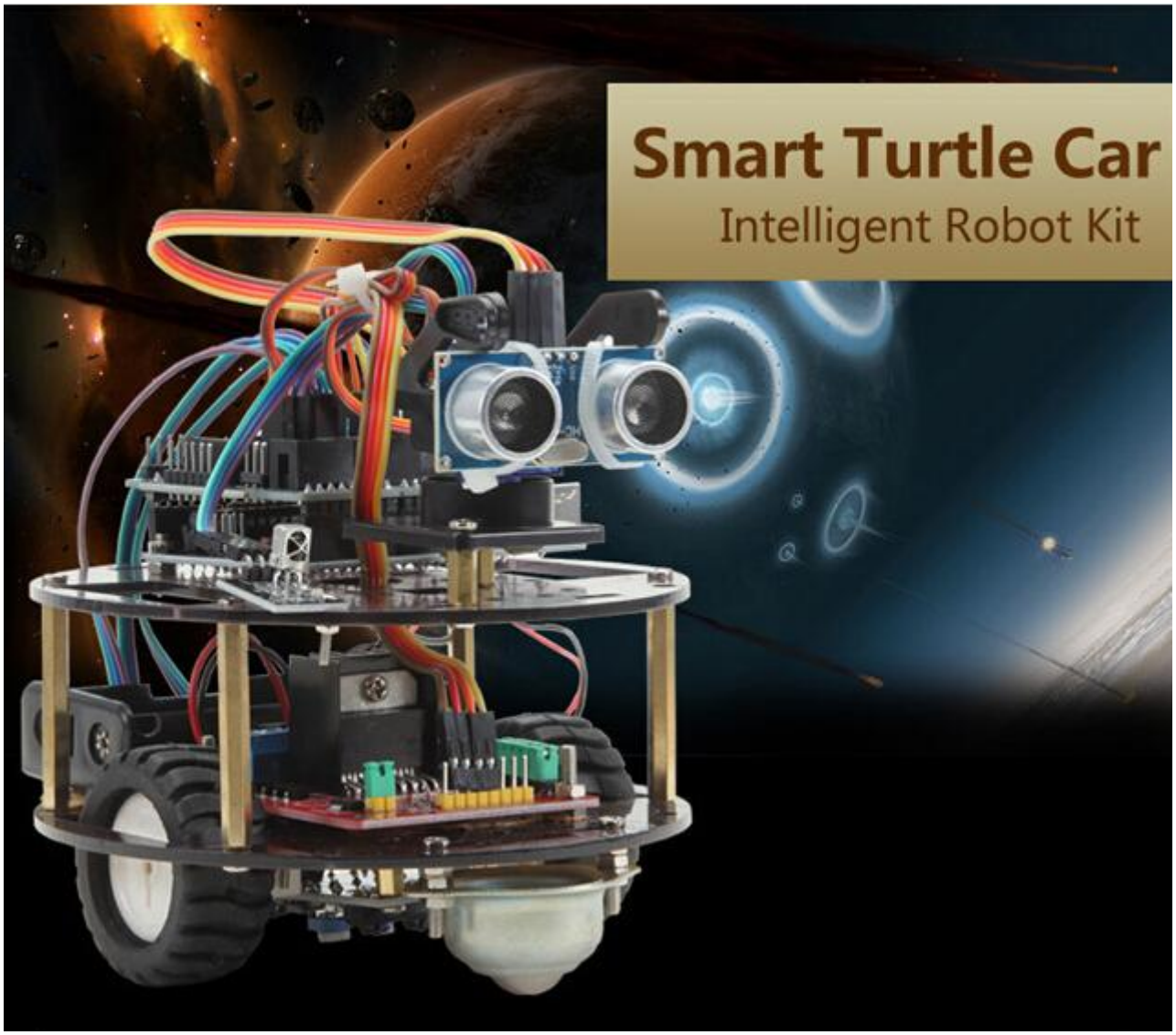
Arduino intelligent turtle (smart car) manual



1. Introduction

ARDUINO intelligent turtle is a learning application development system of microcontroller. It's based on ARDUINO microcontroller series atmega-328. It has functions such as line tracking, obstacle avoidance, infrared remote control and wireless remote control. This kit contains many interesting programs. It can also be expanded to have external circuit module to have other functions. This kit is designed to help you interestingly learn Arduino. You can learn Arduino MCU development ability while having fun.

2.Parameters

1. Motor parameters: voltage range: 1.5-12V; motor shaft length: 10 mm; revolving speed: 6.0 V 100 RPM/min.
2. Use L298N driver module for motor control, true separation from microcontroller.
3. Three line tracking modules, detecting black and white line, high precision, can also be used in fall prevention.
4. Infrared remote communication module makes up the remote control system.
5. Ultrasonic module makes up the obstacle avoidance system.
5. With bluetooth wireless module, remotely control the robot after bluetooth pairing with mobile phone.
6. Can be used under external 7 ~ 12V voltage, also with various sensor modules to realize various functions.

3. Lesson introduction

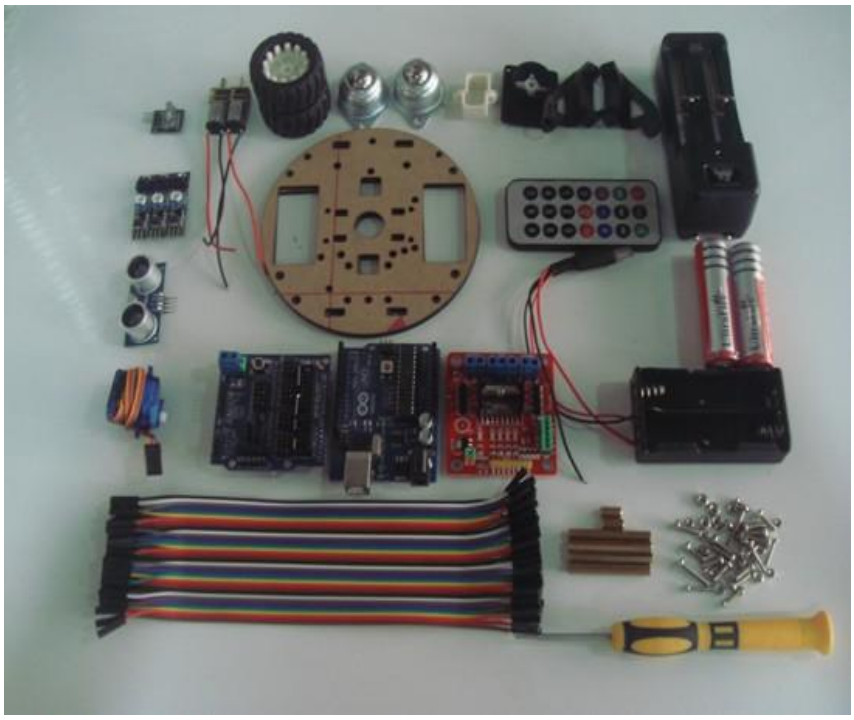
1. Application of L298N motor driver board
2. Tracking smart car
3. Ultrasonic obstacle avoidance smart car
4. Infrared remote control smart car
5. Arduino bluetooth remote control programmable smart car
6. 4 in 1 (line tracking; obstacle avoidance; Infrared remote control; bluetooth remote control) multi-functional program

4. Component list

1. 2* metal gear motor
2. 2* high-quality wheel
3. 2* motor fixed part
4. 2* Bull’s eye omni-directional wheel
5. 2* robot chassis
6. 1* L298N motor driver board

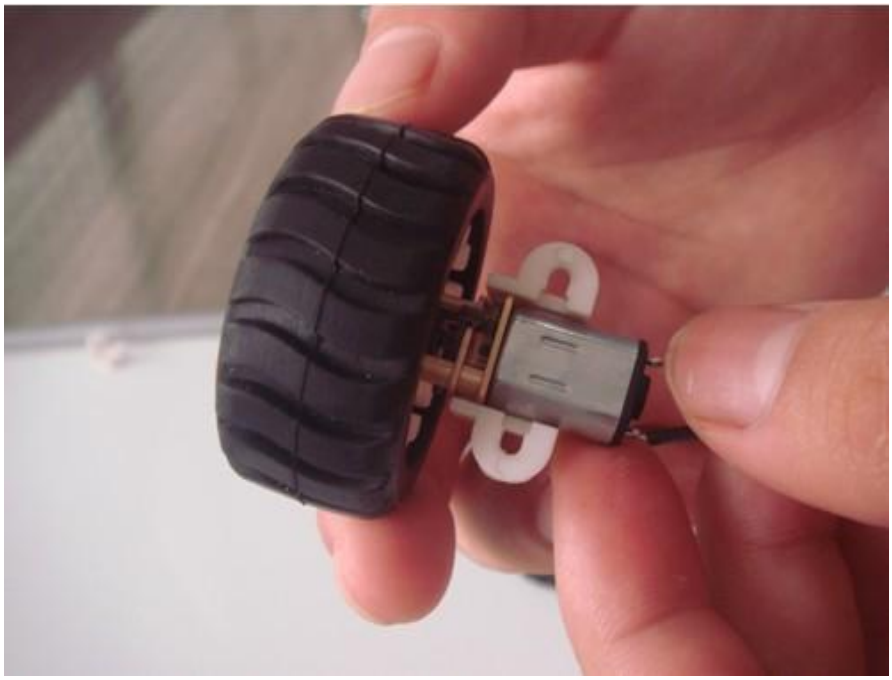
7. 1* ARDUINO UNO328 board
8. 1* ARDUINO sensor shield
9. 1* holder
10. 1* servo motor
11. 1* ultrasonic module
12. 3* line tracking module
13. Infrared receiving sensor
14. MCU remote controller
15. 2 cells of 2000MA 18650 rechargeable batteries
16. 1* 18650 battery case
17. 1* 18650 charger
18. 40* Dupont lines
19. 4* copper nuts with 35MM length; 2* copper nuts with10MM length
20. Several 3MM screws and nuts

