

# WELCOME

Thank you very much for choosing our products.

The data package we provide is a zip compression package. Please unzip the data package before learning. Start from this PDF course.

## Technical support

If you have any questions about the product, don't worry. You can view the "Common problem.PDF" file or contact us.

Our email is [straysnail-wiki@outlook.com](mailto:straysnail-wiki@outlook.com)

We will reply you within one working day and give you a reasonable solution.

## Safety matters

- This product is recommended for children over 6 years old.
- This product needs battery power supply. It is not waterproof. Don't play in water.
- Turn off the power switch when not in use.

## About Stray Snail

- Stray Snail is the brand trademark of Shenzhen Snail Man Intelligent Technology Co., Ltd.
- Mainly for STEAM education products, self developed, including hardware and software design.
- The main products are smart cars, 3D printers and DIY writing plotters, mechanical arms, and some e-learning kits.
- The main control board includes Arduino, raspberry pie, Micro-bit and ESP series.
- Provide customization services: hardware customization, such as the design of sensor modules or the entire product.
- Software customization: Bluetooth and WiFi control APP of Android and iOS, and secondary development customization of scratch3.0 graphical programming.

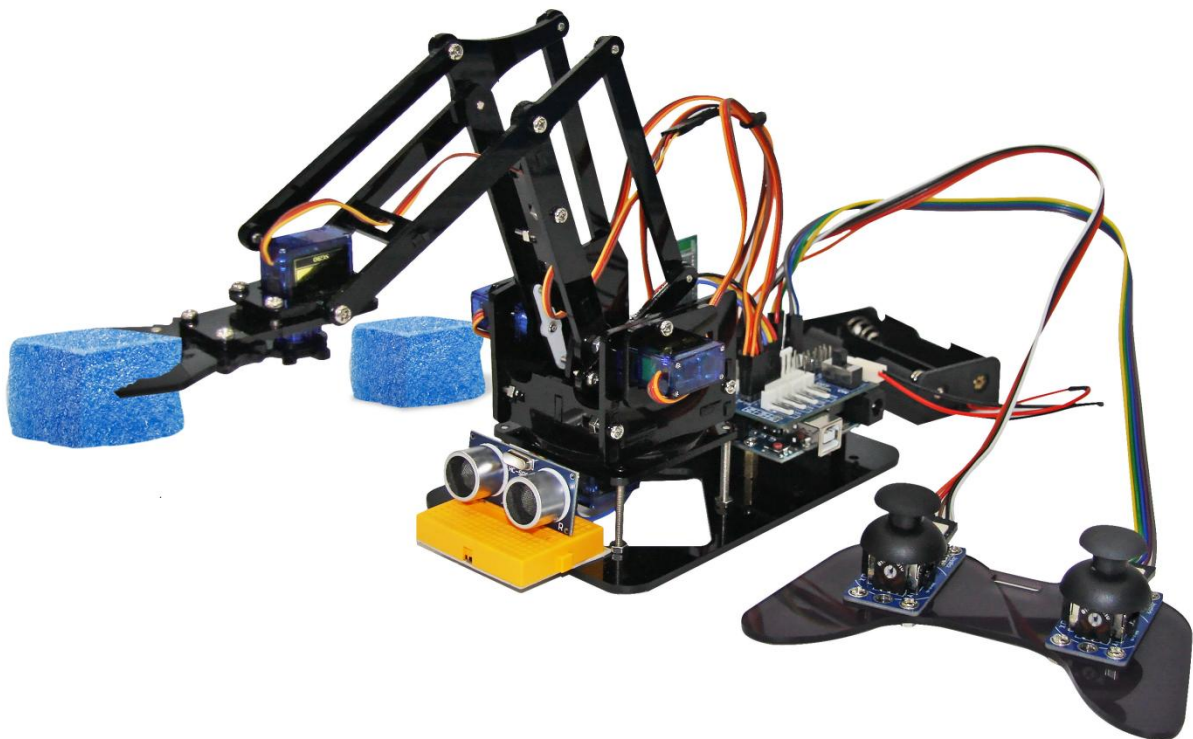
If you want to know more about Stray Snail, you can visit our official website:

<https://www.straysnail.com>

## Copyright

Our product design is patented, and trademarks, materials and software are released under CC agreement.

CC protocol is 《 Attribution-ShareAlike 3.0 Unported 》 .That is, it cannot be used for commercial purposes without our permission.



## I . Product introduction

With the increase of labor costs, the production and manufacturing of many products are gradually mechanized, and the application of mechanical arm is the most extensive, such as parts processing, assembly, handling, welding, spraying, etc. It can replace workers in repetitive, high-intensity and high-risk operations, improve production efficiency, reduce production costs, and improve product quality and reliability.

We designed this robot arm for programming learning, which can learn the structure and programming of the robot arm at a low cost, easy to realize simple automatic handling. We also provide the robot arm teaching function, that is, we can teach the robot arm some actions, and then the robot arm performs the specified actions, which will be very interesting to play.

It supports multiple control modes, such as ultrasonic detection control, rocker handle control, iOS and Android mobile phone control, etc.

# Catalog

WELCOME .....	1
Technical support .....	1
Safety matters .....	1
About Stray Snail .....	1
Copyright .....	2
I . Product introduction .....	3
II. Product features .....	5
III. Product parameters .....	6
IV. Install software and its environment configuration .....	7
1. Install Arduino IDE and serial port driver .....	7
2. Load library file .....	11
3. Turn on the on-board LED of Arduino UNO .....	13
V. Assemble the mechanical arm .....	16
VI. Learn to control the mechanical arm .....	16
1. Control a steering gear .....	17
1.1 Steering gear rotation .....	18
1.2 Use of Servo steering gear library files .....	19
2. Set the handling action of the mechanical arm .....	20
2.1 Realize automatic handling action .....	21
3. Detect objects and carry them automatically .....	26

3.1 Read the distance values measured by the ultrasonic sensor .....	27
3.2 Detect the presence of goods and carry them automatically .....	30
4. Rocker handle controls the mechanical arm .....	36
4.1 Read the values of the rocker handle .....	36
4.2 Rocker controls the mechanical claw .....	38
4.3 Rocker controls the 4-axis mechanical arm .....	40
4.4 Rocker teaching function of the mechanical arm 1 .....	48
4.5 Rocker teaching function of the mechanical arm 2 .....	49
4.6 Rocker teaching function of the mechanical arm 3 .....	49
5. APP controls the mechanical arm .....	50
5.1 Download and read the values of APP .....	52
5.2 Mobile APP virtual rocker controls the rotation of the base .....	57
5.3 Mobile APP drag bar real-time controls the mechanical claw .....	59
5.4 Mobile APP controls the mechanical arm 1 .....	62
5.5 Mobile APP controls the mechanical arm 2 .....	66
5.6 Mobile APP teaching function of the mechanical arm .....	71

## II. Product features

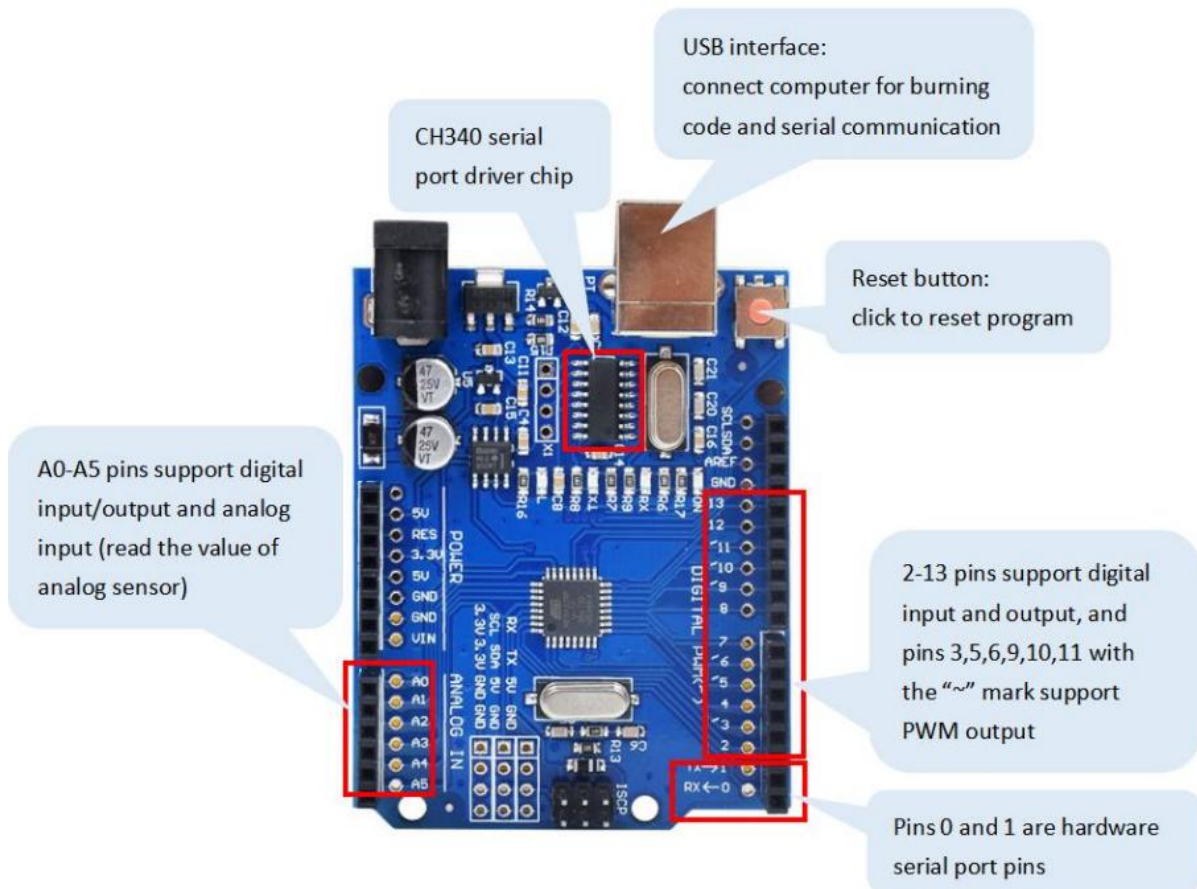
- Beautiful appearance, stable structure, and a mechanical aesthetic feeling when operating.
- Automatic sorting system, which uses ultrasound to identify objects at different distances and place them in designated locations.
- Rocker control, achieving teaching function, teaching the mechanical arm to achieve specified actions.

- Support for Apple iOS and Android mobile APP control.

### III. Product parameters

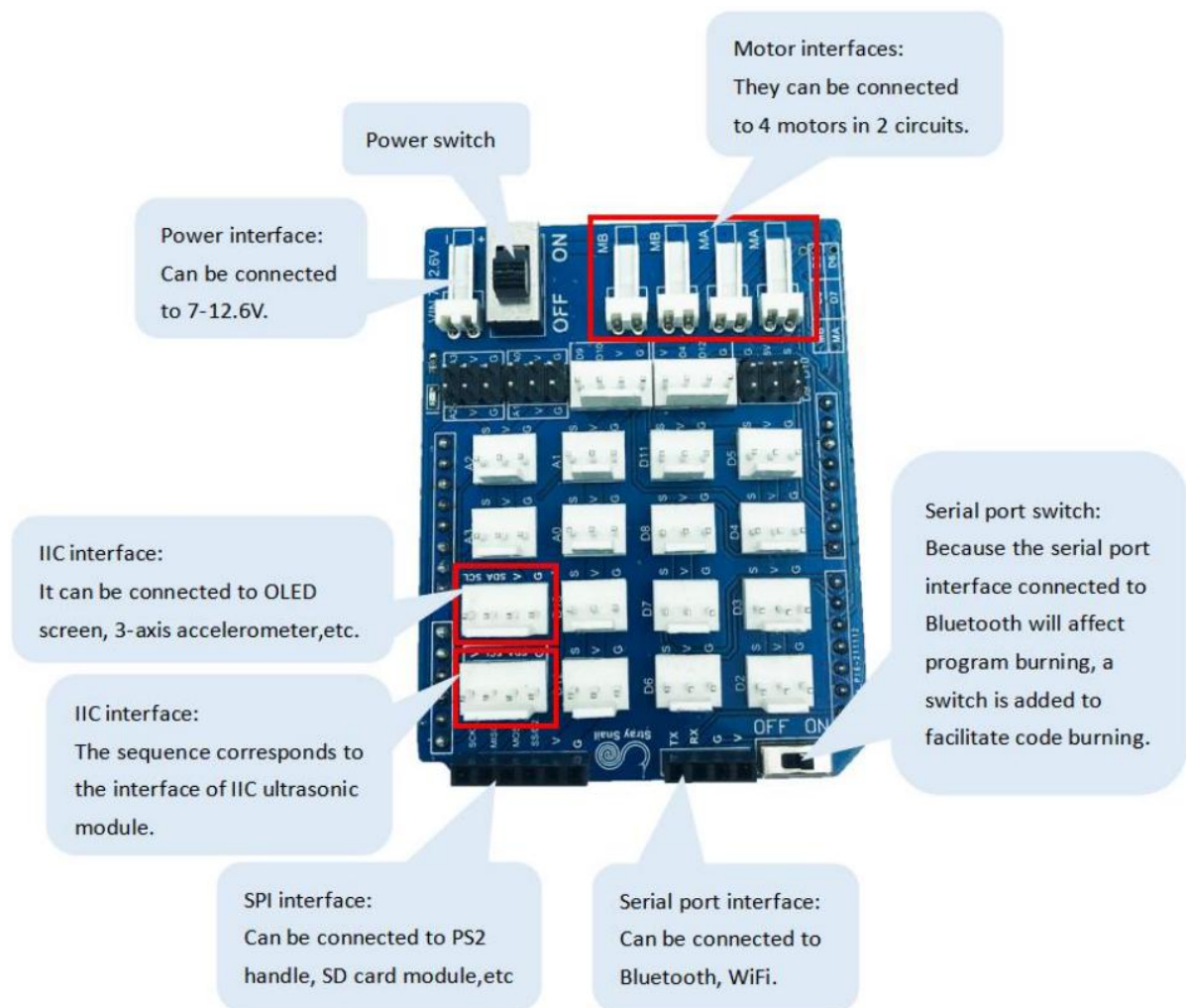
- Input voltage of USB interface of Arduino UNO main board : 5V
- Input voltage of external power interface : 7~12.6V
- Working voltage of all sensor modules : 5V

The illustration of Arduino UNO main board is as follows :



The illustration of expansion board is as follows : Our expansion board is equipped with a TB6612FNG motor driver chip, which can control two motors. Four motor interfaces are led out, allowing you to DIY the robot arm with the mobile platform yourself.





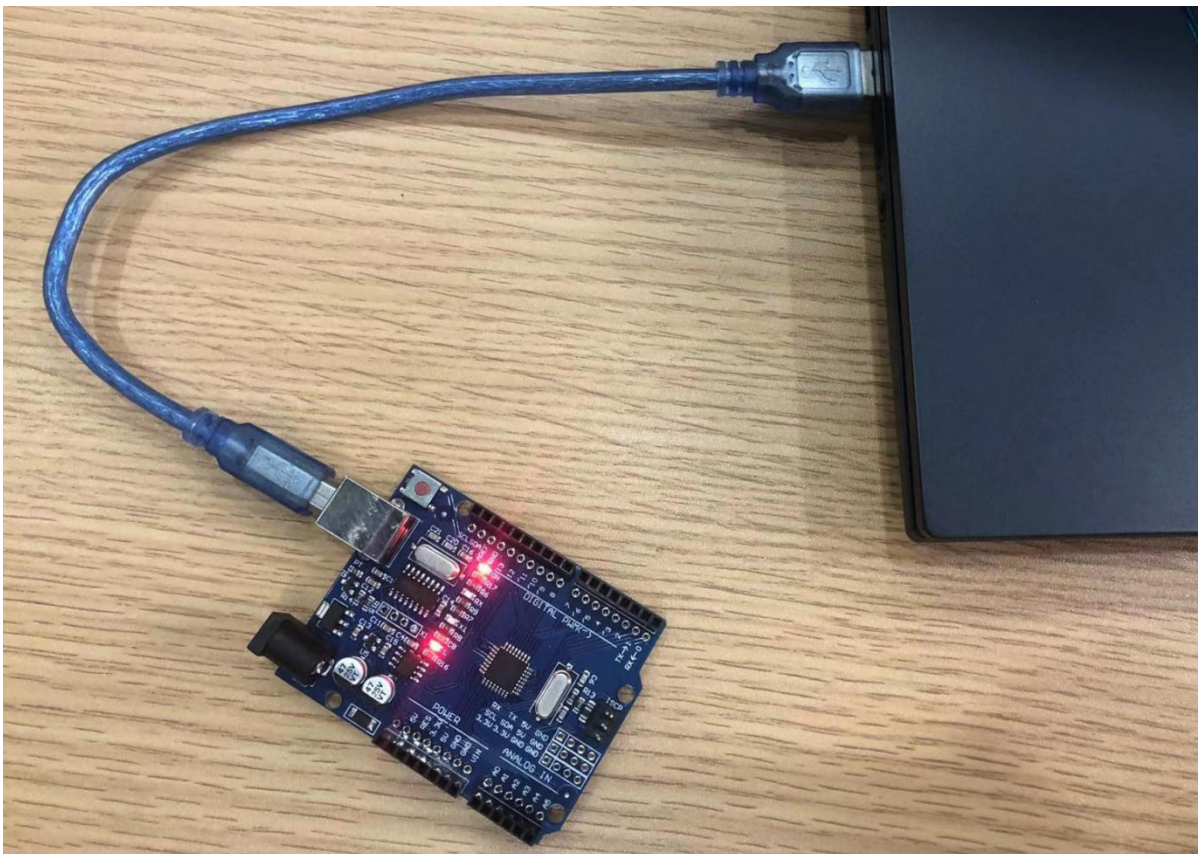
## IV. Install software and its environment configuration

### 1. Install Arduino IDE and serial port driver

(1) If you are still a novice, please open the material provided by us and follow to learn the installation of the Arduino IDE software.

