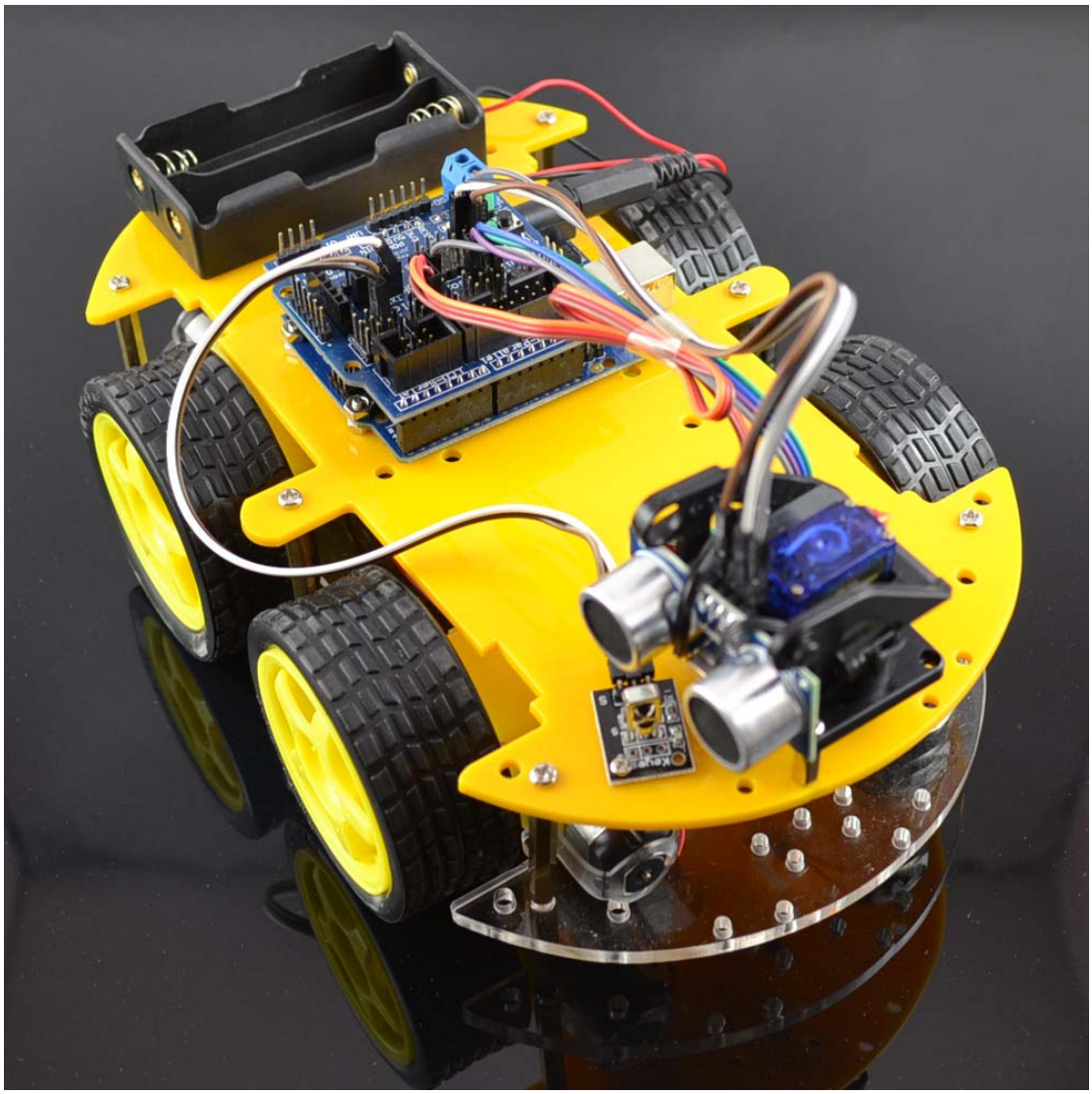
Translated version of Arduino

Arduino Bluetooth utility vehicle manual



One. Brief introduction

ARDUINO Bluetooth utility vehicle is a single-chip learning application development system to arduino microcontroller family atmega-328 core. Complete the hunt, obstacle avoidance, infrared remote control and Bluetooth remote control functions. Package contains a number of interesting programs, and Expansion external circuit module, thereby increasing the car's functionality. designed to allow users in learning ARDUINO microcontroller can from boring theoretical knowledge, the ability to acquire SCM system developed in the play.

Two. Parameters:

A motor parameters: Voltage Range: 6-9V, reduction ratio: 1 to 48

2 control motor selection L298N driver module, microcontroller real isolation.

3 Three groups hunt module detects black and white lines, higher accuracy, can also be used

with anti-drop control.

4 Infrared remote communication module, consisting of the smart car remote control system.

5 ultrasonic modules, car obstacle avoidance system.

5 Bluetooth wireless module can be paired Bluetooth phone remote control robot.

6 can access the external voltage 7 ~ 12V. And can be equipped with a variety of sensor modules to achieve a variety of functions, depending on your imagination.

Three. Experimental Course Introduction

1. L298N motor driver board applications
2. Smart car tracking
3. Ultrasonic obstacle avoidance Smart Car
4. Infrared remote control Smart Car
5. Arduino Bluetooth remote control programmable smart car
6. Four and one (obstacle avoidance hunt infrared remote Bluetooth remote) multifunction program

Four. List

1 gear motor 4
2 high-quality tires 4
3 motor fixing member 4
4.100 * 213 * 5MM Plexiglass plate 1
5.100 * 213 * 5MM Plexiglass plate 1
6.L298N motor driver board 1
7.ARDUINO UNO328 control panel 1
8.ARDUINO sensor expansion board V5 *1
9 PTZ 1
10. Servo 1
11 Ultrasonic Module 1
12 Three groups hunt module
13 IR receiver sensor
14 MCU remote control
16.18650 battery box a
Section 17.18650 battery 2
18.18650 charger 1
19 Bluetooth module 1
20 DuPont line 30
21.1 m A long USB cable
22. Pillars 35MM Long 6 20MM 3 6MM 6
23.3MM Several screw nut

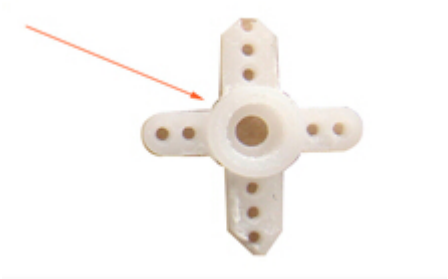
V. head servo installation diagram

The installation Diagram of pan servo ultrasound

Install for ultrasonic, take out the cross, and cut the long legs

take out the cross colloid from servo accessories box as the pic shows below.

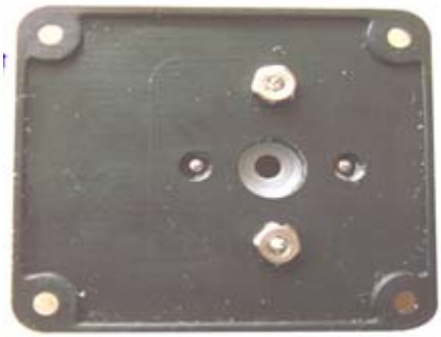
Change the cross into same width and four sided with same length as the pic shows below .



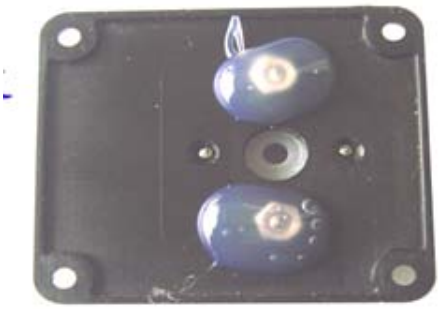
install 2*8 MM screw and 2*5MM Screw in the second hole of the cross and at the bottom of pan as the pic shows below.

fix it below the rotation stage

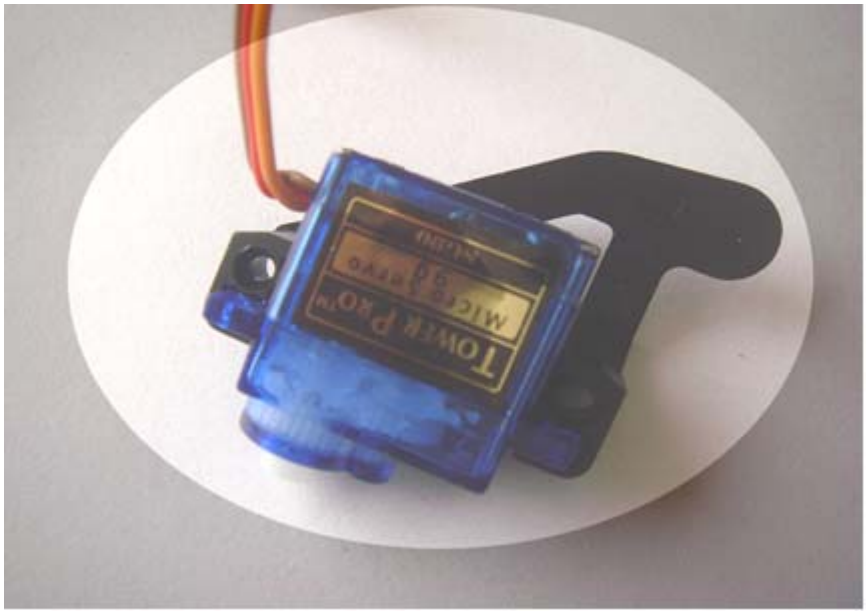
First fill the nut to the 2*8MM screw of Pan’s bottom as the pic shows below.



Then dispenses hot melt adhesive and tighten the screw as the pic shows below.



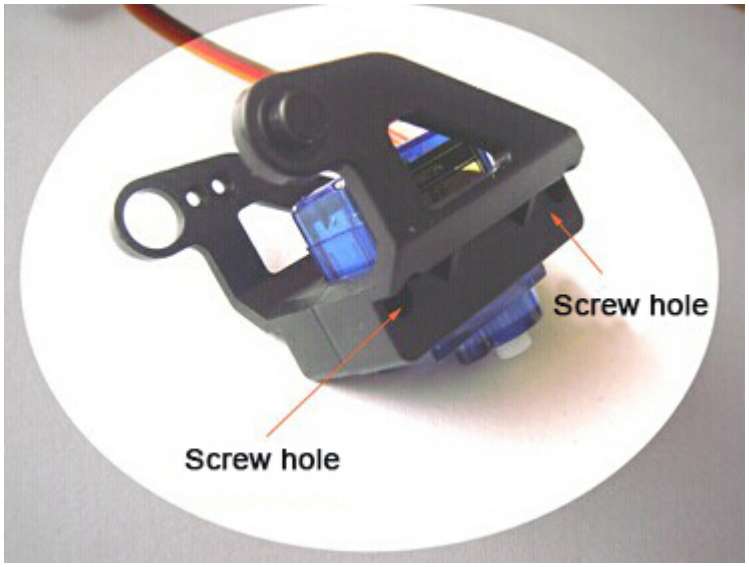
Install two boundaries of Pan in the servo as the pic shows below.



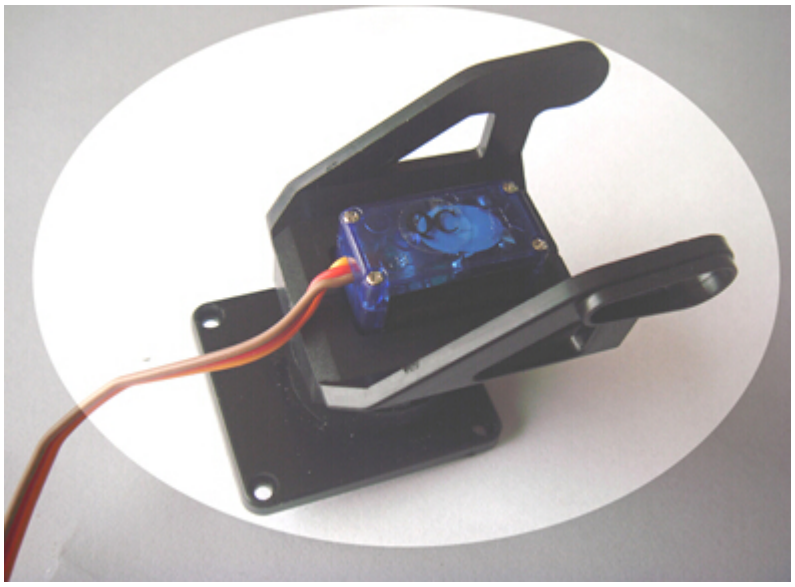
install the servo motor

After installing the boundaries, fill and tighten the screw as the pic shows below.

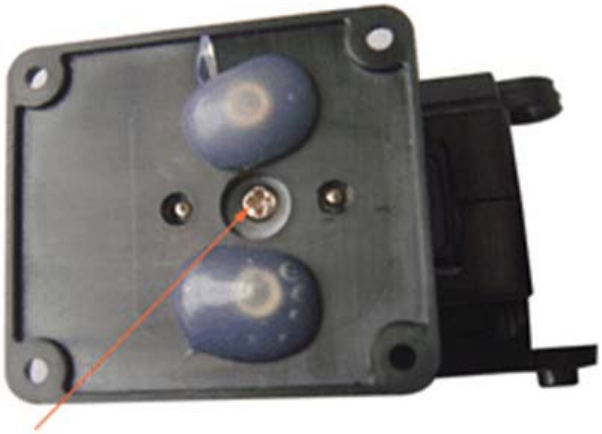
After installing the boundaries, fill and tighten the screw as the pic shows below.



insert the installed servo in the fixed cross colloid and adjust the direction as the pic shows below.



take out 2*6MM screw from the motor components box and install the screw into the mounting hole of the motor as the pic shows below.