5. Installation instruction

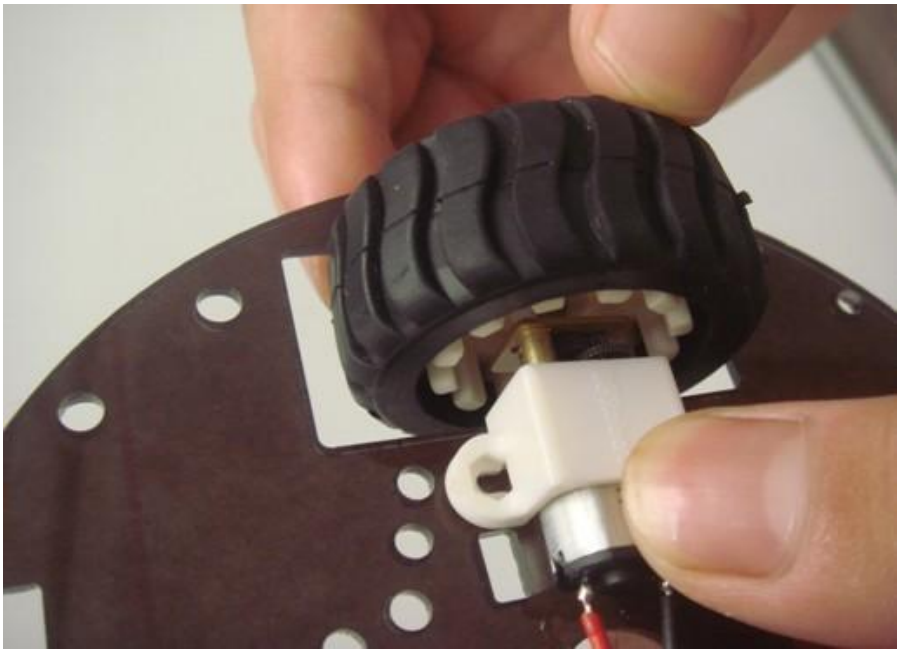


Step 1: insert the shaft of N20 gear motor into the motor hole of the wheel

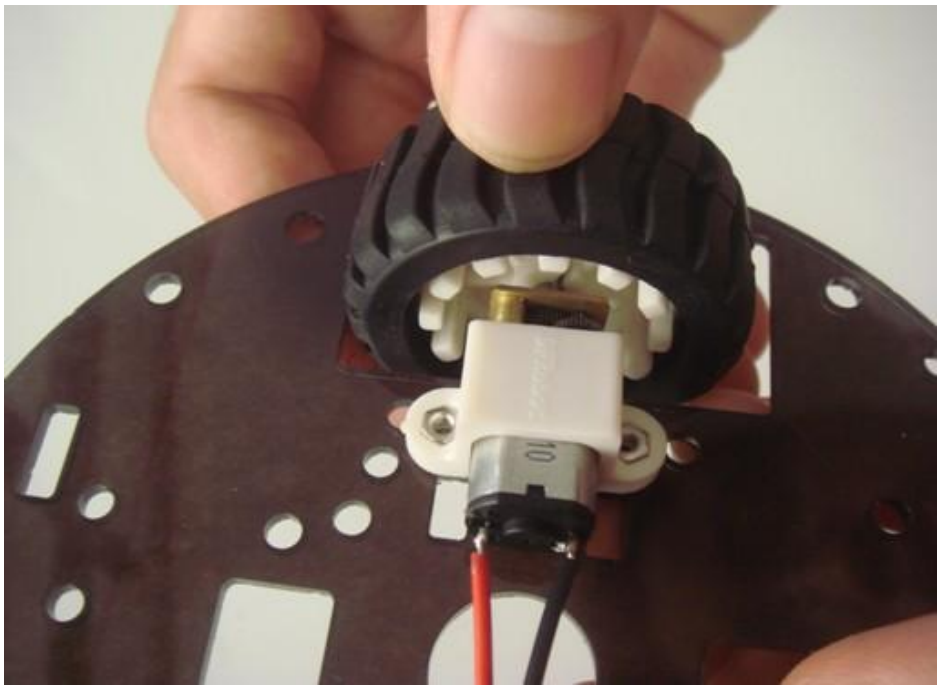
Step 2: install the white N20 motor fixed part to the motor, pay attention to the inner groove of the fixed part. It matches the motor



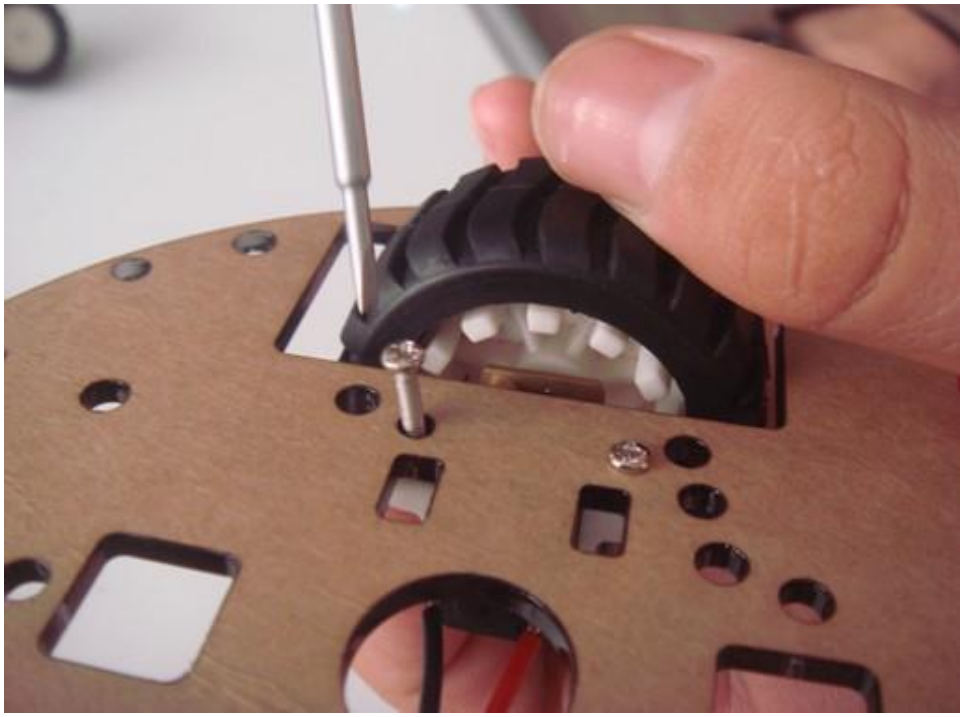
Step 3: align the fixed part's mounting hole with those in the car chassis



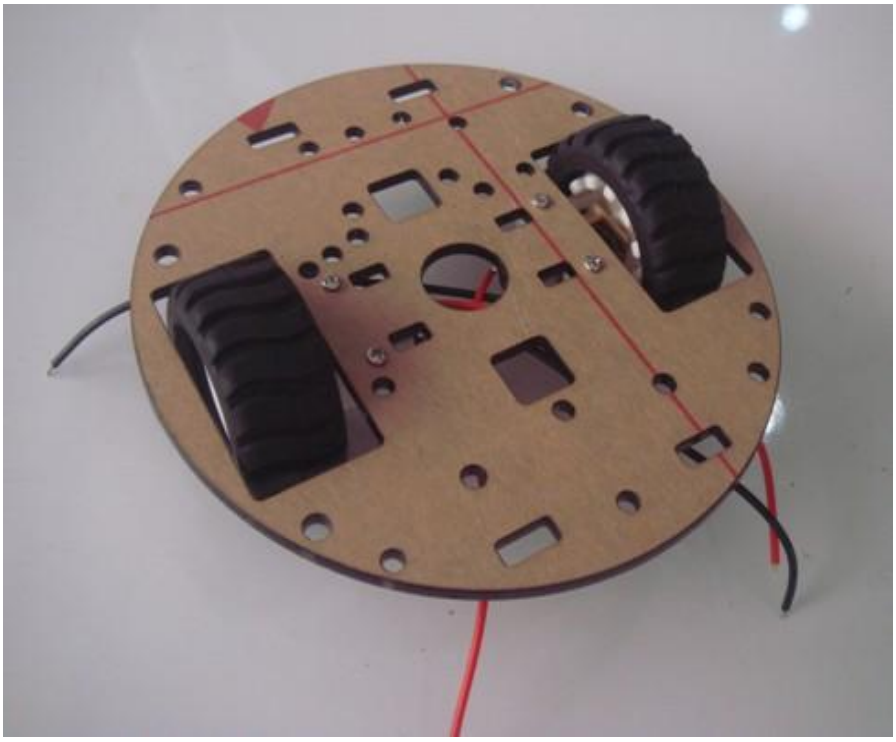
Step 4: place the M2 nut in the mounting hole



Step 5: fix the motor holder with screws

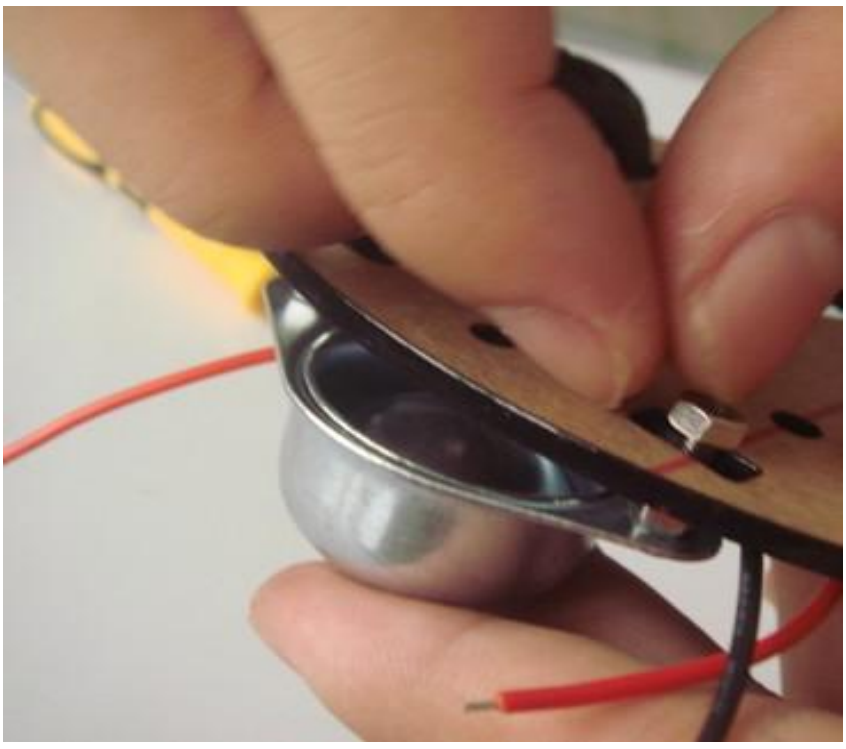


Step 6: install the other half with the same method

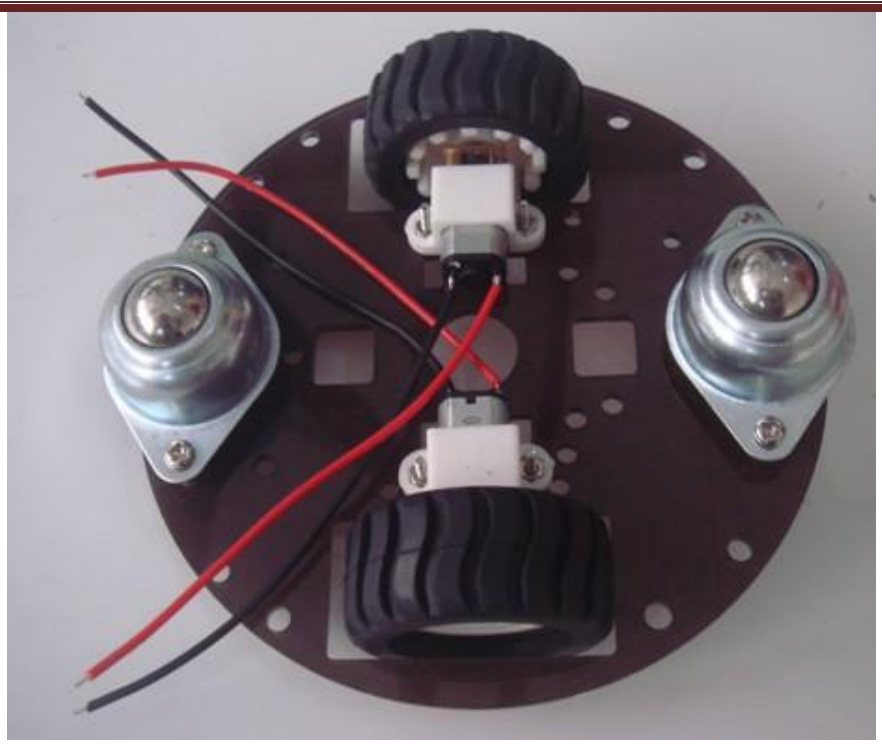


Step 7: place 1 M3*12 screw and 2 M3 nuts in the Bull's eye omni-directional wheel

