



Four way tracking car tutorial

ZYC0043

V2.02.21.024

About ZHIYI.Ltd

ZHIYI.Ltd is a professional company engaged in UNO , NANO A company that develops and sells control boards and various modules or sensors for Arduino . We have also designed starter kits for hobbyists of different levels to learn Arduino. We are located in Shenzhen , China, all our products meet international quality standards and are highly appreciated in various different markets around the world.

Customer service

Establishing cooperation with companies from different countries and regions , and also helping them source high-quality electronic components in China . We look forward to establishing cooperative relations with more companies in the future. If you have any comments or suggestions , please contact us by email robot@zhiyi.ltd

Content

Lesson 1 Install Arduino IDE	7
Introducing Arduino	7
Arduino Board	7
Arduino software	8
Install Arduino (Mac OS X)	16
Install Arduino (Linux)	16
Lesson 2 Adding "Libraries" to the Arduino IDE	17
How to Install Additional Libraries in Arduino IDE 1	17
What are Libraries?	17
Method 1: Import the .zip library	18
Method 2:	20
Lesson 3 The first program code - Blink	23

About this lesson:	23
Main control board :	23
Make your own "Blink" sketches	24
Lesson 4 Auto move	33
Course Introduction	33
Component introduction: L298N	33
How does the L298N module work?	34
Wiring diagram	35
code analysis	37
Lesson 5 Follow the car	39
Course Introduction	39
Component introduction	39

How does ultrasound work?	40
servo motor	41
Wiring diagram	45
LM393 module:	46
The detection distance can be adjusted by the potentiometer knob :	47
How does the infrared detection module work?	47
code analysis	50
follow the trolley principle	51
Lesson 6 Obstacle Avoidance Car	54
Course Introduction	54
Obstacle Avoidance Principle	54
Lesson 7 Tracking Car	57
Four-way tracking module:	57

How does the four-channel tracking module work?	57
Debugging of the tracking module	58
tracking principle	60
Expansion board wiring diagram	62
code analysis	63

Lesson 1 Install Arduino IDE

Introducing Arduino

Arduino is an open source electronics platform based on easy-to-use hardware and software. Suitable for anyone working on interactive projects . Generally speaking, an Arduino project is composed of hardware circuits and software codes.

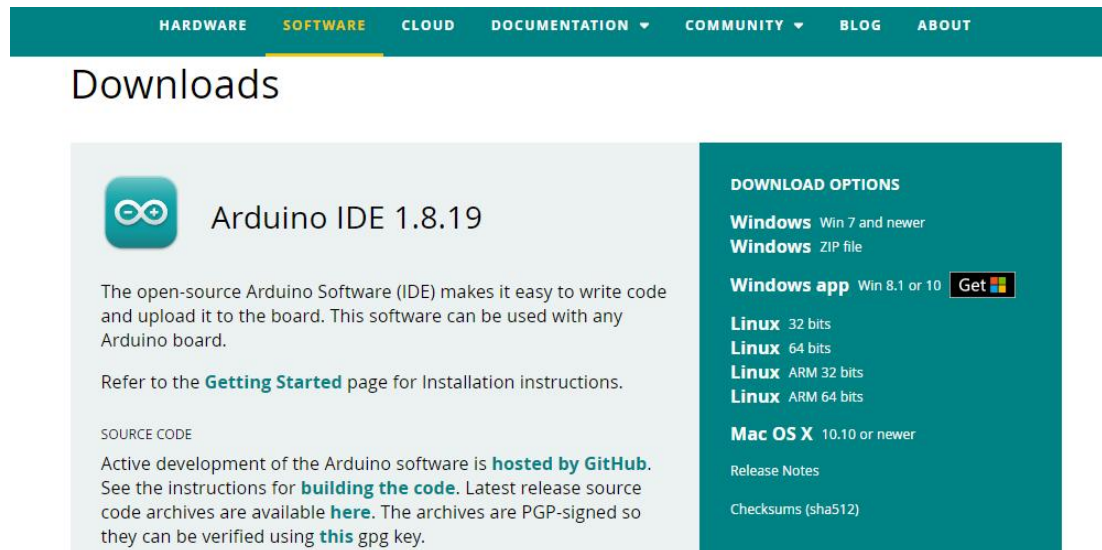
Arduino Board

Arduino Board is a circuit board that integrates a microcontroller, input and output interfaces, etc. The Arduino Board can sense the environment using sensors and receive user actions to control LEDs, motor rotation, and more. We only need to assemble the circuit and write the code for burning to make the product we want . Currently, there are many models of Arduino Boards, and the code is common between different types of boards (due to differences in hardware, some boards may not be fully compatible).

Arduino software

Arduino Integrated Development Environment (IDE) is the software side of the Arduino platform. For writing and uploading code to the Arduino Board. Follow the tutorial below to install the Arduino software (IDE) .


Step 1: Click to go to <https://www.arduino.cc/en/software> webpage and find the following webpage location:



The screenshot shows the Arduino IDE 1.8.19 download page. The top navigation bar includes links for HARDWARE, SOFTWARE (highlighted), CLOUD, DOCUMENTATION, COMMUNITY, BLOG, and ABOUT. The main heading is "Downloads". The page is divided into two columns. The left column features the Arduino logo, the version "Arduino IDE 1.8.19", a description of the IDE as open-source software, a reference to the "Getting Started" page for installation instructions, and a section for "SOURCE CODE" mentioning GitHub and PGP-signed archives. The right column, titled "DOWNLOAD OPTIONS", lists download links for Windows (Win 7 and newer, ZIP file), Windows app (Win 8.1 or 10, with a "Get" button), Linux (32 bits, 64 bits, ARM 32 bits, ARM 64 bits), and Mac OS X (10.10 or newer). It also includes links for "Release Notes" and "Checksums (sha512)".

HARDWARE **SOFTWARE** CLOUD DOCUMENTATION COMMUNITY BLOG ABOUT

Downloads

 **Arduino IDE 1.8.19**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10 [Get](#)

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

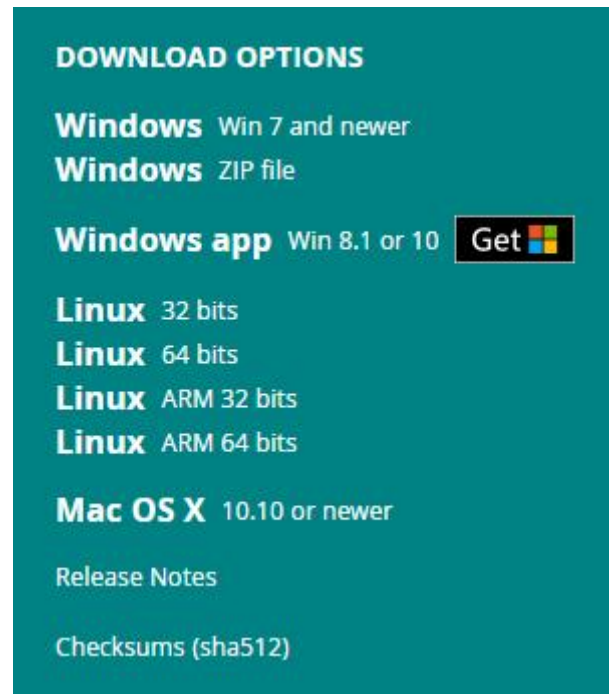
Mac OS X 10.10 or newer

[Release Notes](#)

[Checksums \(sha512\)](#)

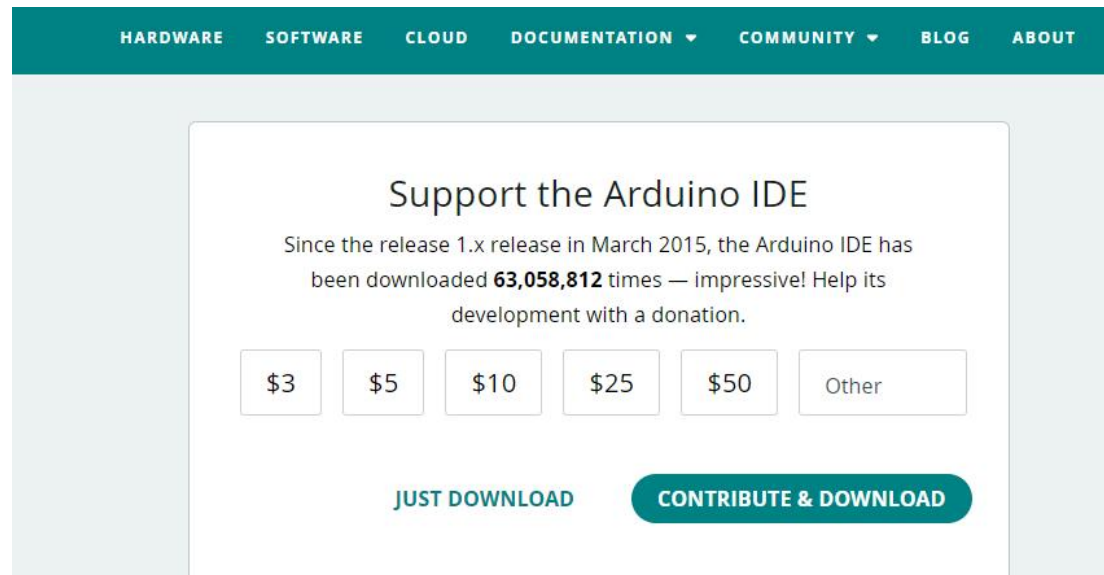
There may be a newer version on the site when you see this tutorial!

Step 2: Download the development software compatible with your computer system, here we take Windows as an example.



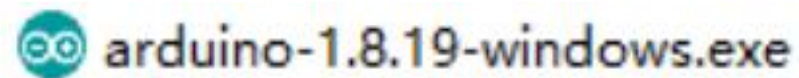
You can choose between an installer (.exe) and a Zip package. We recommend that you use the first "Windows Win7 and newer" to directly install everything you need to use the Arduino software (IDE), including drivers. With the Zip package, you need to install the driver manually. Of course, Zip files are also useful if you want to create portable installations.

Click on "Windows Win7 and newer"

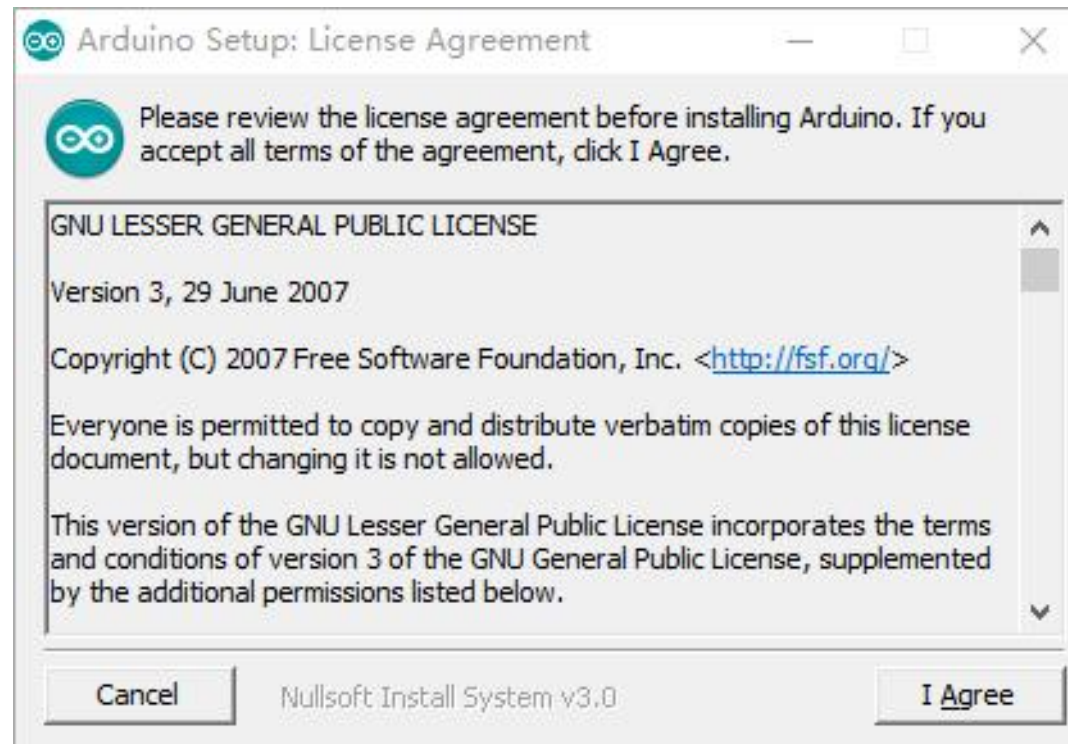


Click on "JUST DOWNLOAD".