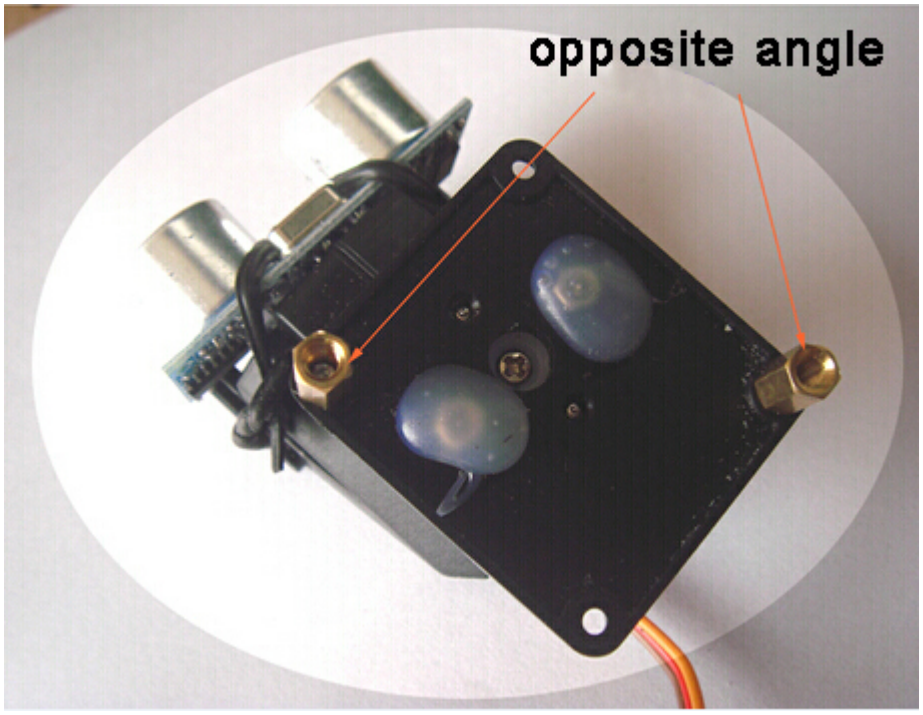


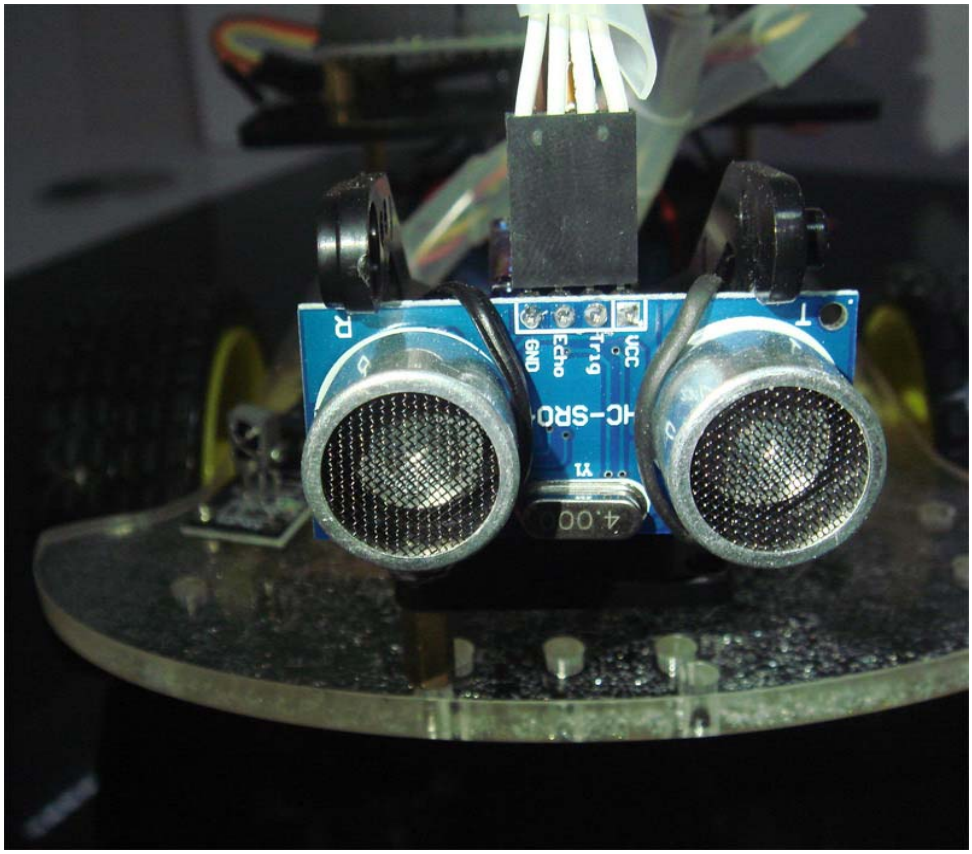
the mounting hole of the motor

Fixing the ultrasound module in the front-end of Pan with armor tapes as the pic shows below.

install the ultrasonic sensor

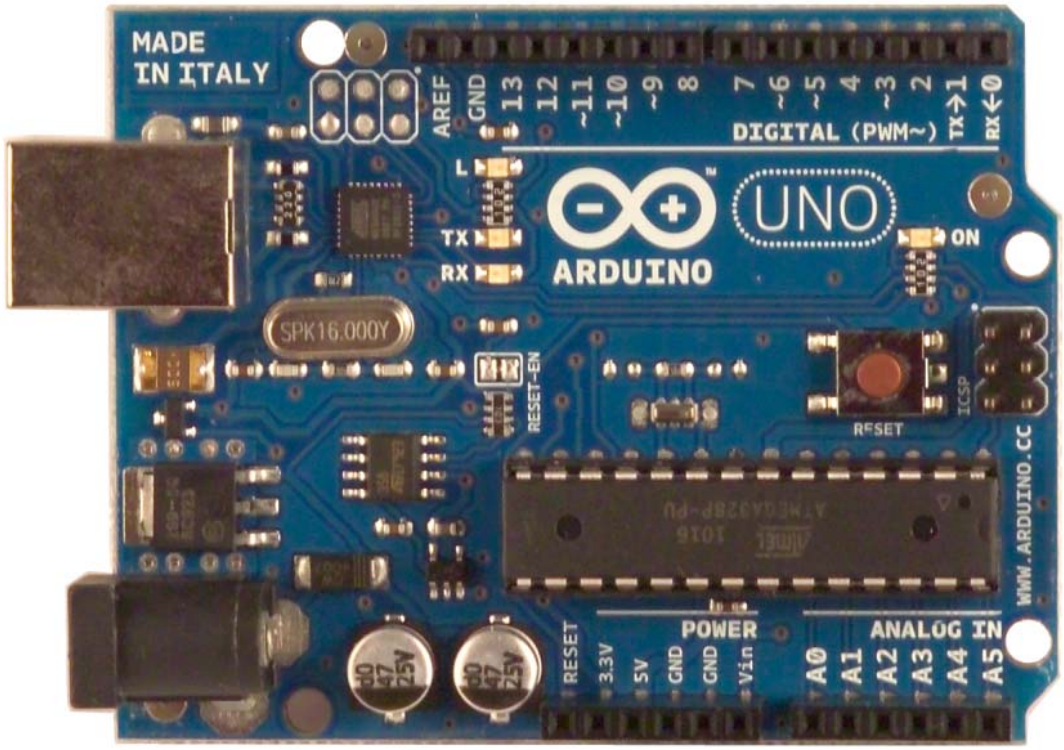
Install the 6MM cooper standoff in the mounting hole of Pan bottom as the pic shows below.



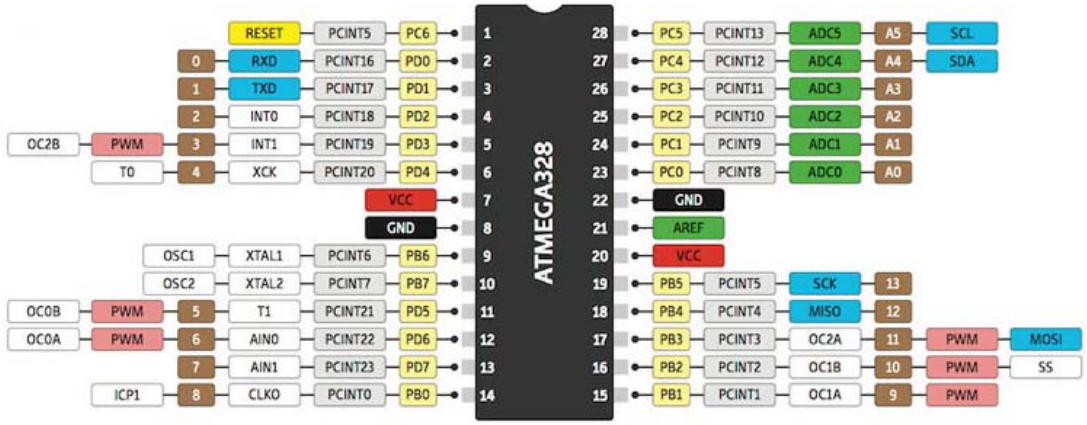


Six. Arduino microcontroller use

1 Introduction



Arduino is an open-source from the Italian project hardware platform that includes a with a simple I / O functions of a circuit board and a set of application development environment software. Arduino can be used to develop interactive products, such as it can be read by a large number of switches and sensor signals, and can control lights, motors, and other kinds of physical devices; Arduino can also develop peripheral devices connected to the PC, can and software on the PC to communicate runtime. Arduino hardware circuit board can be self-welding assembly can also be purchased already assembled modules, and program development environment software you can download and use for free from the Internet.



We look at how to define the Arduino team:

Arduino is an open-source electronics prototyping platform with a flexible, easy to use hardware and software. Arduino is designed for designers, arts and crafts workers, hobbyists and people interested in the development of interactive installations or interactive development environment designed for.

Arduino can receive input signals from various sensors to detect the operating environment, and by controlling the light source, and the other drive motor to affect their surroundings. Microcontroller Arduino board programming using programming language (based on Wiring) and the Arduino development environment (based in Processing). Arduino can run independently,

can also run on a computer with software (for example, Flash, Processing, MaxMSP) to communicate.

Arduino hardware board can self-welding assembly can also be purchased already assembled, the software is free to download from the Arduino website. You can get an open source license hardware reference design (CAD file), and the freedom to modify it to suit your needs.

Arduino is still a little fuzzy definitions, which is the Arduino advantage. Arduino is connected to a variety of tasks that people adhesives. Arduino give the most precise definition is best described using some examples.

You want coffee brewed, the coffee pot on the issue of "creak" sound to remind you with something?

If you want to have a new mail message, the phone will alert notifies you do?

Want a glittering pile toy?

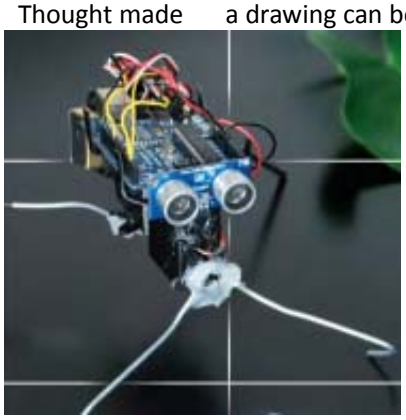
Want to have a voice and drinks distribution function Energy X Professor steam punk style wheelchair?

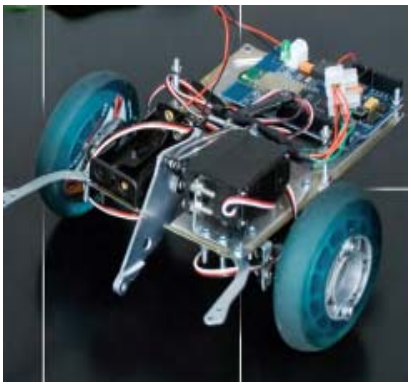
Want a keyboard shortcut can be carried out experimental tests buzzer it?

Want for your son made a "Metroid" arm cannon right?

Homemade want a heart rate monitor, record each time the cyclists deposited into a memory card?

Thought made a drawing can be on the ground, the robot can ride in the snow right?





Arduino think that you can achieve.

For understand electronics or microcontrollers people, this sounds very cool, very interesting, and will want to join this club. This is something kids want to make, you can even fool some of their knowledge in this process school. These projects generated in the science fiction story, these small devices appear in the log. Its common is that these are fantasies, some of the stuff of dreams. But now, these wonderful ideas really are achieved in a timely manner you are not an engineer, but also completely homemade.

This is a big deal, because engineers are often designed for other engineers to develop a platform, rather than as an artist, freak, or young children in order to share an idea and simply connect things together. Arduino team than by the "hard core" of electronic engineers, but by designers, teachers, artists, and I know all the "technical hippies" component (hippies here is a compliment, I hope I did not offend them). Arduino's main base in Italy, every year I see people trying to find articles about Italy "own Google", and in fact they already have, and that is Arduino, they have yet to realize it.

See examples of Arduino projects, you will find these electronics makers are "what is" more interested in, rather than production methods. These enthusiasts often expressed Arduino Arduino does not teach basic electronics, "Bah, could this be true of electronic products ah," they said, "too easy!" Yes, indeed. If you are an artist or designer, not using Arduino in case, you want a light emitting diode flashes, or motor rotation, then I wish you good luck. Of course, if you are willing to spend money and take your electronics technology heavy textbooks to show something, which is not a bad idea. But for there to other people, but they want to use light-emitting diode to decorate Burning Man clothing only.

For some traditional microcontroller Arduino community is how to treat this issue, I think the best example is the AVR Freaks, the official website focuses on AVR processor (also used Arduino).

You might want to AVR Freaks community favorite Arduino, because Arduino AVR microcontrollers can be brought to the public. However, many people do not like all of the site's wacky contraption made of these non-engineers because they will destroy their hierarchy. To quote my favorite (and I hope these words printed on the T-shirt).

"Arduino programming language that children can understand, once hooked" - ArnoldB, AVRfreaks website

In fact, this was the wrong attitude to promote the Arduino fans to create their own communities, helping to build an eclectic Arduino, rather high above the community.

Arduino is simple, but not too simple. It revolves around these ideas to create that students use Arduino to achieve the purpose of: receiving sensor signals, to get some code, and then use these signals and codes. Perhaps not even write code, you can begin after that they cut and paste code. Arduino is a thermal adhesive, rather than precise welding. So no one will be cut off one hand, there will not be labs were destroyed. One Arduino team members will teach arts and crafts people and designers how to use. Every day, Arduino constantly establish and improve learning, professors, and shared code project. The arts and crafts people and designers to use and modify the Processing language systems are Macs. (Processing is Arduino's Big Brother)

Speaking here, Arduino is like a passion, no boundaries, artistic atmosphere of the rally. This is the Arduino become a "do it yourself" successful model of reason? Not only that, we come to understand more specific information.