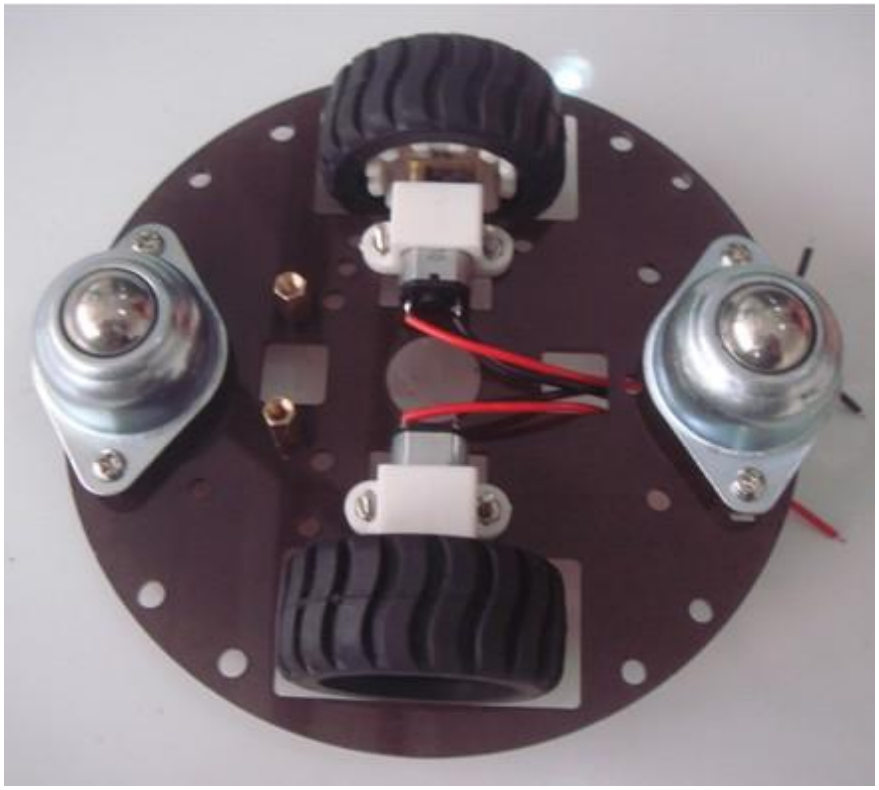
Step 8: fix it to the car chassis with the fastening screw



Step 9: install the other half with the same method

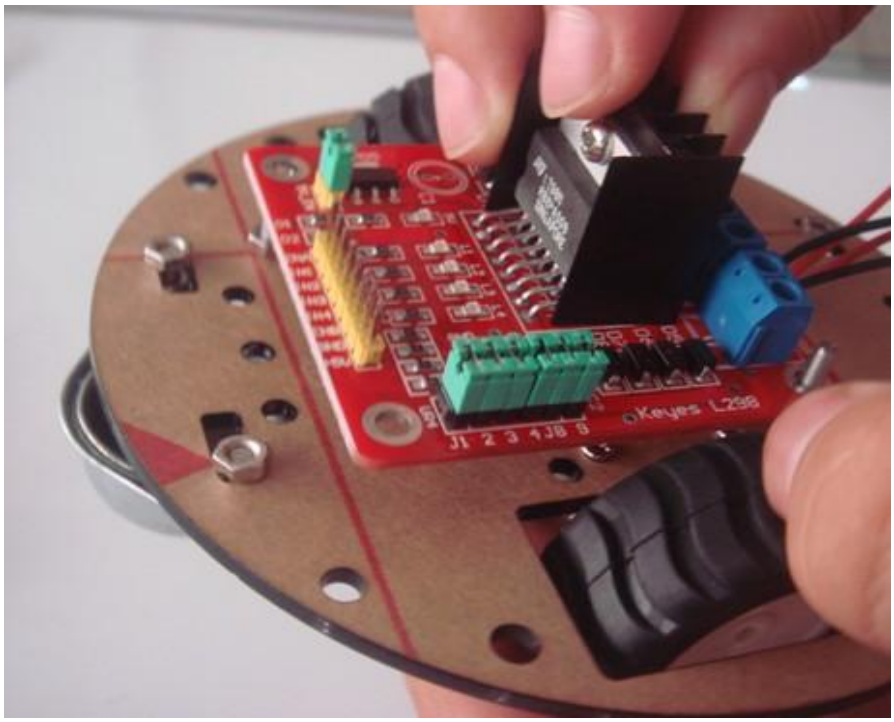


Step 10: install the fastening copper nuts for the line tracking sensor

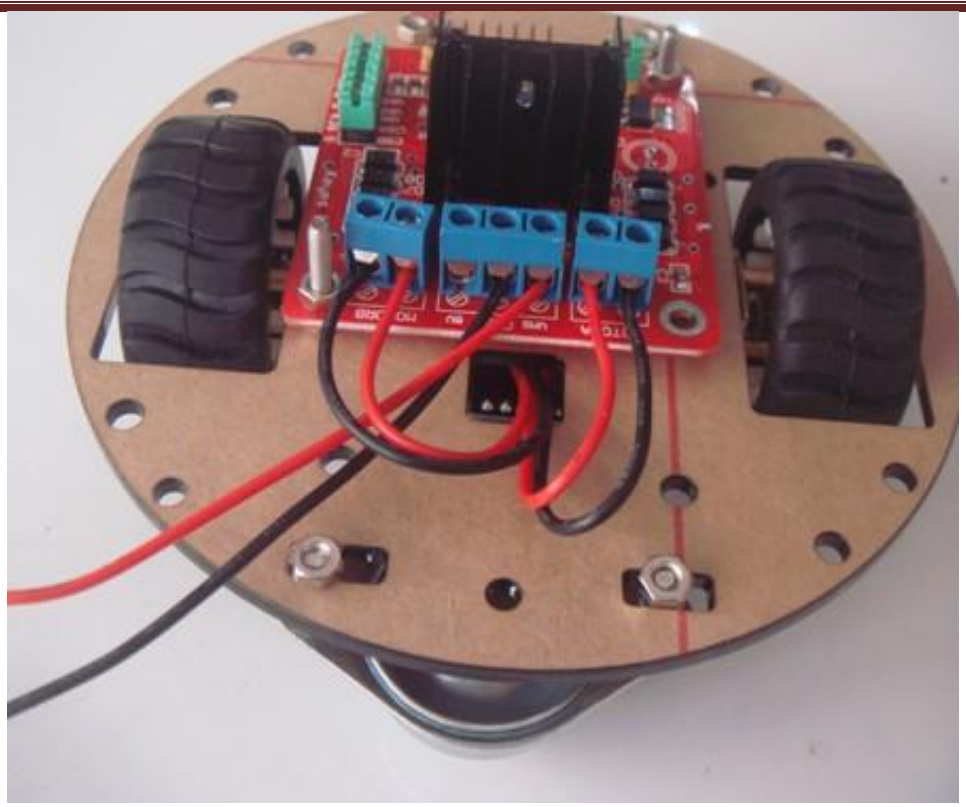


Step 11: install 3 line tracking modules, fasten with M3 screw

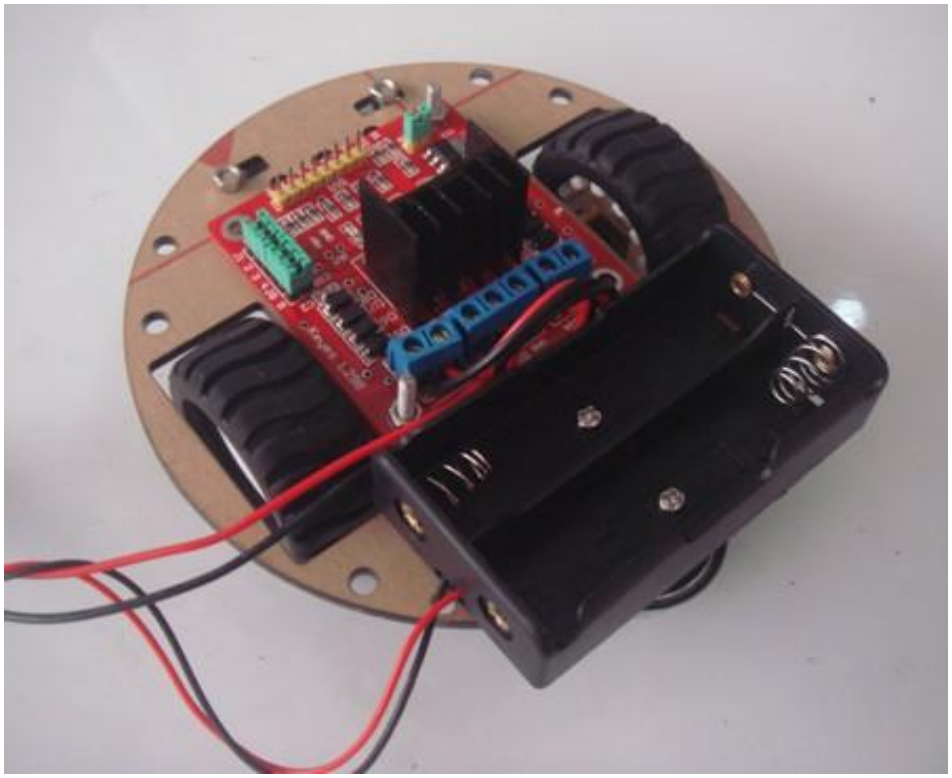
Step 12: install L298N motor driver module



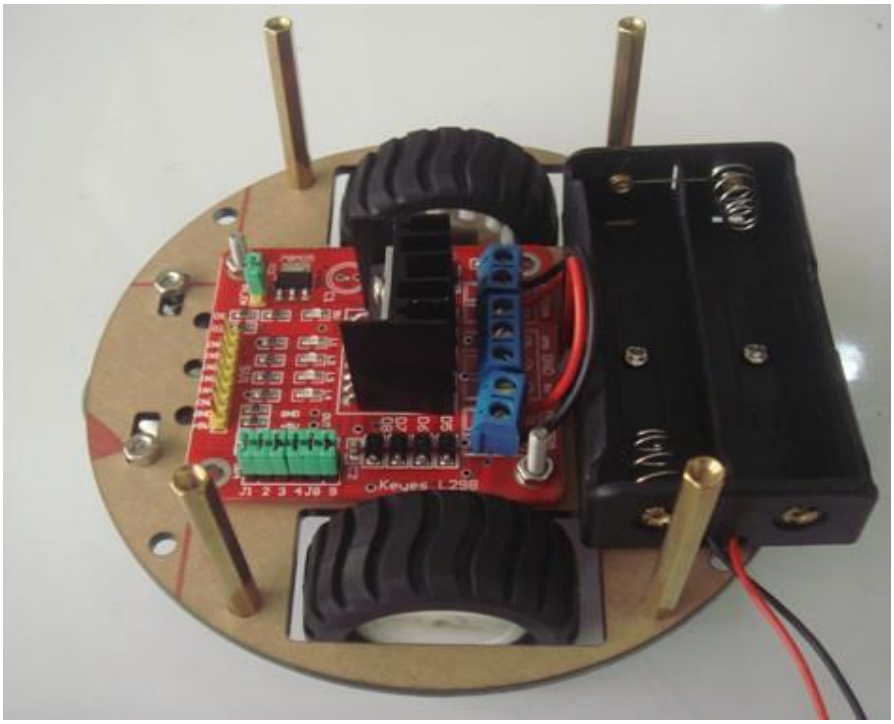
Step 13: correctly connect the wires of the motor and the battery power supply. + of the battery case connected to the VMS of the 298N, - to 298N GND



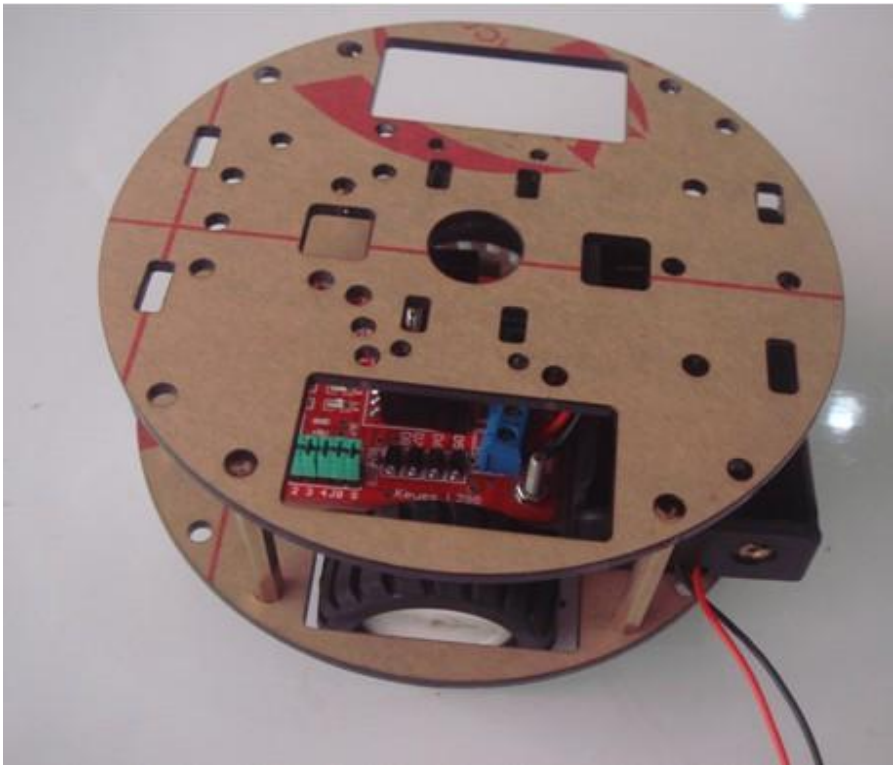
Step 14: install 18650 battery case and fasten with screw



