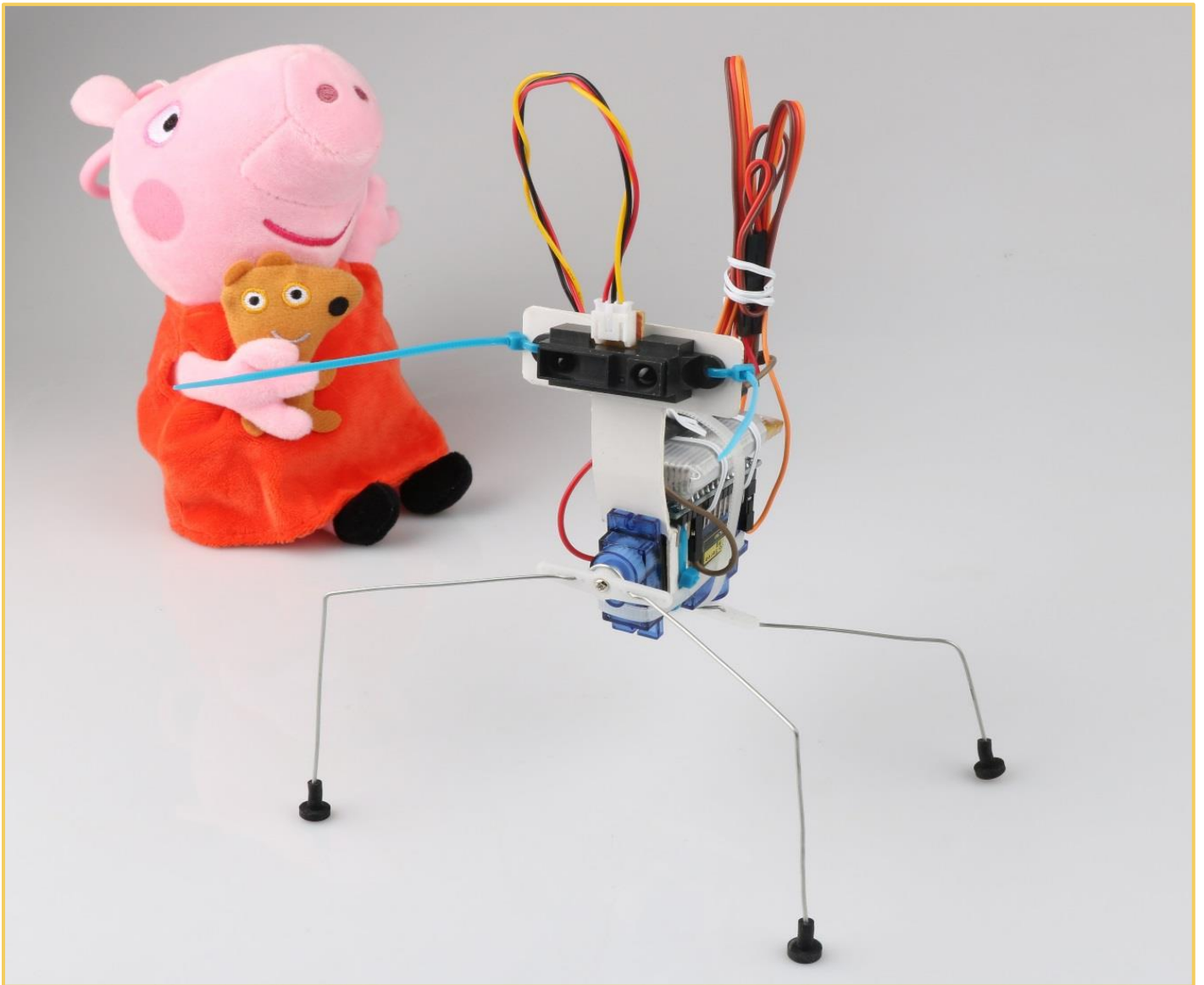


Insect-Robot Instruction Manual



Github : <https://github.com/keywish/keywish-insect-robot-kit>

1.0 What You'll Learn

In this project, you'll create an Insect Robot that walks forward on four legs. Actually it's not really an insect because it only has four legs, insects do have six legs, right? However, the robot got the name because of its shape with the thin wire legs and the infrared sensor as eyes.

We will briefly learn servo, Arduino Nano and infrared sensor and build a body for the insect by binding two servos together and shaping legs for it from an iron wire. Arduino will turn the two servos into one at a time, which moves each pair of metal legs like real legs so the insect can crawl forward. We will also strap a battery on the body so our insect can move autonomously.

Once you've learned the techniques in this chapter, you can easily extend your Insect Robot with new tentacles and sensors and new code. This easy-to-assemble kit is designed to enable creative engineering and robotics learning for age 6 and above. It is the best way to start building a robot on Arduino.

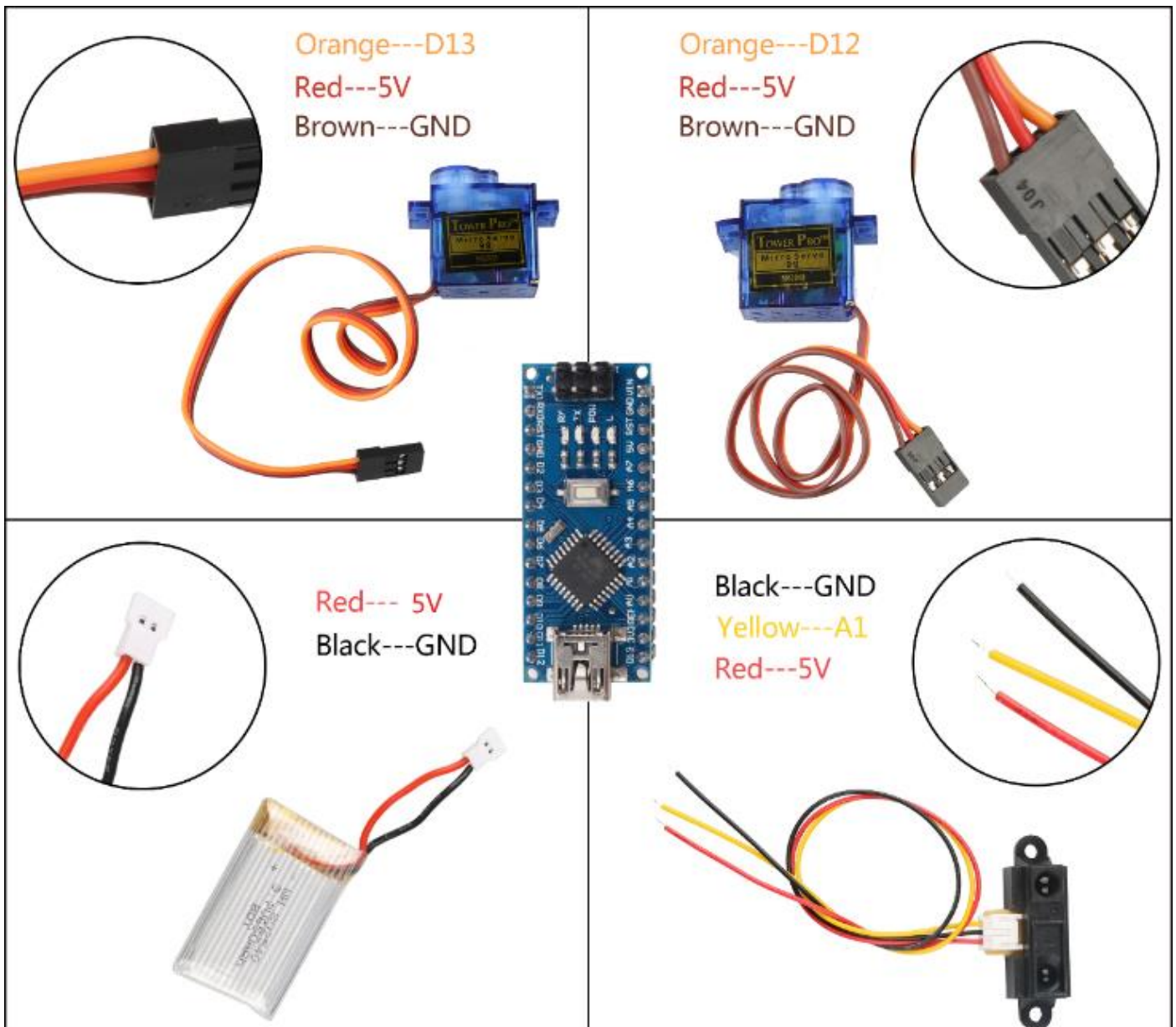
List of Components and Tools

- ◆ Arduino nano V3.0 Atmel ATmega328 Mini-USB Board*1
- ◆ SG90 Servo*2
- ◆ SHARP GP2Y0A41SK0F IR Sensor*1
- ◆ 3.7V/ 600mAh Lithium battery *1(With USB Charger)
- ◆ Steel Wire 200mm x 1mm*1
- ◆ ABS Sheet (50x50mm)*1
- ◆ Double Sided Foam Tape(L/W/H:40x30x3mm)*1
- ◆ Cable Tie 1.8x100mm*5 , 4x200mm*1
- ◆ 3*75mm cross screwdriver*1
- ◆ Some Dupont Lines
- ◆ Scissors*1

Other tools :

Sharp-nose pliers*1

Electric iron



Picture-2 Device list and physical connection

To start program, a Mini USB data line and a computer are needed. After the assembly is completed, you need to use the Arduino IDE to burn the code. Please refer to 《How to start.pdf》

2.0 Components Introduction

2.1 Arduino Nano

The arduino Nano is an ultra-small simple I/O platform based on open source code, it has a big advantage in size compared with UNO R3 board.



Figure-3

2.2 Servo Motor

Servo motors come in different sizes and prices and are based on different technologies. In this context, we are talking about micro servos, which usually used in R/C toys. Servo motors have a servo controller that directs the position of the motor wherever we want. The red wire is power, the gray is GND and the orange is the data wire.

The motor itself is a DC (direct current) motor with gears. Servo motors usually rotate rather slowly with a strong torque. You can buy micro servo motors with either limited rotation or continuous rotation. Limited rotation models work for most purposes, and you can control their movement quite precisely by degrees. If use continuous rotation servos, you can control only speed and direction.

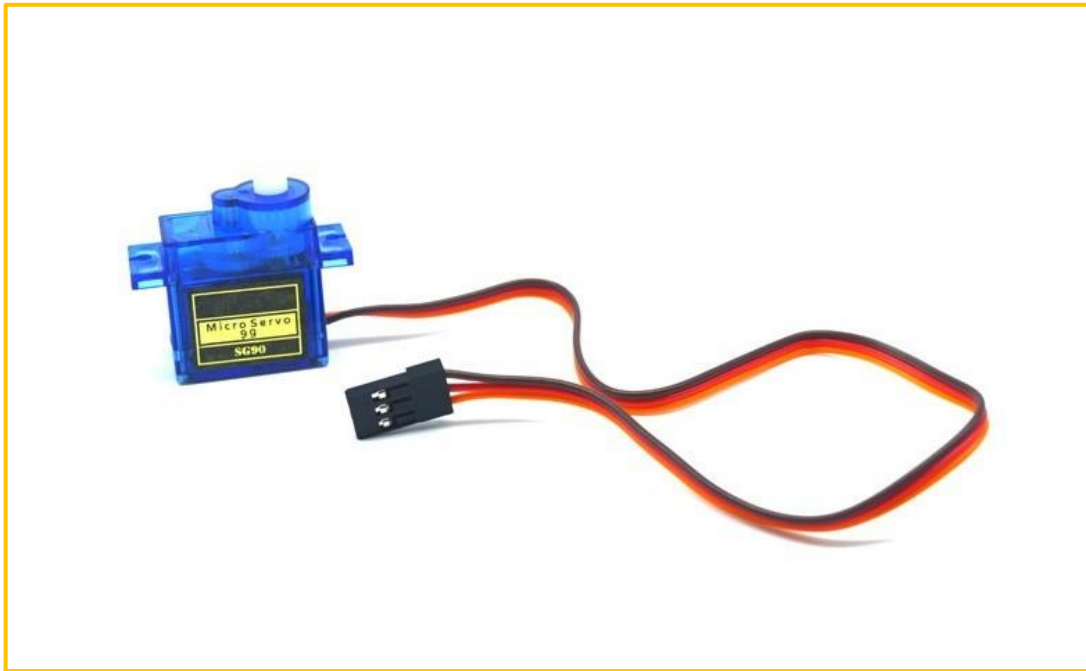


Figure-4

2.3 SHARP GP2Y0A41SK0F

GP2Y0A41SK0F is a distance measuring sensor unit, composed of an integrated combination of PSD (position sensitive detector) , IR-LED (infrared emitting diode) and signal processing circuit. The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method. This device outputs a corresponding voltage to the detection distance. So this sensor can also be used as a proximity sensor.



Figure-5

3.0 Assembly

Through the above steps, we have learned about the devices, functions, parameters and schematic diagrams of insect robots. Now let's start to assemble.

3.1 Fixed two SG90

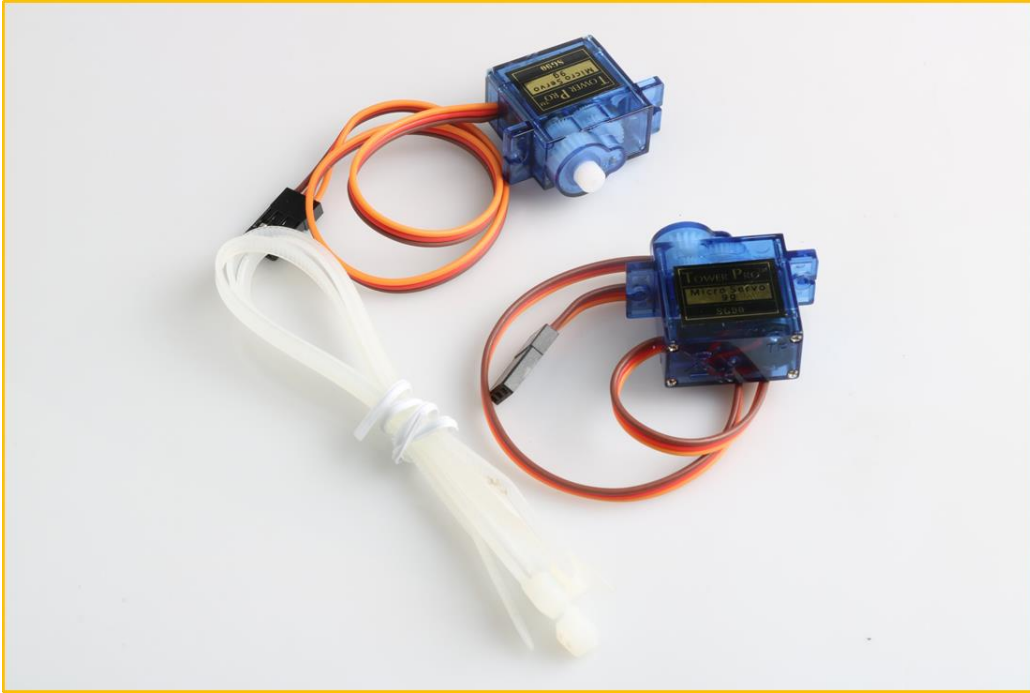


Figure-6

Gluing the two servo motor together with cable

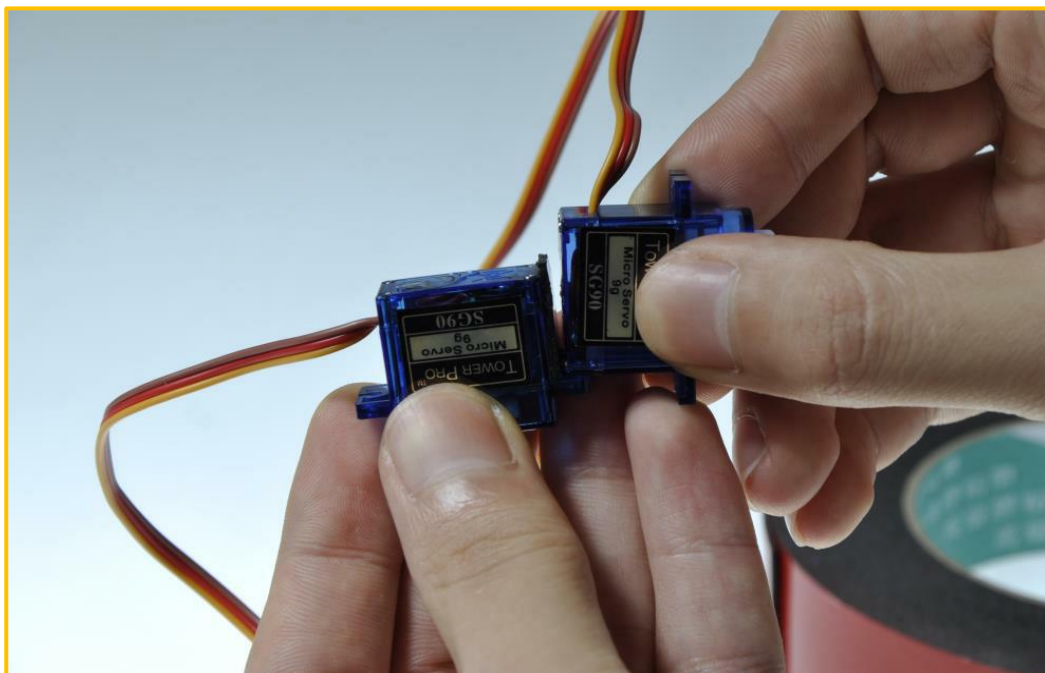


Figure-7

Tying the servo motor tightly with cable tie and cut out the excess.

