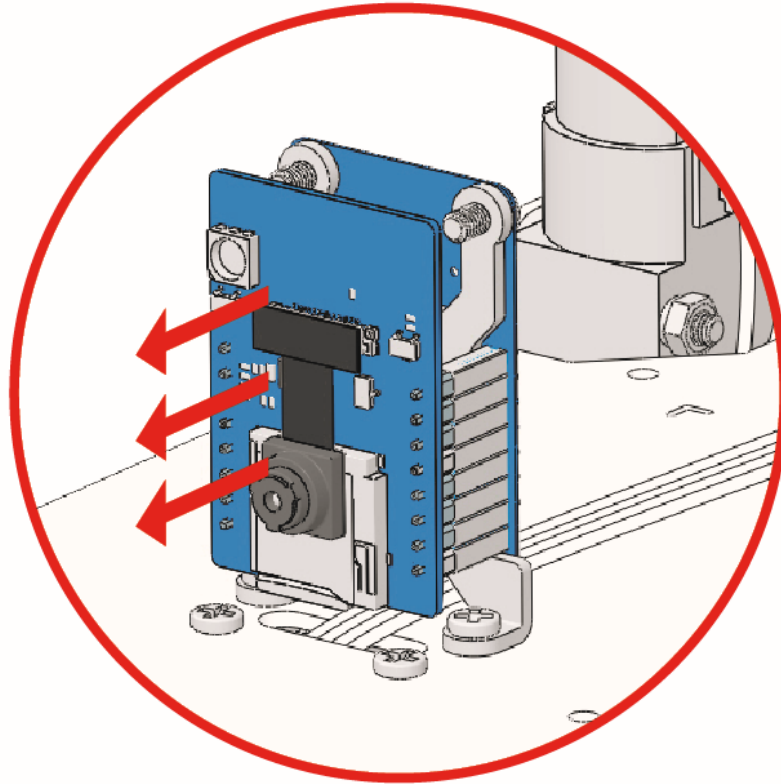


## 6. Car Light


In this project, you will be able to use the remote control to control the RGB light strips on the bottom of the car, making them display different colors.

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Open the `6_car_light.ino` file under the path of `zeus-car-main\examples\6_car_light`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.
4. Now you can use the numbers 1 to 9 on the remote control to control the car to light up different colors, press

0 to turn off, press  to let the car perform a color cycle. Key 1 to 9 corresponding to the color is: red, orange, yellow, green, cyan, blue, purple, pink, white.

## 7. Compass

In this project, Zeus Car will become a compass. You will need to set a heading direction for it, and then no matter how you turn it, it will be facing there.

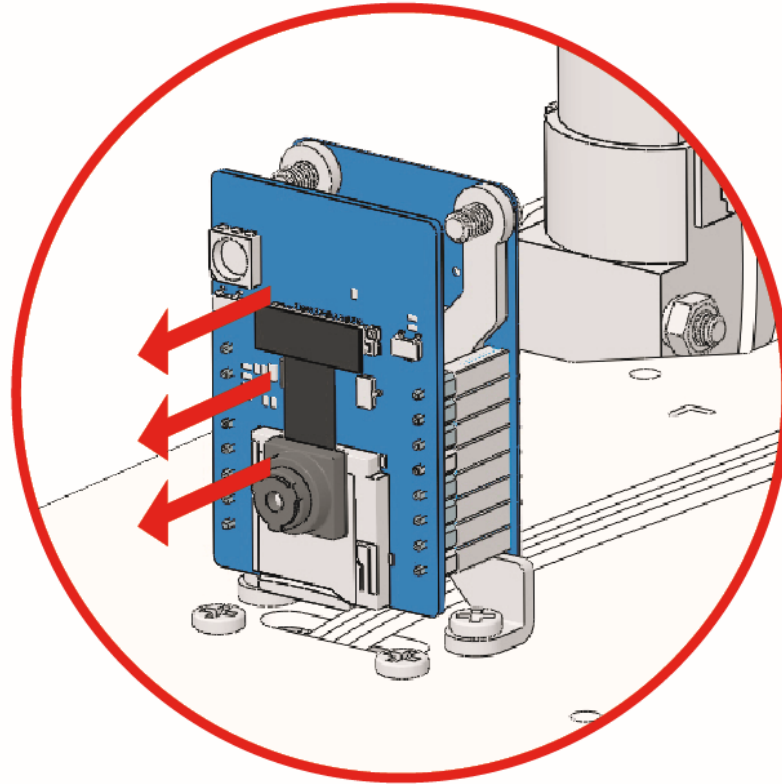
---

**Note:** If electrical wires are present near the Zeus Car, the electromagnetic field will interfere with the operation of the qmc6310 module, thus changing the heading direction of the car.

---

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.





2. Open the `7_compass.ino` file under the path of `zeus-car-main\examples\7_compass`.

3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.

4. First let's calibrate the qmc6310 module.

- Place the car on the ground (take care to stay away from ground with dense wires).

- Press  and the car will rotate in place.
- And stop until the magnetic field at that place is fully recorded.

- If it doesn't stop after more than 2 minutes, you can press  to stop and then continue elsewhere.

5. Turn the car to the direction you want it to face and press . After that, whatever direction you turn it, it will slowly turn back to the set direction.

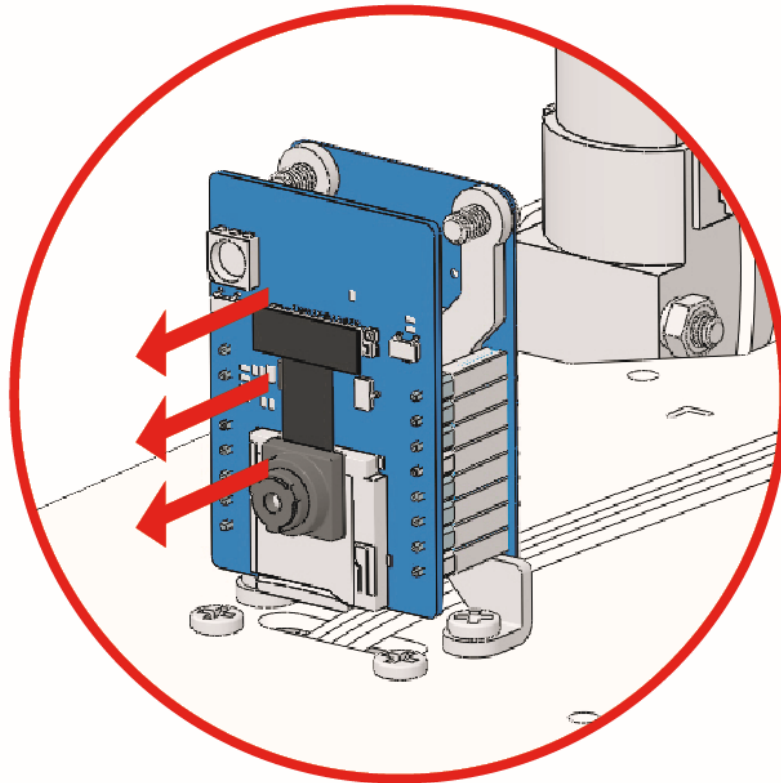
6. The magnetic fields on the ground and on the table are not the same and need to be recalibrated.

## 8. Move Field Centric


This project is based on the previous project to combine the compass function to the movement of Zeus Car. While you control the Zeus Car movement with the remote control, if it is deflected by an external force (e.g. kicked by a dog), it will automatically deflect back to the original direction.

### How to do?

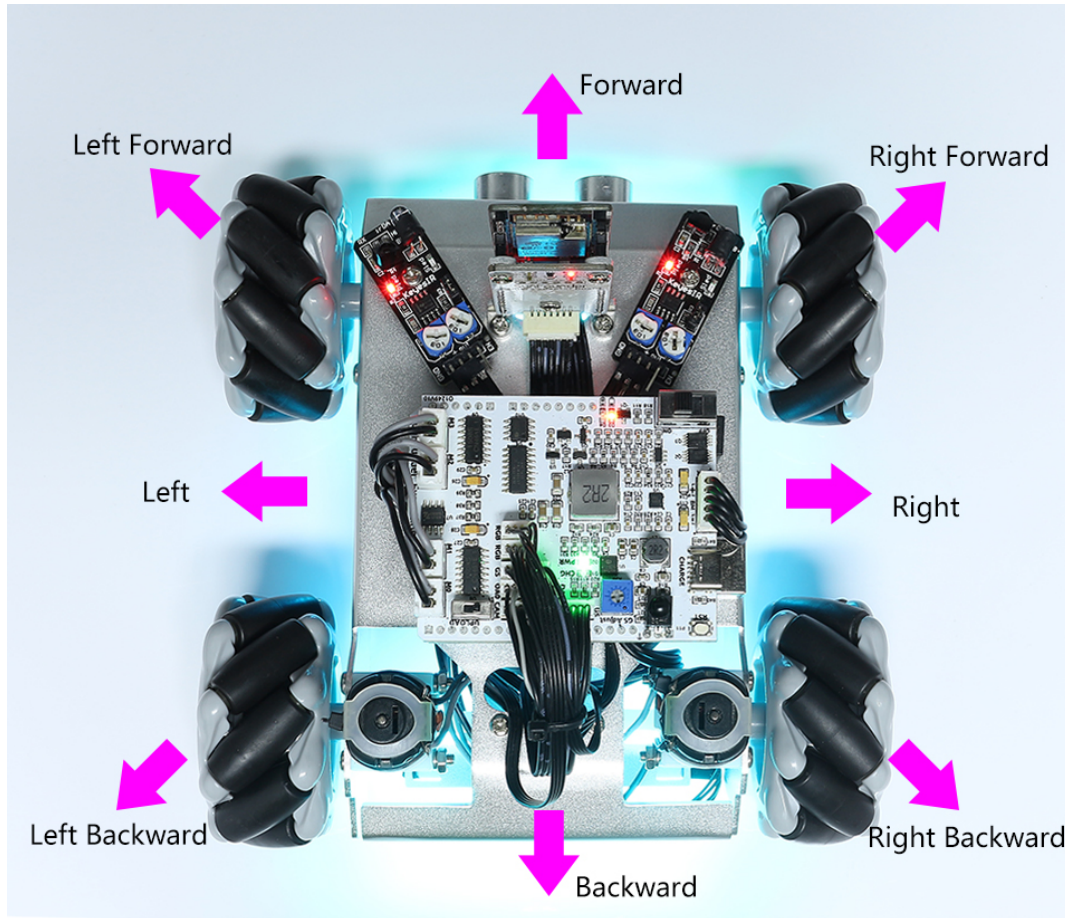
1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.





2. Open the `8_move_field_centric.ino` file under the path of `zeus-car-main\examples\8_move_field_centric`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.
4. Then use the number 1 ~ 9 on the remote control to control the car in 8 directions. After pressing a key, the Zeus

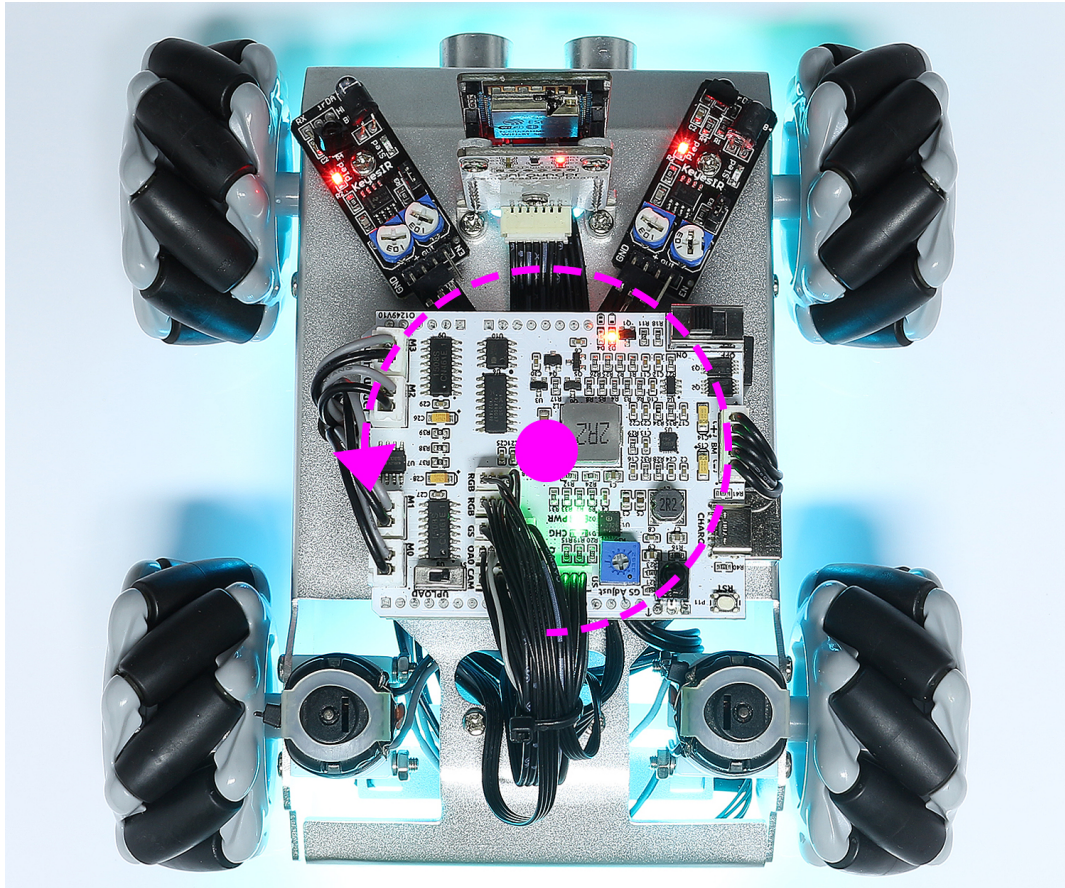
Car will keep moving until you press  or the number key 5.







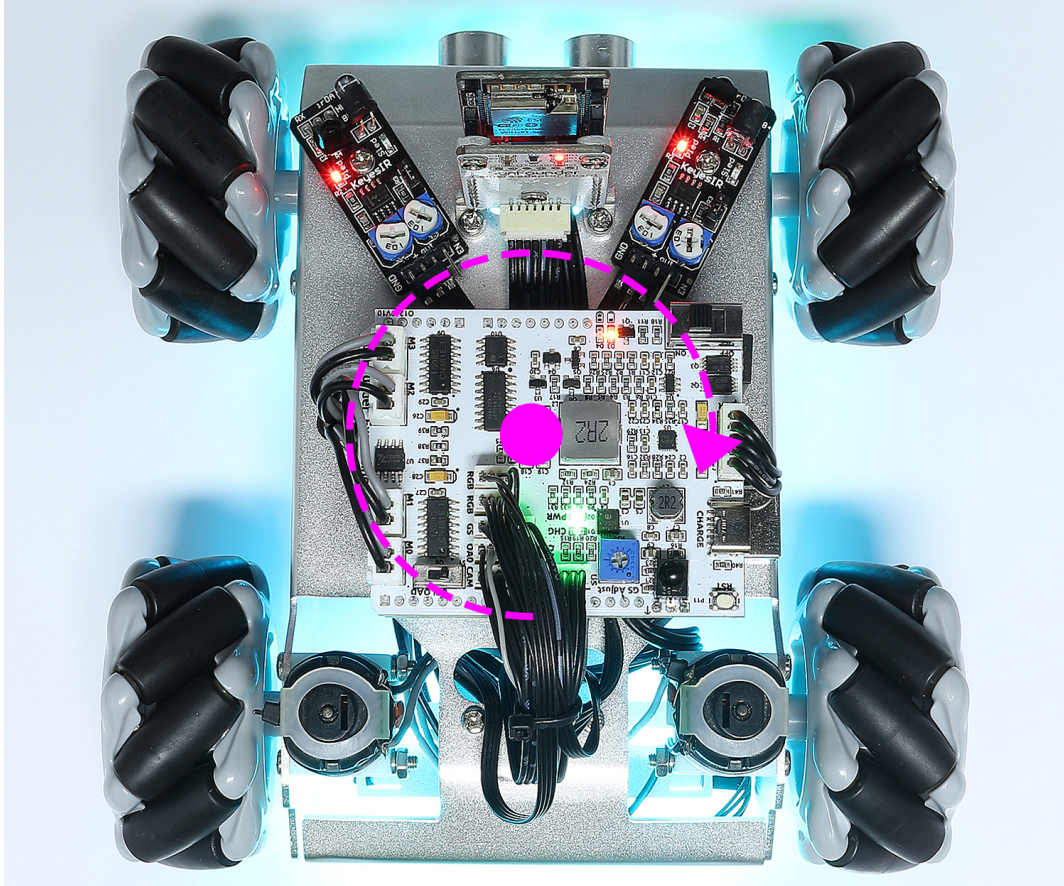
5. When you press  once, the car will rotate counterclockwise with the body as the center and will stop until you press  or the number key 5.







6. Similarly, pressing  once will make the car rotate clockwise, and then it will stop until you press  or the number key 5.

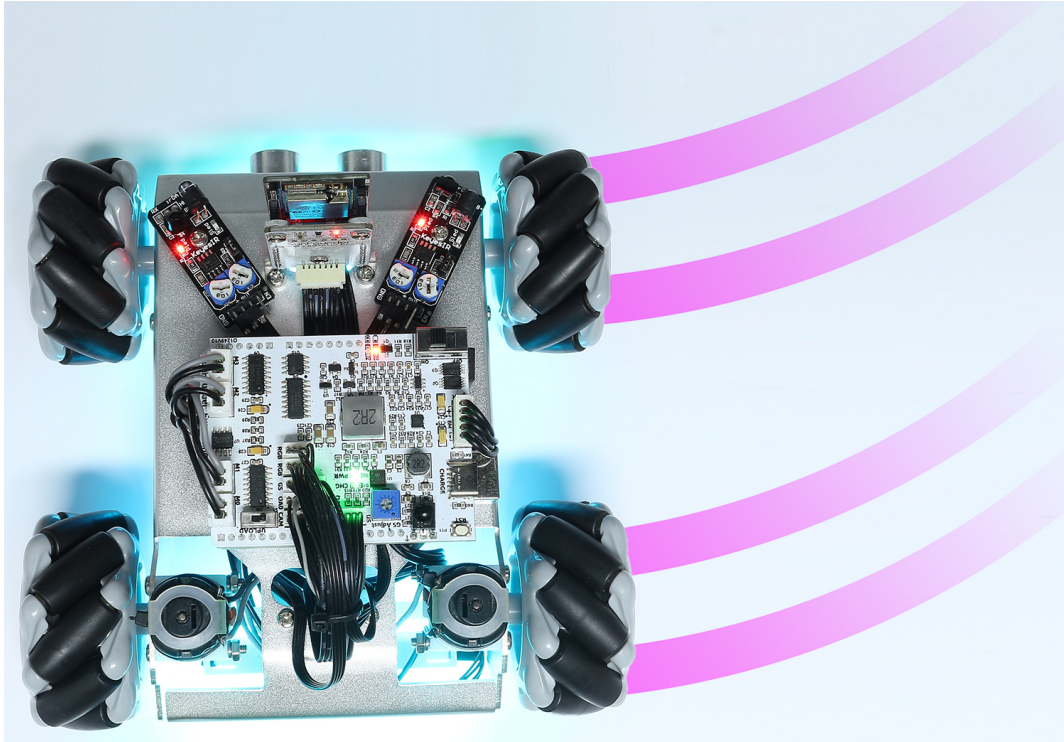




7. Press the  key, the car will drift to the left.




8. Press the  key, the car will drift to the right.



---

**Note:**

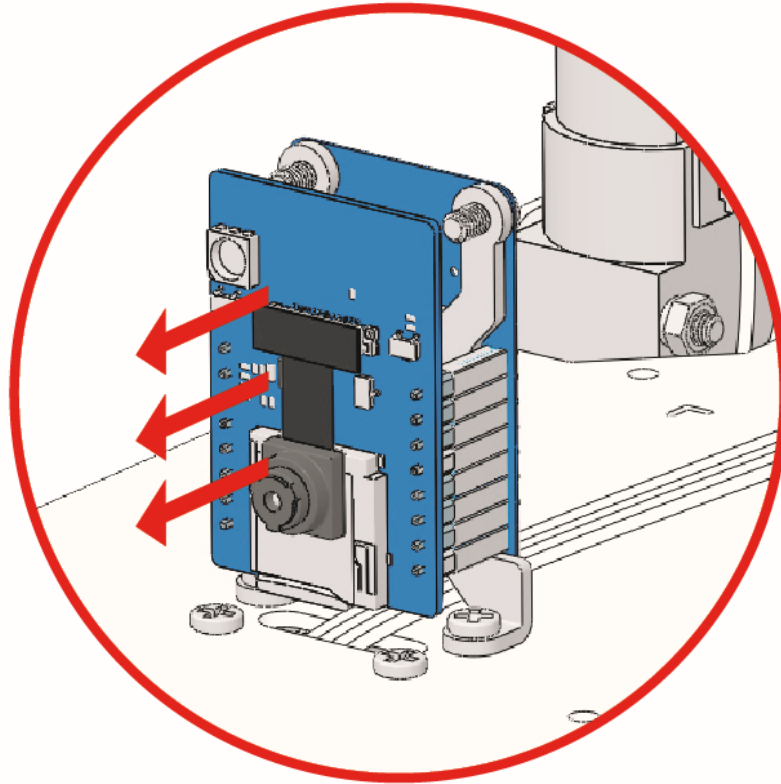
- If the Zeus Car does not move well, the compass may not be calibrated properly, you need to press  to calibrate it.
  - If you pick up the car from the ground to the table, the magnetic field will change and you need to recalibrate it.
- 

## 9. IR Obstacle

In this project, you will learn how to use the IR obstacle avoidance modules on both sides of the Zeus Car.

**How to do?**

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Open the `9_hc165_and_ir_obstacle.ino` file under the path of `zeus-car-main\examples\9_hc165_and_ir_obstacle`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.

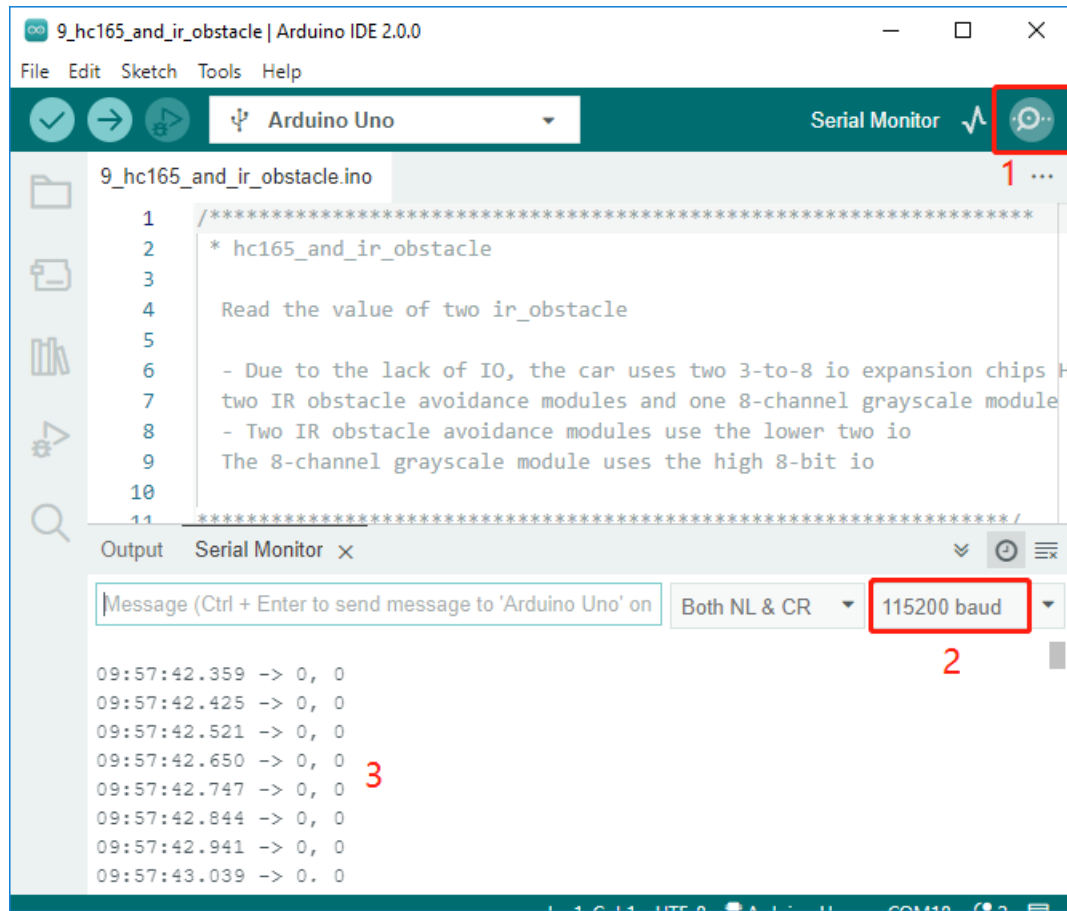
---

**Note:** Do not unplug the USB in this step, because you need to check the data of the two obstacle avoidance modules on your computer.

---

4. Open the serial monitor and make sure the current baud rate is set to 115200, then you can view the printed data.
  - If both obstacle avoidance modules do not detect an obstacle, the serial monitor will print 0, 0.
  - Put your hand in front of one of the obstacle avoidance modules, it will print 1, 0 or 0, 1.

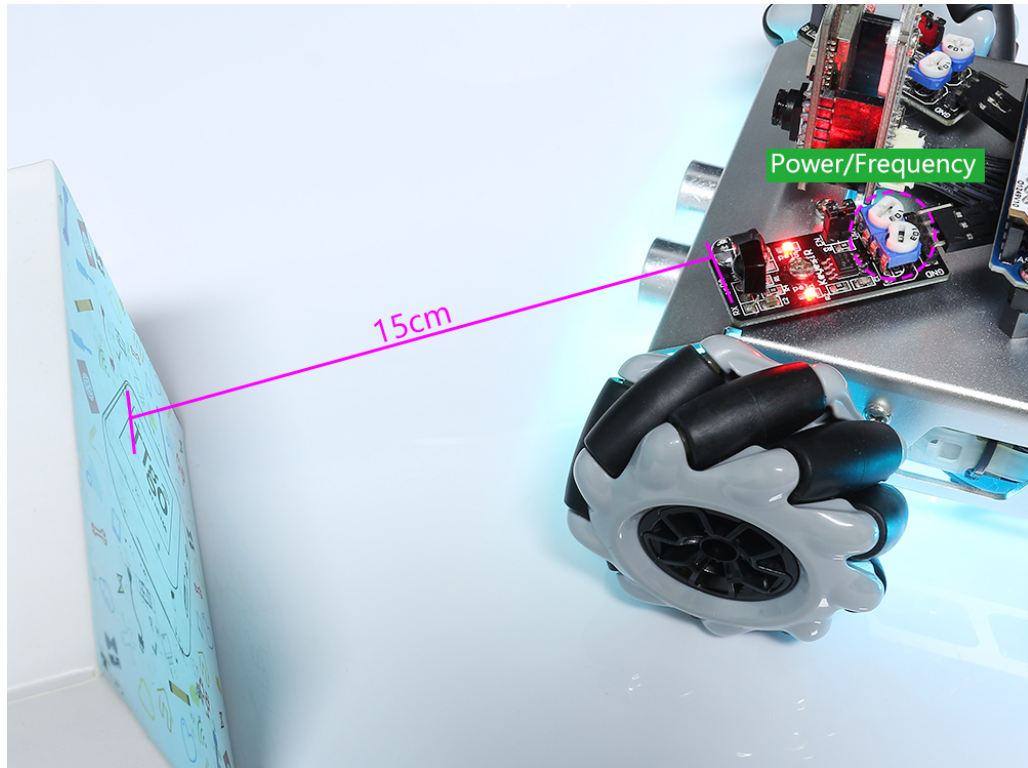




##### 5. Calibrate the IR obstacle avoidance module.

- Start by adjusting the right obstacle avoidance module. During transportation, collisions may cause the transmitter and receiver on the infrared module to tilt. Therefore, you need to manually straighten them.
- Place an obstacle about 15cm away from the IR obstacle avoidance module.
- On the module are two potentiometers, one to adjust the sending power and one to adjust the sending frequency. By adjusting these two potentiometers, you can adjust the detection distance.
- Then you can adjust a potentiometer, and if at 15cm, the signal light on the module illuminates, the adjustment is successful; if it doesn't, adjust another potentiometer.





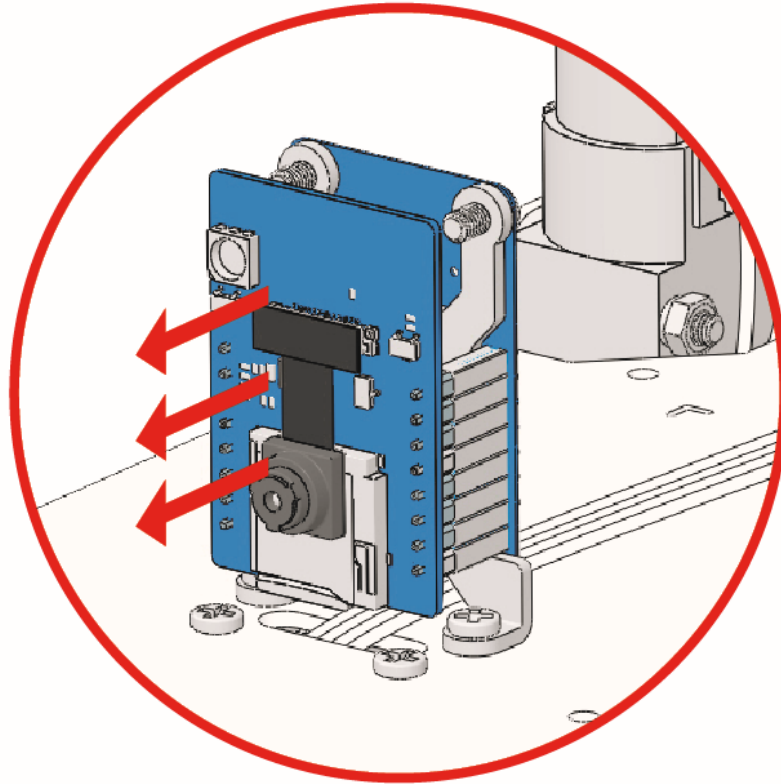
- Calibrate the other obstacle avoidance module in the same way.

## 10. Ultrasonic

In this project, you will learn how to read the distance detected by the ultrasonic module.

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



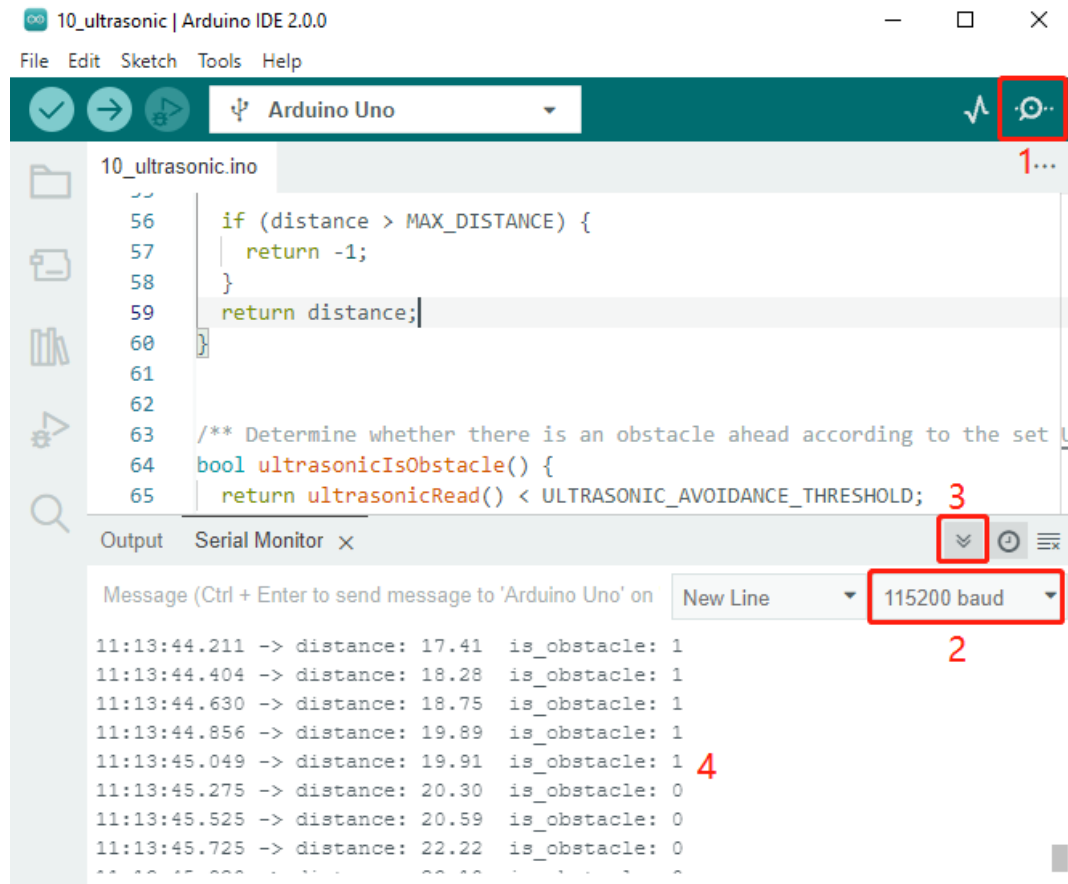
2. Open the `10_ultrasonic.ino` file under the path of `zeus-car-main\examples\10_ultrasonic`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.

---

**Note:** Do not unplug the USB in this step, because you need to check the data of the Ultrasonic module on your computer.

---

4. Open the serial monitor and make sure the current baud rate is set to 115200. It is recommended to click on the **Toggle Autoscroll** icon so that you can see the latest printed data.
  - You can view the printed data like `distance: 21.11 is_obstacle: 0`.
  - If the distance of the obstacle ahead is within 20cm, `is_obstacle: 0` will become `is_obstacle: 1`.

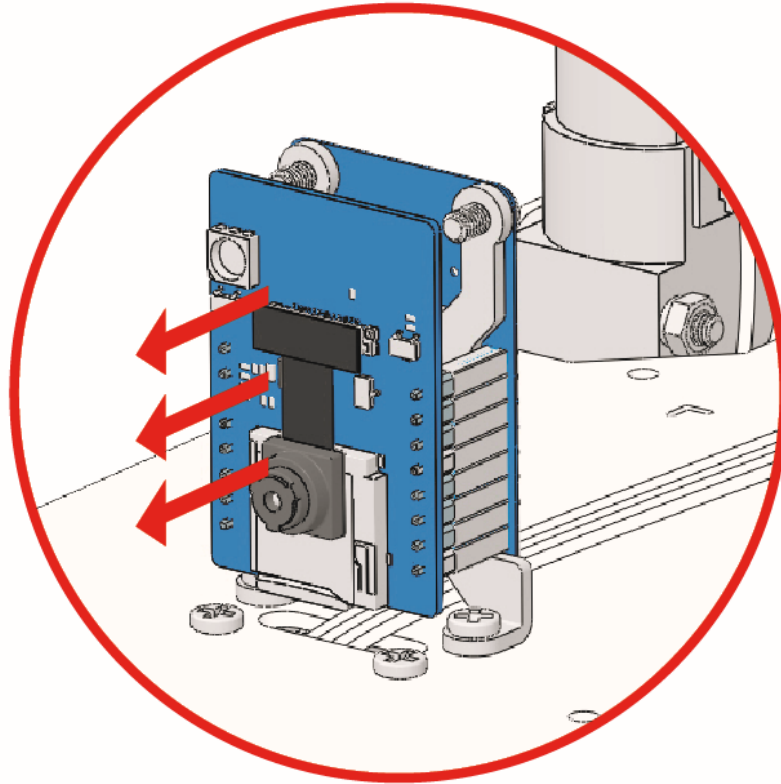


## 11. Obstacle Avoidance

In this project, the Zeus car will move forward automatically, and two obstacle avoidance modules and an ultrasonic module will prevent it from hitting obstacles.

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.