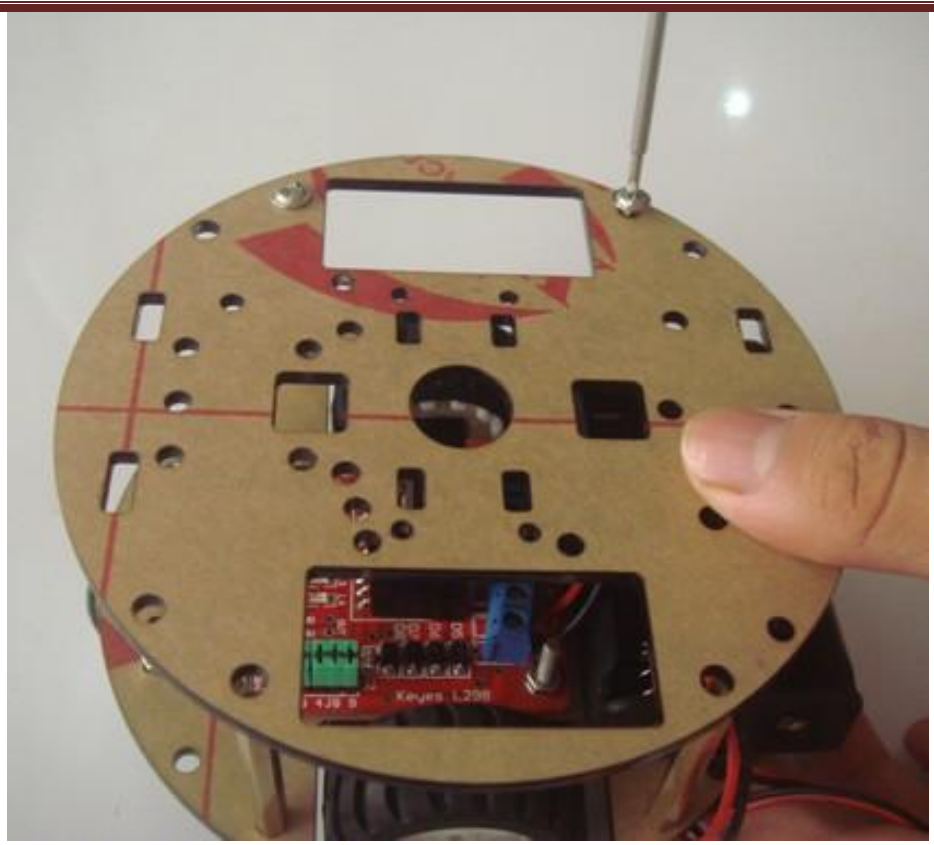
Step 15: install 4 copper nuts between two layers



Step 16: align with the holes of the top layer



Step 17: add fastening screws



Step 18: obstacle avoidance part

云台舵机超声波安装图

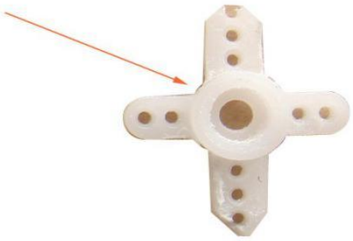
Step 19: servo motor

Take out the cross-shape plastic part from the servo accessory bag

将十字减成四边对等的长度，打磨

Grind the four arms into equal length

成宽度一样



Step 20:

Fix the 2*8MM and 1.2*5MM in the cross-shape part and install it to the holder bottom as picture shows.

如图将2*8MM的和1.2*5MM的螺丝安装到十字的第二个孔装到云台底座上

Step 21:

In the holder bottom, install nuts to the screw

云台底部2*8MM的螺丝位套上螺母

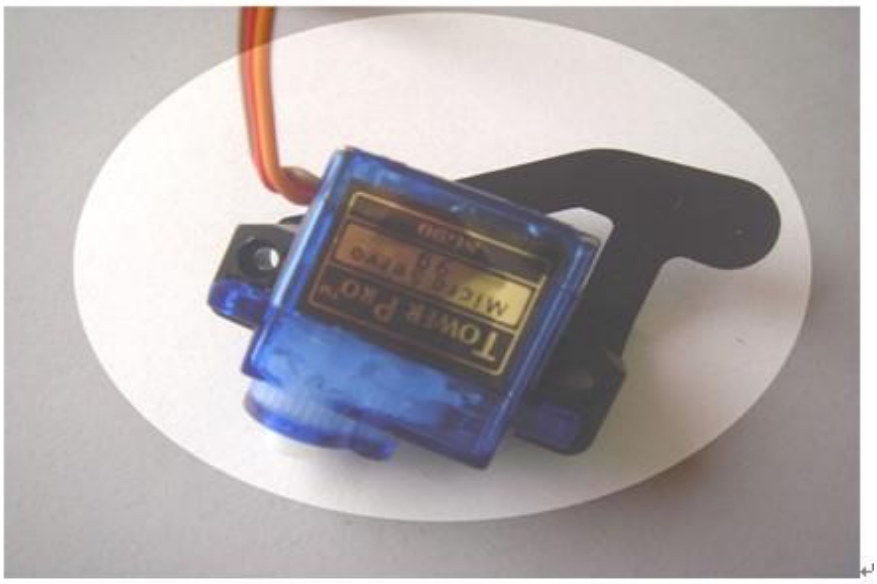
Apply heat staking glue to fix the screw

点上热溶胶固定螺丝位

Step 22:

Install the servo motor to the two sides of the holder

将舵机装上云台的两个边界



Step 23:

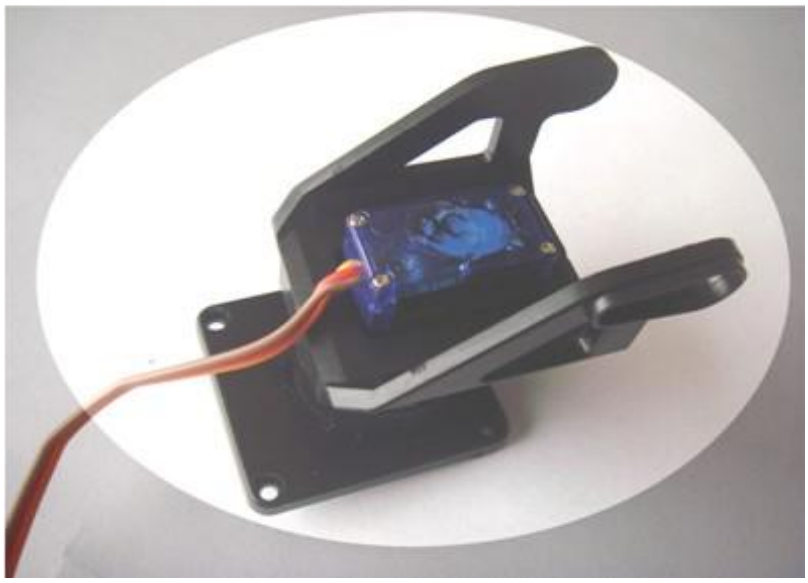
After the above installation, fix it by screws

边羿装好后，装好螺丝紧固

Step 24:

Place the installed Servo motor into the cross-shape plastic part and adjust the direction

将装好的舵机放进固定好的十字胶体
调整好方向



Step 25:

Take out the 2*6MM screw from the servo accessory bag and install it to the below indicated hole

从舵机元件包里取出2*6MM的螺丝

安装到舵机固定孔中

Installation hole

舵机固定孔

Step 26:

Use cable to fix the ultrasonic module to the holder front

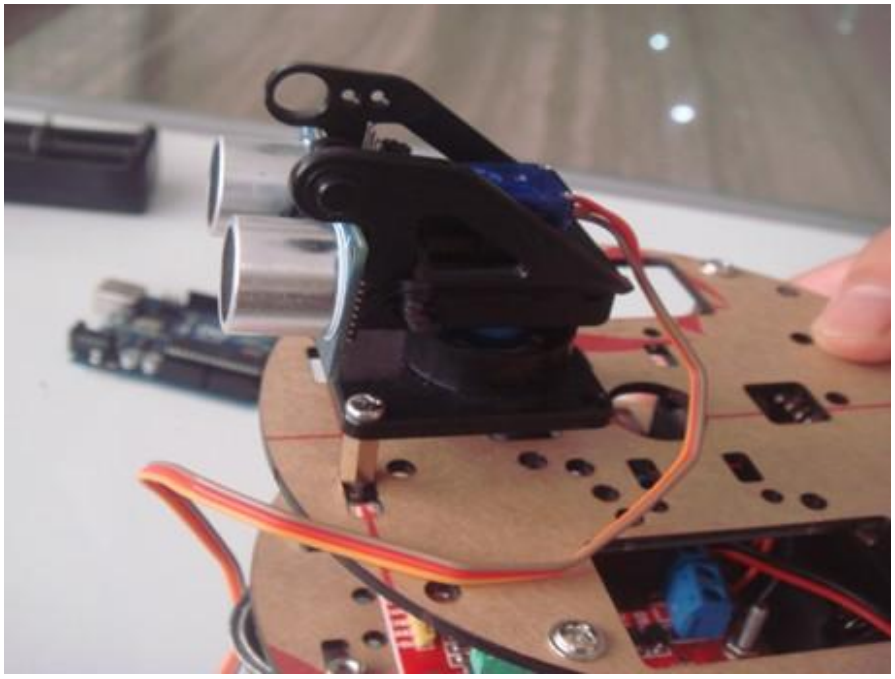
用扎带将超声波模块固定在云台前端

Step 27:

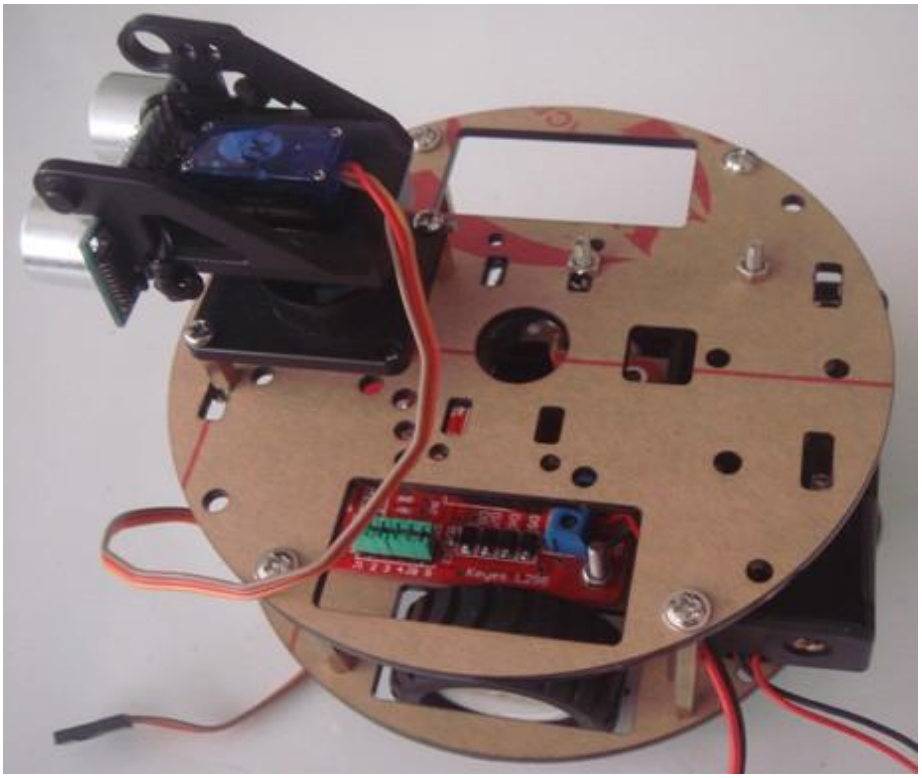
Place the 6MM copper nut into the installation holes on the holder bottom