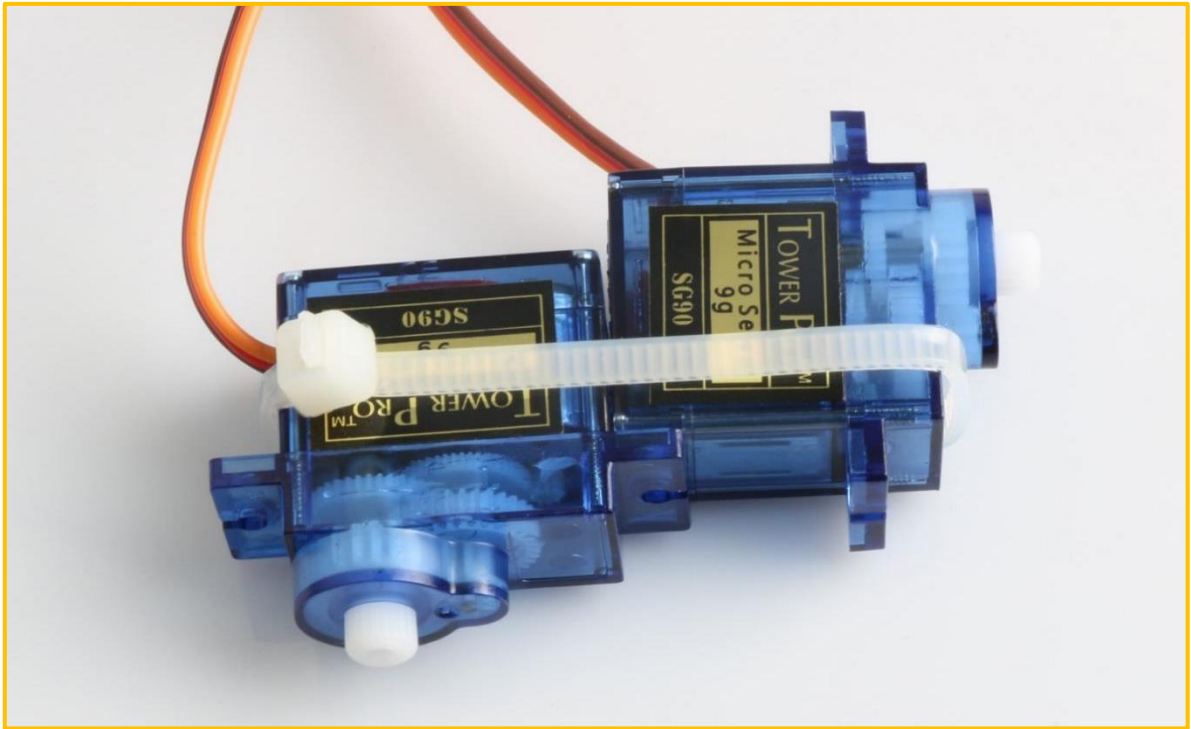


Figure-8

Put a foam on the top of the servo (at A of Figure 9), which is used to support the NANO main control board , as shown in Figure 10.

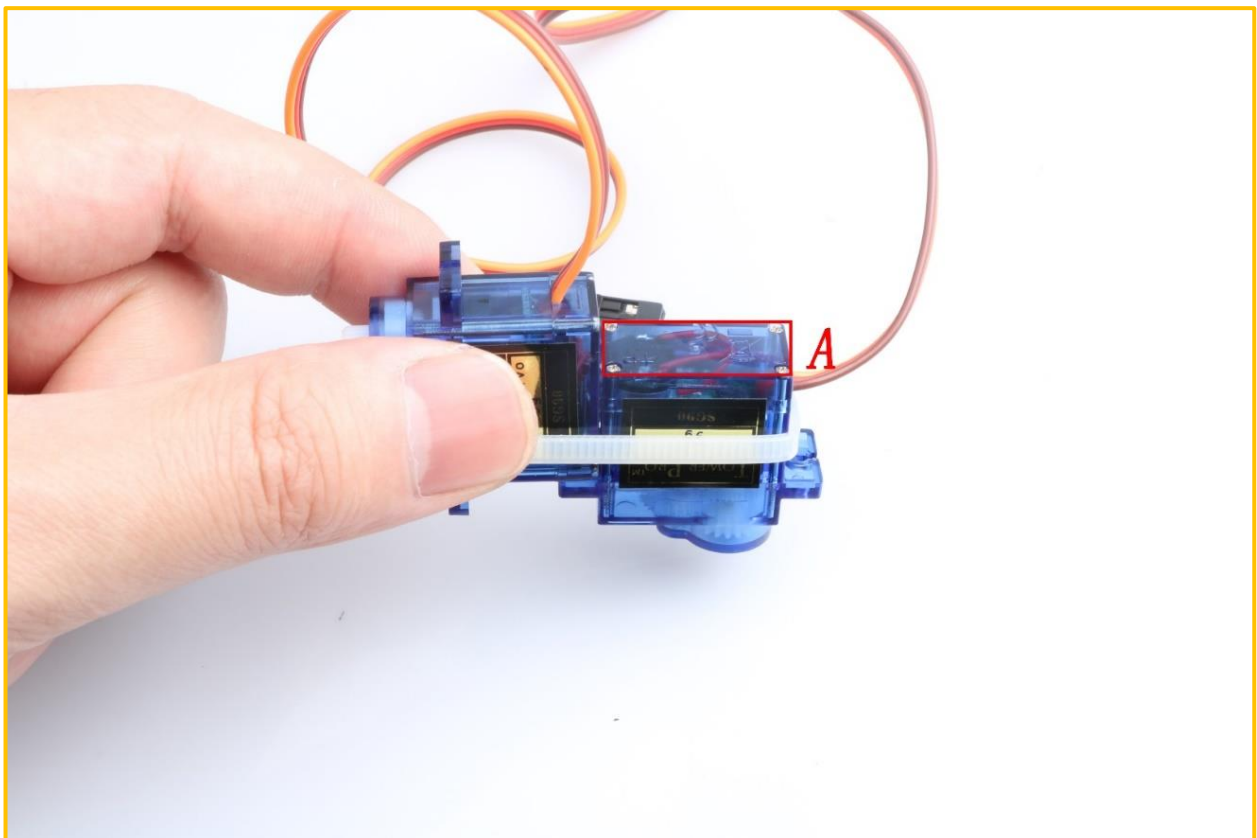


Figure-9

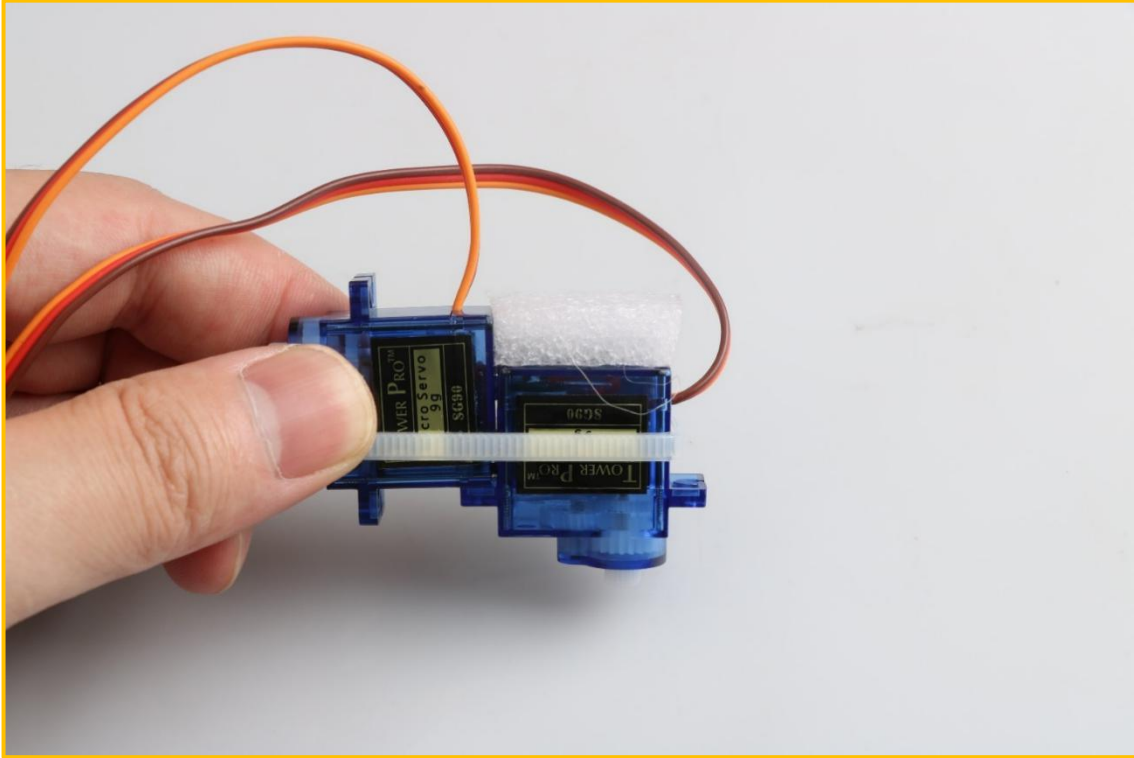


Figure 10

3.2 Making infrared sensor fixed frame

Cut the paper to the shape and size as figure 11 shown.

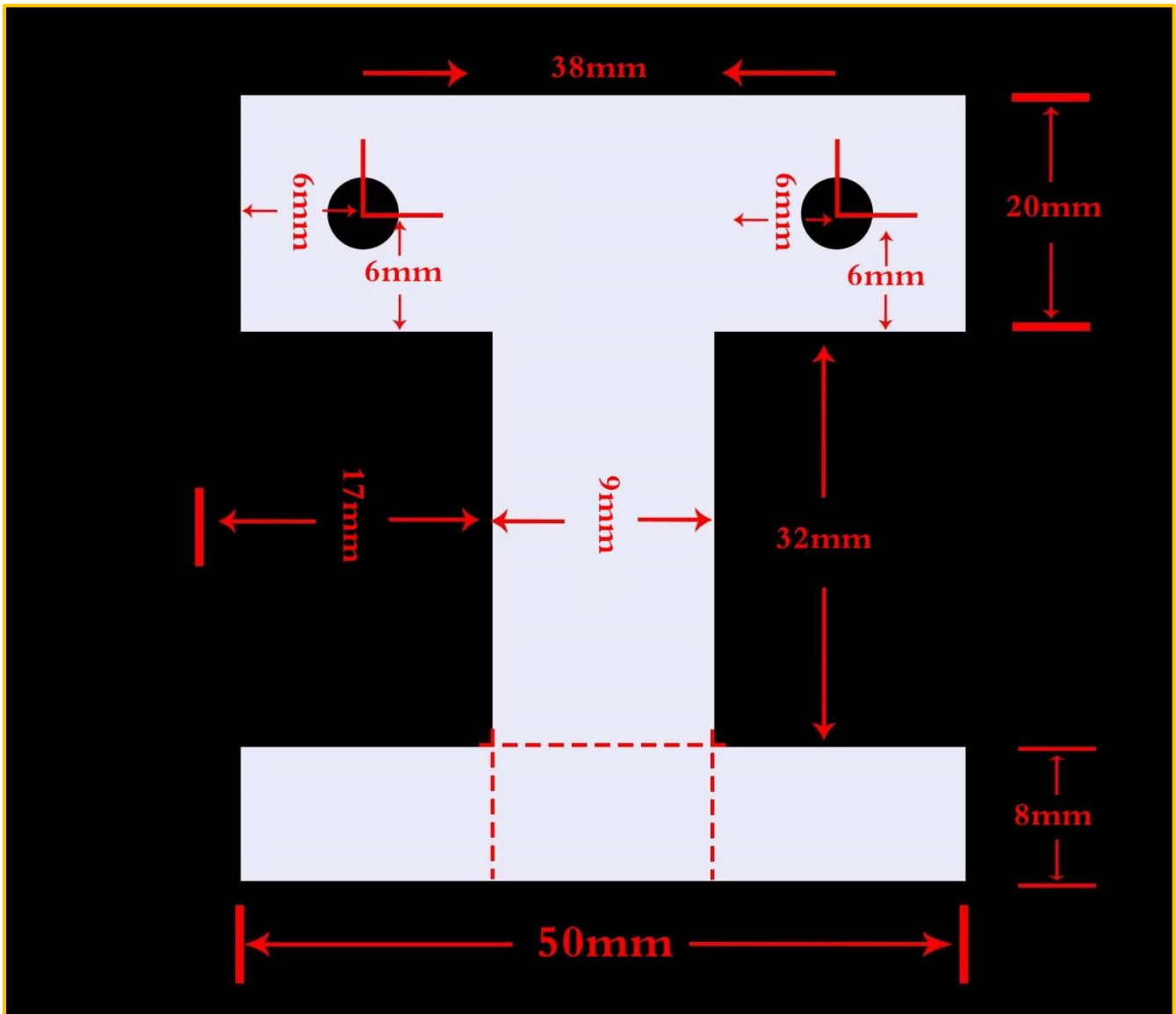


Figure 11

Bend the cutted paper into the shape as figure 12.

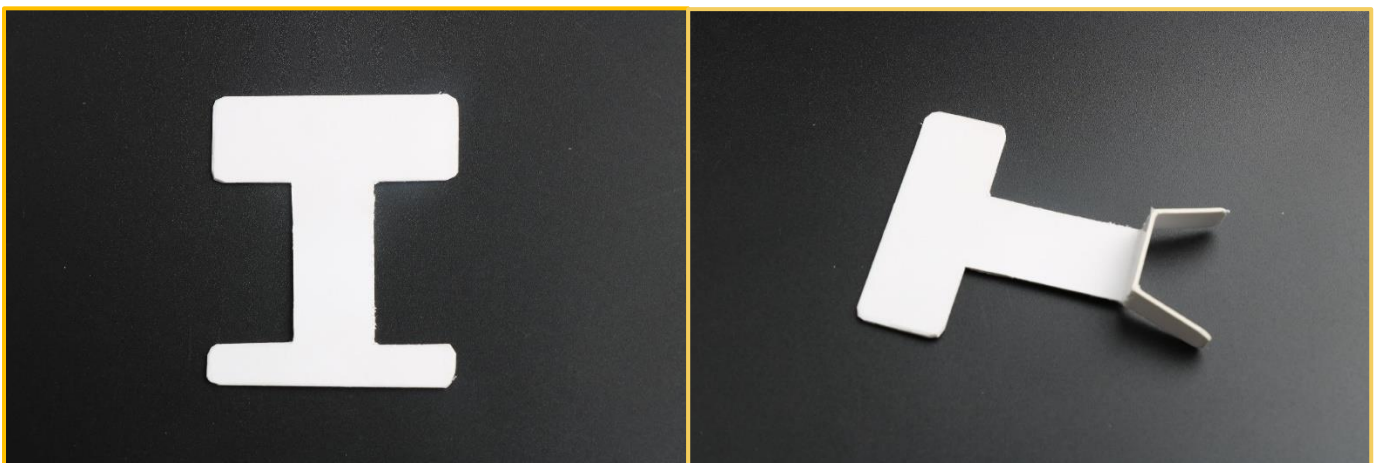


Figure 12

Install the paper on the servo as figure 13.



Figure 13

Fix the paper with cable tie as figure 14.