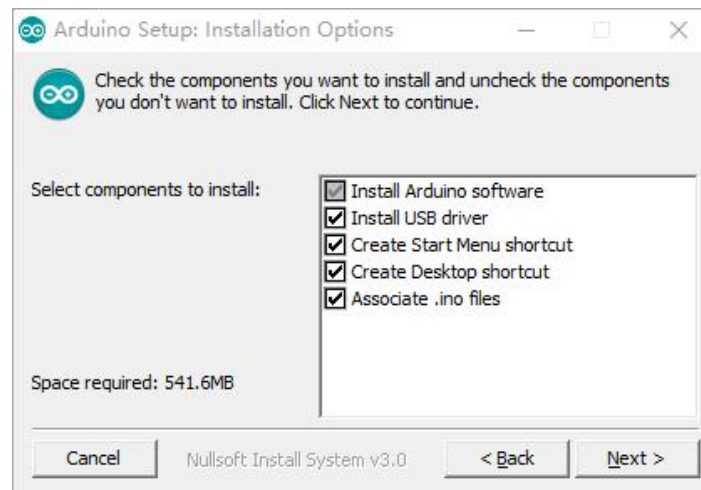
After the download is complete, the installation package file with the "exe" suffix will be obtained



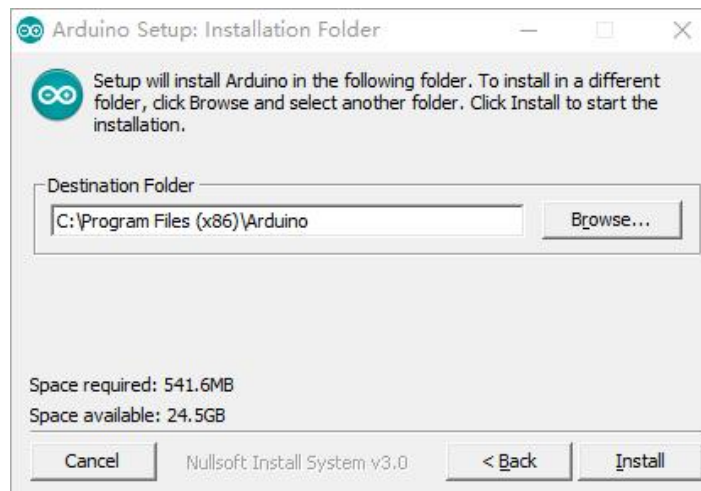
Double click to run the installer



Click "I Agree" to see the following interface



Click "Next"

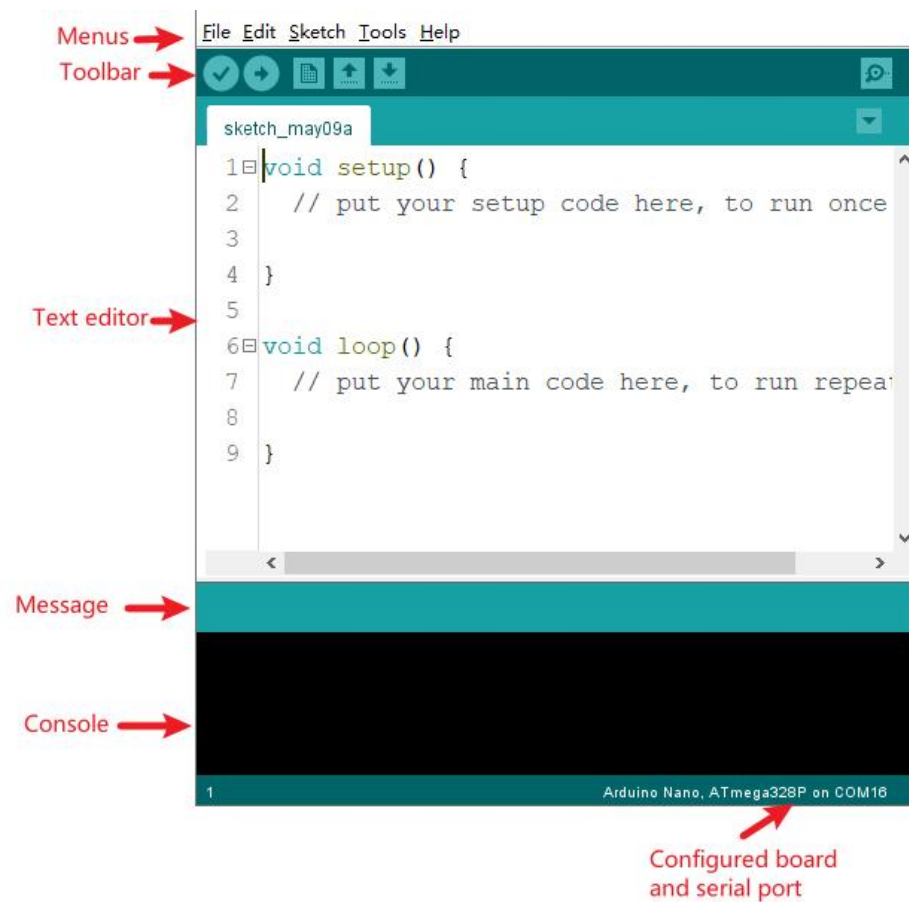


You can press "Browse..." to select the installation path or directly enter the directory you want. Then click "Install" to install. (For Windows users, the driver installation dialog may pop up during the installation process , when it pops up, please allow the installation)



After the installation is complete, an Arduino software shortcut will be generated on the desktop , double click to enter the Arduino software platform environment .

After the installation is complete, open the software to see the software platform interface as shown below:



Programs written using the Arduino software (IDE) are called "Sketch". These "Sketch" are written in a text editor and saved with the file extension ".ino".

The editor has functions for cutting , pasting, and searching and replacing text. The message area provides feedback and displays errors when saving and exporting. The console displays text output by the Arduino software (IDE), including full error messages and other information. The lower right corner of the window displays the configured boards and serial ports. Toolbar buttons allow you to verify and upload programs, create, open and save projects, and open the serial monitor. The positions of the corresponding functions in the toolbar buttons are as follows:



Verify : Compile code to check for errors



Upload : Compile code and upload to circuit board



New : Create a project file



Open : Select an item from an existing library and open it in a new window



Save : Save your project files



Serial Monitor : Open the serial monitor

(It is worth noting that the "ino" file must be saved in a folder with the same name as itself. If the program is not opened in the same name folder, it will be forced to automatically create a folder with the same name.)

Install Arduino (Mac OS X)

Download and unzip the zip file, double-click Arduino.app to enter the Arduino IDE; if there is no Java runtime library in your computer, you will be asked to install it, after the installation is complete, you can run Arduino IDE.

Install Arduino (Linux)

You will have to use the make install command. If you are using Ubuntu system, it is recommended to install Arduino ID from Ubuntu Software Center

Lesson 2 Adding "Libraries" to the Arduino IDE

How to Install Additional Libraries in Arduino IDE 1

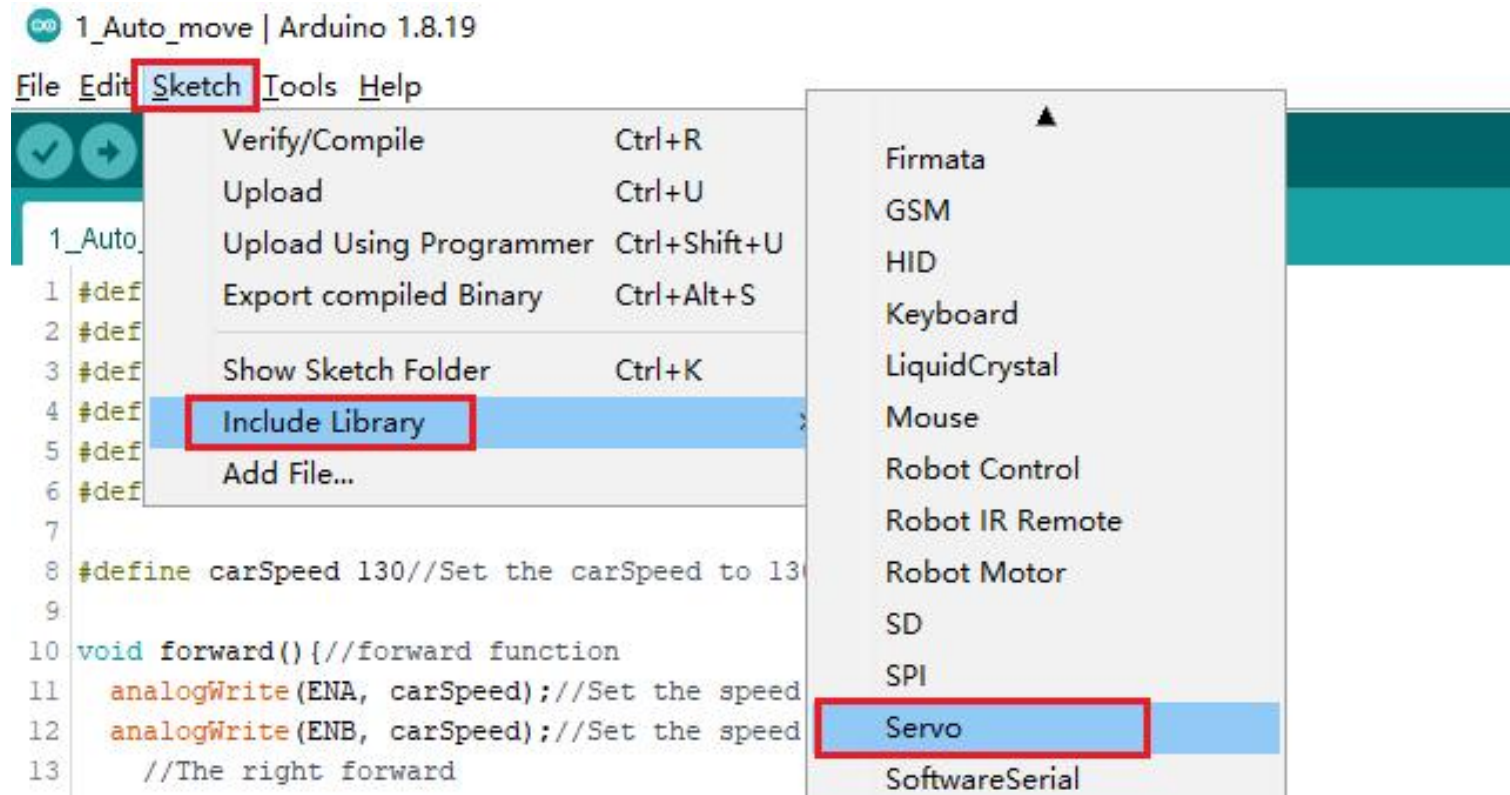
Once you are familiar with the Arduino software and use the built-in functions, you may wish to extend the functionality of the Arduino with other libraries.

What are Libraries?

A library is a set of code that allows you to easily connect to sensors, displays, modules, and more. For example, the LiquidCrystal library allows you to easily talk to character LCD displays.

There are thousands of libraries available for download directly through the Arduino IDE, all of which you can find in the Arduino Library Reference .

The library used in this tutorial is the servo motor: servo. The newer version of the Arduino IDE has integrated this library. You can open the Arduino IDE and find it in Sketch>Include Library. If the library is not available, follow the steps below to install the library That's it.