Library - simple tasks, complex tasks easy to get

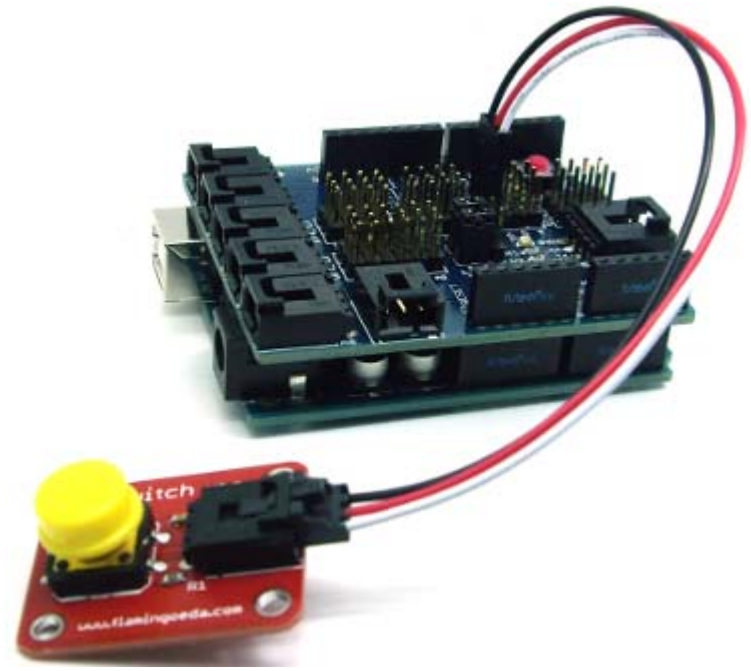
A large package library is used to perform complex tasks, such as writing to the SD memory card, write LCD driver, parsing GPS. There are also some libraries used to do simple things, such as turning the pin or key debounce. If there are 10 chips, we have to install the UART code written 10 times, and frankly, it annoying. If you call Serial.begin (9600) function to handle the register data, then it is much easier.

Lightweight, run directly on the underlying hardware On

Use a certified, easy to understand compilers (We can even say that is the AVR avr-gcc compiler default or standard), the code can be run directly on the underlying hardware On. NET languages compile manner and BASIC language is different. The compiler runs fast, small size, light weight, and you can use HEX (hexadecimal) file for the bulk of the new chip to be programmed.

Sensor

Arduino really take off because it enables analog input into digital input, in other words, you can light, temperature, sound, or any already on the market of low-cost sensor signal input, Arduino can be identified. For digital sensors, Arduino supports SPI (high-speed synchronous serial port) and I 2C The bus. This feature covers 99% of the sensors on the market. Other development platform is not easy to achieve - think if a Beagleboard (great product) and Arduino tied together, just to get the data of the sensor, it is really strange!



Simple, but not too simple

Traditional development boards are often too complex, there are many accessories, such as LCD screen, buttons, LEDs, 7-segment digital tube and so on. Development board shows all its functions. The number of functional Arduino board shows the absolute minimum, if you want to expand the functions, simply increase the Shield (shield). Arduino Shield There are hundreds, from LCD to wireless Internet technology, but to increase the number of Shield set by the user. Expansion Shield features are also easy to make extensions Shield features people will stimulate commerce.

Non-chip maker manufacture

Arduino development board is not designed by the chip manufacturer. Why emphasize this point? Because the chip maker in order to highlight their products unique, they often add some strange things. The Arduino emphasize commonalities rather than differences between the microcontroller. This means that the Arduino is a great platform for beginners, as long as the Arduino board can do things that you can do on any other microcontroller. The basic features will accompany you for a long time.

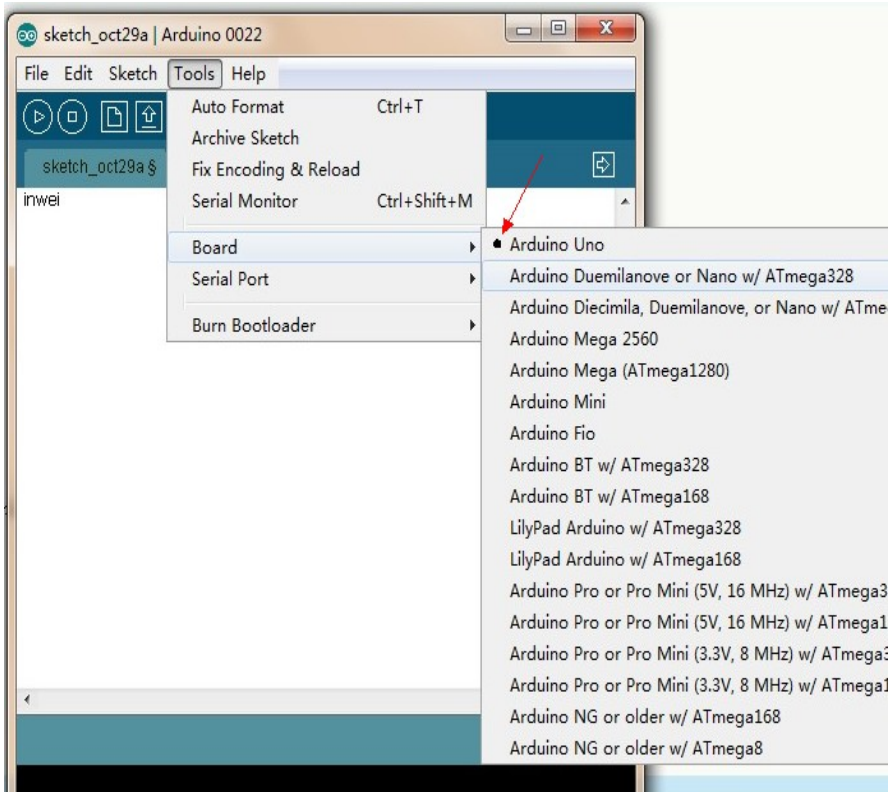
2 Arduino-driven installation and programming procedures

First, download the Arduino development software, web address:
http://arduino.cc/en/Guide/HomePage

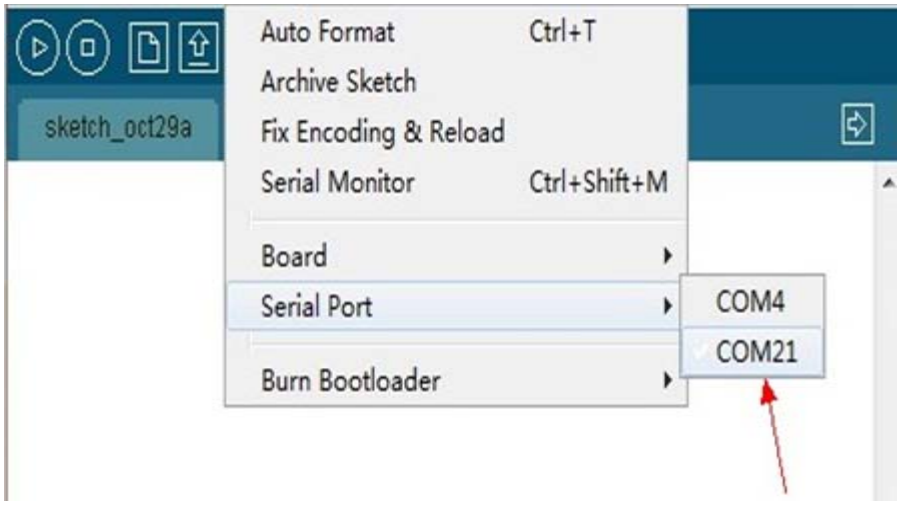
Please follow the official website tutorial to install the software(Mac and Linux OS also supported)

The following program demonstrates the first programmer, light "L" lights

In Arduino-0023 programming interface, click [Tools], move the mouse to the drop-down menu [Board] option in the pop-up submenus seen continuing, [Arduino UNO] if there is a black spot in front of, if not, then the point Click [Arduino UNO] this option.



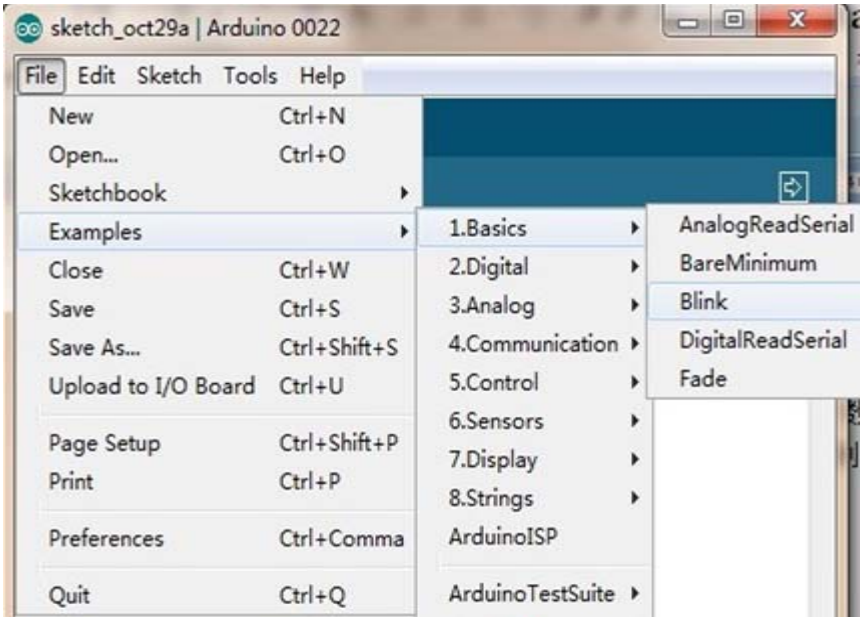
Down to select the correct COM connection. Just remember when installing the hardware requirements of your record that (COMX) What's the value of X? Necessary to use here. If Arduino UNO port just installed is 21, so the mouse click 21.



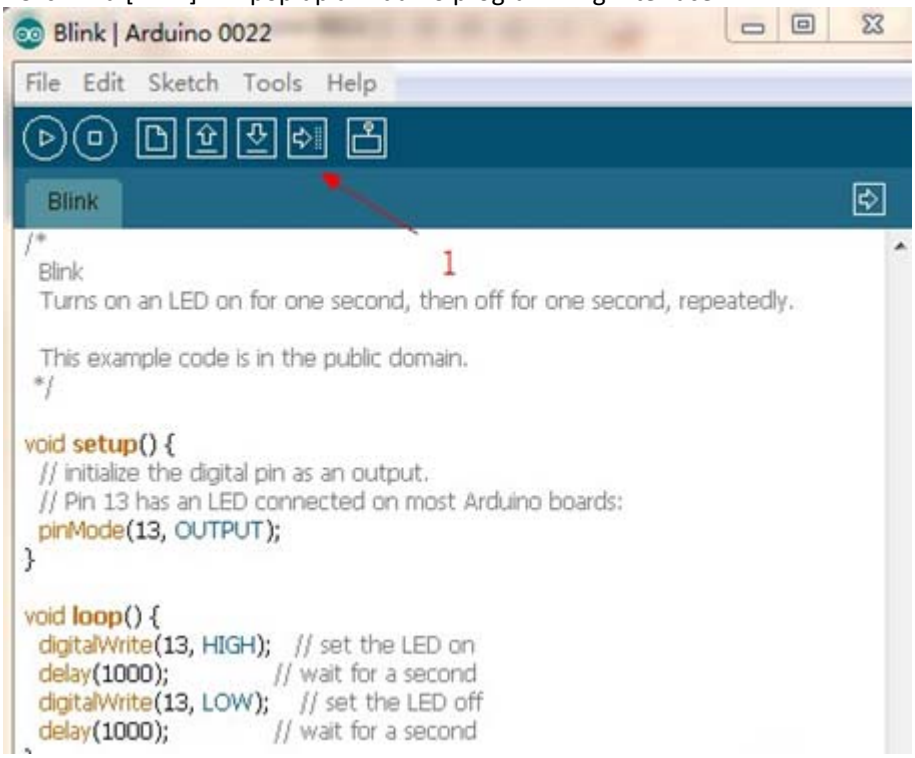
Let come to import a "L" flashing lights sample program, the left mouse button [File]

Inside the pop-up menu, move the mouse to pull down [Examples], the menu extend to the right to [1.Basics]

Move the mouse to [1.Basics] After the menu continues to expand, find the [Blink], left-click [Blink]



Click End [Blink] will pop up a Arduino programming interface



<!--[if !vml]--><!--[endif]--> Direct point left a red arrow icon within the meaning of, you will

find the Arduino UNO motherboards have two yellow light will flash for a while, along with two wild yellow flashing lights off. Text prompt box appears below the program,

L lights on the motherboard with 1 second blinking.

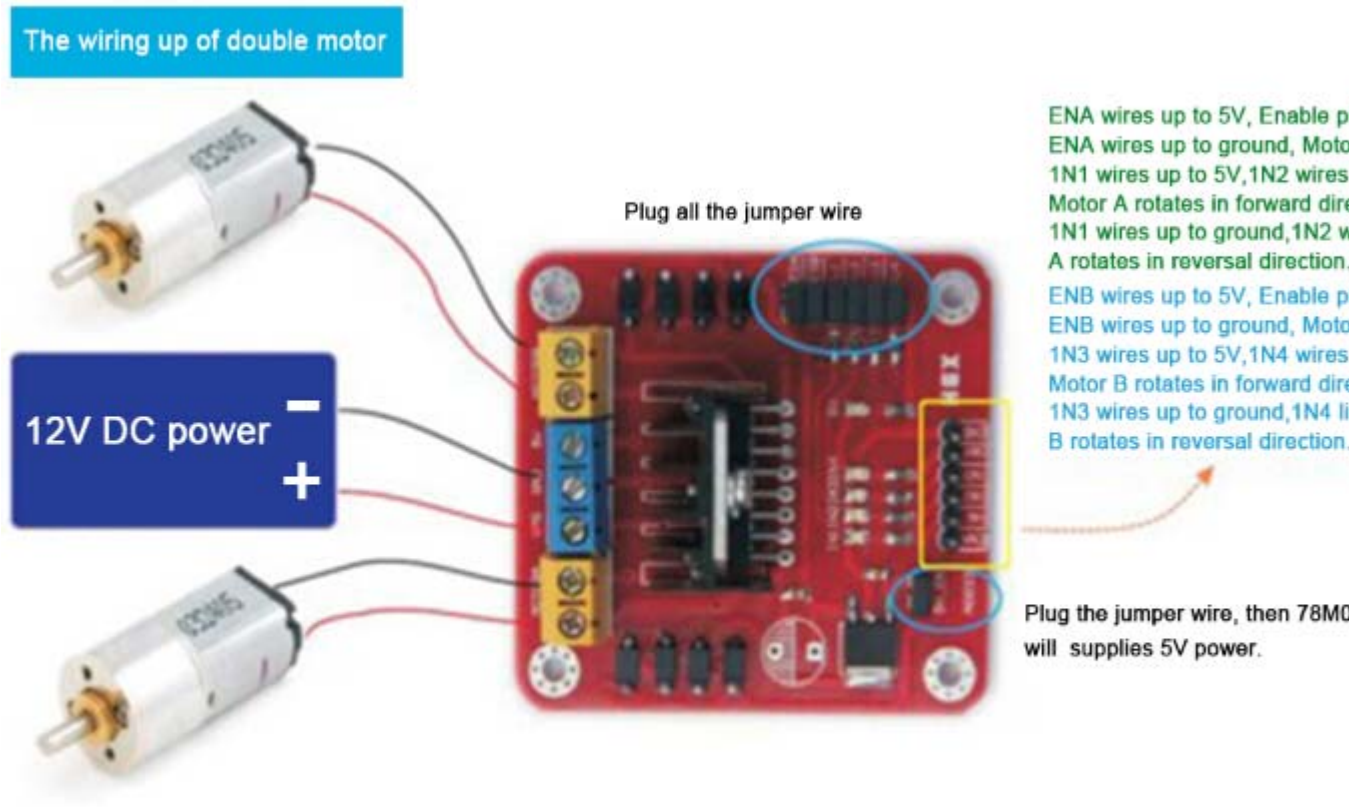
So congratulations, you first program has been a success! ! !