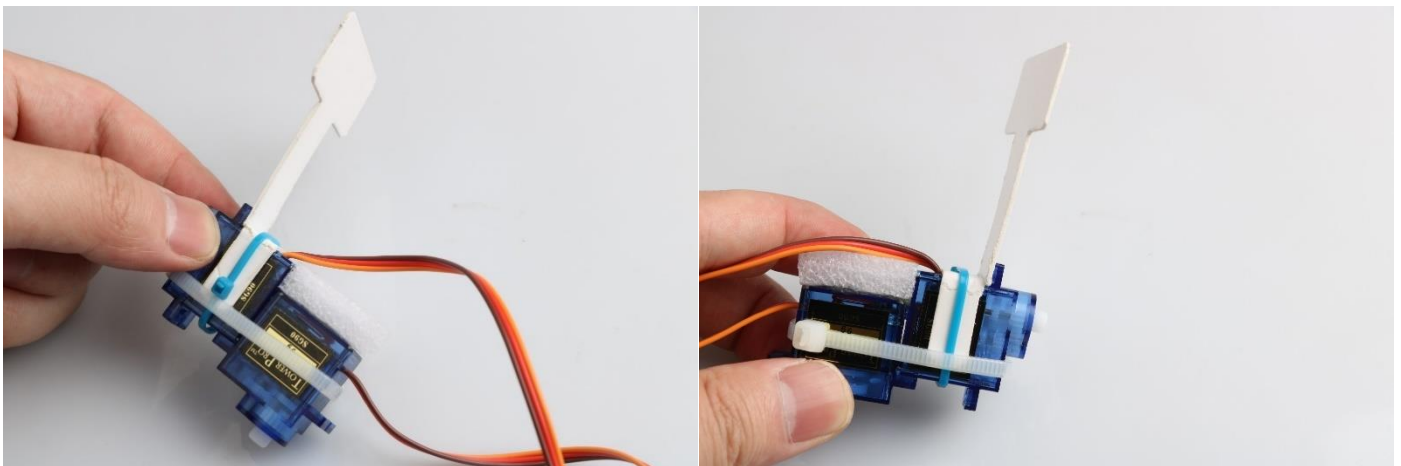


Figure 14

Install the Nano board on the servo as figure 15 but don't fix.

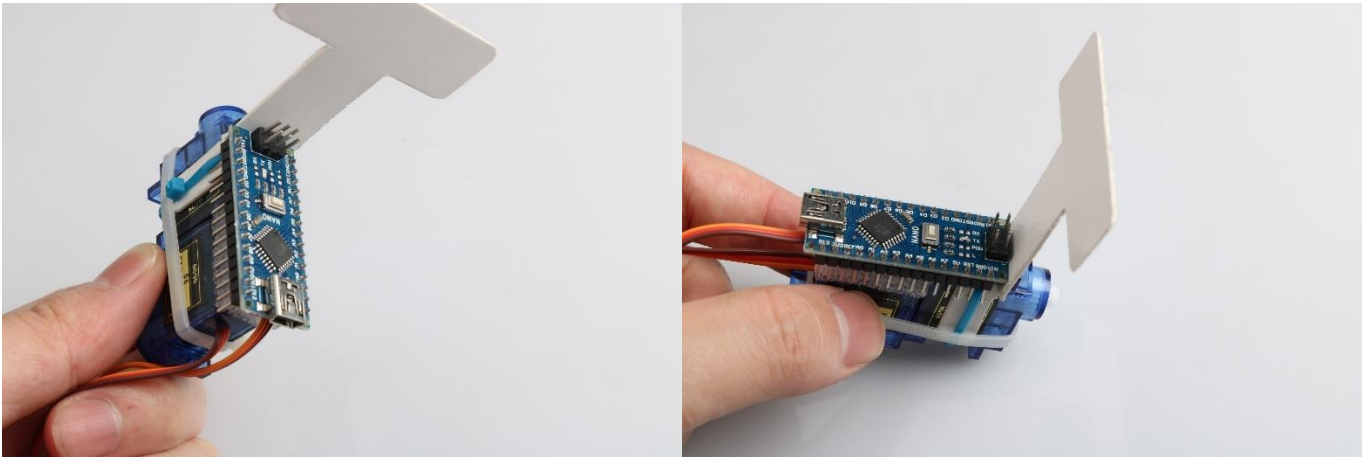


Figure 15

3.3 Install infrared sensor

Use the blue cable tie to tie as figure 16, figure17.

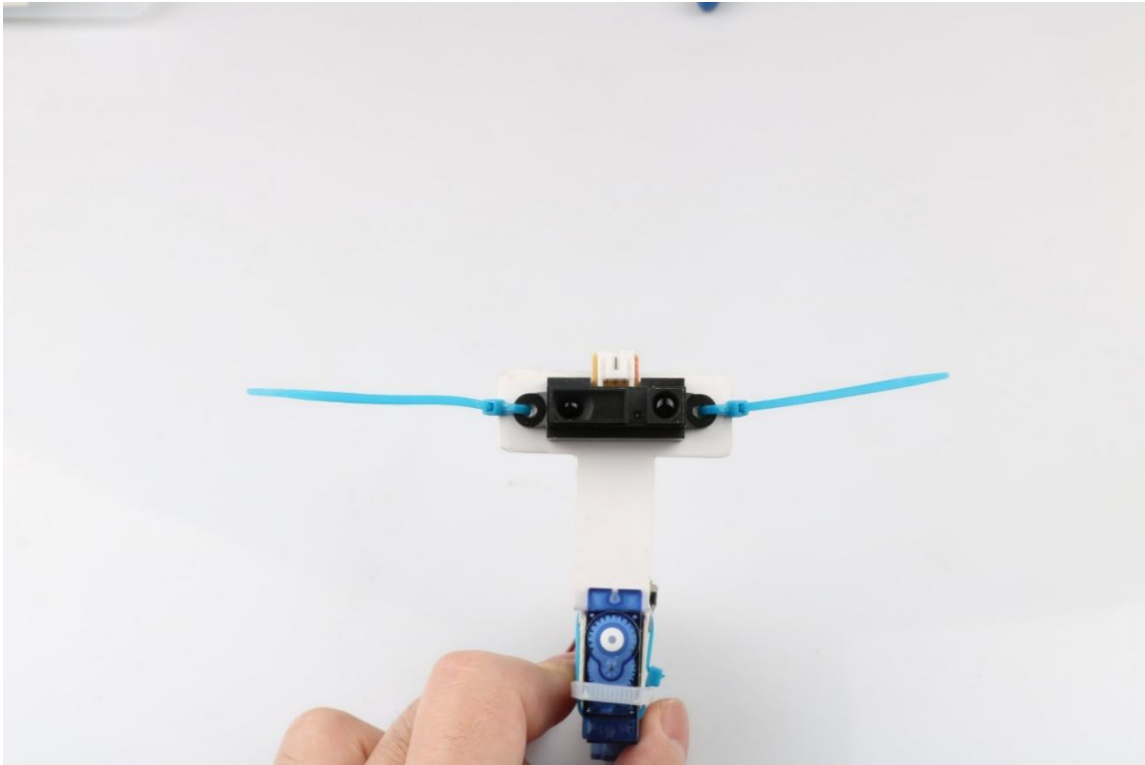


Figure 16

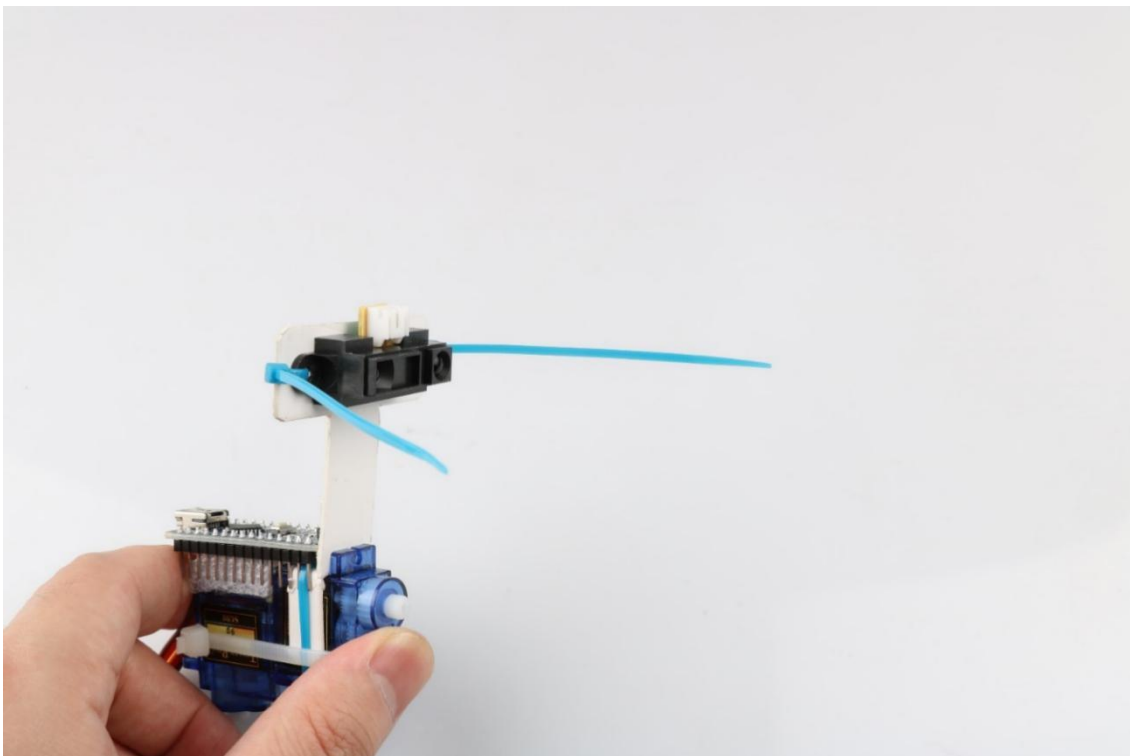


Figure 17

3.4 Making mechanical feet

Step 1: Bend the iron wire to V shape as figure 18, then use plier to bend 90 degrees at the position about 1cm from the previous bend as figure 19 figure 20 .

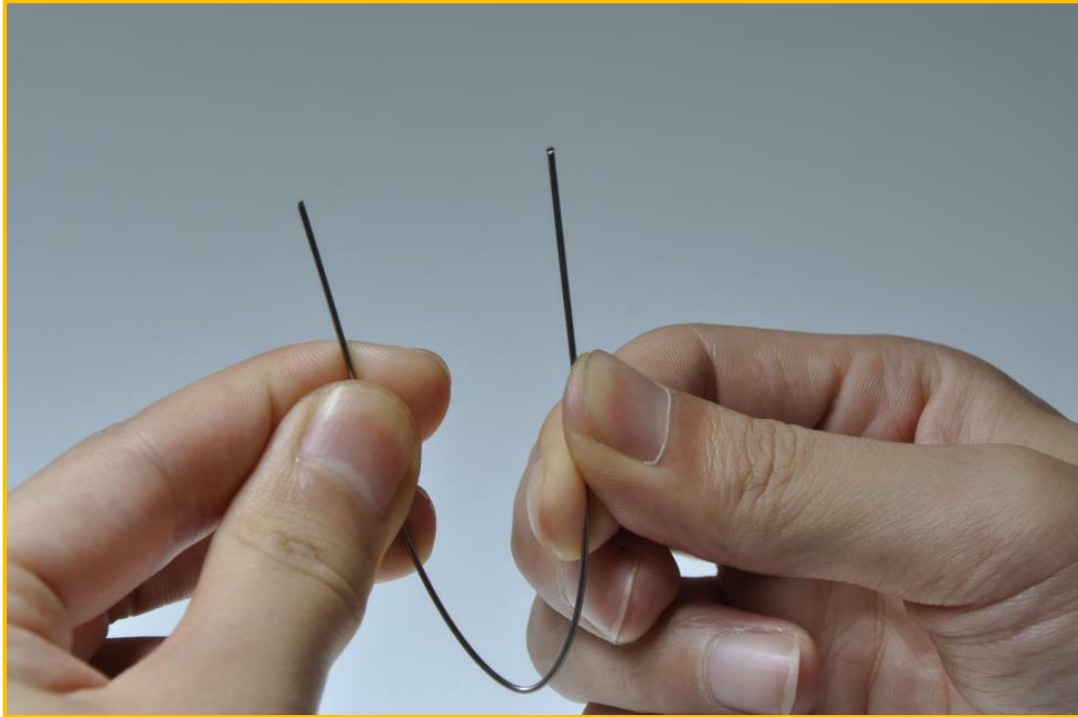


Figure 18



Figure 19



Figure 20

Step2: Pass the 2 feet of the wire through the two holes of the steering wheel.

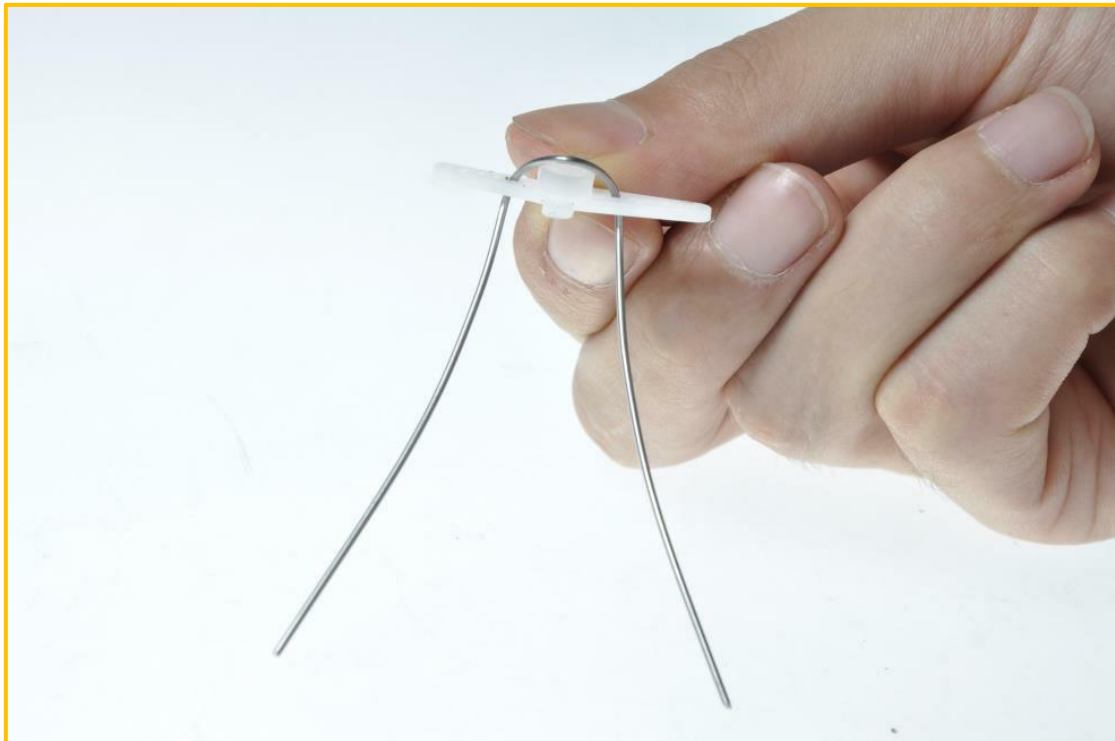


Figure 21

Step3: Use the pliers to bend the two legs of the wire to the other end of the sharp corner as figure 22;
Note: Sharp-nose pliers is recommended to avoid breaking the white support.

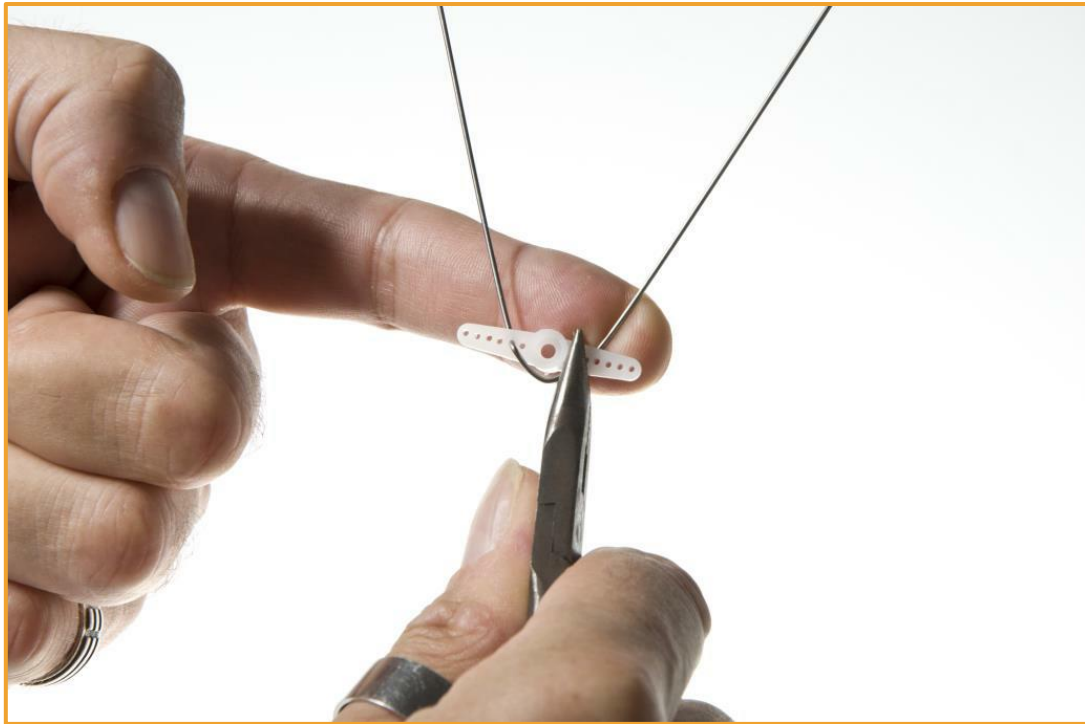


Figure-22

bending, The front leg is completed as Figure-23, and behind legs as Figure -24.