将6MM铜柱装在云台底座安装孔

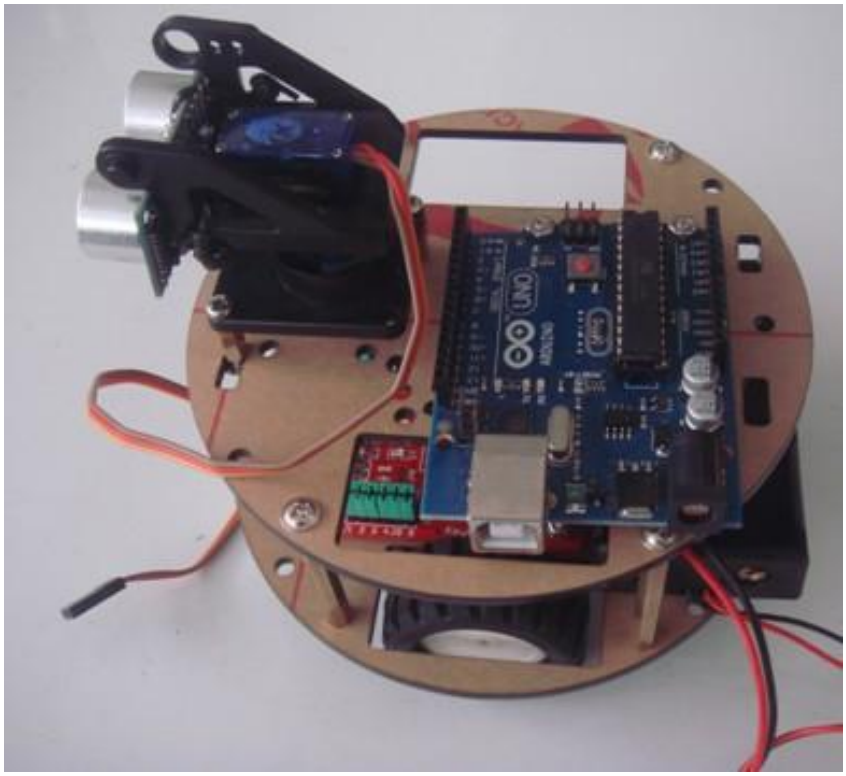
Step 28: install the installed servo holder onto the mounting holes on the chassis



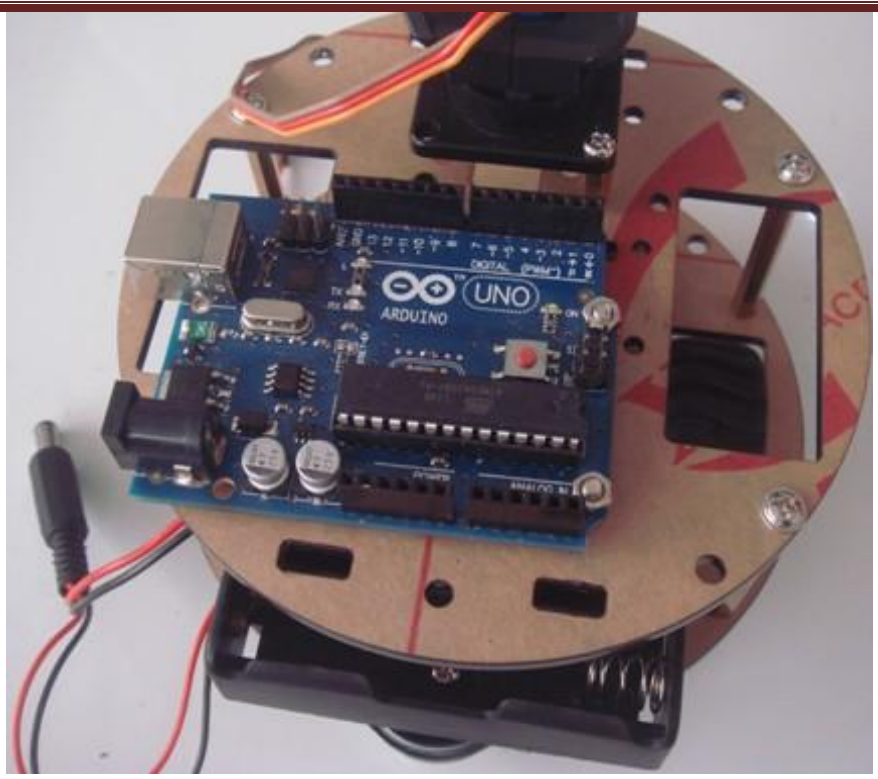
Step 29: install the retaining screw of the main controller board



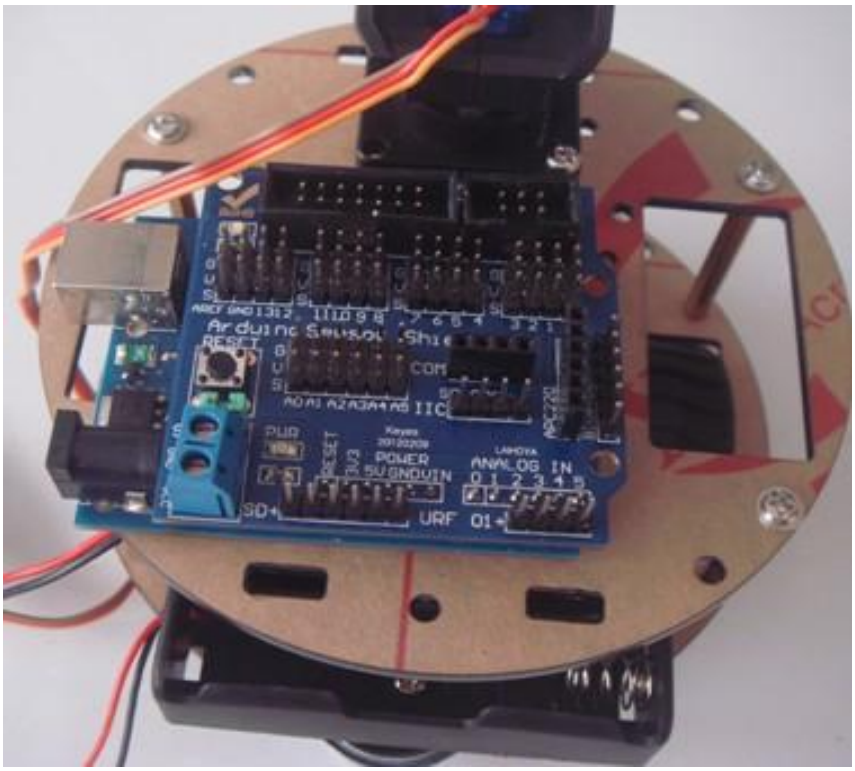
Step 30: fix the ARDUINO



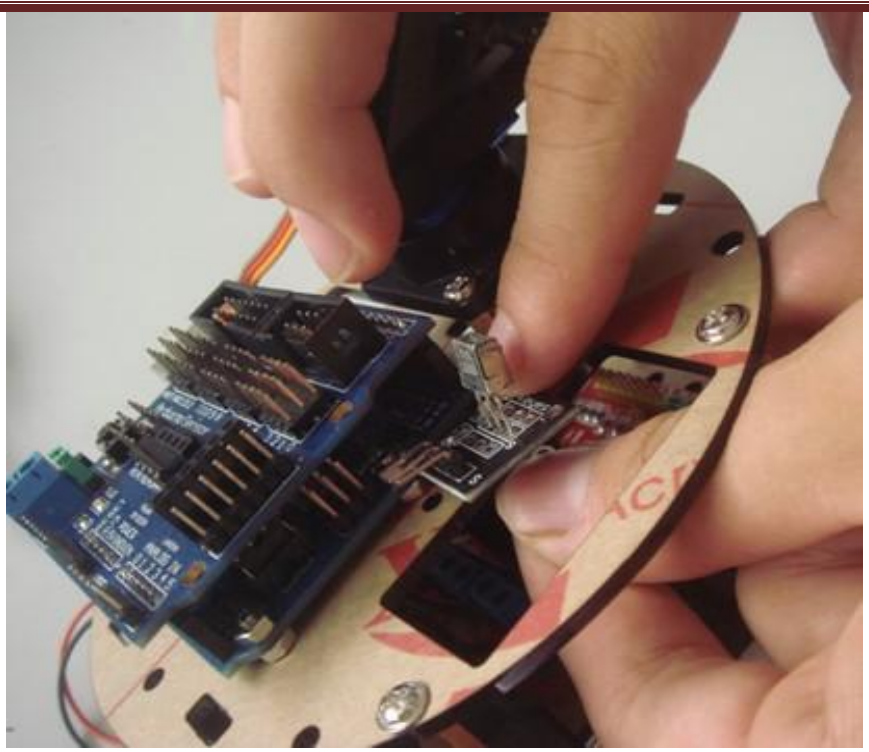
Step 31: picture of the main controller's position



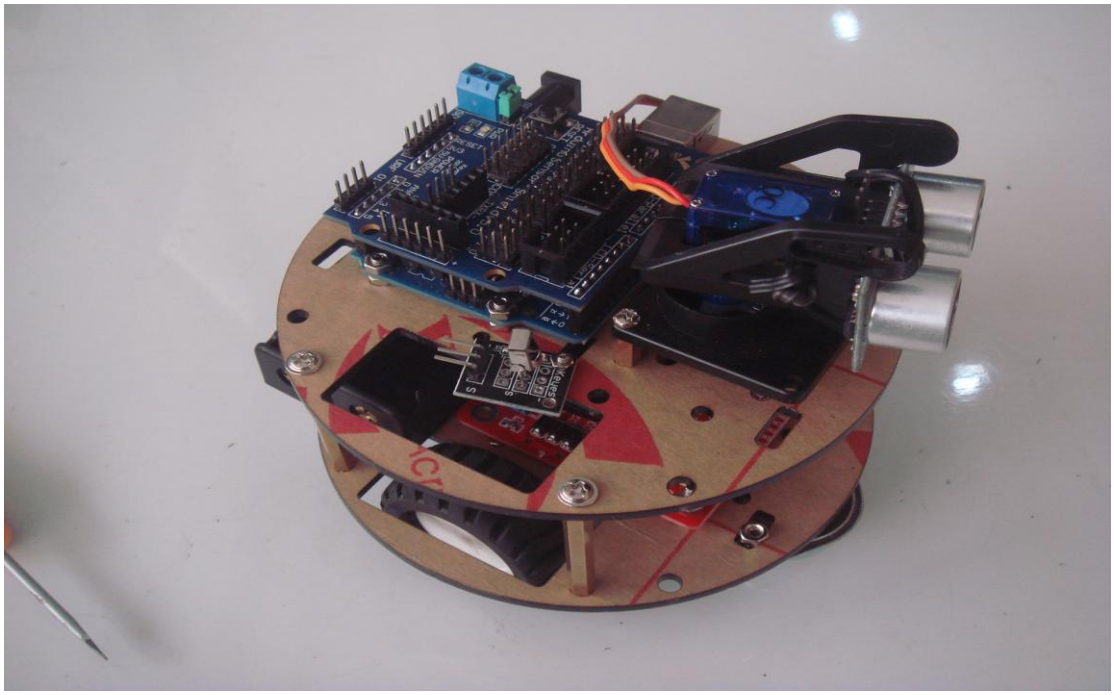
Step 32: place the sensor shield on top of the ARDUINO main control board.