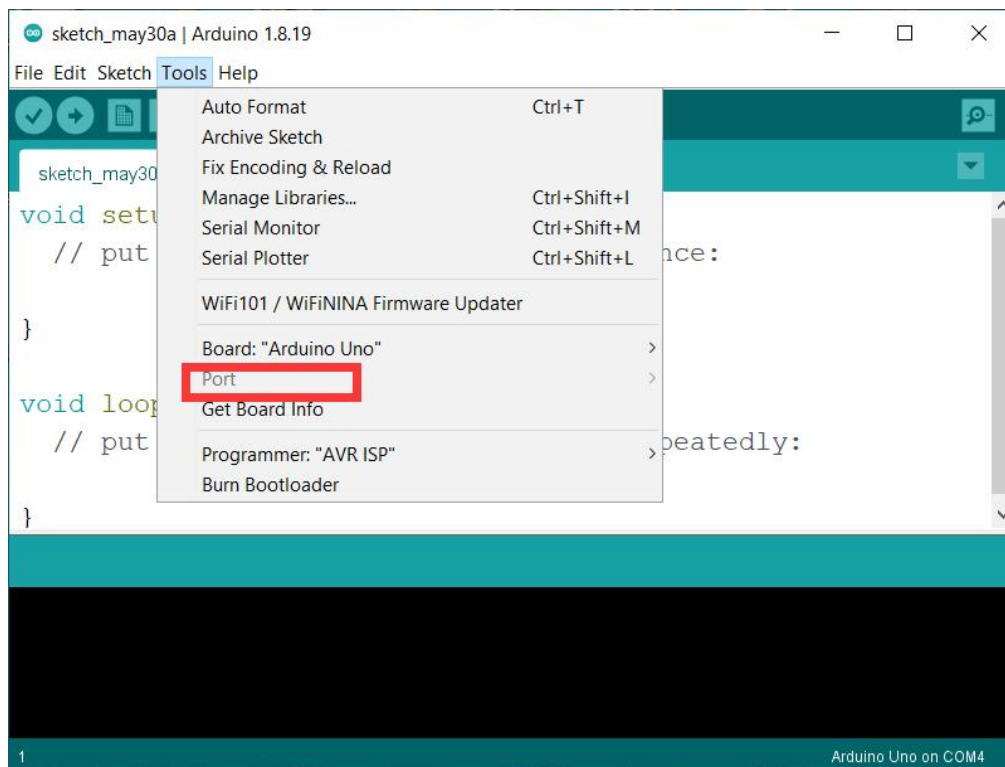
<< Mech... > 1. Install Arduino IDE and CH340 driver a...	
Name	Date modified
Installation tutorial of CH340 driver	5/30/2023 11:26 AM
Libraries	5/30/2023 11:27 AM
Install Arduino IDE in MacOS.docx	2/1/2023 9:44 AM
Installation tutorial of Arduino IDE.docx	2/1/2023 9:44 AM

(2) Connect the USB data cable to the Arduino UNO main board and the USB of the computer.

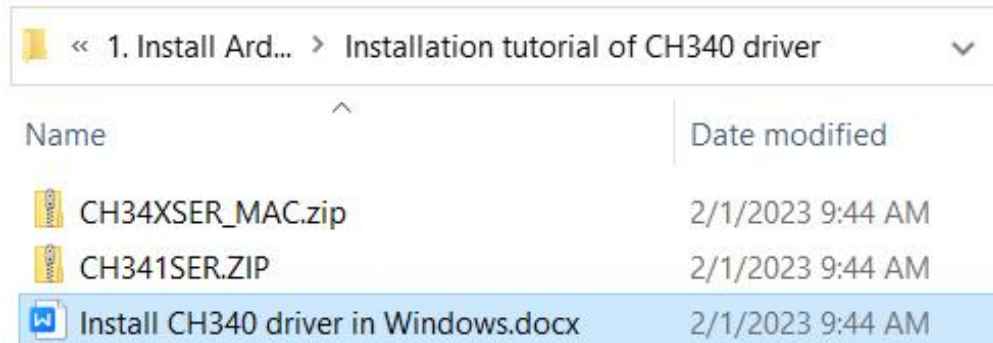


Open the Arduino IDE and click “Tools”. If the CH340 driver cannot be automatically identified, as shown in the following figure, you need to manually install the CH340 driver.

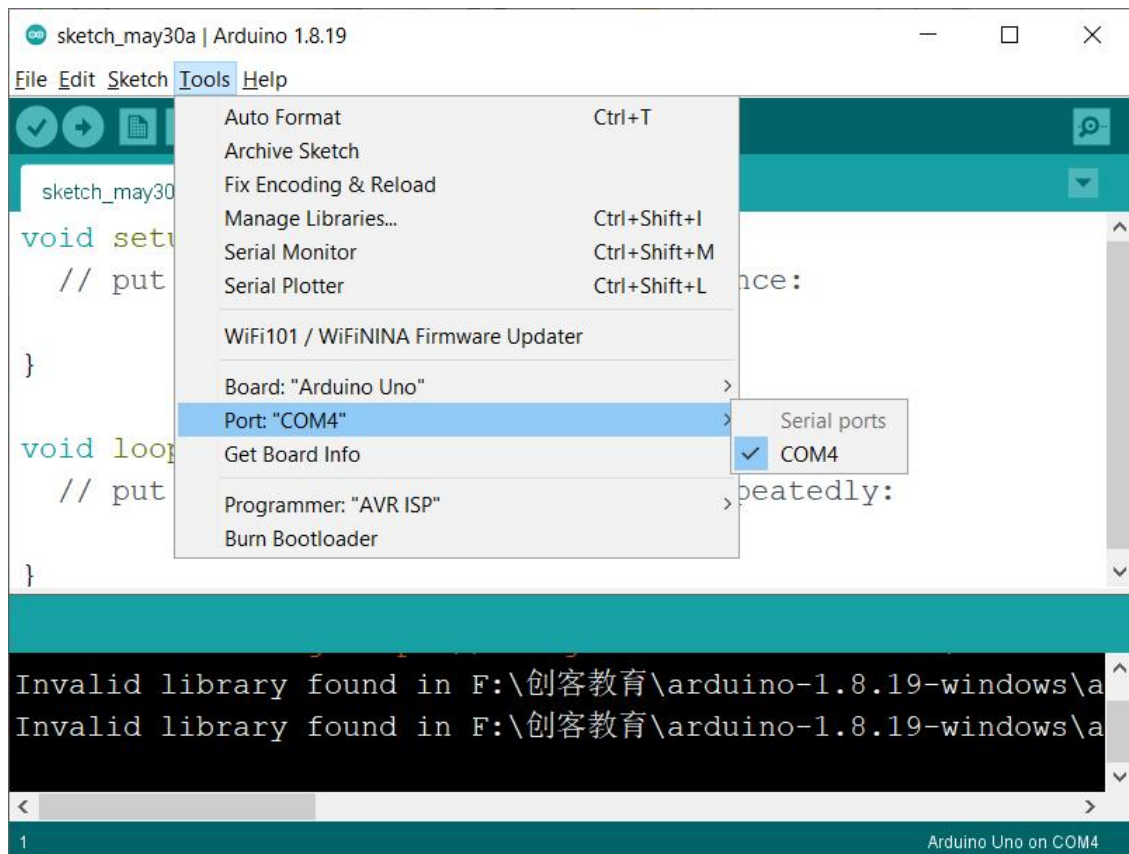




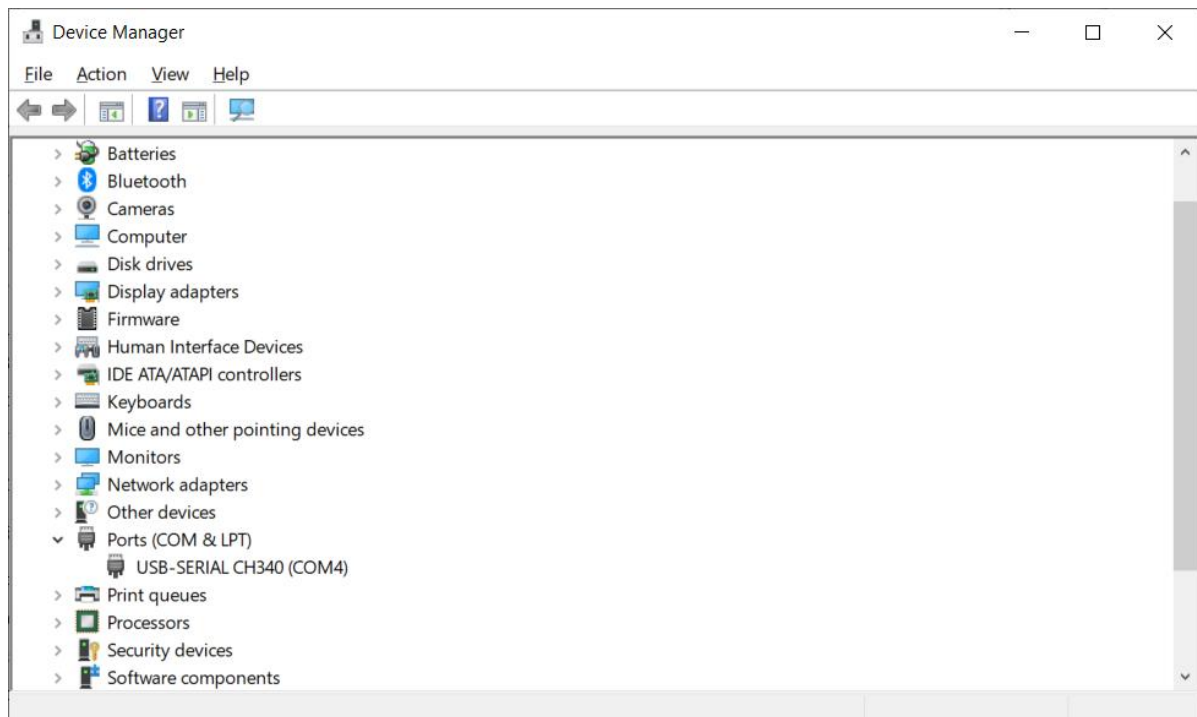
In the material, we provide a CH340 driver installation tutorial. Open the document and follow the steps to install the driver.



If it can be identified, as shown in the figure below.



Open the device manager of the computer to see the CH340 driver, as shown in the following figure.

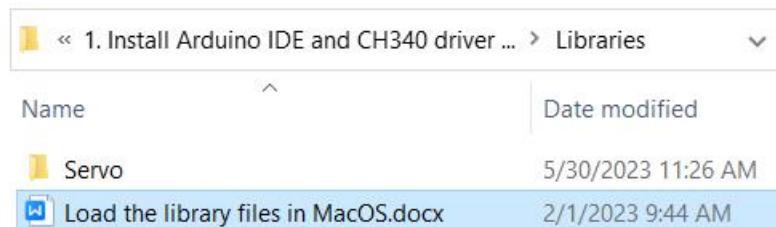


## 2. Load library file

After installing the software, load the library files provided by us, which will be used in later code learning.

### MacOS system loading library:

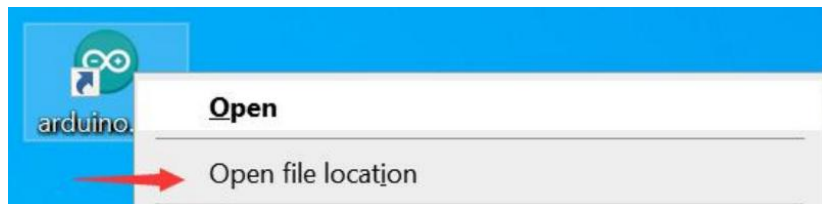
Please open “Load the library files for MacOS”, as shown below.



### Windows system loading library:

Open the library folder provided by us, copy all the library files, and paste them into “libraries” in the Arduino IDE, as shown in the following figure.

(1) Right click the icon of the Arduino IDE on the desktop and select “open file location”.



You will enter the installation location of the Arduino IDE and open the “libraries” folder.

<< arduino-1.8.19-windows > arduino-1.8.19 >			
Search arduino-1.8.19			
Name	Date modified	Type	Size
drivers	2/5/2023 2:52 PM	File folder	
examples	2/5/2023 2:52 PM	File folder	
hardware	2/5/2023 2:52 PM	File folder	
java	2/5/2023 2:52 PM	File folder	
lib	2/5/2023 2:52 PM	File folder	
libraries	5/29/2023 4:27 PM	File folder	
tools	2/5/2023 2:53 PM	File folder	
tools-builder	2/5/2023 2:53 PM	File folder	
arduino.exe	2/5/2023 2:52 PM	Application	72 KB
arduino.l4j.ini	2/5/2023 2:52 PM	Configuration setti...	1 KB
arduino_debug.exe	2/5/2023 2:52 PM	Application	69 KB
arduino_debug.l4j.ini	2/5/2023 2:52 PM	Configuration setti...	1 KB
arduino-builder.exe	2/5/2023 2:52 PM	Application	23,156 KB
libusb0.dll	2/5/2023 2:52 PM	Application extens...	43 KB
msvcp100.dll	2/5/2023 2:52 PM	Application extens...	412 KB
msvcr100.dll	2/5/2023 2:52 PM	Application extens...	753 KB
revisions.txt	2/5/2023 2:52 PM	Text Document	97 KB
wrapper-manifest.xml	2/5/2023 2:52 PM	XML Document	1 KB

<< arduino-1.8.19 > libraries >			
Search libraries			
Name	Date modified	Type	Size
Adafruit_Circuit_Playground	2/5/2023 2:52 PM	File folder	
Adafruit_GFX	2/5/2023 3:03 PM	File folder	
Adafruit_GFX_Library	5/19/2023 5:28 PM	File folder	
Adafruit_NeoPixel-master	2/5/2023 3:03 PM	File folder	
Adafruit_SSD1306-master	2/5/2023 3:03 PM	File folder	
arduino-ds1302-master	5/29/2023 4:27 PM	File folder	
Bridge	2/5/2023 2:52 PM	File folder	
DallasTemperature	5/29/2023 3:20 PM	File folder	
dht11-master	3/22/2023 12:12 PM	File folder	

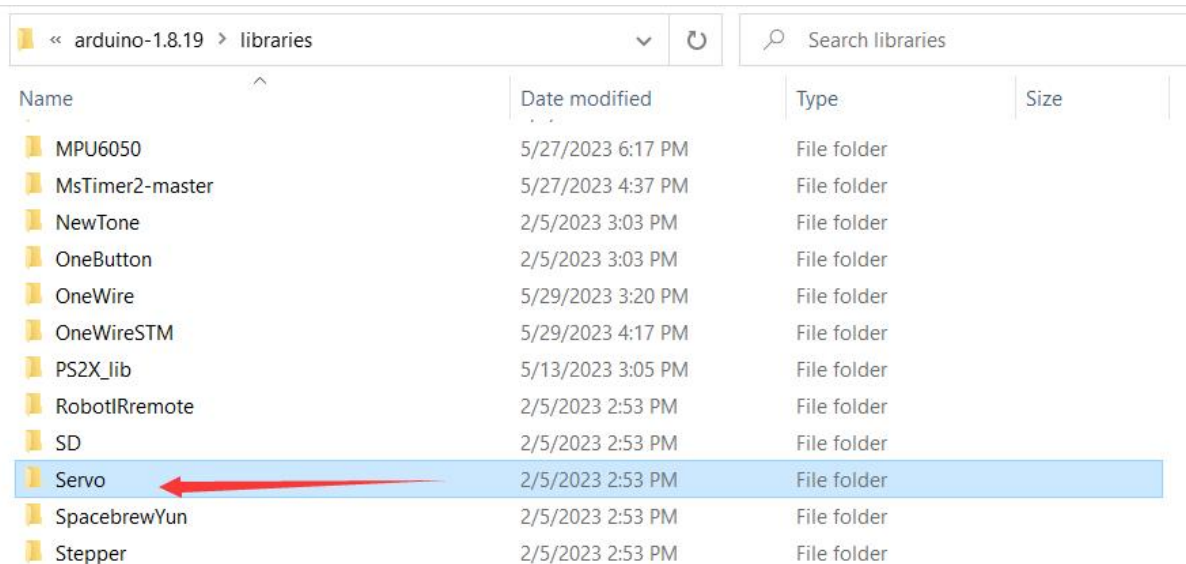
(2)Open the library folder provided by us and copy all the library files.



The screenshot shows the 'Libraries' panel in the Arduino IDE. The breadcrumb path is '<< 1. Install Arduino IDE and CH340 driver ... > Libraries'. The table lists two items: 'Servo' and 'Load the library files in MacOS.docx'. The 'Servo' entry is highlighted with a red rectangle. The 'Date modified' column shows '5/30/2023 11:26 AM' for 'Servo' and '2/1/2023 9:44 AM' for the document.

Name	Date modified
Servo	5/30/2023 11:26 AM
Load the library files in MacOS.docx	2/1/2023 9:44 AM

(3) Paste them into the “libraries” folder of the Arduino IDE you just opened. Done.



The screenshot shows the 'libraries' folder in the Arduino IDE. The breadcrumb path is '<< arduino-1.8.19 > libraries'. The table lists various libraries, including 'MPU6050', 'MsTimer2-master', 'NewTone', 'OneButton', 'OneWire', 'OneWireSTM', 'PS2X\_lib', 'RobotIRremote', 'SD', 'Servo', 'SpacebrewYun', and 'Stepper'. The 'Servo' entry is highlighted with a blue background and a red arrow pointing to it. The 'Date modified' column shows '2/5/2023 2:53 PM' for 'Servo'.