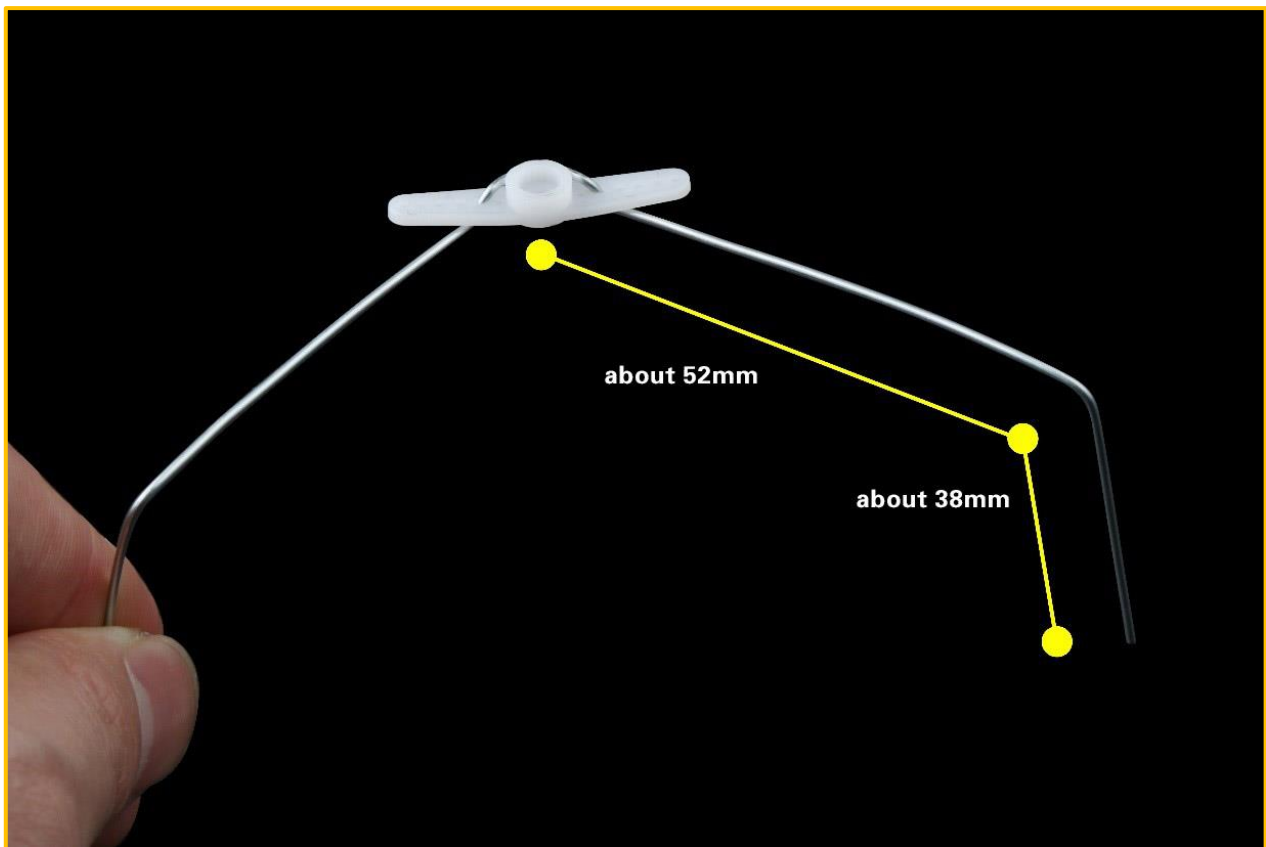


Figure-23

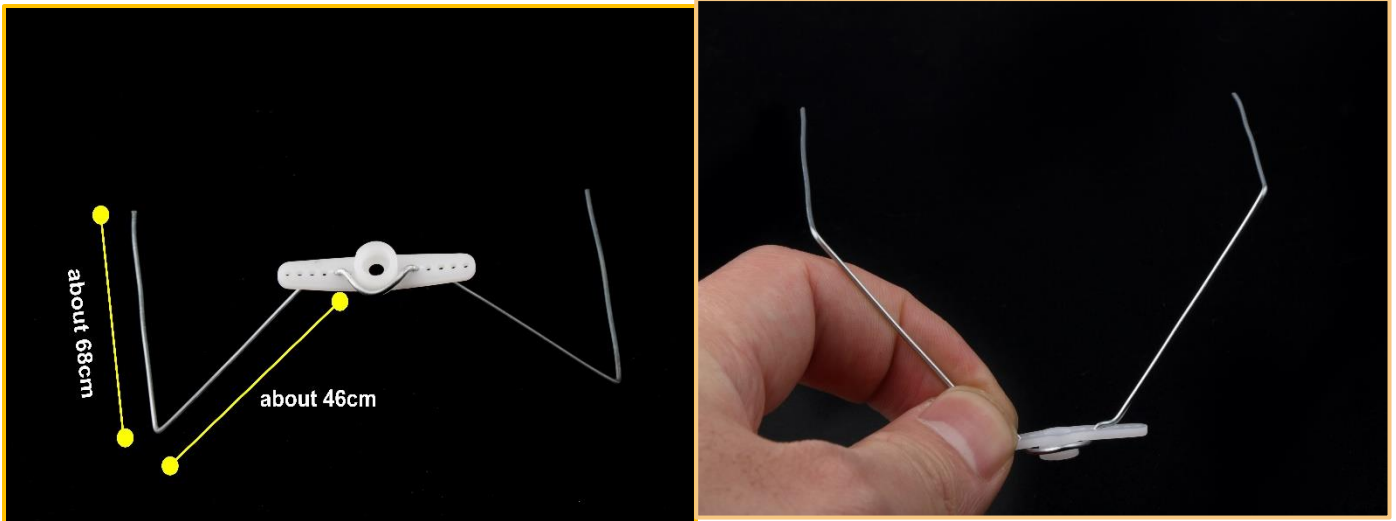


Figure-24

Through the above steps, you have completed the general look of the mechanical legs, Finally, the front and back legs bent, we can refer to the size shown in the figure on the wire Figure-25. Put "shoes" on the four legs of the robot and insert the wire into the slip-resistant particles (to prevent slipping during walking).

Note: The insertion process will be more difficult, you can first use a sharp object to slip a small hole in the non-slip particles (be careful not to prick your hand), so the wire will be easy to insert non-slip particles.

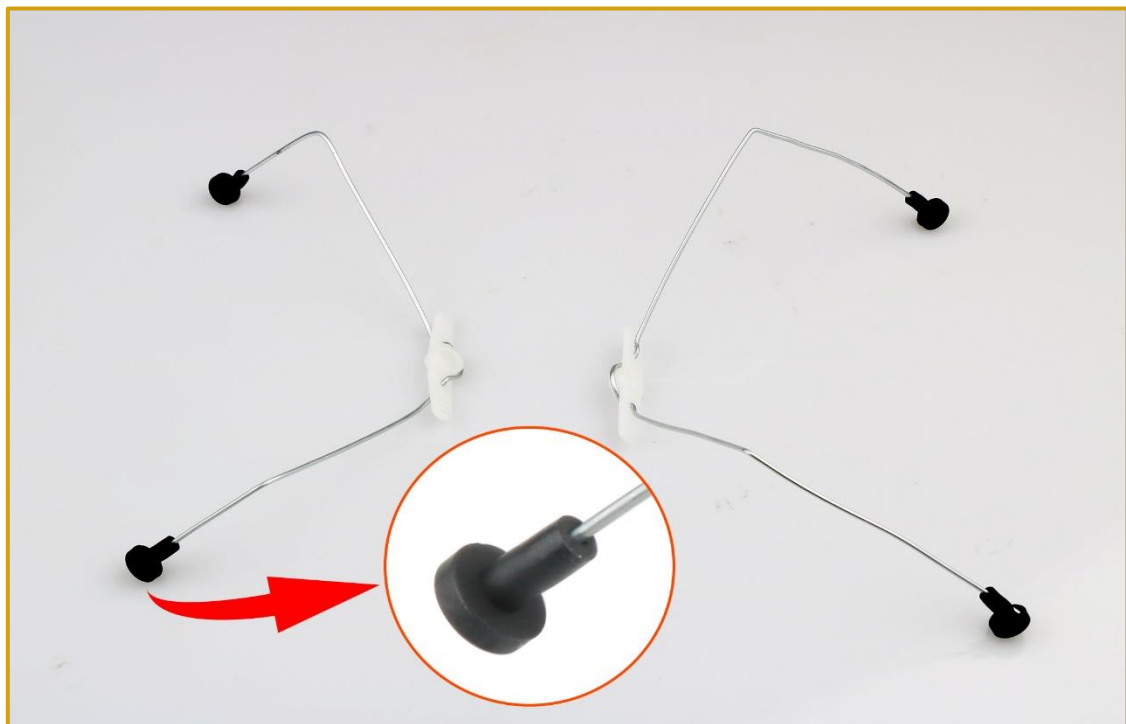


Figure-25

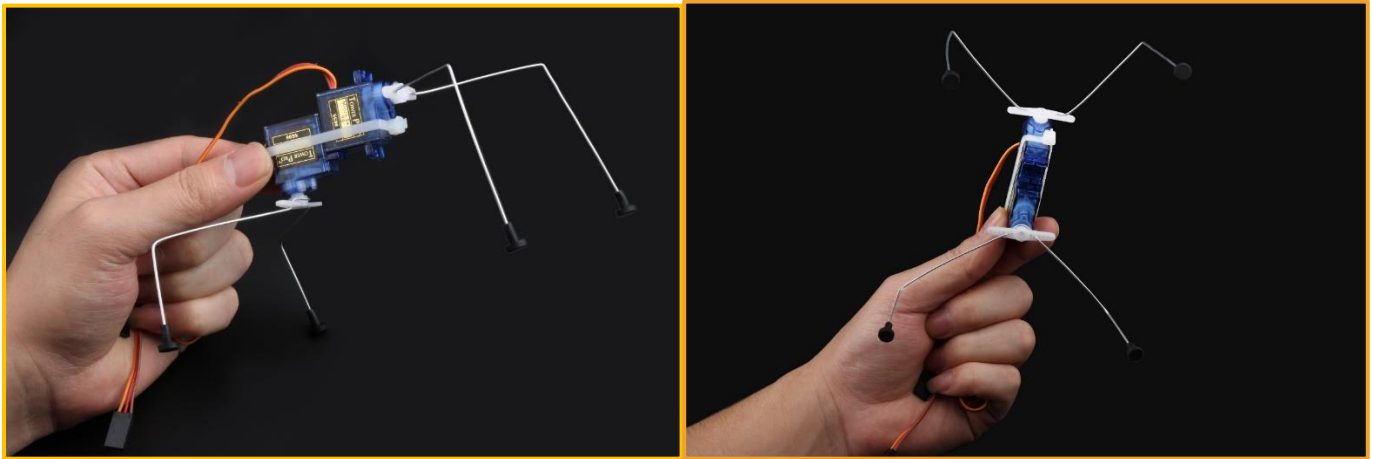


Figure-26

Finally, install the robot feet on the steering gear, if you correctly installed and completed the above steps, the worm robot can stand up as shown in Figure 27



Figure-27

3.5 Making Wire

3.5.1 Making Power Cords

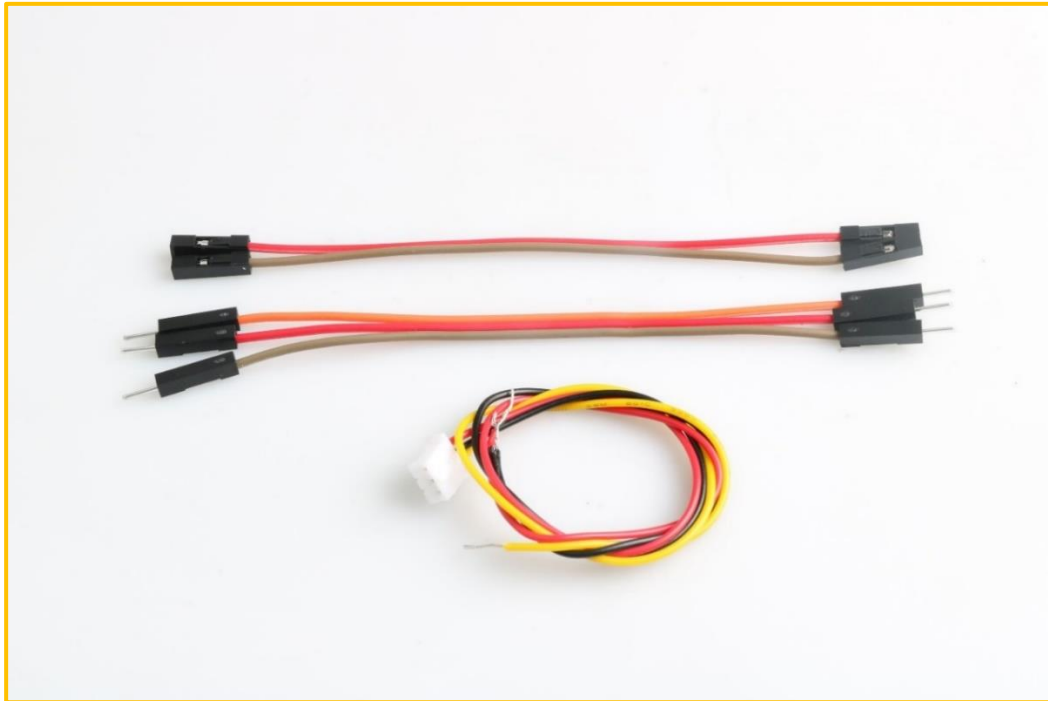


Figure-28

Firstly, cut the Gong Dupont line in Figure 28 from the middle, and remove the end of the Mother DuPont line, as shown in Figure 29.

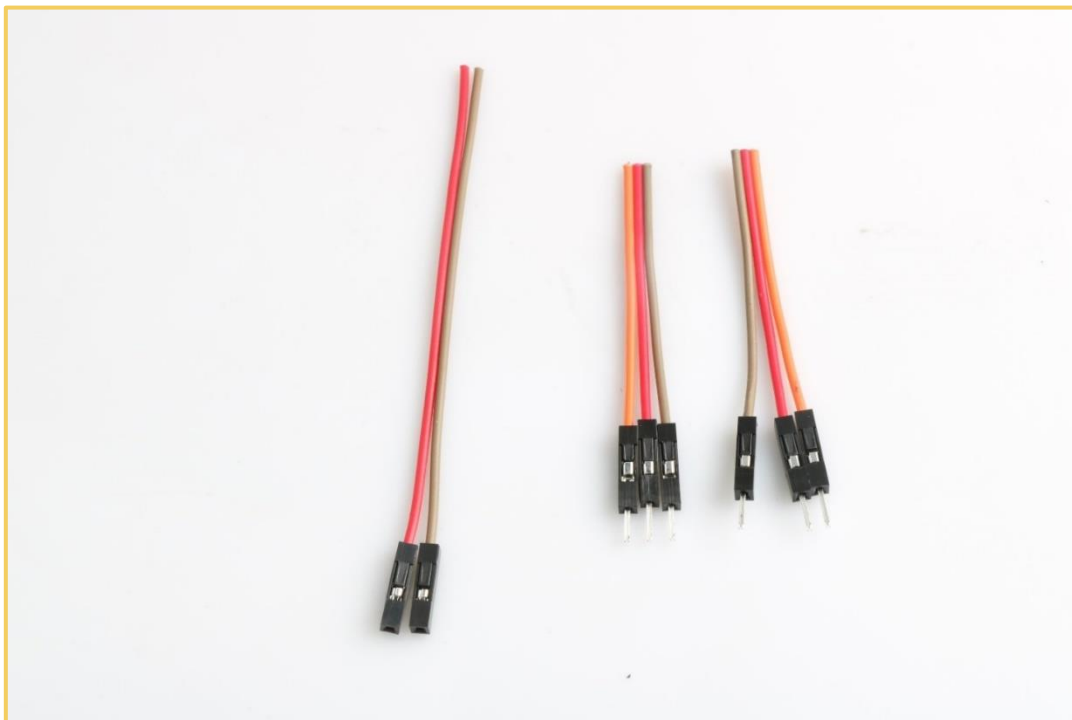


Figure-29

Connect the wires together as shown in Figure 30, and weld the wires with electrical iron.