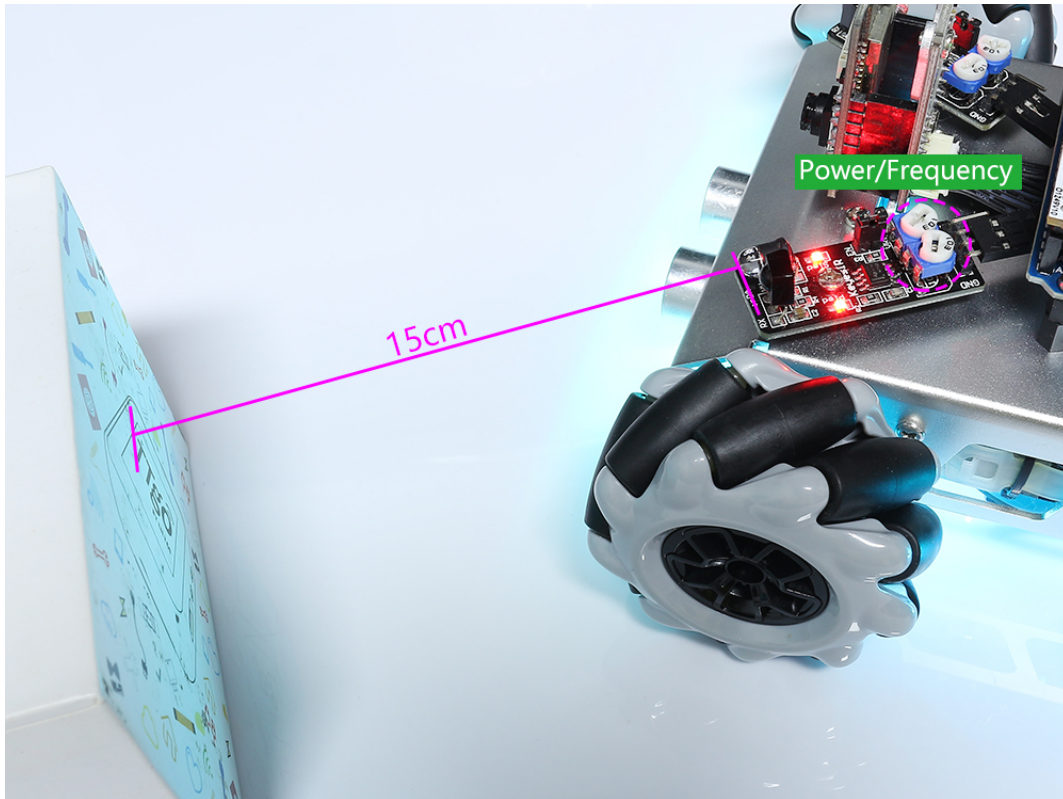


2. Open the `11_obstacle_avoid.ino` file under the path of `zeus-car-main\examples\11_obstacle_avoid`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.
  - Zeus car will move forward.
  - An ultrasonic module detects obstacles in front, if detected, the car turns left.
  - When the left obstacle avoidance module detects an obstacle, the car turns right, and when the right obstacle avoidance module detects an obstacle, the car turns left.

---

**Note:** Before use, you need to adjust the detection distance of the two obstacle avoidance modules to 15CM, the steps are as follows

- Start by adjusting the right obstacle avoidance module. During transportation, collisions may cause the transmitter and receiver on the infrared module to tilt. Therefore, you need to manually straighten them.
- Place an obstacle about 15cm away from the IR obstacle avoidance module.
- On the module are two potentiometers, one to adjust the sending power and one to adjust the sending frequency. By adjusting these two potentiometers, you can adjust the detection distance.
- Then you can adjust a potentiometer, and if at 15cm, the signal light on the module illuminates, the adjustment is successful; if it doesn't, adjust another potentiometer.



- Calibrate the other obstacle avoidance module in the same way.
- 

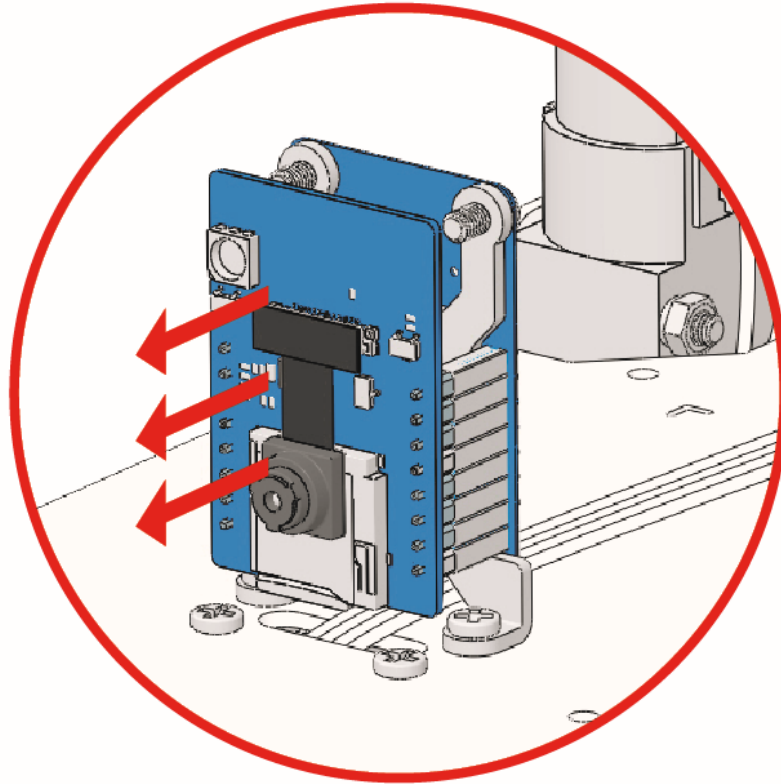
## 12. Follow

In this project, the same two obstacle avoidance modules and an ultrasonic module will be used, but not for obstacle avoidance, but to follow you.

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.





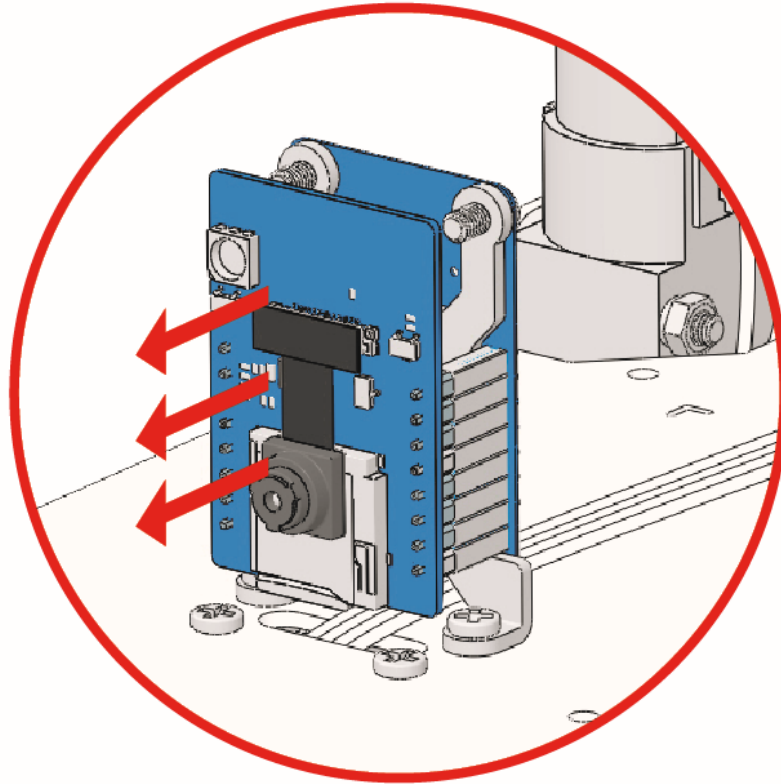
2. Open the `12_follow.ino` file under the path of `zeus-car-main\examples\12_follow`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.
4. In this case, the Zeus Car will not move if there is nothing in front of it. Whenever you move forward, it will follow you if you stand about 20cm away from it.

### 13. Grayscale

In this project, you will learn how to read and calibrate the Omni Grayscale Module on the bottom of the Zeus Car.

#### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



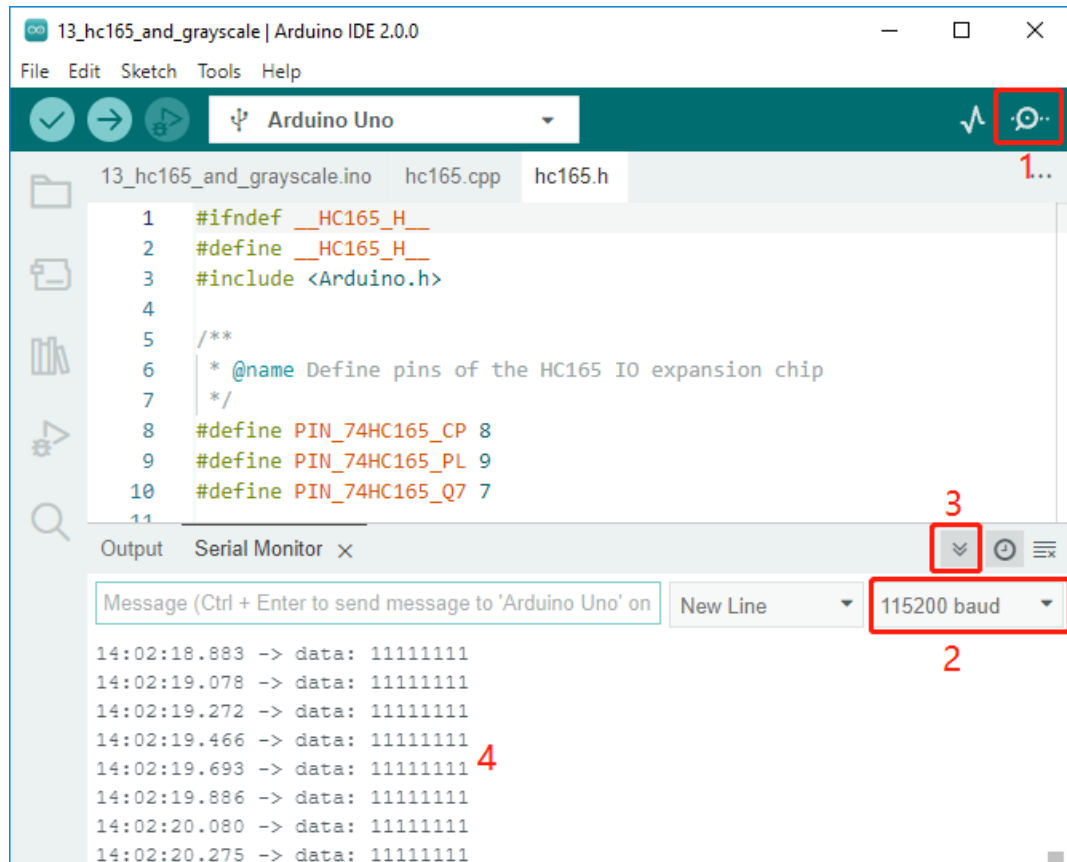
2. Open the `13_hc165_and_grayscale.ino` file under the path of `zeus-car-main\examples\13_hc165_and_grayscale`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.

---

**Note:** Do not unplug the USB in this step, because you need to check the data of the Omni Grayscale Module on your computer.

---

4. Open the serial monitor and make sure the current baud rate is set to 115200. It is recommended to click on the **Toggle Autoscroll** icon so that you can see the latest printed data.

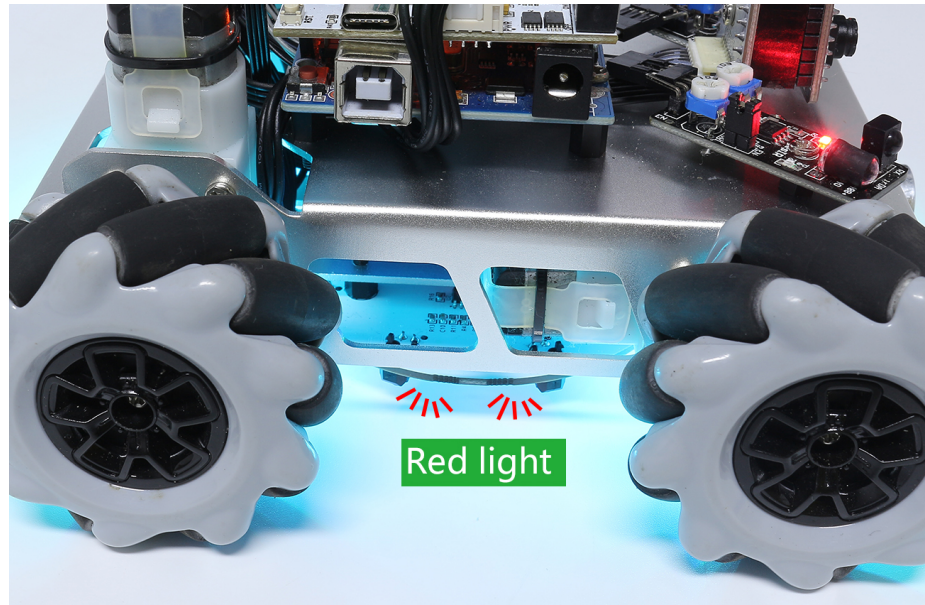


- Turn the car over and you will be able to see the Serial Monitor print out data: 11111111.
- If you cover one of the probes on the module with your hand, you will see the value in the corresponding position change to 0.
- For example, if you cover the U11 probe, you will see data: 01111111.

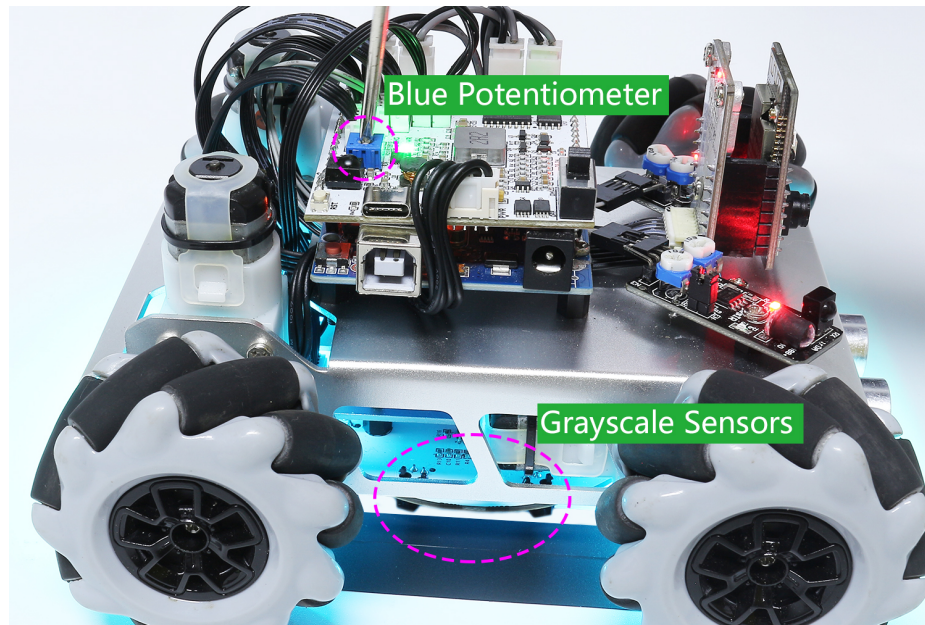
##### 5. Calibrate the Omni Grayscale module.

Since each subfloor has different grayscale values, the factory-set grayscale threshold may not be appropriate for your current environment, so you will need to calibrate this module before use. It is recommended that you need to calibrate it whenever the floor color changes a lot.

- Place the Zeus Car on white surface and turn the potentiometer until the gray sensor light is just illuminated.



- Now let the two greyscale sensors on the side be located just between the black line and white surface, and slowly turn the potentiometer until the signal indicator just goes off.



- You can move repeatedly over the the black line and white surface to make sure that the lights of the greyscale sensor are off when they are between the the black line and white surface and on when they are on the white surface, indicating that the module is successfully calibrated.



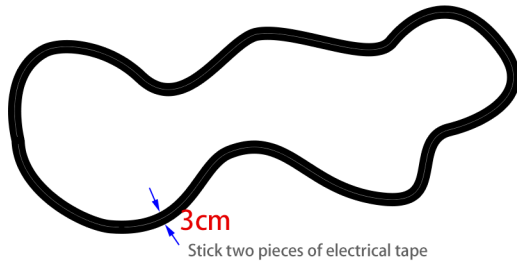
## 14. Line Track

In this project, you will learn how to use the Omni Grayscale Module for line tracking.

Before working on the project, you will need to use black electrical tape to stick out a line of track, which can be a circle, a straight line, or an irregular shape.

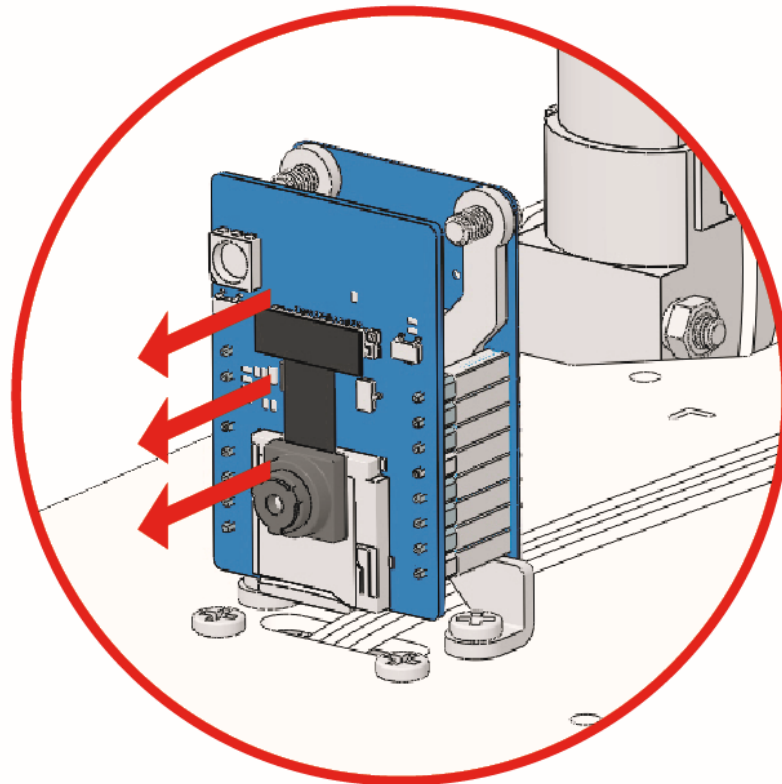
But there are two points to note.

1. This line should be 3cm wide (the thickness of a piece of electrical tape is 1.5cm).
2. The bend angle should not be less than 90°.



### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Open the 14\_line\_track.ino file under the path of zeus-car-main\examples\14\_line\_track.

3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.

4. Next your Zeus Car will move along the line. If your Zeus Car goes off the line, it may mean that you need to recalibrate the [13. Grayscale](#), or turn down its speed.

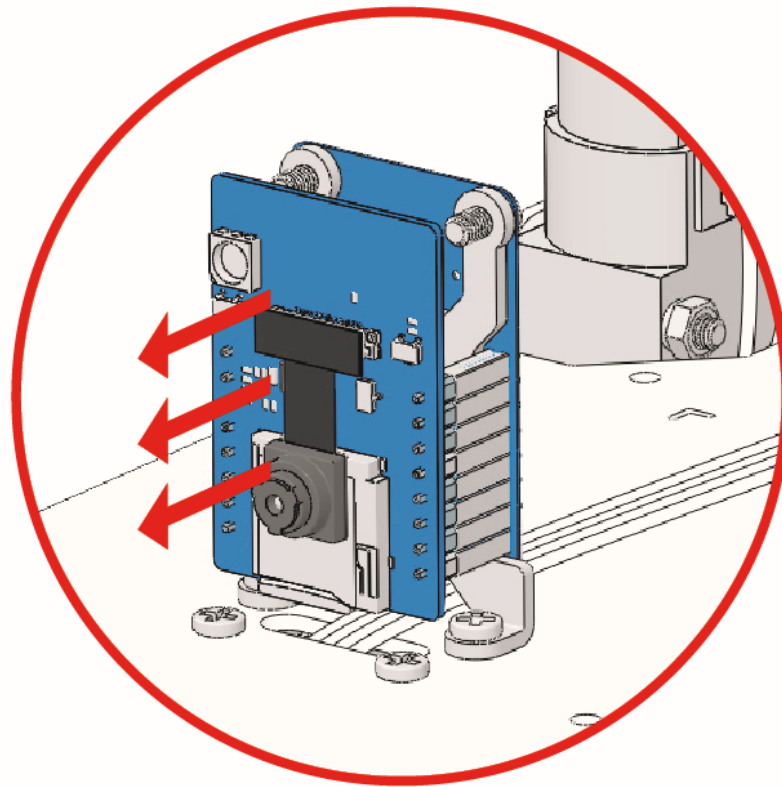
## 15. Line Track Field Centric

A different type of tracer mode will be explored in this project - tracer movement with compass.

The experimental result is almost the same as [14. Line Track](#), however the head of the Zeus Car is always facing a fixed direction; in the previous project it changed with the line direction.

### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Open the `15_line_track_field_centric.ino` file under the path of `zeus-car-main\examples\15_line_track_field_centric`.
3. After the code is uploaded successfully, slide the power switch to ON to start the Zeus Car.
4. Next, your Zeus car will move along the line, but with its head facing in one direction. If your Zeus Car goes off the line, it may mean that you need to recalibrate the [13. Grayscale](#), or turn down its speed.

### 16. AI Detection from APP

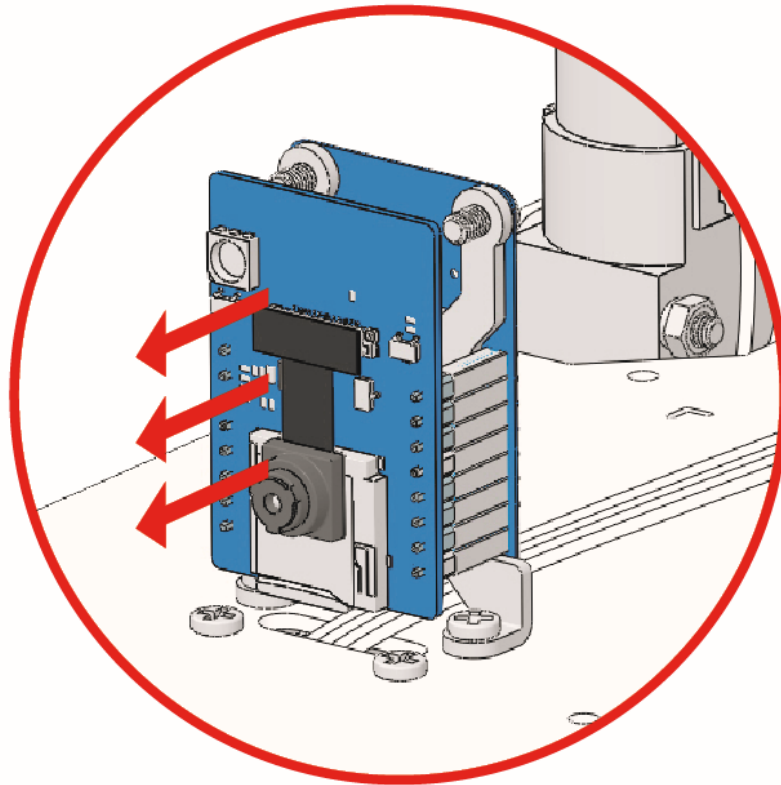
In this project, we will use an app - SunFounder Controller to view real-time video streaming, as well as use the AI detection features on your mobile device, such as face and pose detection.

SunFounder Controller is an application that allows users to customize the controller for controlling their robot or as an IoT platform. 11 kinds of show and control widgets are integrated in this APP, such as Button, Joystick, Gauge, and Radar. The controller page has 17 areas from A~Q, you can place different widgets to customize your own controller.

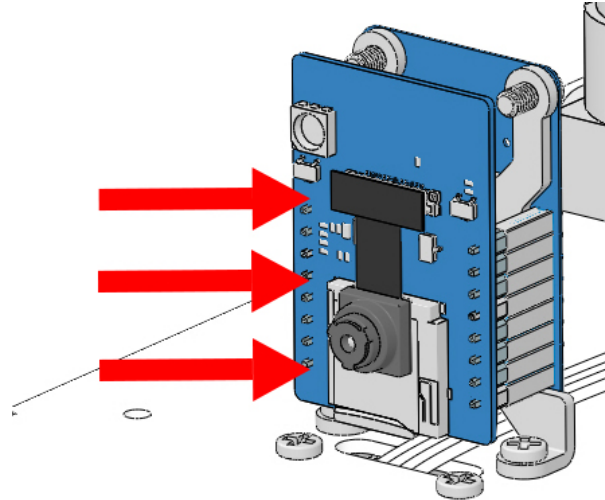
Now take a look at how to use the APP in a simple way.

#### How to do?

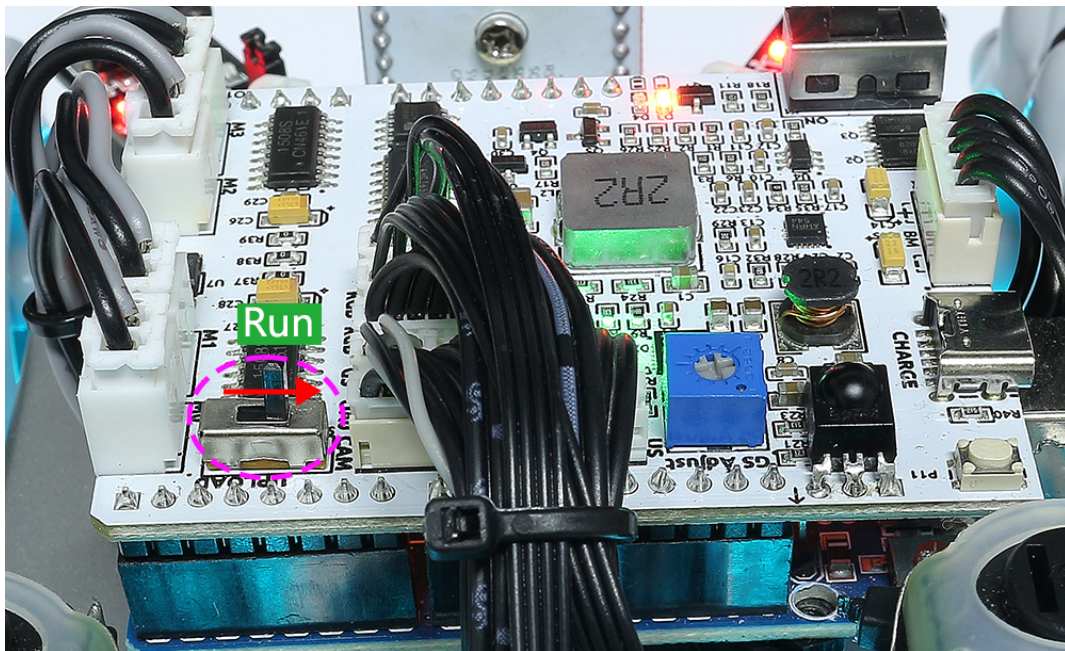
1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



2. Open the `16_ai_detection_from_app.ino` file under the path of `zeus-car-main\examples\16_ai_detection_from_app`.
3. After the code is uploaded successfully, you can plug in the ESP32-CAM and then slide the power switch to ON to start the Zeus Car.

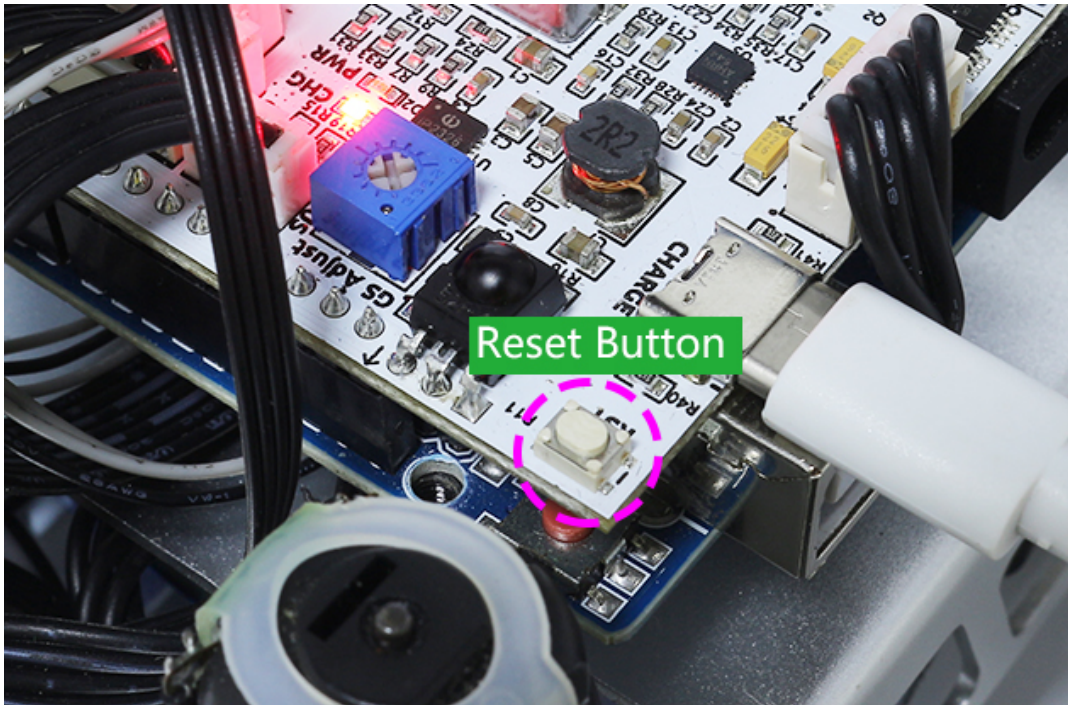


4. Toggle the Upload Switch to the side of Run (right side on this diagram) to start the ESP32 CAM.

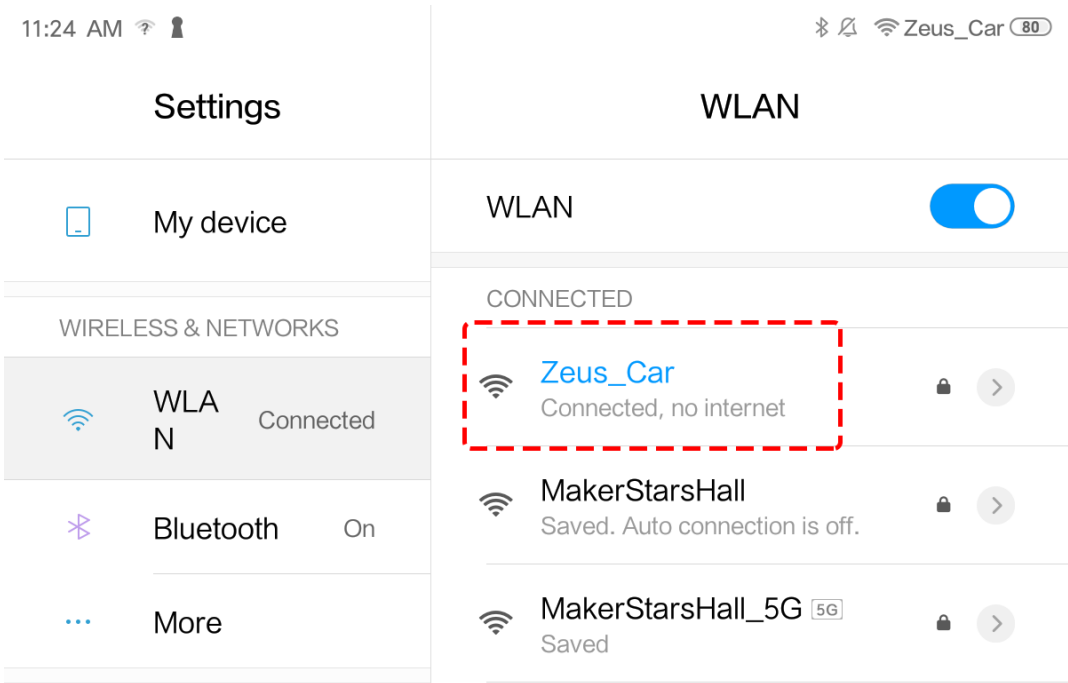


5. Press the reset button to get the Arduino board's program running again.



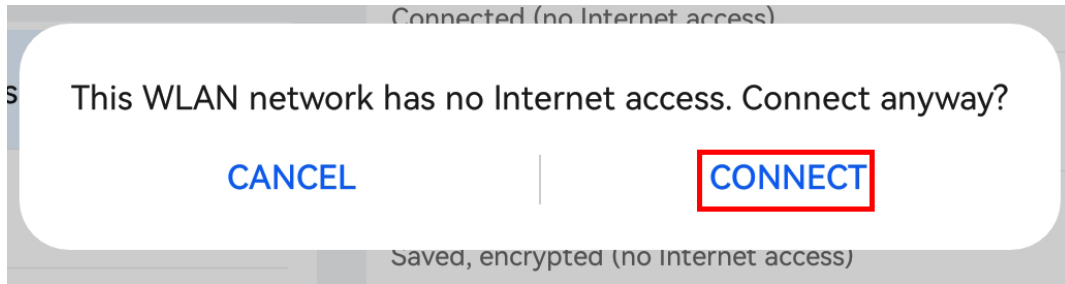


- 6. Now, install [SunFounder Controller](#) from **APP Store(iOS)** or **Google Play(Android)** to your mobile device.
- 7. Connect to Zeus\_Car WLAN.
  - Find Zeus\_Car on the WLAN of the mobile phone (tablet), enter the password 12345678 and connect to it.



- The default connection mode is AP mode. So after you connect, there will be a prompt telling you that there is no Internet access on this WLAN network, please choose to continue connecting.



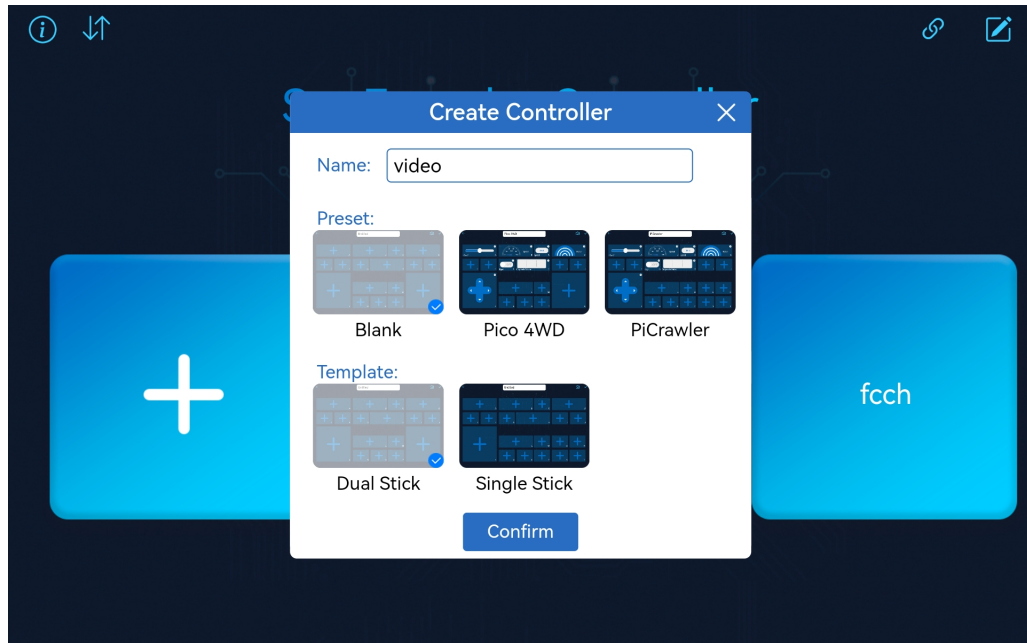


8. Then open the APP and create a controller.

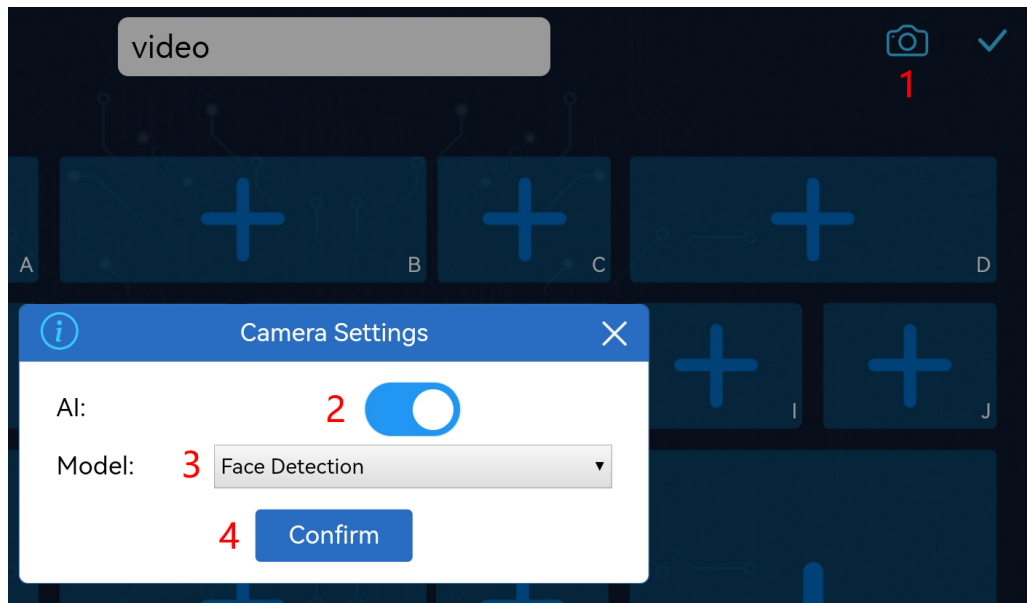
- To add a controller on SunFounder Controller, click the + icon.