Name	Date modified	Type	Size
MPU6050	5/27/2023 6:17 PM	File folder	
MsTimer2-master	5/27/2023 4:37 PM	File folder	
NewTone	2/5/2023 3:03 PM	File folder	
OneButton	2/5/2023 3:03 PM	File folder	
OneWire	5/29/2023 3:20 PM	File folder	
OneWireSTM	5/29/2023 4:17 PM	File folder	
PS2X_lib	5/13/2023 3:05 PM	File folder	
RobotIRremote	2/5/2023 2:53 PM	File folder	
SD	2/5/2023 2:53 PM	File folder	
Servo	2/5/2023 2:53 PM	File folder	
SpacebrewYun	2/5/2023 2:53 PM	File folder	
Stepper	2/5/2023 2:53 PM	File folder	

### 3. Turn on the on-board LED of Arduino UNO

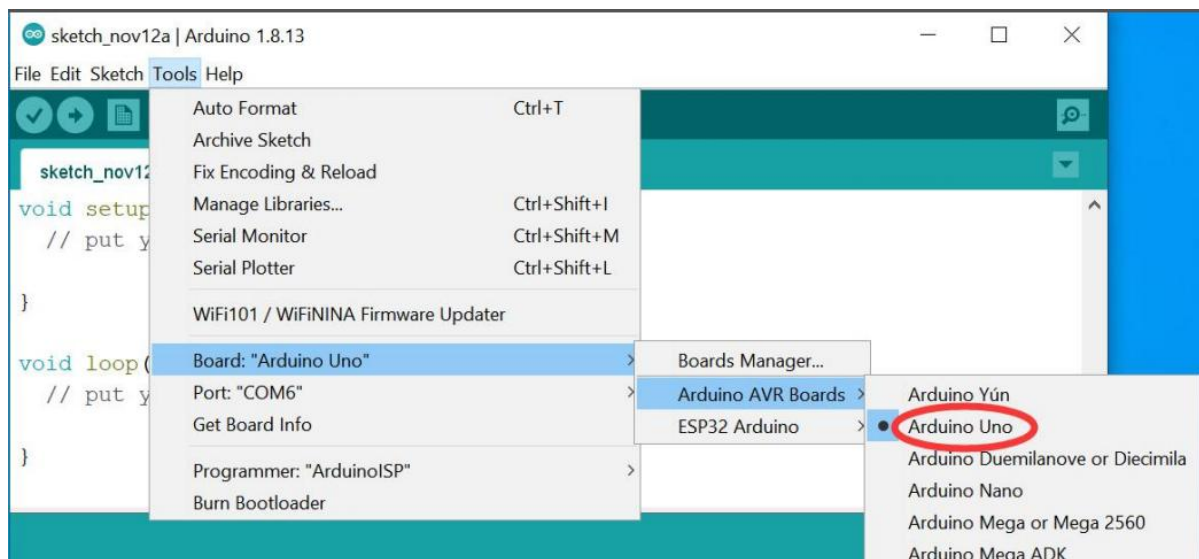
Arduino IDE comes with a lot of example codes. Now we learn to open the code that makes the LED flash in the example and burn it to the Arduino UNO main board.

(1) Take out the data cable provided by us, and connect the computer USB and Arduino UNO USB interface.

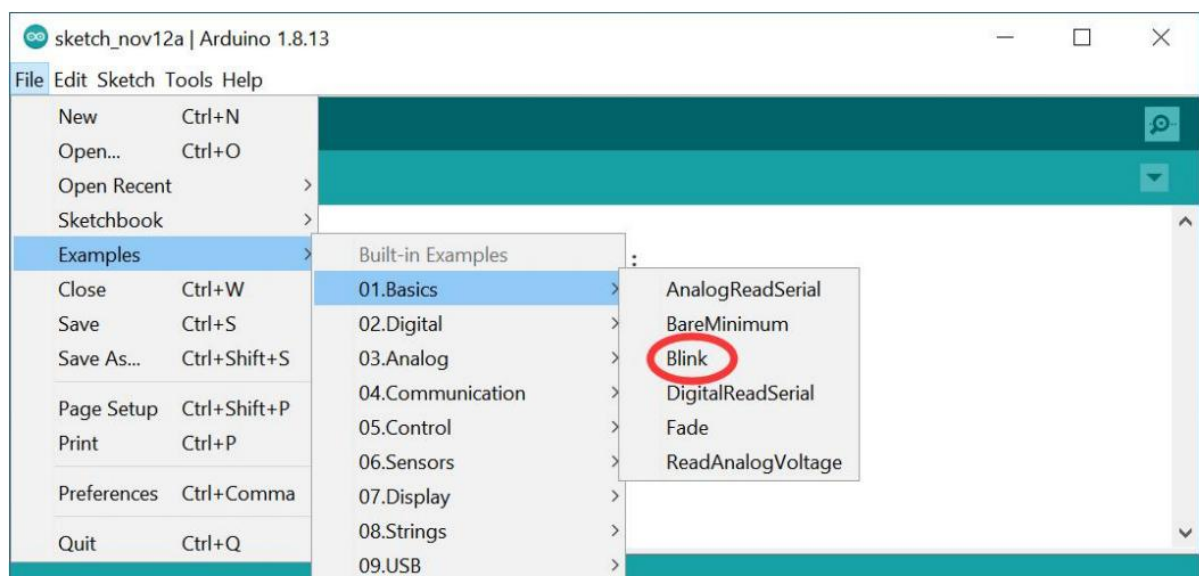
(2) Open the Arduino IDE and select the main board and COM port in “Tool”

Select “Arduino UNO” as the main board. Pay attention to the COM port. Select the one recognized by your computer, which my computer recognizes is COM4.

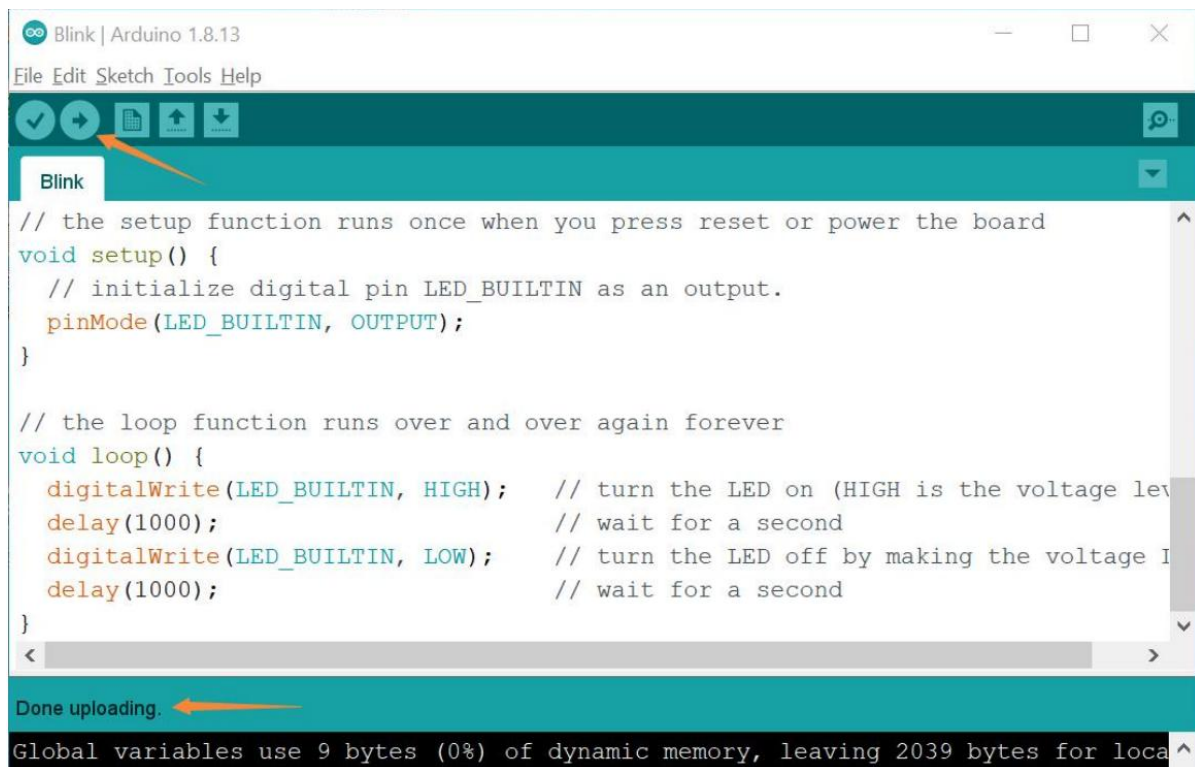




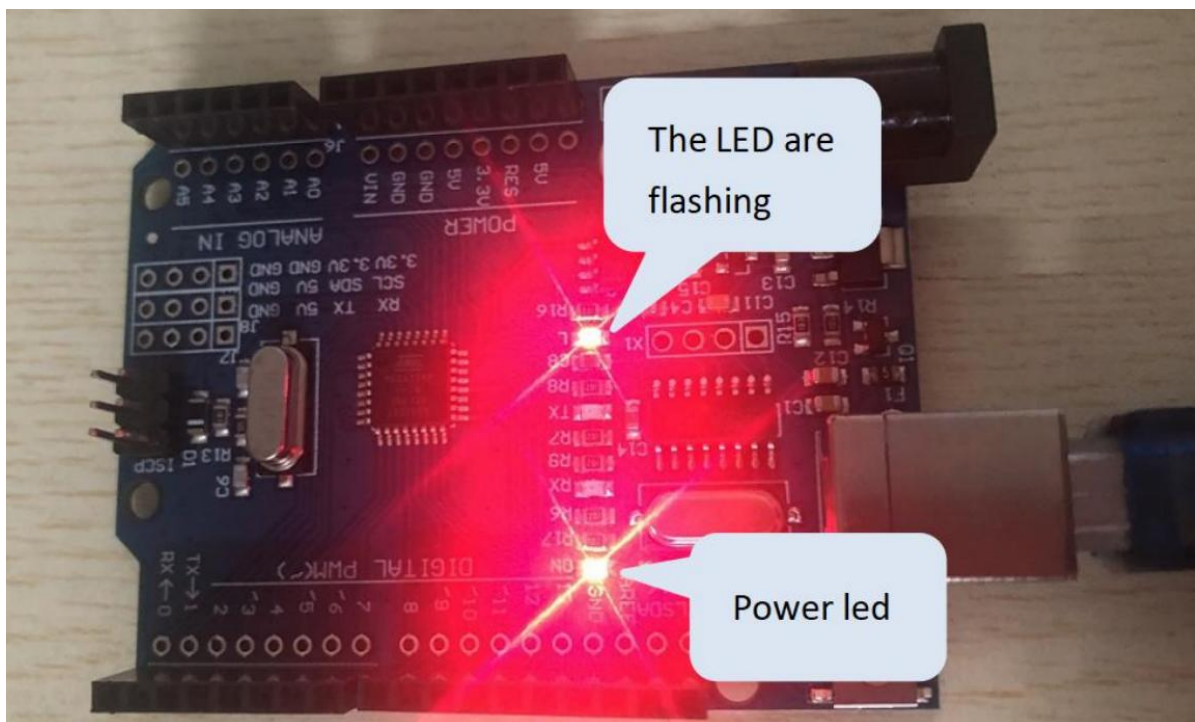
(3) Open the example code coming with the Arduino IDE for turning on the flashing LED.



(4) Click the upload button directly to upload the code to the Arduino UNO main board. If it is successful, you will see that "The upload is successful".



(5) After successful burning, normally you can see the on-board LED of Arduino UNO flashing.



By now, I believe you have a basic understanding of the operation of the Arduino IDE.

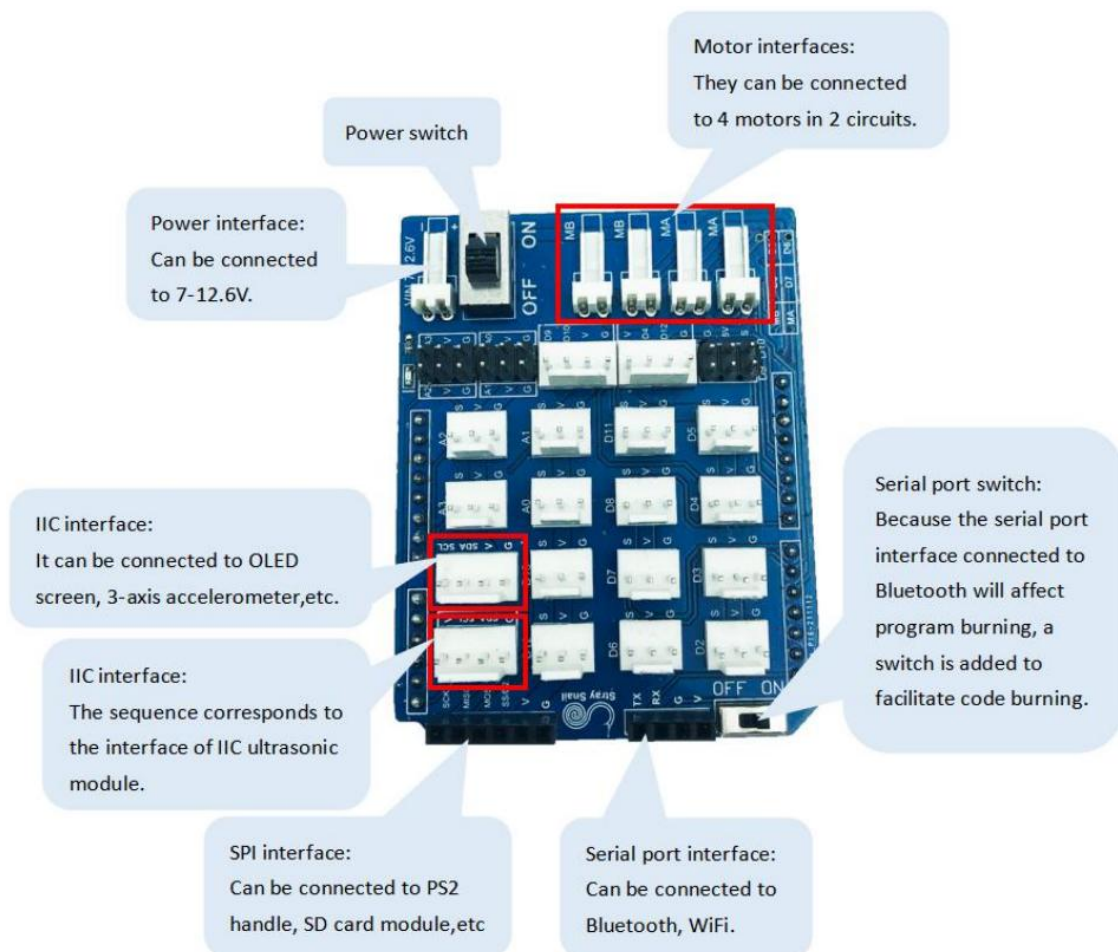
## V. Assemble the mechanical arm

Please open the tutorial we provide for installing the mechanical arm and follow the steps to install it.

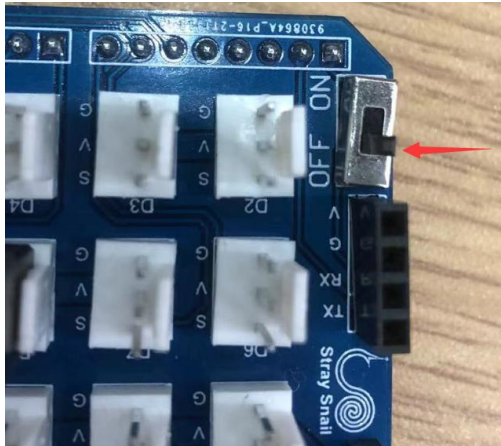
**It must be noted that before installing the steering gear, the code must be uploaded to rotate the steering gear to the specified angle.**

Mechanical arm >	Search Mechanical arm
Name	Date modified
1. Install Arduino IDE and CH340 driver and loa...	5/30/2023 11:26 AM
2.Installation of mechanical arm	5/30/2023 11:28 AM
3.Sample code	5/25/2023 6:01 PM
4.Android APP	5/25/2023 6:01 PM
Main course of mechanical arm.docx	5/30/2023 11:53 AM

## VI. Learn to control the mechanical arm



Note: The Bluetooth switch needs to be turned off in order for the code to upload successfully.



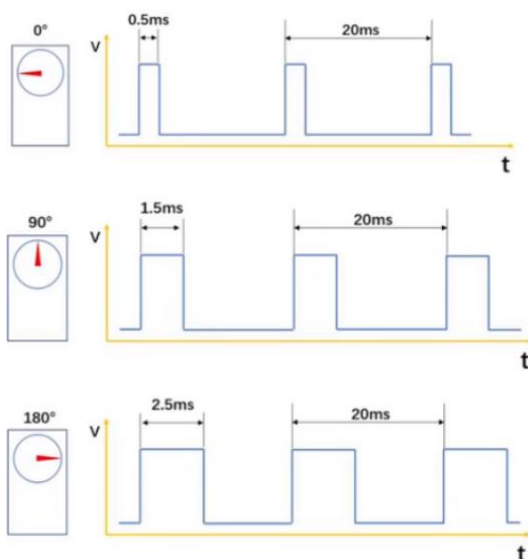
The official learning website for Arduino C language:

<https://www.arduino.cc/reference/en/>

## 1. Control a steering gear

**Steering gear** is a kind of servo driver, which can control the rotation angle accurately. It is widely and commonly used in the joint movement of robots.

**Principle of steering gear control:** The angle of the steering gear is determined by the duration of the control signal pulse, which is called pulse code modulation(PCM). The control of the steering gear generally requires a time base pulse of about 20ms. The high level time in the time base pulse is within the range of 0.5ms~2.5ms, and the corresponding angle for controlling the steering gear is 0~180 degrees, as shown in the figure below.



Steering gear	The IO port of main board or expansion board to connect
Steering gear in the base	D7

Because the steering gear in the base has no angle limit, no matter how much the control angle is, it will not damage the components. So we will first control the base steering gear.

## 1.1 Steering gear rotation

Here, we will write code to control the steering gear based on the control principle mentioned above, achieving a 0-180 degree rotation of the it.

### (1) Example code

```

/*
 * create by straysnail
 * 2023/2/21
 */
#define door_servoPin 7 //Define the pin of the steering gear
int pulsewidth = 0; //Variable, used to calculate the impulse value of the steering gear

void setup() {
    pinMode(door_servoPin, OUTPUT); //Set the pin to output
}

void loop() {
    procedure(door_servoPin, 0); //The steering gear rotates to 0 degree
    delay(300); //Time for steering gear rotation
    procedure(door_servoPin, 90); //90 degrees
    delay(300);
    procedure(door_servoPin, 180); //180 degrees
    delay(300);
    procedure(door_servoPin, 90); //180 degrees
    delay(300);
}

//The function to control the steering gear angle based on the principle of steering gear

```



```
void procedure(int serPin, int myangle)  //The function has two parameters,
serPin is the pin of the steering gear, and myangle is the angle of the steering
gear
{
    for(int i=0; i<10; i++)
    {
        //Calculate the pulse value, which is the time of the high level. The range
of the myangle value is 0-180, corresponding to the pulse width range of 500-2480
        pulsewidth = myangle * 11 + 500;
        digitalWrite(serPin,HIGH);
        delayMicroseconds(pulsewidth); //Calculated high-level time delay
        digitalWrite(serPin,LOW); //Set to low level
        delay((20 - pulsewidth / 1000)); //Delay the remaining low level time
    }
}
```

## (2) Experiment phenomenon

The angle sequence of the steering gear rotation is 0-90-180-90, and it rotates continuously.