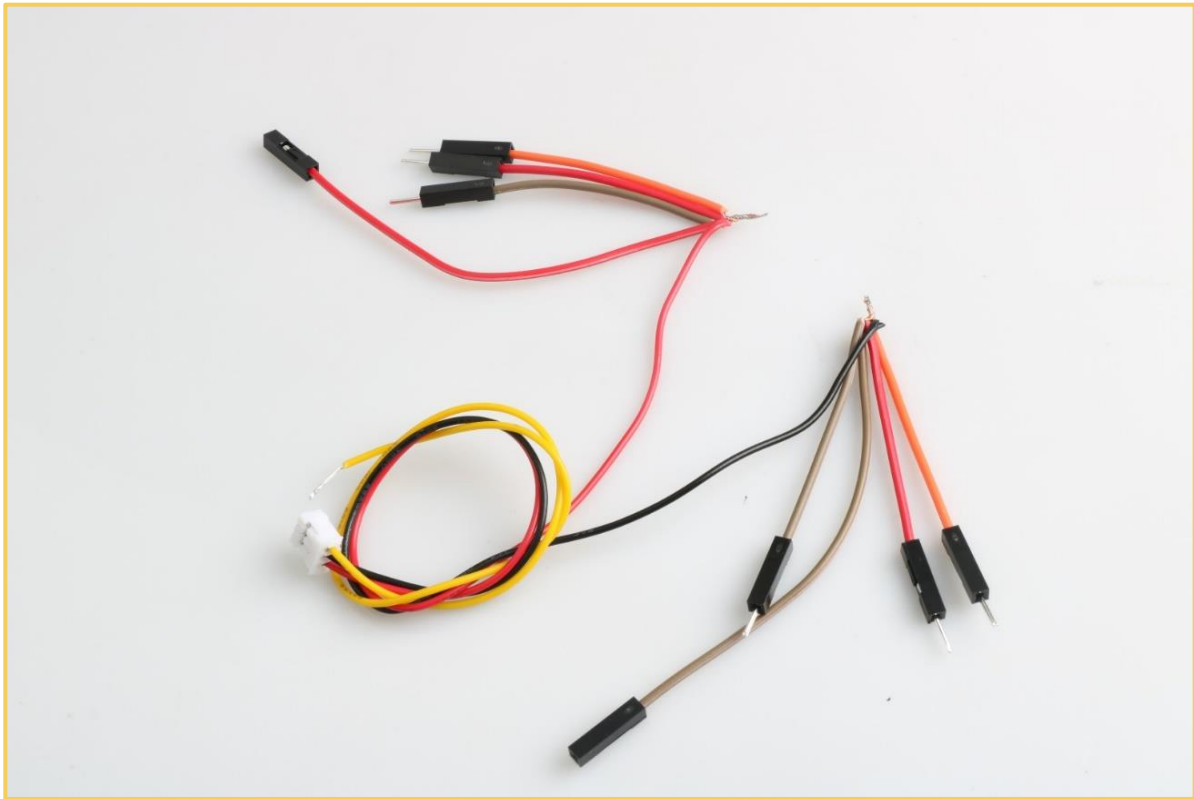


Figure -30

Use the heat shrinkable tube to bind the conductor interface, as shown in Figure31.

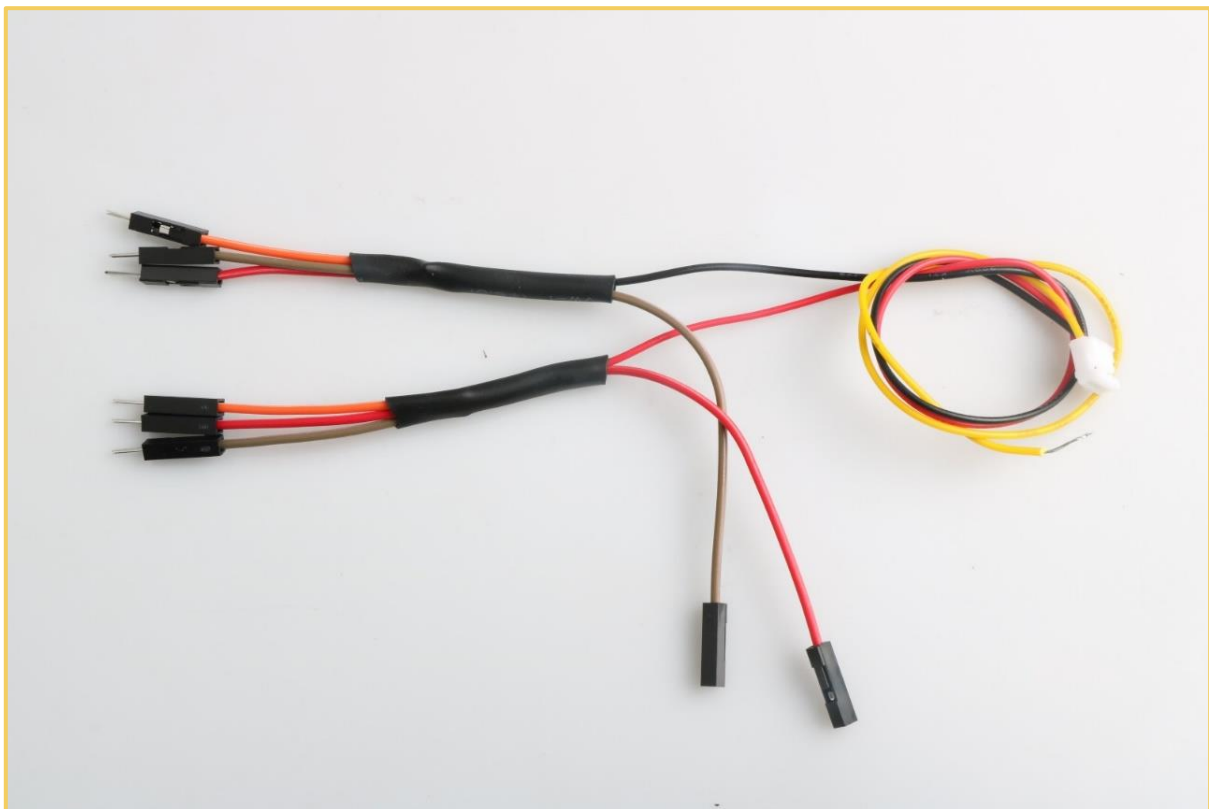


Figure-31

3.5.2 Rectifying the servo wire



Figure-32

First remove the 3P black connector terminal that comes with the servo cable, as shown in Figure 32. After removal, as shown in Figure 33.

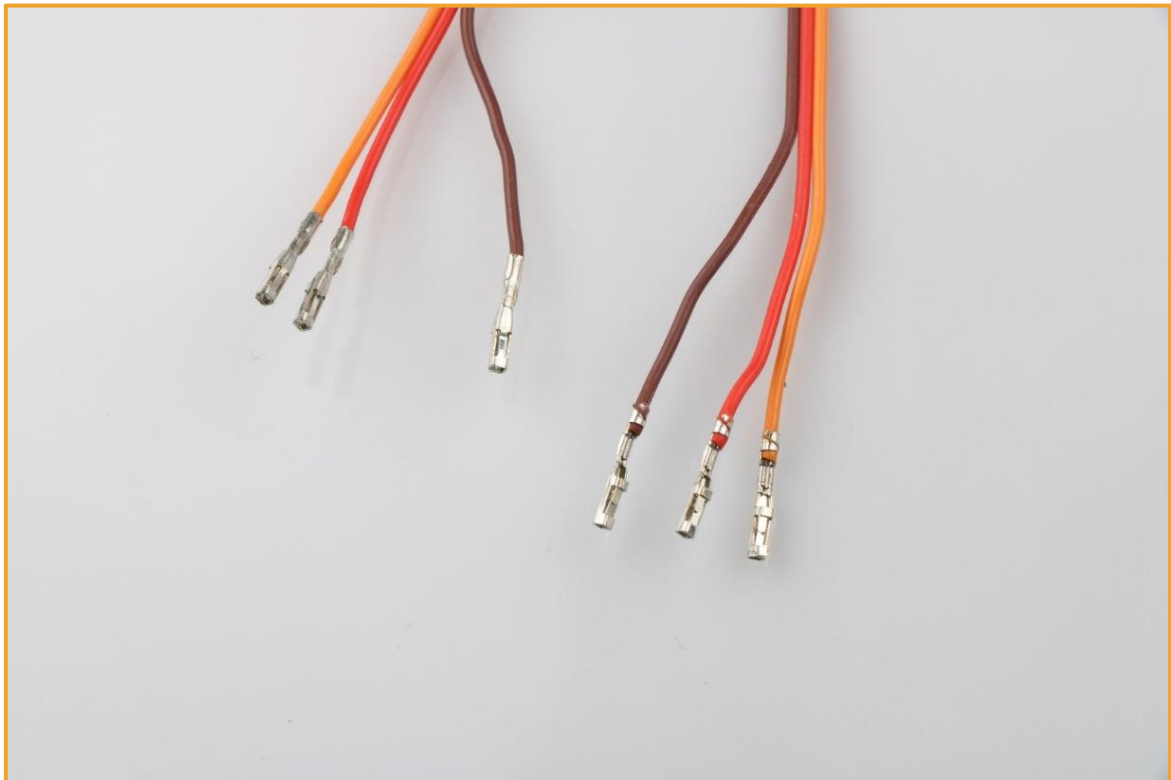


Figure-33

Then install the matching single black terminals, as shown in Figure 34, after the installation is complete as shown in Figure 35



Figure-34

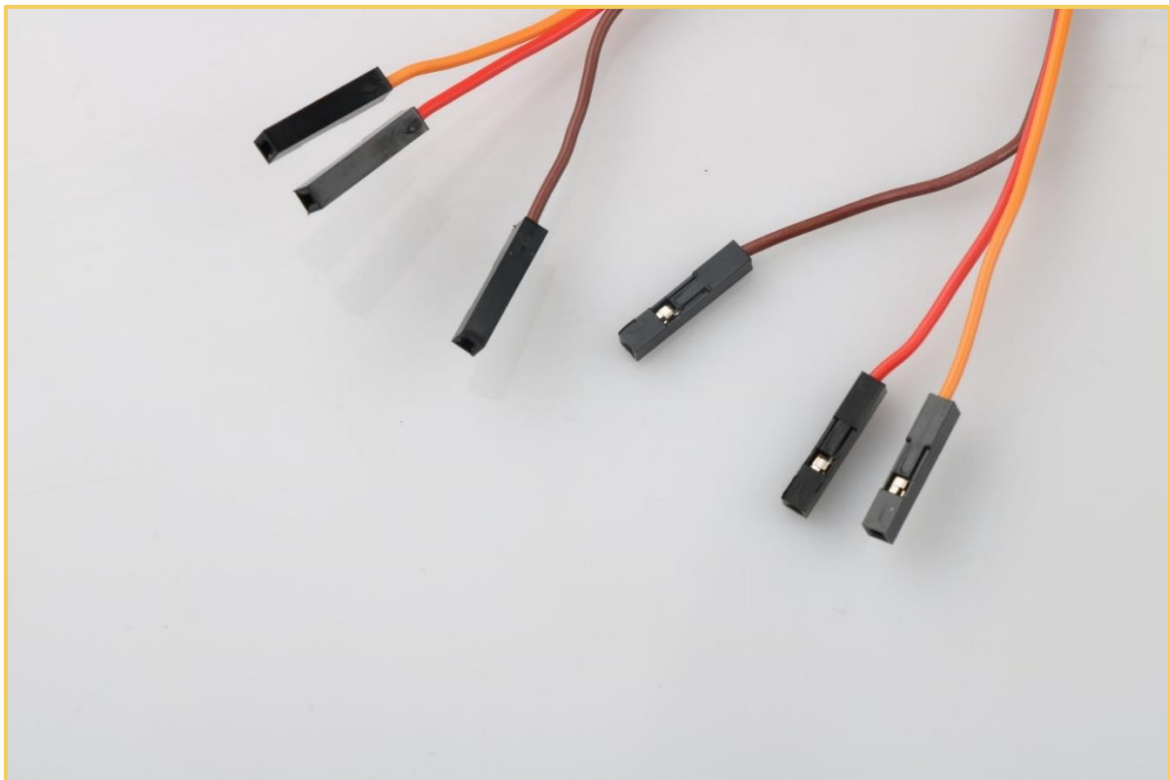


Figure-35

3.6 Battery Installation

Note: In order to facilitate the installation, you can remove the installed Nano main control board from the servo. Firstly, put two pieces of foam paper on the Nano board, as shown in Figure 36.

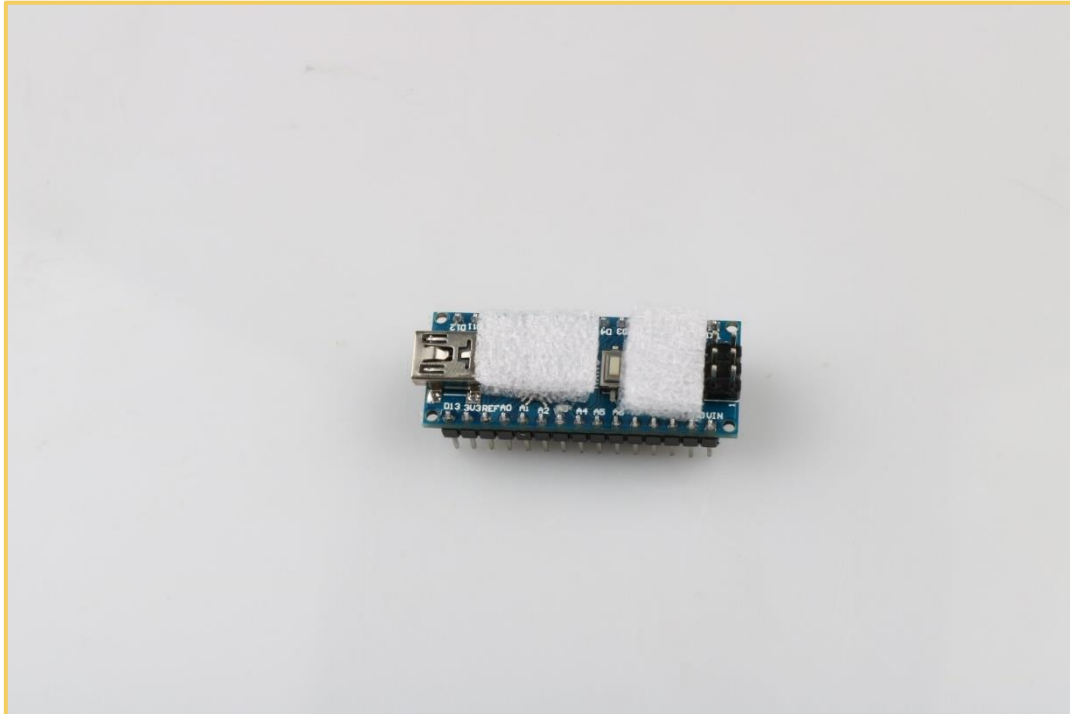


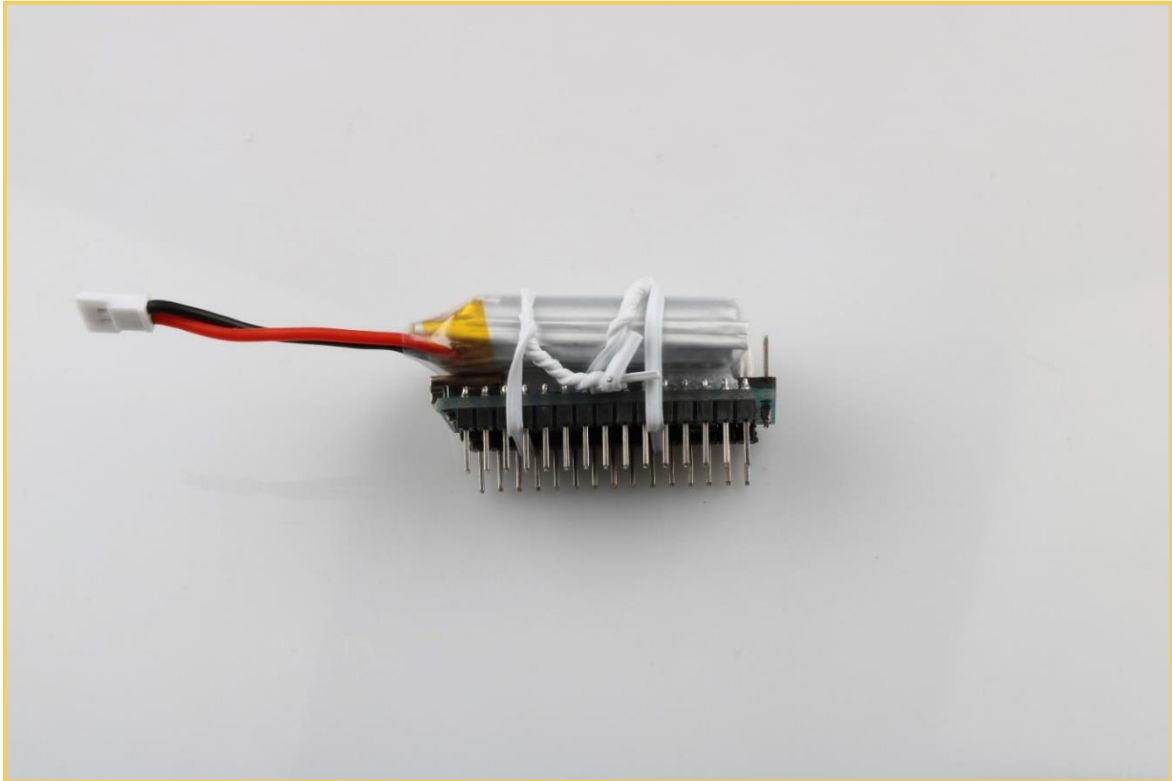
Figure-36

Then put the battery flat on the foam cardboard, as shown in Figure 37.