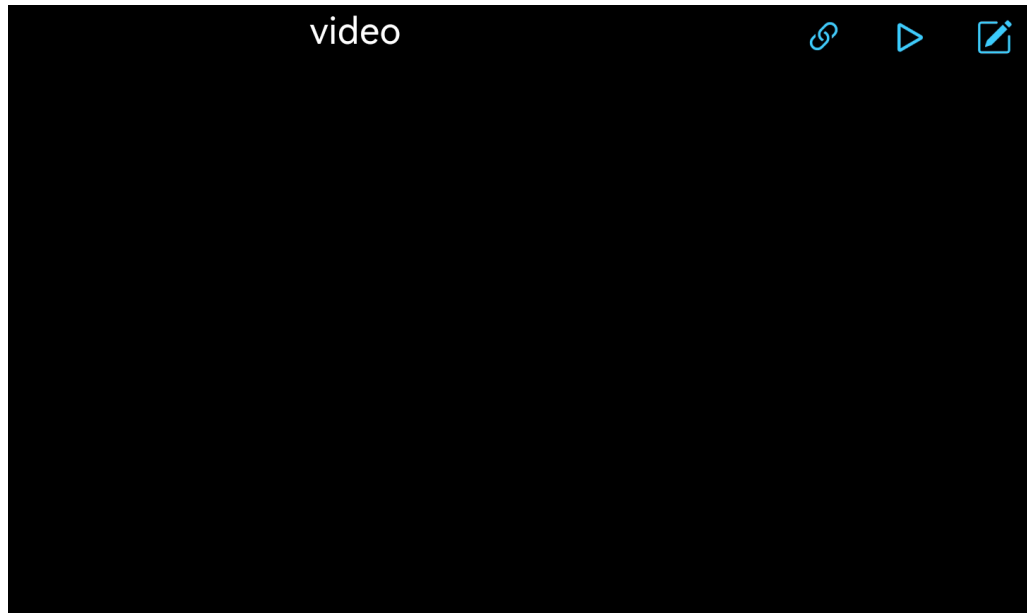
- Select the **Blank** and **Dual Stick** template and give it a name.




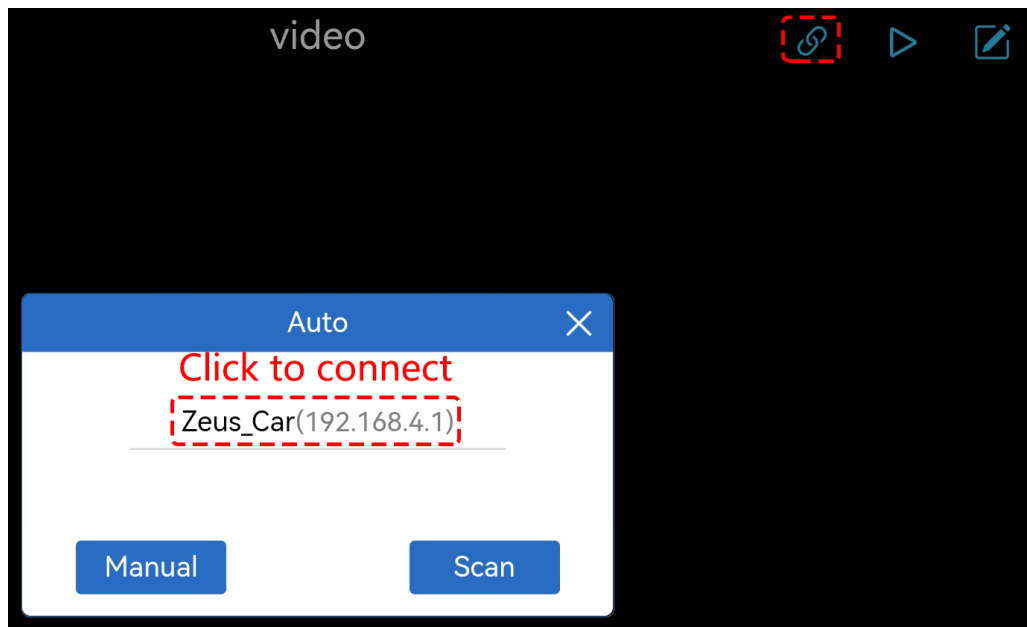
- As you are now inside the controller, click the  icon to enable the AI detection feature. You will find both Face Detection and Pose Detection options.



- Save all settings by clicking on the  button. As no widgets have been selected, the screen is completely black.



- The next step is to connect the Zeus Car to your device via the  button. Wait a few seconds and Zeus\_Car(IP) will appear, click on it to connect.

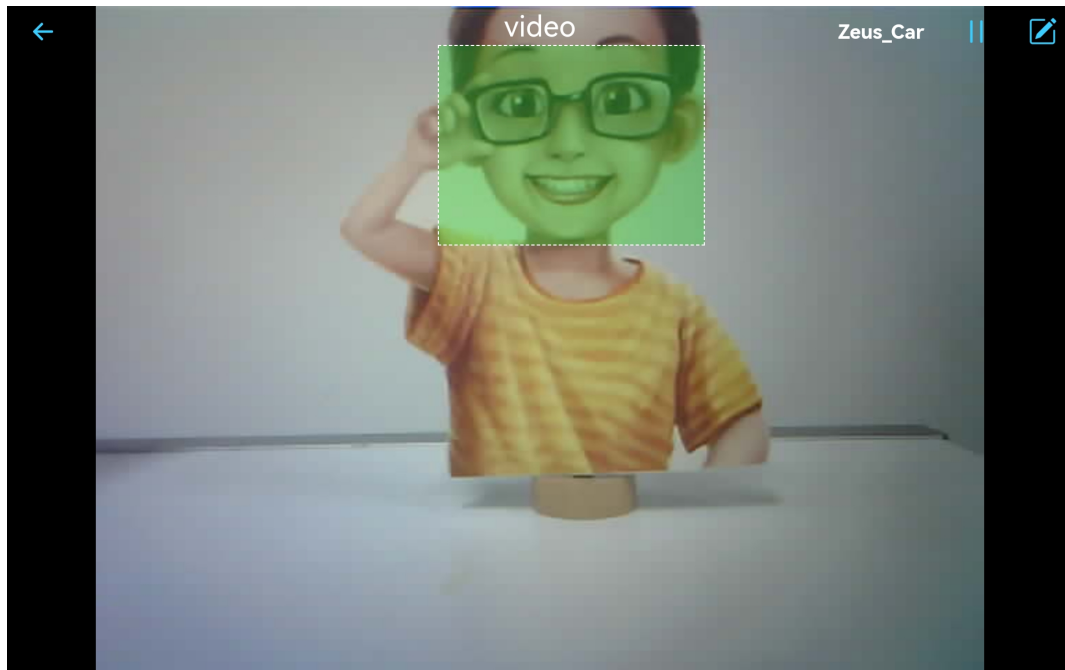


**Note:** Please make sure your Wi-Fi is connected to Zeus\_Car, if you are not seeing the above message for a long time.

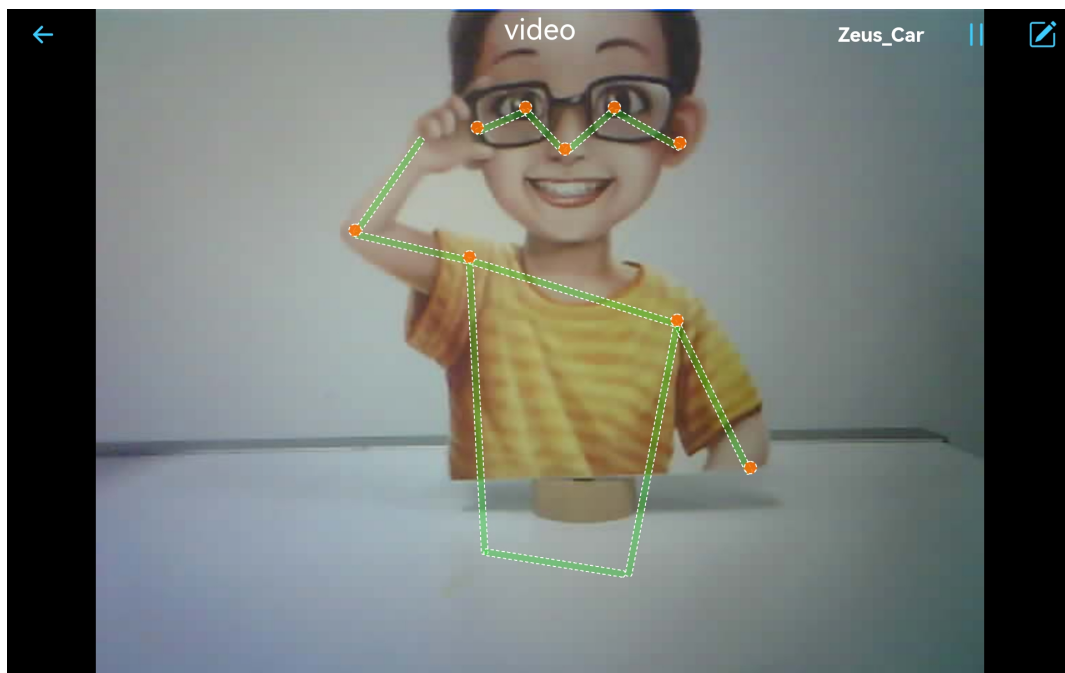
#### 9. Run the Controller.

After the “Connected Successfully” message appears, click the  button, then the camera footage will appear on the app.


If you have Face Detection turned on, then the face that appears in the screen will be framed.



If you have Pose Detection on, then the pose will be depicted.



---

**Note:** If you want to switch to another AI detection, you can click on the  icon and repeat the above steps.

---

## 17. APP Control

In the previous project we simply viewed the footage taken by the camera and used the AI detection function in the SunFounder Controller.

In this project, we will try to control the movement and direction of the Zeus Car using the joystick widget in the app, so now let's see how it works.

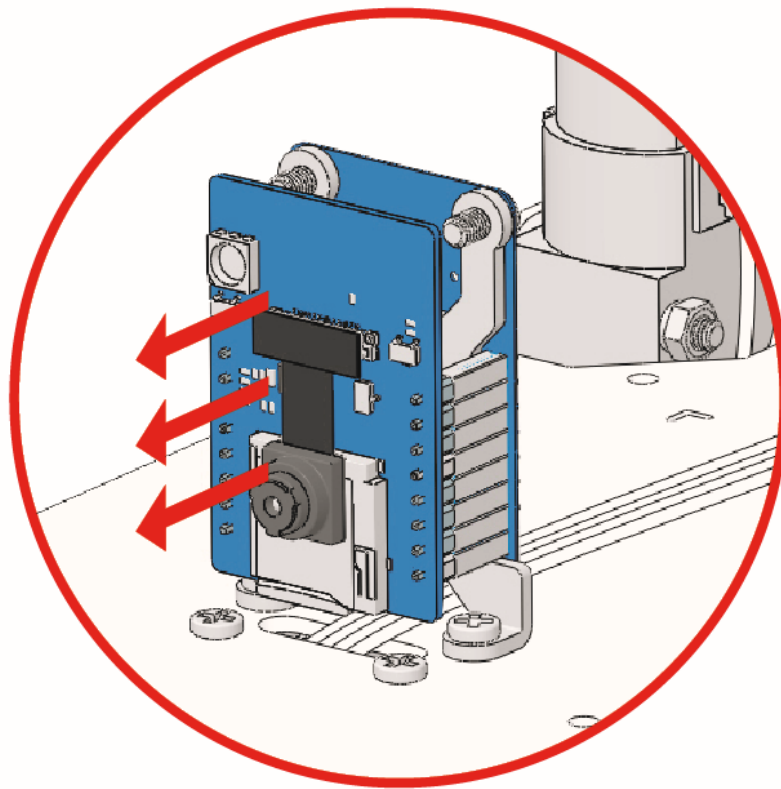
---

**Note:** Please install [SunFounder Controller](#) from **APP Store(iOS)** or **Google Play(Android)**.

---

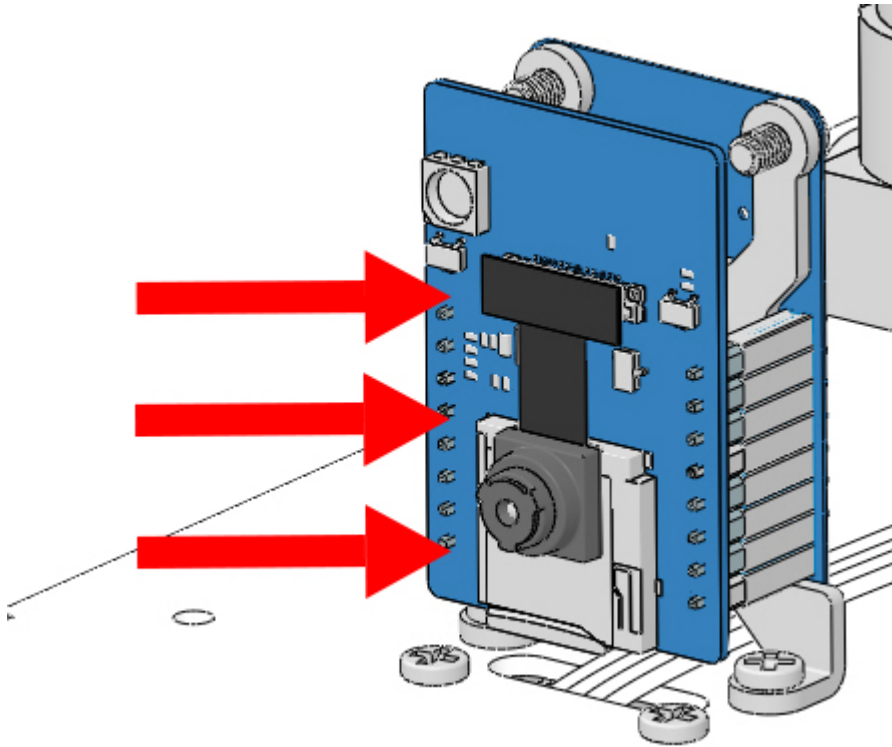
### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.

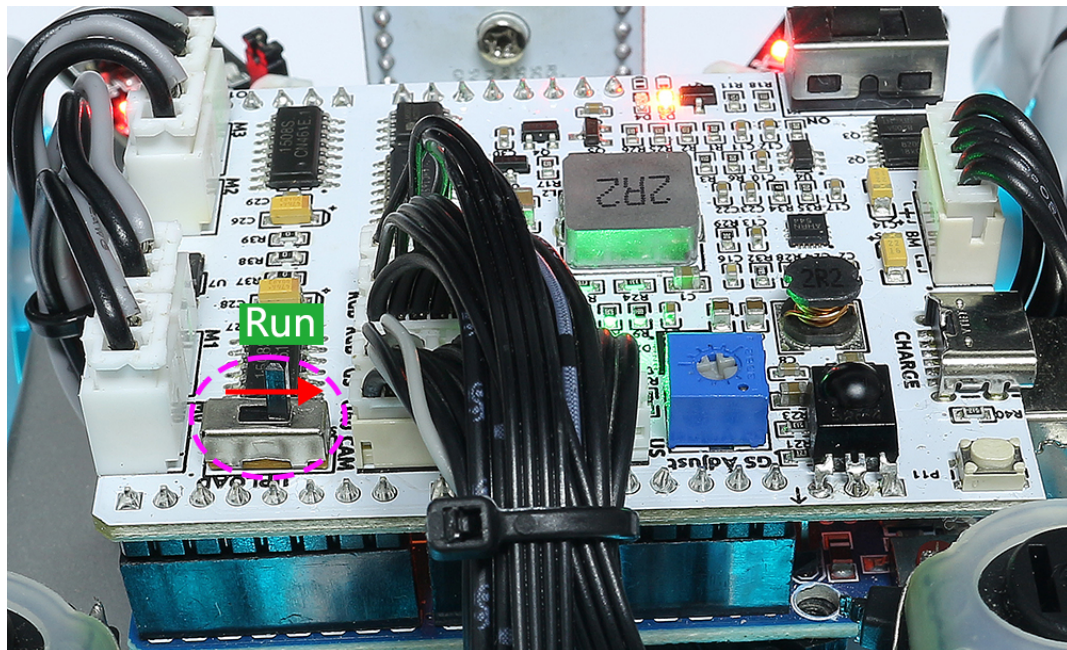


2. Open the `17_app_control.ino` file under the path of `zeus-car-main\examples\17_app_control`.
3. After the code is uploaded successfully, you can plug in the ESP32-CAM and then slide the power switch to ON to start the Zeus Car.

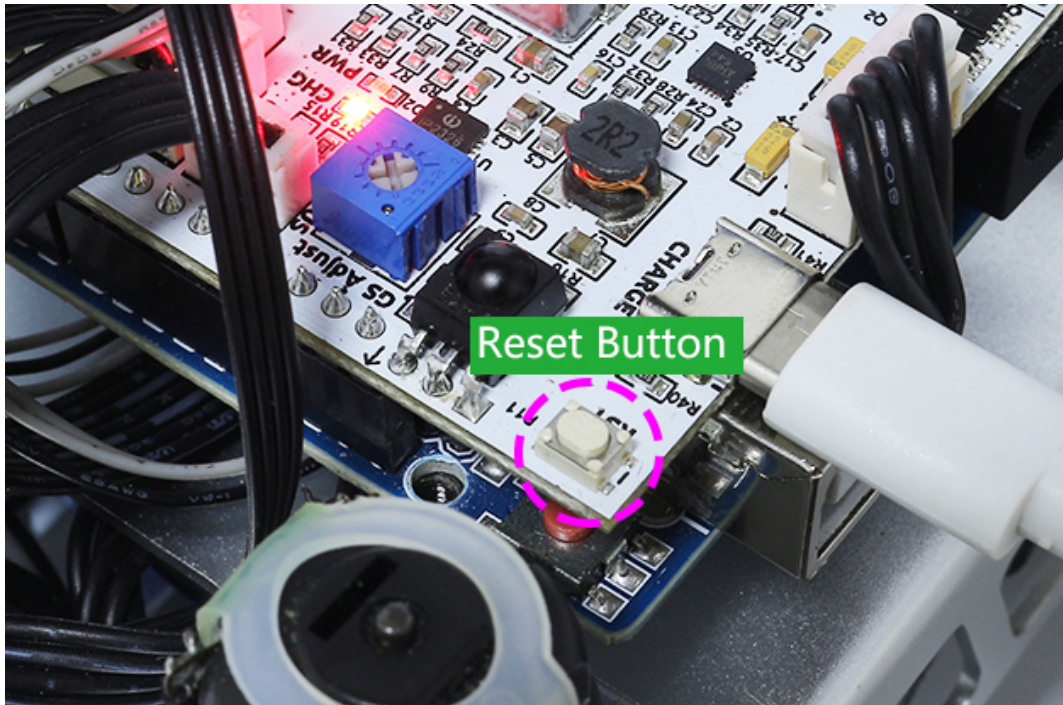




4. Toggle the Upload Switch to the side of Run (right side on this diagram) to start the ESP32 CAM.

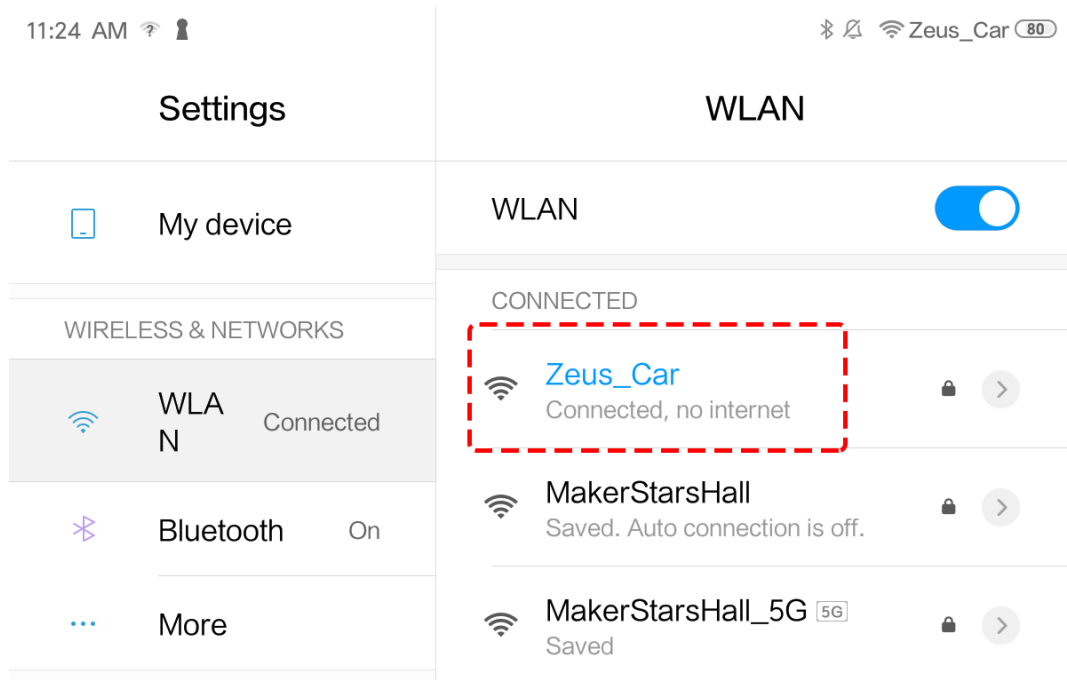


5. Press the reset button to get the Arduino board's program running again.

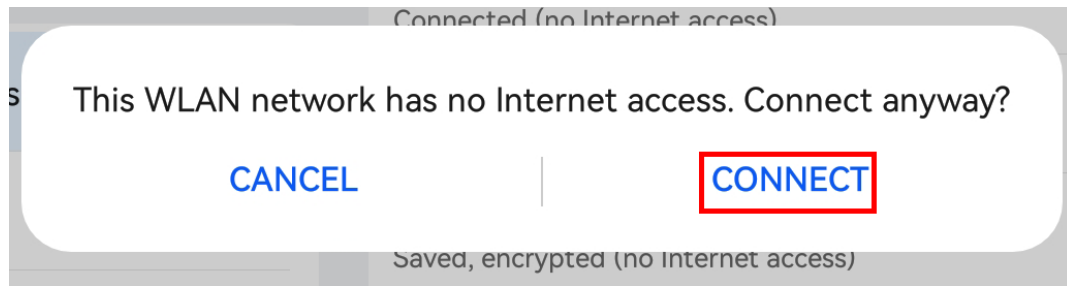


6. Connect to Zeus\_Car WLAN.

- Find Zeus\_Car on the WLAN of the mobile phone (tablet), enter the password 12345678 and connect to it.



- The default connection mode is AP mode. So after you connect, there will be a prompt telling you that there is no Internet access on this WLAN network, please choose to continue connecting.

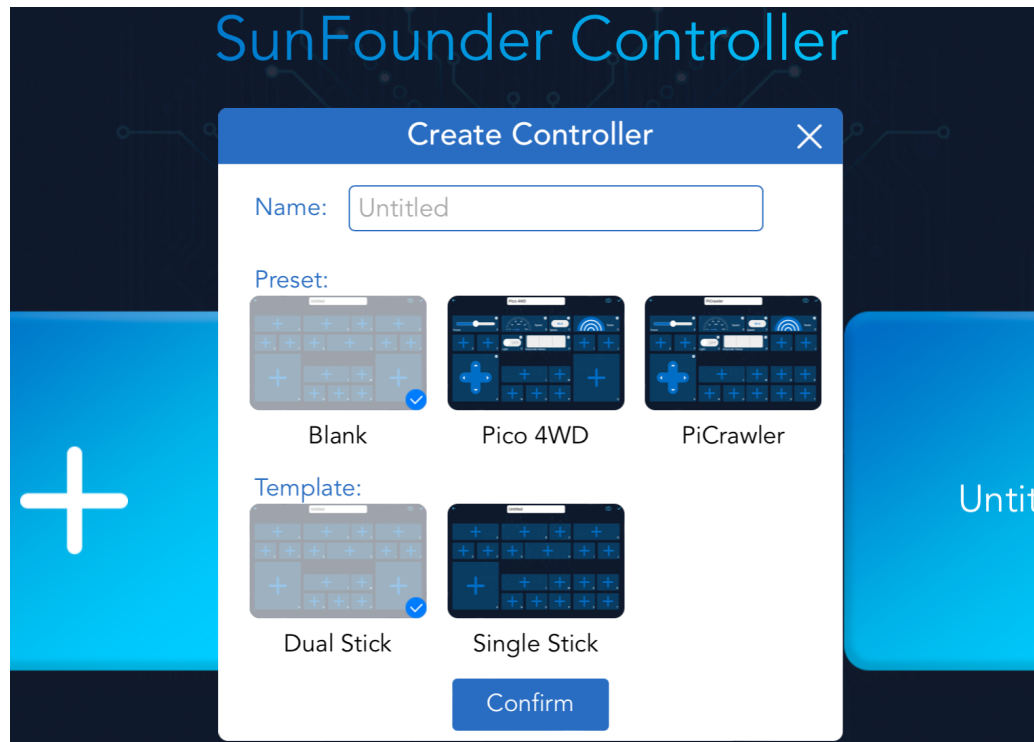


7. Create a controller.

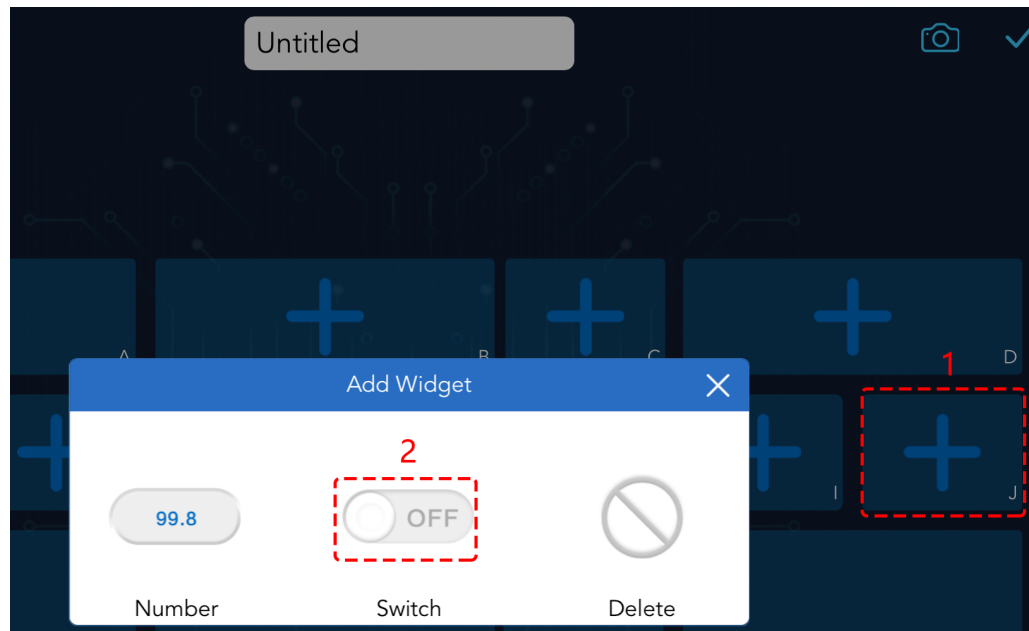
- To add a controller on SunFounder Controller, click the + icon.



- There are preset controllers for some products in the Preset section. But here we choose a **Blank** and **Dual Stick** template.

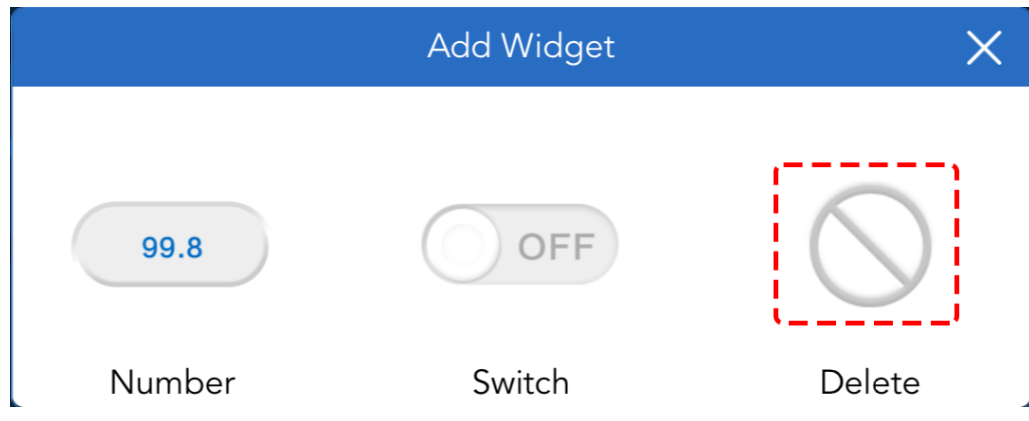


- Click the + icon in the J area and select a Switch widget. You may need to slide to the left to see the widget.

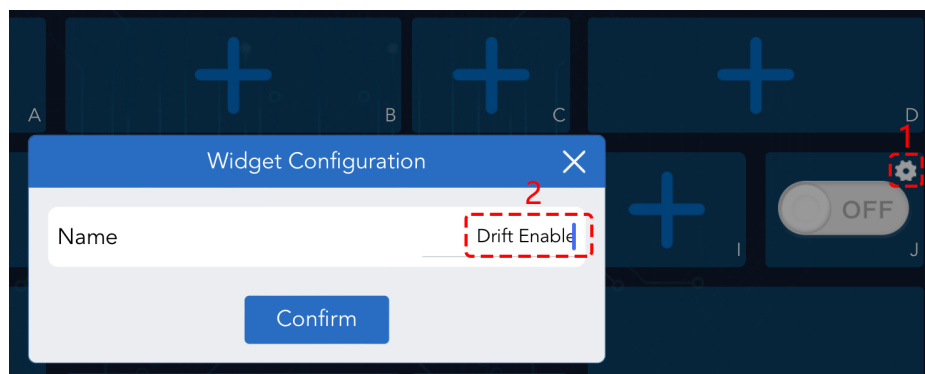



**Note:** If you think you selected the wrong widget, you can click on it again and use the Delete button to remove it.

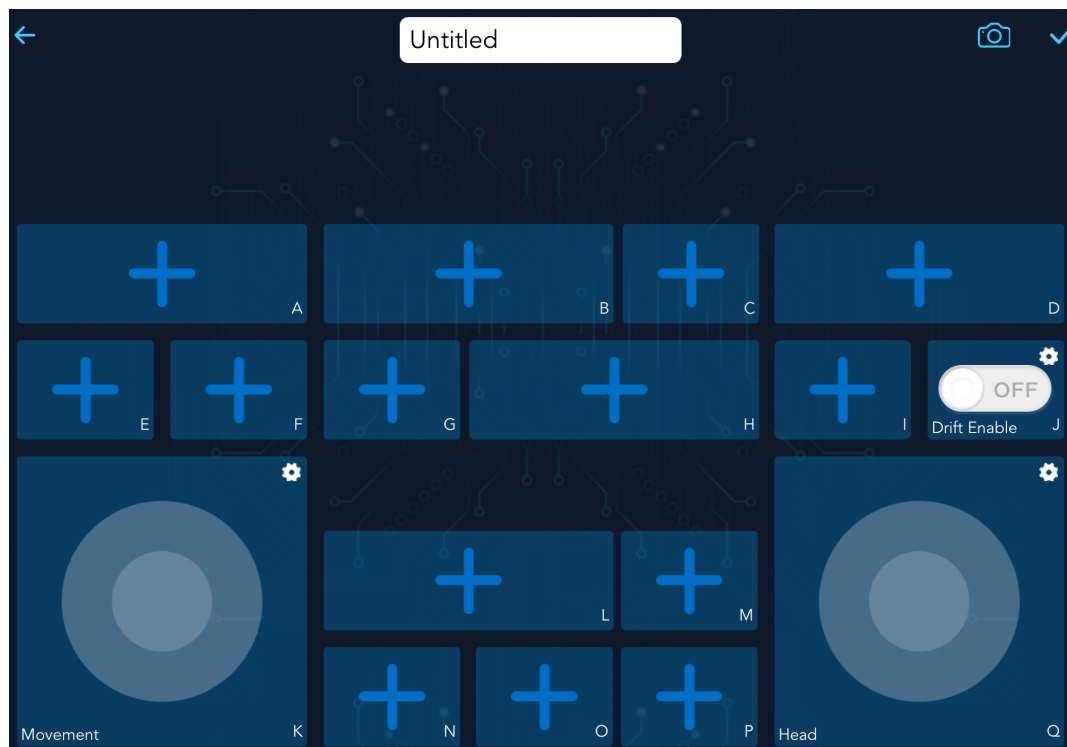





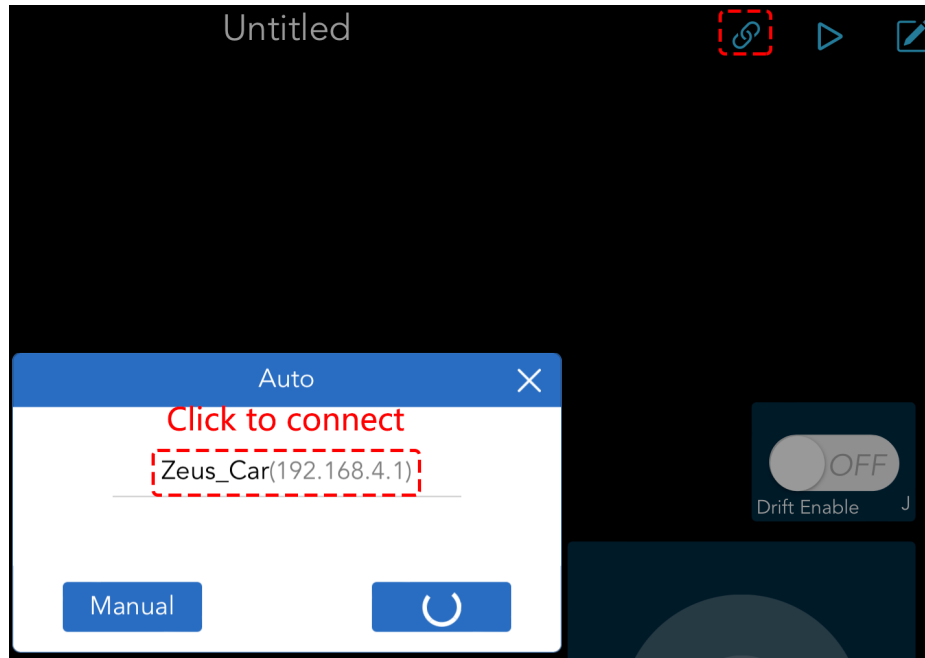
- Click the Settings icon in the upper right corner to give it a name.



- Select a Joystick widget in the K and Q area and give name to both widgets. Then use the  button to save the controller.




- The next step is to connect the Zeus Car to your device via the  button. Wait a few seconds and Zeus\_Car(IP) will appear, click on it to connect.



**Note:** Please make sure your Wi-Fi is connected to Zeus\_Car, if you are not seeing the above message for a long time.

#### 8. Run the Controller.

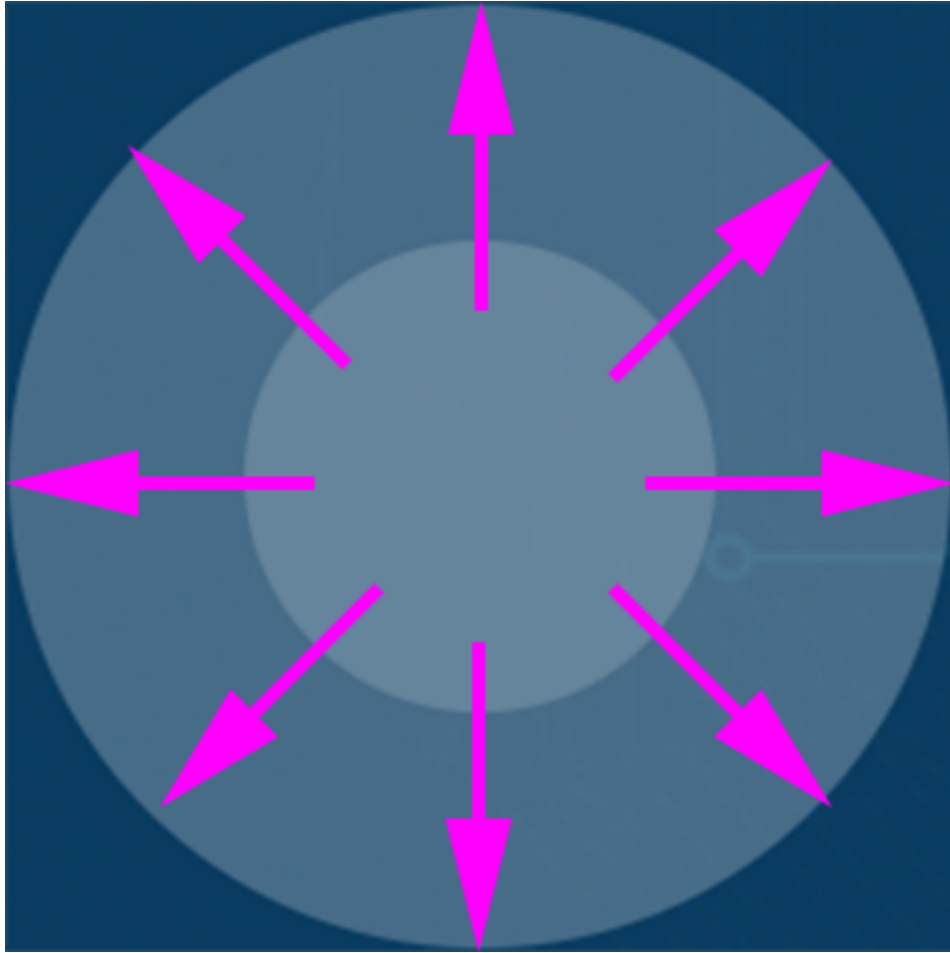
After the “Connected Successfully” message appears, click the  button, then the camera footage will appear on the app, and now you can control your Zeus Car with these widgets.