Step 33: install the infrared receiver

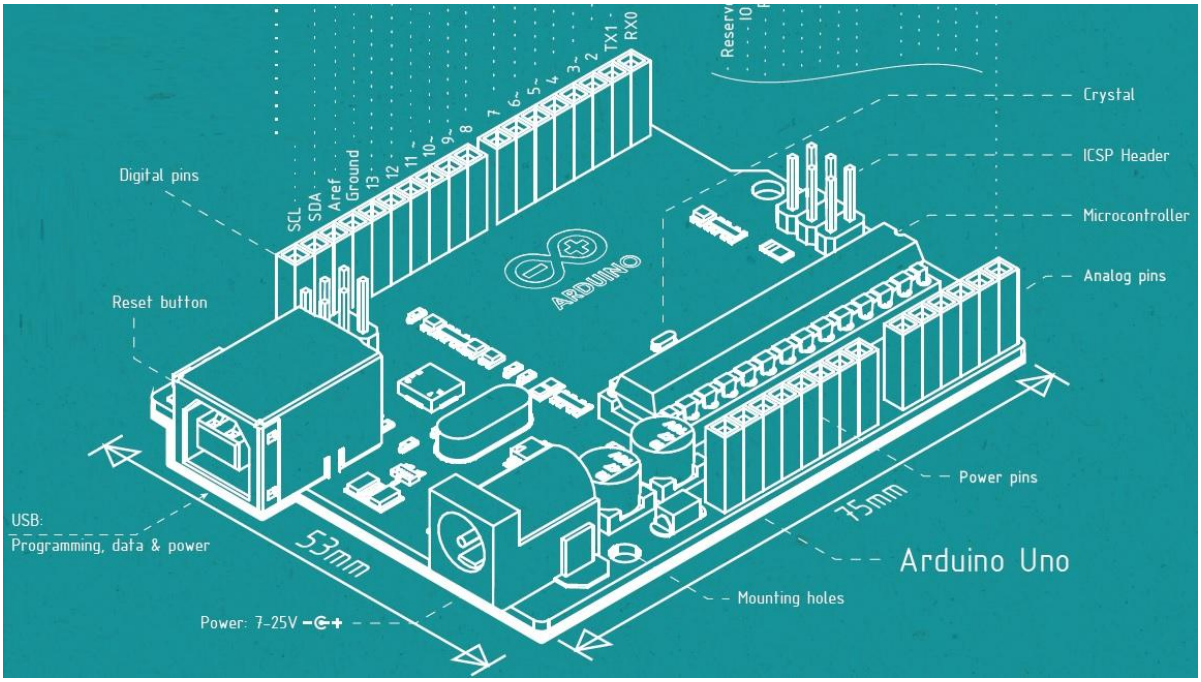


Step 34: installation completed. Thank you!



6. Application of Arduino

1. Introduction



Arduino is an open-source hardware project platform from Italy. This platform includes a circuit board with simple I/O function and program development environment software. It can be used to develop interactive products. For example, it can read signals of multiple switches and sensors, and control light, servo motor and other various physical devices. It can also be developed into PC peripherals to communicate with the software in PC.

2.User

1 | Download the Arduino environment
Get the latest version from the download page.
When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure. Double-click the folder to open it. There should be a few files and sub-folders inside.

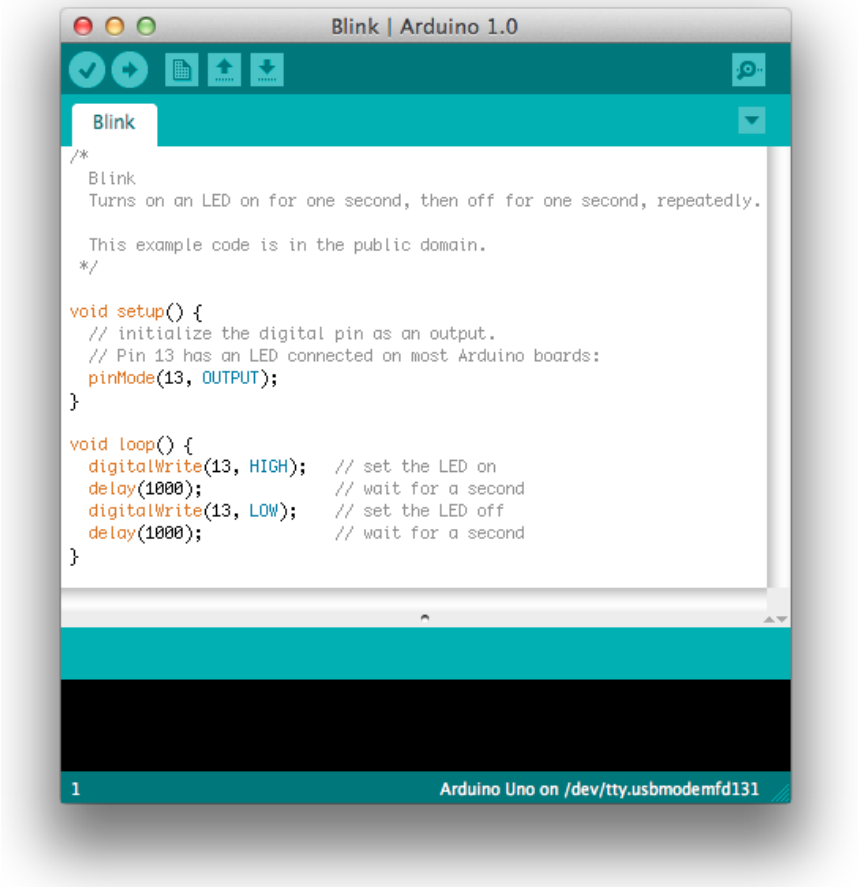
2 | Connect the board
The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either the USB connection to the computer or an external power supply. If you're using an Arduino Diecimila, you'll need to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it's on the two pins closest to the USB port.
Connect the Arduino board to your computer using the USB cable. The green power LED (labelled PWR) should go on.

3 | Install the drivers
Installing drivers for the Arduino Uno or Arduino Mega 2560 with Windows 7, Vista, or XP:
Plug in your board and wait for Windows to begin it's driver installation process. After a few moments, the process will fail, despite its best efforts
Click on the Start Menu, and open up the Control Panel.
While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.
Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)". If there is no COM & LPT section, look under "Other Devices" for "Unknown Device".

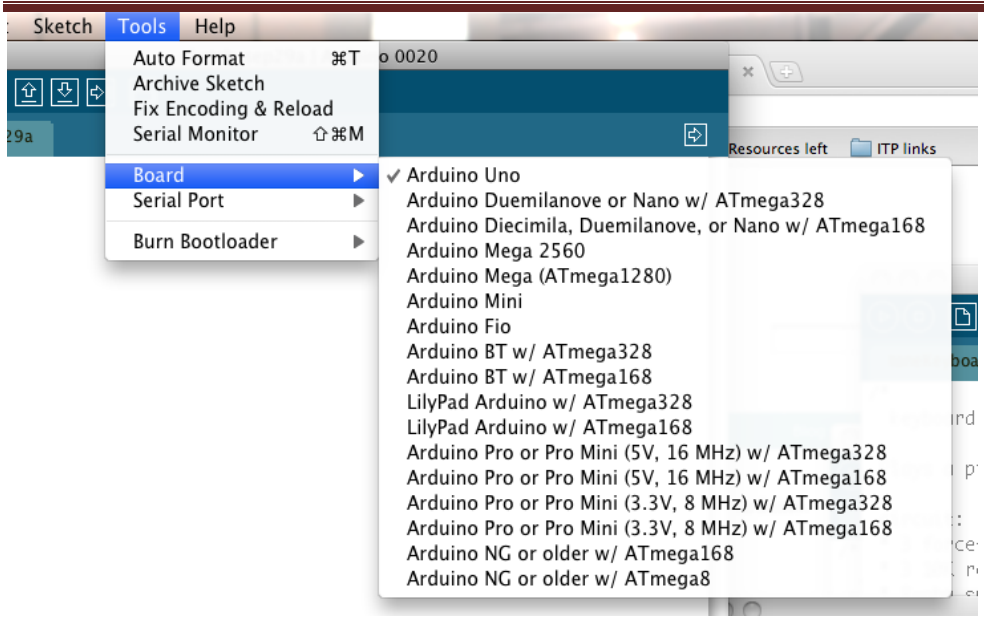
Right click on the "Arduino UNO (COmxx)" port and choose the "Update Driver Software" option.
Next, choose the "Browse my computer for Driver software" option.
Finally, navigate to and select the driver file named "arduino.inf", located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory). If you are using an old version of the IDE (1.0.3 or older), choose the Uno driver file named "Arduino UNO.inf"
Windows will finish up the driver installation from there.
See also: step-by-step screenshots for installing the Uno under Windows XP.
Installing drivers for the Arduino Duemilanove, Nano, or Diecimila with Windows7, Vista, or XP:
When you connect the board, Windows should initiate the driver installation process (if you haven't used the computer with an Arduino board before).
On Windows Vista, the driver should be automatically downloaded and installed. (Really, it works!)
On Windows XP, the Add New Hardware wizard will open:
When asked Can Windows connect to Windows Update to search for software? select No, not this time. Click next.
Select Install from a list or specified location (Advanced) and click next.
Make sure that Search for the best driver in these locations is checked; uncheck Search removable media; check Include this location in the search and browse to the drivers/FTDI USB Drivers directory of the Arduino distribution. (The latest version of the drivers can be found on the FTDI website.) Click next.
The wizard will search for the driver and then tell you that a "USB Serial Converter" was found. Click finish.
The new hardware wizard will appear again. Go through the same steps and select the same options and location to search. This time, a "USB Serial Port" will be found.
You can check that the drivers have been installed by opening the Windows Device Mananger (in the Hardware tab of System control panel). Look for a "USB Serial Port" in the Ports section; that's the Arduino board.

4 | Launch the Arduino application
Double-click the Arduino application. (Note: if the Arduino software loads in the wrong language, you can change it in the preferences dialog. See the environment page for details.)

5 | Open the blink example
Open the LED blink example sketch: File > Examples > 1.Basics > Blink.



6 | Select your board
You'll need to select the entry in the Tools > Board menu that corresponds to your Arduino.



Selecting an Arduino Uno

For Duemilanove Arduino boards with an ATmega328 (check the text on the chip on the board), select Arduino Duemilanove or Nano w/ ATmega328. Previously, Arduino boards came with an ATmega168; for those, select Arduino Diecimila, Duemilanove, or Nano w/ ATmega168. (Details of the board menu entries are available on the environment page.)

7 | Select your serial port

Select the serial device of the Arduino board from the Tools | Serial Port menu. This is likely to be COM3 or higher (COM1and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

8 | Upload the program

Now, simply click the "Upload" button in the environment. Wait a few seconds - you should see the RX and TX leds on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar. (Note: If you have an Arduino Mini, NG, or other board, you'll need to physically present the reset button on the board immediately before pressing the upload button.)

A few seconds after the upload finishes, you should see the pin 13 (L) LED on the board start to blink (in orange). If it does, congratulations! You've gotten Arduino up-and-running.

If you have problems, please see the troubleshooting suggestions.