Figure-37

Tie the battery and the NANO board together with a cable tie, as shown in Figure38.



Servo-38

When finished, reinstall the Nano board onto the servo, as shown in Figure 39.

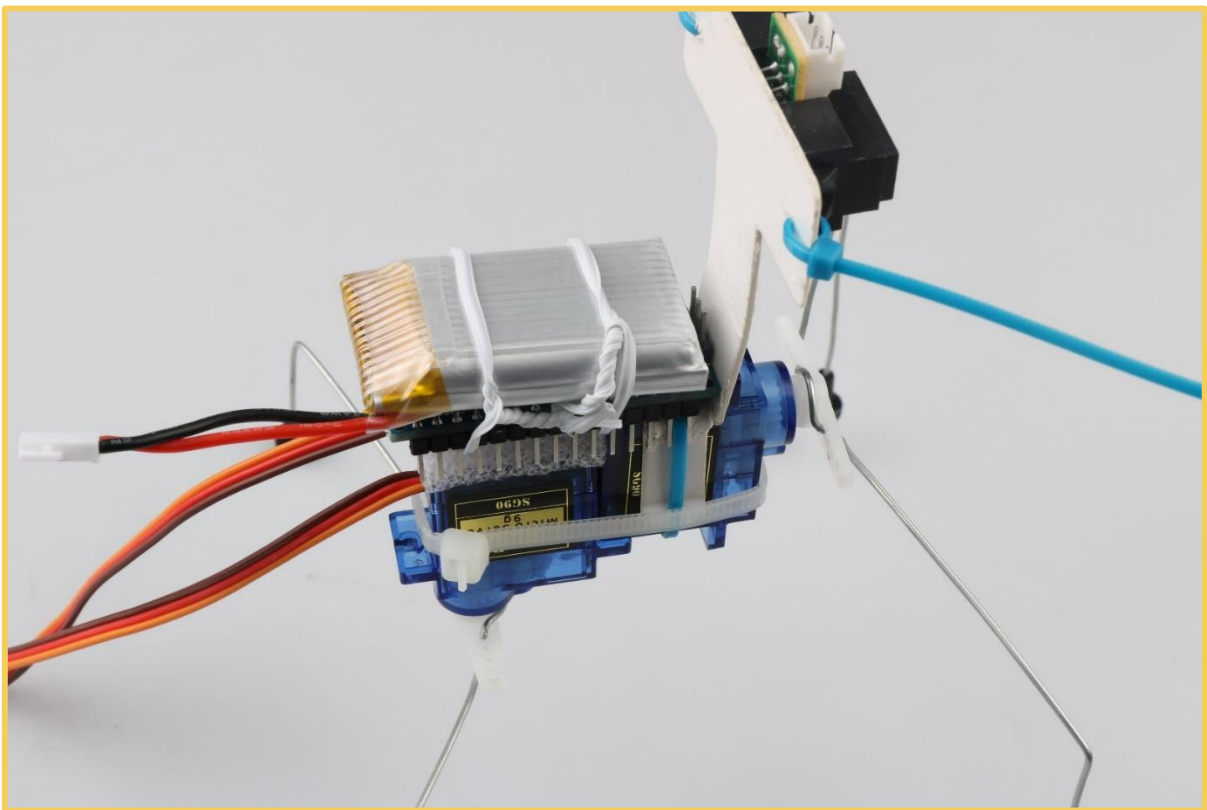


Figure-39

3.7 Wiring of Circuit

Funduino Nano	Front Servo
GND	-
+5v	+
13	data
Funduino Nano	Infrared Sensor
GND	-
+5v	+
12	data
Funduino Nano	Battery
GND	-
+5v	+

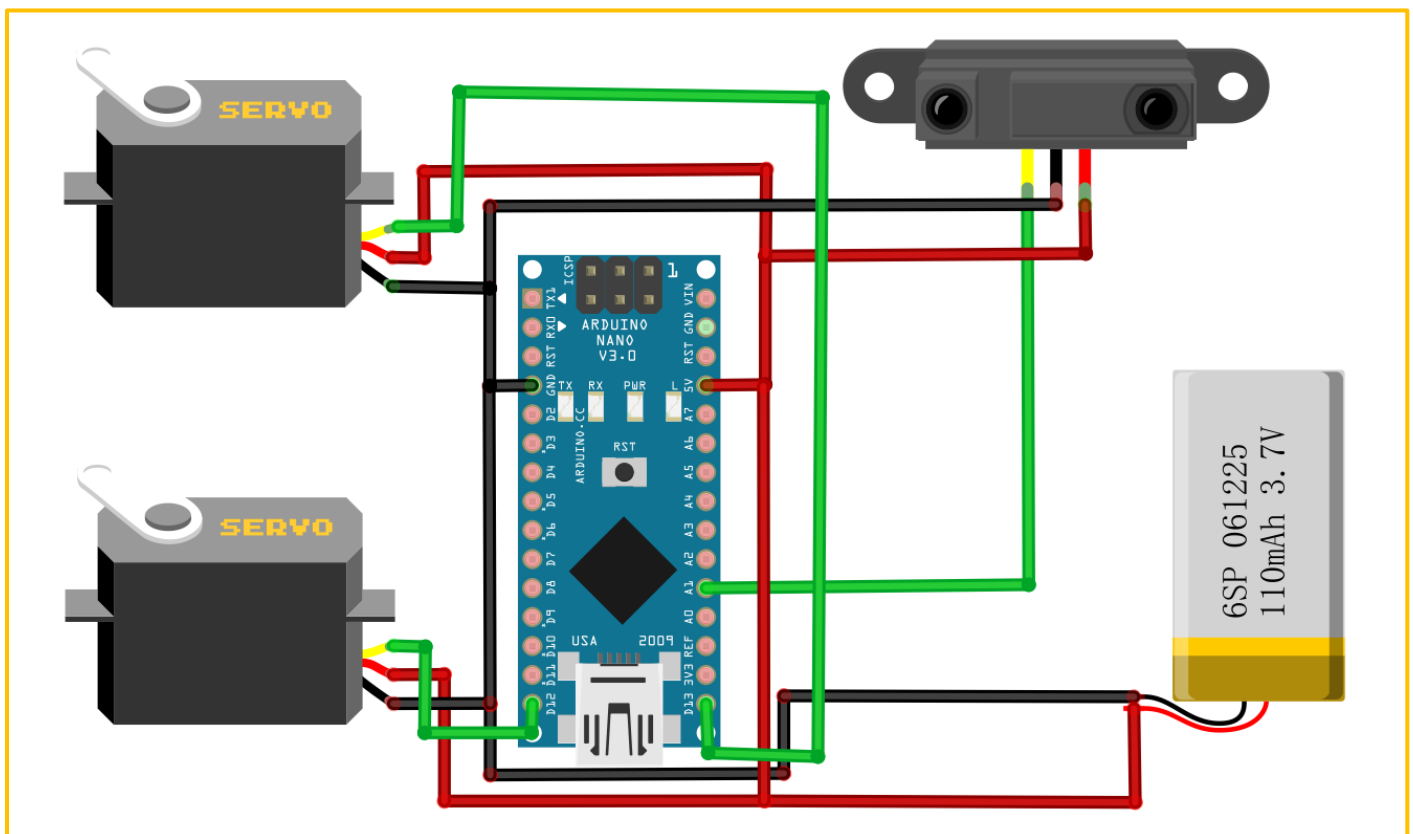
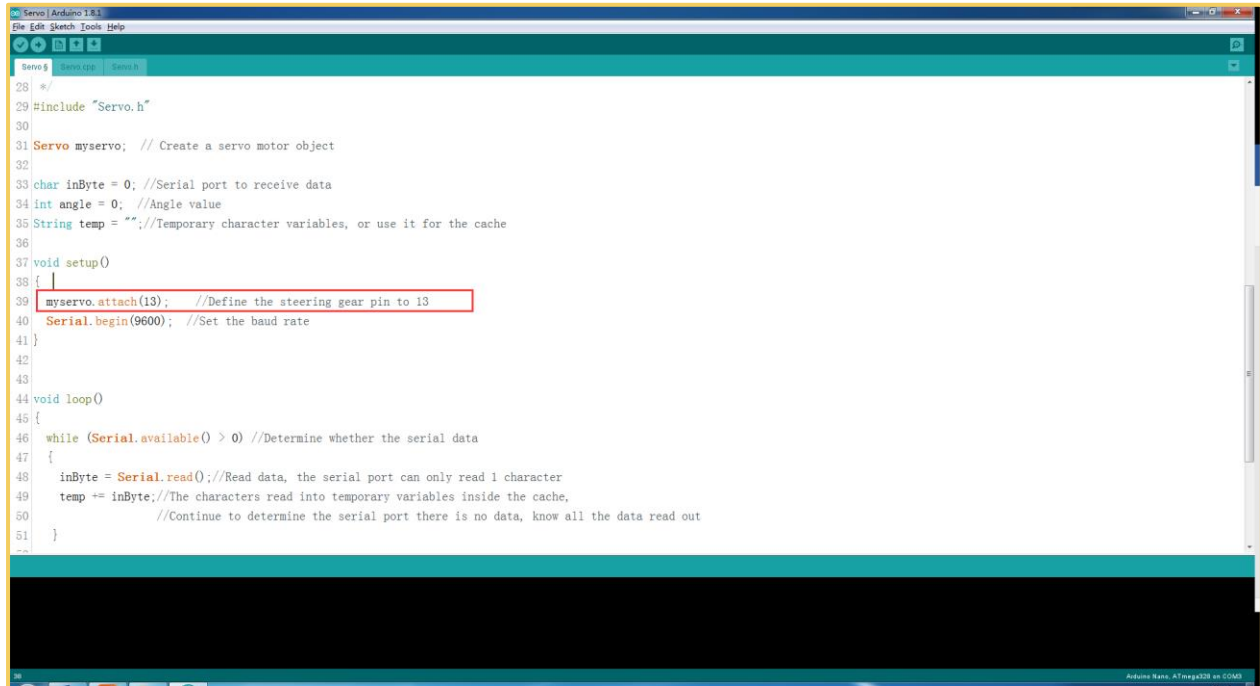


Figure-40 Circuit connection diagram

3.8 Adjusting servo angle

Open the matching program, the folder is called "**Servo Motor\ServoTest\ ServoTest**", and download the program to the NANO board, ensuring that the initial status of both servos is 90°. Open the program as shown in Figure 41.



```

28 /*
29 #include "Servo.h"
30
31 Servo myservo; // Create a servo motor object
32
33 char inByte = 0; //Serial port to receive data
34 int angle = 0; //Angle value
35 String temp = ""; //Temporary character variables, or use it for the cache
36
37 void setup()
38 {
39   myservo.attach(13); //Define the steering gear pin to 13
40   Serial.begin(9600); //Set the baud rate
41 }
42
43
44 void loop()
45 {
46   while (Serial.available() > 0) //Determine whether the serial data
47   {
48     inByte = Serial.read(); //Read data, the serial port can only read 1 character
49     temp += inByte; //The characters read into temporary variables inside the cache,
50     //Continue to determine the serial port there is no data, know all the data read out
51   }
52 }

```

Figure-41

When finished downloading, open the IDE serial assistant and enter any value from 0-180°. Then press Enter and the servo will rotate the corresponding angle.

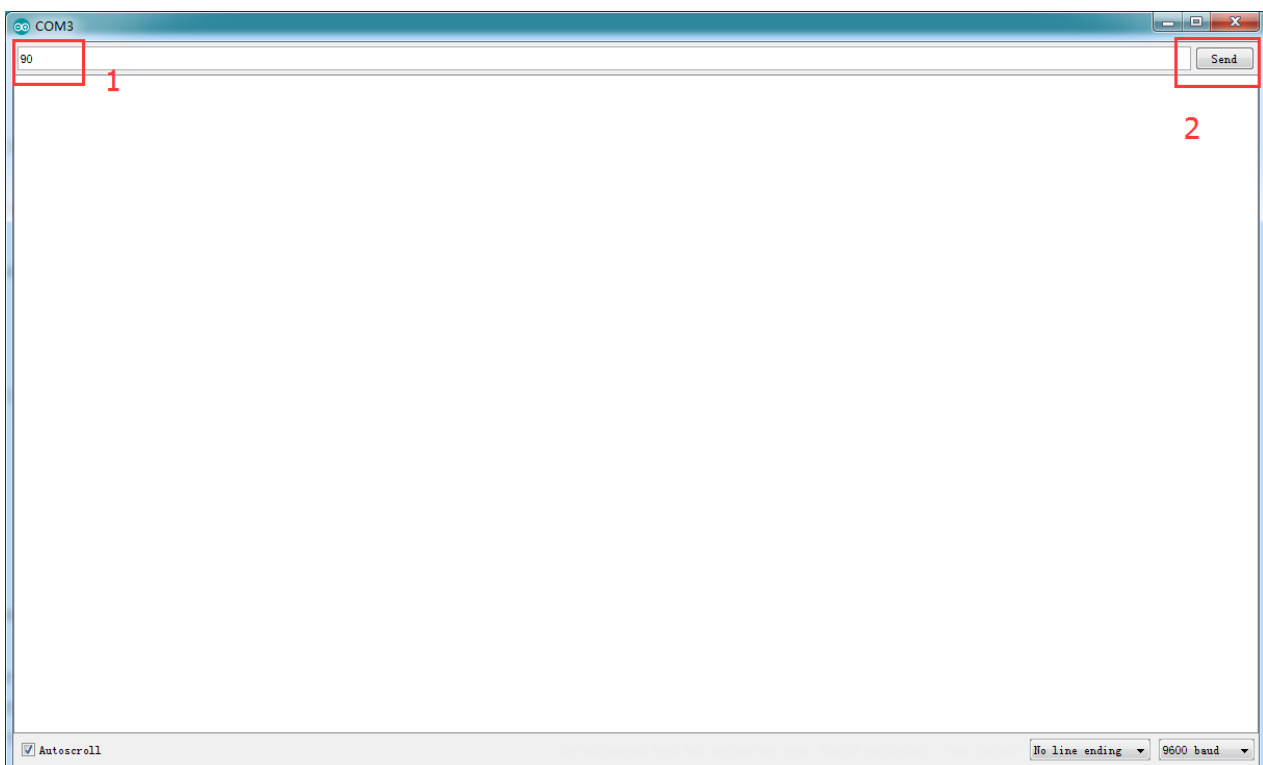


Figure-42

If your mechanical foot is installed correctly, when the servo is rotated to 90°, the mechanical foot should be in the horizontal state, as shown in Figure 43. If it is not in the horizontal state, please rotate the servo to 90° and then adjust the mechanical foot.