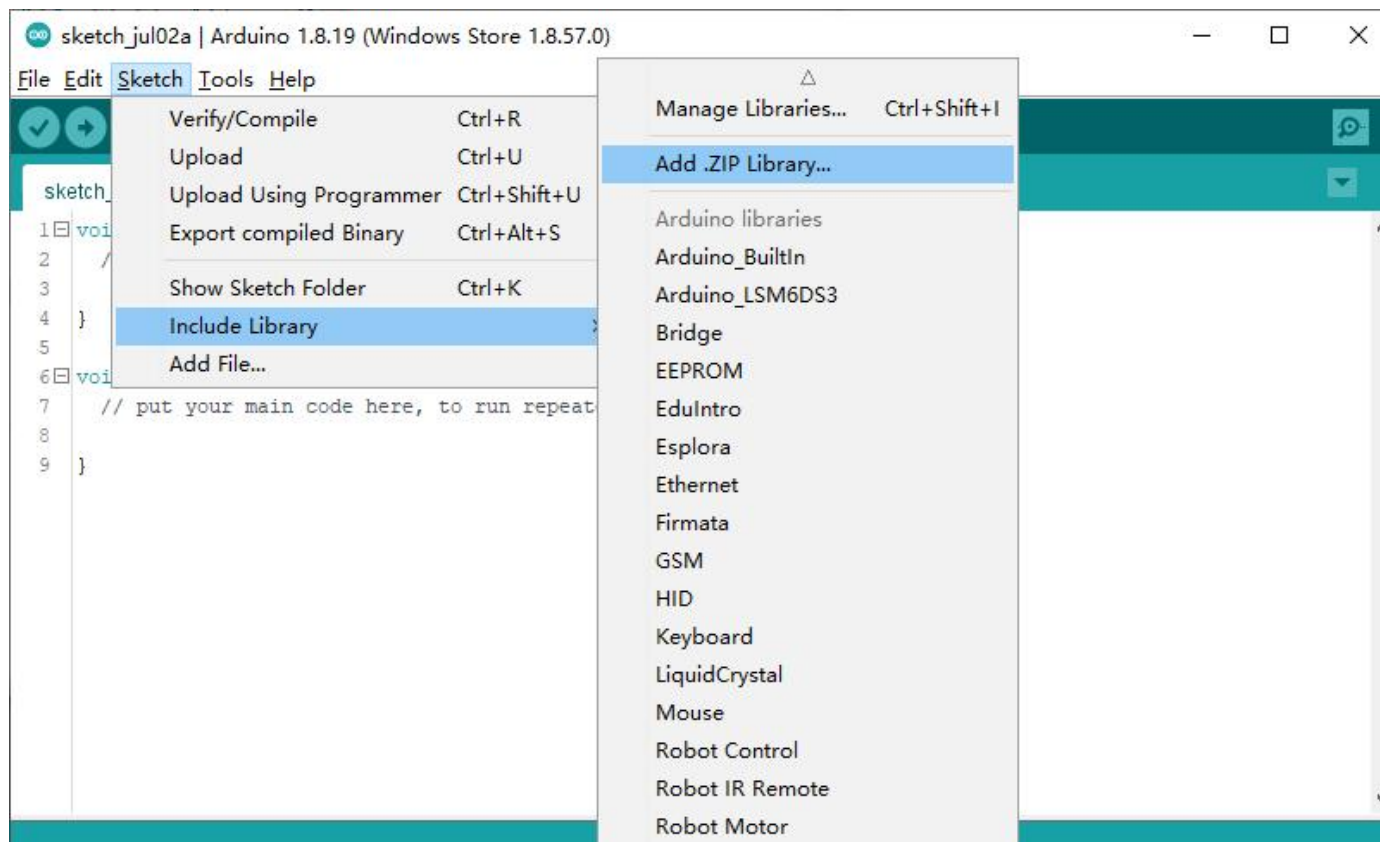


Method 1: Import the .zip library

Libraries are usually distributed as ZIP files or folders. The name of the folder is the name of the library. This folder will contain a .cpp file, a .h file, and usually a keywords.txt file, examples folders, and other files needed by the library. Starting

with version 1.0.5, you can install 3rd party libraries in the IDE. Do not unzip the downloaded library, keep it as is.

In the Arduino IDE, navigate to Sketch > Include Library > Add .ZIP Library , and at the top of the drop-down list, select the "Add .ZIP Library" option.



The system will prompt you to select the library you want to add , navigate to the saved library on your computer as shown below The path location of the servo .zip file (2_Libraries/servo.zip) and open it .

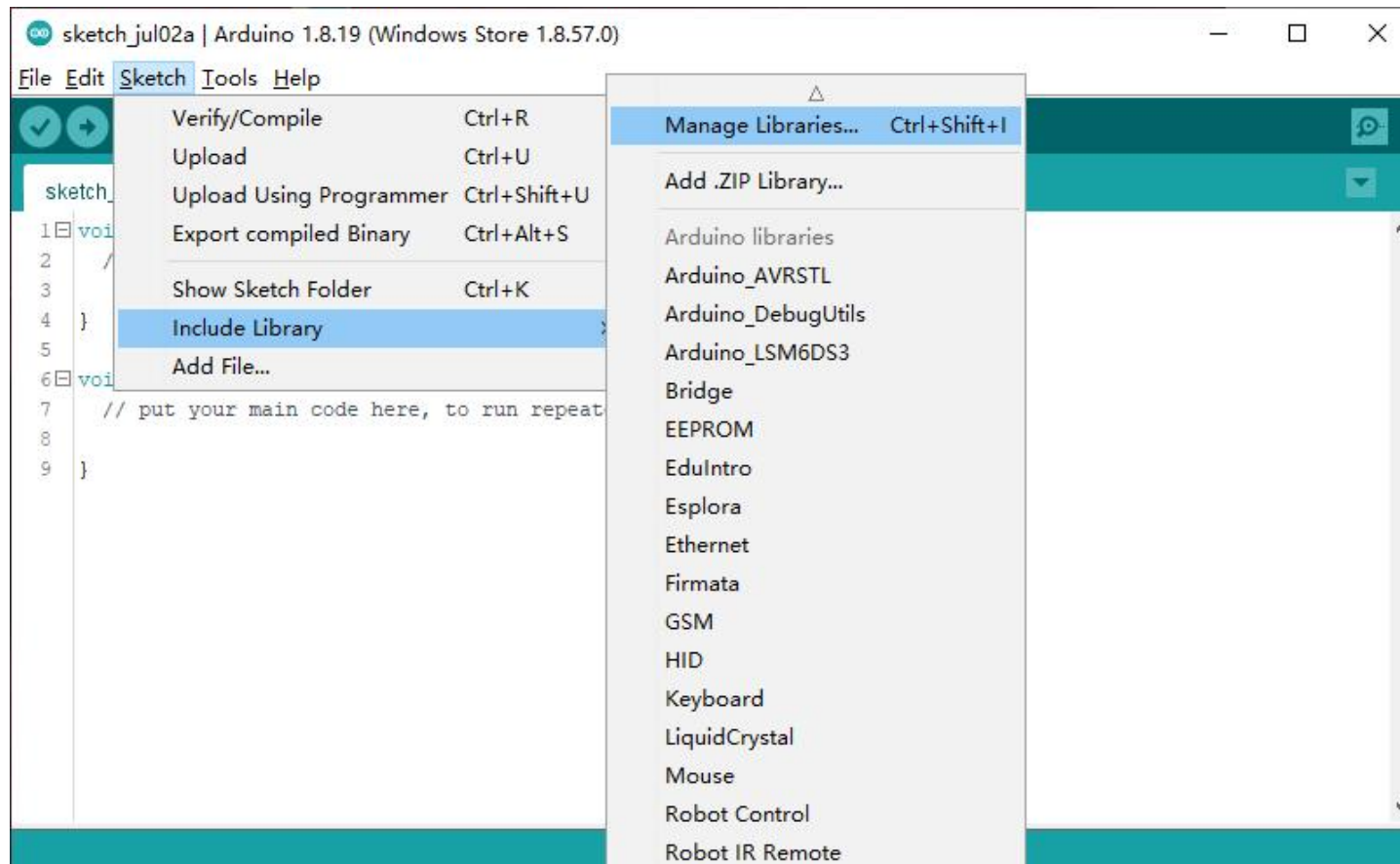


Return to the Sketch > Include Library menu. You should now see the library at the bottom of the drop-down menu , ready to use in your sketches .

Note: This library will be available for sketches, but with older IDE versions, the library's examples will not be exposed in File > Examples until the IDE is restarted.

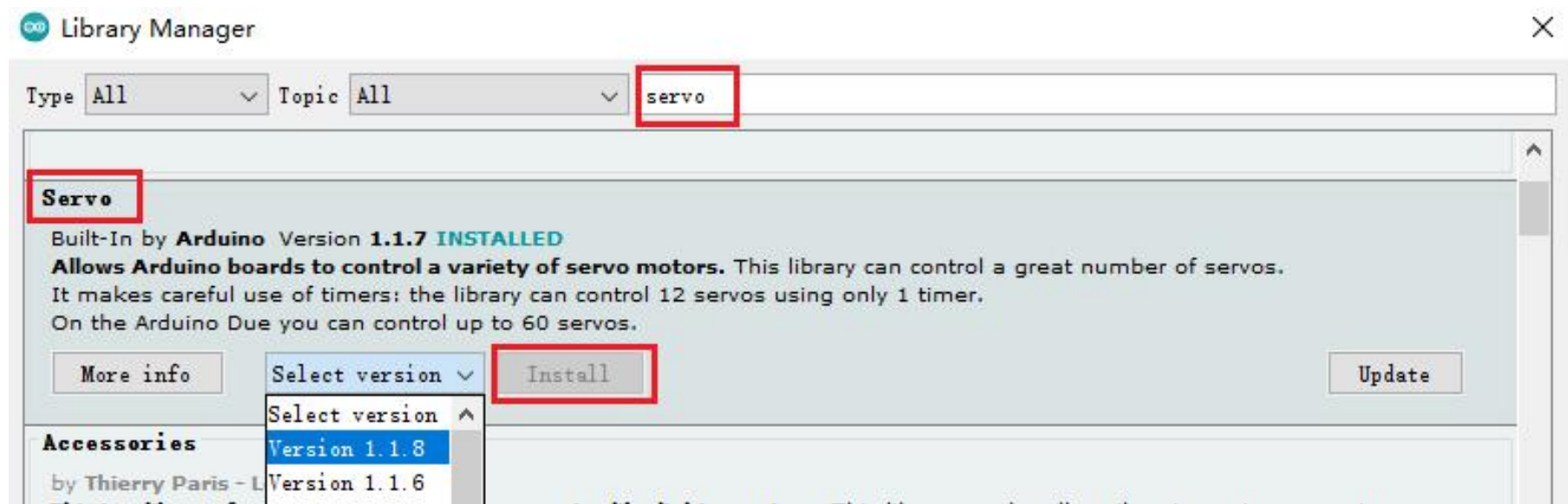
Method 2: (This method requires networking) In addition to adding the library that has been prepared, you can also use the library manager to search and download the desired library

To install new libraries into your Arduino IDE, you can use the library manager (available from IDE 1.6.2 and above). Open the IDE and click the Sketch menu, then click Include Library > Manage Libraries.



The library manager will open and you will see a list of libraries that are installed or ready to be installed. Here, we take the installation of the servo library as an example, and the same is true for installing other libraries. Scroll the list

to find it, then select the version of the library to install, sometimes only one version of the library is available , click Install to install it .



The download may take some time, depending on your connection speed , and you can close the library manager when finished.

Likewise, you can now find new libraries available in the Sketch > Include Library menu.