You might also want to look at:

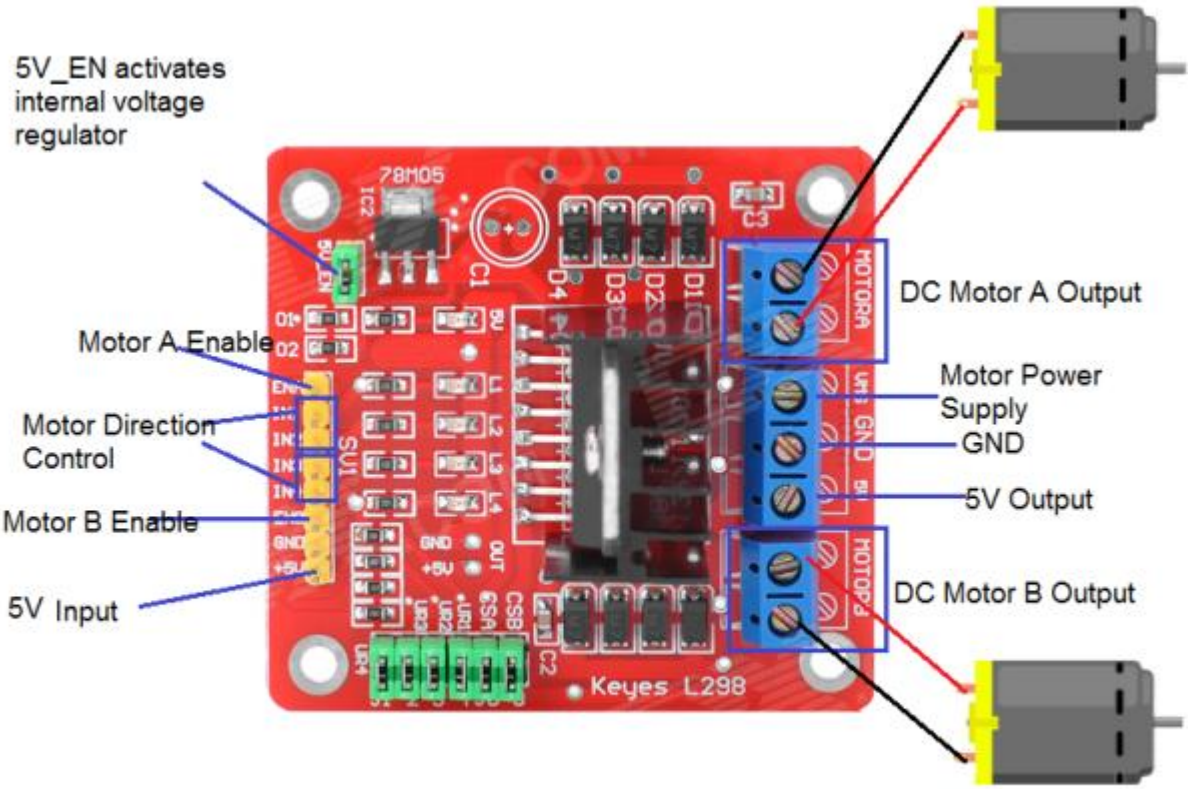
the examples for using various sensors and actuators

the reference for the Arduino language

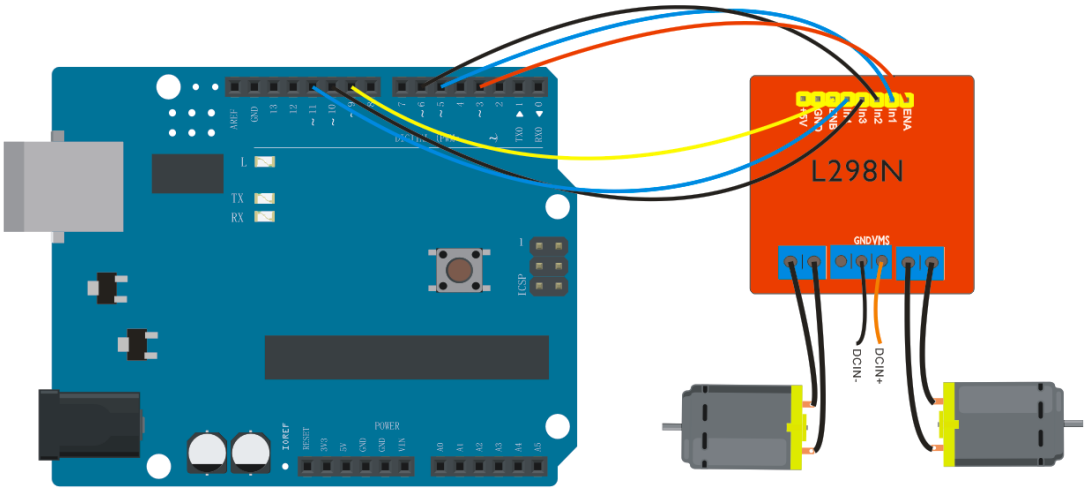
7. Project details

1. The application of L298N motor driver board

For the instruction for L298N driver board please refer to (L298N dual-H bridge DC motor driver board manual). Some of you still don't know how to control dual DC motor. Here are the details.



For the VMS driver part, power supply can be external power source, generally about 9V. For the logic part, power supply can be from the board internally; with terminals in suspense state, they can also be connected to +5V to +7V power. The three pins between each terminal are used to control the dual DC motor. EA and EB are connected to Arduino PWM interface for motor speed regulation. I1, I2, I3, I2 interface, connected to Arduino digital interfaces, are used for controlling the motor going forward, backward, steering and braking. Up until now, the preparatory work is completed. You can begin writing the program now. Here, the program for your reference includes car going straight, backward, turning left, turning right, and braking.



****

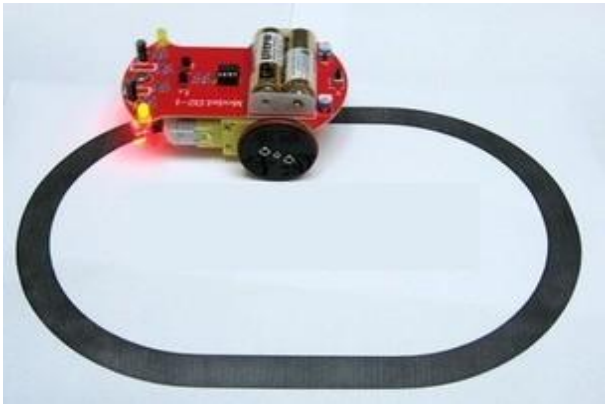
Program:

```
int pinI1=5;// define pin  I1
int pinI2=6;// define pin I2
int speedpin=3;// define  pin EA(PWM speed regulation)
int pinI3=10;// define pin I3
int pinI4=11;// define pin I4
int speedpin1=9;// define  pin EB(PWM speed regulation)
void setup()
{
pinMode(pinI1,OUTPUT);
pinMode(pinI2,OUTPUT);
pinMode(speedpin,OUTPUT);
pinMode(pinI3,OUTPUT);
pinMode(pinI4,OUTPUT);
pinMode(speedpin1,OUTPUT);
}
void loop()
{
// going straight
analogWrite(speedpin,100);// input analog value to set the speed
analogWrite(speedpin1,100);
digitalWrite(pinI4,LOW);// make the DC motor turn(right) anti-clockwise
digitalWrite(pinI3,HIGH);
digitalWrite(pinI1,LOW);// make the DC motor turn(left) clockwise
digitalWrite(pinI2,HIGH);
delay(2000);
// going backwards
analogWrite(speedpin,100);// input analog value to set the speed
analogWrite(speedpin1,100);
digitalWrite(pinI4,HIGH);// make the DC motor turn(right) clockwise
digitalWrite(pinI3,LOW);
digitalWrite(pinI1,HIGH);//make the DC motor turn(left) anti-clockwise
digitalWrite(pinI2,LOW);
delay(2000);
// turning left
analogWrite(speedpin,60);// input analog value to set the speed
analogWrite(speedpin1,60);
digitalWrite(pinI4,LOW);// make the DC motor turn(right) anti-clockwise
```

```
digitalWrite(pinI3,HIGH);
digitalWrite(pinI1,HIGH);//make the DC motor turn(left) anti-clockwise
digitalWrite(pinI2,LOW);
delay(2000);
// turning right
analogWrite(speedpin,60);//input analog value to set the speed
analogWrite(speedpin1,60);
digitalWrite(pinI4,HIGH);//make the DC motor turn(right) clockwise
digitalWrite(pinI3,LOW);
digitalWrite(pinI1,LOW);//make the DC motor turn(left) clockwise
digitalWrite(pinI2,HIGH);
delay(2000);
// braking
digitalWrite(pinI4,HIGH);// make the DC motor brake(right)
digitalWrite(pinI3,HIGH);
digitalWrite(pinI1,HIGH);//make the DC motor brake(left)
digitalWrite(pinI2,HIGH);
delay(2000);
}
```

Note: in the program, there can be other ways to make the motor turning left or right. You can try it out yourself.

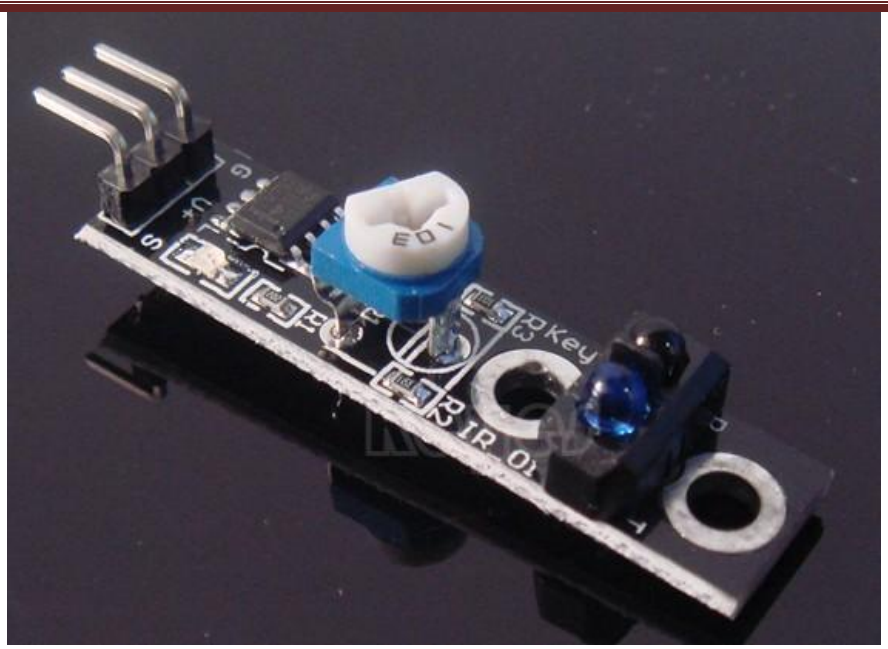
2. Line tracking smart car



Line tracking principle: the working principle of the TCRT5000 infrared double tube is to use the infrared's different reflectivity of different color, and convert the strength of the reflected signal into current signal. For black and white line tracking module, it's high level signal when detecting black line, low level when detecting white line. Detect height is 0-3cm.

Note: you can use a knob potentiometer in the circuit to adjust the sensitivity of this tracking module.

TCRT5000 infrared double tube is widely applied in robot design, and industrial manufacture. It can be used for black and white line tracking robot and industrial counting sensors.