The functions of the three widgets are referenced as follows

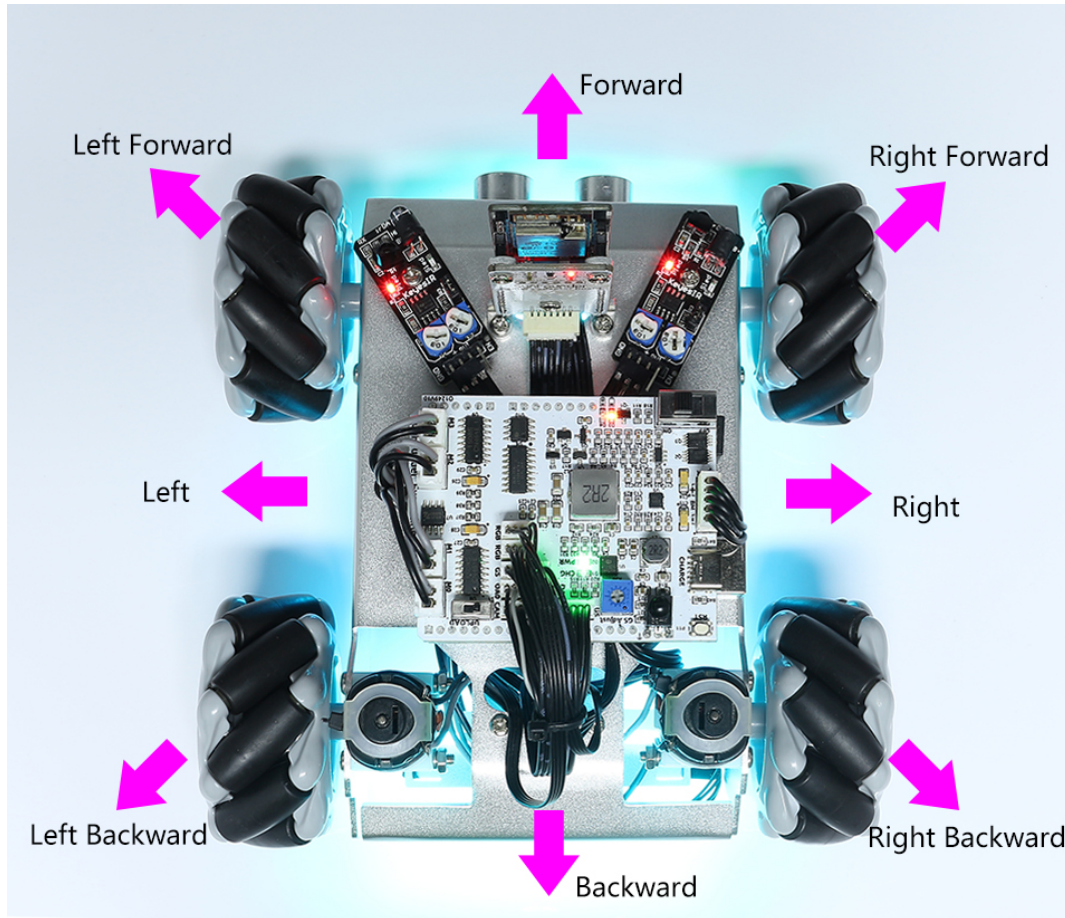
#### **Move in All Directions(K)**



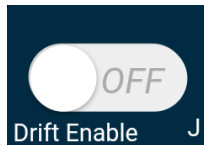
The Zeus Car will move in the appropriate direction when you swipe the widget.



The car moves once every time you slide, so if you don't release your hand all the time, the car keeps moving.



### Drift Enable(J)



Click the Drift Enable button to enable the drift function.

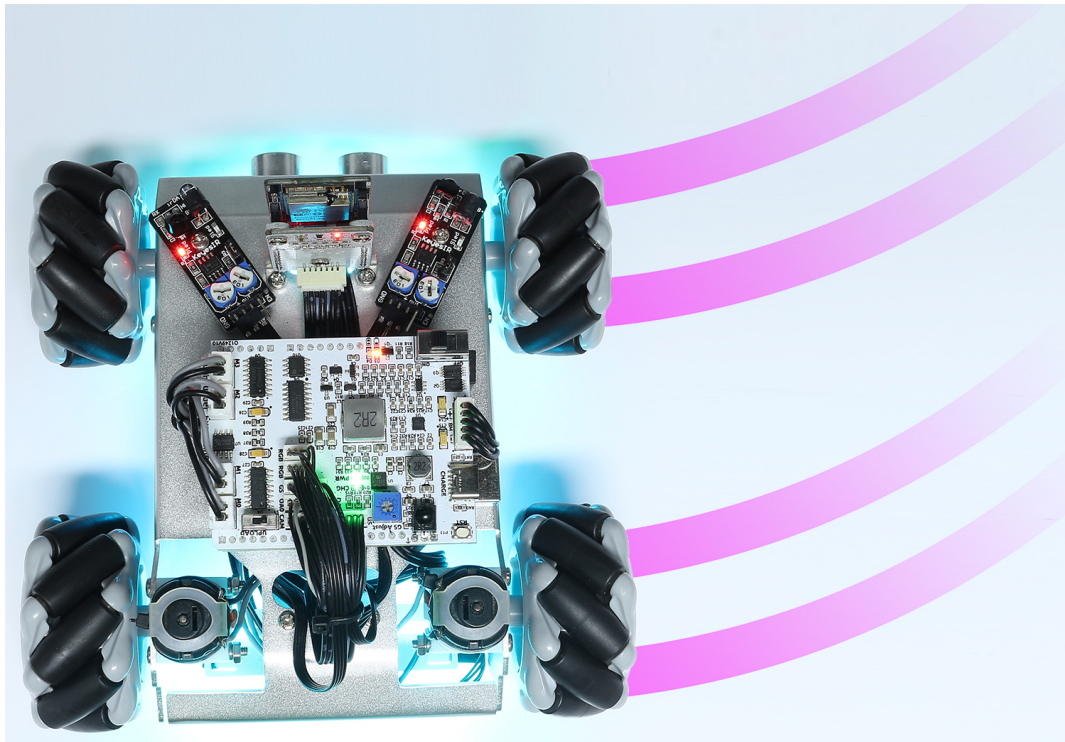


- By sliding the joystick counterclockwise, you will see Zeus Car drift to the right. Upon releasing the hand, the car will stop at its current position.










- Similarly, if you slide the **Q** widget clockwise, the Zeus Car will drift to the left and stop in the current position.

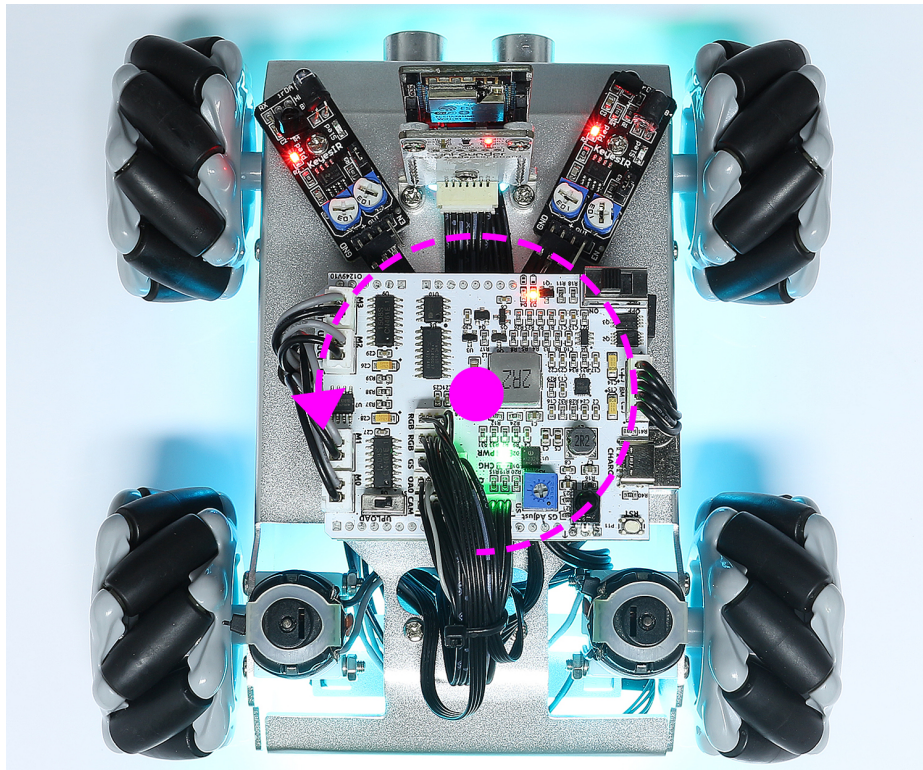


**Control the Drection(Q)**

- When the  button is on, the  widget is used to make the Zeus Car drift left and right.

- When the  widget is off, the  widget is used to control the direction of the car's head.

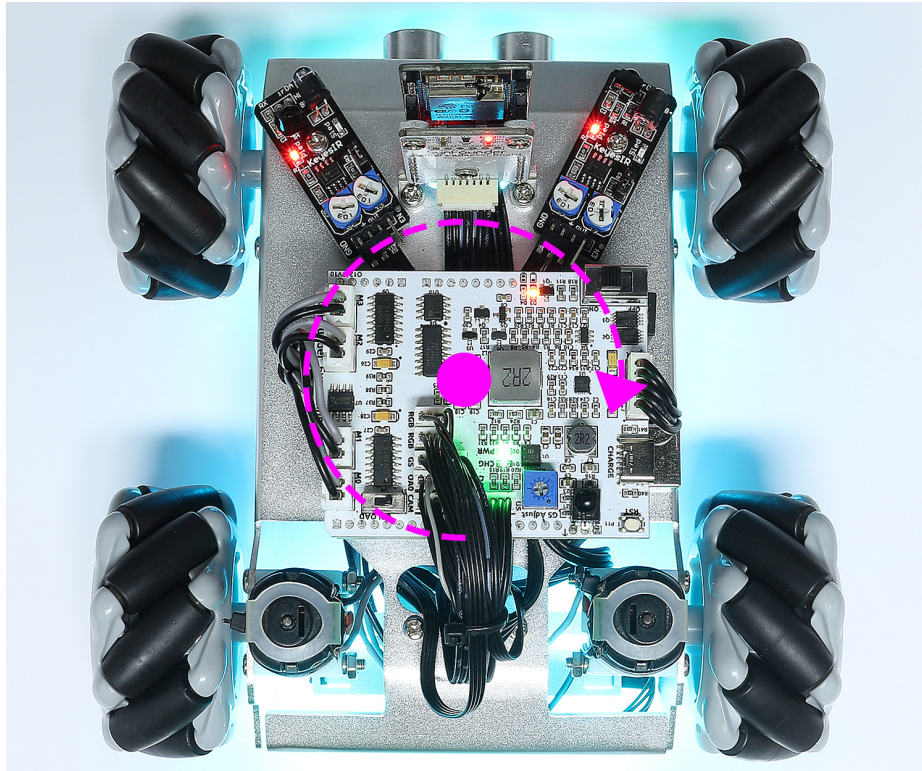
- By sliding the  widget counterclockwise, the car will also rotate counterclockwise. Upon releasing the hand, the head of the car will back to the original direction.







- Similarly the car will rotate clockwise with the widget and return to the original direction when released.



### 18. APP Control Plus

This project integrates Line Track, Follow, Avoid functions based on *17. APP Control*.

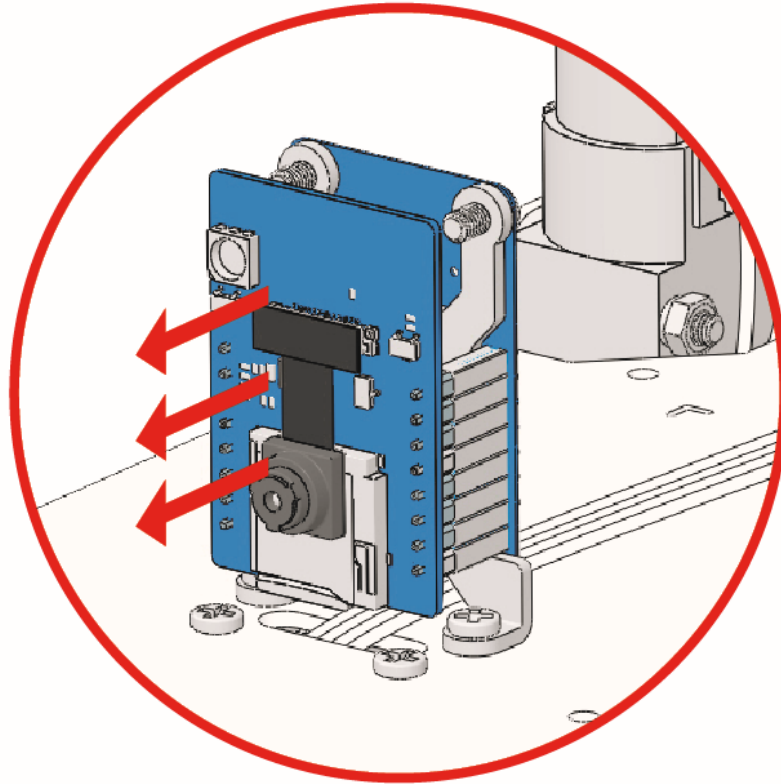
---

**Note:** Please install [SunFounder Controller](#) from **APP Store(iOS)** or **Google Play(Android)**.

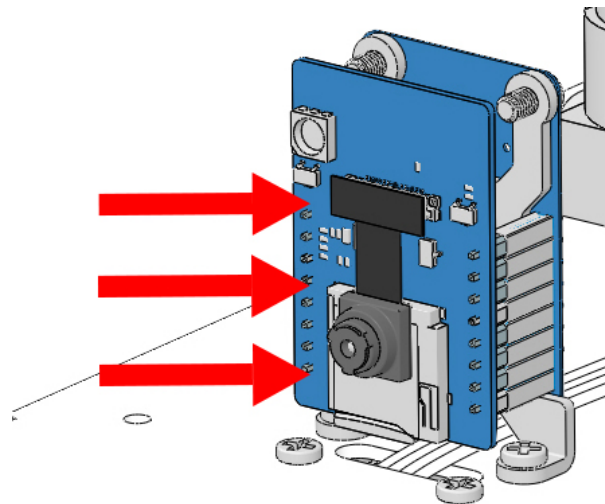
---

#### How to do?

1. The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you're uploading code, you'll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.

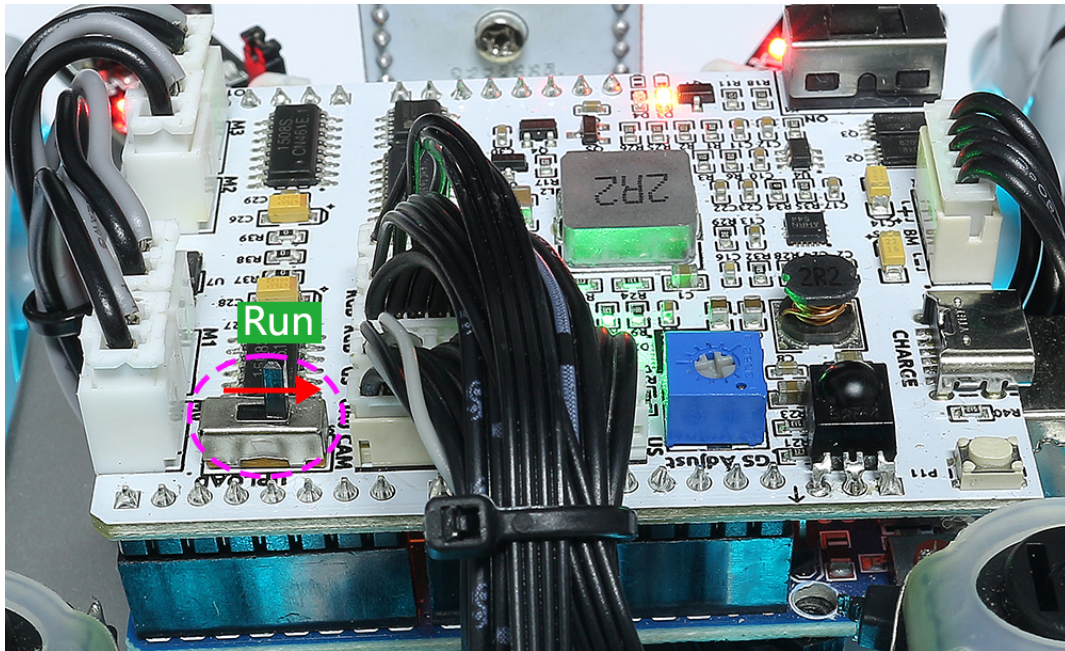


2. Open the `18_app_control_plus.ino` file under the path of `zeus-car-main\examples\18_app_control_plus`.
3. After the code is uploaded successfully, you can plug in the ESP32-CAM and then slide the power switch to ON to start the Zeus Car.

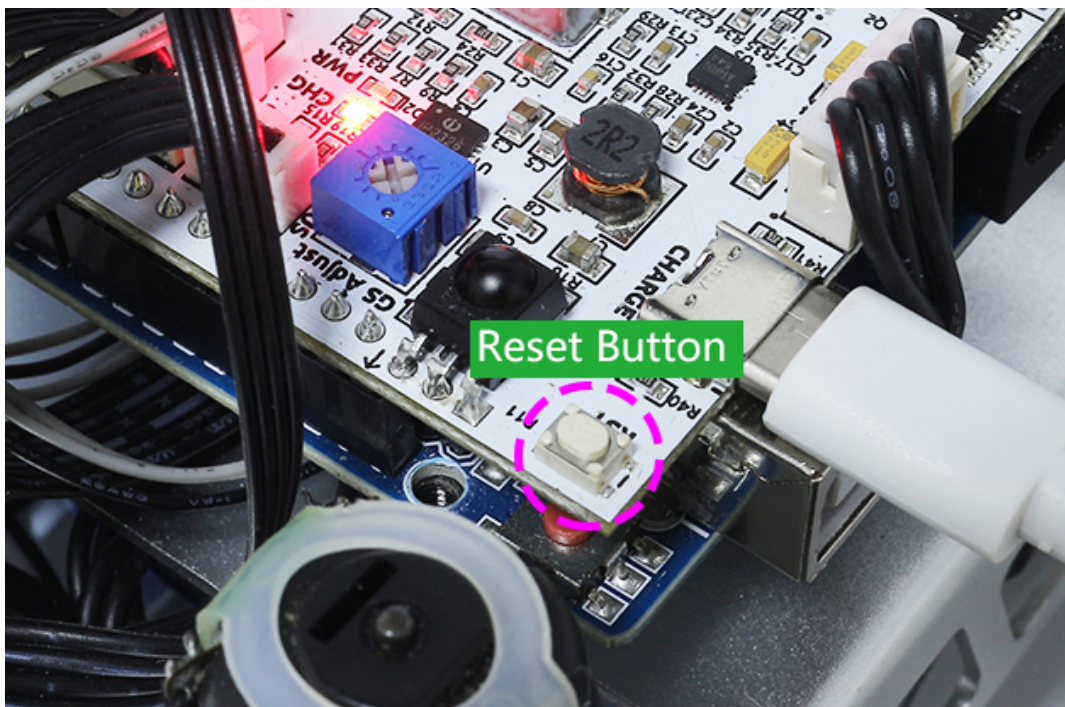


4. Toggle the Upload Switch to the side of Run (right side on this diagram) to start the ESP32 CAM.





5. Press the reset button to get the Arduino board's program running again.

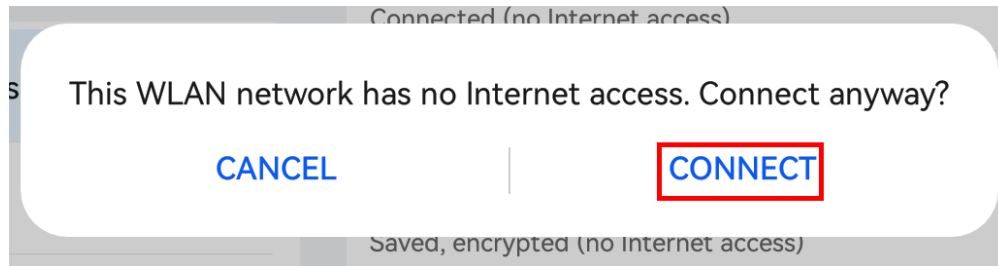


6. Connect to Zeus\_Car WLAN.

Now, connect your mobile device to the local area network (LAN) broadcast by the Zeus Car. This way, your mobile device and the Zeus Car will be on the same network, which will facilitate communication between the applications on your mobile device and the Zeus Car.

- Find Zeus\_Car on the WLAN of the mobile phone (tablet), enter the password 12345678 and connect to it.
- The default connection mode is AP mode. So after you connect, there will be a prompt telling

you that there is no Internet access on this WLAN network, please choose to continue connecting.

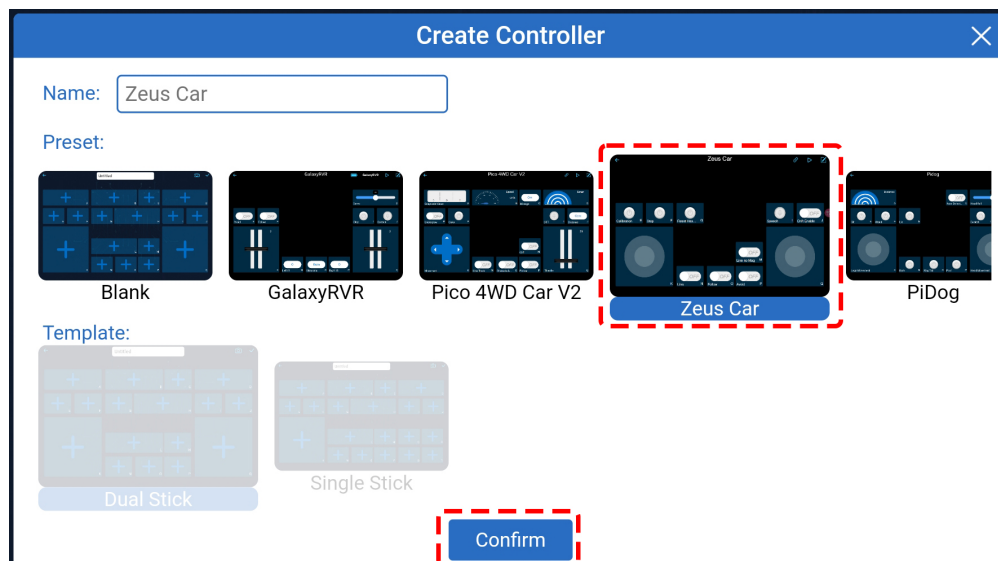


### 7. Create a controller.

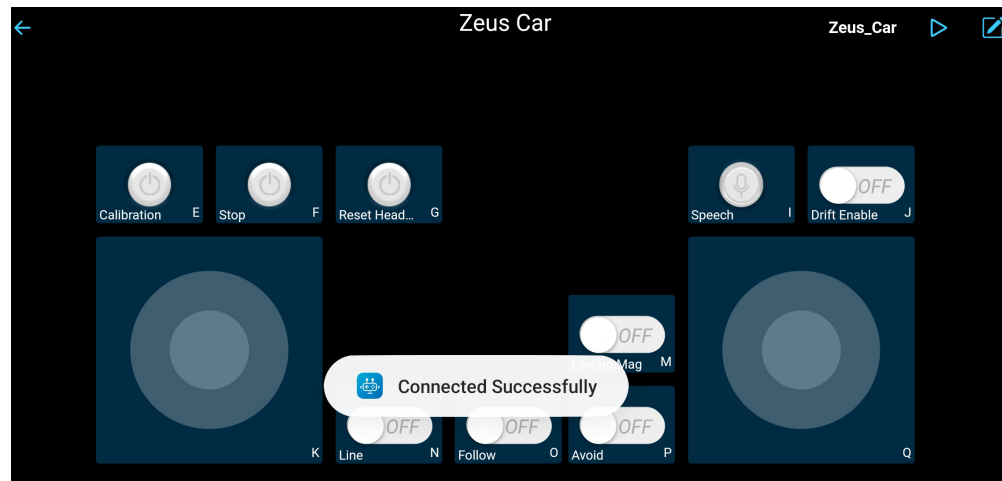
- To add a controller on SunFounder Controller, click the + icon.




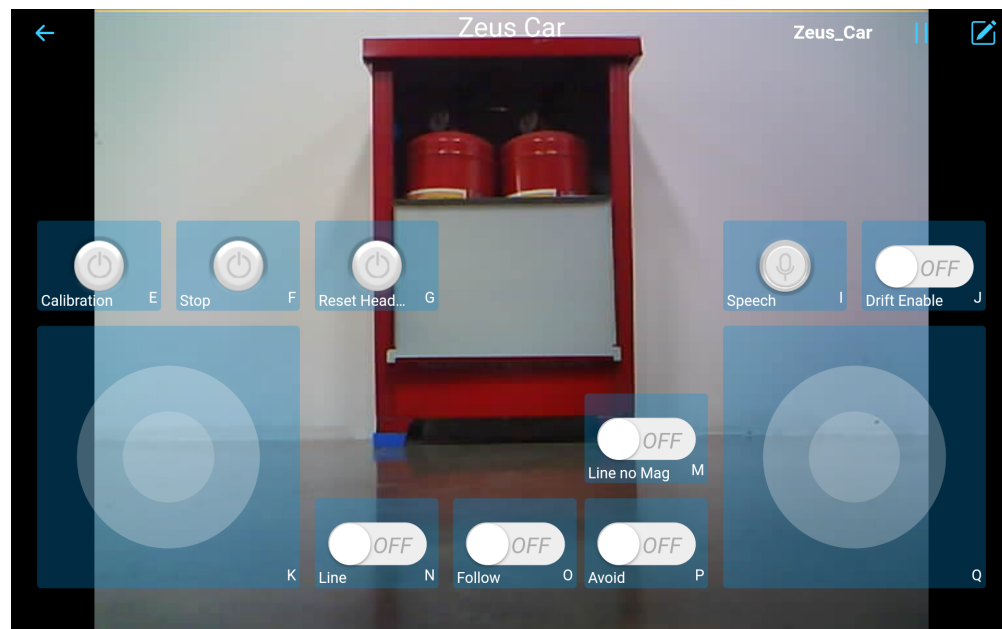
- Preset controllers are available for some products, here we choose **Zeus Car**. Give it a name, or simply tap **Confirm**.



- Once inside, the app will automatically search for the Zeus Car. After a moment, you will see a prompt saying “Connected Successfully.”



- Now, tap the  button enables you to view the live video feed from the camera and control the car using the provided widgets.

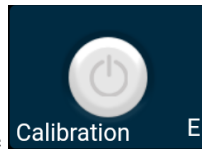


8. Here are the functions of the widgets.

- **Calibration(E)**: Turn on compass calibration.
- **Stop(F)**: Stop all movements of the car.
- **Reset Heading(G)**: After placing the car in one direction with your hand, click on this widget to make this direction as the front of the car movement. This allows you to quickly specify a direction instead of slowly rotating the car to that direction with other widgets.
- **Speech(I)**: Switching to speech control mode.
- **Drift Enable(J)**: Activate the drift function.
- **Move in All Directions(K)**: Control the car to move in all directions.

- *Line Track*: The following two widgets can both switch to line track mode.
  - **Line no Mag(M)**: Switch to line track mode, but not affected by the magnetic field. During the line tracking process, the Zeus Car's orientation will continuously change.
  - **Line(N)**: Switching to line track mode, due to the presence of the magnetic field, the Zeus Car's orientation during line tracking will be oriented towards a specific direction.
- *Follow(O)*: Switching to follow mode.
- *Avoid(P)*: Switch to obstacle avoidance mode.
- *Control the Direction(Q)*: Used to control the head direction.

### Calibration(E)



Turn on compass calibration by clicking the Calibration E button.

Place the Zeus car on the ground. Upon turning on the compass calibration, the car will start rotating counterclockwise and will stop in about 1 minute. If it rotates longer than 2 minutes, the magnetic field here is complicated. Try changing the location and calibrating again.

### Drift Enable(J)




Click the Drift Enable J button to enable the drift function.

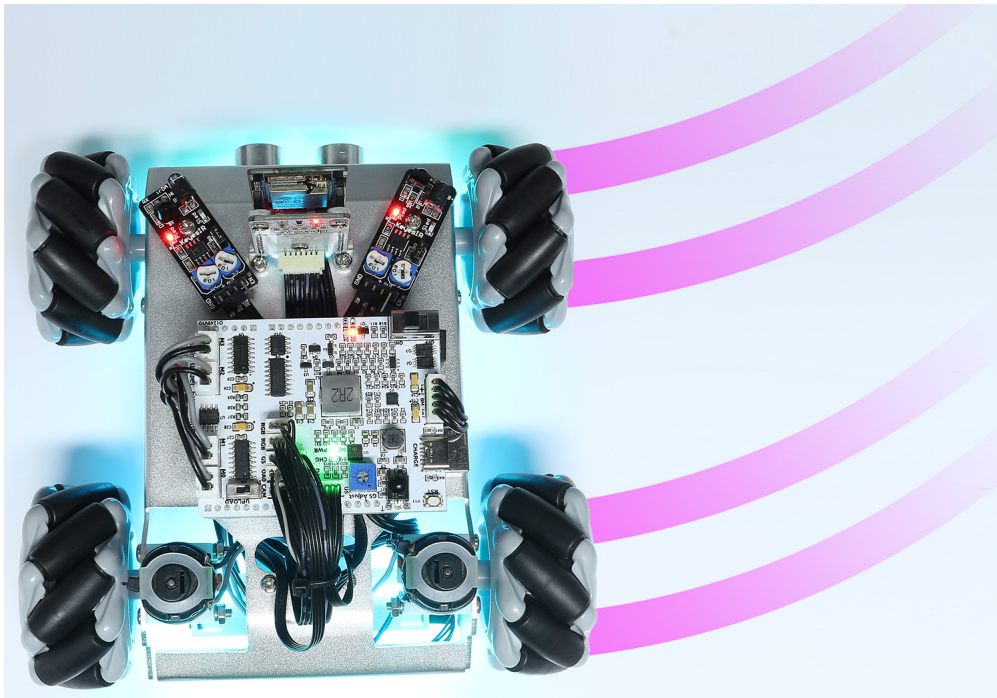


- By sliding the Q widget counterclockwise, you will see Zeus Car drift to the right. Upon releasing the hand, the car will stop at its current position.





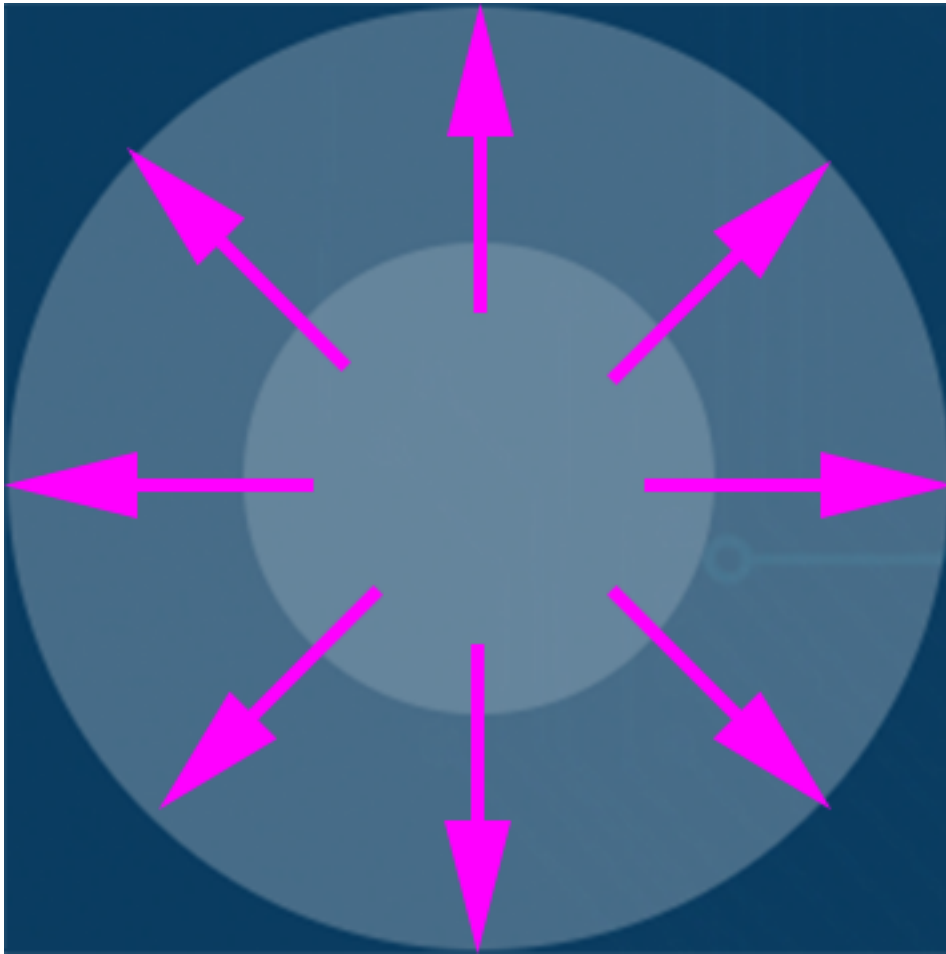
- Similarly, if you slide the  widget clockwise, the Zeus Car will drift to the left and stop in the current position.



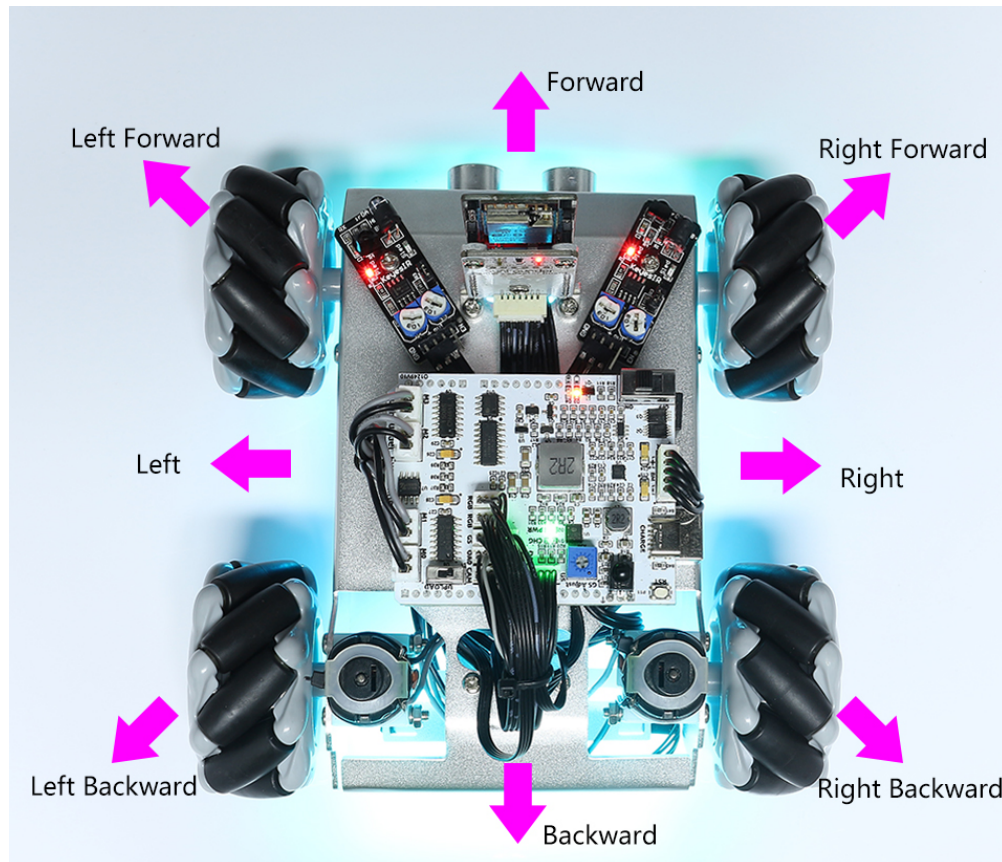
### Move in All Directions(K)



The Zeus Car will move in the appropriate direction when you swipe the K widget.



The car moves once every time you slide, so if you don't release your hand all the time, the car keeps moving.



## Speech(I)



By pressing the **Speech(I)** widget, you can activate the STT feature, where STT stands for Speech to Text.

The SunFounder Controller app integrates with your mobile device's voice recognition engine. Hence, when you tap and hold the **Speech(I)** widget on the SunFounder Controller and speak into your mobile device.

Your device will capture your speech, convert it into text, and send it to the Zeus Car. If this text matches the pre-set commands in your code, the Car will carry out the corresponding actions.

The following are the commands currently preset in the code. Speak any of the following commands and observe how the Zeus Car responds.

- stop: All movements of the car can be stopped.
- pasue: The function is basically the same as Stop, but if the head of the car is not facing the direction originally set, it will slowly move to the set direction.
- forward
- backward
- left forward
- left backward



- right forward
- right backward
- move left
- move right

---

**Note:** The STT (Speech to Text) function requires an internet connection and Google services on Android devices. However, this doesn't work with the pre-set AP (Access Point) mode on the Zeus Car.

In AP mode, the Zeus Car creates a local Wi-Fi network that your mobile device can connect to, but it does not provide internet access.

To use the STT function on Android, switch the car's code from AP to STA mode as outlined in [Q3: How can I use the STT feature on my Android device?](#).