Lesson 3 The first program code - Blink

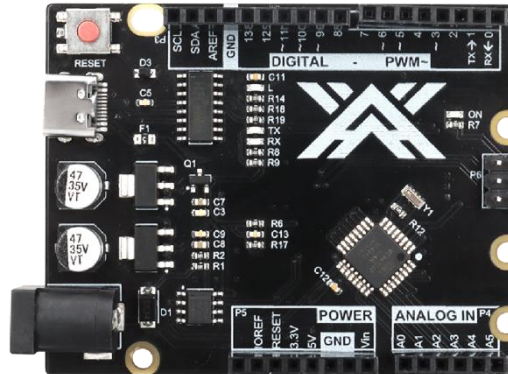
About this lesson:

In this lesson, you'll learn how to program your control board to blink the built-in LEDs, as well as learn the basic steps to download a program. Here we take the ZHIYI motherboard as an example to explain.

Main control board :

There are rows of connectors on both sides of the motherboard for connecting multiple electronic devices and plug-in "modules" that extend their functionality.

It also has an LED that you can control from the sketch , which is built into the motherboard .

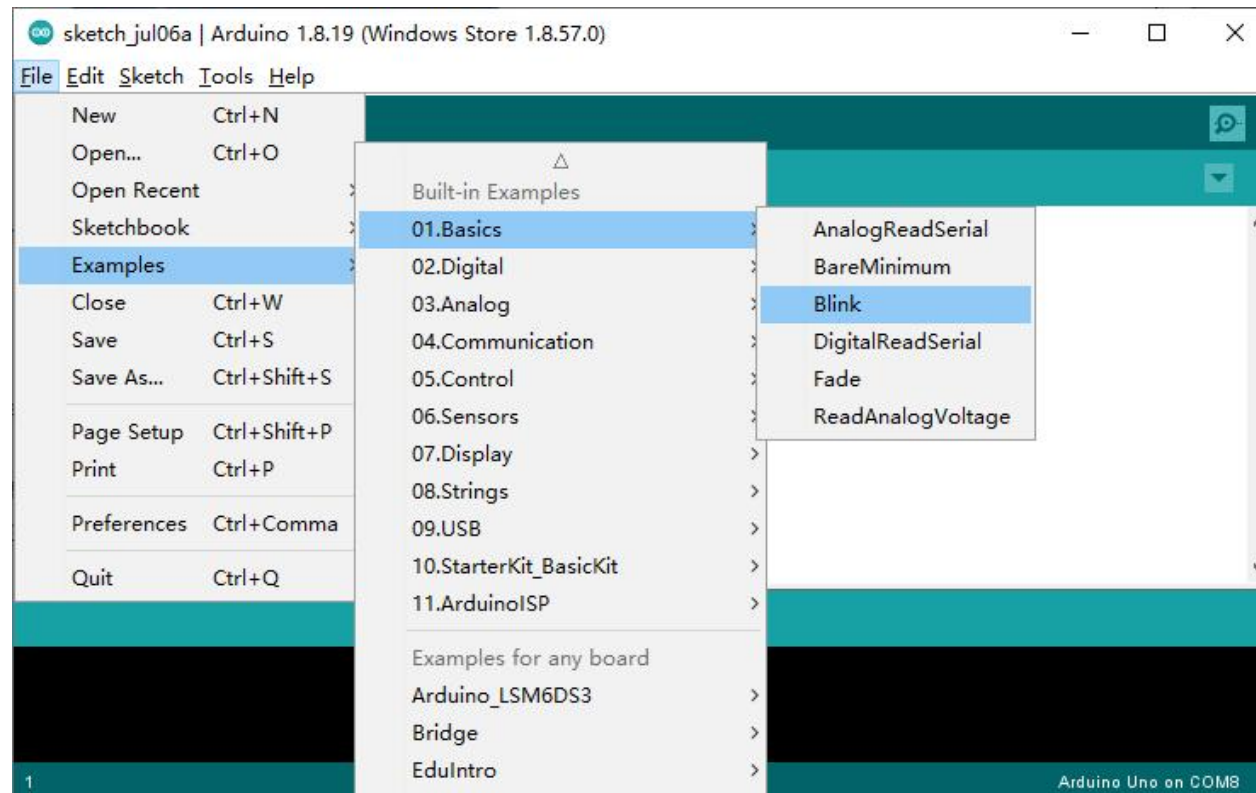


When you connect the board to the USB plug, you may find that its LEDs are already blinking because the "Blink" sketch is pre-installed on the board.

Make your own "Blink" sketches

the board with our own Blink sketch , and then change its blink rate. Connect the board to the computer, set up the Arduino IDE and make sure you can find the correct serial port , and upload the program for testing.

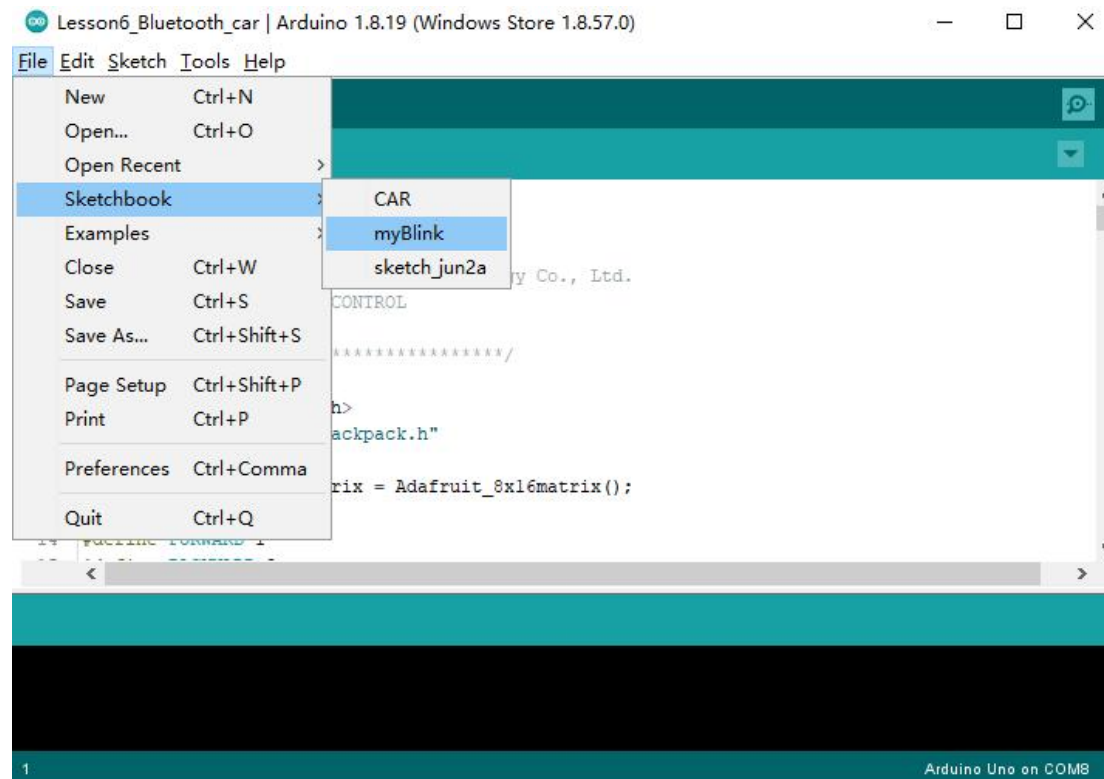
The Arduino IDE includes a number of example sketches that you can load and use , including a "Blink" example sketch for making an "L" LED. In the IDE menu system File > Examples > 01. The " Blink " sketch you will find in Basics .



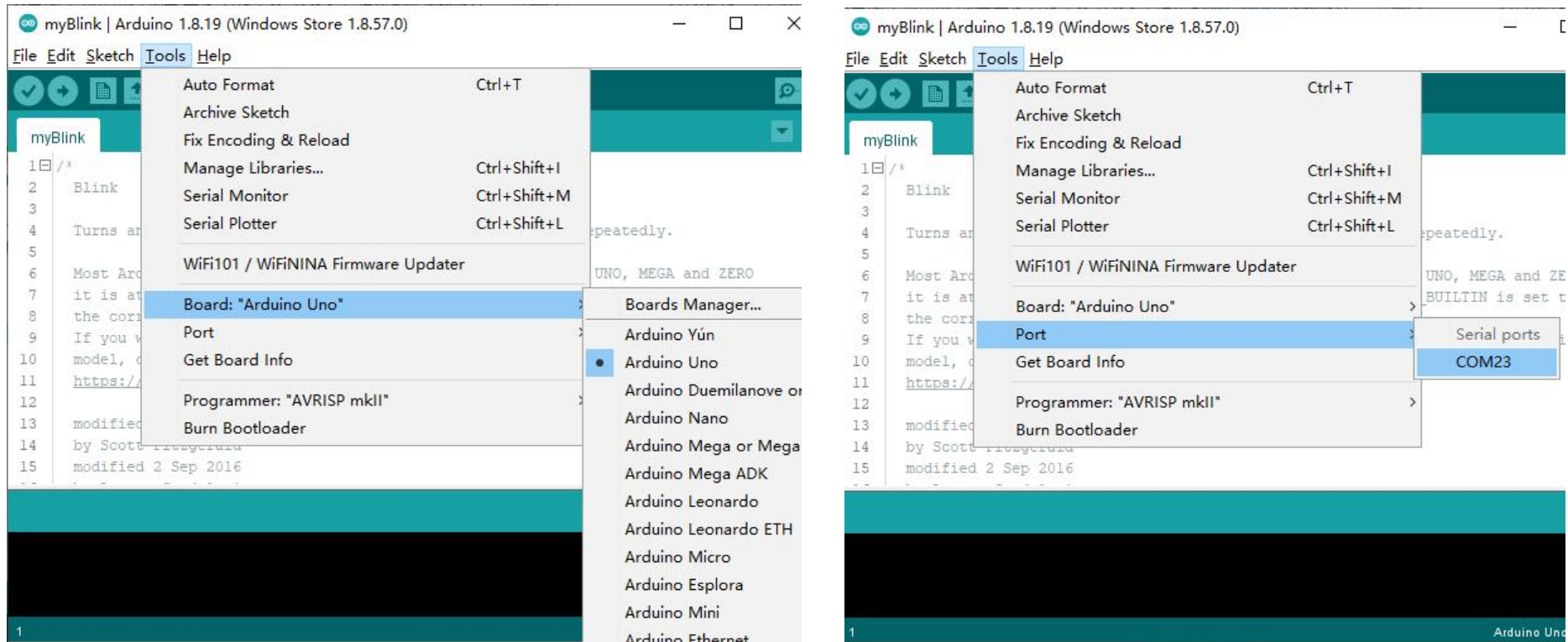
The example sketches included with the Arduino IDE are "read-only". That is, you can upload them to the UNOR3 board, but if you change them, you cannot save them as the same file. Since we're changing this sketch, the first thing you need to do is save a copy of yourself.

From the Arduino IDE's File menu, select "Save As.." and save the sketch as "MyBlink".

You've saved a copy of "Blink" in Sketchbook, which means that if you want to find it again, just open it with the " File > Sketchbook " menu option.

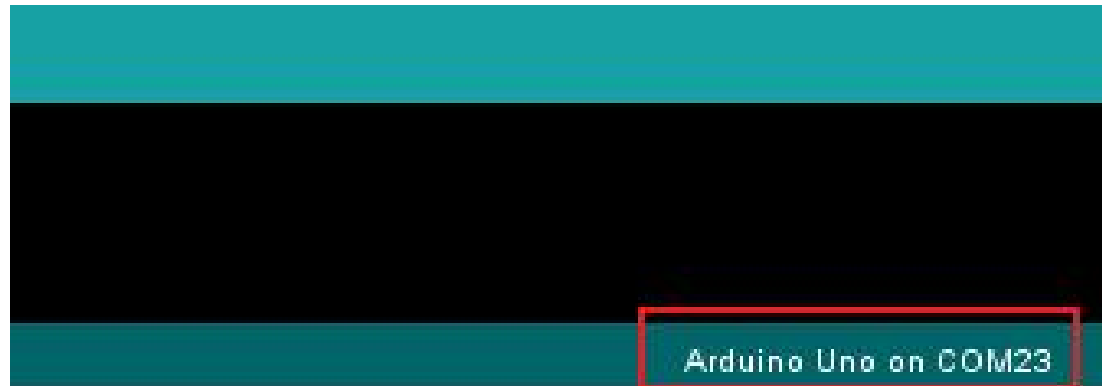


the Arduino board to the computer using a USB cable and check that the Board Type and Serial Port are set correctly.