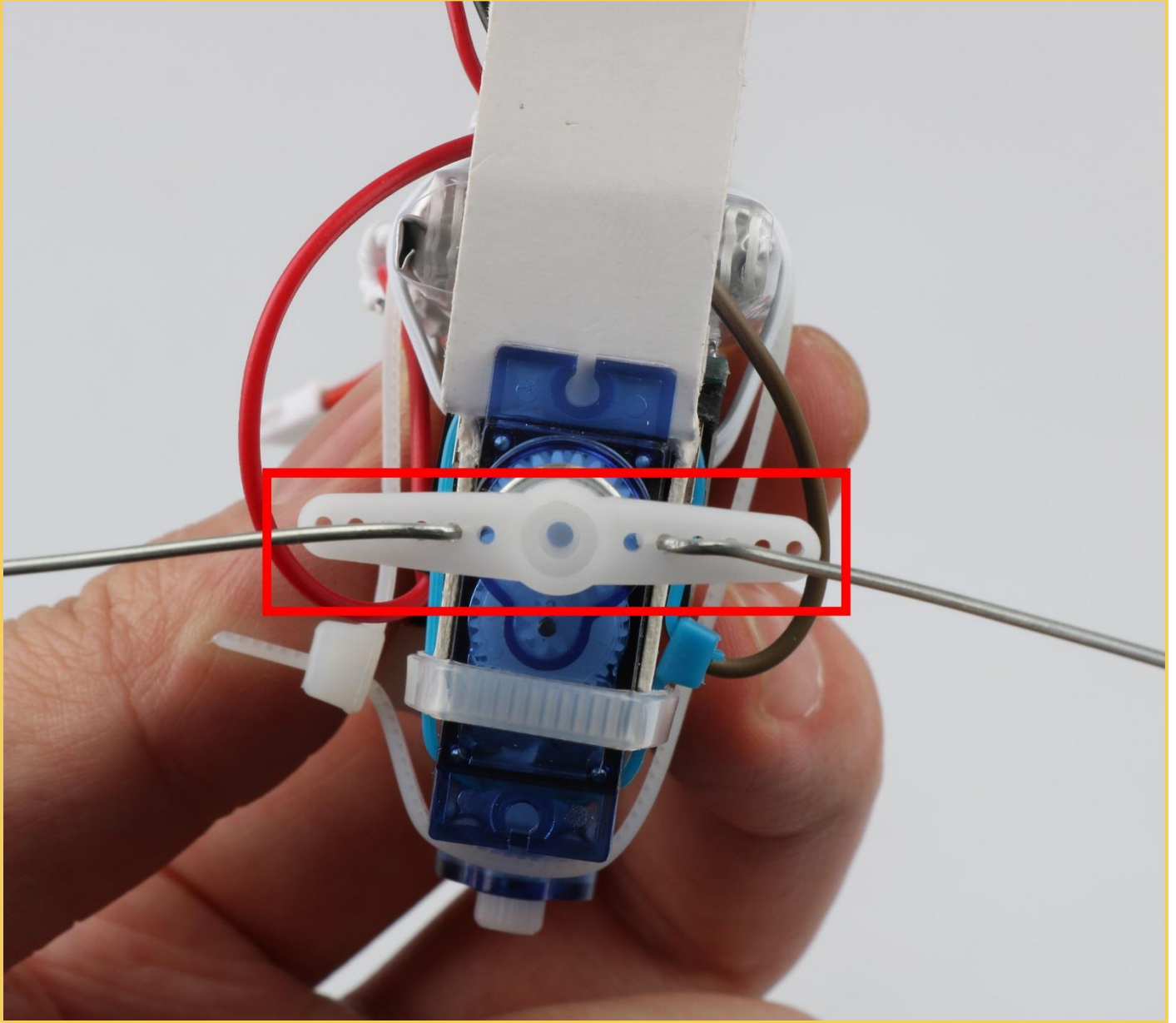


Figure-43

After adjusting the angle, you can completely fix the forefoot

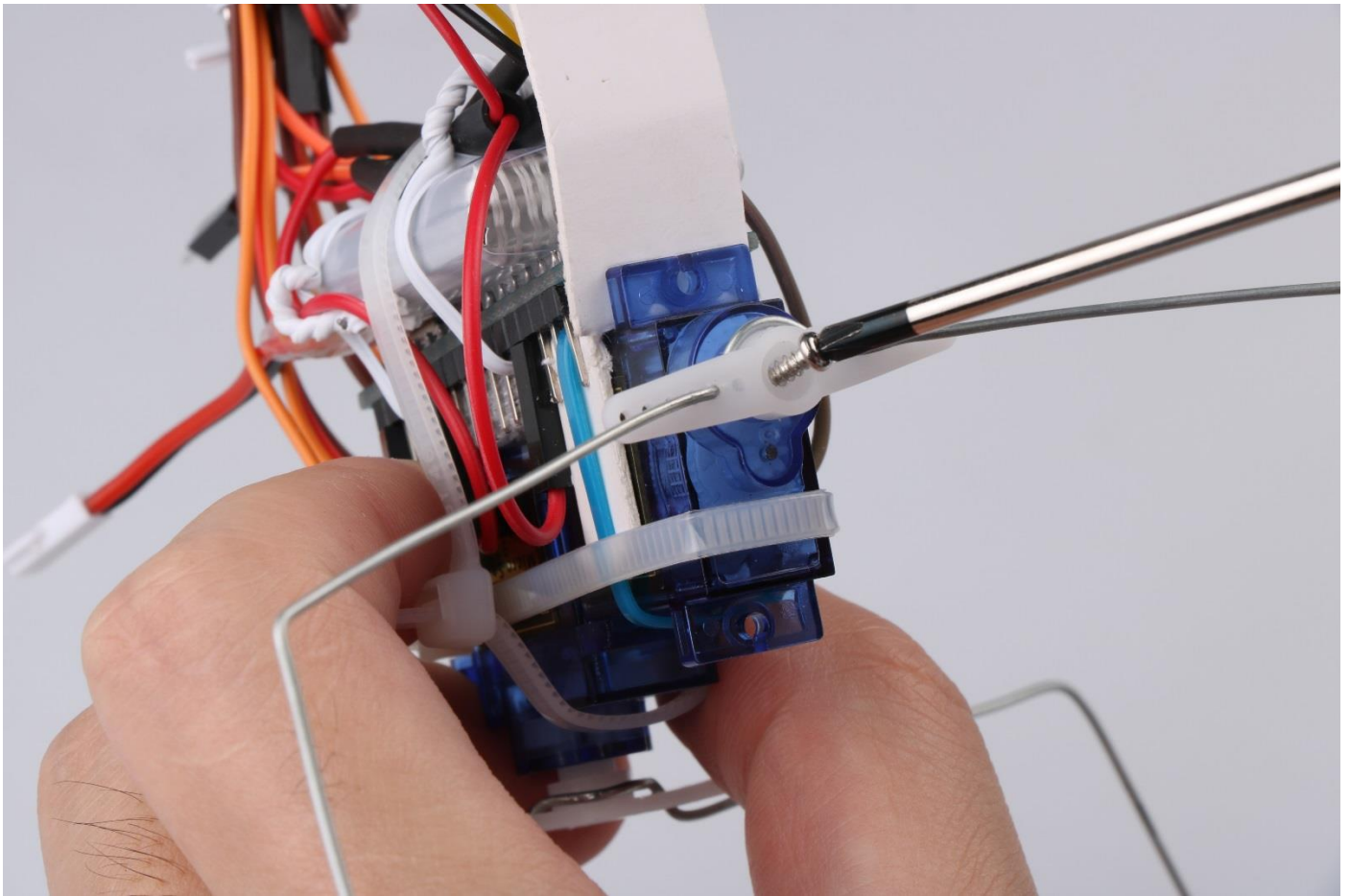


Figure-44

Finally,a lovely worm robot was born, download the program to the NANO board, connect to power, and start a new trip .

Note: If the robot walks slowly or the posture is very unstable, the wire bending angle and height, direction and so on need to be properly adjusted.

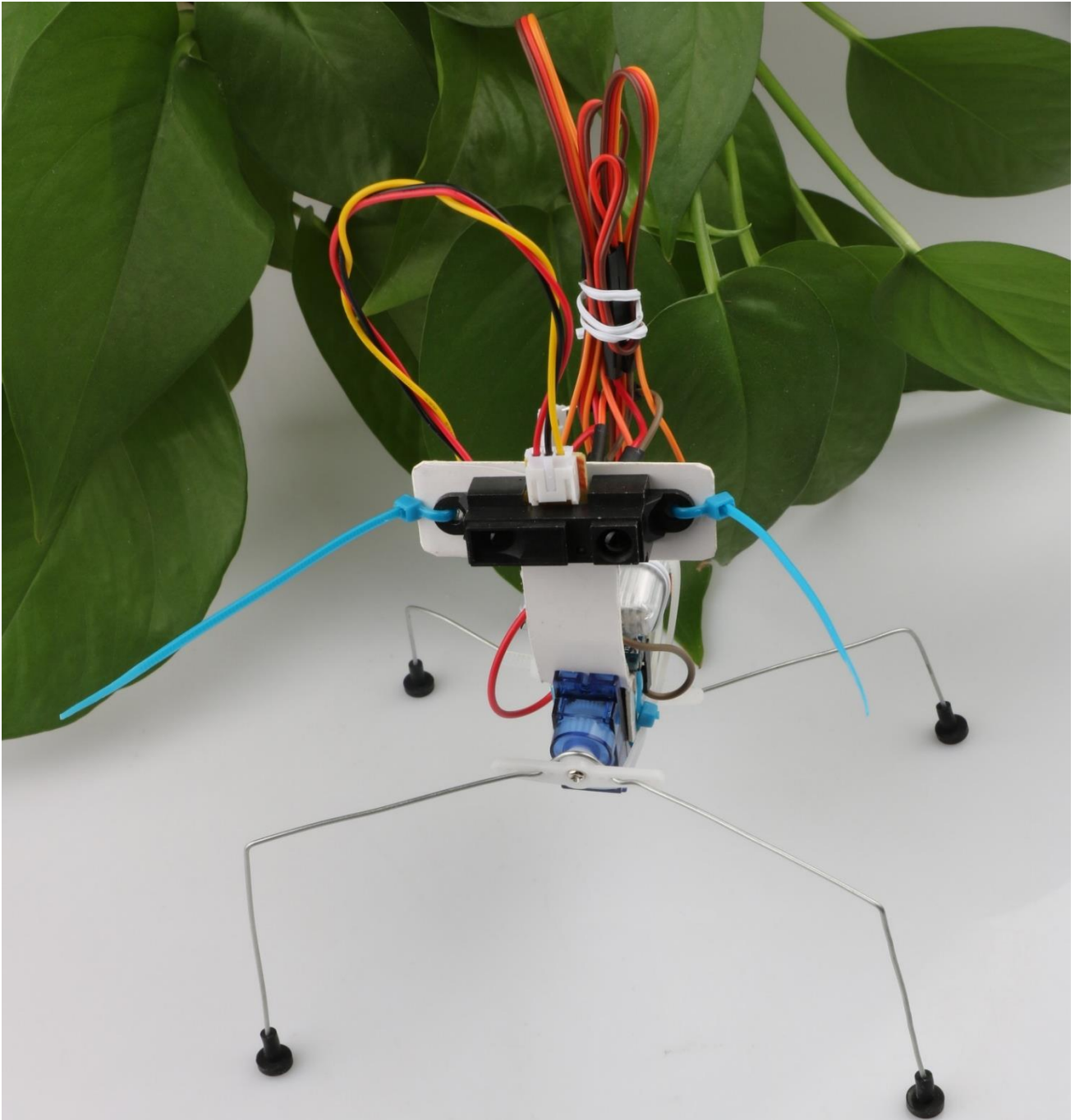
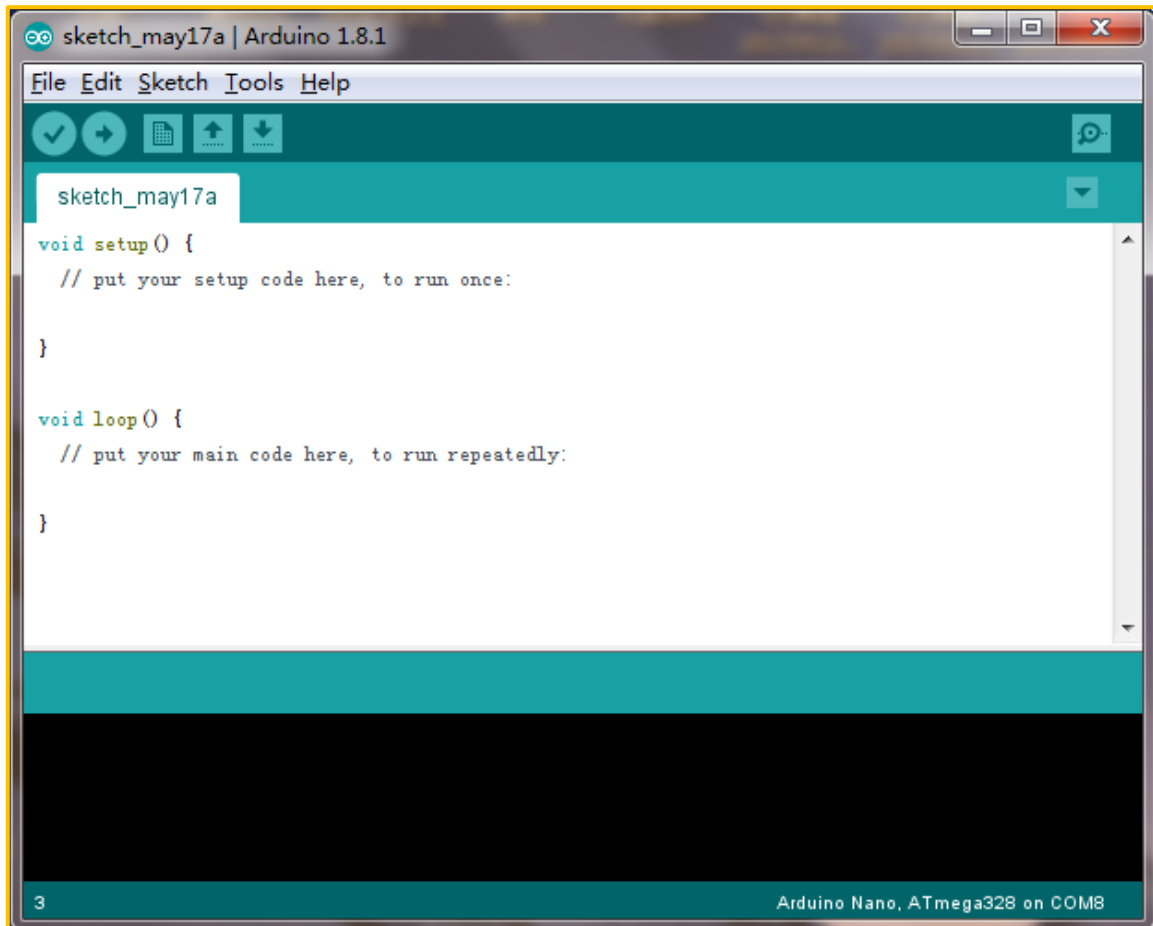