VII. Experimental Details

1.L298N motor driver board applications

L298N bridge driver board Please refer to (L298N dual H-bridge DC motor driver board manual),

here is not to say, but there are some users do not know how to control two DC motors, here to do some detail.



First VMS driver to take power section can be accessed by an external power supply, usually around 9V more appropriate, logical part of the board can take power, that the terminal may be left unconnected, but also access to +5 V-+7 V. About two rows of three terminal pins are used to control two DC motors. EA, EB access ArduinoPWM interfaces for motor control, I1, I2, I3, I4 interfaces are used to control two DC motors forward, backward, steering and brakes, only digital interface to access Arduino.

This preparatory work completed, you can write a program, and here I put the car straight, backward, turn left, turn right, brake functions are written into the program for your reference.

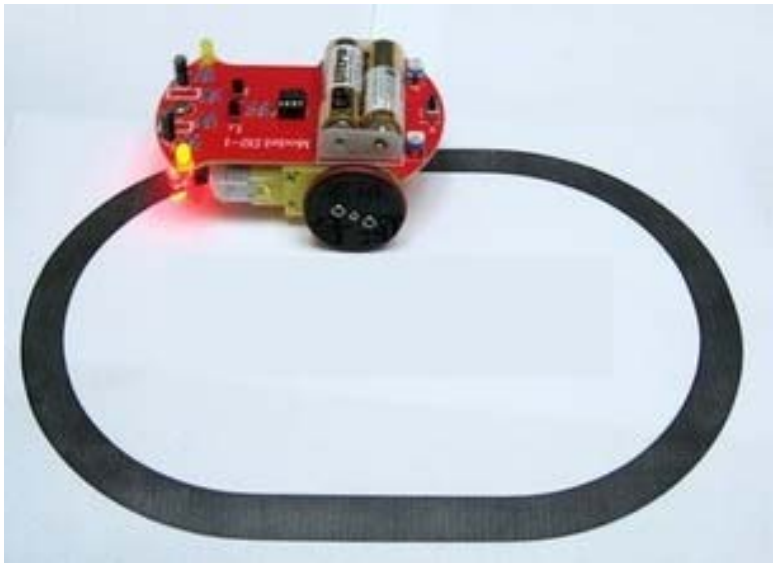
Procedures are as follows:

```
int pinI1 = 8 ;/ / define interfaces I1
int pinI2 = 9 ;/ / define I2 interfaces
int speedpin = 11 ;/ / define EA (PWM control) Interface
int pinI3 = 6 ;/ / define I3 Interface
int pinI4 = 7 ;/ / define I4 Interface
int speedpin1 = 10 ;/ / define EB (PWM control) Interface
void setup ()
{
```

```
pinMode (pinI1, OUTPUT);
pinMode (pinI2, OUTPUT);
pinMode (speedpin, OUTPUT);
pinMode (pinI3, OUTPUT);
pinMode (pinI4, OUTPUT);
pinMode (speedpin1, OUTPUT);
}
void loop ()
{
/ / Straight
analogWrite (speedpin, 100) ;/ / set the speed of the input analog value
analogWrite (speedpin1, 100);
digitalWrite (pinI4, LOW) ;/ / make DC motor (right) turn counterclockwise
digitalWrite (pinI3, HIGH);
digitalWrite (pinI1, LOW) ;/ / make DC motor (left) clockwise
digitalWrite (pinI2, HIGH);
delay (2000);
/ / Back
analogWrite (speedpin, 100) ;/ / set the speed of the input analog value
analogWrite (speedpin1, 100);
digitalWrite (pinI4, HIGH) ;/ / make DC motor (right) clockwise
digitalWrite (pinI3, LOW);
digitalWrite (pinI1, HIGH) ;/ / make DC motor (left) turn counterclockwise
digitalWrite (pinI2, LOW);
delay (2000);
/ / Left
analogWrite (speedpin, 60) ;/ / set the speed of the input analog value
analogWrite (speedpin1, 60);
digitalWrite (pinI4, LOW) ;/ / make DC motor (right) turn counterclockwise
digitalWrite (pinI3, HIGH);
digitalWrite (pinI1, HIGH) ;/ / make DC motor (left) turn counterclockwise
digitalWrite (pinI2, LOW);
delay (2000);
/ / Right
analogWrite (speedpin, 60) ;/ / set the speed of the input analog value
analogWrite (speedpin1, 60);
digitalWrite (pinI4, HIGH) ;/ / make DC motor (right) clockwise
digitalWrite (pinI3, LOW);
digitalWrite (pinI1, LOW) ;/ / make DC motor (left) clockwise
digitalWrite (pinI2, HIGH);
delay (2000);
/ / Brake
digitalWrite (pinI4, HIGH) ;/ / make DC motor (right) brake
digitalWrite (pinI3, HIGH);
digitalWrite (pinI1, HIGH) ;/ / make DC motor (left) brake
digitalWrite (pinI2, HIGH);
delay (2000);
}
```

Note: The program I used to turn left and turn right just turn in a controlled manner, otherwise not list them, you can try it yourself.

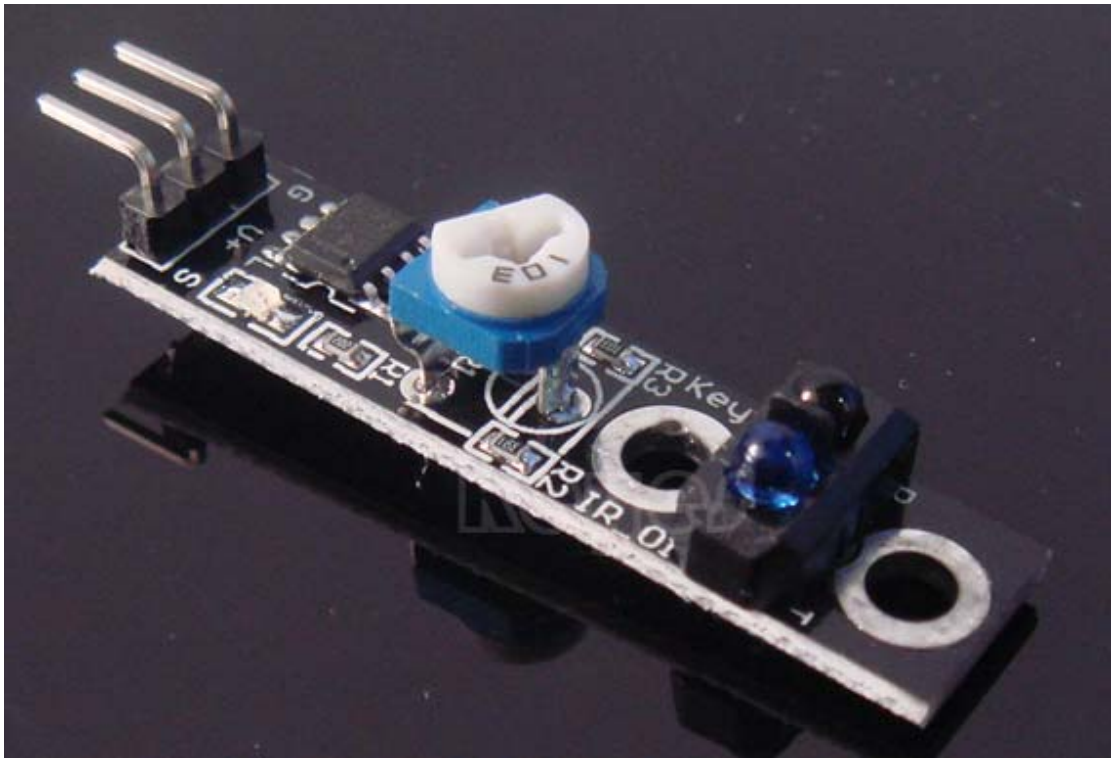
2 Smart car tracking



Tracking module principle: TCRT5000 infrared tube works is the use of infrared reflectivity of color is not the same, the intensity of the reflected signal is converted into a current signal. Black and white tracing module detects a black high efficient white is detected active low, detection height of 0 - 3cm .

Note: You can use the circuit potentiometer knob to adjust the sensitivity of the black and white tracing.

TCRT5000 infrared tube in robot design, is widely used in industrial manufacturing. Available in black and white tracing TCRT5000 making robots, industrial counting sensors.



Usage:

1 sensor interface has three rows of pins, respectively, GND, VCC, OUT. VCC and GND for the supply side, OUT is the signal output.

(2) an object is detected, the low level output signal terminal; no object is detected, the signal-side output high.

3 main judgment signal output is 0 or 1, you can determine whether the object exists.

Performance parameters:

1: detection distance, testing about White 2 cm . Depending on the type of color different from white farthest.

2 Supply voltage:. 2.5V ~ 12V, do not exceed 12V. (Note: It is best to use a low voltage power supply, the power supply voltage is too high sensor life becomes shorter 5V power supply is better.)

3. Operating current, 5V when 18 ~ 20ma. After extensive testing when the sensor hardware is set to 18 ~ 20ma Current best performance, mainly in the anti-jamming capability.

4 object is detected, the low level output signal terminal; no object is detected, the signal-side output high.

5 TTL level sensor output can be connected directly to the 3.3V or 5V microcontroller IO port.

Black or white line detection principle

1 Use small black light reflectance of this feature, when the plane's color is not black, the sensor emitted infrared light reflected back by the majority. So low sensor output level 0.

(2) When there is a flat black line, black line sensor in the party, because black reflective capacity is weak, reflected infrared light rarely reach the action level sensor, the sensor also outputs 1.

(3) We just use the microcontroller to determine the output of the sensor is 0 or 1, will be able to detect the black line.

4 white line detection principle and the principle of detecting the black line, like the detection of the white line, white line around the black color also relatively close, and then adjust the infrared sensor above the adjustable resistance, reduced sensitivity has been adjusted to just The color is not detected until the periphery, so you can detect the white line.

Experimental procedure:

```
int pin = 7 ;/ / Defines interface Arduino digital pin7
int val ;/ / define variables
void setup ()
{
pinMode (ledPin, OUTPUT) ;/ / set the digital interface output interface 7
Serial.begin (9600) ;/ / set the serial port baud rate to 9600kbps
}
void loop ()
{
value val = digitalRead (pin) ;/ / read digital interface
Serial.println (val) ;/ value / output digital interface
}
```

Tracking car

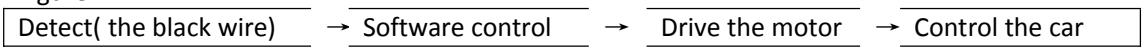
Learn hunt module after we started doing a hunt of their own car

The design is based on a simple Arduino control system automatically tracing the car, including the trolley system configuration of hardware and software design methods. Car with Arduino UNO as the control center, using infrared photoelectric sensors to detect trace black pavement and road surface detection signal back to the Arduino microcontroller. Arduino microcontroller signal collected are analyzed to determine, in a timely manner to control the drive motor to adjust the car turned, allowing the car to automatically drive along the black track to achieve car automatically tracing purposes.

Based on the design of a DC motor automatically tracing the car, so the car can automatically detect ground black track, and a black car traveling along the track. System solutions block

diagram shown in Figure 1-1.

Figure1-1



Block diagram of Figure 1-1 System program

Car tracking principle

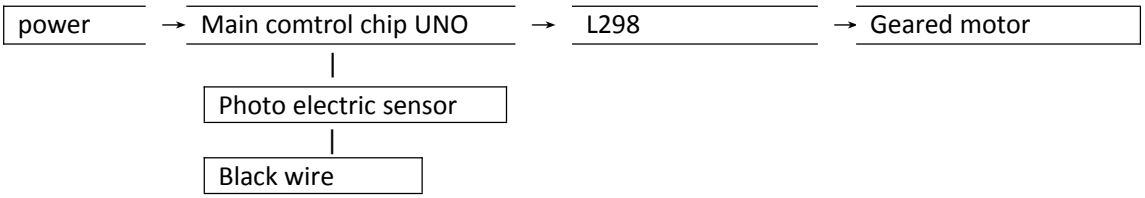
Here is the tracking on the trolley through the black line white floor, the floor of the black and white lines of the different light reflection coefficients, according to the strength of the received reflected light to determine the "road." The method is usually taken infrared detection method.

Infrared detection method, i.e., the surface of the object using infrared different colors have different characteristics of reflection properties in the course of driving the car constantly emit infrared light towards the ground, the infrared diffuse reflection occurs when light encounters a white paper floor, the reflected light is housed in a small car receiver tube receiver; If you encounter black line is the infrared light is absorbed, small car receiver tube not receive infrared light. SCM received on whether the reflected infrared light to determine the basis of the location of the black line and trolley walking routes. Infrared detector detects a limited distance,

Control System Design

Automatic tracking car control system, consisting of the main part of the power control circuit module, infrared detection module, motor and drive module, the control system block diagram shown in Figure 2-1.

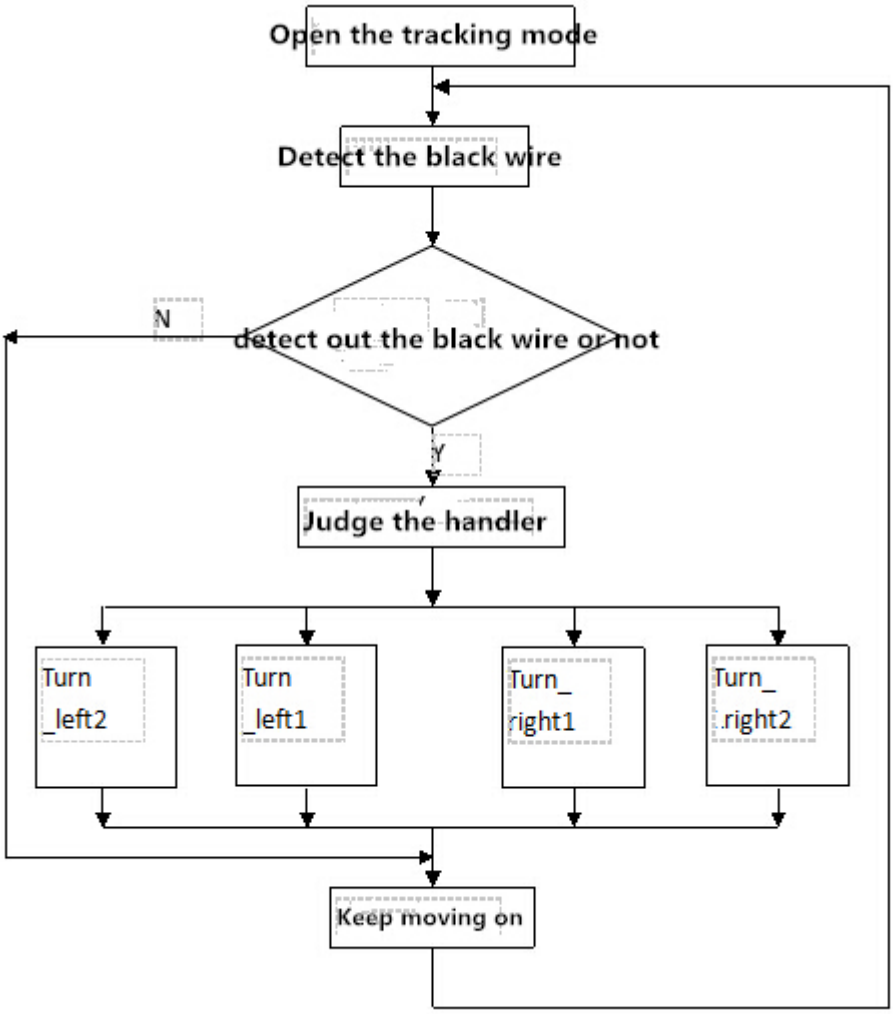
Figure2-1 The structure of control system.