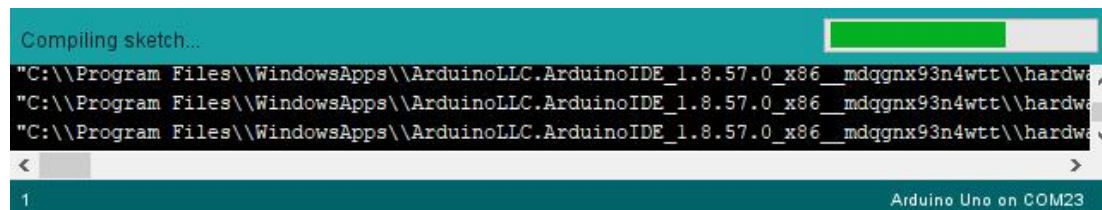


Note: The board type here is Uno and COM23 . If you are using 2560, then you will have to choose Mega 2560 as the Board Type , the other selection methods are the same. The serial port appears to be different for everyone, although COM2

3 is chosen here , it could be COM3 or COM4 on your computer. A correct COM port should be COMx compliant (arduino X XX) standard.



After clicking the "Upload" button, the program starts uploading, and at this time, the LED on the Arduino will start blinking as the sketch is transferred.



The transfer is complete, "Transfer complete" appears



During the "Compile Sketch.." process , you may receive the following error message:



This could mean that your board isn't connected at all, or the driver isn't installed (if needed) or the wrong serial port is selected wrong . If this happens to you, please check your IDE settings and motherboard connections . After the upload is complete , the board LEDs should reboot and start blinking.

Note that a large portion of this sketch is made up of annotations. These are not actual program instructions; instead, they just explain how to make the program work. They are there for your ease of reading . Everything between " /* " and " */ " at the top of the sketch is a block comment that explains what the sketch is for.

A single-line comment starts with "//", and everything up to the end of the line is considered a comment.

The first part of the code is:

```
// the setup function runs once when you press reset or power the board
void setup () {
// initialize digital pin LED_BUILTIN as an output.
  pinMode (LED_BUILTIN, OUTPUT);
}
```

Every sketch requires a "setup" function , aka "Void" setup()" function, which is executed when the reset button is pressed.

It is executed whenever the board is reset for any reason, such as first powering up or after uploading a sketch .

The next step is to name the pin and set the output, here " LED_BUILTIN " is set as the output port. On most Arduinos, including UNO , pin 13 is the pin corresponding to the LED, and for the convenience of programming, the program has set

the LED_BUILTIN variable to this pin, so there is no need to rename it to pin 13 for direct use.

The sketch must also have a "loop" function. Unlike the "setup" function, which only runs once, after a reset, the "loop" function will restart as soon as it finishes running the command.

```
// the loop function runs over and over again forever
void loop () {
  digitalWrite (LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay ( 1000 ); // wait for a second
  digitalWrite (LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay ( 1000 ); // wait for a second
}
```

Inside the loop function, the command first turns on the LED pin (high), then "delays for 1000 ms (1 second), then turns off the LED pin and pauses for one second.

You are now going to make your LED blink faster. The key, as you might have guessed, is to change the parameters in "delay ()".

```
32 void loop() {  
33     digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is  
34     delay(1000);                        // wait for a second  
35     digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by makin  
36     delay(1000);                        // wait for a second  
37 }
```

This delay is in milliseconds, so if you want the "LED" to blink twice as fast, change the value from "1000" to "500" . This will pause for half a second at each delay instead of a second. Upload the sketch again and you should see the "LED" start blinking faster .

So far, you have understood and mastered the basic Arduino programming knowledge and the basic steps of downloading programs, which lays a good foundation for learning complex projects later.

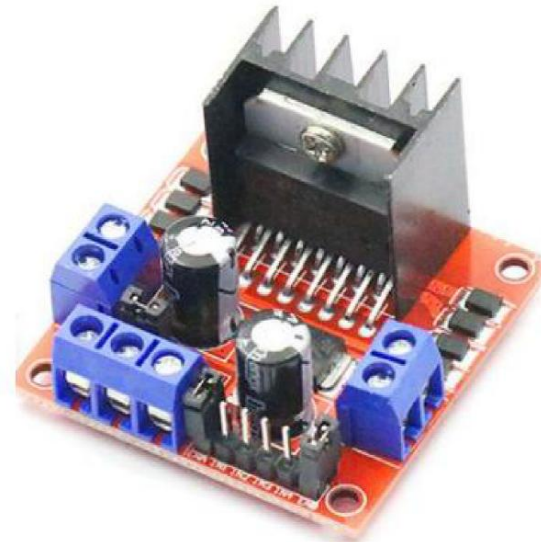
Lesson 4 Auto move

Course Introduction

In the last lesson, we learned the basic method of uploading programs. In this lesson, we will learn to understand the important drive components and master the control principles of car movement.

Component introduction: L298N

1. Input voltage of logic part: $6 \sim 7V$
2. Drive part input voltage v_s : $4.8 \sim 46v$
3. The working current of the logic part is: $\leq 36ma$
4. Driving part IO working current: $\leq 2A$
5. Maximum power dissipation: $25W$ ($t=75^{\circ}C$)
6. Control signal input level:
High level: $2.3V \leq V_{IN} \leq v_s$



Low level: $-0.3V \leq V_{IN} \leq 1.5V$

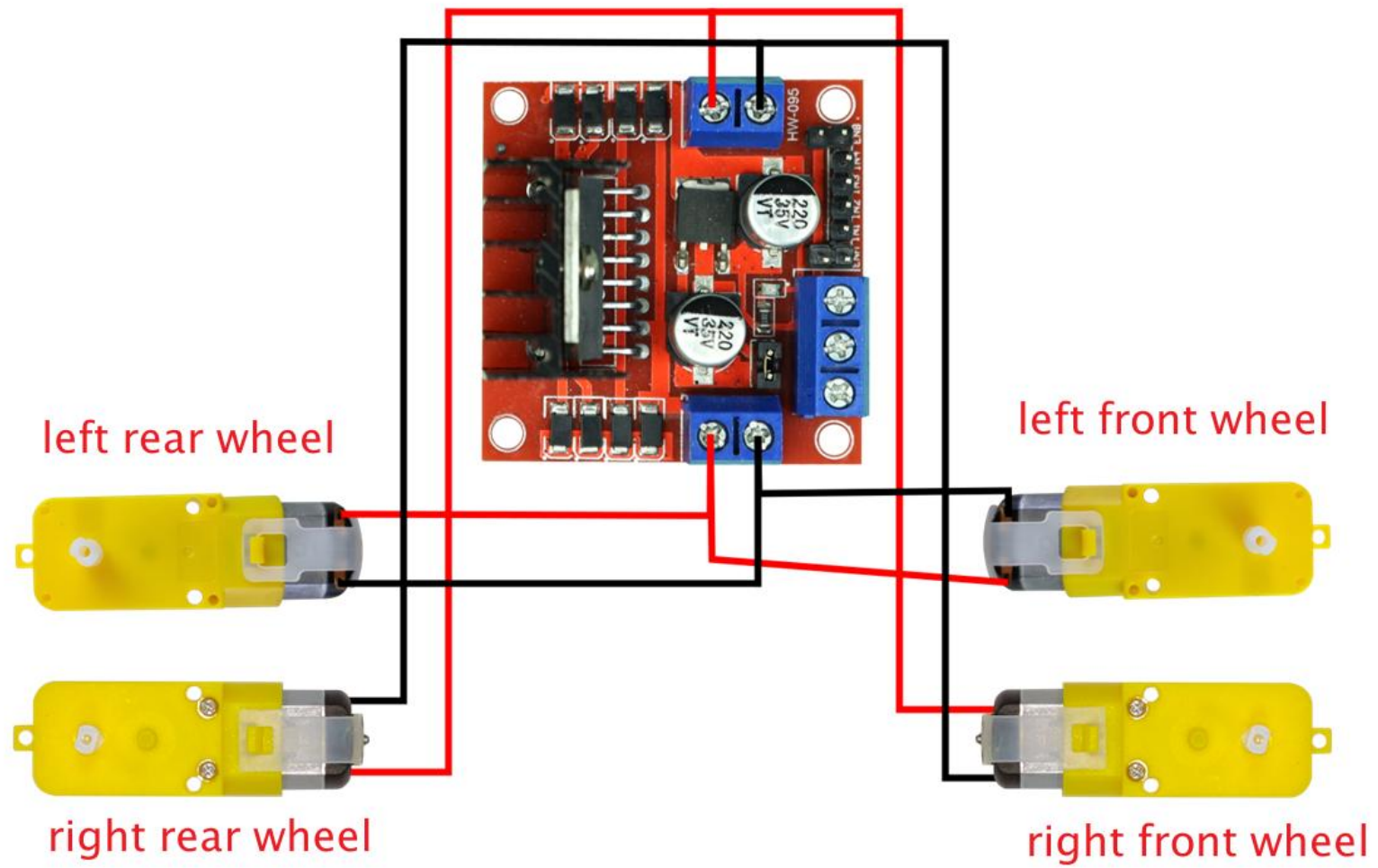
7. Working temperature: $-25^{\circ}\text{C} \sim +130^{\circ}\text{C}$
8. Drive mode: dual-channel high-power H-bridge drive

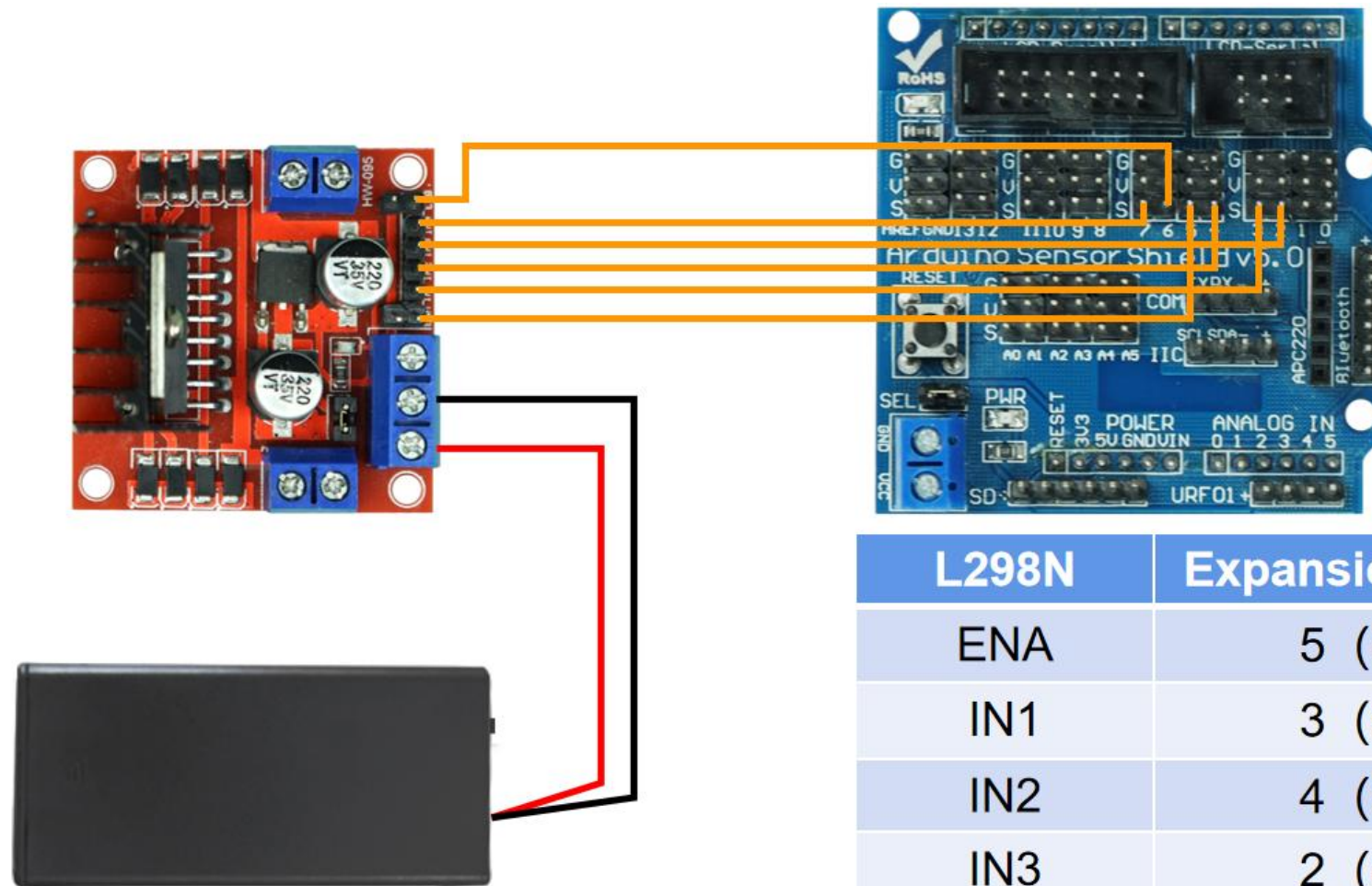
How does the L298N module work?