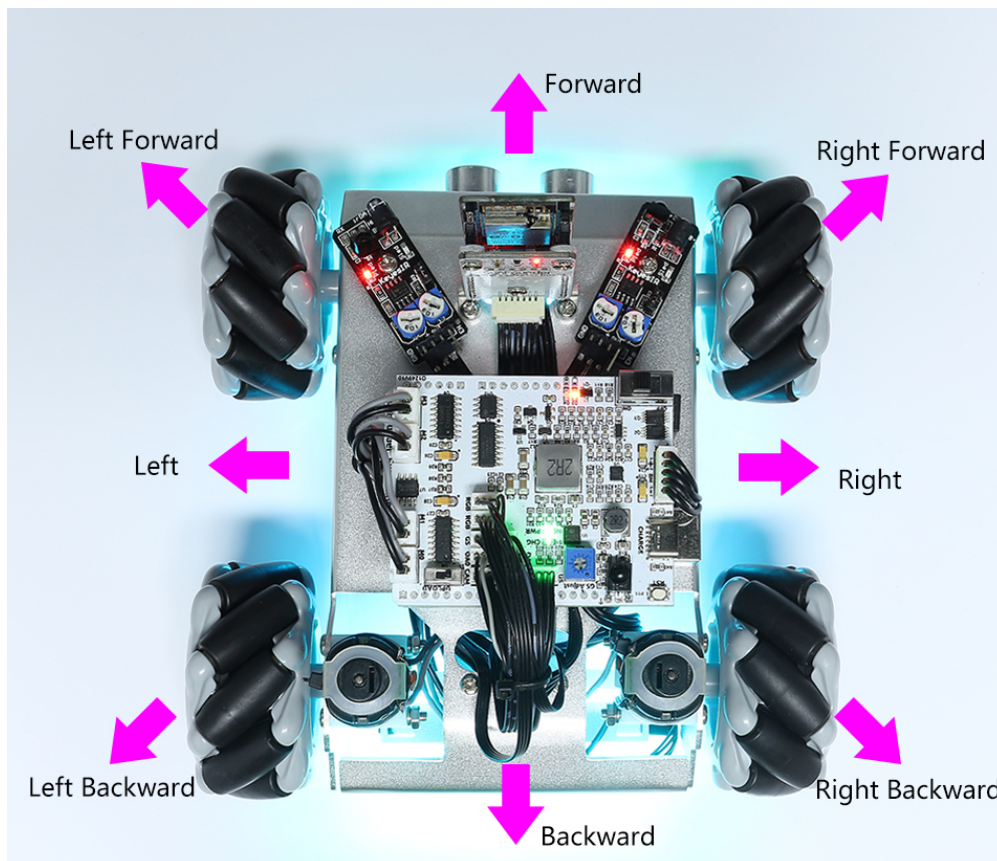
---

---

**Note:** iOS devices, using an offline voice recognition engine, work fine in both AP and STA modes.

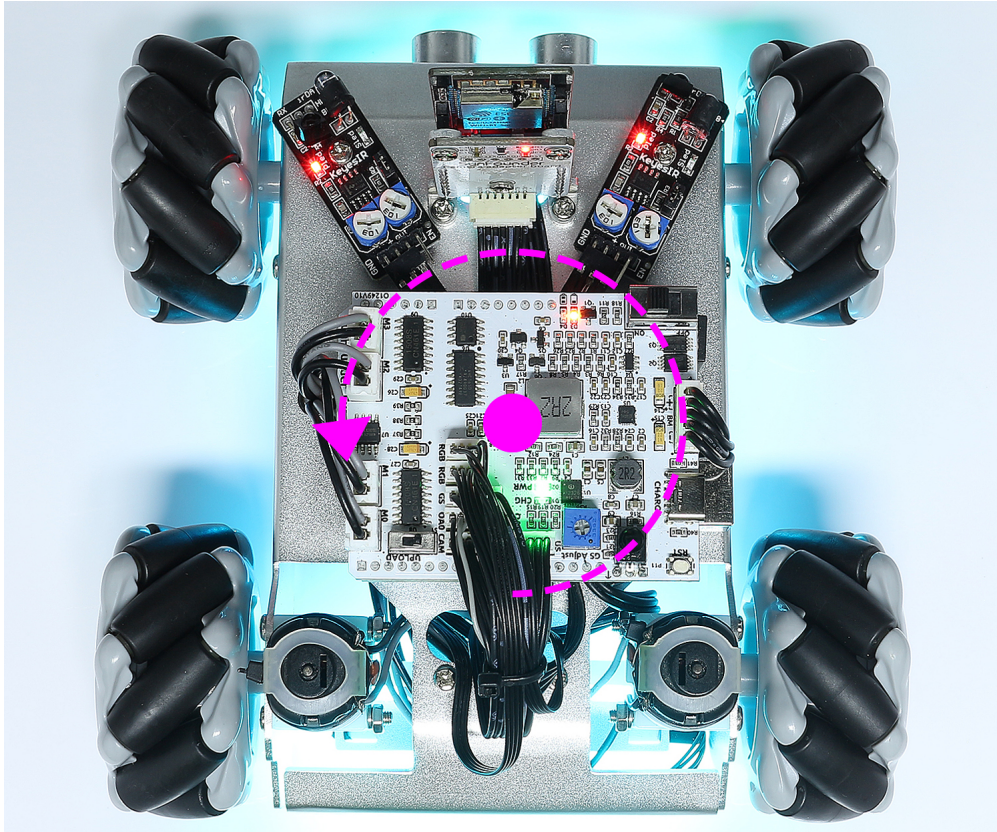
---

After the car receives the above 8 commands, it will keep moving in the corresponding direction unless it receives stop or pasue commands.

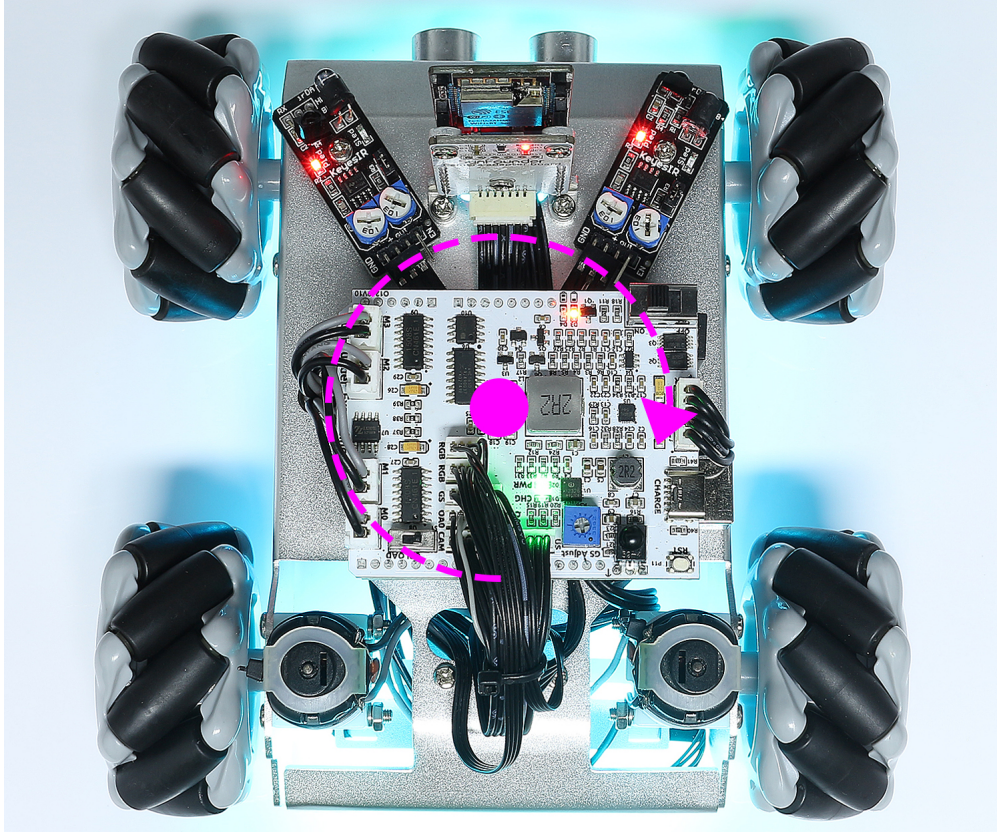


- **turn left:** This command will make the car to turn left 45° with the body as the center, then it will move forward or stop according to the previous state. If the previous state is stop, it will stop after turning left 45°; if it is forward, it will move forward after turning.

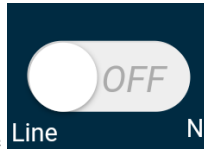




- **turn right:** This command will make the car turn  $45^\circ$  to the right with the body as the center, and again will move forward or stop depending on the previous state.



## Line Track



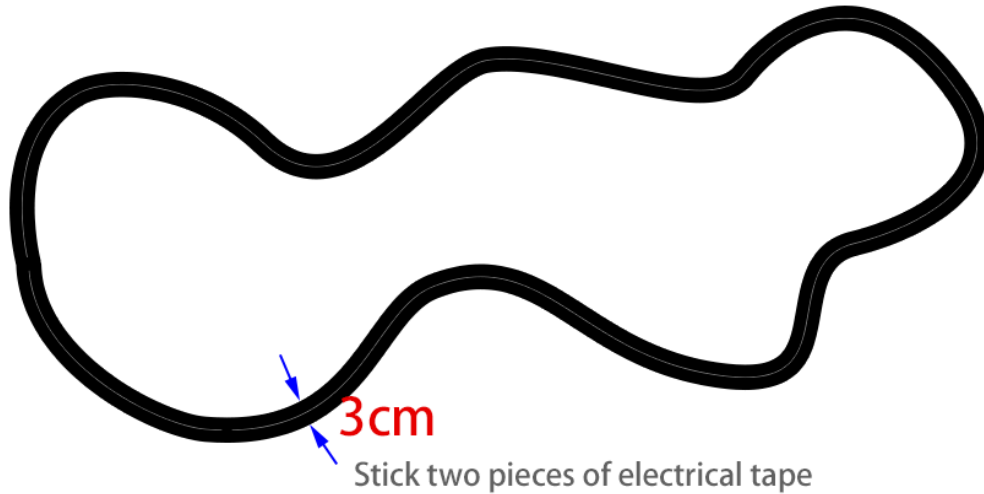
Click on the **Line** widget to switch to the line track mode.

Two modes of line track are available on the Zeus Car, one with its head always facing the direction of movement and one with its head facing a fixed direction. Here, the second mode is selected.

1. Stick a 3cm wide line

There are eight sensors on the Omni grayscale module, and the distance between each sensor is between 2 and 3 cm. There must be two sensors to detect the black line simultaneously. Therefore, the line you stick must be at least 3cm wide and the bend angle should not be less than 90°.

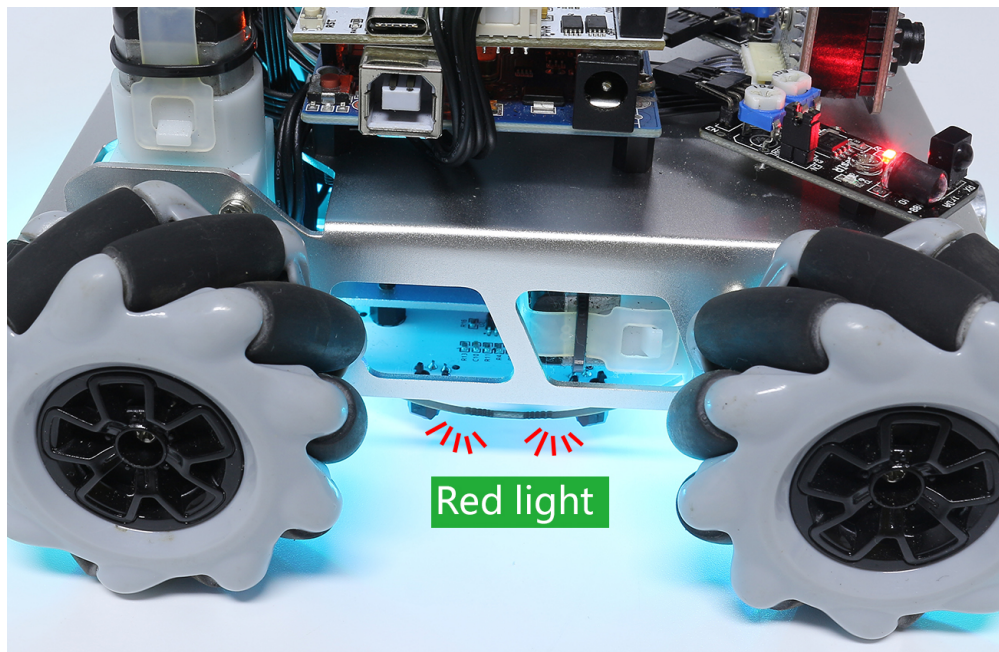




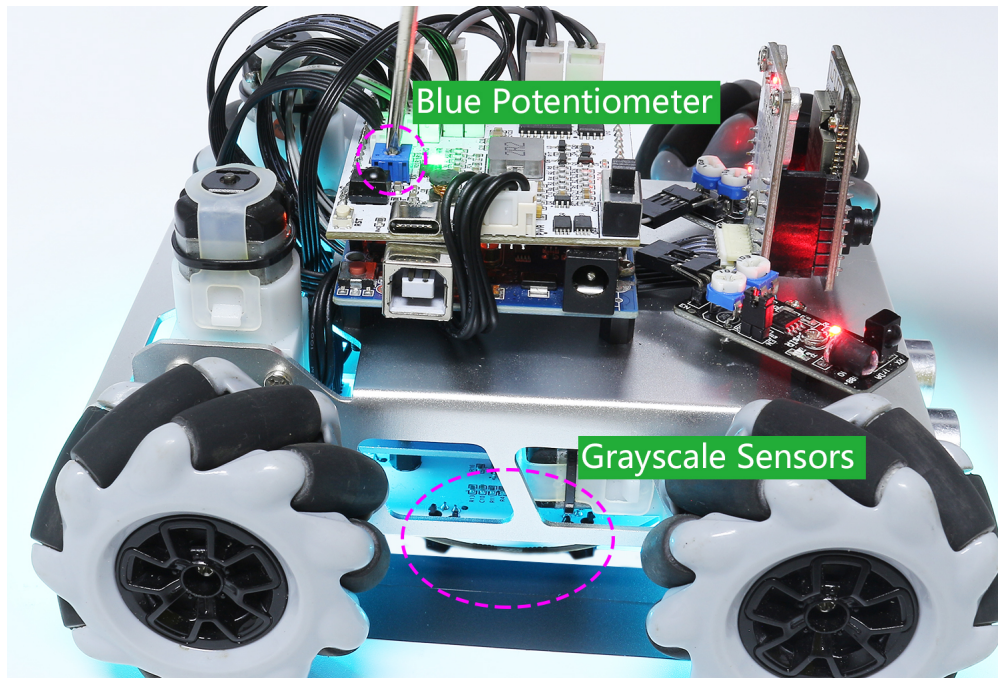
2. Calibrate the Omni Grayscale module.

Since each subfloor has different grayscale values, the factory-set grayscale threshold may not be appropriate for your current environment, so you will need to calibrate this module before use. It is recommended that you need to calibrate it whenever the floor color changes a lot.

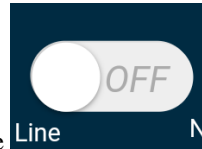
- Place the Zeus Car on white surface and turn the potentiometer until the gray sensor light is just illuminated.



- Now let the two greyscale sensors on the side be located just between the black line and white surface, and slowly turn the potentiometer until the signal indicator just goes off.

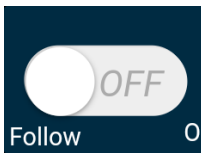


- You can move repeatedly over the the black line and white surface to make sure that the lights of the greyscale sensor are off when they are between the the black line and white surface and on when they are on the white surface, indicating that the module is successfully calibrated.



3. Place the Zeus Car on your stickied line, click the **Line** widget, and it will track the line.
4. Due to the high environmental requirements of the Omni grayscale module, it is recommended to calibrate it a few more times if the tracking effect is not satisfactory (off-track).

## Follow(O)

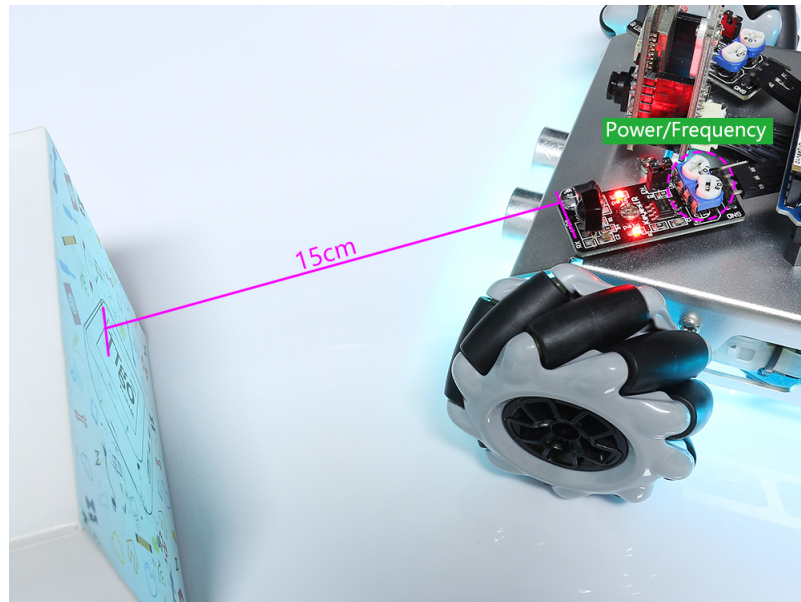


Click the **Follow** widget to switch to follow mode.

The ultrasonic sensor detects obstacles in front (20 cm) and follows them forward. These two obstacle avoidance modules allow the car to follow left or right, but they need to be calibrated (15cm) before use.

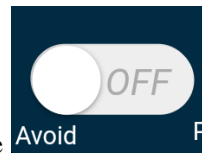
1. Calibrate the IR obstacle avoidance module.
  - Start by adjusting the right obstacle avoidance module. During transportation, collisions may cause the transmitter and receiver on the infrared module to tilt. Therefore, you need to manually straighten them.
  - Place an obstacle about 15cm away from the IR obstacle avoidance module.

- On the module are two potentiometers, one to adjust the sending power and one to adjust the sending frequency. By adjusting these two potentiometers, you can adjust the detection distance.
- Then you can adjust a potentiometer, and if at 15cm, the signal light on the module illuminates, the adjustment is successful; if it doesn't, adjust another potentiometer.



- Calibrate the other obstacle avoidance module in the same way.
2. Place Zeus car on a table or the ground and let it follow your hand or other obstacles.

### Avoid(P)



When you want to go into obstacle avoidance mode, click the **Avoid P** widget, but first reference the *Follow(O)* to calibrate the two obstacle avoidance modules.


- Zeus car will move forward.
- An ultrasonic module detects obstacles in front, if detected, the car turns left.
- When the left obstacle avoidance module detects an obstacle, the car turns right, and when the right obstacle avoidance module detects an obstacle, the car turns left.

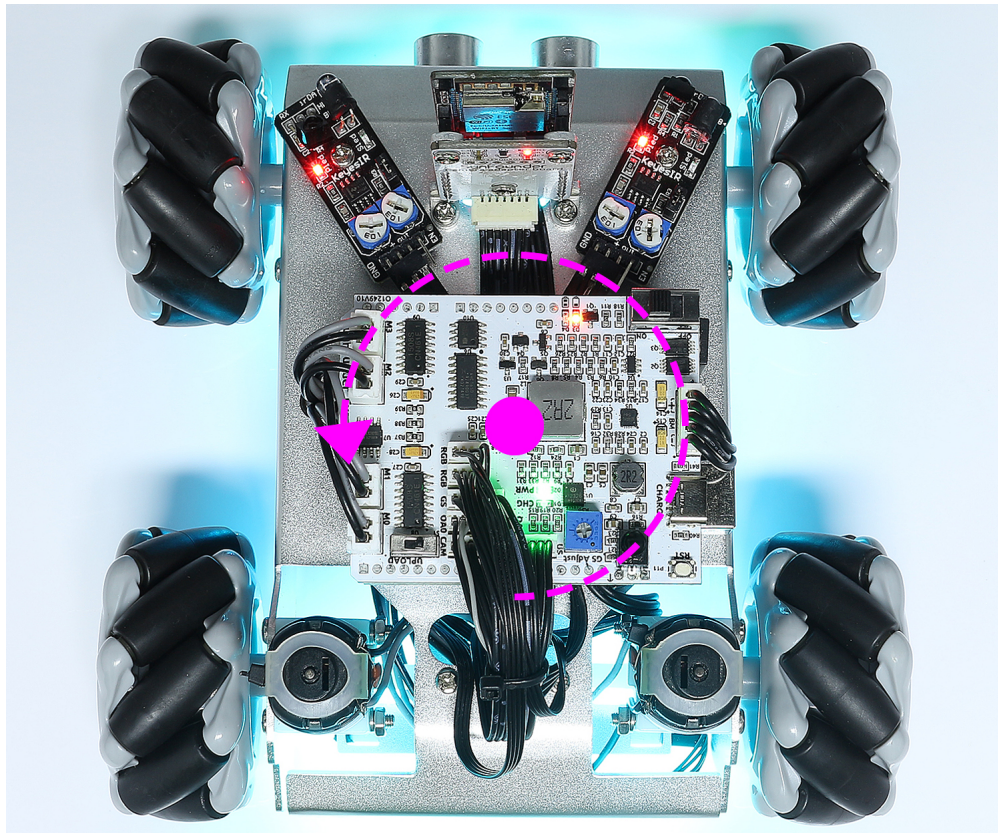


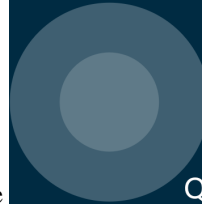
## Control the Drection(Q)


- When the  button is on, the  widget is used to make the Zeus Car drift left and right.

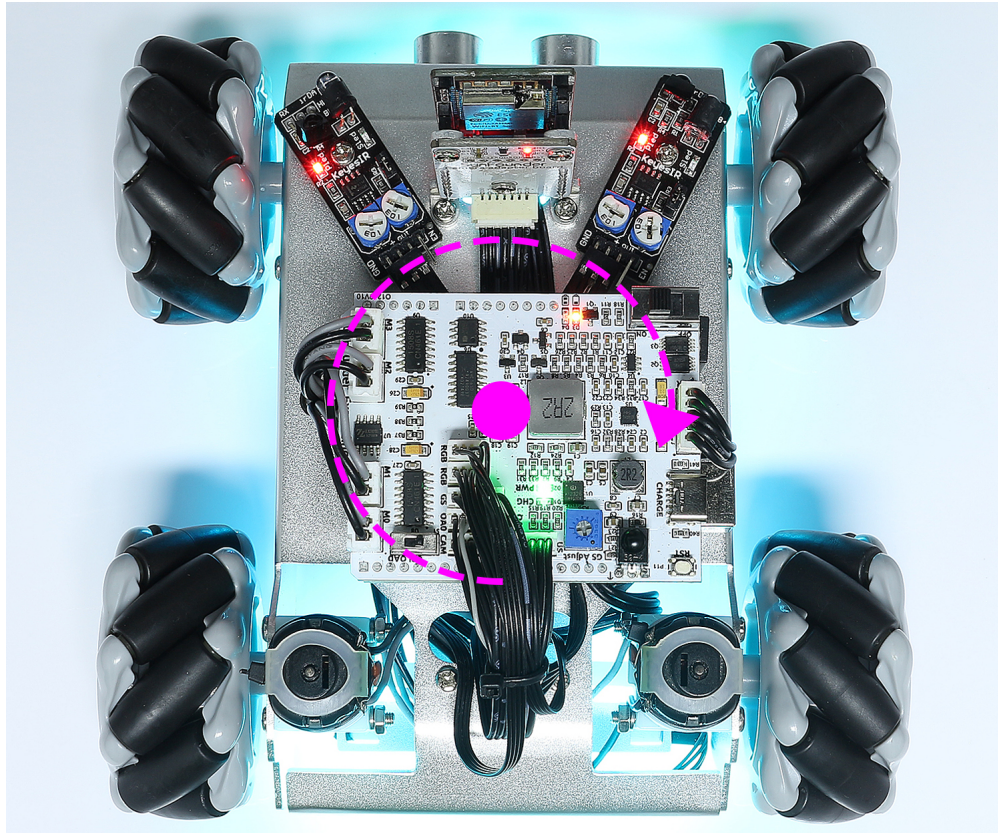
- When the  widget is off, the  widget is used to control the direction of the car's head.

- By sliding the  widget counterclockwise, the car will also rotate counterclockwise. Upon releasing the hand, the head of the car will back to the original direction.





- Similarly the car will rotate clockwise with the  widget and return to the original direction when released.



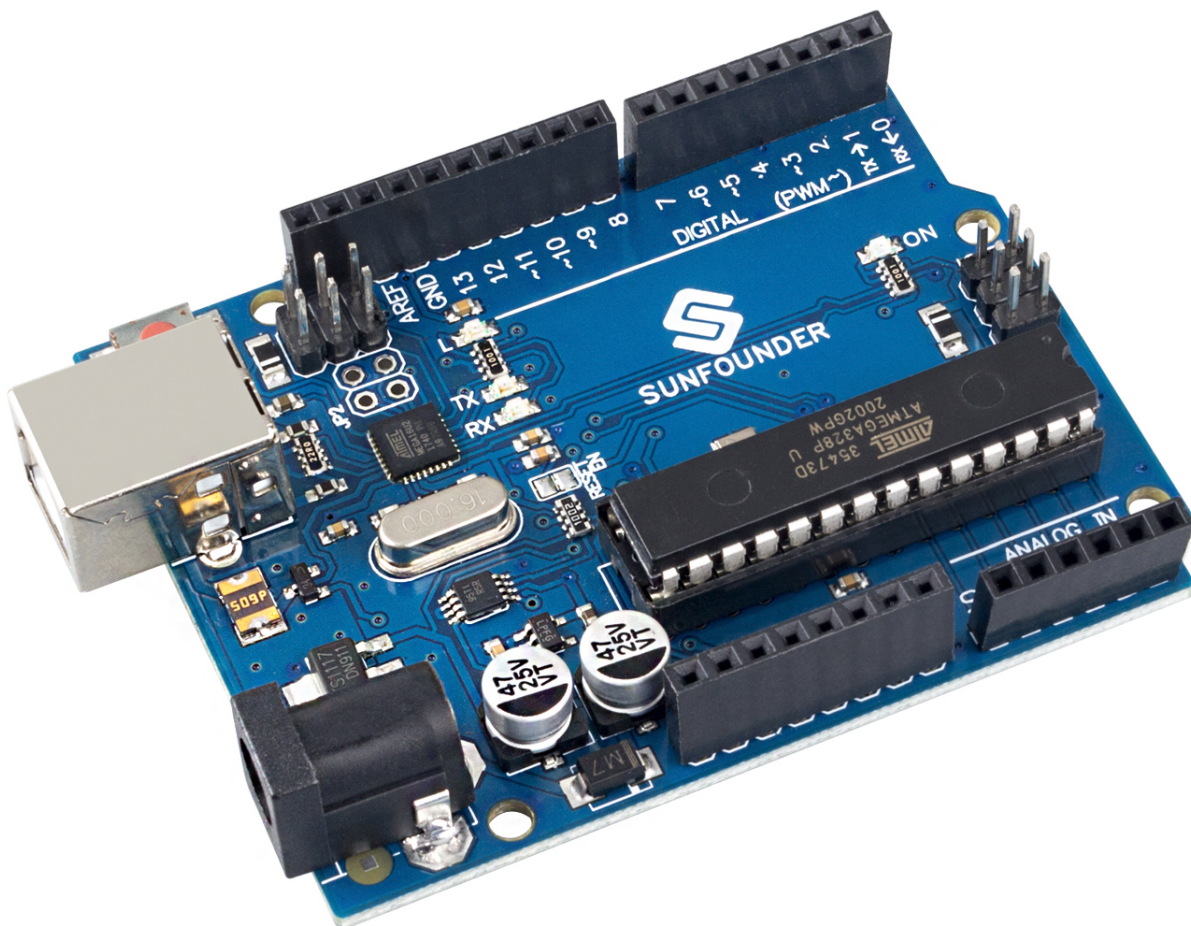


## HARDWARE

When you are writing code, you may need to know how each module works or the role of each pin, then please see this chapter.

In this chapter you will find a description of each module's function, technical parameters and working principle.

### 2.1 SunFounder R3 Board



---

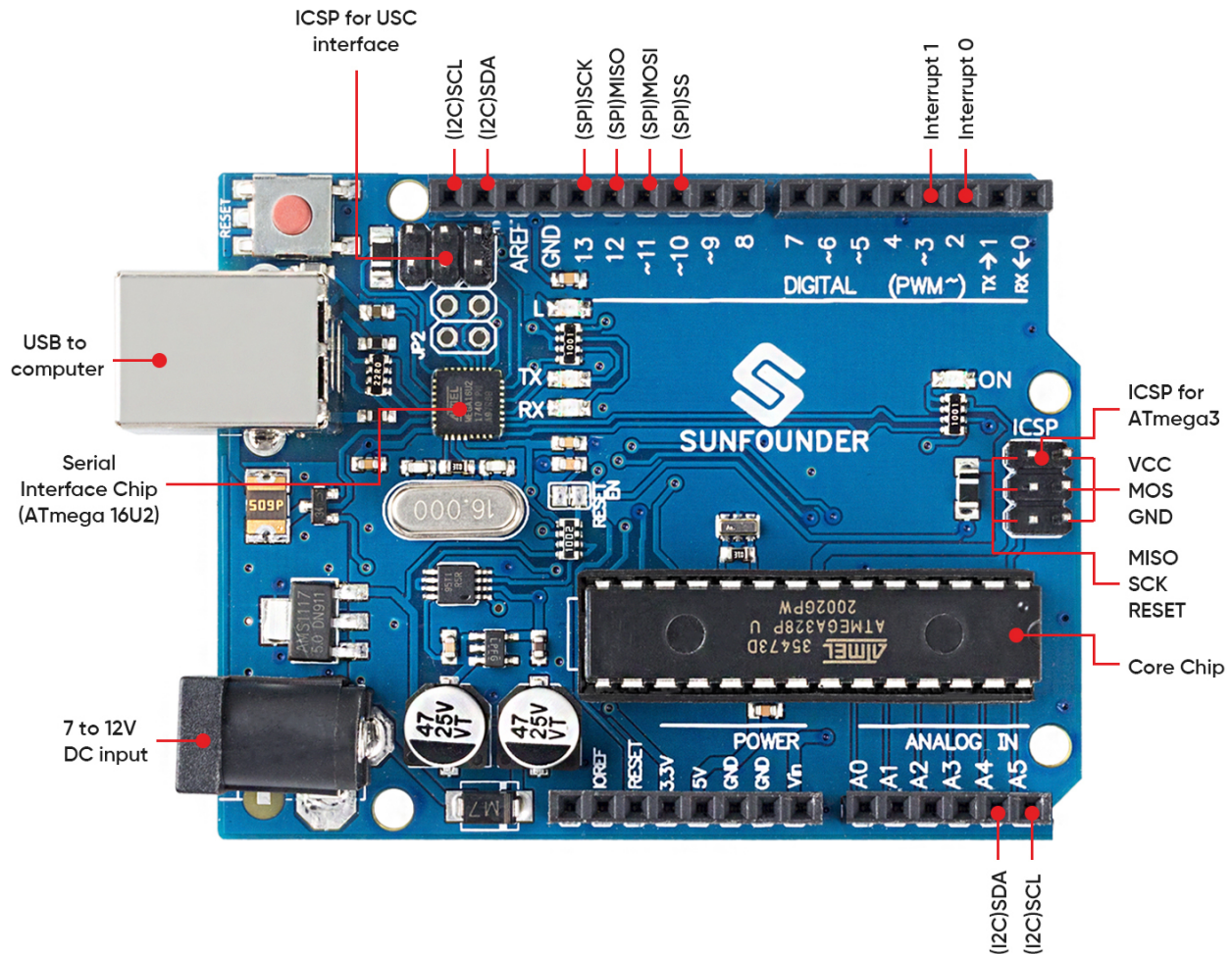
**Note:** The SunFounder R3 board is a mainboard with almost the same functions as the [Arduino Uno](#), and the two

---

boards can be used interchangeably.

SunFounder R3 board is a microcontroller board based on the ATmega328P ([datasheet](#)). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

### Technical Parameters



- MICROCONTROLLER: ATmega328P
- OPERATING VOLTAGE: 5V
- INPUT VOLTAGE (RECOMMENDED): 7-12V
- INPUT VOLTAGE (LIMIT): 6-20V
- DIGITAL I/O PINS: 14 (0-13, of which 6 provide PWM output(3, 5, 6, 9-11))
- PWM DIGITAL I/O PINS: 6 (3, 5, 6, 9-11)
- ANALOG INPUT PINS: 6 (A0-A5)
- DC CURRENT PER I/O PIN: 20 mA
- DC CURRENT FOR 3.3V PIN: 50 mA

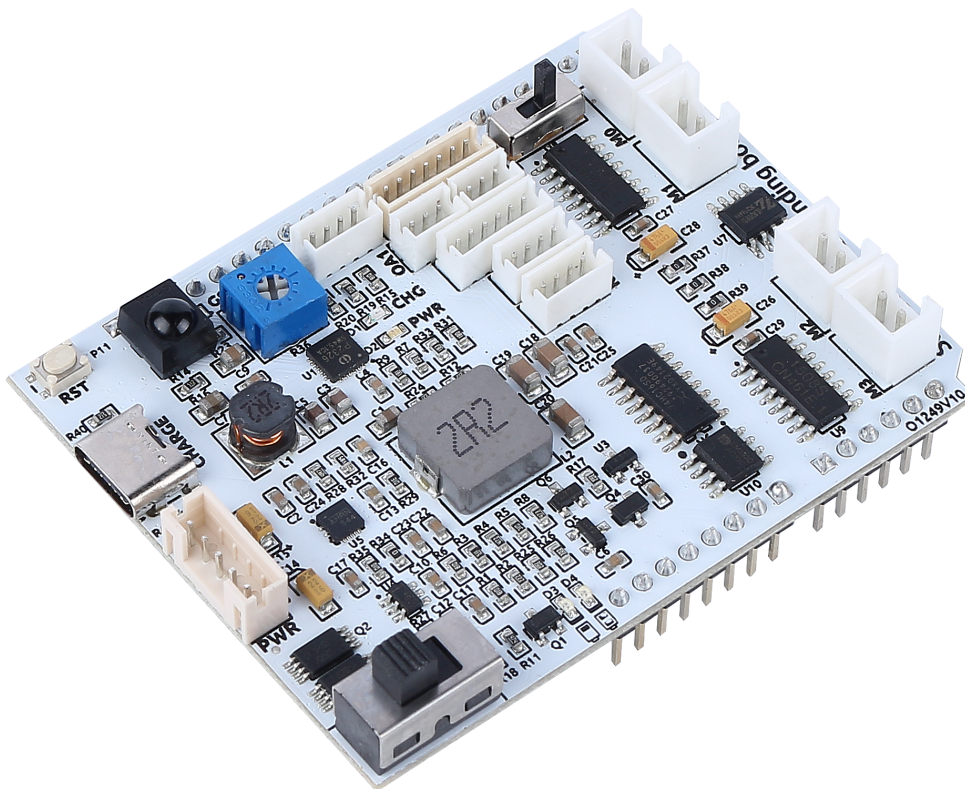


- FLASH MEMORY: 32 KB (ATmega328P) of which 0.5 KB used by bootloader
- SRAM: 2 KB (ATmega328P)
- EEPROM: 1 KB (ATmega328P)
- CLOCK SPEED: 16 MHz
- LED\_BUILTIN: 13
- LENGTH: 68.6 mm
- WIDTH: 53.4 mm
- WEIGHT: 25 g
- I2C Port: A4(SDA), A5(SCL)

#### What's More

- [Arduino IDE](#)
- [Arduino Programming Language Reference](#)
- [Download and Install Arduino IDE 2.0](#)
- [ATmega328P Datasheet](#)

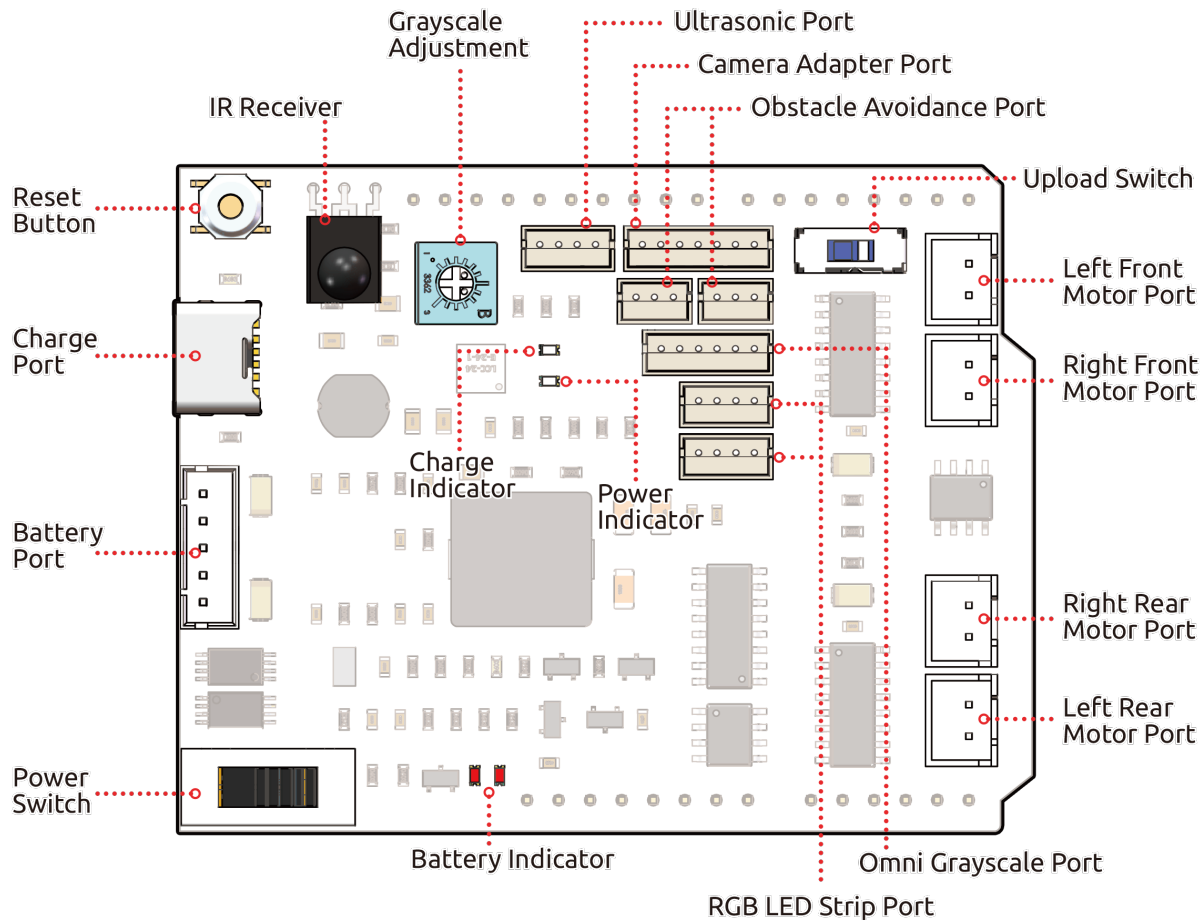
## 2.2 Zeus Car Shield



This is an all-in-one expansion board designed for Arduino by SunFounder, which contains various module ports such as motor, light bar, obstacle avoidance, grayscale, ESP32 CAM and ultrasonic module. There is also a built-in HS0038B IR receiver for remote control.