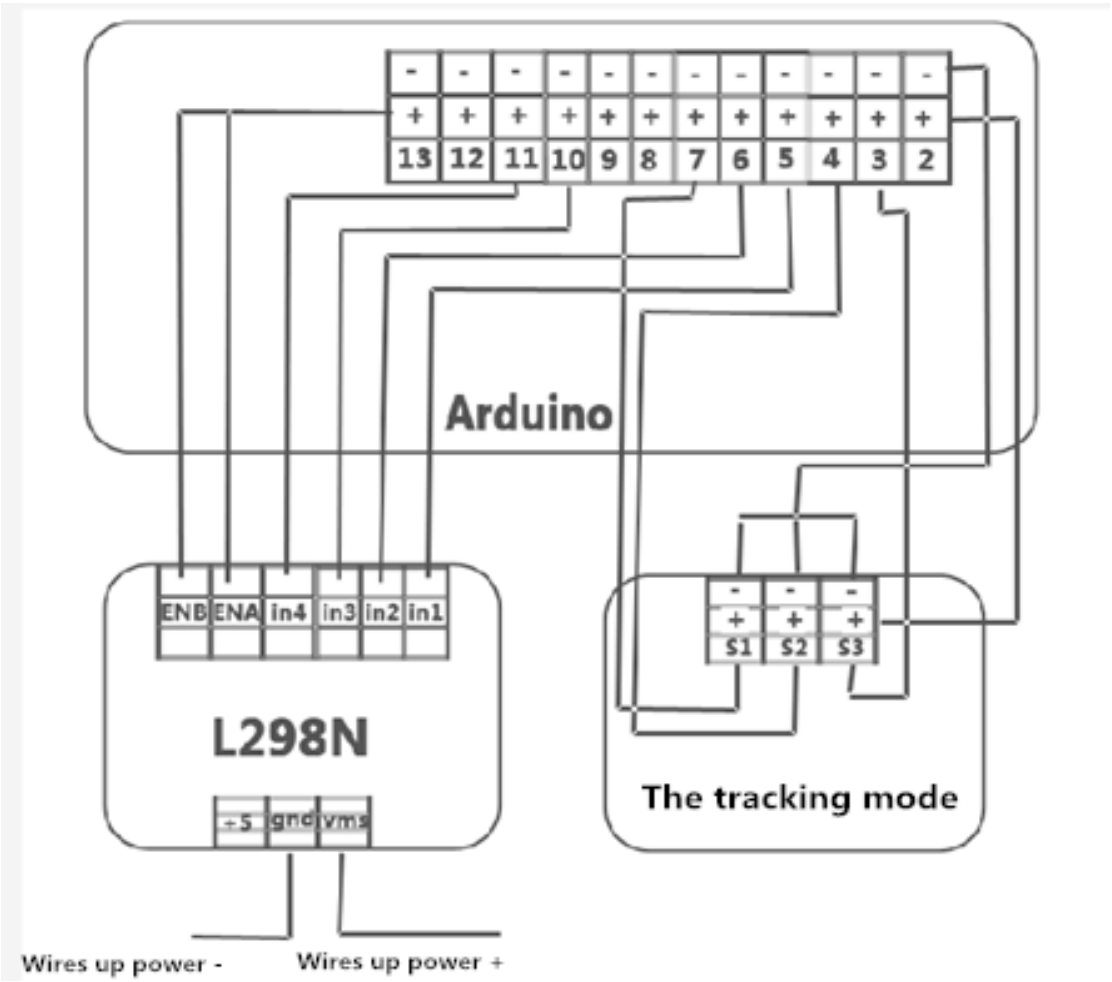
Car tracking flowchart

After the car into tracking mode, which began non-stop scanning probe connected with the MCU I / O port, upon detection of an I / O port has a signal that entered judgment process, first determine the three detectors in Which one to detect the black line.

Figure3-1 The processing of Tracking



Arduino car tracking wiring diagram



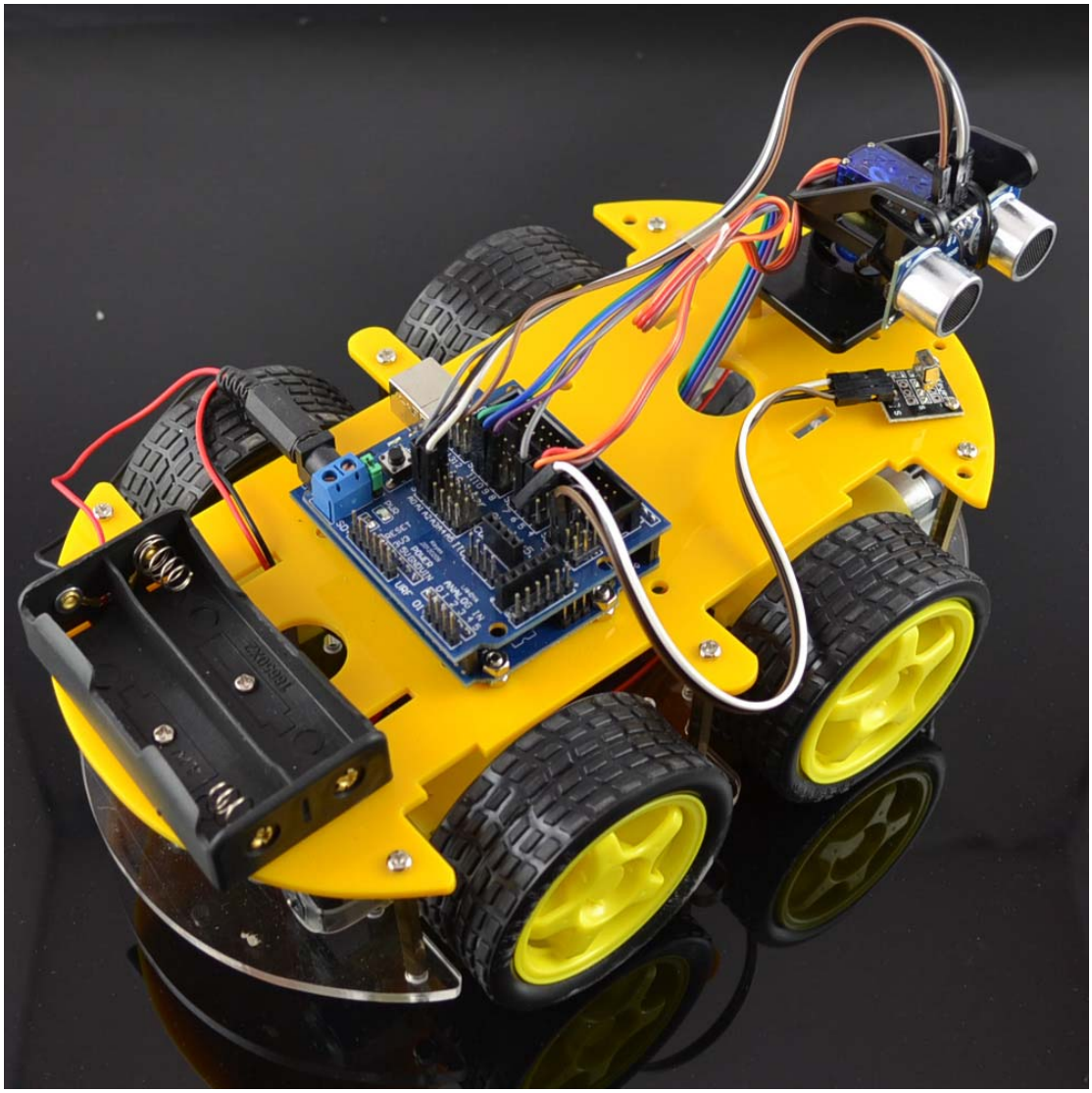
Tracking car Arduino program:

```
int MotorRight1 = 5;
int MotorRight2 = 6;
int MotorLeft1 = 10;
int MotorLeft2 = 11;
const int SensorLeft = 7;              / / Left sensor input pin
const int SensorMiddle = 4;            / / The sensor input pin
const int SensorRight = 3;             / / Right sensor input pin
int SL;           / / Left sensor status
int SM;           / / The sensor status
int SR;           / / Right sensor status

void setup ()
{
   Serial.begin (9600);
   pinMode (MotorRight1, OUTPUT);      / / Pin 8 (PWM)
   pinMode (MotorRight2, OUTPUT);      / / Pin 9 (PWM)
   pinMode (MotorLeft1,      OUTPUT);      / / Pin 10 (PWM)
   pinMode (MotorLeft2,      OUTPUT);      / / Pin 11 (PWM)
   pinMode (SensorLeft, INPUT); / / define left Sensors
   pinMode (SensorMiddle, INPUT) ;/ / definition sensors
   pinMode (SensorRight, INPUT); / / definition of the right sensor
}
```

```
void loop ()
  {
  SL = digitalRead (SensorLeft);
  SM = digitalRead (SensorMiddle);
  SR = digitalRead (SensorRight);

          if (SM == HIGH) / / in sensors in black areas
        {
            if (SL == LOW & SR == HIGH) / / left and right black white, turn left
            {
                digitalWrite (MotorRight1, LOW);
                digitalWrite (MotorRight2, HIGH);
                analogWrite (MotorLeft1, 0);
                analogWrite (MotorLeft2, 80);
            }
            else if (SR == LOW & SL == HIGH) / / left and right black white, turn right
            {
                analogWrite (MotorRight1, 0) ;/ / right turn
                analogWrite (MotorRight2, 80);
                digitalWrite (MotorLeft1, LOW);
                digitalWrite (MotorLeft2, HIGH);
            }
          else       / / Both sides white, straight
            {
                digitalWrite (MotorRight1, LOW);
                digitalWrite (MotorRight2, HIGH);
                digitalWrite (MotorLeft1, LOW);
                digitalWrite (MotorLeft2, HIGH);
                  analogWrite (MotorLeft1, 200);
                analogWrite (MotorLeft2, 200);
                analogWrite (MotorRight1, 200);
                analogWrite (MotorRight2, 200);
            }
        }
         else / / the sensors in the white area
       {
            if (SL == LOW & SR == HIGH) / / left and right black white, fast turn left
         {
                digitalWrite (MotorRight1, LOW);
                digitalWrite (MotorRight2, HIGH);
                digitalWrite (MotorLeft1, LOW);
                digitalWrite (MotorLeft2, LOW);
         }
           else if (SR == LOW & SL == HIGH) / / left and right black white, quick right turn
         {
              digitalWrite (MotorRight1, LOW);
              digitalWrite (MotorRight2, LOW);
              digitalWrite (MotorLeft1, LOW);
              digitalWrite (MotorLeft2, HIGH);
         }
           else / / are white, stop
         {
         digitalWrite (MotorRight1, HIGH);
```

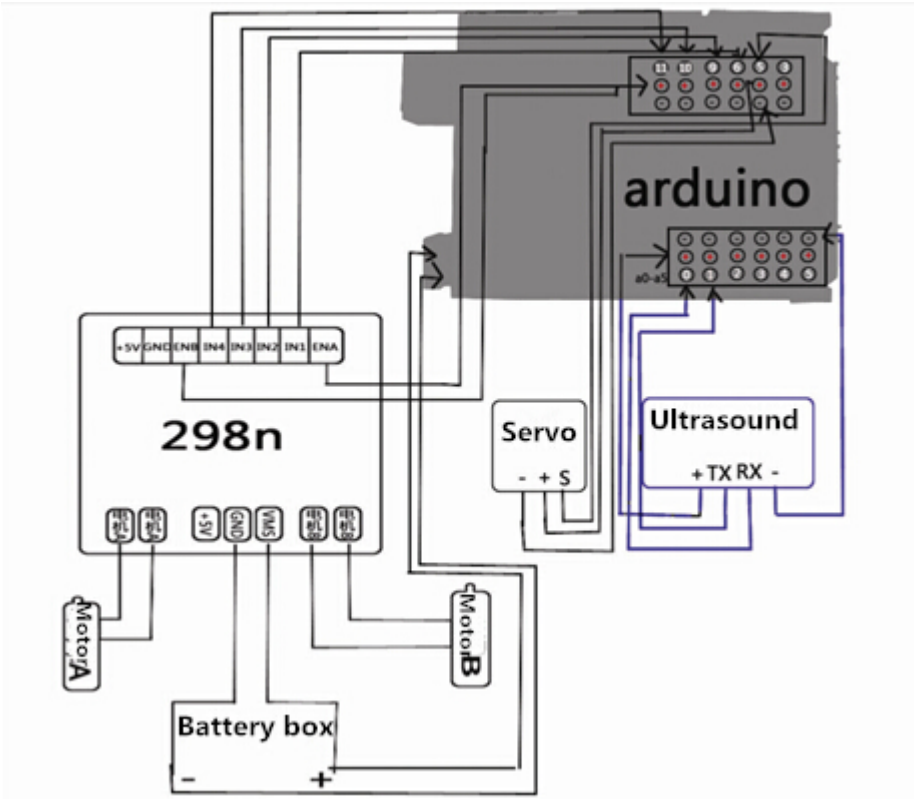
```
digitalWrite (MotorRight2, LOW);
digitalWrite (MotorLeft1, HIGH);
digitalWrite (MotorLeft2, LOW);;
}}}
```

3 ultrasonic obstacle avoidance Smart Car



Ultrasonic Intelligent obstacle avoidance achieve convenient computing simple, easy to do real-time control, and measurement accuracy can meet the practical requirements, it became common obstacle avoidance. Ultrasonic use reference (Arduino Ultrasonic Ranging instructions).