**Tips:** *“void procedure(int serPin, int myangle)” is a sub function in the code.*

*Sub functions are often reused in code. Just write a sub function and call it*

*again, which will make the code look concise. Therefore, be familiar with the*

*use of sub functions.*

## 1.2 Use of Servo steering gear library files

Using servo steering gear library files will make the code more concise and simple, and the internal timer is used to keep the steering gear at the specified angle.

First, control the steering gear in the base to rotate.

### (1) Example code

```
#include <Servo.h>

Servo myservo;  // Create a steering gear instance to control

int pos = 0;    // Variable, used for steering gear angle

void setup() {
  myservo.attach(7); // Initialize the steering gear
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // Turning the steering gear from 0 degrees to 180
    degrees
    // One step forward is one degree
    myservo.write(pos);           // Control the steering gear to rotate to the degrees
    of pos variable
    delay(15);                    // Delay function, controlling the speed of the steering
    gear rotation
  }
  for (pos = 180; pos >= 0; pos -= 1) { //Control the steering gear to rotate from 180 degree
    to 0
    myservo.write(pos);
    delay(15);
  }
}
```

## (2) Experiment phenomenon

The base of the mechanical arm rotates back and forth.

## 2. Set the handling action of the mechanical arm

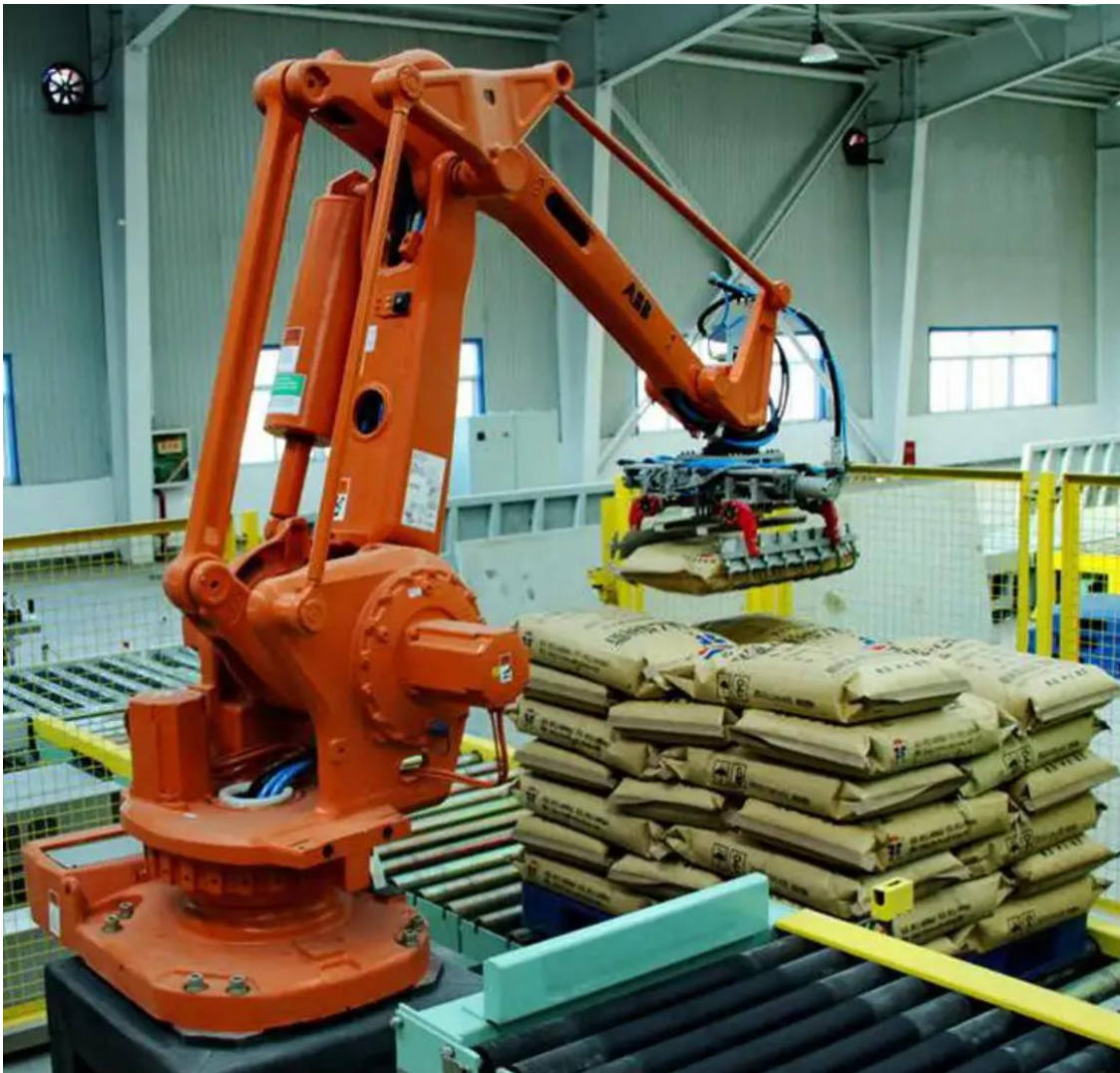
The rotation directions of each joint of the mechanical arm are shown in the table below:

Steering gear	When going to 0 degree	When going to 180 degree
Steering gear 1	Turn to the far right	Turn to the far left

(Base)		
Steering gear 2 (Left)	Swing the rod connecting the steering gear 2 forward ( Cannot be lower than 90 degrees to avoid damaging the mechanical arm)	Swing the rod connecting the steering gear 2 backward ( Cannot be higher than 160 degrees to avoid damaging the mechanical arm)
Steering gear 3 (Right)	Swing the rod connecting the steering gear 3 backward ( Cannot be lower than 30 degrees to avoid damaging the mechanical arm)	Swing the rod connecting the steering gear 3 forward ( Cannot be higher than 170 degrees to avoid damaging the mechanical arm)
Steering gear 4 (Claw)	Mechanical claw opens, and fully opens at 0 degrees	Mechanical claw closes, and fully closes at 90 degrees

## 2.1 Realize automatic handling action

Some mechanical arms in factories repeatedly perform an action to move goods, and now we use this mechanical arm to achieve this.



### (1) Programming idea

A simple set of handling actions, where the mechanical arm grabs the goods that are farther to the right and moves them to a position that is closer to the left.

- |  |
|--|
| 1.Rotate the base of the mechanical arm to the right.  |
| 2.Open the mechanical claw.  |
| 3.Swing the left steering gear back and lift the mechanical claw, also in order to grasp goods at a longer distance. |
| 4.Swing the right steering gear forward to move the mechanical claw forward and downward.                            |
| 5.Mechanical claw grasps the object.   |

6.Swing the right steering gear back to move the mechanical claw back and up.

7.Rotate the base of the mechanical arm to the left.

8.Swing the left steering gear forward to bring the goods closer.

9.Swing the steering gear on the right forward to extend and lower the mechanical claw.

10.The mechanical claw lowers the goods.

11.Swing the right steering gear back and lift and retract the mechanical claw.

12.Mechanical claw closes.

## (2) Example code

```
#include <Servo.h>
Servo myservo1; // Create 4 steering gear instance to control
Servo myservo2;
Servo myservo3;
Servo myservo4;
int pos1=90, pos2=120, pos3=90, pos4=90; //Angle value at the initial state of the mechanical arm

void setup()
{
  myservo1.attach(7); // attaches the servo on pin 9 to the servo object
  myservo2.attach(10);
  myservo3.attach(9);
  myservo4.attach(8);
  //Initialize the mechanical arm
  myservo1.write(pos1);
  myservo2.write(pos2);
  myservo3.write(pos3);
  myservo4.write(pos4);
  delay(1000);
}
```



```
void loop()
{
    //Rotate to the right
    for(pos1;pos1>0;pos1--)
    {
        myservo1.write(pos1); //Step by step, turn the steering gear to the specified angle
        delay(5);             //Delay to control the rotation speed of the steering gear
    }
    delay(500);

    // Open the claw
    for(pos4;pos4>0;pos4--)
    {
        myservo4.write(pos4);
    }
    delay(500);

    //Swing the steering gear on the left back to lift the mechanical claw up
    for(pos2;pos2<140;pos2++)
    {
        myservo2.write(pos2);
        delay(5);
    }

    // Swing the steering gear on the right forward to extend and lower the mechanical claw
    for(pos3;pos3<150;pos3++)
    {
        myservo3.write(pos3);
        delay(5);
    }
    delay(500);

    // Grasp the goods
    for(pos4;pos4<50;pos4++)
    {
        myservo4.write(pos4);
    }
    delay(500);

    // Swing the right steering gear back, lift and retract the mechanical claw
    for(pos3;pos3>90;pos3--)
    {
        myservo3.write(pos3);
```

```
    delay(5);
}
delay(500);

// Rotate to the left
for(pos1;pos1<180;pos1++)
{
    myservo1.write(pos1);
    delay(5);
}
delay(500);

//Swing the steering gear on the left forward to bring the goods closer
for(pos2;pos2>120;pos2--)
{
    myservo2.write(pos2);
    delay(5);
}

// Swing the steering gear on the right forward to extend and lower the mechanical claw
for(pos3;pos3<140;pos3++)
{
    myservo3.write(pos3);
    delay(5);
}
delay(500);

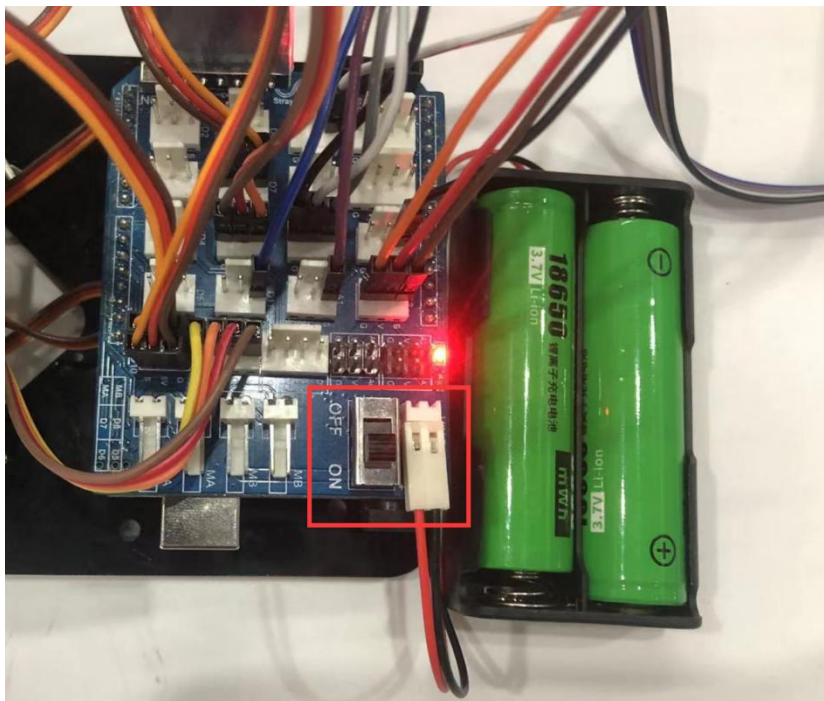
// Open the mechanical claw and lower the goods
for(pos4;pos4>0;pos4--)
{
    myservo4.write(pos4);
}
    delay(500);

// Swing the right steering gear back, lift and retract the mechanical claw
for(pos3;pos3>90;pos3--)
{
    myservo3.write(pos3);
    delay(5);
}
delay(500);
// Mechanical claw closes
```

```
for(pos4;pos4<90;pos4++)  
{  
    myservo4.write(pos4);  
}  
delay(500);  
}
```

### (3) Experiment phenomenon

Note: It is necessary to connect an external power supply, otherwise the USB cable current is not high enough and the robotic arm cannot operate.



The mechanical arm repeats the execution, grabbing the far goods on the right and moving it to a closer position on the left.

## 3. Detect objects and carry them automatically

In the previous lesson, we implemented a simple handling action, but we found that

the mechanical arm would perform that set of actions regardless of whether there were goods present or not. This would be a waste of energy in practical use. Therefore, in this lesson, we will add a step to detect the presence of goods. Detecting goods requires sensors. We use an ultrasonic module, which has the advantage of accurately detecting the distance of obstacles ahead.

### 3.1 Read the distance values measured by the ultrasonic sensor



**Ultrasonic wave:** The ultrasonic module is almost a must-have for maker robots, simple and practical. Looks like eyes, with an effective detection distance of within 2 meters.

One eye of the ultrasonic module emits a certain frequency of ultrasound and the other receives reflected ultrasound. The speed of sound waves is mainly related to the propagation medium and temperature. We often use 340m/s, which is the speed at which sound waves propagate in standard air at 15 °C. We also need to know the time from transmitting sound waves to receiving sound waves to calculate the distance of the object in front.

The calculation formula is:  $\text{distance} = 340\text{m/s} * t/2$

**Little knowledge:** The frequency of sound heard by the human ear has a range, and when the frequency of sound exceeds the range of human hearing, it belongs to ultrasound.

#### (1) Example code

```
#define trigPin 4 //Ultrasonic emission pin
#define echoPin 12 //Ultrasonic receiving pin
long duration, distance; //Variable
```

```
void setup() {
  Serial.begin(9600);          //Set the baud rate to 9600
  pinMode(trigPin, OUTPUT);    //Set to output
  pinMode(echoPin, INPUT);     //Set to input
}

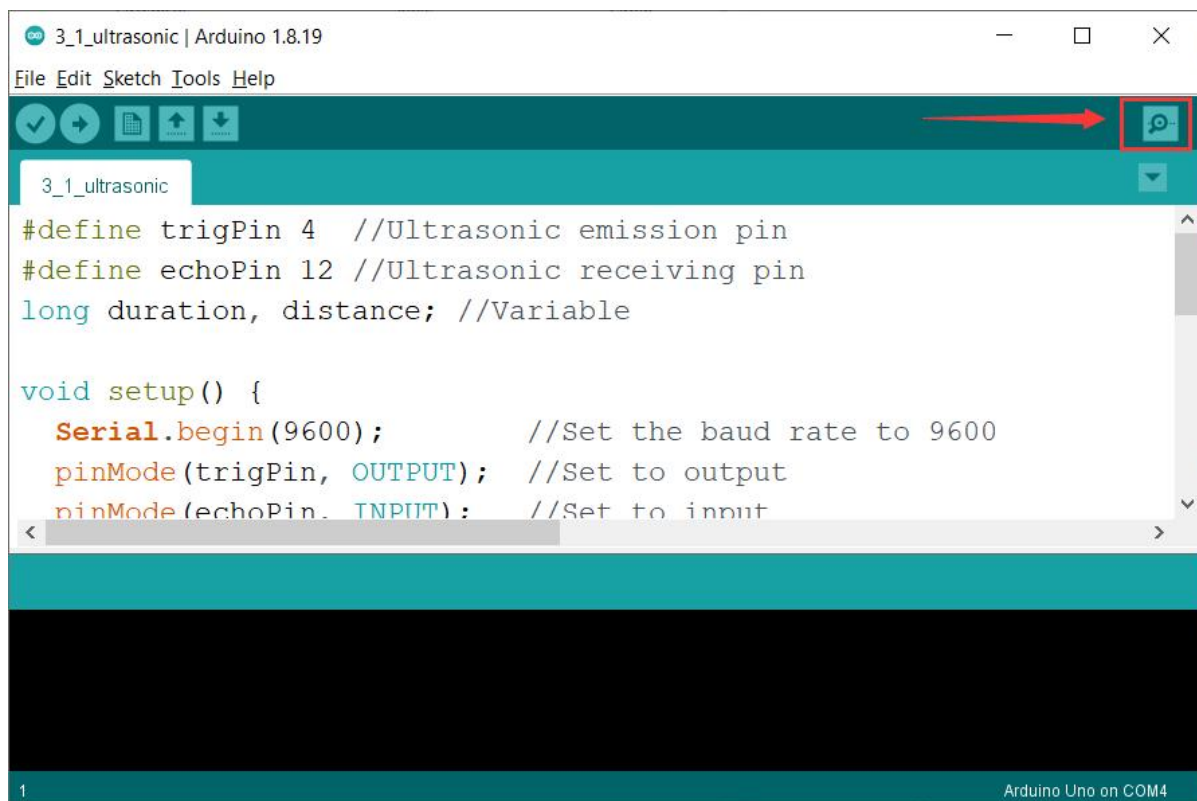
void loop() {
  //The ultrasonic sensor emits a signal
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  //The ultrasonic sensor receives the returned signal, and the time from emitting the signal
  to receiving the signal is assigned to duration
  duration = pulseIn(echoPin, HIGH);
  //Calculate the distance in centimeters, and the speed of sound is 0.034cm/ms
  distance = duration * 0.034 / 2;
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  delay(500);
}
```

## (2) Experiment phenomenon

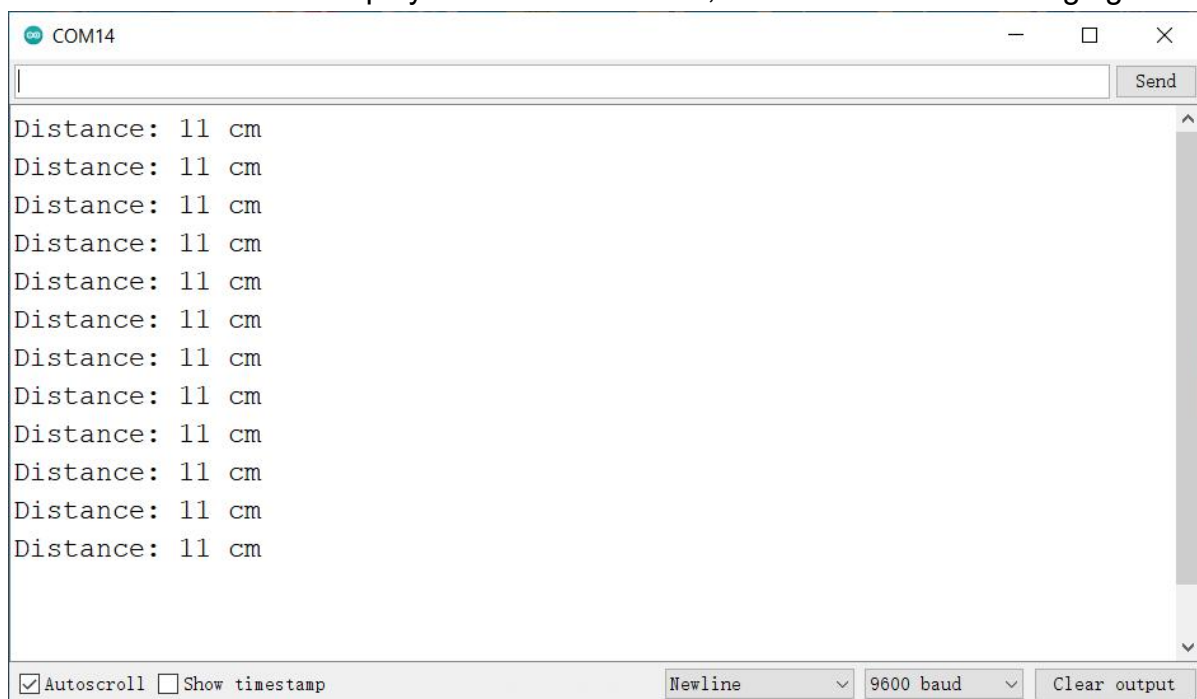
Open the serial port monitor, set the baud rate to 9600, and you can see the distance values detected by the ultrasonic sensor printed in the serial port monitor area.