This expansion board also has a built-in charging circuit, which can charge the battery with PH2.0-5P interface, and the estimated charging time is 130 minutes.

### Zeus Car Shield Pinout



- **Reset Button**
  - Press this button to reset the program on the Arduino board.
- **Charge Port**
  - After plugging into the 5V/2A USB-C port, it can be used to charge the battery for 130min.
- **Battery Port:**
  - 6.6V~8.4V PH2.0-5P power input.
  - Powering the Zeus Car Shield and Arduino board at the same time.
- **Power Switch**
  - Slide to ON to power on the Zeus Car Shield.
- **IR Receiver**
  - This is an HS0038B IR receiver with the signal pin connected to pin 2 of the Arduino board.

- *Grayscale Module Related*

- Grayscale adjustment potentiometer: used to set the reference voltage for Omni grayscale module
- Grayscale Port: Used to connect Omni Grayscale module.

- **Indicators**

- **Charge Indicator:** Glows red when the shield is charging through the USB-C port.
- **Power Indicator:** Glows green when the power switch is in the “ON” position.
- **Battery Indicator:** Two orange indicators represent different battery levels. They flash during charging and turn off when the battery needs charging.

- *Ultrasonic Port*

- To connect the ultrasonic module, both Trig & Echo pins are connected on pin 10 of the Arduino board.

- *Camera Adapter Port*

- The Camera Adapter Board port.

- *Obstacle Avoidance Port*

- Used for connecting two IR obstacle avoidance modules, the signal pins of the two ports are connected to Q0 and Q1 of 74HC165.

- *RGB LED Strip Port*

- For connecting 2 RGB LED Strips, the three pins of the strip are connected to 12, 13 and 11 respectively.

- **Run Switch**

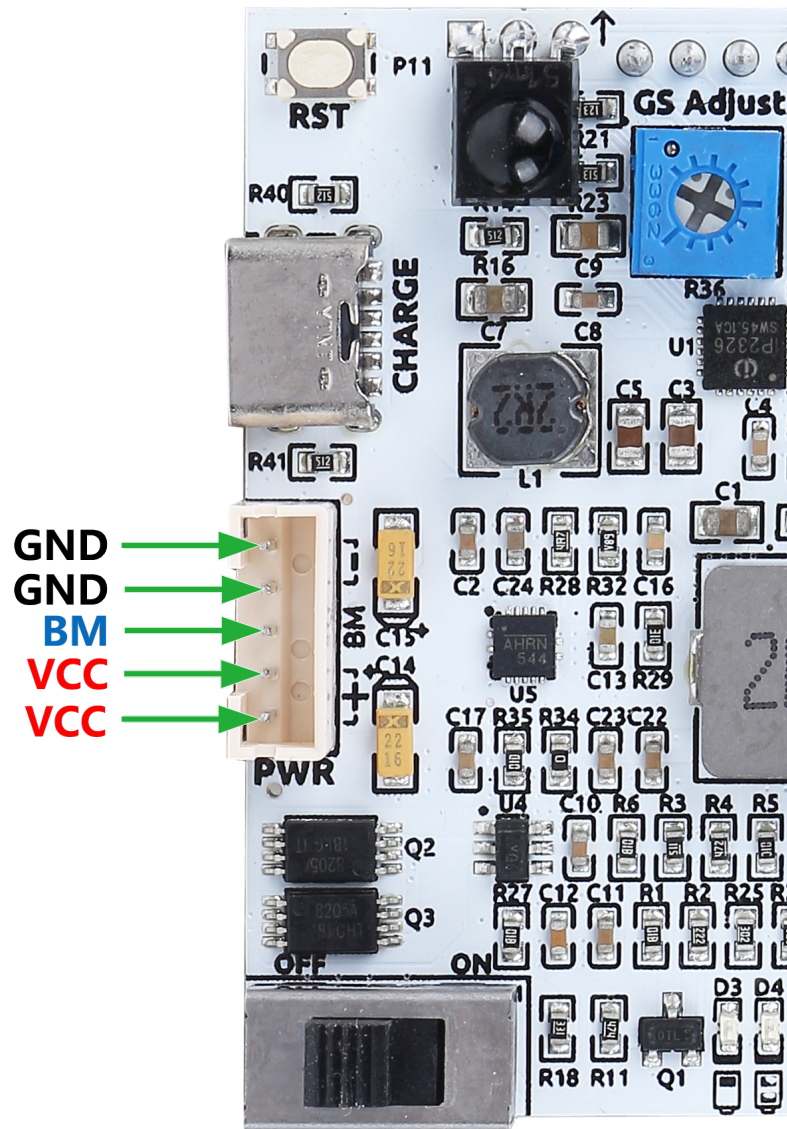
- When you need to use the camera, you need to toggle this switch to the other side so that the ESP32-CAM can communicate with the Arduino board.

- *Motor Port*

- 4 groups of motor ports.

## 2.2.1 Battery Port

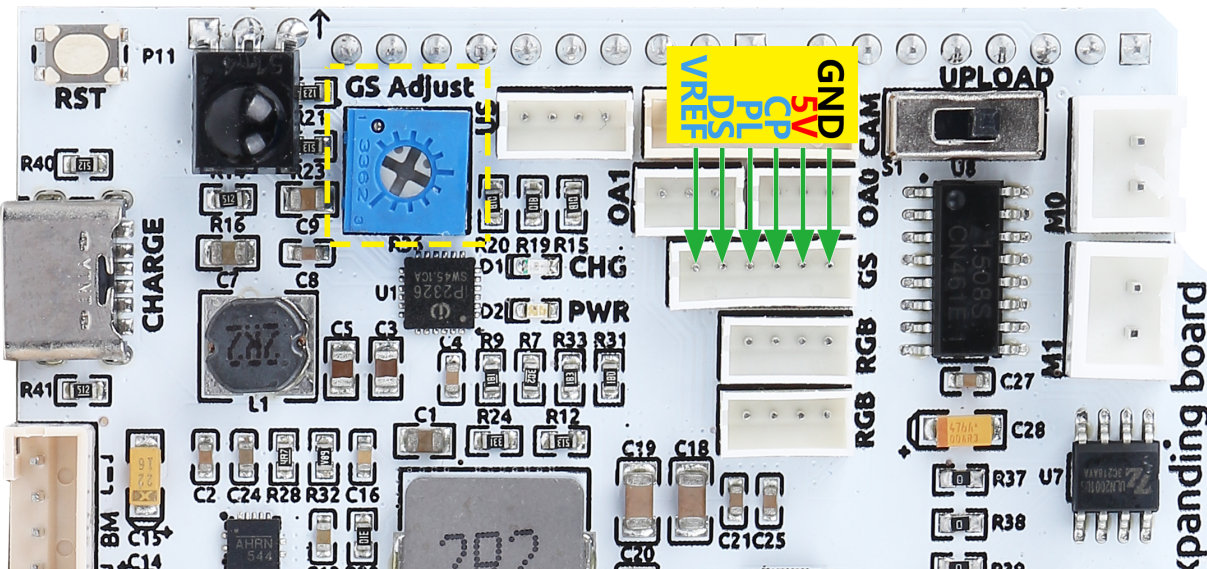
Here is the pinout diagram for the battery interface. The type is PH2.0-5P, and the power input range is 6.6V~8.4V.



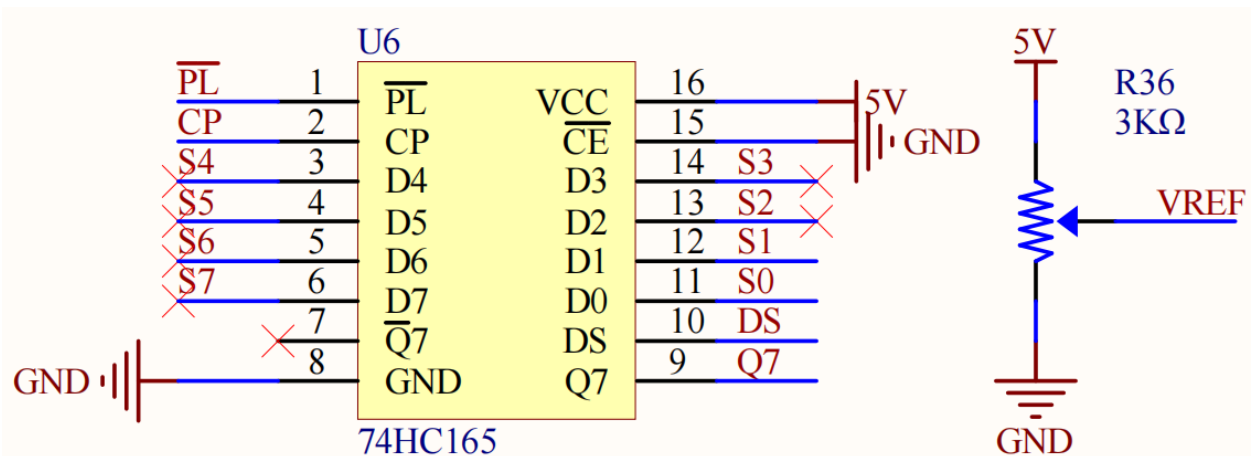


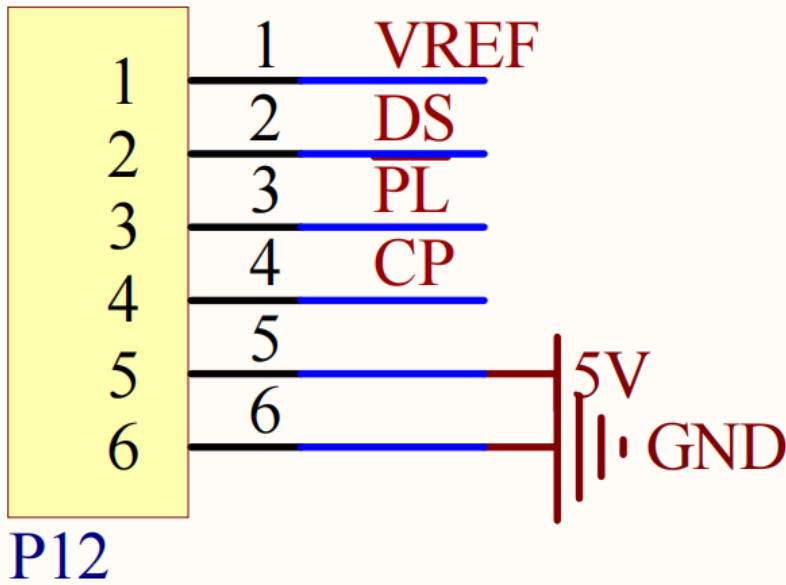
## 2.2.2 Grayscale Module Related

A blue potentiometer on the Zeus Car Shield is used to adjust the grayscale module's sensitivity to different environments by setting the reference voltage for the grayscale module. Through the VREF pin, the grayscale module receives the set reference voltage.



Here is the schematic diagram. The values of the grayscale module are transferred from the 74HC165 chip to the Arduino board. Since the grayscale module itself comes with a 74HC165 chip, the cascade of these two chips will transfer 16 bits of data to the Arduino board - the first 8 bits are grayscale sensor data, and the last two are IR obstacle avoidance data.



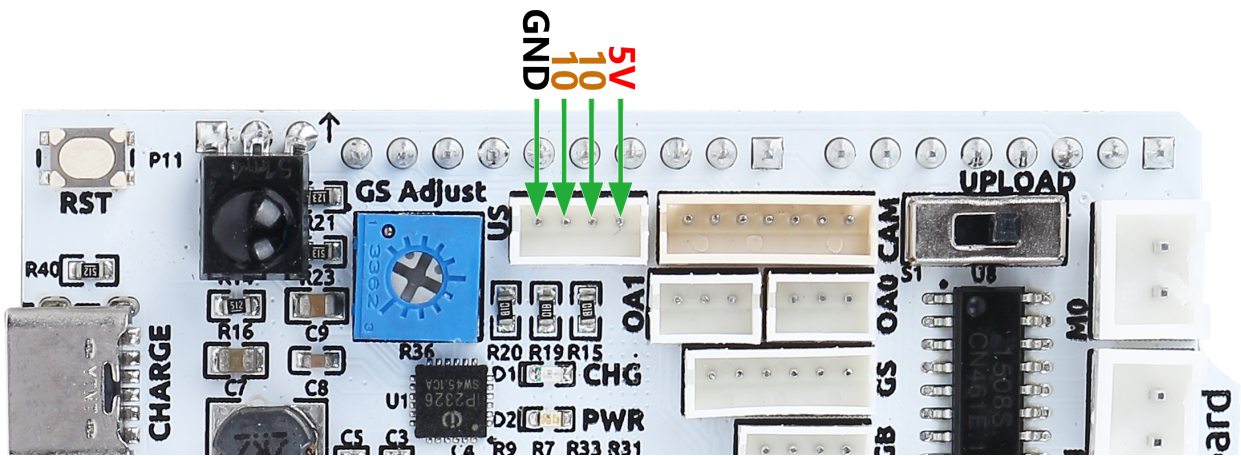


The pin mapping of this port on the Arduino board is shown below.

Arduino Board	Zeus Car Shield
7	74HC165 Q7
8	74HC165 CP
~9	74HC165 PL

### 2.2.3 Ultrasonic Port

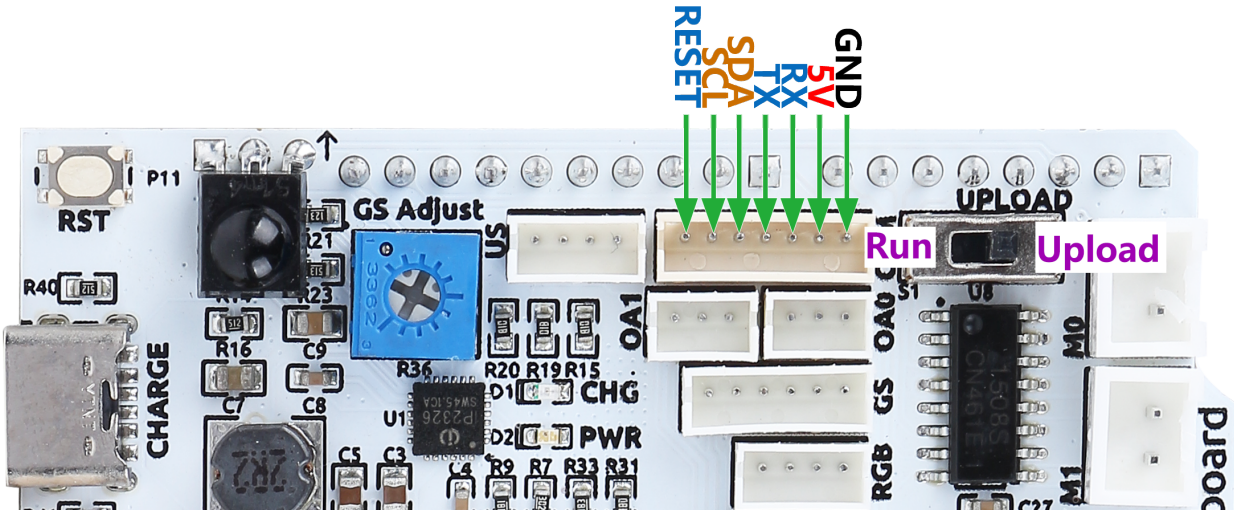
This is the pinout for the ZH1.5-4P ultrasonic port, with the Trig & Echo pins connected to pin 10 of the Arduino board.



## 2.2.4 Camera Adapter Port

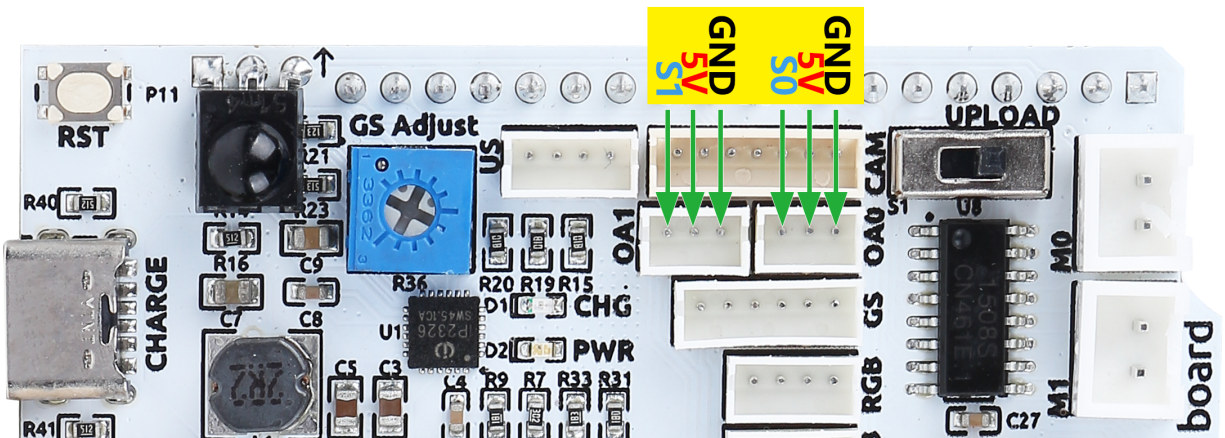
The camera adapter interface pin diagram is shown here, the type is ZH1.5-7P.

- TX and RX are used for ESP32 CAM.
- SDA and SCL are for QMC6310.



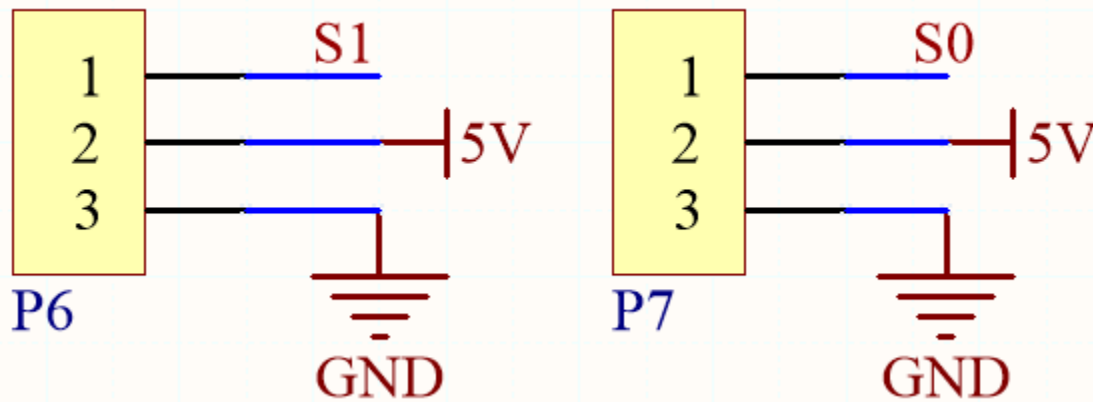
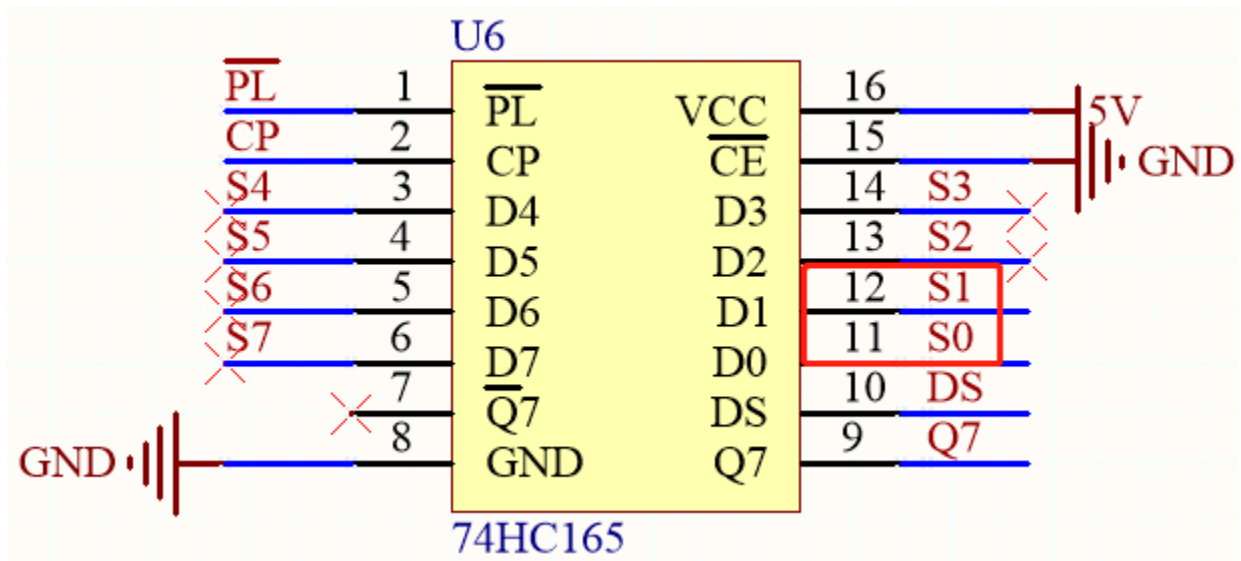
## 2.2.5 Obstacle Avoidance Port

Below is the pinout diagram of the two ZH1.5-3P obstacle avoidance ports.



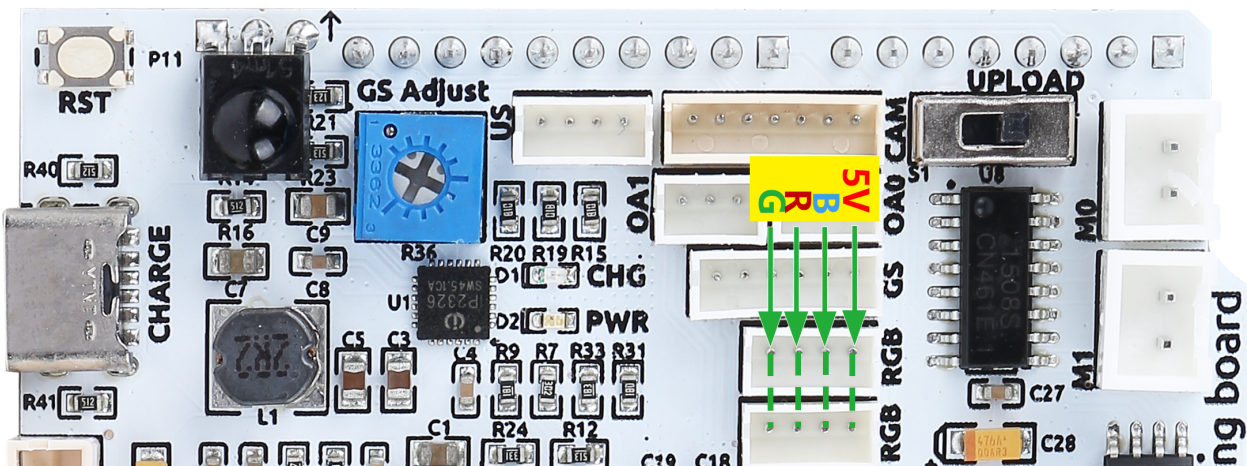
As shown in the circuit diagram, S1 and S0 refer to Q0 and Q1 on the 74HC165 chip. Two 74HC165 cascades transfer the data of the two IR obstacle avoidance modules and the grayscale sensor to the Arduino board. The first eight bits are grayscale sensor data, and the last two bits represent IR obstacle avoidance data.





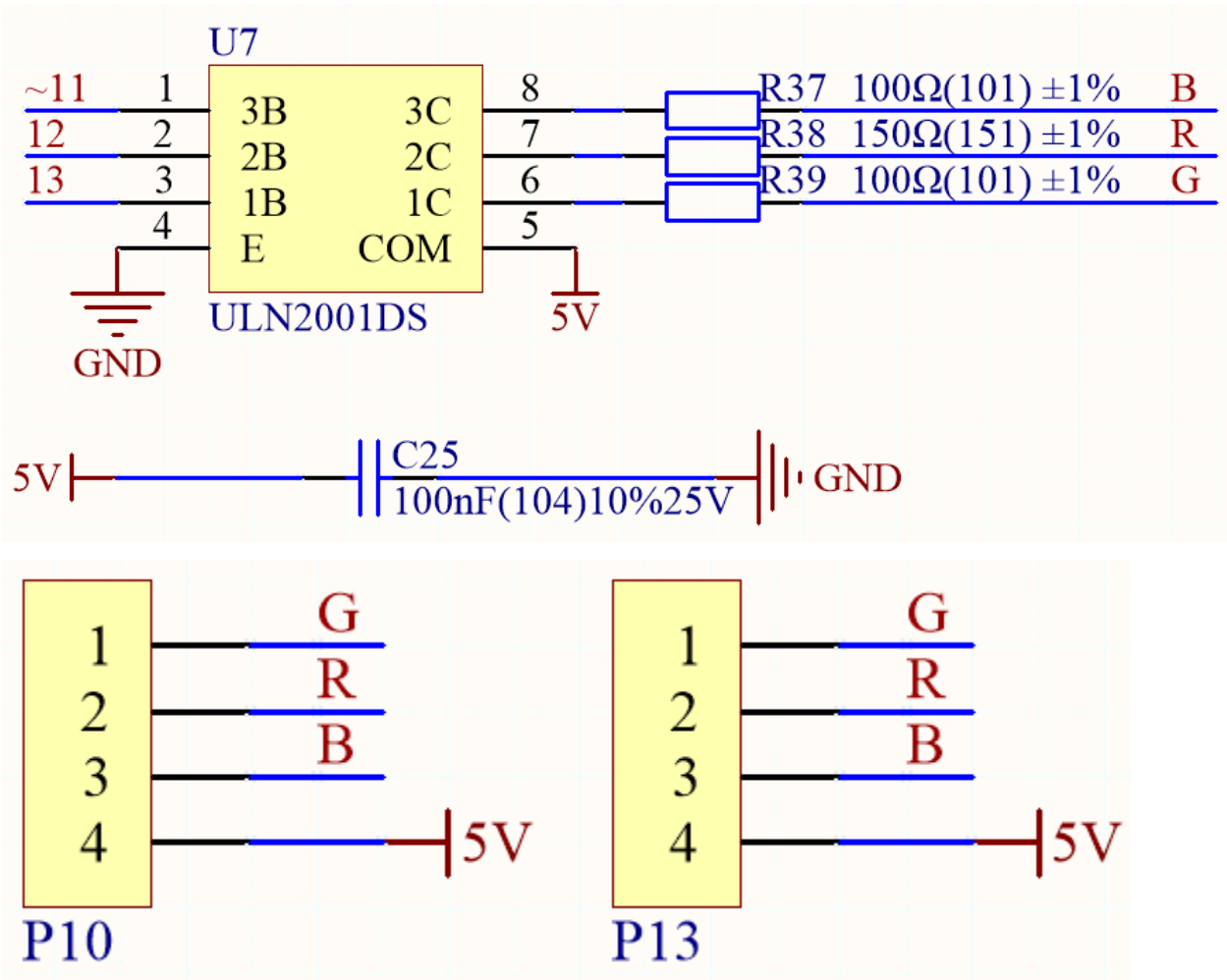
## 2.2.6 RGB LED Strip Port

Below is the pinout diagram of the two RGB LED Strip, they are connected in parallel and the pinouts are the same.



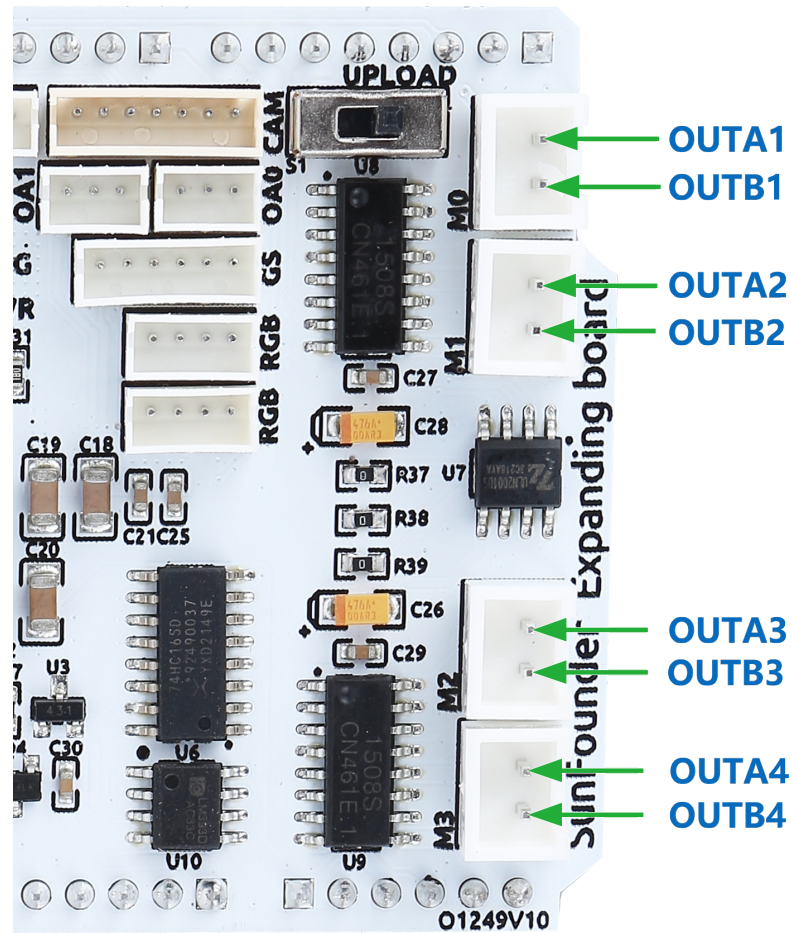
Here is the schematic.





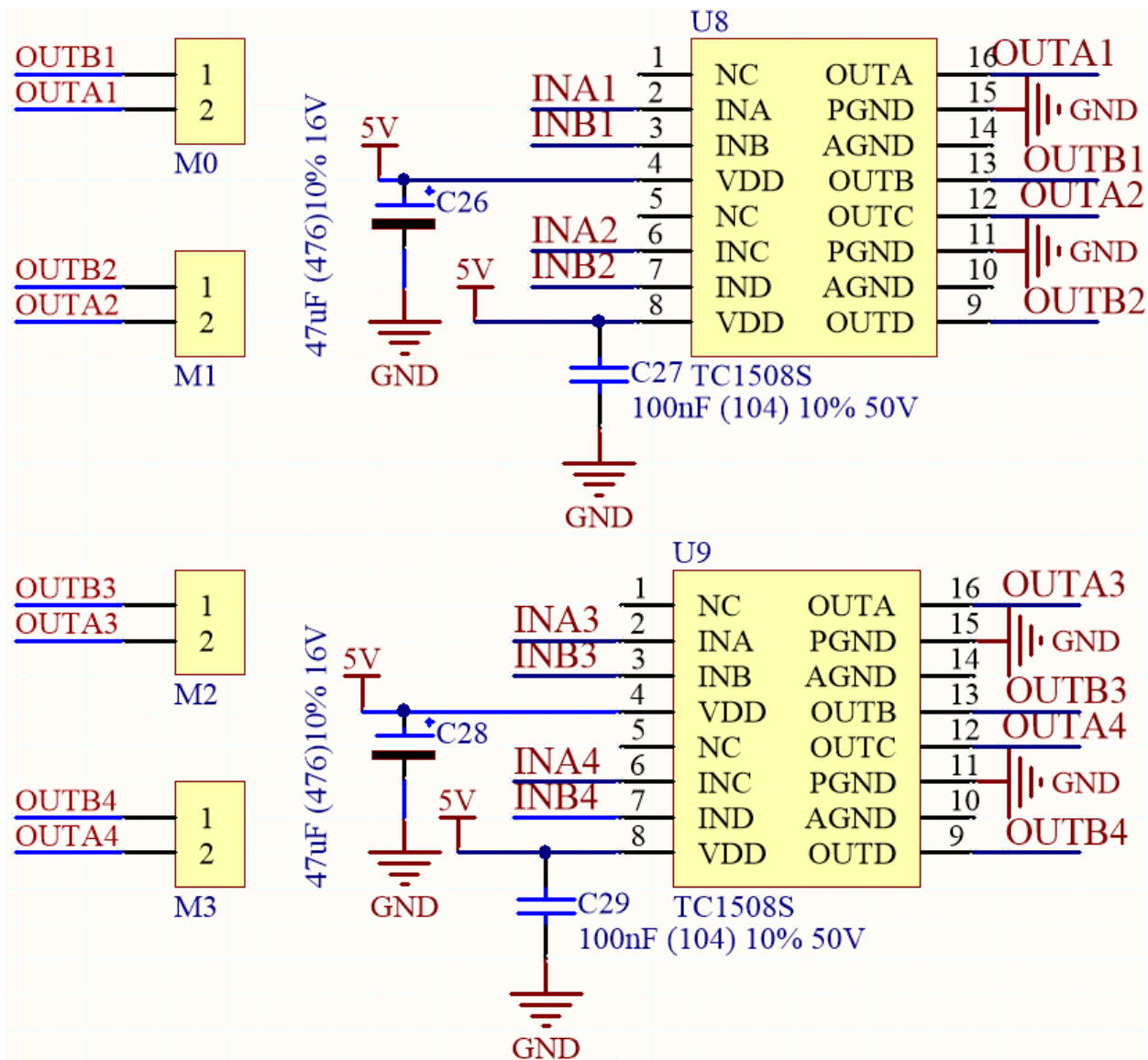
### 2.2.7 Motor Port

Here is the pinout of the 4 sets of motor ports.



These 4 sets of motors are driven by 2 TC1508S chips, which is a dual-channel motor driver chip with a maximum continuous output current of up to 1.8A/per channel.

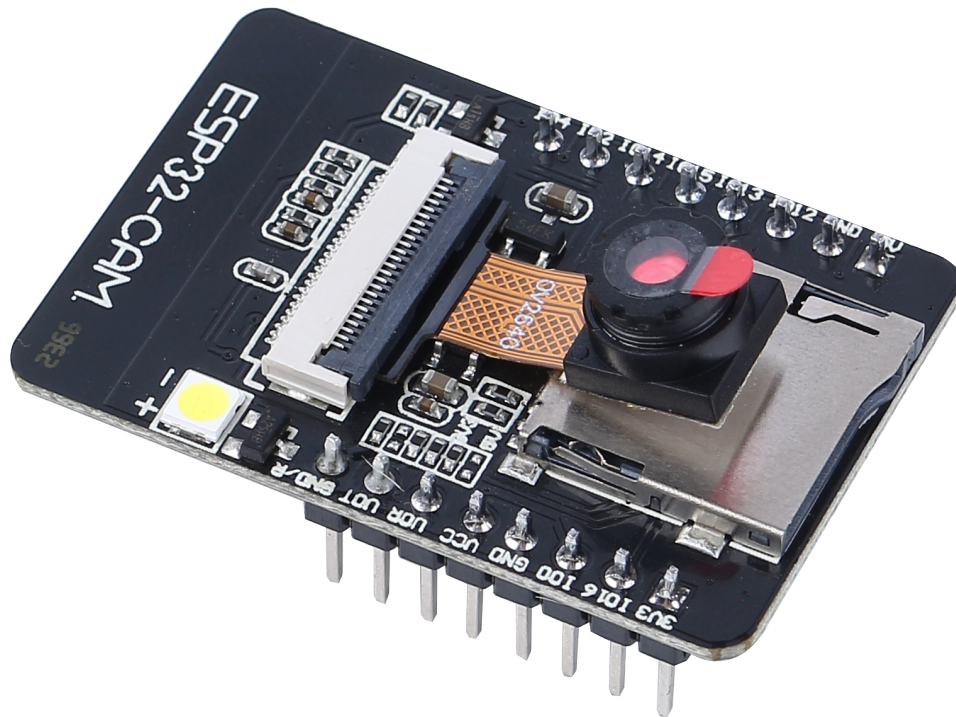
Here is the schematic.