1. ENA, ENB control enable
2. out1,out2 and out3,out4 control the motor
3. IN1, IN2, IN3, IN4 are connected to the control level to control the positive and negative rotation .

State	ENA	ENB	IN1	IN2	IN3	IN4
Forward	carSpeed	carSpeed	1	0	1	0
Back	carSpeed	carSpeed	0	1	0	1
Left	carSpeed	carSpeed	0	1	1	0
Right	carSpeed	carSpeed	1	0	0	1
Stop	0	0	0	0	0	0

Wiring diagram





L298N	Expansion board
ENA	5 (S)
IN1	3 (S)
IN2	4 (S)
IN3	2 (S)
IN4	7 (S)
ENB	6 (S)

code analysis

Open the code file (path: 4_Arduino_Code\1_Auto_move\Auto_move.ino)

define motor and pwm pins

```
#define ENA 5
#define ENB 6
#define IN1 3
#define IN2 4
#define IN3 2
#define IN4 7
```

Define the motor speed, you can modify the value by yourself (100~220 is more suitable) to get different motion speeds

```
#define carSpeed 130 //Set the carSpeed to 130
```

For example, the forward() function will move forward at the speed you set

```
void forward () { //forward function
    analogWrite (ENA, carSpeed); //Set the speed of ENA
    analogWrite (ENB, carSpeed); //Set the speed of ENB
```

Of course, you can also enter the value directly

```
analogWrite (ENA, 200 ); //Set the speed of ENA
```

Set function to define baud rate 9600 and pin as output

```
void setup () {  
  Serial.begin ( 9600 ) ; _  
  pinMode (IN1, OUTPUT);  
  pinMode (IN2, OUTPUT);  
  pinMode (IN3, OUTPUT);  
  pinMode (IN4, OUTPUT);  
  pinMode (ENA, OUTPUT);  
  pinMode (ENB, OUTPUT);  
  stop ();  
}
```

Loop function, execute forward, backward, left turn and right turn for one second each, of course, you can also modify 1000 to 3000 to see how it works.

```
void loop () {  
  forward ();  
  delay ( 1000 );  
  back ();  
  delay ( 1000 );  
  left ();  
  delay ( 1000 );  
  right ();  
  delay ( 1000 );  
}
```

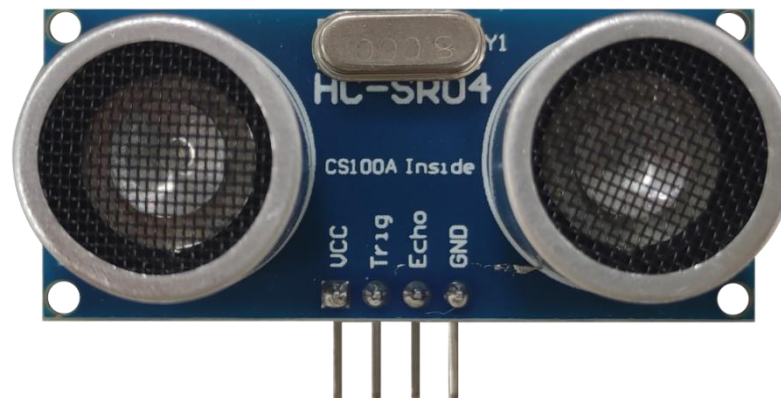
Lesson 5 Follow the car

Course Introduction

Ultrasonic ranging is a very useful and widely used measurement method. This lesson mainly focuses on understanding the working principle of the ultrasonic module, mastering the connection of the ultrasonic circuit diagram, and learning how to measure the distance of the ultrasonic module through programming.

Component introduction

HC-SR04 Ultrasonic Module:



Working voltage	DC-5V
Working current	15mA
Working frequency	40KHz
Distance range	2cm ~ 400cm _
Measuring angle	15 °
Input trigger signal	10 US TTL pulse
Output echo signal	Output TTL level signal, proportional to the range

How does ultrasound work?

1. Transmitter (trig pin) sends signal: high frequency sound;
2. When the signal hits an object, it is reflected;
3. Receiver (echo pin): Receives the signal reflected from it.

of the ultrasonic wave from the ultrasonic transmitter through the gas medium to the receiver, and multiply this time by the

speed of sound in the air to obtain the sound propagation. the distance.

The ultrasonic transmitter emits ultrasonic waves in a certain direction, and the MCU starts timing at the same time. The ultrasonic waves are transmitted in the air, and they return immediately when they encounter obstacles on the way. The ultrasonic receiver receives the reflected waves and stops the timing immediately.

From the time T recorded by the timer, the distance (s) from the launch point to the obstacle can be calculated .

$$\text{Formula: } S = VT/2$$

Four factors limit the maximum measurable distance of an ultrasound system: the amplitude of the ultrasound, the texture of the reflector, the angle between the reflected and incident sound waves, and the sensitivity of the receiving transducer. The ability of the receiving transducer to directly receive the acoustic pulse will determine the minimum measurable distance.

servo motor

SG90 module:



The control signal from the channel enters the signal modulation chip receiver to obtain the DC bias voltage. It has an internal reference circuit that generates a 20ms period and a reference signal 1.5ms width. The obtained DC bias voltage is compared with the voltage of the potentiometer to obtain the voltage difference output. Finally, the positive and negative output of the voltage difference is sent to the motor driver chip to determine the positive and negative rotation of the motor. When the motor speed is fixed, the drive potentiometer rotates through the cascade reduction gear, so that the voltage difference is 0, and the motor stops rotating.

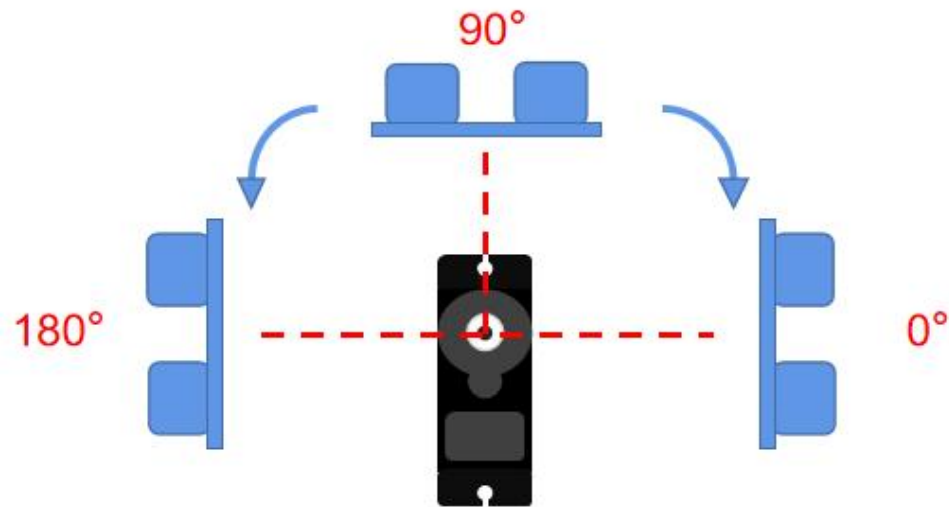
How does the SG90 module work?

Steering gear control: The control of the steering gear generally requires a time base pulse of about 20ms. The high-level part of this pulse controls the angle. The pulse range is generally 0.5ms~2.5ms. Take the servo motor with a servo of 180

degrees as an example.

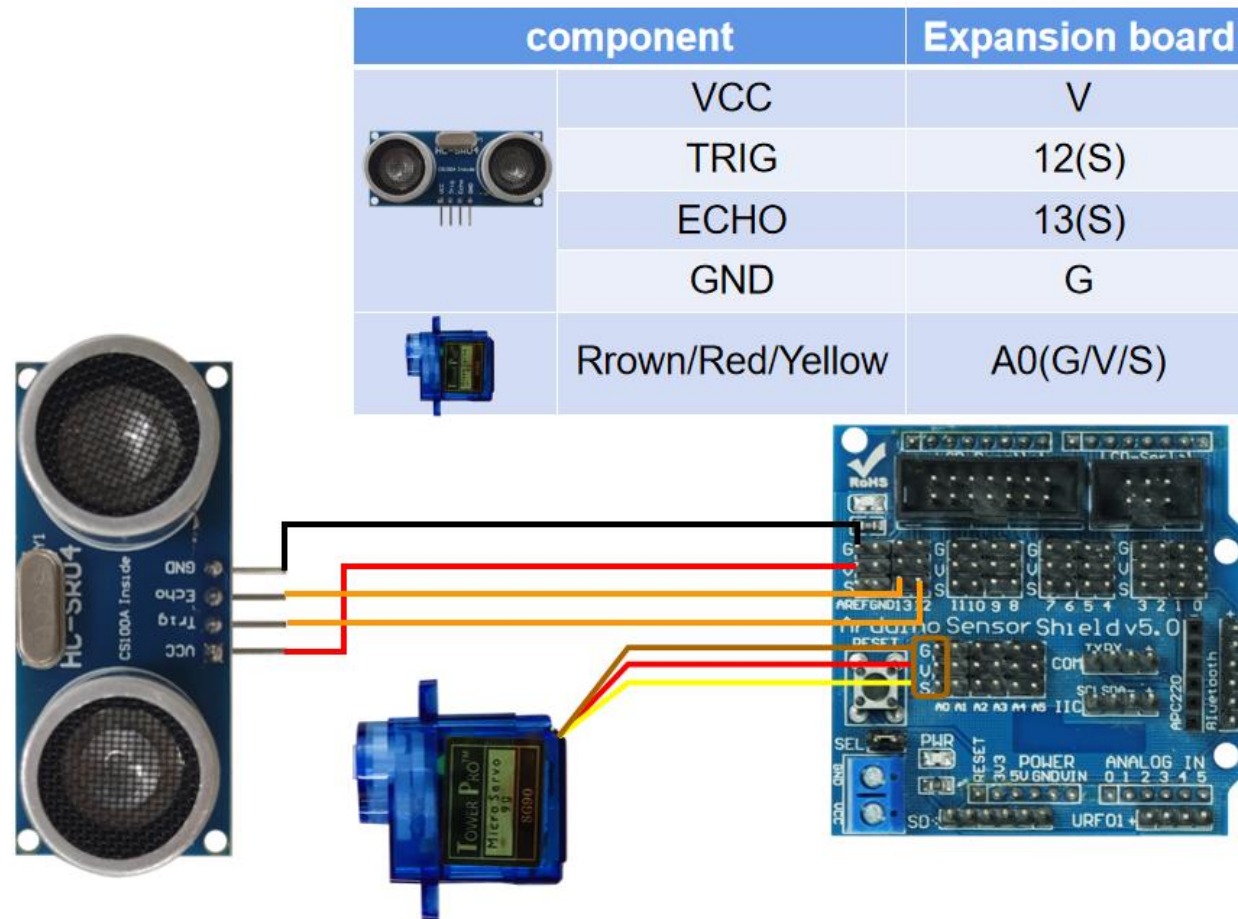
0.5ms	0°
1.0ms	45°
1.5ms	90°
2.0ms	135°

SG90 module installation method:

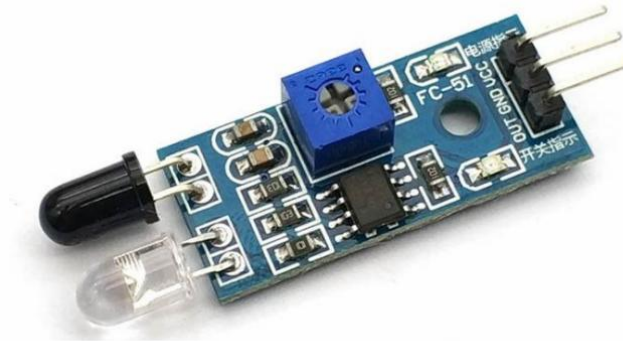


As shown in the picture above (left), the front of the car is facing up. When installing the ultrasonic wave, first make sure that the initial position is on the right. The operation method is to use the white mechanical handle (as shown in the right picture above) to try to twist the central axis of the servo motor, and observe 0° and 180° Location. Slowly turn the central shaft of the servo motor clockwise to the limit position. To ensure that the rotation range of the ultrasonic wave ($0\sim 180^\circ$) is on the left, front and right, install the ultrasonic wave in the direction of 0° , and then turn it counterclockwise by 90° to be the front.