Open the serial port monitor of the Arduino IDE, as shown in the following figure:





Set the baud rate and display the distance values, as shown in the following figure



**Tips:** A serial monitor is a very useful tool, commonly used to print sensor values and detect program error locations. It is important to be proficient in using a serial monitor.

### 3.2 Detect the presence of goods and carry them automatically

The rotation directions of each joint of the mechanical arm are shown in the table below:

Steering gear	When going to 0 degrees	When going to 180 degrees
Steering gear 1 (Base)	Turn to the far right	Turn to the far left
Steering gear 2 (Left)	Swing the rod connecting the steering gear 2 forward ( Cannot be lower than 90 degrees to avoid damaging the mechanical arm)	Swing the rod connecting the steering gear 2 backward ( Cannot be higher than 160 degrees to avoid damaging the mechanical arm)
Steering gear 3 (Right)	Swing the rod connecting the steering gear 3 backward ( Cannot be lower than 30 degrees to avoid damaging the mechanical arm)	Swing the rod connecting the steering gear 3 forward ( Cannot be higher than 170 degrees to avoid damaging the mechanical arm)
Steering gear 4 (Claw)	Mechanical claw opens, and fully opens at 0 degrees	Mechanical claw closes, and fully closes at 90 degrees

#### (1) Programming idea

1. The ultrasonic module detects if there is any goods in front of it.
2. If no goods is detected, the mechanical arm will not move.
3. If object is detected in front:  ①Judge the distance of the goods. If the distance is relatively close, the mechanical arm will transport the goods to the right.

②Judge the distance of the goods. If the distance is relatively far, the mechanical arm will transport the goods to the left.

## (2) Example code

```
#include <Servo.h>
Servo myservo1;
Servo myservo2;
Servo myservo3;
Servo myservo4;
int pos1=100, pos2=120, pos3=90, pos4=90; //Angle values at the initial state of the
mechanical arm

#define trigPin 4
#define echoPin 12
long duration, distance; //Variable

void setup()
{
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    //Initialize the mechanical arm
    myservo1.attach(7);
    myservo2.attach(10);
    myservo3.attach(9);
    myservo4.attach(8);
    //Initialize the angle value of the mechanical arm
    myservo1.write(pos1);
    myservo2.write(pos2);
    myservo3.write(pos3);
    myservo4.write(pos4);
    delay(1000);
}

void loop()
{
    int val = distance_value(); //Read the distance value detected by the ultrasonic module
    Serial.println(val);
    // If nearby goods is detected
```

```
if(val < 11) {
    pos4 = myservo4.read();
    for(; pos4>0; pos4--) {
        myservo4.write(pos4); //Open the mechanical claw
        delay(10); //Adjust the rotation speed of the steering gear
    }
    pos2 = myservo2.read();
    for(; pos2>95; pos2--) {
        myservo2.write(pos2); //Swing the rod connecting the left steering gear forward
        delay(10); //Adjust the rotation speed of the steering gear
    }
    pos3 = myservo3.read();
    for(;pos3<140;pos3++) {
        myservo3.write(pos3); //Swing the rod connecting the right steering gear forward
        delay(10); //Adjust the rotation speed of the steering gear
    }
    pos4 = myservo4.read();
    for(; pos4<40; pos4++) {
        myservo4.write(pos4); //Grab
        delay(10); //Adjust the rotation speed of the steering gear
    }
    pos3 = myservo3.read();
    for(;pos3>90;pos3--) {
        myservo3.write(pos3); //Lift
        delay(10); //Adjust the rotation speed of the steering gear
    }
    pos2 = myservo2.read();
    for(; pos2<120; pos2++) {
        myservo2.write(pos2); //Swing the rod connecting the left steering gear forward

        delay(10); //Adjust the rotation speed of the steering gear
    }
    pos1 = myservo1.read();
    for(;pos1>20;pos1--) {
        myservo1.write(pos1); //Turn to the right
        delay(1); //Adjust the rotation speed of the steering gear
    }
    pos3 = myservo3.read();
    for(;pos3<150;pos3++) {
        myservo3.write(pos3); //Swing the rod connecting the right steering gear forward
        delay(10); //Adjust the rotation speed of the steering gear
    }
}
```

```
pos4 = myservo4.read();
for(; pos4>0; pos4--) {
    myservo4.write(pos4); //Open the mechanical claw
    delay(10); //Adjust the rotation speed of the steering gear
}

pos3 = myservo3.read();
for(;pos3>90;pos3--) {
    myservo3.write(pos3); //Lift
    delay(10); //Adjust the rotation speed of the steering gear
}

pos1 = myservo1.read();
for(;pos1<100;pos1++) {
    myservo1.write(pos1); //Turn to the middle
    delay(1); //Adjust the rotation speed of the steering gear
}

}

//Detected goods that are relatively far away
if((val > 11) && (val < 23)) {
    pos4 = myservo4.read();
    for(; pos4>0; pos4--) {
        myservo4.write(pos4); //Open the mechanical claw
        delay(10); //Adjust the rotation speed of the steering gear
    }

    pos2 = myservo2.read();
    for(; pos2<160; pos2++) {
        myservo2.write(pos2); //Swing the rod connecting the left steering gear forward
        delay(10); //Adjust the rotation speed of the steering gear
    }

    pos3 = myservo3.read();
    for(;pos3<180;pos3++) {
        myservo3.write(pos3); //Swing the rod connecting the right steering gear forward
        delay(10); //Adjust the rotation speed of the steering gear
    }

    pos4 = myservo4.read();
    for(; pos4<42; pos4++) {
        myservo4.write(pos4); //Grab
        delay(10); //Adjust the rotation speed of the steering gear
    }

    pos3 = myservo3.read();
```

```

for(;pos3>90;pos3--) {
    myservo3.write(pos3); //Lift
    delay(10); //Adjust the rotation speed of the steering gear
}

pos2 = myservo2.read();
for(; pos2>120; pos2--) {
    myservo2.write(pos2); //Swing the rod connecting the left steering gear forward
    delay(10); //Adjust the rotation speed of the steering gear
}

pos1 = myservo1.read();
for(;pos1<150;pos1++) {
    myservo1.write(pos1); //Turn to the left
    delay(1); //Adjust the rotation speed of the steering gear
}

pos3 = myservo3.read();
for(;pos3<150;pos3++) {
    myservo3.write(pos3); //Swing the rod connecting the right steering gear forward
    delay(10); //Adjust the rotation speed of the steering gear
}

pos4 = myservo4.read();
for(; pos4>0; pos4--) {
    myservo4.write(pos4); //Open the mechanical claw
    delay(10); //Adjust the rotation speed of the steering gear
}

pos3 = myservo3.read();
for(;pos3>90;pos3--) {
    myservo3.write(pos3); //Lift
    delay(10); //Adjust the rotation speed of the steering gear
}

pos1 = myservo1.read();
for(;pos1>96;pos1--) {
    myservo1.write(pos1); //Turn to the middle
    delay(1); //Adjust the rotation speed of the steering gear
}
}

//Sub function, ultrasonic ranging
int distance_value() {
    //The ultrasonic sensor emits a signal
    digitalWrite(trigPin, HIGH);

```

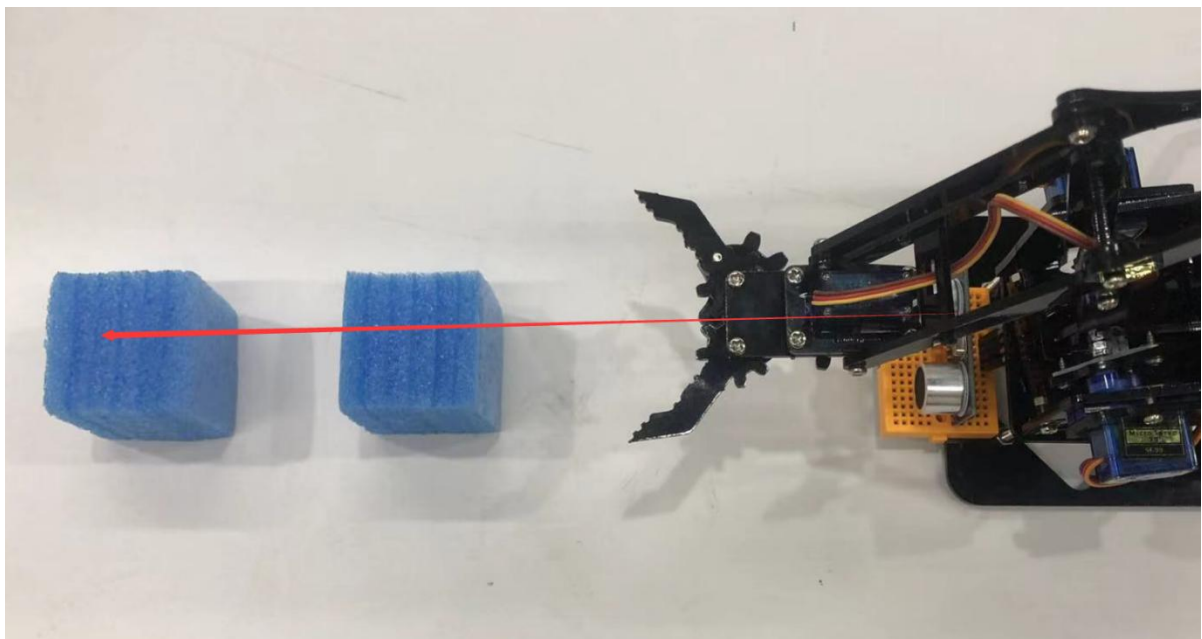


```
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
//The ultrasonic sensor receives the returned signal, and the time from emitting the signal
to receiving the signal is assigned to duration
duration = pulseIn(echoPin, HIGH);
//Calculate the distance in centimeters, and the speed of sound is 0.034cm/ms
distance = duration * 0.034 / 2;
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
delay(100);
return distance; //Return the detected distance value
}
```

### (3) Experiment phenomenon

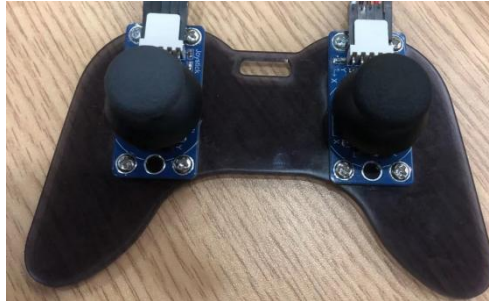
We have provided two pearl cotton blocks for the experiment.

Because ultrasonic sensor detects linear distance, we need to adjust the mechanical arm and pearl cotton blocks so that the mechanical claw, ultrasonic sensor, and pearl cotton blocks are all on the same line. If yours are not on the same line, please adjust the angle of the mechanical arm base yourself.



## 4. Rocker handle controls the mechanical arm

Previously, we achieved the automatic grabbing and handling of goods by the mechanical arm, all of which were automated. However, it would be more interesting if we could use our hands to control the mechanical arm, so we added a rocker handle that consists of two PS2 rocker modules.



**Rocker module:** The PS2 rocker module has three axes, namely the X, Y, and Z axes. The X and Y axes return analog values, while the Z axis is a button, returning digital values. The analog value range is 0~1024, and the key returns 0 or 1.

### 4.1 Read the values of the rocker handle

#### (1) Example code

```
/*
 * create by straysnail
 * 2023/3/11
 */

//Define the pin of the rocker module
#define x1 A0
#define y1 A1
#define b1 11
#define x2 A2
#define y2 A3
#define b2 12

void setup() {
    Serial.begin(9600);
    //Set pin mode to input
    pinMode(x1, INPUT);
    pinMode(y1, INPUT);
}
```

```
pinMode(b1, INPUT);
pinMode(x2, INPUT);
pinMode(y2, INPUT);
pinMode(b2, INPUT);

}

void loop() {
    // Read the value of the left rocker module
    int x1Val = analogRead(x1); //Read the analog value of the X axis
    int y1Val = analogRead(y1); //Read the analog value of the Y axis
    int b1Val = digitalRead(b1); //Read the digital value of the Z axis
    // Read the value of the right rocker module
    int x2Val = analogRead(x2);
    int y2Val = analogRead(y2);
    int b2Val = digitalRead(b2);
    Serial.print("x1 = ");
    Serial.print(x1Val);
    Serial.print(" ");
    Serial.print("y1 = ");
    Serial.print(y1Val);
    Serial.print(" ");
    Serial.print("b1 = ");
    Serial.print(b1Val);
    Serial.print(" ");
    Serial.print("x2 = ");
    Serial.print(x2Val);
    Serial.print(" ");
    Serial.print("y2 = ");
    Serial.print(y2Val);
    Serial.print(" ");
    Serial.print("b2 = ");
    Serial.println(b2Val);
    delay(100);
}
```

## (2) Experiment phenomenon

Open the serial port monitor and you can see the printed values of the handle. By manipulating the handle, you can see the changes in the printed values.



### (1) Programming idea

When the rocker is moved to the right, the mechanical claw gradually closes.

```
#include <Servo.h> //Import the steering gear library files
Servo myservo4;    //Create a steering gear instance to control

#define right_Y A3  //Define the pin of the Y-axis of rocker 2
int pos4 = 90;     //Variable, used to control the steering gear angle of the mechanical claw
int y2 = 0;

void setup() {
  Serial.begin(9600);
```

```
pinMode(right_Y, INPUT);
myservo4.attach(8);
myservo4.write(pos4); //The mechanical claw closes
}

void loop() {
  y2 = analogRead(right_Y); //Read the value of the rocker
  // Serial.print("y2 = ");
  // Serial.println(y2Val);

  if(y2<50) //If the right rocker is moved to the right
  {
    pos4=pos4+1; //The current angle value of the steering gear 4, self adding 1
    if(pos4>90) //At 90 degrees, the mechanical claw is fully closed
    {
      // (Change the value according to the actual situation)
      pos4=90;
    }
    Serial.println(pos4);
    myservo4.write(pos4); //The steering gear 4 executes the action, and the claw gradually
    closes
    delay(8); //Adjust the rotation speed of the steering gear
  }
  if(y2>1000) //If the right rocker is moved to the left
  {
    pos4=pos4-1; //The current angle value of the steering gear 4
    if(pos4 < 0) //At 0 degrees, it is the maximum angle
    {
      pos4=0;
    }
    Serial.println(pos4);
    myservo4.write(pos4); //The steering gear 4 executes the action, and the claw gradually
    opens
    delay(8); //Adjust the rotation speed of the steering gear
  }
}
```

### (3) Experiment operation and phenomenon

When the right rocker is moved to the right, the steering gear gradually turns to 0 degrees and the mechanical claw opens.

When the right rocker is moved to the left, the steering gear gradually turns to 90

degrees and the mechanical claw closes.

### 4.3 Rocker controls the 4-axis mechanical arm

Steering gear	When going to 0 degrees	When going to 180 degrees
Steering gear 1 (base)	Turn to the far right	Turn to the far left
Steering gear 2 (left)	Swing the rod connecting the steering gear 2 forward (cannot be lower than 90 degrees to avoid damaging the mechanical arm)	Swing the rod connecting the steering gear 2 backward (cannot be higher than 160 degrees to avoid damaging the mechanical arm)
Steering gear 3 (right)	Swing the rod connecting the steering gear 3 backward (cannot be lower than 30 degrees)	Swing the rod connecting the steering gear 3 forward (cannot be higher than 170 degrees)
Steering gear 4 (claw)	When the steering gear is at 0 degrees, the mechanical claw is fully open	When the steering gear is at 90 degrees, the mechanical claw is fully closed

#### (1) Programming idea

Definition of the rocker module:

Left rocker module: X-axis:  $x_1$  , Y-axis:  $y_1$  , Z-axis:  $z_1$

Right rocker module: X-axis:  $x_2$  , Y-axis:  $y_2$  , Z-axis:  $z_2$

$y_1$  controls the steering gear of the base

When the left rocker is moved to the left, the base of the mechanical arm gradually rotates to the left.

When the left rocker is moved to the right, the base of the mechanical arm



gradually rotates to the right.

x1 controls the left steering gear

When the left rocker is moved to the front, The rod on the left side of the mechanical arm swings forward.