So the corresponding control pins of the 4 motor ports are shown below.

Arduino Board	Zeus Car Shield
~3	OUTA1
4	OUTB1
~5	OUTA2
~6	OUTB2
A0	OUTB4
A1	OUTA4
A2	OUTB3
A3	OUTA3

## 2.3 ESP32 CAM



The ESP32-CAM is a very small camera module with the ESP32-S chip that costs approximately \$10. Besides the OV2640 camera, and several GPIOs to connect peripherals, it also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve to clients.

The module can work independently as the smallest system, with a size of only 27\*40.5\*4.5mm, and a deep sleep current as low as 6mA.

ESP32-CAM can be widely used in various IoT applications, suitable for home smart devices, industrial wireless control, wireless monitoring, QR wireless identification, wireless positioning system signals and other IoT applications. It is an ideal solution for IoT applications.

### Technical Specifications

Module Model	ESP32-CAM
Package	DIP-16
Size	27*40.5*4.5±0.2mm
SPI Flash	default 32Mbit
RAM	Internal 520KB + External 8MB PSRAM
Bluetooth	Bluetooth 4.2 BR/EDR and BLE standards
Wi-Fi	802.11 b/g/n/e/i
Support Interfaces	UARTSPII2CPWM
Support TF Card	up to 4G
IO Pins	9
Serial Port Speed	default 115200 bps
Image Output Format	JPEG(only OV2640 support),BMP,GRAYSCALE
Spectrum range	2400 ~2483.5MHz

continues on next page

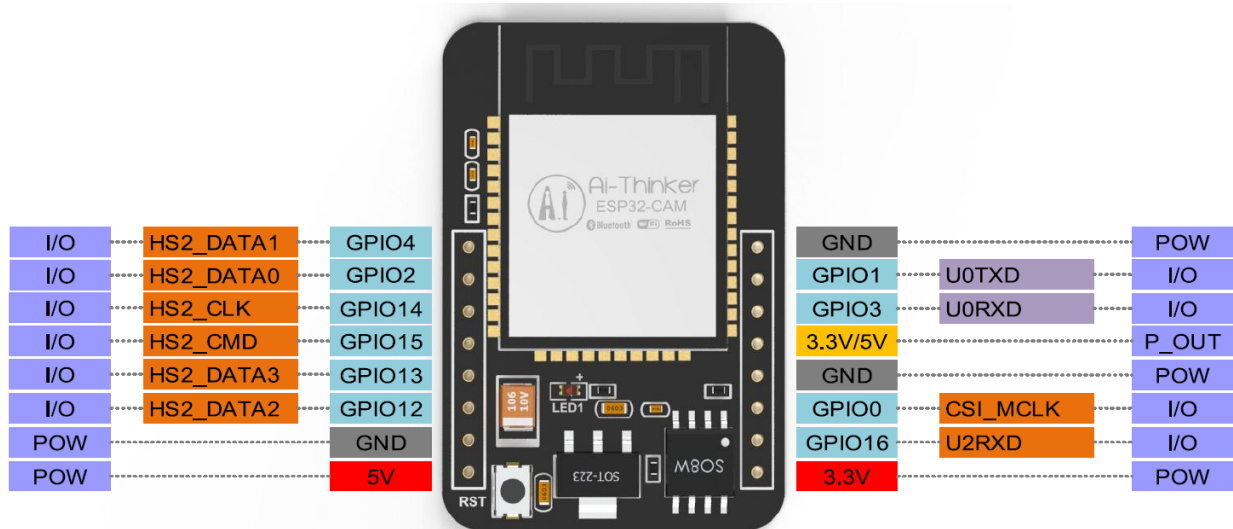


Table 1 – continued from previous page

Antenna Type	On-board PCB antenna, gain 2dBi
Transmit Power	802.11b: 17±2 dBm (@11Mbps)
	802.11g: 14±2 dBm (@54Mbps)
	802.11n: 13±2 dBm (@MCS7)
Receive Sensitivity	CCK, 1 Mbps: -90dBm,
	CCK, 11 Mbps: -85 dBm
	6 Mbps (1/2 BPSK): -88 dBm
	54 Mbps (3/4 64-QAM): -70dBm
	MCS7 (65 Mbps, 72.2 Mbps): -67dBm
Power Consumption	Flash off: 180mA@5V,
	Flash on and brightness to maximum: 310mA@5V
	Deep-sleep: the lowest power consumption can reach 6mA@5V
	Moderm-sleep: minimum 20mA@5V
	Light-sleep: minimum 6.7mA@5V
Security	WPA/WPA2/WPA2-Enterprise/WPS
Power supply range	4.75-5.25V
Operating Temperature	-20 °C ~ 70 °C
Storage Environment	-40 °C ~ 125 °C , < 90%RH

### ESP32-CAM Pinout

The following figure shows the ESP32-CAM pinout (AI-Thinker module).



- There are three **GND** pins and three pins for power: 3.3V, 5V and either 3.3V or 5V.
- **GPIO 1** and **GPIO 3** are the serial pins. You need these pins to upload code to your board.
- Additionally, **GPIO 0** also plays an important role, since it determines whether the ESP32 is in flashing mode or not. When **GPIO 0** is connected to **GND**, the ESP32 is in flashing mode.
- The following pins are internally connected to the microSD card reader:
  - GPIO 14: CLK
  - GPIO 15: CMD
  - GPIO 2: Data 0
  - GPIO 4: Data 1 (also connected to the on-board LED)

- GPIO 12: Data 2
- GPIO 13: Data 3

### Note

- Please make sure that the input power of the module is at least 5V 2A, otherwise the picture may have water lines.
- The ESP32 GPIO32 pin controls the camera power. When the camera is working, please pull GPIO32 low.
- Since GPIO0 is connected to the camera XCLK, please leave GPIO0 in the air when using it, and do not connect it to high or low level.
- The default firmware is already included in the factory, and no additional download is provided. Please be careful if you need to re-burn other firmware.

### Document

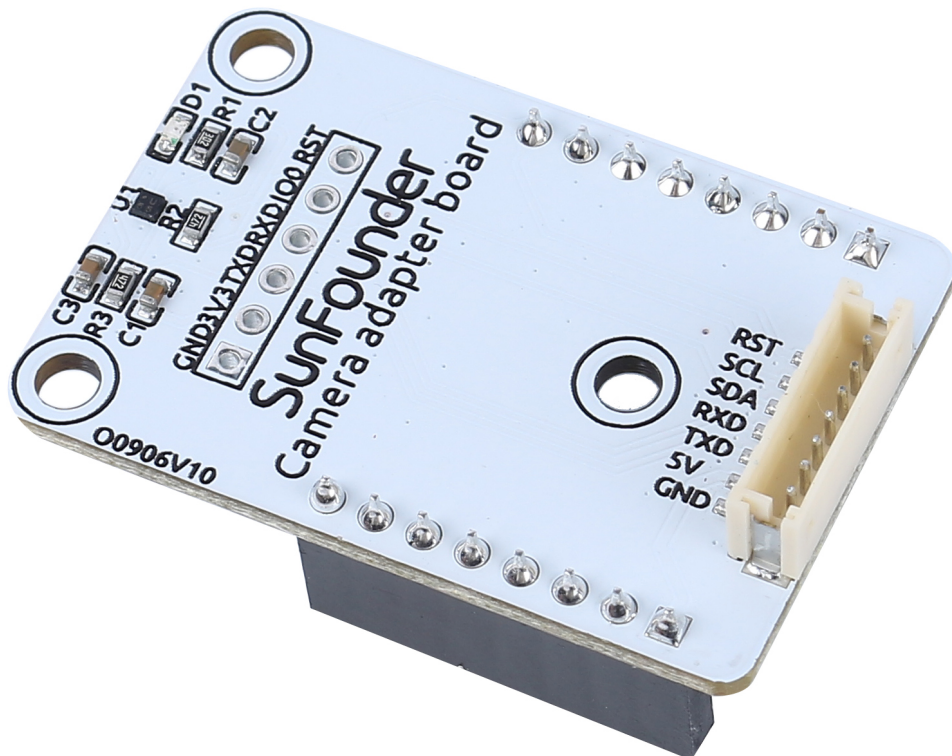
- Schematic diagram:
- Camera specification (English version):

---

**Note:** All information above comes from

---

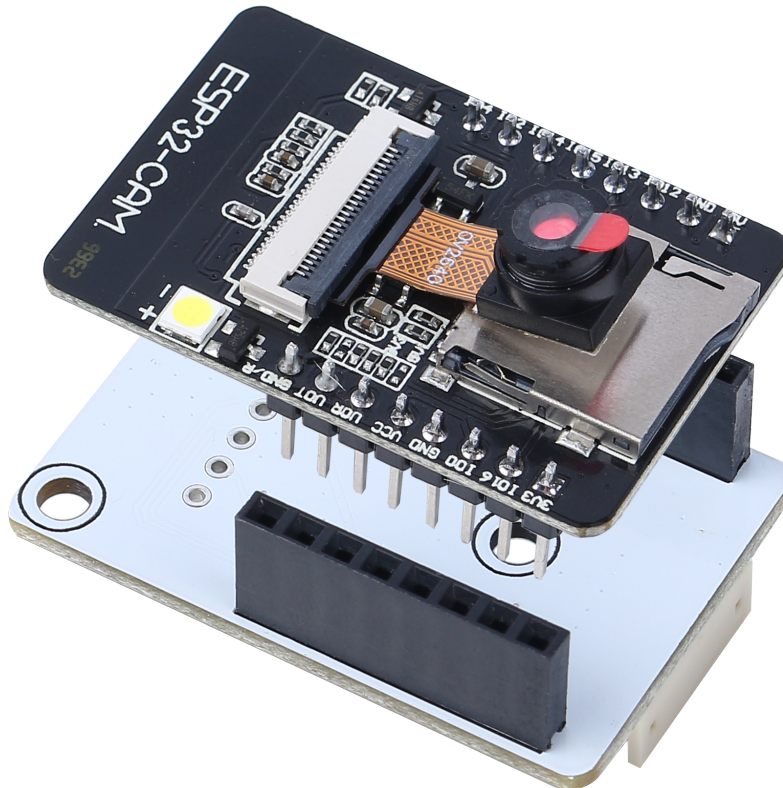
## 2.4 Camera Adapter Board



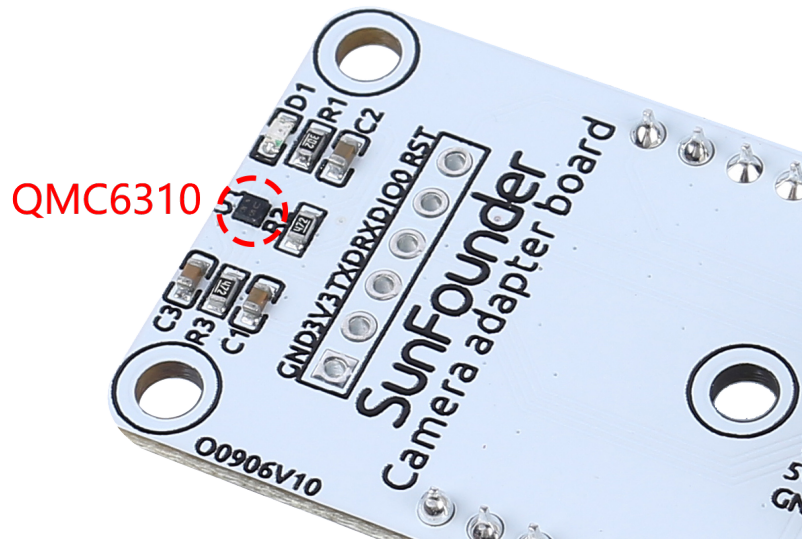
- **RST:** Used to reset the ESP32-CAM.
- **SCL:** Serial data pin for QMC6310

- **SDA**: Serial clock pin of the QMC6310
- **RXD**: The RXD of ESP32-CAM, you need to upload code to ESP32-CAM through these two serial pins, RXD and TXD.
- **TXD**: TXD of ESP32-CAM
- **5V**: 5V DC Supply Input
- **GND**: Ground Input

The Camera Adapter Board, as the name implies, is an expansion board for the ESP-32 CAM, used to expand the ESP32-CAM so that it can be secured to the robot, and can be easily wired.



Also because the geomagnetic chip QMC6310 is susceptible to interference from motors, we put it on this camera adapter board to keep it as far away from the motors as possible.



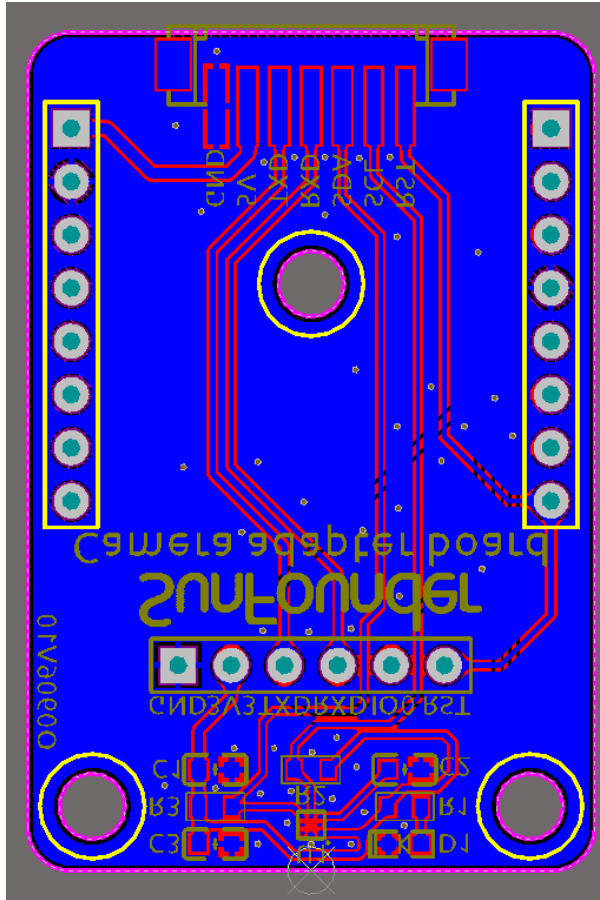
### Features

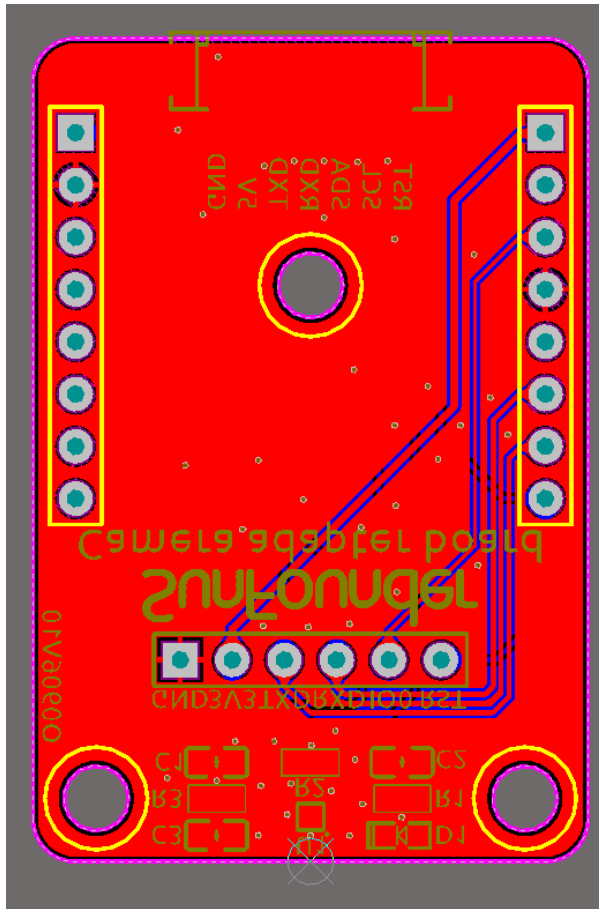
- Working voltage: 5V
- Interface Model: ZH1.5, 7P
- Dimension: 40mm x 27mm x 15mm
- Communication protocol: UART and I2C

### Documents

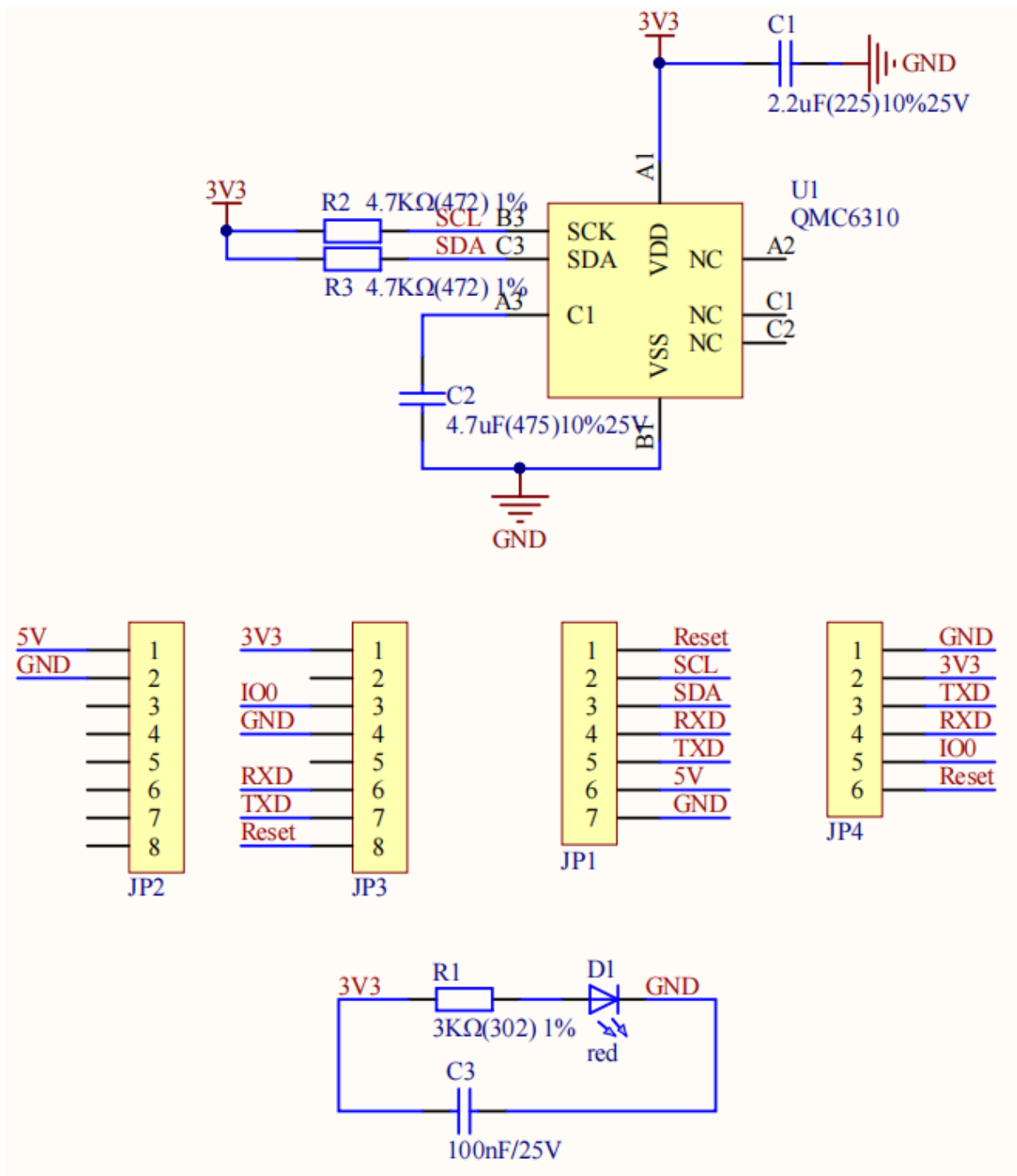
- PCB







- Schematic



### About QMC6310

The QMC6310 is a three-axis magnetic sensor, which integrates magnetic sensors and signal condition ASIC into one silicon chip. This Land Grid Array package (LGA) is targeted for applications such as e-compass, map rotation, gaming and personal navigation in mobile and wearable devices.

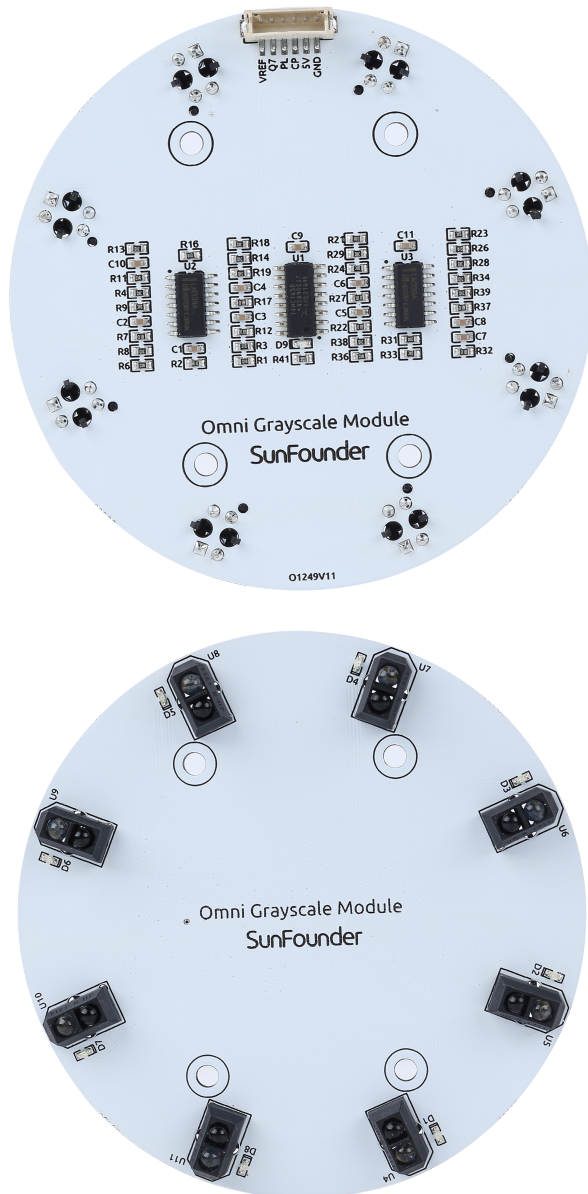
The QMC6310 is based on state-of-the-art, high resolution, magneto-resistive technology. Along with the custom-designed 16-bit ADC ASIC, it offers the advantages of low noise, high accuracy, low power consumption, offset cancellation and temperature compensations. QMC6310 enables 1° to 2° compass heading accuracy. The I<sup>2</sup>C serial bus

allows for easy interface.

The QMC6310 is in a 1.2x1.2x0.53mm<sup>3</sup> surface mount 8-pin LGA package.

•

## 2.5 Omni Grayscale Module



- **VREF**: Reference voltage input pin. The value of each sensor is compared to this reference voltage to determine whether to output high or low.
- **Q7**: Serial output from the last stage
- **PL**: Asynchronous parallel load input (active LOW)



- **CP:** Clock input (LOW-to-HIGH edge-triggered)
- **5V:** 3.3 to 5V DC Supply Input
- **GND:** Ground Input

This is an Omni Grayscale module for line following and edge detection. Omni means omnidirectional, which means that the module has 8 TCRT5000 transmitting sensors distributed in a circle to detect black lines in any direction.

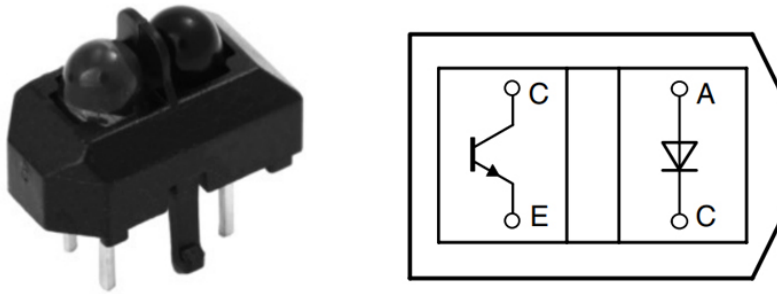
This allows a robot car like the Zeus Car with Mecanum wheels to follow the line at various angles without having the head of the car facing forward.

The sensitivity of the module in the current environment can be adjusted by modifying the VREF value. In the Zeus Car Shiled, the blue potentiometer is used to adjust the value of the VREF pin.

### Working Principle

•

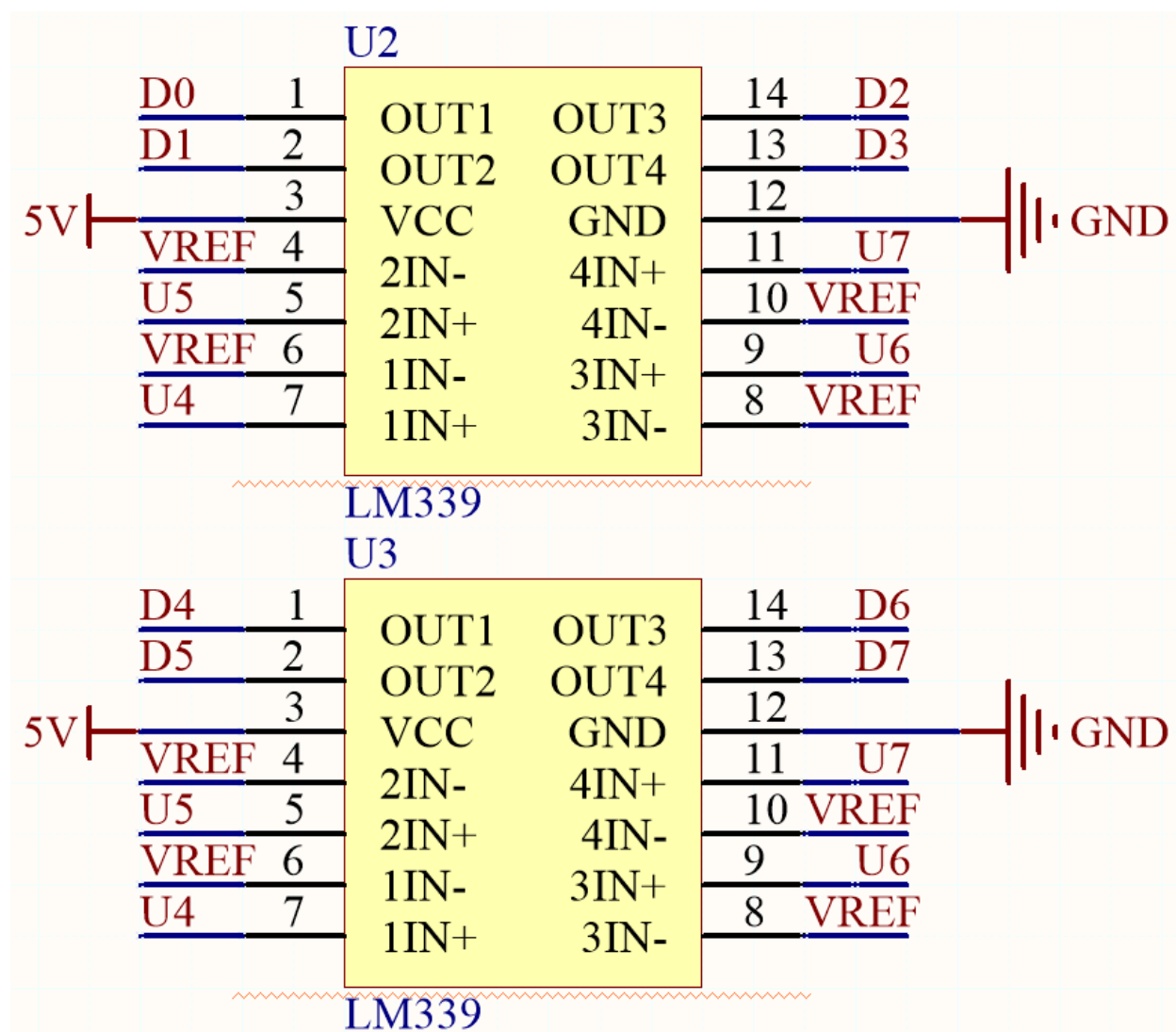
In the module, eight TCRT5000 transmitter sensors are integrated, which are based on infrared optical reflection and contain an infrared light-emitting diode and a phototransistor covered with a lead material to block visible light.



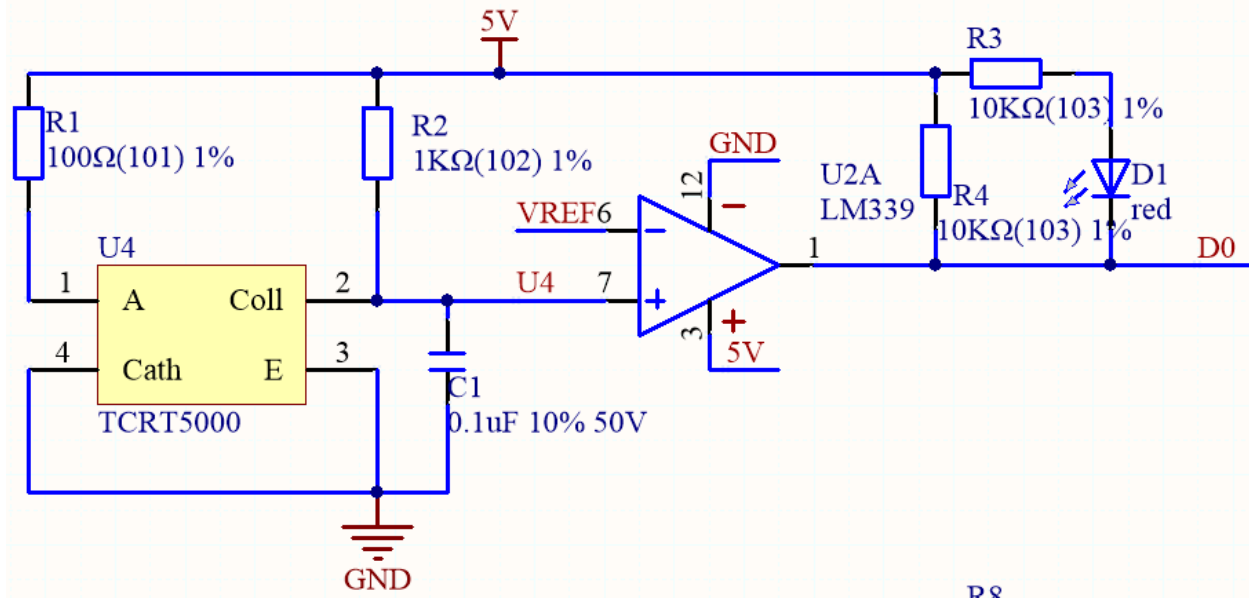
When working, the infrared light-emitting diode of TCRT5000 continuously emits infrared light (invisible light) with a wavelength of 950nm. When the emitted infrared light is not reflected back by the obstacle or the reflection intensity is insufficient, the phototransistor does not work. When the infrared light is reflected with sufficient intensity and received by the phototransistor at the same time, the phototransistor is in working condition and provides output.

•

There are 2 LM339 chips on this Omni Grayscale Module, containing a total of 8 differential comparators. These differential comparators are used to compare the current sensor value with a reference value to determine whether to output a high or low level. This way you know if a black line is detected.



Below is a schematic of one of the channels.



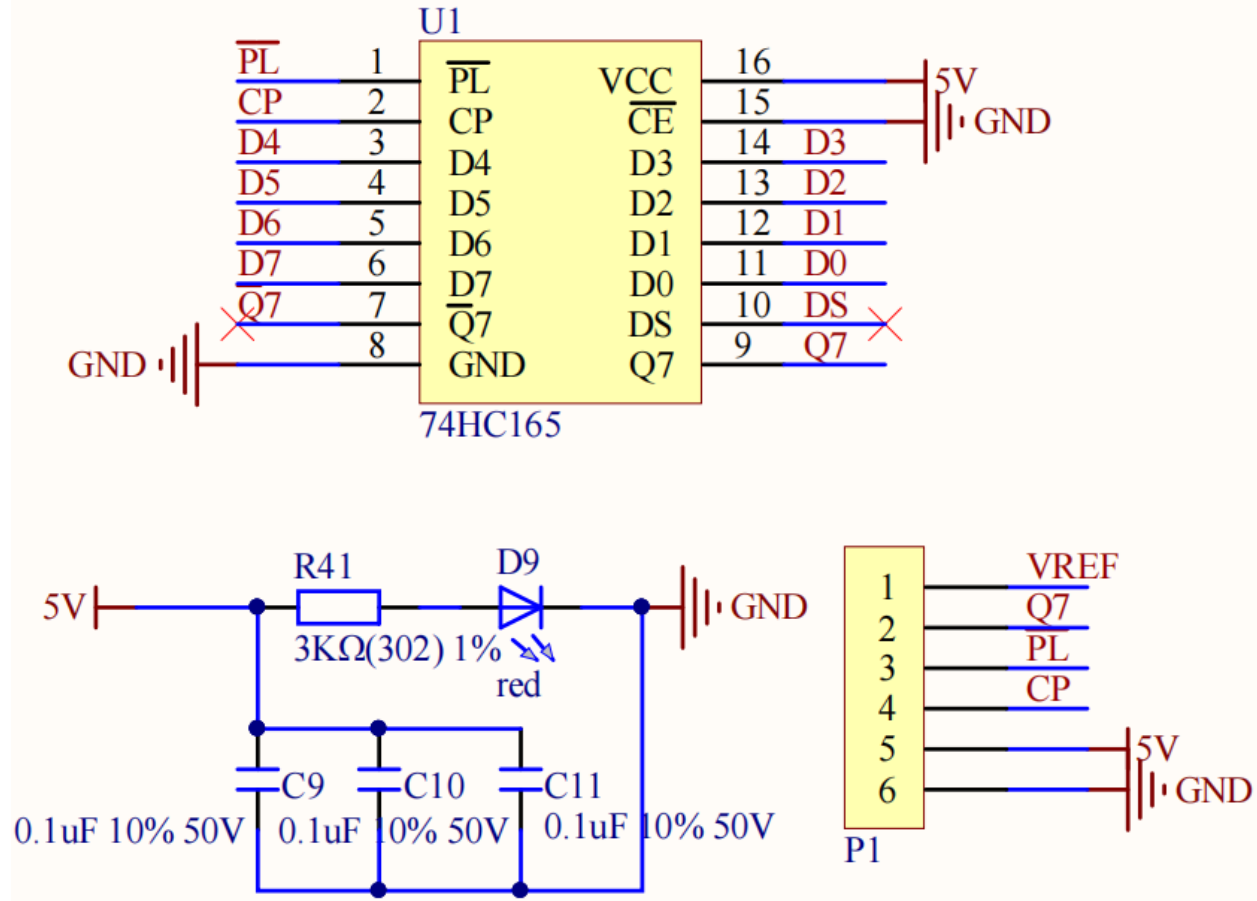
- Set a reference voltage at the VREF pin (this reference voltage is set via a potentiometer on the Zeus Car Shield) and add this reference voltage to the inverting input (-) of the comparator.
- Add the collector of the TCRT5000 sensor's phototransistor to the in-phase input (+) of the comparator.
- When the infrared ray emitted by the TCRT5000 sensor is not reflected back or the reflection intensity is insufficient, the photosensitive transistor does not work and the collector is connected to the pull-up resistor to 5V at this time, so the in-phase input (+) of the comparator is greater than the inverting input (-).
- The comparator output is high and the indicator does not light up. And vice versa.
- Since the black surface absorbs light, it reflects less infrared light, so on the black surface, the comparator outputs high and the indicator does not light up.
- The white surface reflects more infrared rays, and the photosensitive transistor conducts, so the value of the in-phase input is smaller than the inverted input, and the comparator outputs low, and the indicator lights up.

These 8 sensor data are transferred to the Arduino board via 74HC165 (8-bit parallel input serial output shift register).

•

The 74HC165 is an 8-bit parallel input serial output shift register, which can get mutually exclusive serial outputs (Q0 and Q7) at the final stage. When the parallel read (PL) input is low, the parallel data input from D0 to D7 port will be read into the register asynchronously. And when PL is high, the data will enter the register serially from the DS input, moving one bit to the right on the rising edge of each clock pulse (Q0 → Q1 → Q2, etc.). Using this feature, the parallel-to-serial expansion can be achieved by simply binding the Q7 output to the next level of DS input.

The clock input of the 74HC165 is a "gated or" structure that allows one of the inputs to be used as a low active clock enable (CE) input. The CP and CE pin assignments are independent and can be interchanged for wiring convenience if necessary. CE is allowed to rise from low to high only when CP is high. CP or CE should be set high before the PL rising edge to prevent data displacement in the active state of PL.