Wiring diagram



LM393 module:



induction module light can sense the environment. It has a pair of infrared transmitting and receiving tubes. When the detection direction encounters an obstacle, the infrared receiving tube receives the reflected infrared frequency signal . After processing by the comparison circuit, the indicator light is on, and the signal output interface is at the same time. Output digital signal (low level signal) . The effective distance range of induction is 2~30cm, the working voltage is 3.3V~5V , and it has the characteristics of small interference, convenient assembly and convenient use .

The detection distance can be adjusted by the potentiometer knob :

We hope that when no object is detected, only the power light is on, and the signal light is not on (as shown on the left below), and when an object is detected, the signal light is on (as shown on the right below). If no object is detected, both lights are on. This is because the detection distance is too far. It needs to be adjusted counterclockwise to reduce the sensitivity, but it cannot be adjusted too small.



How does the infrared detection module work?

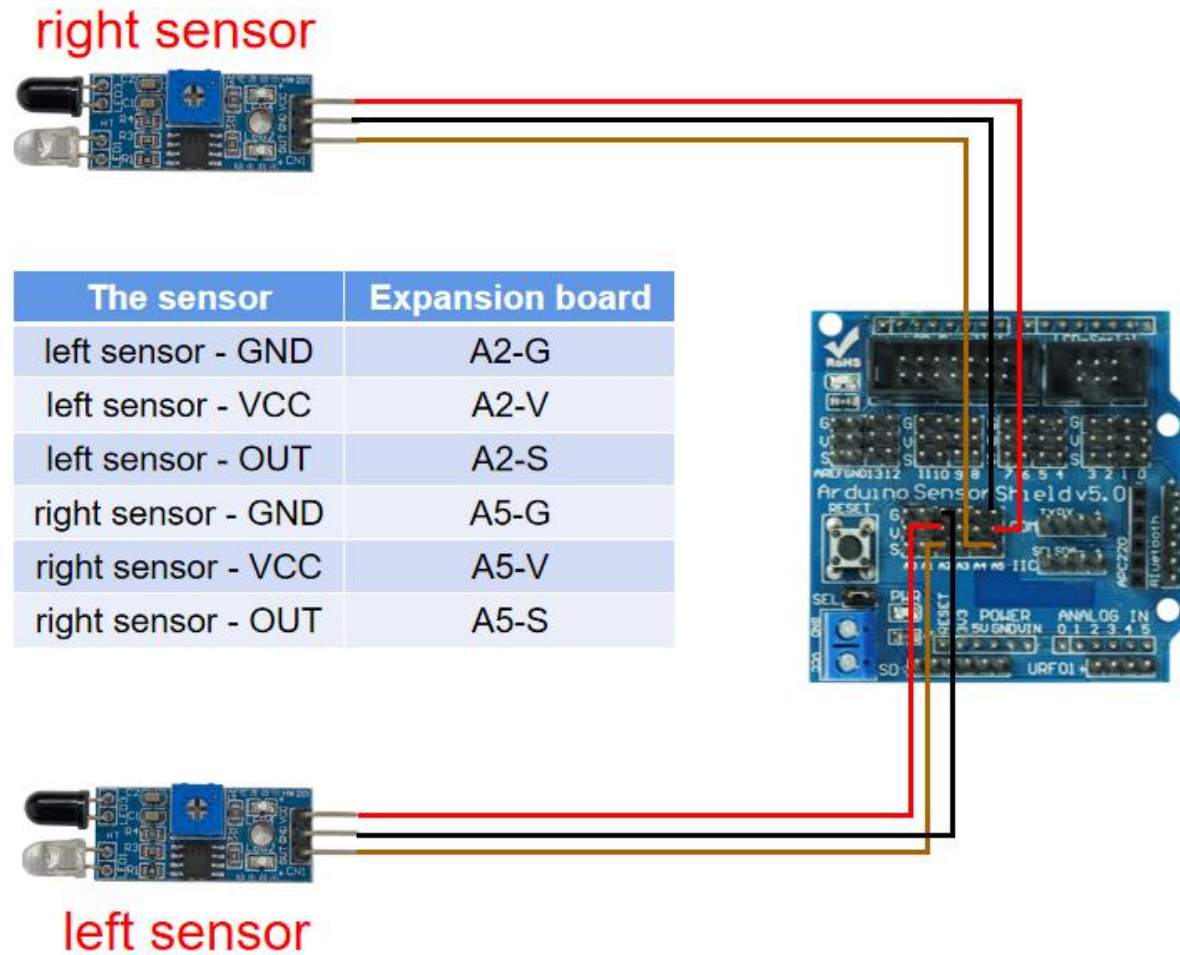
1. When the module detects the signal of the obstacle ahead, the indicator light on the green circuit board lights up, and the OUT port continuously outputs a low level signal. The detection distance module is 2~30cm, and the detection angle is 35°.

2. The sensor actively detects infrared reflection, so the shape and shape of the reflectivity target is the key to the detection range. Among them, the detection distance of black is small, and the detection distance of white is large.
3. The output port OUT of the sensor module can be directly connected to the IO port of the microcontroller;
4. The 3-5V DC power supply can be used to supply power to the module . When the power is turned on, the red power indicator lights up.

Connection method:

VCC---VCC; GND --- GND ; OUT---S (signal)

Wiring diagram



code analysis

Open the code file (path: 4_Arduino_Code\2_follow\2_follow.ino)

Define ultrasonic transmit and receive pins, define ultrasonic detection distance, left and right sensors and other variables

```
#include <Servo.h>
#define Trig 12 //Pin Tring connects to D12
#define Echo 13 //Pin Echo connects to D13
float cm; //Distance variable
int Sensor1 = A2; //pin A2
int Sensor2 = A5; //pin A5
int SensorLeft;
int SensorRight;
```

Initial setup

```
void setup()
{
    myservo.attach(A0); // attaches the servo on pin A0 to the servo object
    //Set the pin mode
    pinMode(ENA, OUTPUT);
    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    pinMode(ENB, OUTPUT);
}
```

```
pinMode(IN3, OUTPUT);
pinMode(IN4, OUTPUT);
```

```
pinMode(Sensor1, INPUT);
pinMode(Sensor2, INPUT);
pinMode(Trig, OUTPUT);
pinMode(Echo, INPUT);
Serial.begin(9600);
myservo.write ( 100 ) ;
}
```

follow the trolley principle

If the sensor on the right detects a signal, but the sensor on the left does not detect a signal, and the distance of the obstacle detected by the ultrasonic wave is between 5cm and 10cm, turn right . Similarly, if the left sensor detects the signal, it will turn to the left. Otherwise, it will be judged that the ultrasonic detection distance in front is between 10 and 20 cm, and then it will move forward. If it is greater than 20 cm or less than 3 cm, it will stop. Finally, the sensors on both sides will detect the signal but the ultrasonic When the distance is less than 5cm, the car will back up.

```
if (SensorLeft == HIGH && SensorRight == LOW && ( cm > 5 && cm < 10 ))
{
    right();
```

```

}
else if(SensorLeft == LOW && SensorRight == HIGH&& ( cm >5 && cm <10))
{
    left();
}
else if(SensorLeft == HIGH && SensorRight == HIGH&&( cm >11 && cm <20))
{
    forward();
}
else if(SensorLeft == HIGH && SensorRight == HIGH && ( cm >20 || cm <3))
{
    stop();
}
else if((SensorLeft == LOW && SensorRight == LOW)|| ( cm <5))
{
    back();
}
if (SensorLeft == LOW)
{
    myservo.write ( 160 ) ; _
}
else if (SensorRight == LOW)
{
    myservo.write ( 40 ) ;
}
else

```

```
{  
    myservo.write ( 100 ) ;  
}
```

Ultrasonic detection distance function

```
float GetDistance()  
{  
    float distance;  
    // Send a low short pulse to Trig to trigger the ranging  
    digitalWrite(Trig, LOW); //Send a low level to Trig  
    delayMicroseconds(2);  
    digitalWrite(Trig, HIGH);  
    delayMicroseconds(10);  
    digitalWrite(Trig, LOW);  
    distance = pulseIn (Echo, HIGH) / 58.00 ;  
  
    Serial . print ( "Distance = " );  
    Serial . println (distance); //The serial output distance is converted into cm  
  
    return distance;  
}
```

Lesson 6 Obstacle Avoidance Car

Course Introduction

In the previous course, we have already understood and learned the relevant knowledge of ultrasonic, servo motor and LM393 module. In this course, we will complete an automatic obstacle avoidance car.

Obstacle Avoidance Principle

The ultrasonic detection distance of the car is less than or equal to 18cm in front (that is, when an obstacle is encountered), the car stops the servo motor to drive the ultrasonic wave to turn right to obtain the right obstacle distance, then turn left and obtain the left obstacle distance and save them to the variables "rightDistance" and "leftDistance" respectively. middle. Then compare the distances, and which side has a greater distance, that is, which side to turn to without being hindered, so as to avoid obstacles.