When the left rocker is moved to the rear, The rod on the left side of the mechanical arm swings backward.

z1 button, click to disconnect all the steering gears of the mechanical arm

y2 controls the claw to open or close

When the right rocker is moved to the left, The mechanical claw gradually opens.

When the right rocker is moved to the right, The mechanical claw gradually closes.

x2 controls the right steering gear

When the right rocker is moved to the front, The rod on the right side of the mechanical arm swings forward.

When the right rocker is moved to the rear, The rod on the right side of the mechanical arm swings backward.

z2 button, press and hold to reconnect all the steering gears of the mechanical arm.

## (2) Example code

```
#include <Servo.h> //Add the steering gear library files
Servo myservo1; // Create a steering gear instance to control
Servo myservo2;
Servo myservo3;
Servo myservo4;

int pos1=90, pos2=130, pos3=90, pos4=90; //Define variables for four steering gear angles
and assign initial values (i.e. the attitude angle values at startup)
```

```
const int left_X = A0; //Define the pin of the left X-axis as A0
const int left_Y = A1; //Define the pin of the left Y-axis as A1
const int left_key = 11; //Define the pin of the right button as 7 (i.e. the value of Z-axis)

const int right_X = A2; //Define the pin of the right X-axis as A2
const int right_Y = A3; //Define the pin of the right Y-axis as A3
const int right_key = 12; //Define the pin of the right button as 8 (i.e. the value of Z-axis)

int x1,y1,z1; //Define variables to store the read rocker values
int x2,y2,z2;
boolean flag = 0;
boolean flag2 = 0;
int count = 0;
int i = 0;

void setup()
{
  myservo1.attach(7); //Set the control pin of the steering gear 1 as A1
  myservo2.attach(10); //Set the control pin of the steering gear 2 as A0
  myservo3.attach(9); //Set the control pin of the steering gear 3 as D8
  myservo4.attach(8); //Set the control pin of the steering gear 4 as D9
  //Status at startup
  myservo1.write(pos1);
  myservo2.write(pos2);
  myservo3.write(pos3);
  myservo4.write(pos4);
  delay(1500);

  pinMode(right_key, INPUT); //Set the left and right buttons to input
  pinMode(left_key, INPUT);
  Serial.begin(9600); // Set the baud rate as 9600
}

void loop()
{
  x1 = analogRead(left_X); //Read the value of the left X-axis
  y1 = analogRead(left_Y); //Read the value of the left Y-axis
  z1 = digitalRead(left_key); //Read the value of the left Z-axis

  x2 = analogRead(right_X); //Read the value of the right X-axis
  y2 = analogRead(right_Y); //Read the value of the right Y-axis
```

```
z2 = digitalRead(right_key); //Read the value of the right Z-axis
//delay(5); //Reduce the speed of all the steering gears

//The claw
claw();
//Rotates
base();
//The right steering gear
right_ser();
//The left steering gear
left_ser();
//Add the function of button control to release the steering gear to avoid damage caused
by prolonged operation
if(z1 == 1) //If the left rocker button is pressed
{
    flag = 1;
    flag2 = 1;
    while(flag == 1)
    {
        release_servo(); //Release the steering gear to avoid damage caused by prolonged
operation
        z2 = digitalRead(right_key); //Read the value of the right Z-axis
        if(z2 == 1) { //If the right rocker button is pressed
            delay(10); //Eliminate button jitter and reduce errors
            if(z2 == 1) {
                while(flag2 == 1) {
                    i++;
                    delay(5);
                    z2 = digitalRead(right_key); //Read the value of the right Z-axis
                    if(z2 == 0) { //The right button is released
                        flag2 = 0;
                    }
                    Serial.println(i);
                }
            }
            if(i > 100) { //If the right button is long pressed
                flag = 0; //Exit the circle
                i = 0;
                myservo1.attach(7); //Connect the steering gears
                myservo2.attach(10);
                myservo3.attach(9);
                myservo4.attach(8);
            } else {
```

```
        flag2 = 1; //Keep the circle
        flag = 1; //Keep the circle
        i = 0;
    }

}

}

}

}

}

//The claw
void claw()
{
    //The claw
    if(y2<50) //If the right rocker is moved to the right
    {
        pos4=pos4+1; //The current angle value of the steering gear 4, self adding 1
        if(pos4>90) //At 90 degrees, the mechanical claw is fully closed
        {
            // (Change the value based on the actual situation)
            pos4=90;
        }
        Serial.println(pos4);
        myservo4.write(pos4); //The steering gear 4 executes the action, and the claw gradually
        closes
        delay(8); //Adjust the speed of the rotation of the steering gear
    }
    if(y2>1000) //If the right rocker is moved to the left
    {
        pos4=pos4-1; //The current angle value of the steering gear 4
        if(pos4 < 0) //At 0 degrees, it is the maximum angle
        {
            pos4=0;
        }
        Serial.println(pos4);
        myservo4.write(pos4); //The steering gear 4 executes the action, and the claw gradually
        opens
        delay(8); //Adjust the speed of the rotation of the steering gear
    }
}

//Rotate
```

```
void base()
{
    if(y1<50) //If the left rocker is moved to the right
    {
        pos1=pos1-1; //Pos1 subtracts 1 from itself
        if(pos1<1) //Limit the angle of right turn
        {
            pos1=1;
        }
        myservo1.write(pos1); //The steering gear 1 executes the action, and the mechanical arm
rotates to the right
        delay(5); //Adjust the speed of the rotation of the steering gear
    }
    if(y1>1000) //If the left rocker is moved to the left
    {
        pos1=pos1+1; //Pos1 adds 1 from itself
        if(pos1>180) //Limit the angle of left turn
        {
            pos1=180;
        }
        myservo1.write(pos1); //The mechanical turns left
        delay(5); //Adjust the speed of the rotation of the steering gear
    }
}

//Right steering gear
void right_ser()
{
    if(x2<50) //If the right rocker is moved to the front
    {
        pos3=pos3+1;
        if(pos3>160) //Limit the angle
        {
            pos3=160;
        }
        myservo3.write(pos3); //Swing the right steering gear rod forward
        delay(10); //Adjust the speed of the rotation of the steering gear
    }
    if(x2>1000) //If the right rocker is moved to the rear
    {
        pos3=pos3-1;
        if(pos3<60) //Limit the angle of descent
```

```
{
    pos3=60;
}

myservo3.write(pos3); //Swing the right steering gear rod backward
delay(10); //Adjust the speed of the rotation of the steering gear
}
}

//Left steering gear
void left_ser()
{
    if(x1<50) //If the left rocker is moved to the front
    {
        pos2=pos2-1;
        if(pos2<90) //Limit the angle of forward swing
        {
            pos2=90;
        }
        myservo2.write(pos2); //Swing the left steering gear rod forward
        delay(10); //Adjust the speed of the rotation of the steering gear
    }
    if(x1>1000) //If the left rocker is moved to the rear
    {
        pos2=pos2+1;
        if(pos2>170) //Limit the angle of backward swing
        {
            pos2=170;
        }
        myservo2.write(pos2); //Swing the left steering gear rod backward
        delay(10); //Adjust the speed of the rotation of the steering gear
    }
}

//Release the steering gears
void release_servo() {
    delay(500);
    myservo1.detach(); //Release the steering gears
    myservo2.detach();
    myservo3.detach();
    myservo4.detach();
}
```



### (3) Experiment operation and phenomenon

Definition of the rocker module:

Left rocker module: X-axis:  $x_1$  , Y-axis:  $y_1$  , Z-axis:  $z_1$

Right rocker module: X-axis:  $x_2$  , Y-axis:  $y_2$  , Z-axis:  $z_2$

$y_1$  controls the steering gear of the base

When the left rocker is moved to the left, the base of the mechanical arm gradually rotates to the left.

When the left rocker is moved to the right, the base of the mechanical arm gradually rotates to the right.

$x_1$  controls the left steering gear of the mechanical arm

When the left rocker is moved to the front, The rod on the left side of the mechanical arm swings forward.

When the left rocker is moved to the rear, The rod on the left side of the mechanical arm swings backward.

$z_1$  button, click to disconnect all the steering gears of the mechanical arm.

$y_2$  controls the mechanical claw to open or close

When the right rocker is moved to the left, the mechanical claw gradually opens.

When the right rocker is moved to the right, the mechanical claw gradually closes.

$x_2$  controls the right steering gear of the mechanical arm

When the right rocker is moved to the front, The rod on the right side of the mechanical arm swings forward.

When the right rocker is moved to the rear, The rod on the right side of the mechanical arm swings backward.

z2 button, Press and hold to reconnect all the steering gears of the mechanical arm.

Click the button on the left rocker to release the steering gears, which means that the steering gears no longer maintain the set angles to avoid heating and damage caused by prolonged operation. Long press the button on the right rocker to re-enter the rocker control function.

#### 4.4 Rocker teaching function of the mechanical arm 1

What is the teaching function of a mechanical arm? It is to control the mechanical arm, remember each action of it, and then have the mechanical arm repeatedly execute the remembered action.

The teaching function of the mechanical arm is also a common function of industrial mechanical arms. Because most of the mechanical arms work in the factory requires debugging and teaching.

The program for the final function of the mechanical arm is quite complex, and we will implement it step by step, first realizing to remember an action.

##### (1) Programming idea

Short press the right rocker button to read the angle values of the four steering gears and save them.

Long press the right rocker button, and the mechanical arm will operate to the saved action.

Click the left rocker button to disconnect all the steering gears.

Long press the right rocker button to reconnect all the steering gears.

##### (2) Example code

The program is quite long, please open the example program to view.

4\_4\_Teaching\_function1

2023/3/13 10:24

##### (3) Experiment phenomenon

The rocker controls the movement of the mechanical arm. Short press the right

button to read the angle values of all the steering gears and save them. The rocker controls the operation of the mechanical arm, and then long press the right rocker button to automatically move the mechanical arm to the previously saved posture. Click the left rocker button to disconnect all the steering gears. Long press the right rocker button to reconnect all the steering gears.

## 4.5 Rocker teaching function of the mechanical arm 2

The teaching function is further improved to save multiple postures of the mechanical arm, which can then be executed once.

### (1) Programming idea

By defining four arrays to store the angle values of each of the four steering gears, the posture can be saved. Then the steering gears execute the angle values stored in the array one by one.

### (2) Example code

The program is quite long, please open the example program to view.

4\_5\_Teaching\_function2 2023/3/13 10:41

### (3) Experiment phenomenon

The rocker controls the movement of the mechanical arm. Short press the right button to save the first pose of the mechanical arm. The rocker controls the mechanical arm to move to another pose. Short press the right button to save the second pose of the mechanical arm. The same operation can save up to 11 poses. Then press and hold the right rocker button, and the mechanical arm will automatically operate all the previously saved postures. Click the left rocker button to disconnect all the steering gears. Long press the right rocker button to reconnect all the steering gears.

## 4.6 Rocker teaching function of the mechanical arm 3

The complete version of the mechanical arm teaching function can save multiple postures of the mechanical arm and cycle through the saved postures.

### (1) Programming idea

Define 4 arrays to store the angle values of each of the 4 steering gears.
Short press the right rocker button to save the mechanical arm posture. You can save 11 postures.

Long press the right rocker button, and the mechanical arm will cycle through the saved posture.
In the mechanical arm cycle execution state, press and hold the left rocker button to exit the mechanical arm cycle.
Click the left rocker button to disconnect all the steering gears.
Long press the right rocker button to reconnect all the steering gears.

## (2) Example code

The program is quite long, please open the example program to view.




## (3) Experiment phenomenon

The rocker controls the mechanical arm.
Short press the right rocker button to save the mechanical arm posture. You can save 11 postures.
Long press the right rocker button, and the mechanical arm will cycle through the saved posture.
In the mechanical arm cycle execution state, press and hold the left rocker button to exit the mechanical arm cycle.
Click the left rocker button to disconnect all the steering gears.
Long press the right rocker button to reconnect all the steering gears.

## 5. APP controls the mechanical arm

**Android APP** download: Search **straysnail** in Google play

Or directly use the installation package we provide:

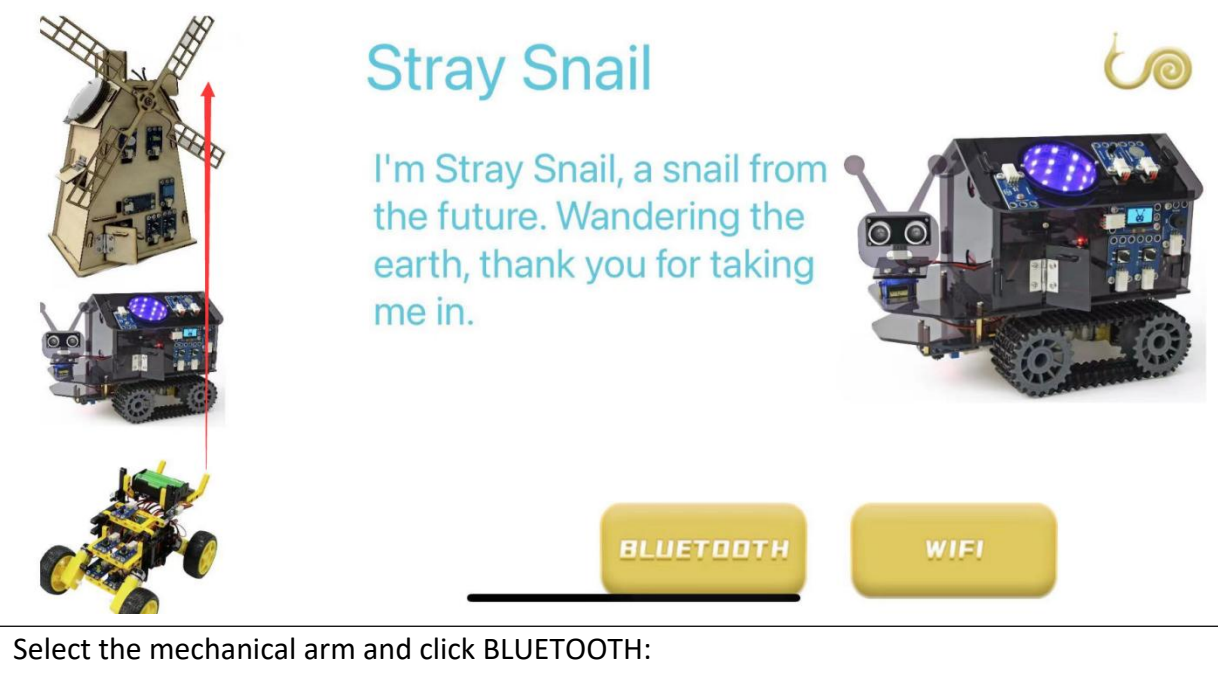
Mechanical arm > 4.Android APP	▼	↺	🔍 Search 4.Android APP
Name	Date modified	Type	
 straysnail.apk	5/7/2023 8:12 PM	APK File	




**Apple APP** download: Search **straysnail** in APP store

### APP interface introduction:

Drag the product diagram on the left side of the APP interface upward to see the diagram of the mechanical arm. Click on the mechanical arm diagram to select the mechanical arm, and then click the Bluetooth button.



Drag the product diagrams upward:







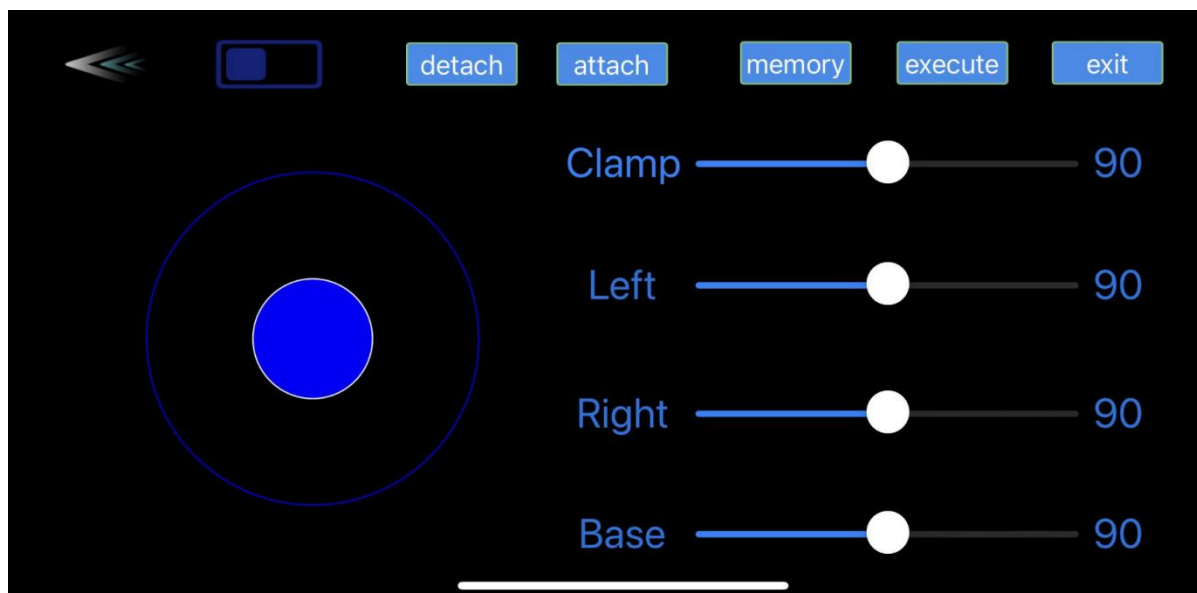
## Robot arm

This is a 4 degree of freedom manipulator. There are automatic handling, remote sensing control, demonstration education function, iOS and Android APP control





Interface of controlling the mechanical arm:



## 5.1 Download and read the values of APP

The Bluetooth app controls the mechanical arm by sending character or numerical commands to the Bluetooth module of the mechanical arm. So what characters are the buttons on our app? In order to visually see what characters are received, we print the received characters on the serial port monitor.



**Note:** Before uploading the code, it is necessary to turn off the Bluetooth switch, as both Bluetooth and USB use the Arduino UNO serial port 0 and 1, which can cause the upload of the code to fail.