Ultrasonic intelligent wiring diagram;



A: Motor connection

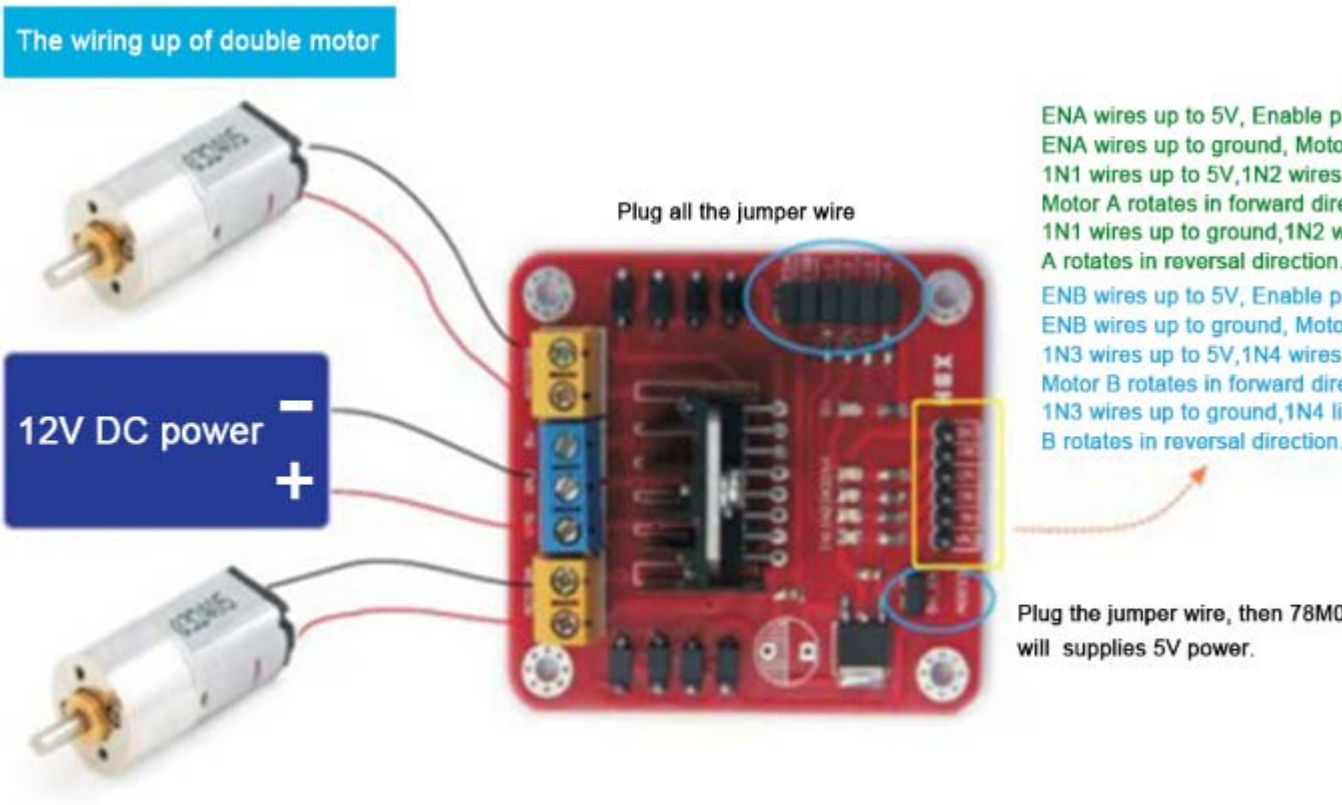
L298N motor a pick of MOTOA

The two then L298N motor MOTOB

Two: L298N supply deal

5 with 6 way power to the battery box to take L298N motor drive module power, the other way to ARDUINO board power supply,

L298N motor drive module to the power supply + pole to L298N of VMS interface power - L298N the GND connection interfaces, +5 V Interface L298N vent panel is not connected.



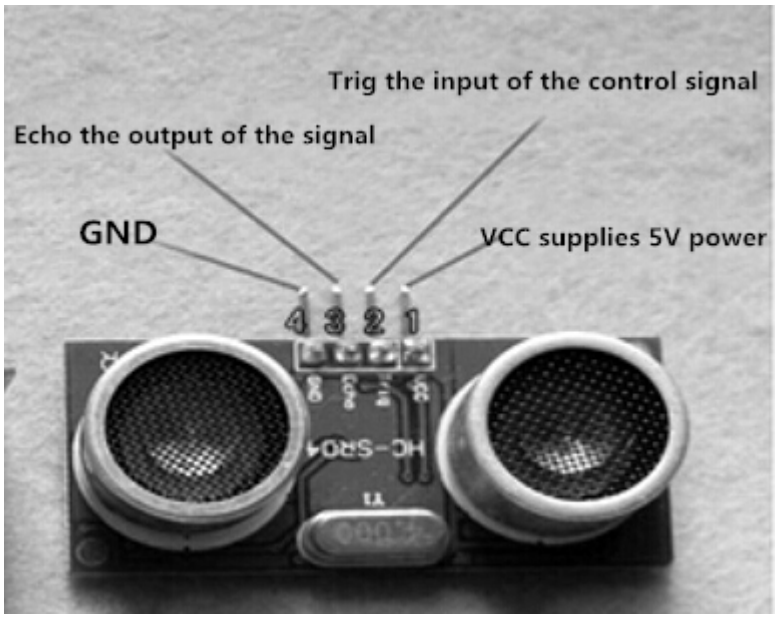
Three: the motor and steering enabled (with the program)

int pinLB = 6; / / After defining the 6 pin left, then to the foot force plate PWM6
int pinLF = 9; / / Define the 9 pin left, then to the foot force plate PWM9
int pinRB = 10; / / Define pin 10 right rear, then to force the foot plate PWM10
int pinRF = 11; / / Define the 11-pin front right, then to the foot force plate PWM11

Four: Servo connections

myservo.attach (5); / / Define servo motor output section 5 pin (PWM)

Five: ultrasonic sensor connection
Ultrasonic sensors have four legs
VCC +5 V connection
TRIQ signal input
ECHO signal output
GND Ground




```
int inputPin = A0;        / / Define pin ultrasonic signal reception
int outputPin = A1;       / / Define pin ultrasonic signal transmitter

Ultrasonic Smart car obstacle avoidance procedures (ARDUINO)
      L = Left
      R = Right
      F = front
      B = after
* /
# Include <Servo.h>
int pinLB = 6;            / / Define pin left after 6
int pinLF = 9;            / / Define the 9-pin front left

int pinRB = 10;           / 10 pin definitions right rear /
int pinRF = 11;           / / Define the 11-pin front right

int inputPin = A0;        / / Define pin ultrasonic signal reception
int outputPin = A1;       / / Define pin ultrasonic signal transmitter

int Fspeedd = 0;              / /-Speed
int Rspeedd = 0;              / / Right speed
int Lspeedd = 0;              / / Left-speed
int directionn = 0;         / / Front Left = 8 after = 2 = 4 Right = 6
Servo myservo;                / / Set myservo
int delay_time = 250; / / settling time after steering servo motors

int Fgo = 8;                  / / Forward
int Rgo = 6;                  / / Right
int Lgo = 4;                  / / Left
int Bgo = 2;                  / / Reverse

void setup ()
   {
   Serial.begin (9600);           / / Define motor output pin
```

```
  pinMode (pinLB, OUTPUT); / / pin 8 (PWM)
  pinMode (pinLF, OUTPUT); / / pin 9 (PWM)
  pinMode (pinRB, OUTPUT); / / pin 10 (PWM)
  pinMode (pinRF, OUTPUT); / / pin 11 (PWM)

  pinMode (inputPin, INPUT);          / / Define ultrasound input pin
  pinMode (outputPin, OUTPUT);       / / Define ultrasonic output pin

  myservo.attach (5);          / / Define servo motor output section 5 pin (PWM)
  }
void advance (int a)           / / Forward
      {
       digitalWrite (pinRB, LOW);       / / The motor (rear right) action
       digitalWrite (pinRF, HIGH);
       digitalWrite (pinLB, LOW);       / / The motor (left rear) action
       digitalWrite (pinLF, HIGH);
       delay (a * 100);
      }

void right (int b)             / / Turn right (single wheel)
      {
       digitalWrite (pinRB, LOW);        / / The motor (rear right) action
       digitalWrite (pinRF, HIGH);
       digitalWrite (pinLB, HIGH);
       digitalWrite (pinLF, HIGH);
       delay (b * 100);
      }
void left (int c)              / / Turn left (single wheel)
      {
       digitalWrite (pinRB, HIGH);
       digitalWrite (pinRF, HIGH);
       digitalWrite (pinLB, LOW);        / / The motor (left rear) action
       digitalWrite (pinLF, HIGH);
       delay (c * 100);
      }
void turnR (int d)             / / Turn right (wheel)
      {
       digitalWrite (pinRB, LOW);       / / The motor (rear right) action
       digitalWrite (pinRF, HIGH);
       digitalWrite (pinLB, HIGH);
       digitalWrite (pinLF, LOW);       / / The motor (front left) action
       delay (d * 100);
      }
void turnL (int e)             / / Turn left (wheel)
      {
       digitalWrite (pinRB, HIGH);
       digitalWrite (pinRF, LOW);        / / The motor (front right) action
       digitalWrite (pinLB, LOW);        / / The motor (left rear) action
       digitalWrite (pinLF, HIGH);
       delay (e * 100);
      }
void stopp (int f)              / / Stop
      {
       digitalWrite (pinRB, HIGH);
```

```
  digitalWrite (pinRF, HIGH);
  digitalWrite (pinLB, HIGH);
  digitalWrite (pinLF, HIGH);
  delay (f * 100);
 }
void back (int g)                    / / Check out
 {

  digitalWrite (pinRB, HIGH);       / / The motor (rear right) action
   digitalWrite (pinRF, LOW);
  digitalWrite (pinLB, HIGH);       / / The motor (left rear) action
  digitalWrite (pinLF, LOW);
  delay (g * 100);
 }

void detection ()                  / / Measure three angles (0.90.179)
 {
    int delay_time = 250;         Settling time / / servo motor after turning
    ask_pin_F ();                    / / Read from front

  if (Fspeedd <10)                 / / If the distance is less than 10 cm in front of
   {
   stopp (1);                       / / Clear the output data
   back (2);                        / / Check out 0.2 seconds
   }

   if (Fspeedd <25)                / / If the distance is less than 25 cm in front of
   {
      stopp (1);                      / / Clear the output data
      ask_pin_L ();                   / / Read from left
      delay (delay_time);             / / Wait for a stable servo motor
      ask_pin_R ();                   / / Read from the right
      delay (delay_time);             / / Wait for a stable servo motor

      if (Lspeedd> Rspeedd)         / / If the distance is greater than the right from the left
      {
       directionn = Rgo;              / / Right away
      }

      if (Lspeedd <= Rspeedd)         / / If the left is less than or equal to the distance from
the right
      {
       directionn = Lgo;              / / Turn Left
      }

      if (Lspeedd <10 && Rspeedd <10)         / / If the distance to the left and right are less
than 10 cm distance
      {
       directionn = Bgo;              / / To go after
      }
   }
   else                                 / / Add as front not less than (greater than) 25 cm
   {
      directionn = Fgo;                / / Move forward
```

```
          }

        }
  void ask_pin_F ()         / / Measure the distance from the front
        {
            myservo.write (90);
          digitalWrite (outputPin, LOW);          / / Let ultrasonic transmitter low voltage 2 μ s
          delayMicroseconds (2);
          digitalWrite (outputPin, HIGH);        / / Let ultrasonic transmitter high voltage 10 μ s,
where at least 10 μ s
          delayMicroseconds (10);
          digitalWrite (outputPin, LOW);           / / Maintain low voltage ultrasonic transmitter
          float Fdistance = pulseIn (inputPin, HIGH);        / / Read worse time difference
          Fdistance = Fdistance/5.8/10;                / / Time to turn to the distance (unit: cm)
          Serial.print ("F distance:");              / / Output distance (unit: cm)
          Serial.println (Fdistance);                  / / Display the distance
          Fspeedd = Fdistance;                           / / Read into the distance Fspeedd (former
speed)
        }
     void ask_pin_L ()         / / Measure the distance from the left
        {
          myservo.write (5);
          delay (delay_time);
          digitalWrite (outputPin, LOW);          / / Let ultrasonic transmitter low voltage 2 μ s
          delayMicroseconds (2);
          digitalWrite (outputPin, HIGH);        / / Let ultrasonic transmitter high voltage 10 μ s,
where at least 10 μ s
          delayMicroseconds (10);
          digitalWrite (outputPin, LOW);           / / Maintain low voltage ultrasonic transmitter
          float Ldistance = pulseIn (inputPin, HIGH);        / / Read worse time difference
          Ldistance = Ldistance/5.8/10;                / / Time to turn to the distance (unit: cm)
          Serial.print ("L distance:");               / / Output distance (unit: cm)
          Serial.println (Ldistance);                  / / Display the distance
          Lspeedd = Ldistance;                           / / Read into the distance Lspeedd (left-speed)
        }
  void ask_pin_R ()         / / Measure the distance from the right
        {
          myservo.write (177);
          delay (delay_time);
          digitalWrite (outputPin, LOW);          / / Let ultrasonic transmitter low voltage 2 μ s
          delayMicroseconds (2);
          digitalWrite (outputPin, HIGH);        / / Let ultrasonic transmitter high voltage 10 μ s,
where at least 10 μ s
          delayMicroseconds (10);
          digitalWrite (outputPin, LOW);           / / Maintain low voltage ultrasonic transmitter
          float Rdistance = pulseIn (inputPin, HIGH);        / / Read worse time difference
          Rdistance = Rdistance/5.8/10;                / / Time to turn to the distance (unit: cm)
          Serial.print ("R distance:");               / / Output distance (unit: cm)
          Serial.println (Rdistance);                  / / Display the distance
          Rspeedd = Rdistance;                           / / Will read into the distance Rspeedd
(Right-speed)
        }

  void loop ()
```

```
    {
      myservo.write (90);       / / Let servo motor position ready to return to the pre-prepared
next time measurement
      detection ();             / / Measure the angle and direction of judgment to where to move

     if (directionn == 2)       / / If directionn (direction) = 2 (reverse)
     {
       back (8);                          / /       Retrogression (car)
       turnL (2);                         / / Move slightly to the left (to prevent stuck in dead
alley)
       Serial.print ("Reverse");         / / Display direction (backwards)
     }
     if (directionn == 6)                / / If directionn (direction) = 6 (right turn)
     {
       back (1);
       turnR (6);                         / / Right
       Serial.print ("Right");          / / Display direction (turn left)
     }
     if (directionn == 4)                / / If directionn (direction) = 4 (turn left)
     {
       back (1);
       turnL (6);                        / / Left
       Serial.print ("Left");           / / Display direction (turn right)
     }
     if (directionn == 8)                / / If directionn (direction) = 8 (forward)
     {
      advance (1);                       / / Normal Forward
      Serial.print ("Advance");         / / Display direction (forward)
      Serial.print ("          ");
     }
    }
```