Open the code file (path: 4_Arduino_Code\3_Obstacle_avoidance\3_Obstacle_avoidance.ino)

```
if (middleDistance <= 18 )  
{  
    stop ();
```

```
delay ( 500 );
myservo.write(10);
delay(500);
rightDistance = GetDistance();//getDistance();

delay(500);
myservo.write(100);
delay(500);
myservo.write(180);
delay(500);
leftDistance = GetDistance();//getDistance();

delay(500);
myservo.write(100);
delay(500);

if(rightDistance > leftDistance){
    back();
    delay(300);
    right();
    delay(300);
}
else if(rightDistance < leftDistance) {
    back();
    delay(300);
    left ();
}
```

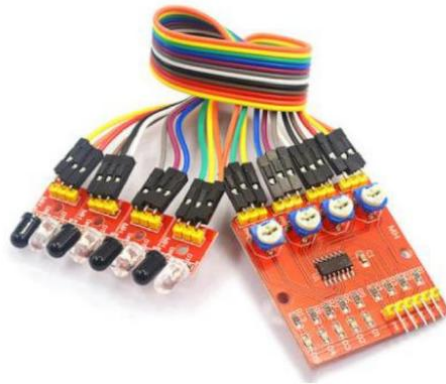
```
        delay ( 300 );  
    }  
    else {  
        forward ();  
    }  
}  
else {  
    forward ();  
}
```

In the process of ultrasonic obstacle avoidance, if the sensors on the left and right sides detect an obstacle (that is, a low level), it will also avoid it.

```
if (!SensorLeft){  
    right ();  
    delay ( 300 );  
}  
else if (!SensorRight){  
    left ();  
    delay ( 300 );  
}  
else {  
    forward ();  
}
```


Lesson 7 Tracking Car

Four-way tracking module:



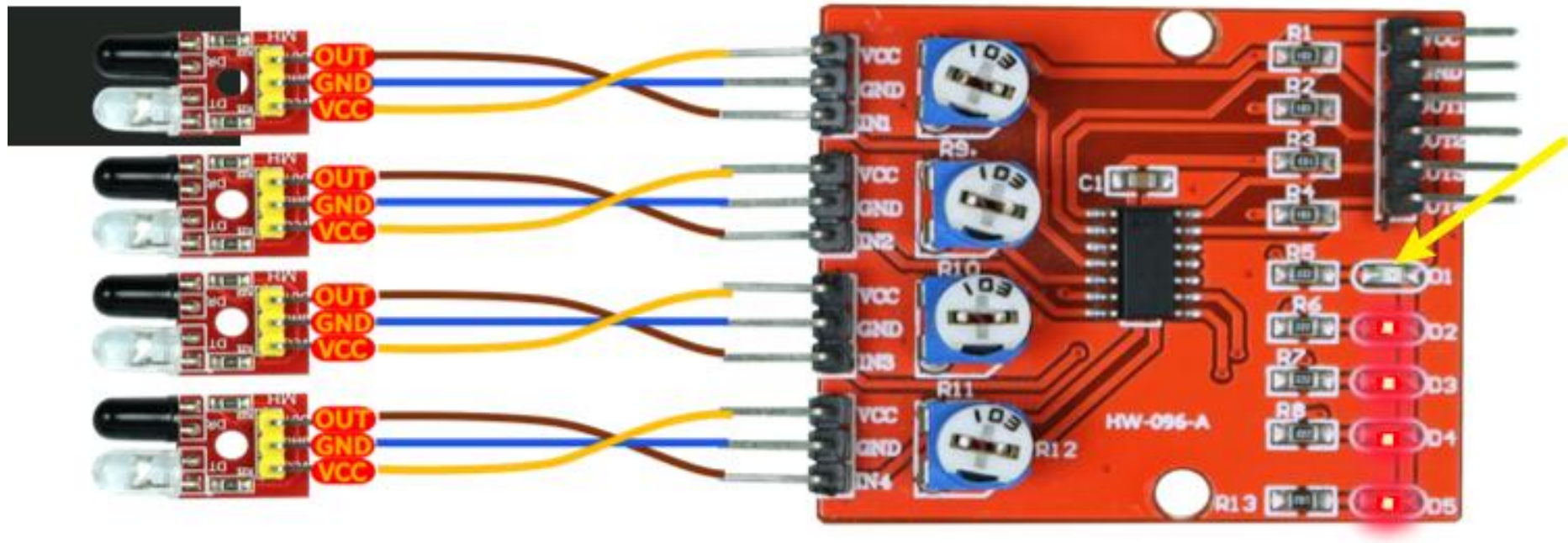
How does the four-channel tracking module work?

The sensor belongs to infrared reflection detection, that is to say, the color of the surface of different infrared light objects has different reflection intensity characteristics, and the infrared light is continuously emitted to the ground during the driving process of the car . When the infrared light encounters the white floor, it is diffusely reflected and reflected light It is received by the receiving tube installed in the car ; if it encounters a black line, the infrared light is absorbed, and the

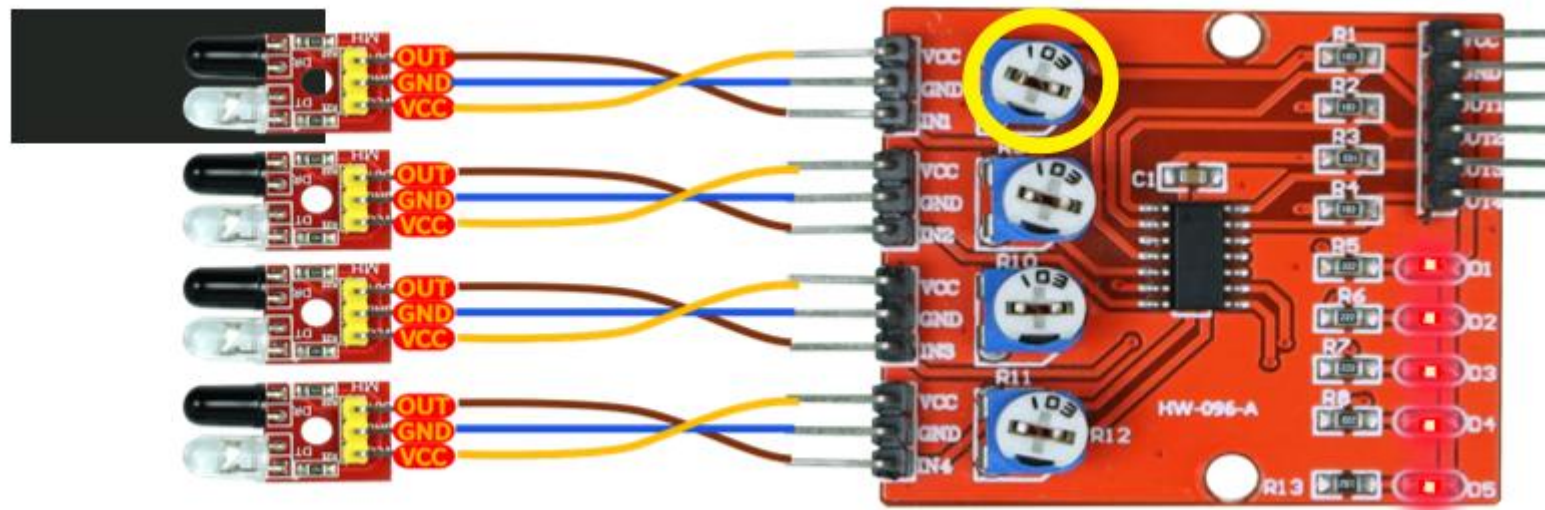
receiving tube on the car does not receive infrared light . Therefore , the position of the black line and the driving route of the car can be determined based on whether the reflected infrared light is received or not.

Debugging of the tracking module

As shown in the figure, when the power is turned on, the power light D5 lights up, and the four infrared transmitting and receiving tubes connected from top to bottom correspond to the four lights D1/D2/D3/D4 respectively. In order to achieve the tracking effect, we also need to The sensitivity of the infrared transmitting and receiving tube in the tracking module is adjusted to the ideal state: when the sensor detects a black line (non-white object), the corresponding light is off, and the other lights are on.



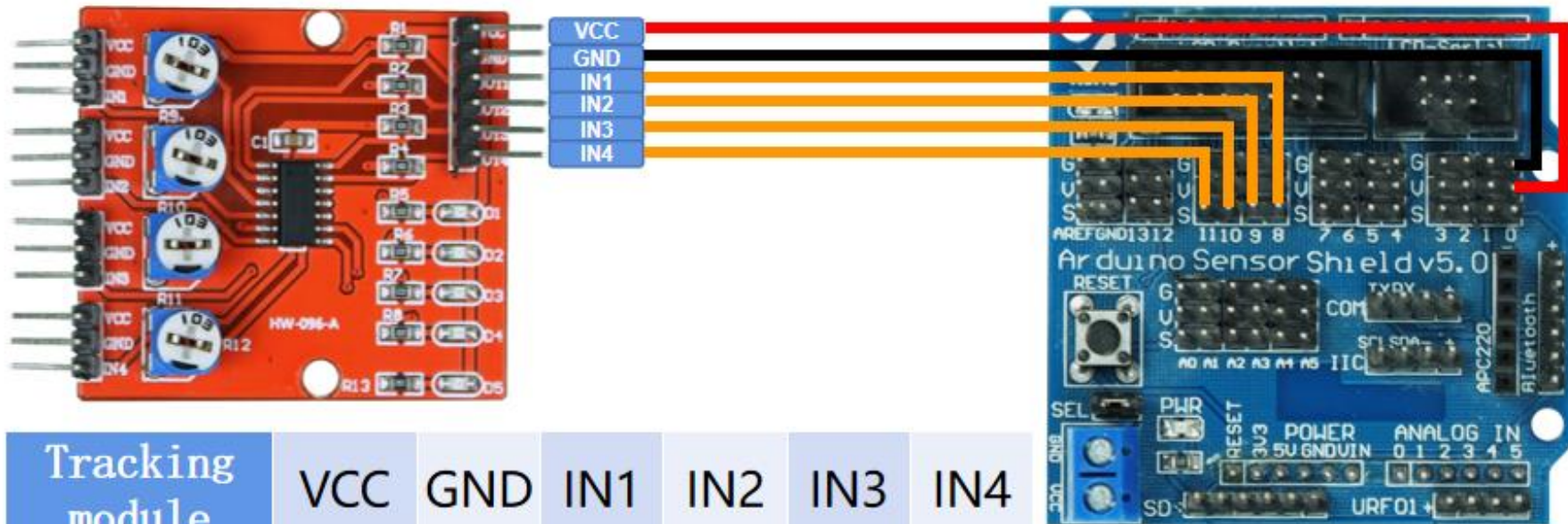
When it is found that the black line is detected but the corresponding light is not off, you need to use a small tool to turn the potentiometer (circled position), and slowly adjust to the ideal critical value. (Of course, some errors can be corrected by programming, so don't worry about it)



tracking principle

By analyzing the situation of the black line detected by the four-way tracking sensor, the motion trajectory of the car is determined. When only the rightmost sensor detects the black line, the D1 light is off, which means that the car is moving to the left of the black line. At this time, it is necessary to turn to the right to adjust. Similarly, when the leftmost sensor detects a black line, it needs to turn left to adjust. The car is allowed to go straight when the two middle sensors detect the black

Expansion board wiring diagram



Tracking module	VCC	GND	IN1	IN2	IN3	IN4
Expansion board	V	G	S(8)	S(9)	S(10)	S(11)

code analysis

Open the code file (path: 4_Arduino_Code\4_tracking\4_tracking.ino)

Define four tracking sensors and set pins as inputs

```
int Sensor1;  
int Sensor2;  
int Sensor3;  
int Sensor4;
```

```
pinMode ( 8 , INPUT);  
pinMode ( 9 , INPUT);  
pinMode ( 10 , INPUT);  
pinMode ( 11 , INPUT);
```

The body of the loop function obtains the detection information of the train of thought tracking module, and performs the motion control of the car according to the detection result (the high level means that the black line is detected).

```
void loop ()  
{  
  
Sensor1 = digitalRead ( 8 );  
Sensor2 = digitalRead ( 9 );
```

```
Sensor3 = digitalRead(10);
Sensor4 = digitalRead(11);

if ((Sensor4 == HIGH || Sensor3 == HIGH) && (Sensor2 == LOW || Sensor1 == LOW))
{
    analogWrite(ENA, 150);
    analogWrite(ENB, 150);
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    Serial.println("left");
}

else if((Sensor4 == LOW || Sensor3 == LOW) && (Sensor2 == HIGH || Sensor1 == HIGH))
{
    analogWrite(ENA, 150);
    analogWrite(ENB, 150);

    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, HIGH);
    Serial.println("right");
}

else if(Sensor4 == LOW && Sensor3 == LOW && Sensor2 == LOW && Sensor1 == LOW)
```



```
{
  analogWrite(ENA, 180);
  analogWrite(ENB, 180);
  //FORWARD
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  Serial.println("run");
}
else if((Sensor3 == HIGH || Sensor2 == HIGH ) && Sensor4 == LOW && Sensor1 == LOW)
{
  analogWrite(ENA, 180);
  analogWrite(ENB, 180);
  //FORWARD
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  Serial.println("run");
}
else
{
  analogWrite(ENA, 0);
  analogWrite(ENB, 0);
  digitalWrite(IN1, LOW);
```

```
digitalWrite(IN2,LOW);  
digitalWrite(IN3, LOW);  
digitalWrite(IN4, LOW);  
}  
}
```