### (1) Example code

```
char bleStr;

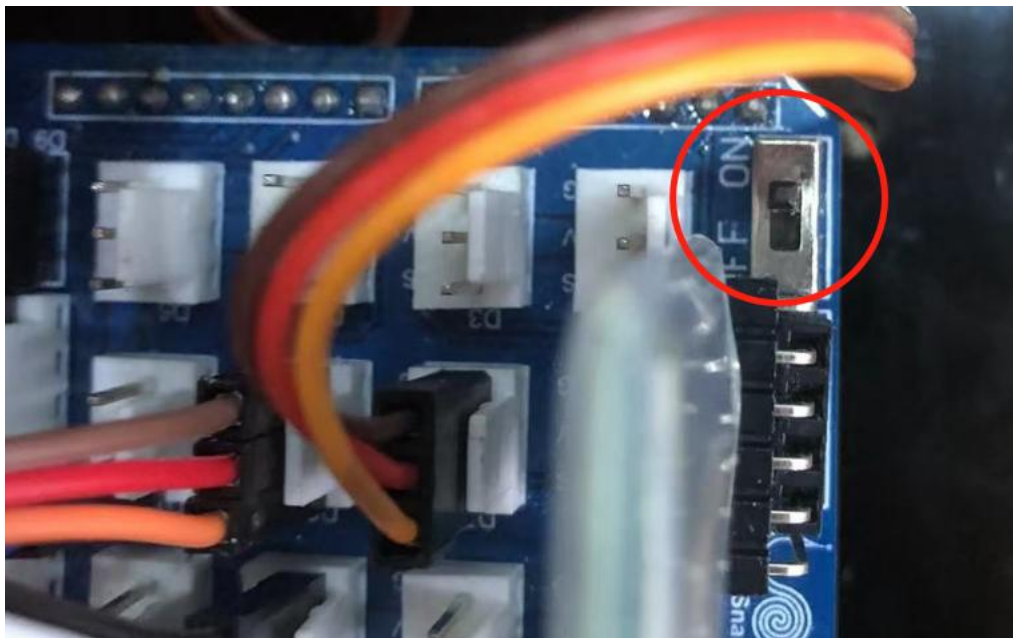
void setup() {
    Serial.begin(9600);
}

void loop() {
    while(Serial.available() > 0) //Judge whether the serial port area has
received a value
    {
        bleStr = Serial.read(); //Read the value of the serial port area
        Serial.println(bleStr);
    }
}
```

### (2) Experiment phenomenon

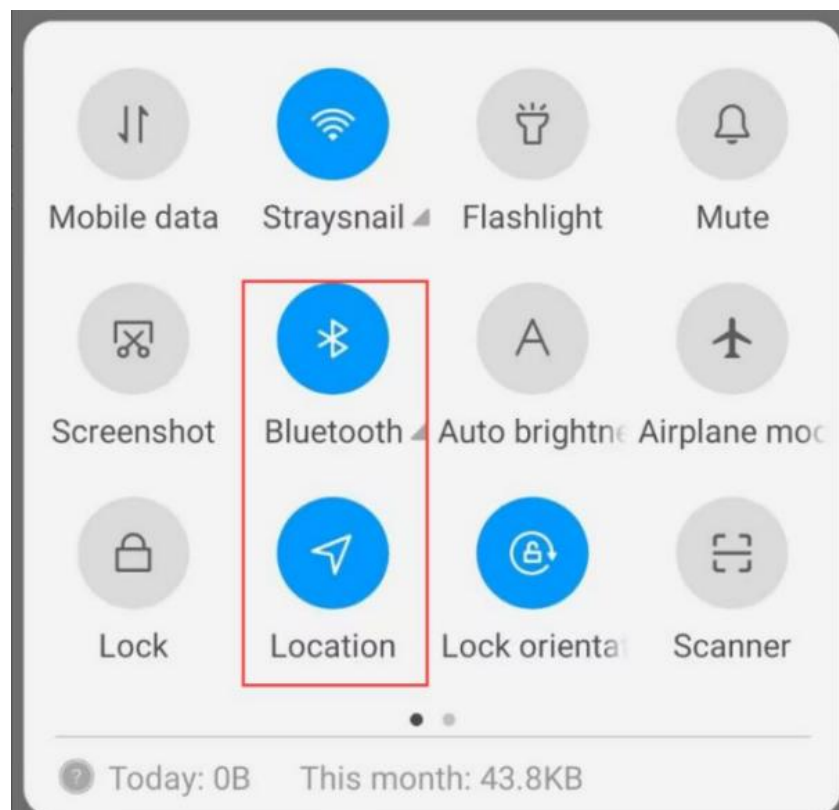
Open the serial port monitor, turn on the Bluetooth switch of the mechanical arm (that is, pull the switch on the right side of the expansion board), open the mobile APP, connect to Bluetooth, click the button on the interface, and you can see that the serial port monitor prints the received characters.

1. After burning the code, turn on the Bluetooth switch, which means turning the Bluetooth toggle switch to ON, as shown in the following figure.



2. Open the APP for Bluetooth search and connection

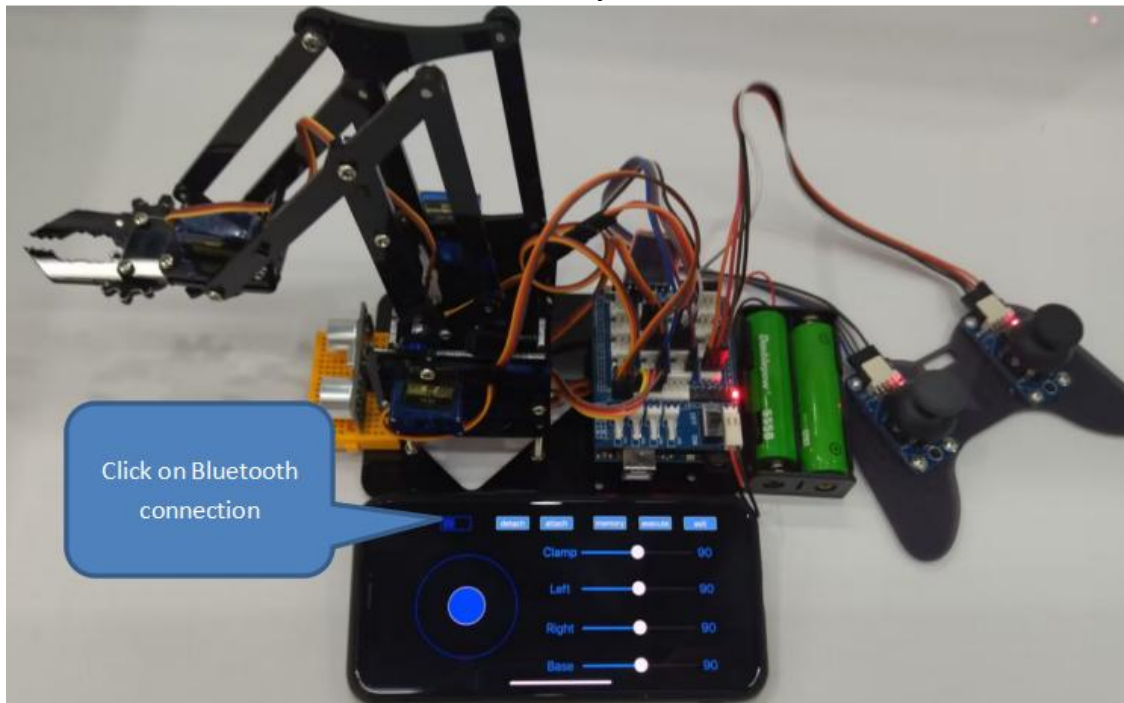
**Android phones** require **Bluetooth** and **location information** to be turned on, as shown in the following figure. **Apple phones** just need to turn on **Bluetooth**.



## Two ways to connect Bluetooth:

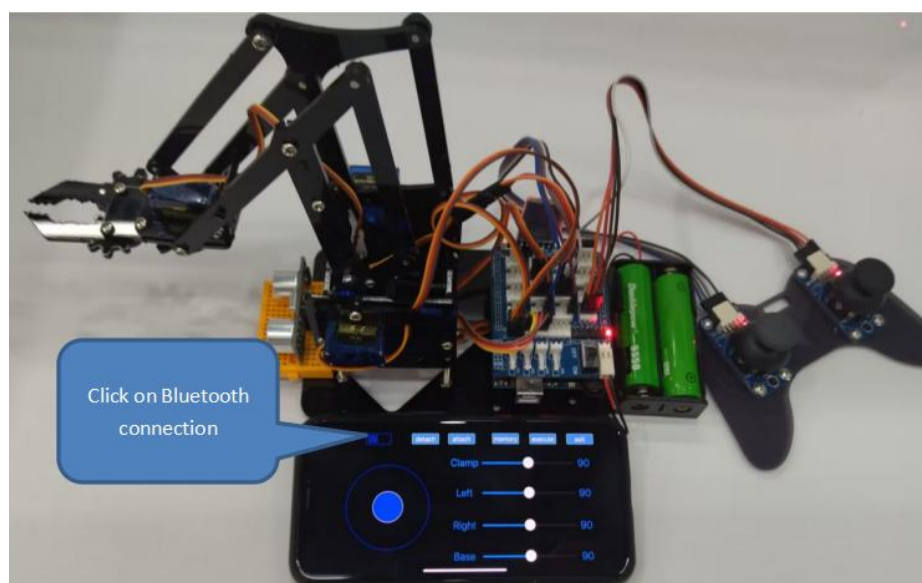
### (1) Close to connect

Open the APP on your phone, approach the mechanical arm, and then click the Bluetooth button. Bluetooth can automatically connect.

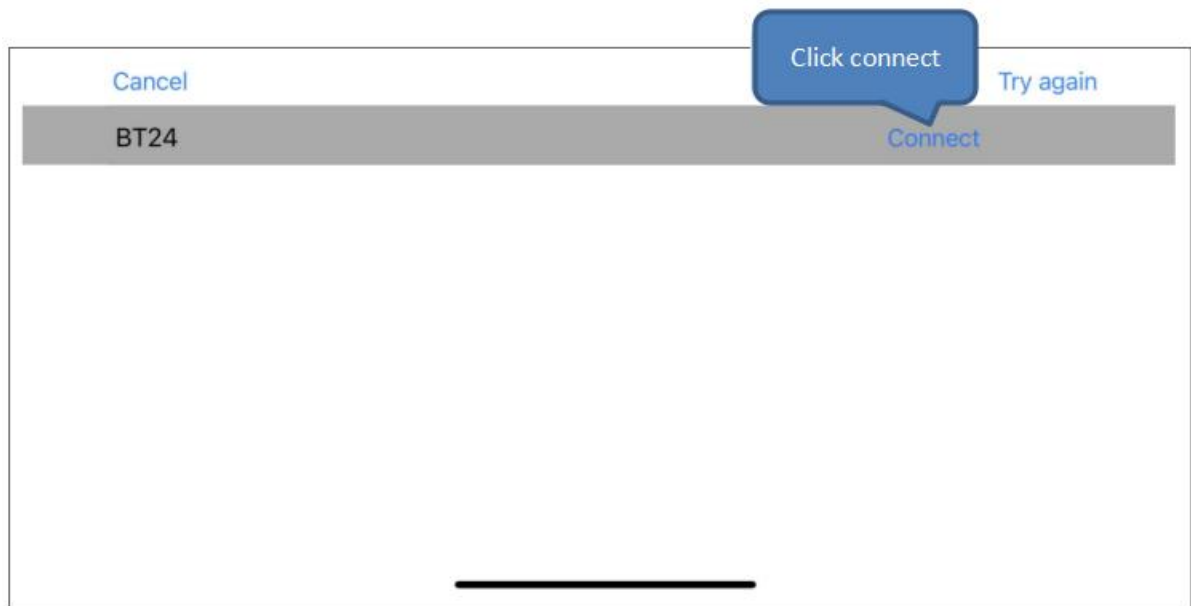


### (2) Manual connection

Open the APP interface and click the Bluetooth button, as shown in the following figure.

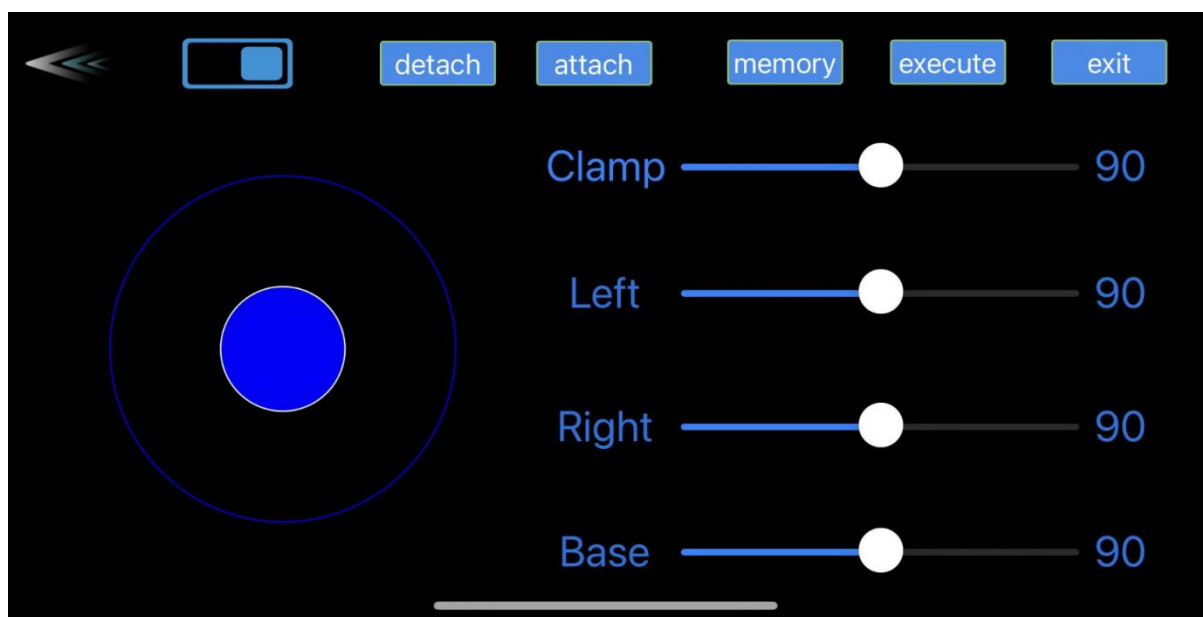


The Bluetooth name is BT24 or JDY-23.



### (3) Send a command

After successful connection, you can click the button on the interface and drag the drag bar to print the received value in the Arduino IDE serial port monitor.



Open the serial port monitor of the Arduino IDE to see the received Bluetooth button values.



## 5.2 Mobile APP virtual rocker controls the rotation of the base

In the previous section, we received the values from the Bluetooth APP, and we can see that the values printed on the serial port have the start character \$ and the end character \*. Now we need to filter out the start and end characters and take the middle command value.

### (1) Example code

```
#include <Servo.h> //Add the steering gear library files
Servo myservo1; // Create steering gear instances to control
Servo myservo2;
Servo myservo3;
Servo myservo4;

int pos1=90, pos2=130, pos3=90, pos4=90; // Define variables for 4 steering
gear angles and assign initial values (i.e. the attitude angle value at startup)

char bleVal;

void setup() {
  Serial.begin(9600);
  myservo1.attach(7); //Set the control pin of steering gear 1 as A1
  myservo2.attach(10); //Set the control pin of steering gear 2 as A0
```

```
myservo3.attach(9); //Set the control pin of steering gear 3 as D8
myservo4.attach(8); //Set the control pin of steering gear 4 as D9
//Posture at startup
myservo1.write(pos1);
myservo2.write(pos2);
myservo3.write(pos3);
myservo4.write(pos4);
delay(1000);

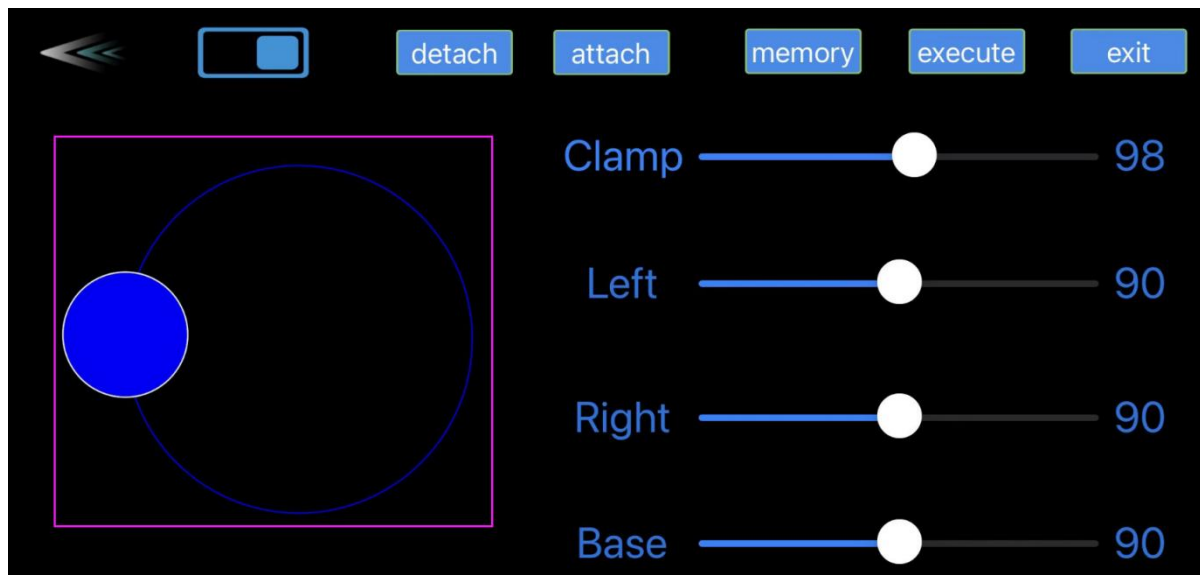
}

void loop() {
  if(Serial.available() > 0)
  {
    //Read the value of the serial port cache until encountering *, which means
    that all characters except for * are assigned to bleVal1
    String bleVal1 = Serial.readStringUntil('*');
    if(bleVal1[0] == '$') //Judge if the starting value is $
    {
      bleVal = bleVal1[1]; //Assign valid command values to bleVal
      Serial.println(bleVal);
    }
  }
  //When the virtual rocker is dragged to the left, the command value sent is
  L
  if(bleVal == 'L') {
    pos1=pos1+1; //Pos1 self adding 1
    if(pos1>180) //Limit the angle of left turn
    {
      pos1=180;
    }
    myservo1.write(pos1); //The mechanical arm turns left
    delay(5); //Adjust the rotation speed of the steering gear
  }
  //When the virtual rocker is dragged to the right, the command value sent is
  R
  if(bleVal == 'R') {
    pos1=pos1-1; //Pos1 subtracts 1 from itself
    if(pos1<1) //Limit the angle of right turn
    {
      pos1=1;
    }
  }
}
```

```
myservo1.write(pos1); //Steering gear 1 executes the action, and the
mechanical arm rotates to the right
    delay(5); //Adjust the rotation speed of the steering gear
}
}
```

## (2) Experiment phenomenon

Turn on the Bluetooth switch, and after successfully connecting the mobile APP to Bluetooth, drag the rocker to the left to turn the mechanical arm to the left. Drag the rocker to the right to turn the mechanical arm to the right.