4 infrared remote intelligent vehicle test

Experiment before you go:

1 first IRremote library folders into libraries directory under Arduino

2 open IrReceive.pde measured their infrared remote control code (in the Serial Monitor can display IRcode), then IRcode record, and then to revise their programs which can be infrared codes.

```
/ *
   * IRRemote infrared remote control code test
   *         Examples 1.2: Show infrared protocol type, such as NEC, Sony SIRC, Philips RC5, Philips
RC6 and other agreements
   * /
# Include <IRremote.h>                        / / Function library references IRRemote

const int irReceiverPin = 2;                / / OUTPUT signals IR receiver connected to pin 2

IRrecv irrecv (irReceiverPin);              / / Define an object to receive infrared signals
IRrecv
decode_results results;                     / / Decoding results will result in structural
variables in decode_results
```

```
 void setup ()
 {
    Serial.begin (9600);                                  / / Open Serial port, the communication
speed is 9600 bps
    irrecv.enableIRIn ();                                / / Start infrared decoding
 }

 / / Display the type of infrared protocol
 void showIRProtocol (decode_results * results)
 {
    Serial.print ("Protocol:");

    / / Judgment infrared protocol types
    switch (results-> decode_type) {
      case NEC:
         Serial.print ("NEC");
         break;
      case SONY:
         Serial.print ("SONY");
         break;
      case RC5:
         Serial.print ("RC5");
         break;
      case RC6:
         Serial.print ("RC6");
         break;
      default:
         Serial.print ("Unknown encoding");
    }

    / / Print the infrared codes to Serial port
    Serial.print (", irCode:");
    Serial.print (results-> value, HEX);          / / Infrared code
    Serial.print (",       bits: ");
    Serial.println (results-> bits);              / / Number of bits coded infrared
 }

 void loop ()
 {
    if (irrecv.decode (& results)) {                / / Decoding is successful, you receive a set of
infrared signals
        showIRProtocol (& results);                    / / Display the type of infrared protocol
        irrecv.resume ();                               / / Continue to accept a set of infrared
signals
    }
 }
```

The measured infrared key code into the program to replace the infrared part of the control

```
//******红外控制部分********
long advence = 0x00EF807F;
long back = 0x00EFA05F;
long stop = 0x00EF906F;
long left = 0x00EF00FF;
long right = 0x00EF40BF;
```

<!--[if !vml]--> <!--[endif]-->

Infrared remote intelligent vehicle program

```
/ / ****** Infrared remote intelligent vehicle program *******
# Include <IRremote.h>
int RECV_PIN = A0;
int pinLB = 6 ;/ / define interfaces I1
int pinLF = 9 ;/ / define I2 interfaces
int pinRB = 3 ;/ / define I3 Interface
int pinRF = 5 ;/ / define I4 Interface
/ / ****** Infrared control section ********
long advence = 0x00EF    807F    ;
long back = 0x00EFA    05F    ;
long stop = 0x00EF    906F    ;
long left = 0x00EF00FF;
long right = 0x00EF40BF;

IRrecv irrecv (RECV_PIN);
decode_results results;
void dump (decode_results * results) {
   int count = results-> rawlen;
   if (results-> decode_type == UNKNOWN)
      {
        Serial.println ("Could not decode message");
      }
   else
    {
      if (results-> decode_type == NEC)
         {
           Serial.print ("Decoded NEC:");
         }
      else if (results-> decode_type == SONY)
         {
           Serial.print ("Decoded SONY:");
         }
      else if (results-> decode_type == RC5)
         {
           Serial.print ("Decoded RC5:");
         }
      else if (results-> decode_type == RC6)
         {
           Serial.print ("Decoded RC6:");
         }
       Serial.print (results-> value, HEX);
       Serial.print ("(");
       Serial.print (results-> bits, DEC);
       Serial.println ("bits)");
    }
```

```
        Serial.print ("Raw (");
        Serial.print (count, DEC);
        Serial.print ("):");

   for (int i = 0; i <count; i + +)
       {
         if ((i% 2) == 1) {
         Serial.print (results-> rawbuf [i] * USECPERTICK, DEC);
        }
       else
        {
         Serial.print (- (int) results-> rawbuf [i] * USECPERTICK, DEC);
        }
       Serial.print ("");
        }
         Serial.println ("");
        }

void setup ()
   {
   pinMode (RECV_PIN, INPUT);
   pinMode (pinLB, OUTPUT);
   pinMode (pinLF, OUTPUT);

   pinMode (pinRB, OUTPUT);
   pinMode (pinRF, OUTPUT);

Serial.begin (9600);
   irrecv.enableIRIn (); / / Start the receiver
   }

int on = 0;
unsigned long last = millis ();

void loop ()
{
     if (irrecv.decode (& results))
     {
       / / If it's been at least 1/4 second since the last
       / / IR received, toggle the relay
       if (millis () - last> 250)
          {
            on = on!;
/ /                 digitalWrite (8, on HIGH:? LOW);
            digitalWrite (13, on HIGH:? LOW);
            dump (& results);
          }
       if (results.value == advence)
       {DigitalWrite (pinRB, LOW) ;/ / make DC motor (right) GO
       digitalWrite (pinRF, HIGH);
       digitalWrite (pinLB, LOW) ;/ / make DC motor (left) GO
       digitalWrite (pinLF, HIGH);}

       if (results.value == back)
```

```
      {DigitalWrite (pinRB, HIGH) ;/ / make DC motor (right) BACK
      digitalWrite (pinRF, LOW);}

    if (results.value == left)
    {DigitalWrite (pinRB, LOW) ;/ / make DC motor (right) STOP
      digitalWrite (pinRF, HIGH);
      digitalWrite (pinLB, HIGH) ;/ / make DC motor (left) GO
      digitalWrite (pinLF, LOW);}


    if (results.value == right)
    {DigitalWrite (pinRB, HIGH) ;/ / make DC motor (right) GO
      digitalWrite (pinRF, LOW);
      digitalWrite (pinLB, HIGH) ;/ / make DC motor (left) STOP
      digitalWrite (pinLF, HIGH);}

    if (results.value == stop)
       {
      digitalWrite (pinRB, HIGH) ;/ / make DC motor (right) STOP
      digitalWrite (pinRF, HIGH);
      digitalWrite (pinLB, HIGH) ;/ / make DC motor (left) STOP
      digitalWrite (pinLF, HIGH);

   }

    last = millis ();
    irrecv.resume (); / / Receive the next value
  }
}
```

5 Smart Car Bluetooth phone control

Arduino via Bluetooth communication

<!--[if !vml]--> <!--[endif]--> <!--[if !vml]--> <!--[endif]--> Bluetooth - This name came from a tenth-century Danish King Harald Blatand, Blatand meaning in the English language can be interpreted as Bluetooth (Bluetooth).

The so-called Bluetooth (Bluetooth) technology, in fact, is a short-range radio technology, the use of "Bluetooth" technology that can effectively simplify PDAs, notebook computers and communications mobile phone handsets and other mobile communications terminal equipment, but also able to successfully These simplify the communication device and the Internet (Internet) between them so that the modern data transmission between the communication device and the Internet more quickly and efficiently, to broaden the wireless communication path.

Because it is the first to deal with the Bluetooth module, today or take a small test chopper, make Arduino and pc successfully communicate it. Let's wiring, the board +5 V connection Bluetooth VCC, GND motherboard connector Bluetooth-GND, TX motherboard connection Bluetooth RX, RX connected Bluetooth TX. After the success of the Bluetooth module connected to the power supply and PC, Bluetooth module power indicator flashes green link light is lit.

Here's a look at the program, I let my Arduino receives input "r", the interface is pin13 LED flash once, and then output the words keyes.

Procedures are as follows:

```
char val;
int ledpin = 13;
void setup ()
{
Serial.begin (9600);
pinMode (ledpin, OUTPUT);
}
void loop ()
{
val = Serial.read ();
if (val == 'r')
{
digitalWrite (ledpin, HIGH);
delay ((500);
digitalWrite (ledpin, LOW);
delay (500);
Serial.println ("keyes");
}
}
```

Here we learn about the Arduino Bluetooth remote control programmable intelligent car. Can be controlled via Bluetooth forward, backward, turn left, turn right, etc., simple keys, computer and mobile phone two control modes. (Android mobile operating system support 2.3.7 Above. Computer must own Bluetooth)

The first time you need to use the car phone with Bluetooth pairing (after the first pairing later in the wireless device location would not), see the following first steps:

1 Turn on Bluetooth Oh I remember the phone, open the Bluetooth software will alert the user to open

2 Then, as shown in the text prompts, connect Bluetooth devices, Bluetooth devices paired scans Oh, otherwise you can not connect the car.

3 matching car, the password is "1234" try it.

Then it can flourish.

Arduino Bluetooth remote control programmable smart car program:

```
/ / ******************************
int MotorRight1 = 5;
int MotorRight2 = 6;
int MotorLeft1 = 10;
int MotorLeft2 = 11;


void setup ()
{
   Serial.begin (9600);
   pinMode (MotorRight1, OUTPUT);       / / Pin 8 (PWM)
   pinMode (MotorRight2, OUTPUT);       / / Pin 9 (PWM)
   pinMode (MotorLeft1,       OUTPUT);       / / Pin 10 (PWM)
   pinMode (MotorLeft2,       OUTPUT);       / / Pin 11 (PWM)
}

void go () / / Forward
{
            digitalWrite (MotorRight1, LOW);
            digitalWrite (MotorRight2, HIGH);
            digitalWrite (MotorLeft1, LOW);
```

```
            digitalWrite (MotorLeft2, HIGH);

}

void left () / / turn right
{
          digitalWrite (MotorRight1, HIGH);
          digitalWrite (MotorRight2, LOW);
          digitalWrite (MotorLeft1, LOW);
          digitalWrite (MotorLeft2, HIGH);

}
void right () / / turn left
{
          digitalWrite (MotorRight1, LOW);
          digitalWrite (MotorRight2, HIGH);
          digitalWrite (MotorLeft1, HIGH);
          digitalWrite (MotorLeft2, LOW);

}
void stop () / / stop
{
         digitalWrite (MotorRight1, LOW);
         digitalWrite (MotorRight2, LOW);
         digitalWrite (MotorLeft1, LOW);
         digitalWrite (MotorLeft2, LOW);

}
void back () / / Check out
{
            digitalWrite (MotorRight1, HIGH);
            digitalWrite (MotorRight2, LOW);
            digitalWrite (MotorLeft1, HIGH);
            digitalWrite (MotorLeft2, LOW);;

}

void loop ()
{
   char val = Serial.read ();
   Serial.write (val);
   if (-1! = val) {
      if ('W' == val)
      go ();
      else if ('A' == val)
      left ();
      else if ('D' == val)
      right ();
      else if ('S' == val)
      back ();
      else if ('Q' == val)
         stop ();
      delay (500);
      }
```

```
    else
    {
       / / Stop ();
       delay (500);
    }
 }
```