### Features

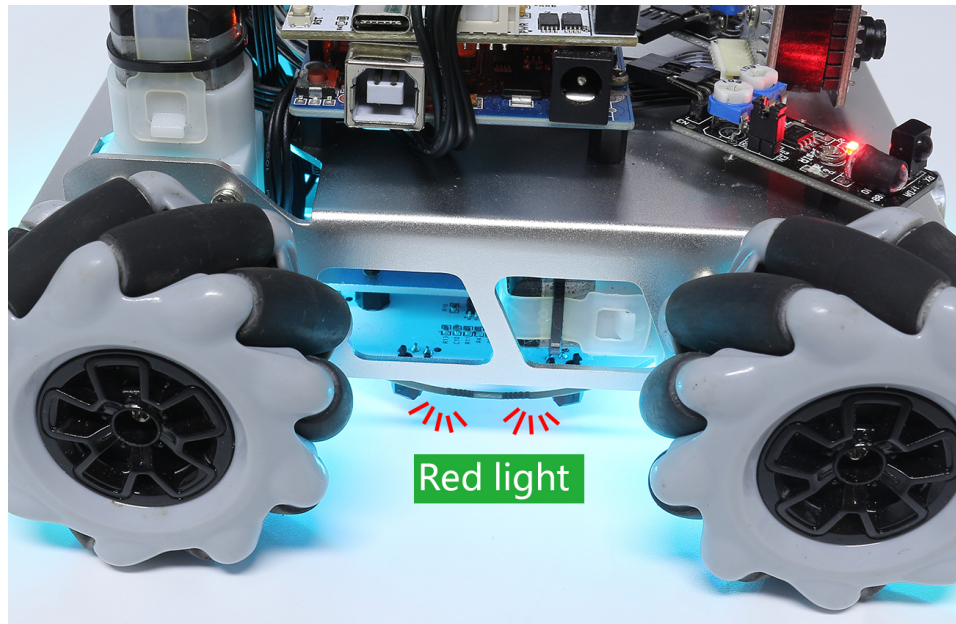
- Operating Voltage: 3.3 ~ 5V
- Output: digital (on/off)
- Asynchronous 8-bit parallel load
- Synchronous serial input
- Detection Threshold: adjustable by VREF pin
- Sensor Type: TCRT5000
- Connector Model: ZH1.5-6P
- Operating Temperature: -10 °C to +50 °C
- Dimensions: 80mm x 80mm

### Calibrate the Module

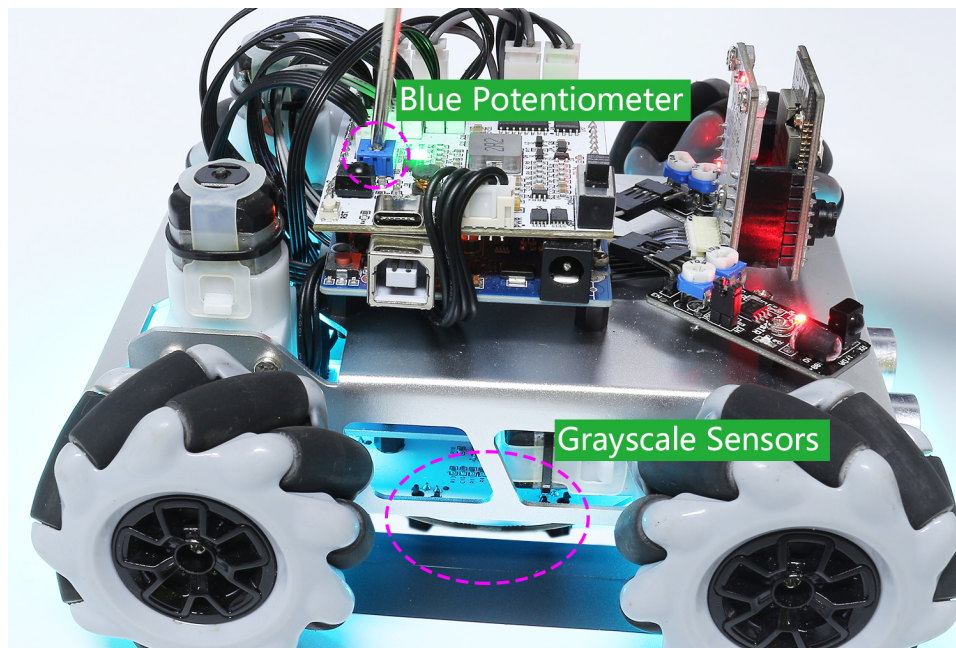
Since each subfloor has different grayscale values, the factory-set grayscale threshold may not be appropriate for your current environment, so you will need to calibrate this module before use. It is recommended that you need to calibrate it whenever the floor color changes a lot.

- Place the Zeus Car on white surface and turn the potentiometer until the gray sensor light is just illuminated.





- Now let the two greyscale sensors on the side be located just between the black line and white surface, and slowly turn the potentiometer until the signal indicator just goes off.



- You can move repeatedly over the the black line and white surface to make sure that the lights of the greyscale sensor are off when they are between the the black line and white surface and on when they are on the white surface, indicating that the module is successfully calibrated.

## 2.6 Ultrasonic Module



- **TRIG:** Trigger Pulse Input
- **ECHO:** Echo Pulse Output
- **GND:** Ground
- **VCC:** 5V Supply

This is the HC-SR04 ultrasonic distance sensor, providing non-contact measurement from 2 cm to 400 cm with a range accuracy of up to 3 mm. Included on the module is an ultrasonic transmitter, a receiver and a control circuit.

You only need to connect 4 pins: VCC (power), Trig (trigger), Echo (receive) and GND (ground) to make it easy to use for your measurement projects.

### Features

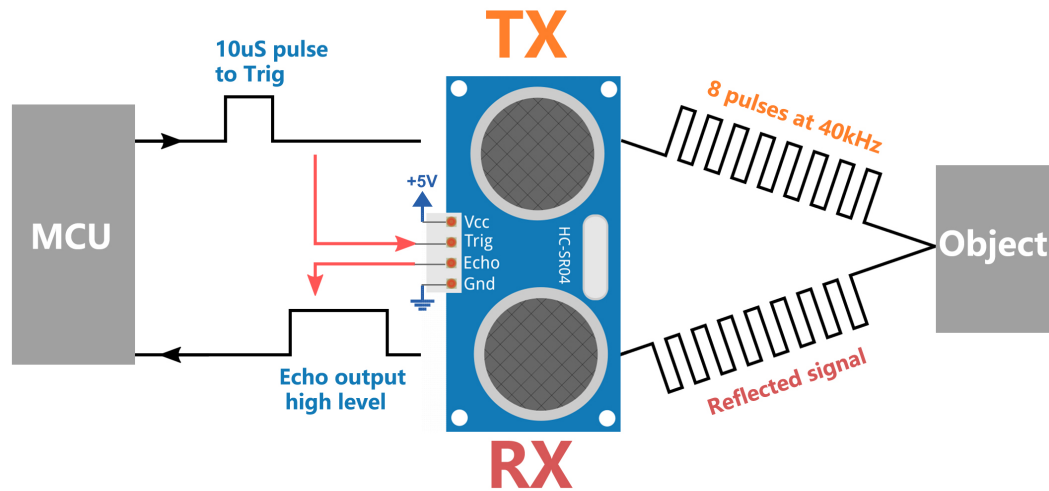
- Working Voltage: DC5V
- Working Current: 16mA
- Working Frequency: 40Hz
- Max Range: 500cm
- Min Range: 2cm
- Trigger Input Signal: 10uS TTL pulse
- Echo Output Signal: Input TTL level signal and the range in proportion
- Connector: XH2.54-4P
- Dimension: 46x20.5x15 mm

### Principle

The basic principles are as follows:

- Using IO trigger for at least 10us high level signal.
- The module sends an 8 cycle burst of ultrasound at 40 kHz and detects whether a pulse signal is received.

- Echo will output a high level if a signal is returned; the duration of the high level is the time from emission to return.
- Distance = (high level time x velocity of sound (340M/S)) / 2



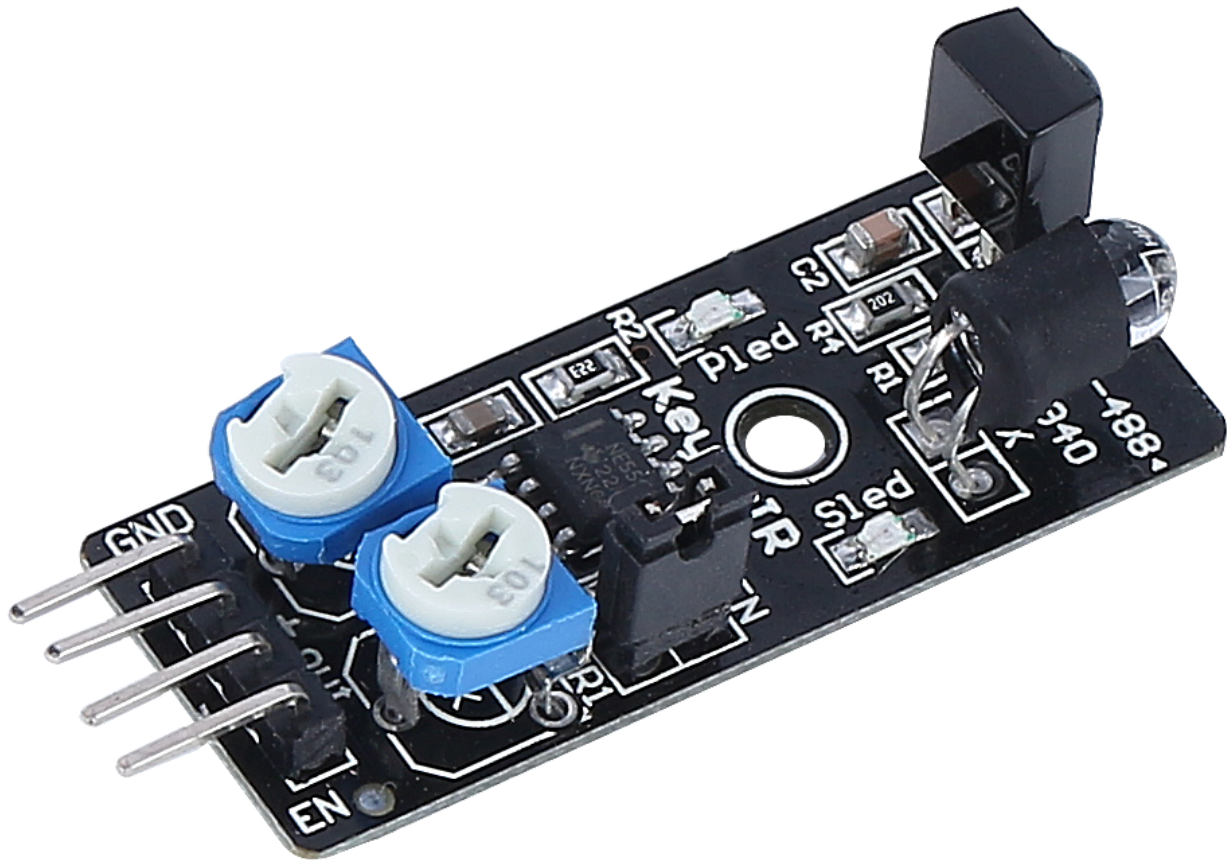
Formula:

- $\text{us} / 58 = \text{centimeters distance}$
- $\text{us} / 148 = \text{inch distance}$
- $\text{distance} = \text{high level time} \times \text{velocity (340M/S)} / 2$

#### Application Notes

- This module should not be connected under power up, if necessary, let the module's GND be connected first. Otherwise, it will affect the work of the module.
- The area of the object to be measured should be at least 0.5 square meters and as flat as possible. Otherwise, it will affect results.

## 2.7 IR Obstacle Avoidance Module



- **GND:** Ground Input
- **+**: 3.3 to 5V DC Supply Input
- **Out:** Signal output pin, default is high and output low when obstacle is detected
- **EN:** Module enable pin. When it is low, the module is working and connected to GND by jumper cap by default.

This is a common IR obstacle avoidance module that uses a pair of IR transmitting and receiving components. Basically, the transmitter emits infrared light, and when the detection direction encounters an obstacle, the infrared light is back and received by the receiver tube. At this time, the indicator lights up. After circuit processing, it outputs a low level signal.

Sensing distance 2-40cm, with excellent anti-interference ability. There are different reflectivity levels in different colors of objects, so the darker the object, the closer to black the detection distance is short. The 2-30cm range of this sensor is detected against a white wall.

When the enable pin is at a low level, the module works. As soon as the jumper cap is plugged in, the EN pin is connected to GND, and the module is always working. If you want to control the EN pin by code, you need to remove the jumper cap.





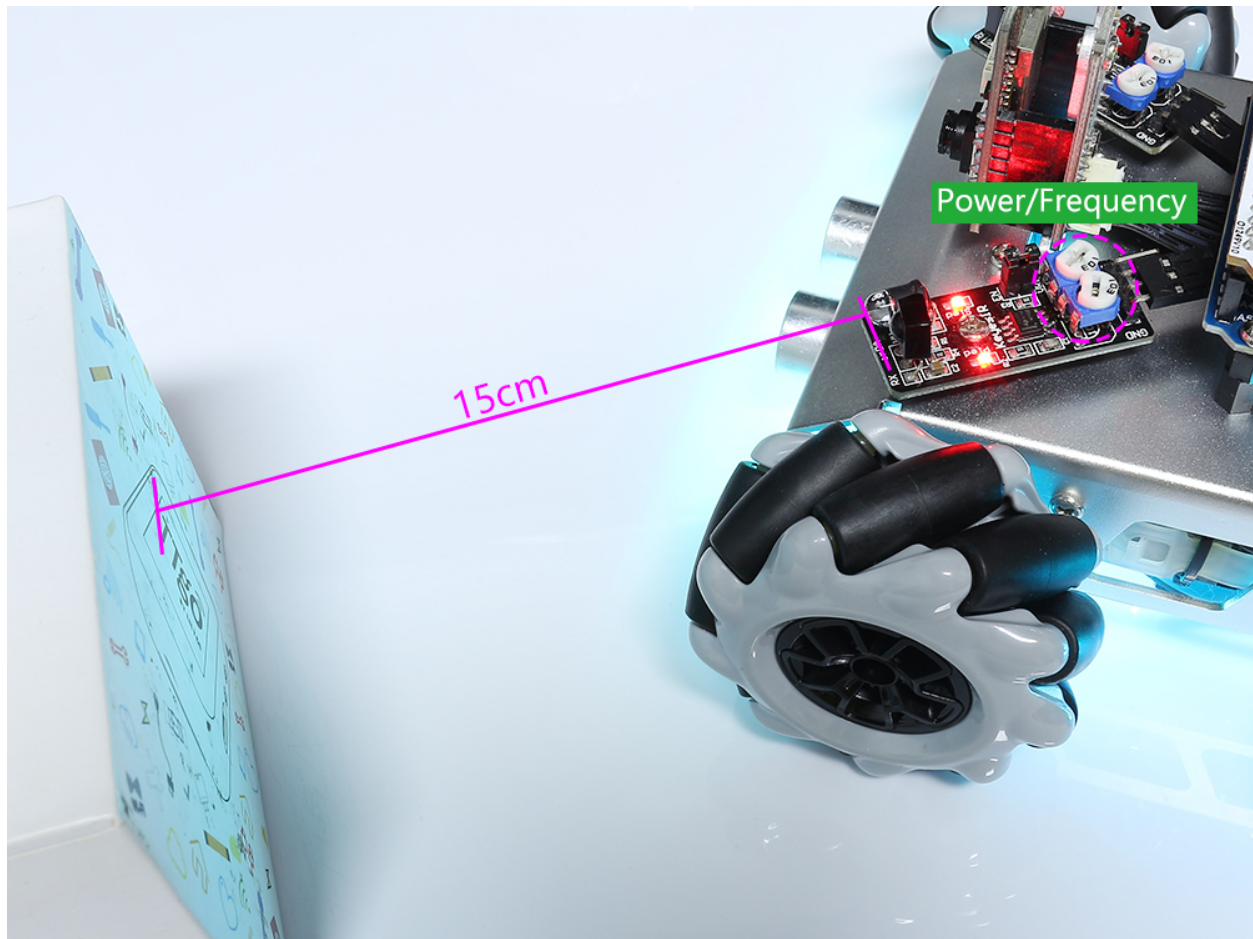
### Adjust the detection distance

Due to the different light environment, the factory-set detection distance may not be applicable, so you need to adjust its actual detection distance before use.

There are two potentiometers on the module, one for adjusting the transmitting power and one for adjusting the transmitting frequency, and by adjusting these two potentiometers you can adjust its effective distance.

You can place a white obstacle in front of the module at the distance you want, adjust a potentiometer on the module until the indicator light on the module just lights up, and then repeatedly move the obstacle to see if the indicator light on the module lights up at the distance you need.

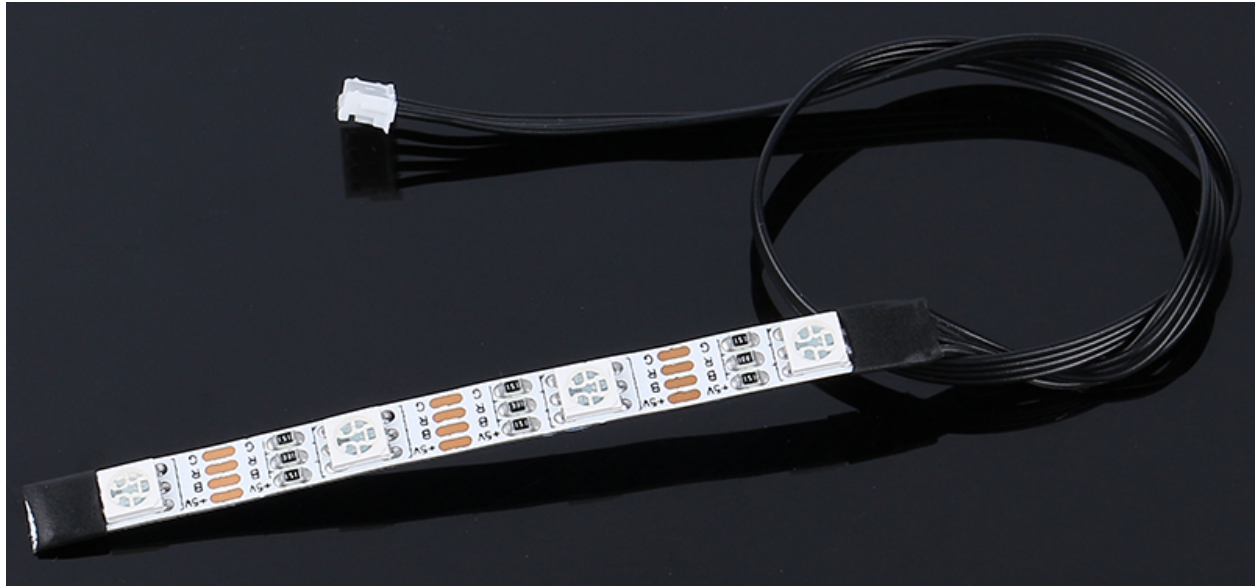
If in the position you need, the indicator does not light up or is a long light does not go out, you can then go to adjust another potentiometer.



## Features

- operating voltage: 3.3 V to 5 V
- output: digital (on/off)
- detection threshold: adjustable by 2 potentiometers
- distance range: 2 to 40 cm
- adjustment R5: frequency adjustment 38 kHz (already optimized)
- adjustment R6: IR LED duty cycle adjustment (already optimized)
- operating temperature: -10 °C to +50 °C
- effective angle: 35°
- I/O interface: 4 wire interface (- / + / S / EN)
- dimensions: 45 x 16 x 10 mm
- weight: 9 g

## 2.8 4 RGB LEDs Strip



- **+5V**: Common anode of the three LEDs and needs to connect to DC 5V
- **B**: Cathode of the blue LED
- **R**: Cathode of the red LED
- **G**: Cathode of the green LED

The RGB strip is composed of 4 RGB LEDs and has the ability to produce any shade of any color, it is a mixture of three primary colors; red, blue and green.

The RGB LEDs used are 5050, connected in a common anode way. Each LED is an independent circuit, can be cut along the switch at will, without damaging other parts. Made of FPC board, backed with double-sided adhesive, can be bent and secured at will.

### Features

- Work Voltage: DC5V
- Color: Full color RGB
- Working Temperature: -15-50
- RGB Type: 5050RGB
- Current: 150mA (single circuit)
- Power: 1.5W
- Light Strip Thickness: 2mm
- Light Strip Width: 5.5mm
- Cable: ZH1.5-4P, 25cm, 28AWG, Black

## 2.9 18650 Battery



- **VCC:** Battery positive terminal, here there are two sets of VCC and GND is to increase the current and reduce the resistance.
- **Middle:** To balance the voltage between the two cells and thus protect the battery.
- **GND:** Negative battery terminal.

This is a custom battery pack made by SunFounder consisting of two 18650 batteries with a capacity of 2200mAh. The connector is PH2.0-5P, which can be charged directly after being inserted into the Zeus Car shield.

### Pins Out

#### Features

- Battery charge: 5V/2A
- Battery output: 5V/5A
- Battery capacity: 3.7V 2000mAh x 2
- Battery life: 90min
- Battery charge time: 130min
- Connector: PH2.0, 5P



## 2.10 TT Motor



This is a TT DC motor with a gear ratio of 1:48, it comes with 2 x 250mm wires with XH2.54-2P connector.

You can power these motors with 3 ~ 6VDC, but of course, they will go a little faster at higher voltages.

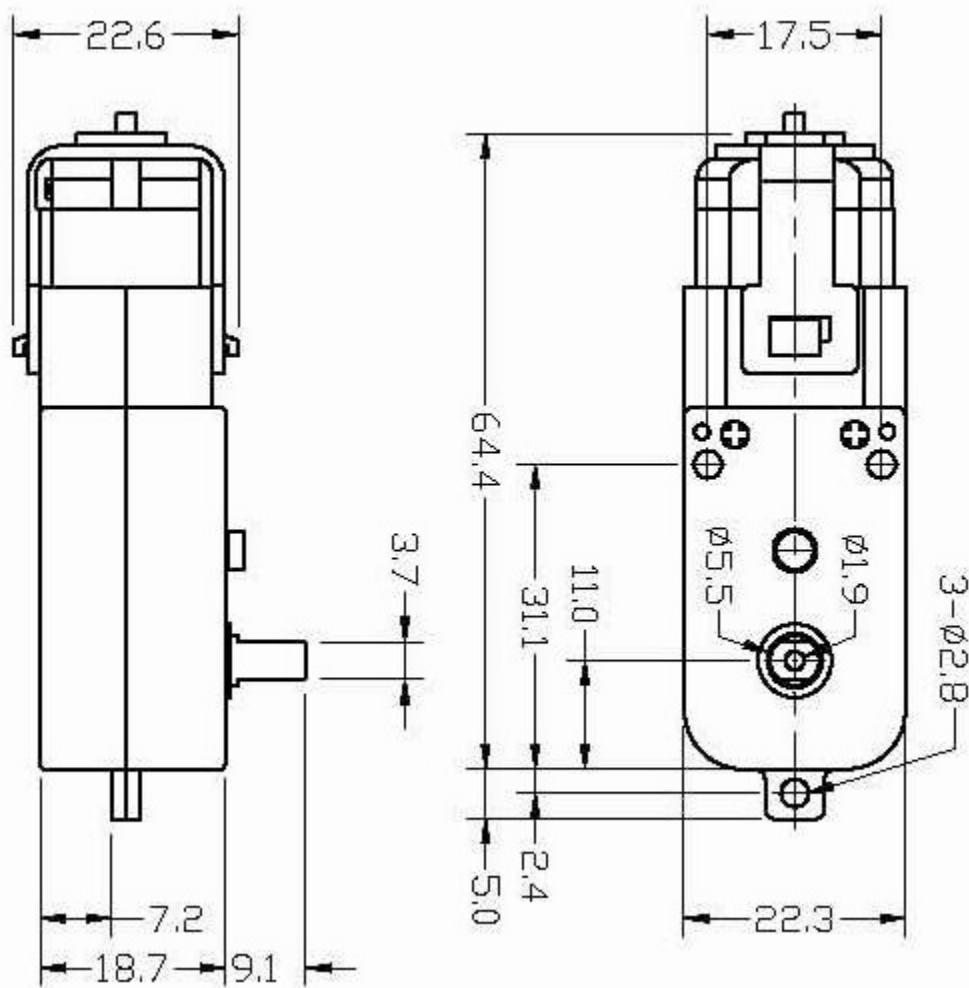
Note that these are very basic motors with no built-in encoder, speed control or position feedback. The voltage goes in and the spin comes out. There will be variation from motor to motor, so if you need precise motion, you'll need a separate feedback system.

### Features

- Rated Voltage: 3~6V
- Continuous No-Load Current: 150mA +/- 10%
- Min. Operating Speed (3V): 90+/- 10% RPM
- Min. Operating Speed (6V): 200+/- 10% RPM
- Stall Torque (3V): 0.4kg.cm
- Stall Torque (6V): 0.8kg.cm
- Gear Ratio: 1:48
- Body Dimensions: 70 x 22.3 x 36.9mm
- Wires: Gray and Black, 24AWG, 250mm
- Connector: XH2.54-2P
- Weight: 30.6g

## Dimensional Drawing

Unit: mm



## 2.11 Mecanum Wheel

What is Mecanum Wheel?



The Mecanum Wheel is an omnidirectional wheel design for a land-based vehicle to move in any direction. It is sometimes called the Swedish wheel or Ilon wheel after its inventor.

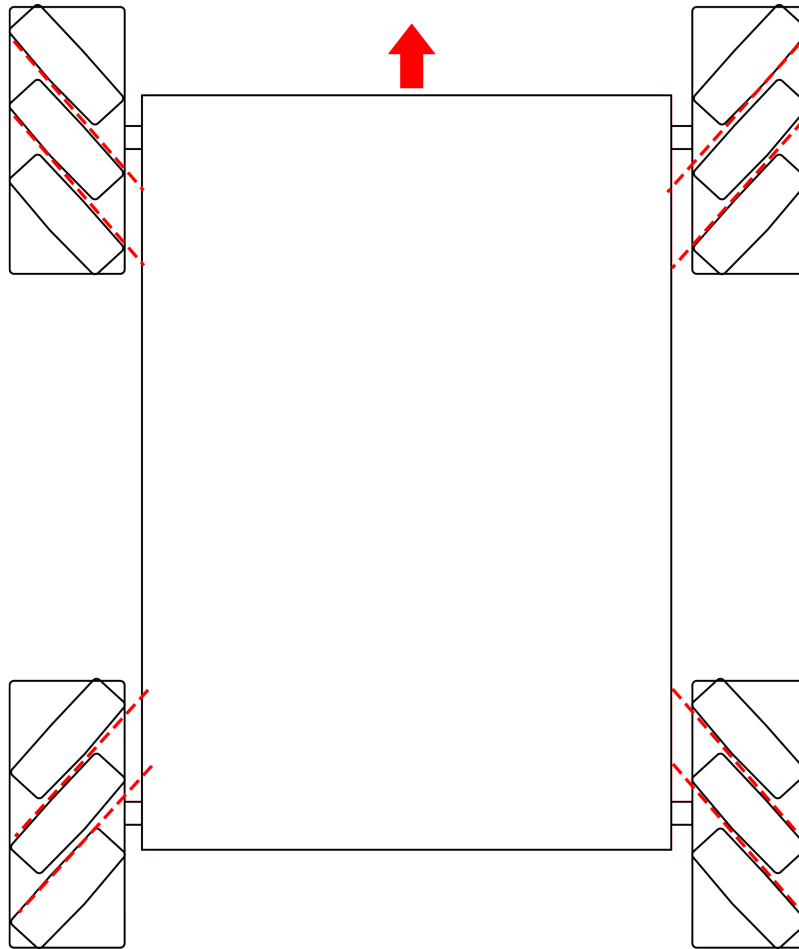
The Mecanum Wheel is a form of tireless wheel, with a series of rubberized external rollers obliquely attached to the whole circumference of its rim. These rollers typically each have an axis of rotation at  $45^\circ$  to the wheel plane and at  $45^\circ$  to the axle line.

Each Mecanum wheel is an independent non-steering drive wheel with its own powertrain, and when spinning generates a propelling force perpendicular to the roller axle, which can be vectored into a longitudinal and a transverse component in relation to the vehicle.

The Mecanum Wheel can be divided into left-hand and right-hand wheels that are mirror images of each other, depending on the angle of  $45^\circ$ .



Mecanum Wheels are usually used in groups of four, as shown in the figure below



**Features**

- Diameter: 60mm
- Thickness: 30.62mm
- Rollers: 9pcs
- Angle: 45°
- Color: Black
- Material: plastic + rubber



### 3.1 Q1: Compilation error: SoftPWM.h: No such file or directory?

If you get a “Compilation error: SoftPWM.h: No such file or directory” prompt, it means you don’t have the SoftPWM library installed.

Please refer to *Install the Required Libraries* to install the two required libraries SoftPWM and IRLremote.

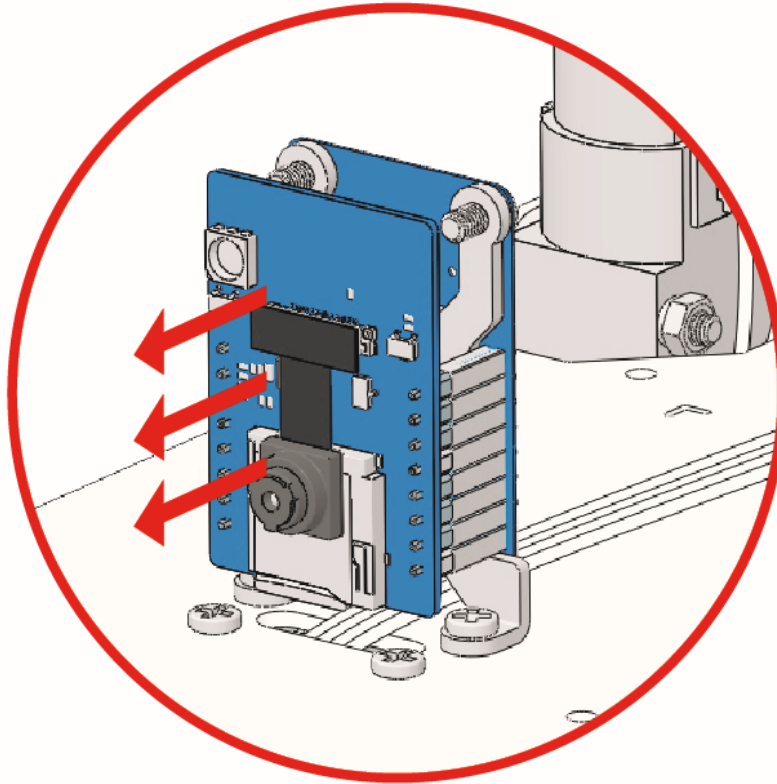
### 3.2 Q2: avrdude: stk500\_getsync() attempt 10 of 10: not in sync: resp=0x6e?

If the following message keeps appearing after clicking the Upload button when the board and port have been selected correctly.

```
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 1 of 10: not in sync: resp=0x00
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 2 of 10: not in sync: resp=0x00
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 3 of 10: not in sync: resp=0x00
```

At this point, you need to make sure that the ESP32 CAM is unplugged.

The ESP32-CAM and the Arduino board share the same RX (receive) and TX (transmit) pins. So, when you’re uploading code, you’ll need to first disconnect the ESP32-CAM to avoid any conflicts or potential issues.



### 3.3 Q3: How can I use the STT feature on my Android device?

The STT feature requires the Android mobile device to be connected to the Internet and to install the **Google service component**.

Now follow the steps below.

1. Modify the AP mode of Zeus\_Car.ino file to STA mode.

- Open the Zeus\_Car.ino file located in the zeus-car-main/Zeus\_Car directory.
- Then comment out the AP mode related code. Uncomment the STA mode related code and fill in the SSID and PASSWORD of your home Wi-Fi.

```
/** Configure Wifi mode, SSID, password*/
// #define WIFI_MODE WIFI_MODE_AP
// #define SSID "Zeus_Car"
// #define PASSWORD "12345678"


#define WIFI_MODE WIFI_MODE_STA
#define SSID "xxxxxxxxxx"
#define PASSWORD "xxxxxxxxxx"
```

- Save this code, select the correct board (Arduino Uno) and port, then click the **Upload** button to upload it to the R3 board.

2. Search google in Google Play, find the app shown below and install it.

17:42

Bluetooth xiaoming\_PC (25)

 google**Brave Private Web Browser**

Ad • Brave Software • Communication

*Block ads on every video*

4.7 ★ 91 MB 100M+

**Google**

Google LLC • Tools • Browsers

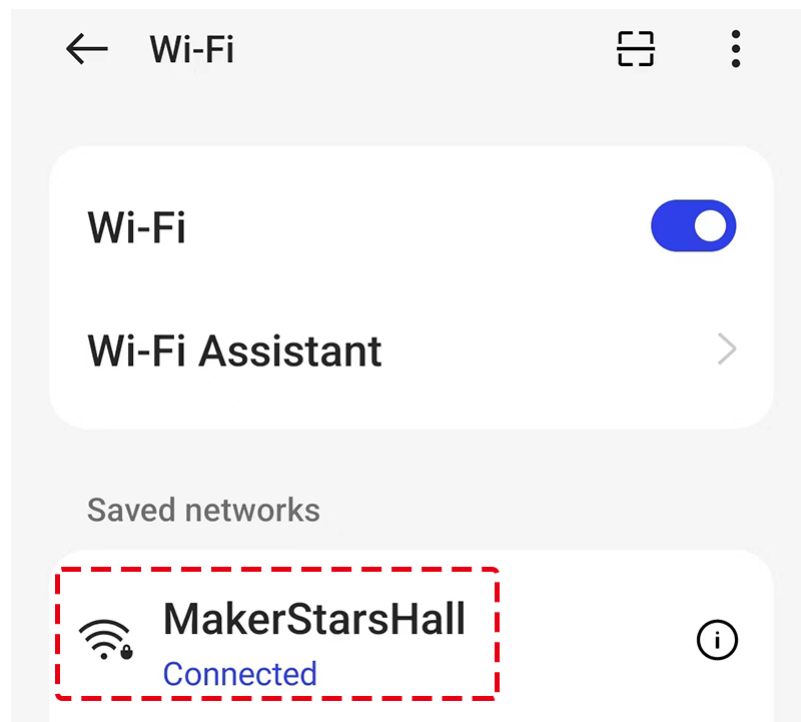
4.3 ★ 73 MB 10B+


**Google Chrome: Fast & Secure**

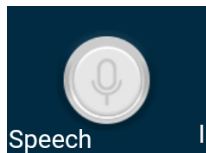
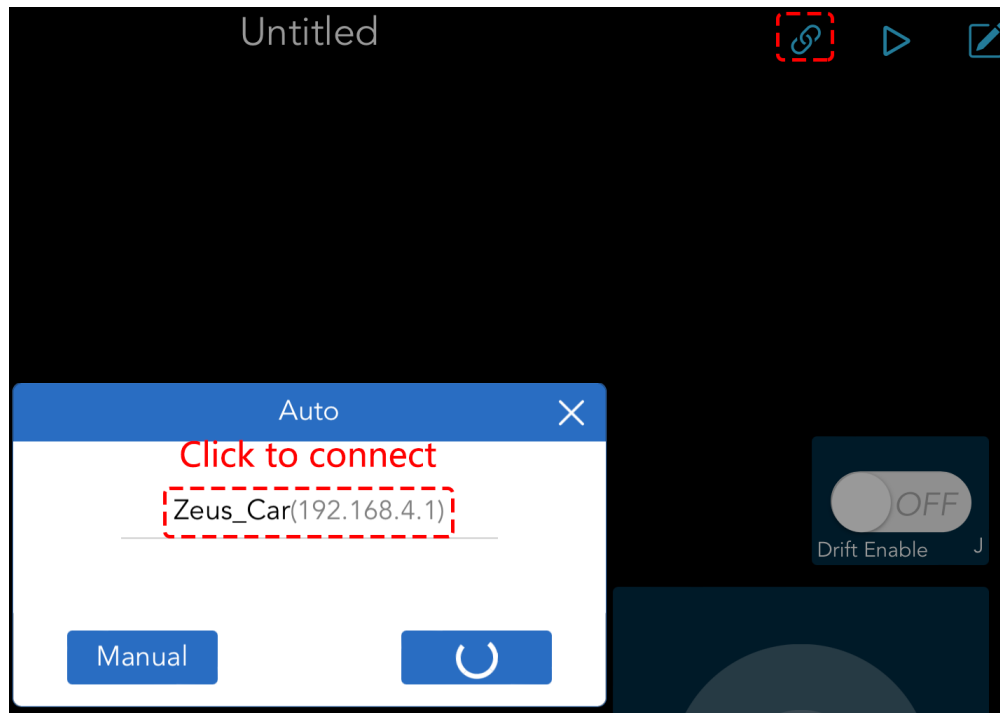
Google LLC • Communication • Tools • Browsers


✓ Installed

3. Connect your mobile device to the same Wi-Fi as you wrote in the code.



4. Open the controller previously created in SunFounder Controller and connect it to Zeus\_Car through the  button.



5. Tap and hold the **Speech** widget after clicking the  button. A prompt will appear indicating that it is listening. Say the following command to move the car.

- stop: All movements of the car can be stopped.
- pasue: The function is basically the same as Stop, but if the head of the car is not facing the direction originally set, it will slowly move to the set direction.
- forward
- backward
- left forward
- left backward
- right forward
- right backward
- move left
- move right

### Copyright Notice

All contents including but not limited to texts, images, and code in this manual are owned by the SunFounder Company. You should only use it for personal study, investigation, enjoyment, or other non-commercial or nonprofit purposes, under the related regulations and copyrights laws, without infringing the legal rights of the author and relevant right holders. For any individual or organization that uses these for commercial profit without permission, the Company reserves the right to take legal action.