## 5.3 Mobile APP drag bar real-time controls the mechanical claw

In addition to providing virtual rocker control, we also created a drag bar for real-time control, which means that when the drag bar is dragged, the mechanical arm immediately moves with it.

We added the function of dragging bars to control the mechanical claw based on the program from the previous lesson.

### (1) Example code

```
#include <Servo.h> //Add the steering gear library files
Servo myservo1; // create steering gear instances to control
Servo myservo2;
```



```
Servo myservo3;
Servo myservo4;

int pos1=90, pos2=130, pos3=90, pos4=90; // Define variables for 4 steering
gear angles and assign initial values (i.e. the attitude angle value at startup)

char bleVal;

void setup() {
  Serial.begin(9600);
  myservo1.attach(7); //Set the control pin of steering gear 1
  myservo2.attach(10);
  myservo3.attach(9);
  myservo4.attach(8);
  //Posture at startup
  myservo1.write(pos1);
  myservo2.write(pos2);
  myservo3.write(pos3);
  myservo4.write(pos4);
  delay(1000);
}

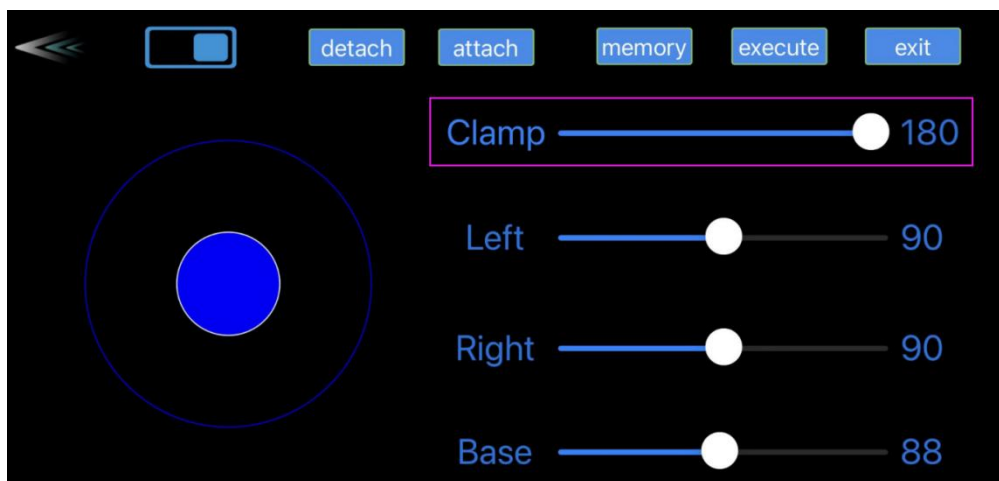
void loop() {
  if(Serial.available() > 0)
  {
    String bleVal1 = Serial.readStringUntil('*');
    if(bleVal1[0] == '$') //If the first character received is '$'
    {
      bleVal = bleVal1[1];
      //Serial.println(bleVal);
      //If the second character received is 'c'
      if(bleVal1[1] == 'c') {
        //Convert the second to last character of a string to an integer and assign
it to pos4
        pos4 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos4 = map(pos4, 0, 180, 90, 0); //Mapping function, converting values
from 0~180 to 90~0
        myservo4.write(pos4); //Control the steering gear execution of the
mechanical claw
        //Serial.println(pos4);
      }
    }
  }
}
```

```
    }  
  }  
}  
if(bleVal == 'L') {  
  pos1=pos1+1; //pos1 self adding 1  
  if(pos1>180) //Limit the angle of left turn  
  {  
    pos1=180;  
  }  
  myservo1.write(pos1); //The mechanical arm turns left  
  delay(5); //Adjust the rotation speed of the steering gear  
}  
if(bleVal == 'R') {  
  pos1=pos1-1; //pos1 subtracts 1 from itself  
  if(pos1<1) //Limit the angle of right turn  
  {  
    pos1=1;  
  }  
  myservo1.write(pos1); //Steering gear 1 executes the action, and the  
  mechanical arm rotates to the right  
  delay(5); //Adjust the rotation speed of the steering gear  
}  
}
```

## (2) Experiment phenomenon

When uploading the program, remember to turn off the Bluetooth switch first.

Turn on the Bluetooth switch, drag the Clamp drag bar on the APP interface, and you can see that the mechanical claw follows to move.



## 5.4 Mobile APP controls the mechanical arm 1

Implement virtual rocker control of mechanical arm and drag bar control of mechanical arm in APP interface

### (1) Example code

```
#include <Servo.h> //Add the steering gear library files
Servo myservo1; // create steering gear instances to control
Servo myservo2;
Servo myservo3;
Servo myservo4;

int pos1=90, pos2=130, pos3=90, pos4=90; // Define variables for 4 steering
gear angles and assign initial values (i.e. the attitude angle value at startup)

char bleVal;

void setup() {
  Serial.begin(9600);
  myservo1.attach(7); //Set the control pin of steering gear 1 as A1
  myservo2.attach(10); //Set the control pin of steering gear 2 as A0
  myservo3.attach(9); //Set the control pin of steering gear 3 as D8
  myservo4.attach(8); //Set the control pin of steering gear 4 as D9
  //Posture at startup
  myservo1.write(pos1);
  myservo2.write(pos2);
  myservo3.write(pos3);
  myservo4.write(pos4);
  delay(1000);
}
```

```
}

void loop() {
  if(Serial.available() > 0)
  {
    String bleVal1 = Serial.readStringUntil('*');
    if(bleVal1[0] == '$') //If the first character received is '$'
    {
      bleVal = bleVal1[1];
      //Serial.println(bleVal);
      //If the second character received is 'c'
      if(bleVal1[1] == 'c') { //Control the steering gear of the claw
        //Convert the second to last character of a string to an integer and assign
        it to pos4
        pos4 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos4 = map(pos4, 0, 180, 90, 0); //Mapping function, converting values
        from 0~180 to 90~0. When the drag value increases, the claw opens
        myservo4.write(pos4);
        Serial.println(pos4);
      }
      if(bleVal1[1] == 'l') { //Control the left steering gear
        pos2 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos2 = map(pos2, 0, 180, 160, 90); //Mapping function, converting values
        from 0~180 to 160~90
        myservo2.write(pos2);
        Serial.println(pos2);
      }
      if(bleVal1[1] == 'r') { //Control the right steering gear
        pos3 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos3 = map(pos3, 0, 180, 20, 160); //Mapping function, converting values
        from 0~180 to 0~90
        myservo3.write(pos3);
        Serial.println(pos3);
      }
      if(bleVal1[1] == 'b') { //Control the right steering gear
        pos1 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos1 = map(pos1, 0, 180, 180, 0); //Mapping function. When the drag value
```

```
increases, the mechanical arm turns right
    myservo1.write(pos1);
    Serial.println(pos1);
}
}
}
if(bleVal == 'L') {
    pos1=pos1+1; //pos1 self adding 1
    if(pos1>180) //Limit the angle of left turn
    {
        pos1=180;
    }
    myservo1.write(pos1); //The mechanical arm turns left
    delay(5); //Adjust the rotation speed of the steering gear
}
if(bleVal == 'R') {
    pos1=pos1-1; //pos1 subtracts 1 from itself
    if(pos1<1) //Limit the angle of right turn
    {
        pos1=1;
    }
    myservo1.write(pos1); //The steering gear 1 executes action, and the
mechanical arm turns right
    delay(5); //Adjust the rotation speed of the steering gear
}
if(bleVal == 'Q') {
    pos2=pos2-1;
    if(pos2<90) //Limit the angle of swinging forward
    {
        pos2=90;
    }
    myservo2.write(pos2); //The rod of left steering gear swings forward
    delay(10); //Adjust the rotation speed of the steering gear
}
if(bleVal == 'Z') {
    pos2=pos2+1;
    if(pos2>170) //Limit the angle of swinging backward
    {
        pos2=170;
    }
    myservo2.write(pos2); //The rod of left steering gear swings backward
    delay(10); //Adjust the rotation speed of the steering gear
}
```

```
}  
  
if(bleVal == 'E') {  
    pos3=pos3+1;  
    if(pos3>180) //Limit the angle  
    {  
        pos3=180;  
    }  
    myservo3.write(pos3); //The rod of right steering gear swings forward  
    delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'C') {  
    pos3=pos3-1;  
    if(pos3<60) //Limit the angle of descent  
    {  
        pos3=60;  
    }  
    myservo3.write(pos3); //The rod of right steering gear swings backward  
    delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'F') {  
    pos3=pos3+1;  
    pos2=pos2+1;  
    if(pos3>160) //Limit the angle  
    {  
        pos3=160;  
    }  
    if(pos2>160) //Limit the angle of swinging backward  
    {  
        pos2=160;  
    }  
    myservo3.write(pos3); //The rod of right steering gear swings forward  
    myservo2.write(pos2); //The rod of left steering gear swings backward  
    delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'B') {  
    pos3=pos3-1;  
    pos2=pos2-1;  
    if(pos3<90) //Limit the angle of descent  
    {  
        pos3=90;  
    }  
}
```

```
if(pos2<120)    //Limit the angle of swinging forward
{
    pos2=120;
}
myservo3.write(pos3); //The rod of right steering gear swings forward
myservo2.write(pos2); //The rod of left steering gear swings backward
delay(10);    //Adjust the rotation speed of the steering gear
}
}
```

## (2) Experiment phenomenon

Turn on the Bluetooth switch

Virtual rocker forward and backward controls the extension and retraction of the mechanical arm.

Virtual rocker left and right controls the left and right turn of the mechanical arm.

Move the virtual rocker to the upper left and lower left corners to control the swing of the left steering gear of the mechanical arm.

Move the virtual rocker to the upper right and lower right corners to control the swing of the right steering gear of the mechanical arm.

4 drag bars can each control the rotation of four steering gears in real-time.

## 5.5 Mobile APP controls the mechanical arm 2

The APP controlling the mechanical arm adds functions for disconnecting and reconnecting the steering gears.

### (1) Example code

```
#include <Servo.h> //Add the steering gear library files
Servo myservo1; // create steering gear instances to control
Servo myservo2;
Servo myservo3;
Servo myservo4;

int pos1=90, pos2=130, pos3=90, pos4=90; //Define variables for 4 steering gear
angles and assign initial values (i.e. the attitude angle value at startup)

char bleVal;
```



```
void setup() {
  Serial.begin(9600);
  myservo1.attach(7); //Set the control pin of steering gear 1 as A1
  myservo2.attach(10); //Set the control pin of steering gear 2 as A0
  myservo3.attach(9); //Set the control pin of steering gear 3 as D8
  myservo4.attach(8); //Set the control pin of steering gear 4 as D9
  //Posture at startup
  myservo1.write(pos1);
  myservo2.write(pos2);
  myservo3.write(pos3);
  myservo4.write(pos4);
  delay(1000);
}

void loop() {
  if(Serial.available() > 0)
  {
    String bleVal1 = Serial.readStringUntil('*');
    if(bleVal1[0] == '$') //If the first character received is '$'
    {
      bleVal = bleVal1[1];
      //Serial.println(bleVal);
      //If the second character received is 'c'
      if(bleVal1[1] == 'c') { //Control the steering gear of the claw
        //Convert the second to last character of a string to an integer and assign
it to pos4
        pos4 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos4 = map(pos4, 0, 180, 90, 0); //Mapping function, converting values
from 0~180 to 90~0. When the drag value increases, the claw opens
        myservo4.write(pos4);
        Serial.println(pos4);
      }
      if(bleVal1[1] == 'l') { //Control the left steering gear
        pos2 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos2 = map(pos2, 0, 180, 160, 90); //Mapping function, converting values
from 0~180 to 160~90.
        myservo2.write(pos2);
        Serial.println(pos2);
      }
    }
  }
}
```

```
    if(bleVal1[1] == 'r') {    //Control the right steering gear
        pos3 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos3 = map(pos3, 0, 180, 20, 160);    //Mapping function, converting values
from 0~180 to 0~90.
        myservo3.write(pos3);
        Serial.println(pos3);
    }
    if(bleVal1[1] == 'b') {    //Control the right steering gear
        pos1 =
String(String(bleVal1).substring(2,String(bleVal1).length())).toInt();
        pos1 = map(pos1, 0, 180, 180, 0);    //Mapping function, When the drag value
increases, the mechanical arm turns right
        myservo1.write(pos1);
        Serial.println(pos1);
    }
}
}
if(bleVal == 'L') {
    pos1=pos1+1;    //pos1 self adding 1
    if(pos1>180)    //Limit the angle of turning left
    {
        pos1=180;
    }
    myservo1.write(pos1);    //The mechanical arm turns left
    delay(5);    //Adjust the rotation speed of the steering gear
}
if(bleVal == 'R') {
    pos1=pos1-1;    //pos1 subtracts 1 from itself
    if(pos1<1)    //Limit the angle of turning right
    {
        pos1=1;
    }
    myservo1.write(pos1);    //The steering gear 1 executes action, and the
mechanical arm turns right
    delay(5);    //Adjust the rotation speed of the steering gear
}
if(bleVal == 'Q') {
    pos2=pos2-1;
    if(pos2<90)    //Limit the angle of swinging forward
    {
        pos2=90;
    }
}
```

```
}  
myservo2.write(pos2); //The rod of the left steering gear swings forward  
delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'Z') {  
    pos2=pos2+1;  
    if(pos2>170) //Limit the angle of swinging backward  
    {  
        pos2=170;  
    }  
    myservo2.write(pos2); //The rod of the left steering gear swings backward  
    delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'E') {  
    pos3=pos3+1;  
    if(pos3>180) //Limit the angle  
    {  
        pos3=180;  
    }  
    myservo3.write(pos3); //The rod of the right steering gear swings forward  
    delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'C') {  
    pos3=pos3-1;  
    if(pos3<60) //Limit the angle of descent  
    {  
        pos3=60;  
    }  
    myservo3.write(pos3); //The rod of the right steering gear swings backward  
    delay(10); //Adjust the rotation speed of the steering gear  
}  
  
if(bleVal == 'F') {  
    pos3=pos3+1;  
    pos2=pos2+1;  
    if(pos3>160) //Limit the angle  
    {  
        pos3=160;  
    }  
    if(pos2>160) //Limit the angle of swinging backward  
    {  
        pos2=160;  
    }  
}
```

```
myservo3.write(pos3); //The rod of the right steering gear swings forward
myservo2.write(pos2); //The rod of the left steering gear swings backward
delay(10); //Adjust the rotation speed of the steering gear
}

if(bleVal == 'B') {
    pos3=pos3-1;
    pos2=pos2-1;
    if(pos3<90) //Limit the angle of descent
    {
        pos3=90;
    }
    if(pos2<120) //Limit the angle of swinging forward
    {
        pos2=120;
    }
    myservo3.write(pos3); //The rod of the right steering gear swings forward
    myservo2.write(pos2); //The rod of the left steering gear swings backward
    delay(10); //Adjust the rotation speed of the steering gear
}

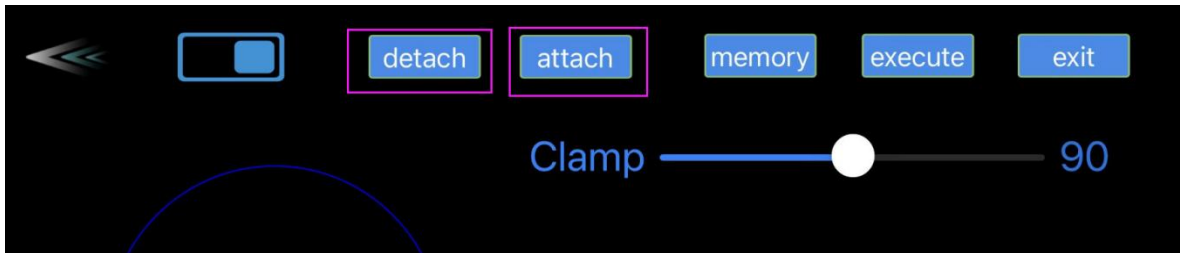
if(bleVal == 't') {
    myservo1.detach();
    myservo2.detach();
    myservo3.detach();
    myservo4.detach();
}

if(bleVal == 'y') {
    myservo1.attach(7);
    myservo2.attach(10);
    myservo3.attach(9);
    myservo4.attach(8);
}
}
```

## (2) Experiment phenomenon

Turn on the Bluetooth switch

Virtual rocker forward and backward controls the extension and retraction of the mechanical arm.

Virtual rocker left and right controls the left and right turn of the mechanical arm.
Move the virtual rocker to the upper left and lower left corners to control the swing of the left steering gear of the mechanical arm.
Move the virtual rocker to the upper right and lower right corners to control the swing of the right steering gear of the mechanical arm.
4 drag bars can each control the rotation of four steering gears in real-time.
Click the detach button to disconnect all the steering gears
Click the attach button to reconnect all the steering gears


## 5.6 Mobile APP teaching function of the mechanical arm

APP controlling adds teaching function

### (1) Example code

The program is quite long, please open the example program to view.

 5\_6\_APP\_Teaching

5/25/2023 6:01 PM

File folder

### (2) Experiment phenomenon

Click the memory button to save the current posture of the mechanical arm, which can save 20 postures. Click the execute button to cycle through all saved postures. Click the exit button to exit the loop action.