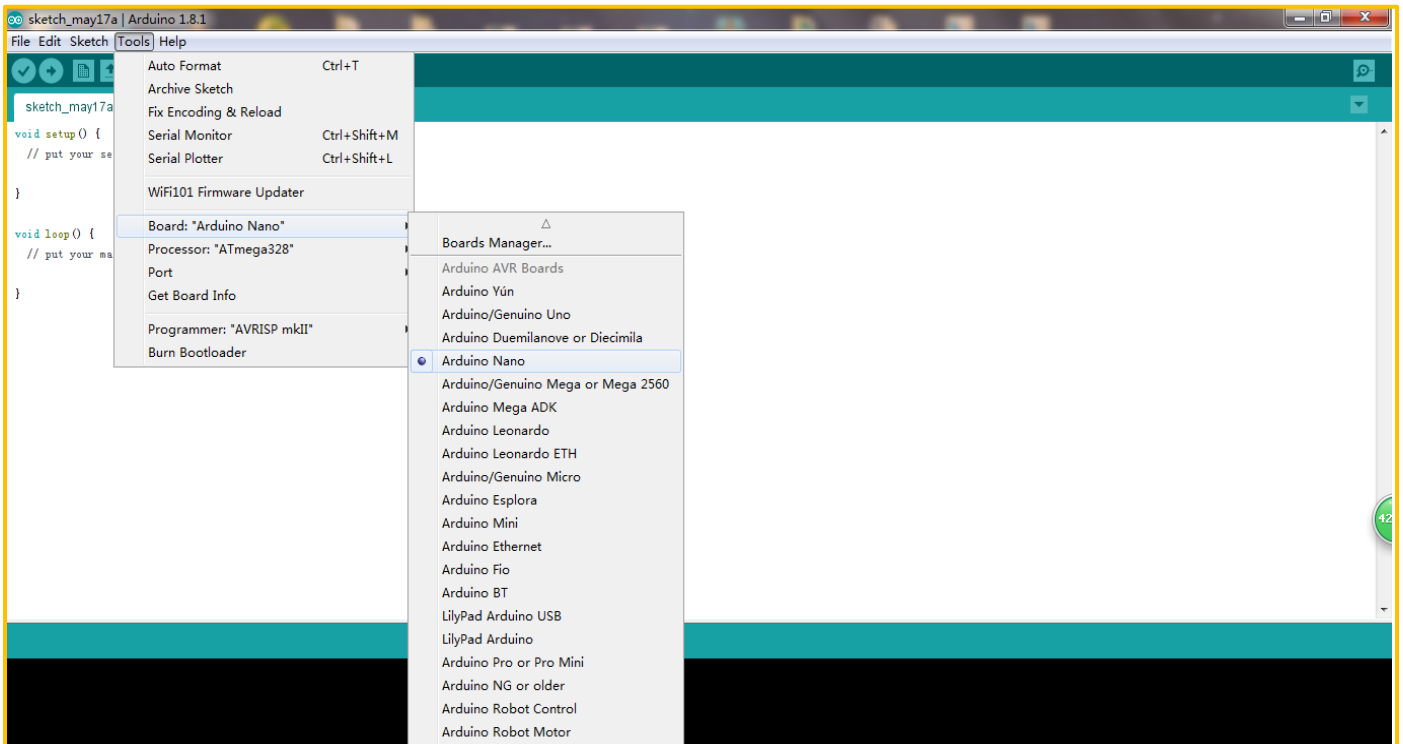
Figure-45

Programming

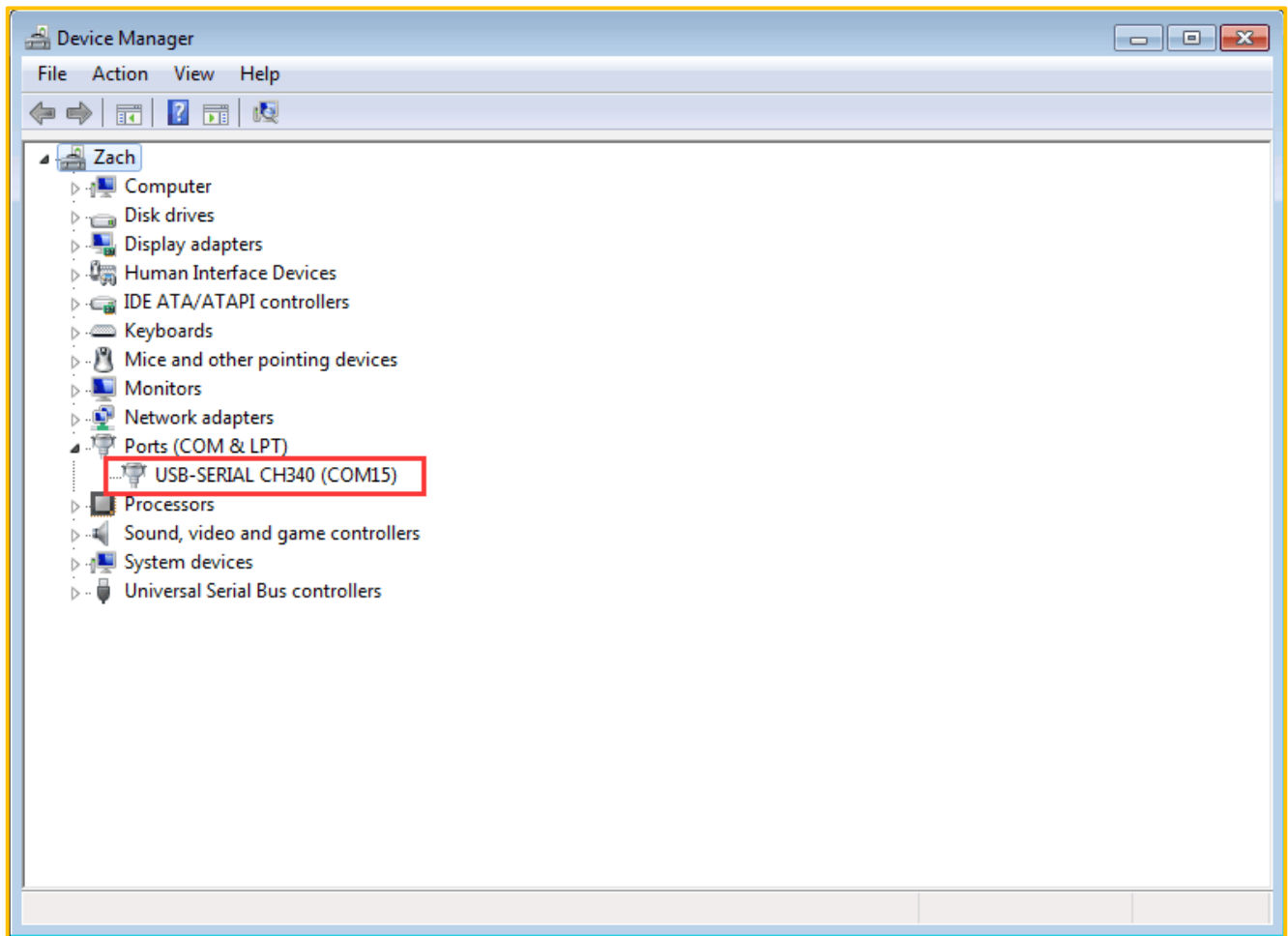
Step1: Arduino IDE, a programming software, is required to be installed into your computer. If already installed, please skip this step. The software can be obtained from <https://www.arduino.cc/en/Main/Software>



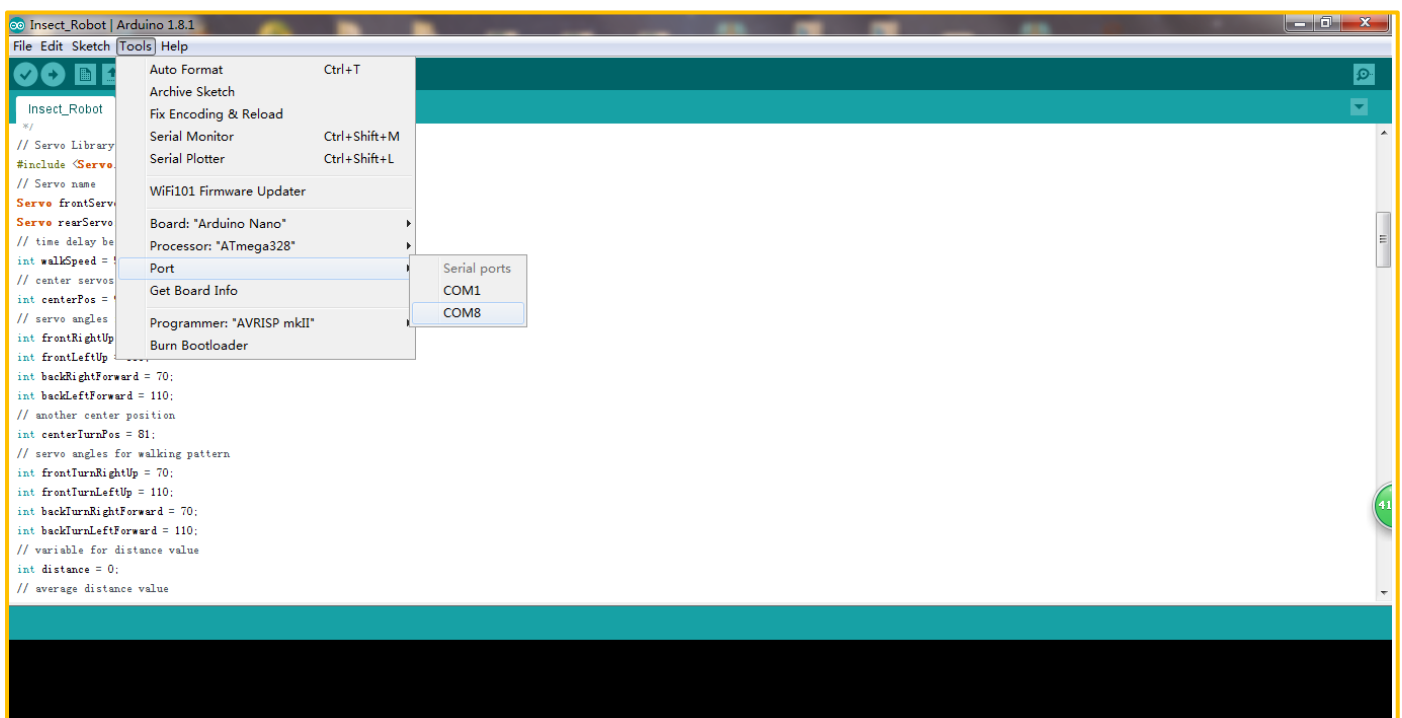
Step2: Choose “Arduino Nano” in Tools/Board




Step3: In the device manager of the Windows control panel, locate the port corresponding to Arduino Nano, such as COM15.

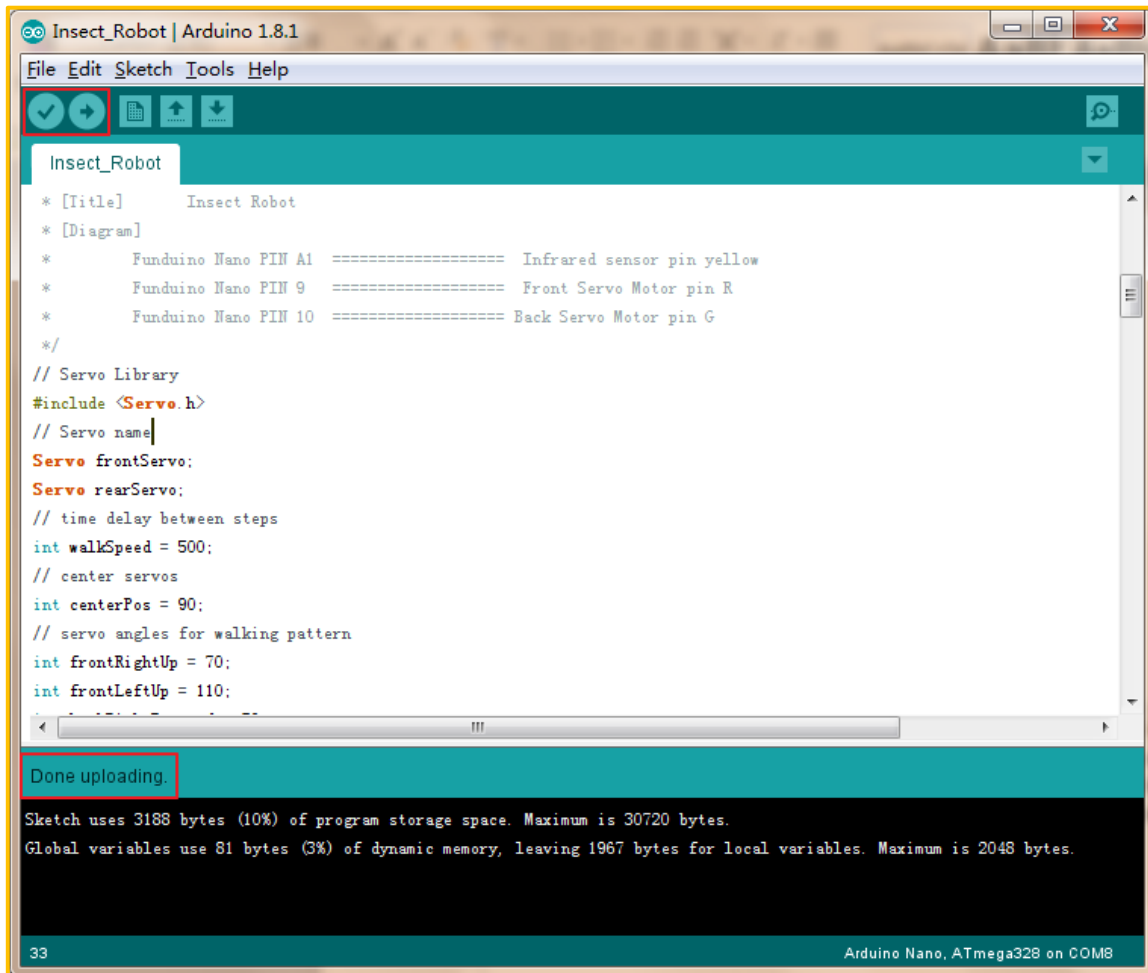


In the following Arduino programming software Tools/Port, selecting the port number you have found, such as COM8.



Step4: Open the *Insect_Robot.ino* program file, and then click  to check the program.

Finally, click on  to upload the program into the controller. The white LED on the controller blinks for about 2 seconds, and the lower end of the window shows Done Uploading, indicating that the program has been uploaded successfully.



Step5: Now the robot has been already completed, and the program has been burned into controller. As long as we connect the +3.7V pin of the battery to the extended +5V pin on Funduino Nano, the insect robot is ready to take his first step.

Code

```
// Servo Library
#include <Servo.h>
// Servo name
Servo frontServo;
Servo rearServo;
// time delay between steps
int walkSpeed = 500;
// center servos
int centerPos = 90;
// servo angles for walking pattern
int frontRightUp = 70;
int frontLeftUp = 110;
int backRightForward = 70;
int backLeftForward = 110;
// another center position
int centerTurnPos = 81;
// servo angles for walking pattern
int frontTurnRightUp = 70;
int frontTurnLeftUp = 110;
int backTurnRightForward = 70;
int backTurnLeftForward = 110;
// variable for distance value
int distance = 0;
// average distance value
int distanceCheck = 0;
// Array for distance values
int collectDistance[5];
// Variables for counters
int i;
int f;
int r;
// assign analog pin A1
```

```
int sensorPin = A1;
// distance value for danger close. Bigger values are greater distance and
// smaller values are lower distance
int dangerDistance = 350;
/* Setup function */
void setup()
{
// attach servos
  frontServo.attach(12);
  rearServo.attach(13);
  // assign sensor
  pinMode(sensorPin, INPUT);
  // center servos
  frontServo.write(centerPos);
  rearServo.write(centerPos);
  // wait 3 seconds for start walking
  delay(3000);
  //Serial.begin(9600); // serial data setup
}
/* distance check function */
void scan()
{
  // read 5 distance values
    for (i = 0; i < 5; i = i + 1) {
      distanceCheck = analogRead(sensorPin);
      collectDistance[i] = distanceCheck;
      // serial output for testing
        //Serial.print (i);
        //Serial.print(" = ");
      //Serial.println(collectDistance[i]);
    }
    // checksum of the 5 distance values for getting an average value.
    This will prevent the robot to change behavior by reading one wrong value
```

```
distance =
(collectDistance[0]+collectDistance[1]+collectDistance[2]+collectDistance[3]+collectDistance[4])/5;
    delay(20);
}
// walk forward
void moveForward()
{
    // loop for the servo angels to smoothen the steps
    for (f = 0; f < 39; f++){
        frontRightUp++;
        backLeftForward--;
        frontServo.write(frontRightUp);
        rearServo.write(backLeftForward);
        delay(10);
    }
    // loop for the servo angels to smoothen the steps
    for (r = 0; r < 39; r++){
        frontRightUp--;
        backLeftForward++;
        frontServo.write(frontRightUp);
        rearServo.write(backLeftForward);
        delay(10);
    }
}
// walk backwards to the left
void moveBackRight()
{
    frontServo.write(frontRightUp);
    rearServo.write(backRightForward-6);
    delay(110);
    frontServo.write(centerPos);
    rearServo.write(centerPos-6);
}
```



```
delay(80);
frontServo.write(frontLeftUp+9);
rearServo.write(backLeftForward-6);
delay(110);
frontServo.write(centerPos);
rearServo.write(centerPos);
delay(80);
}
// walk forward to the left
void moveTurnLeft()
{
frontServo.write(frontTurnRightUp);
rearServo.write(backTurnLeftForward);
delay(110);
frontServo.write(centerTurnPos);
rearServo.write(centerTurnPos);
delay(80);
frontServo.write(frontTurnLeftUp);
rearServo.write(backTurnRightForward);
delay(110);
frontServo.write(centerTurnPos);
rearServo.write(centerTurnPos);
delay(80);
}
// blink LED. This function can be called in any situation you want. Just
add led(); in the code where you want to blink the LED.
void led(){
    // loop for the LED to blink it 5 times for 0.05 sec on and 0.1 sec off
    for(int l=0; l<=5; l++) {
        digitalWrite(13, HIGH);
        delay(50);
        digitalWrite(13, LOW);
        delay(100);
    }
}
```

```
// that's the loop. This is repeatedly called as long as the robot is powered
on
void loop()
{
    // call function for checking the distance
    scan();
    //Serial.println(distance);
    if (distance > 1){ // filters out the zero readings
        // an obstacle is being detected
        if (distance > dangerDistance) {
            // LED at Pin 13 (standard) blinks 5x
            led();
            // 4 steps backward left
            for(int i=0; i<=3; i++) {
                moveBackRight();
                delay(walkSpeed);
            }
            // 4 steps forward left
            for(int i=0; i<=3; i++) {
                moveTurnLeft();
                delay(walkSpeed);
            }
        } else {
            // all clear, no obstacle detected. Just walk forward
            moveForward();
            delay(walkSpeed/100);
        }
    }
}
```