6 Four and one (Hunt Avoidance Infrared remote control Bluetooth Remote Control) Multifunction program

```
 / / ******************************
 # Include <IRremote.h>
 # Include <Servo.h>
 / / ********************** Definition of motor pin ******************** ****
 int MotorRight1 = 5;
 int MotorRight2 = 6;
 int MotorLeft1 = 10;
 int MotorLeft2 = 11;
 int counter = 0;
 const int irReceiverPin = 2; / / OUTPUT signals IR receiver connected to pin 2

 char val;
 / / ********************** Set to detect the IRcode ****************** *******
 long IRfront = 0x00FFA25D;                / / Forward code
 long IRback = 0x00FF629D;                 / / Check out
 long IRturnright = 0x00FFC23D;           / / Right
 long IRturnleft = 0x00FF02FD;            / / Left
 long IRstop = 0x00FFE21D;                 / / Stop
 long IRcny70 = 0x00FFA857;                / / CNY70 self-propelled mode
 long IRAutorun = 0x00FF    906F    ;            / / Self-propelled mode ultrasound
 long IRturnsmallleft = 0x00FF22DD;
 / / ************************ Defined CNY70 pin ******************
*****************
 const int SensorLeft = 7;              / / Left sensor input pin
 const int SensorMiddle = 4;            / / The sensor input pin
 const int SensorRight = 3;             / / Right sensor input pin
 int SL;           / / Left sensor status
 int SM;           / / The sensor status
 int SR;           / / Right sensor status
 IRrecv irrecv (irReceiverPin);       / / Define an object to receive infrared signals IRrecv
 decode_results results;               / / Decoding results will result in structural variables in
decode_results
 / / ************************ Defined ultrasound pin ***************** ************
 int inputPin = 13; / / define pin ultrasonic signal receiver rx
 int outputPin = 12; / / define ultrasonic signal transmitter pin 'tx
 int Fspeedd = 0; / / in front of distance
 int Rspeedd = 0; / / the right distance
 int Lspeedd = 0; / / left distance
 int directionn = 0; / / = 8 post = 2 front left and right = 6 = 4
 Servo myservo; / / set myservo
```

```
int delay_time = 250; / / settling time after steering servo motors
int Fgo = 8; / / Forward
int Rgo = 6; / / turn right
int Lgo = 4; / / turn left
int Bgo = 2; / / reverse
/ / ********************************************** ****************** (SETUP)
void setup ()
{
   Serial.begin (9600);
   pinMode (MotorRight1, OUTPUT);       / / Pin 8 (PWM)
   pinMode (MotorRight2, OUTPUT);       / / Pin 9 (PWM)
   pinMode (MotorLeft1,       OUTPUT);       / / Pin 10 (PWM)
   pinMode (MotorLeft2,       OUTPUT);       / / Pin 11 (PWM)
   irrecv.enableIRIn ();             / / Start infrared decoding
    pinMode (SensorLeft, INPUT); / / define left Sensors
   pinMode (SensorMiddle, INPUT) ;/ / definition sensors
   pinMode (SensorRight, INPUT); / / definition of the right sensor
   digitalWrite (2, HIGH);
   pinMode (inputPin, INPUT); / / define ultrasound input pin
   pinMode (outputPin, OUTPUT); / / define ultrasonic output pin
   myservo.attach (9); / / define servo motor output section 5 pin (PWM)


   }
/ / ********************************************** ****************** (Void)
void advance (int a) / / Forward
{
            digitalWrite (MotorRight1, LOW);
            digitalWrite (MotorRight2, HIGH);
            digitalWrite (MotorLeft1, LOW);
             digitalWrite (MotorLeft2, HIGH);
            delay (a * 100);
}
void right (int b) / / turn right (single wheel)
{
          digitalWrite (MotorLeft1, LOW);
          digitalWrite (MotorLeft2, HIGH);
          digitalWrite (MotorRight1, LOW);
          digitalWrite (MotorRight2, LOW);
          delay (b * 100);
}
void left (int c) / / turn left (single wheel)
{
         digitalWrite (MotorRight1, LOW);
         digitalWrite (MotorRight2, HIGH);
         digitalWrite (MotorLeft1, LOW);
         digitalWrite (MotorLeft2, LOW);
         delay (c * 100);
}
void turnR (int d) / / turn right (wheel)
{
         digitalWrite (MotorRight1, HIGH);
         digitalWrite (MotorRight2, LOW);
         digitalWrite (MotorLeft1, LOW);
```

```
          digitalWrite (MotorLeft2, HIGH);
          delay (d * 100);
 }
 void turnL (int e) / / turn left (wheel)
 {
          digitalWrite (MotorRight1, LOW);
          digitalWrite (MotorRight2, HIGH);
          digitalWrite (MotorLeft1, HIGH);
          digitalWrite (MotorLeft2, LOW);
          delay (e * 100);
 }
 void stopp (int f) / / Stop
 {
         digitalWrite (MotorRight1, LOW);
         digitalWrite (MotorRight2, LOW);
         digitalWrite (MotorLeft1, LOW);
         digitalWrite (MotorLeft2, LOW);
         delay (f * 100);
 }
 void back (int g) / / Check out
 {
              digitalWrite (MotorRight1, HIGH);
              digitalWrite (MotorRight2, LOW);
              digitalWrite (MotorLeft1, HIGH);
              digitalWrite (MotorLeft2, LOW);;
              delay (g * 100);
 }
 void detection () / / measure three angles (front Left. Right)
 {
       int delay_time = 250; / / settling time after steering servo motors
       ask_pin_F (); / / read from front

       if (Fspeedd <10) / / if the distance is less than 10 cm in front of
     {
          stopp (1); / / clear the output data
          back (2); / / Check out 0.2 seconds
     }
       if (Fspeedd <25) / / if the distance is less than 25 cm in front of
     {
          stopp (1); / / clear the output data
          ask_pin_L (); / / read from left
          delay (delay_time); / / wait for stable servo motor
          ask_pin_R (); / / read the right distance
          delay (delay_time); / / wait for stable servo motor

          if (Lspeedd> Rspeedd) / / If the distance is greater than the right from the left
        {
          directionn = Lgo; / / go left
        }

          if (Lspeedd <= Rspeedd) / / if the distance is less than or equal to the left to the right
distance
          {
          directionn = Rgo; / / go right
```

```
          }

          if (Lspeedd <15 && Rspeedd <15) / / if the distance to the left and right are less than 10
cm distance
         {
             directionn = Bgo; / / to go after
          }
        }
        else / / add as greater than 25 cm in front of
      {
          directionn = Fgo; / / to move forward
      }
  }
  / / ********************************************
*******************************
  void ask_pin_F () / / Measure the distance from the front
  {
  myservo.write (90);
  digitalWrite (outputPin, LOW); / / make ultrasonic transmitter low voltage 2 μ s
  delayMicroseconds (2);
  digitalWrite (outputPin, HIGH); / / make ultrasonic transmitting high voltage 10 μ s, where at
least 10 μ s
  delayMicroseconds (10);
  digitalWrite (outputPin, LOW); / / maintain low voltage ultrasonic transmitter
  float Fdistance = pulseIn (inputPin, HIGH); / / read worse time difference
  Fdistance = Fdistance/5.8/10; / / will turn to time distance (unit: cm)
  Serial.print ("F distance:"); / / output distance (unit: cm)
  Serial.println (Fdistance); / / display the distance
  Fspeedd = Fdistance; / / will enter Fspeedd (former speed) from Reading
  }
  / / ********************************************
*******************************
  void ask_pin_L () / / Measure the distance from the left
  {
  myservo.write (177);
  delay (delay_time);
  digitalWrite (outputPin, LOW); / / make ultrasonic transmitter low voltage 2 μ s
  delayMicroseconds (2);
  digitalWrite (outputPin, HIGH); / / make ultrasonic transmitting high voltage 10 μ s, where at
least 10 μ s
  delayMicroseconds (10);
  digitalWrite (outputPin, LOW); / / maintain low voltage ultrasonic transmitter
  float Ldistance = pulseIn (inputPin, HIGH); / / read worse time difference
  Ldistance = Ldistance/5.8/10; / / will turn to time distance (unit: cm)
  Serial.print ("L distance:"); / / output distance (unit: cm)
  Serial.println (Ldistance); / / display the distance
  Lspeedd = Ldistance; / / will be read into the distance Lspeedd (left-speed)
  }
  / / ********************************************
*******************************
  void ask_pin_R () / / Measure the distance from the right
  {
  myservo.write (5);
  delay (delay_time);
```

```
  digitalWrite (outputPin, LOW); / / make ultrasonic transmitter low voltage 2 μ s
  delayMicroseconds (2);
  digitalWrite (outputPin, HIGH); / / make ultrasonic transmitting high voltage 10 μ s, where at
least 10 μ s
  delayMicroseconds (10);
  digitalWrite (outputPin, LOW); / / maintain low voltage ultrasonic transmitter
  float Rdistance = pulseIn (inputPin, HIGH); / / read worse time difference
  Rdistance = Rdistance/5.8/10; / / will turn to time distance (unit: cm)
  Serial.print ("R distance:"); / / output distance (unit: cm)
  Serial.println (Rdistance); / / display the distance
  Rspeedd = Rdistance; / / will be read into the distance Rspeedd (Right-speed)
  }
  / / **********************************************
***************************** (LOOP)
  void loop ()
  {
            SL = digitalRead (SensorLeft);
           SM = digitalRead (SensorMiddle);
           SR = digitalRead (SensorRight);
           performCommand ();
  / / ********************************************** **************************
normal remote mode
     if (irrecv.decode (& results))
          {                  / / Decoding is successful, you receive a set of infrared signals
  / ********************************************** ********************** /
           if (results.value == IRfront) / / Forward
            {
              advance (10) ;/ / forward
            }
  / ********************************************** ********************** /
           if (results.value ==       IRback) / / Check out
            {
              back (10) ;/ / after retirement
            }
  / ********************************************** ********************** /
           if (results.value == IRturnright) / / turn right
           {
              right (6); / / turn right
           }
  / ********************************************** ********************** /
          if (results.value == IRturnleft) / / turn left
          {
             left (6); / / turn left);
          }
  / ********************************************** ********************** /
        if (results.value == IRstop) / / Stop
      {
          digitalWrite (MotorRight1, LOW);
          digitalWrite (MotorRight2, LOW);
          digitalWrite (MotorLeft1, LOW);
          digitalWrite (MotorLeft2, LOW);
        }
  / / ********************************************** **********************
cny70 model black self-propelled mode: LOW White:
```

```
  if (results.value == IRcny70)
{
   while (IRcny70)
   {
      SL = digitalRead (SensorLeft);
      SM = digitalRead (SensorMiddle);
      SR = digitalRead (SensorRight);

      if (SM == HIGH) / / in sensors in black areas
      {
          if (SL == LOW & SR == HIGH) / / left and right black white, turn left
          {
              digitalWrite (MotorRight1, LOW);
              digitalWrite (MotorRight2, HIGH);
              analogWrite (MotorLeft1, 0);
              analogWrite (MotorLeft2, 80);
          }
            else if (SR == LOW & SL == HIGH) / / left and right black white, turn right
          {
              analogWrite (MotorRight1, 0) ;/ / right turn
              analogWrite (MotorRight2, 80);
              digitalWrite (MotorLeft1, LOW);
              digitalWrite (MotorLeft2, HIGH);
          }
          else       / / Both sides white, straight
          {
              digitalWrite (MotorRight1, LOW);
              digitalWrite (MotorRight2, HIGH);
              digitalWrite (MotorLeft1, LOW);
              digitalWrite (MotorLeft2, HIGH);
              analogWrite (MotorLeft1, 200);
              analogWrite (MotorLeft2, 200);
              analogWrite (MotorRight1, 200);
              analogWrite (MotorRight2, 200);
        }
      }
      else / / the sensors in the white area
     {
          if (SL == LOW & SR == HIGH) / / left and right black white, fast turn left
       {
              digitalWrite (MotorRight1, LOW);
            digitalWrite (MotorRight2, HIGH);
            digitalWrite (MotorLeft1, LOW);
            digitalWrite (MotorLeft2, LOW);
       }
         else if (SR == LOW & SL == HIGH) / / left and right black white, quick right turn
       {
           digitalWrite (MotorRight1, LOW);
           digitalWrite (MotorRight2, LOW);
           digitalWrite (MotorLeft1, LOW);
           digitalWrite (MotorLeft2, HIGH);
       }
         else / / are white, stop
       {
```

```
            digitalWrite (MotorRight1, HIGH);
            digitalWrite (MotorRight2, LOW);
            digitalWrite (MotorLeft1, HIGH);
            digitalWrite (MotorLeft2, LOW);;
            }
          }
           if (irrecv.decode (& results))
           {
                    irrecv.resume ();
                            Serial.println (results.value, HEX);
                     if (results.value == IRstop)
                    {
                       digitalWrite (MotorRight1, HIGH);
                       digitalWrite (MotorRight2, HIGH);
                       digitalWrite (MotorLeft1, HIGH);
                       digitalWrite (MotorLeft2, HIGH);
                       break;
                    }
           }
         }
          results.value = 0;
      }
 / / ********************************************** self-propelled mode ultrasound
**********************
     if (results.value == IRAutorun)
          {
                while (IRAutorun)
            {
                  myservo.write (90); / / return to the pre-prepared so that the servo motor
position once the measure under preparation
                 detection (); / / measure the angle and direction of judgment to where to move
                   if (directionn == 8) / / If directionn (direction) = 8 (forward)
                 {
                    if (irrecv.decode (& results))
                {
                   irrecv.resume ();
                   Serial.println (results.value, HEX);
                   if (results.value == IRstop)
                   {
                      digitalWrite (MotorRight1, LOW);
                      digitalWrite (MotorRight2, LOW);
                      digitalWrite (MotorLeft1, LOW);
                      digitalWrite (MotorLeft2, LOW);
                      break;
                   }
                }
                        results.value = 0;
                        advance (1); / / normal forward
                        Serial.print ("Advance"); / / display direction (forward)
                        Serial.print ("");
                   }
                 if (directionn == 2) / / If directionn (direction) = 2 (reverse)
               {
                   if (irrecv.decode (& results))
```

```
 {
    irrecv.resume ();
    Serial.println (results.value, HEX);
    if (results.value == IRstop)
    {
       digitalWrite (MotorRight1, LOW);
       digitalWrite (MotorRight2, LOW);
       digitalWrite (MotorLeft1, LOW);
       digitalWrite (MotorLeft2, LOW);
       break;
    }
 }
      results.value = 0;
        back (8); / / reverse (car)
      turnL (3); / / move slightly to the left (to prevent stuck in dead alley)
      Serial.print ("Reverse"); / / display direction (backwards)
}
   if (directionn == 6) / / If directionn (direction) = 6 (right turn)
{
  if (irrecv.decode (& results))
 {
        irrecv.resume ();
      Serial.println (results.value, HEX);
    if (results.value == IRstop)
    {
       digitalWrite (MotorRight1, LOW);
       digitalWrite (MotorRight2, LOW);
       digitalWrite (MotorLeft1, LOW);
       digitalWrite (MotorLeft2, LOW);
       break;
    }
 }
    results.value = 0;
       back (1);
       turnR (6); / / turn right
       Serial.print ("Right"); / / display direction (turn left)
}
   if (directionn == 4) / / If directionn (direction) = 4 (turn left)
{
    if (irrecv.decode (& results))
 {
    irrecv.resume ();
    Serial.println (results.value, HEX);
    if (results.value == IRstop)
      {
       digitalWrite (MotorRight1, LOW);
       digitalWrite (MotorRight2, LOW);
       digitalWrite (MotorLeft1, LOW);
       digitalWrite (MotorLeft2, LOW);
       break;
    }
 }
          results.value = 0;
         back (1);
```

```
                turnL (6); / / turn left
                Serial.print ("Left"); / / display direction (turn right)
            }

              if (irrecv.decode (& results))
            {
              irrecv.resume ();
              Serial.println (results.value, HEX);
              if (results.value == IRstop)
              {
                digitalWrite (MotorRight1, LOW);
                digitalWrite (MotorRight2, LOW);
                digitalWrite (MotorLeft1, LOW);
                digitalWrite (MotorLeft2, LOW);
                break;
              }
            }
          }
                results.value = 0;
        }
/ ********************************************** ********************** /
       else
     {
            digitalWrite (MotorRight1, LOW);
            digitalWrite (MotorRight2, LOW);
            digitalWrite (MotorLeft1, LOW);
            digitalWrite (MotorLeft2, LOW);
      }


          irrecv.resume ();          / / Continue to accept a set of infrared signals
   }
}

void performCommand () {
  if (Serial.available ()) {
      val = Serial.read ();
  }
    if (val == 'f') {/ / Forward
      advance (10);
    } Else if (val == 'z') {/ / Stop Forward
      stopp (10);
    } Else if (val == 'b') {/ / Backward
      back (10);
    } Else if (val == 'y') {/ / Stop Backward
        back (10);
    }       else if (val == 'l') {/ / Right
      turnR (10);
    } Else if (val == 'r') {/ / Left
      turnL (10);
    } Else if (val == 'v') {/ / Stop Turn
      stopp (10);
    } Else if (val == 's') {/ / Stop
      stopp (10);
```

```
        }

}
```

Trademark Notice:

Robotale and graphics Easy Interactive Technology Co., Ltd. is a branch registered trademark. Based on the continuous improvement and upgrading of products, the company changed at any time and where the information or products mentioned without notice. Without our prior written consent or authorization, can not arbitrarily theft, copying, publishing partial description of the product or the entire contents.

Disclaimer:

Users do not use this product for any application (such as experimental, contests, secondary development), users at their own risk. Company for direct, indirect or consequential damage arising from the use of this product (including loss of personal safety, profit loss of credibility, etc.) assumes no responsibility for children under 14 years old must be conducted using the product accompanied by an adult associated experiment.

Errata Description:

To be able to convey the right to use the product information, we spend a lot of time and effort on this manual, you want users to be able to carefully read the contents, but inevitably there are omissions. If errors are found, welcome to contact us by e-mail jmddz925@126.com. To make the manual more perfect, providing the latest and most detailed information, we will continue to improve the content of the manual supplement.