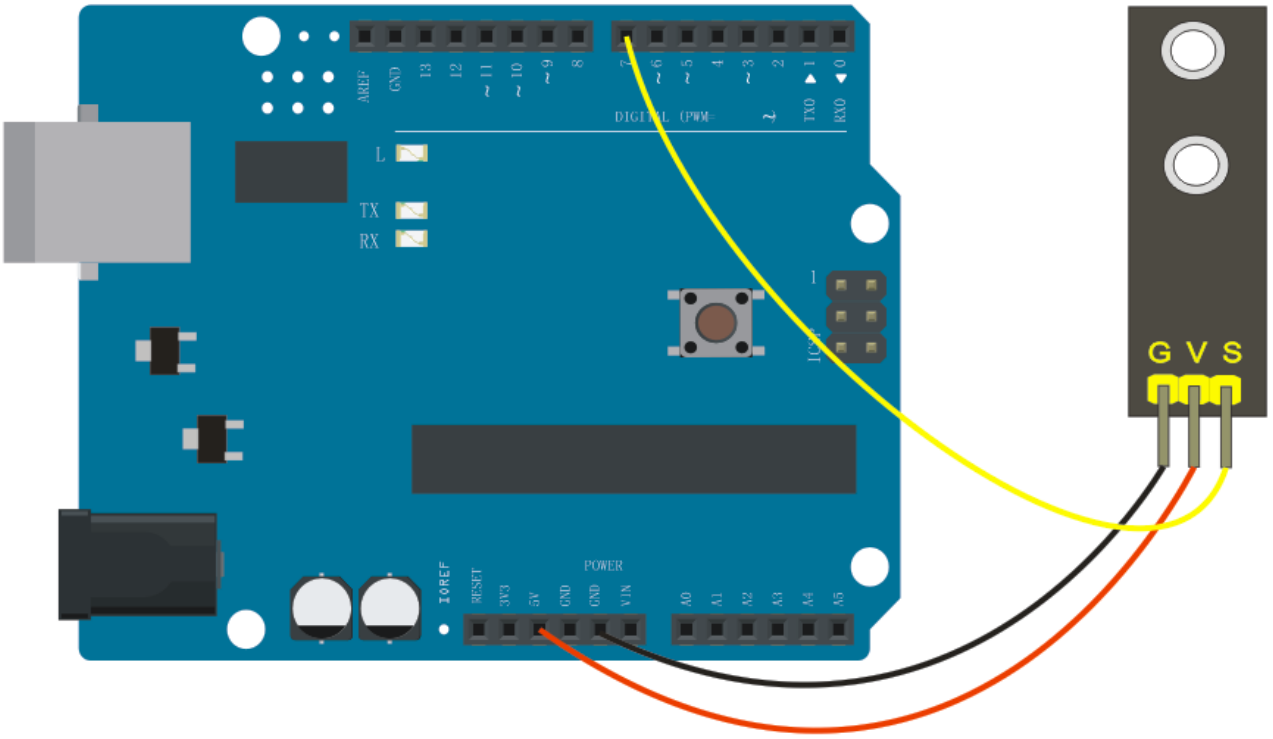


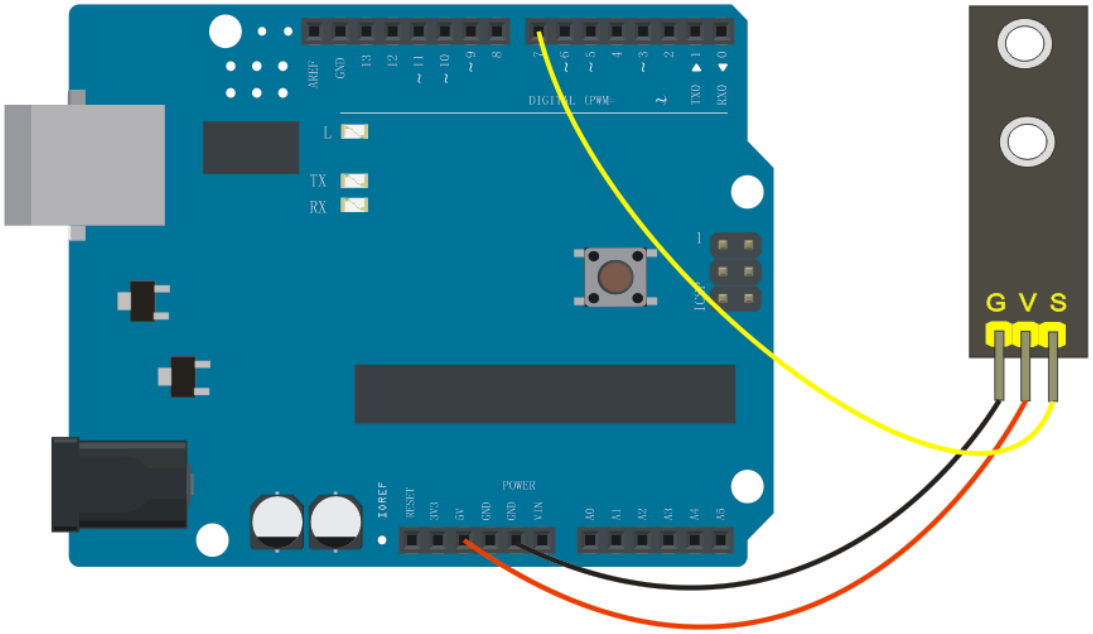
Usage:

1. For the sensor, there are 3 pins, namely GND, VCC, OUT. VCC and GND are for power supply. OUT is for signal output.
2. When it detects an object, the signal end outputs low level; when there is no object detected, the signal end outputs high level.
3. By judging whether the signal output is 0 or 1, it can know whether the object exists.

Performance parameter:

1. Detection distance: about 2cm for white paper floor. Different distance depends on the color





Test program:

```
int inputpin=7;// definedigital pin 7 as
detection port
int val;// define variable
void setup()
{
pinMode(inputpin,INPUT);// set digital pin 7 as
output
Serial.begin(9600);// set baud rate at 9600 for
the serial port
}
void loop()
{
val=digitalRead(inputpin);// read the value of
the digital port
Serial.println(val);// output the value of the
digital port}
```

****

Line tracking smart car

After some basic knowledge of line tracking module, let's make our own line tracking smart car.

This design of line tracking car system is base on Arduino . It includes software and hardware design. The controller unit of the car is Arduino UNO, using infrared photoelectric sensor to detect the black track and feedback the information to the Arduino board. Arduino board will have analysis of the acquired information, timely control the driver motor to adjust the moving direction of the car. So the car can drive automatically along the black track, realizing the line tracking function.

the system design is as below figure 1-1.

Detection(black line)	Software control	Driver motor	Car control

Figure 1-1

Car tracking principle: here, tracking means the car will move along the black line on the white floor. Because the black line and the white floor has different reflection coefficient, the track can be determine by the strength of the reflected light. Usually, it adopts the method of infrared detection. That is to use infrared ray's feature of reflection over different color. When the car is driving, it will continually emit infrared ray to the ground. When the infrared ray encounters the floor made of white paper. It will be reflected. The reflected ray will be received by the receiver of the car. If it encounters the black track, infrared ray will be absorbed so the receiver won't receive infrared ray. The microcontroller will determine the location of the black line and the moving direction of the car. Note that the detection rage of the infrared detector is limited.
The control system of the line tracking car consists of a main control circuit module, power supply module, infrared detecting module, motor, and a driver module. The control system design is as below figure 2-1.

Power Supply	Main controller UNO	L298	Gear motor

Photoelectric sensor

Black line

Figure2-1

Flow chart of line tracking car
When the car enters the tracking mode, it begins constantly scanning I/O port of the MCU. When the I/O port picks up a signal, it will enter the processing; firstly determine which detector detects the black line.

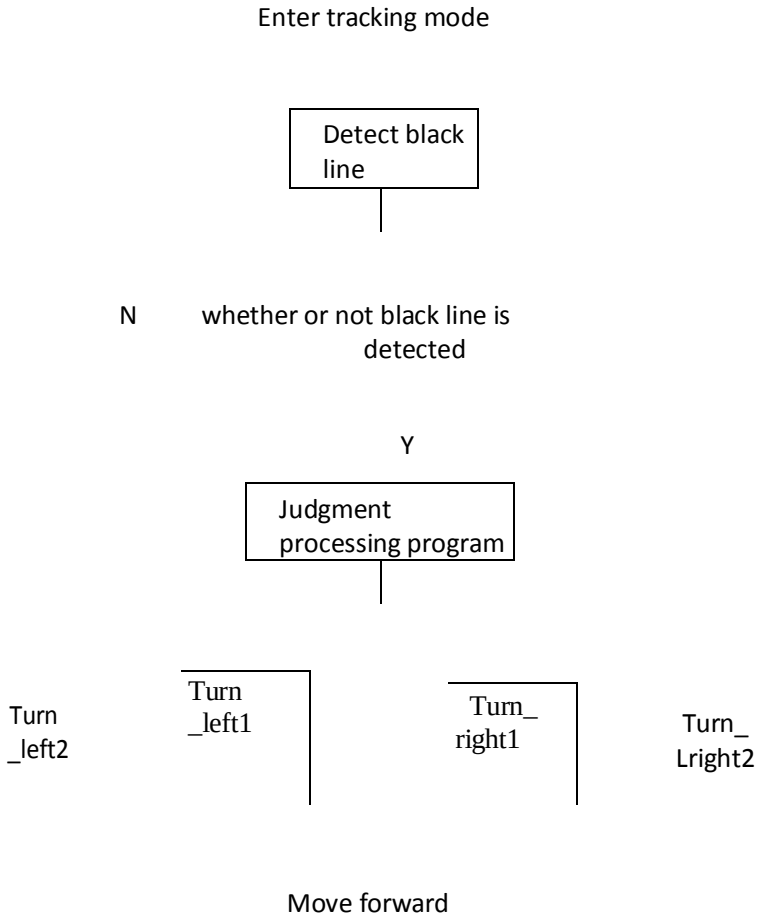
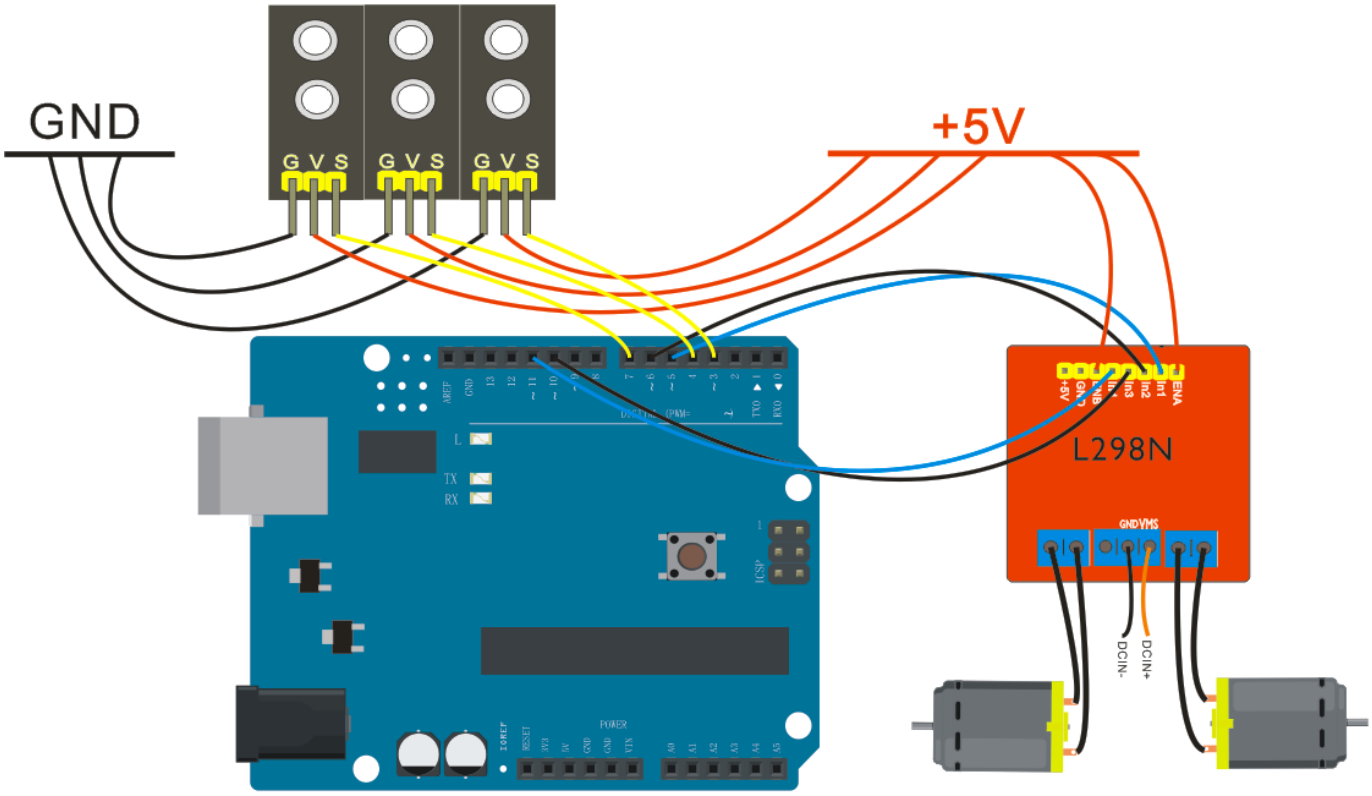


Figure3-1

Circuit wiring of Arduino tracking car



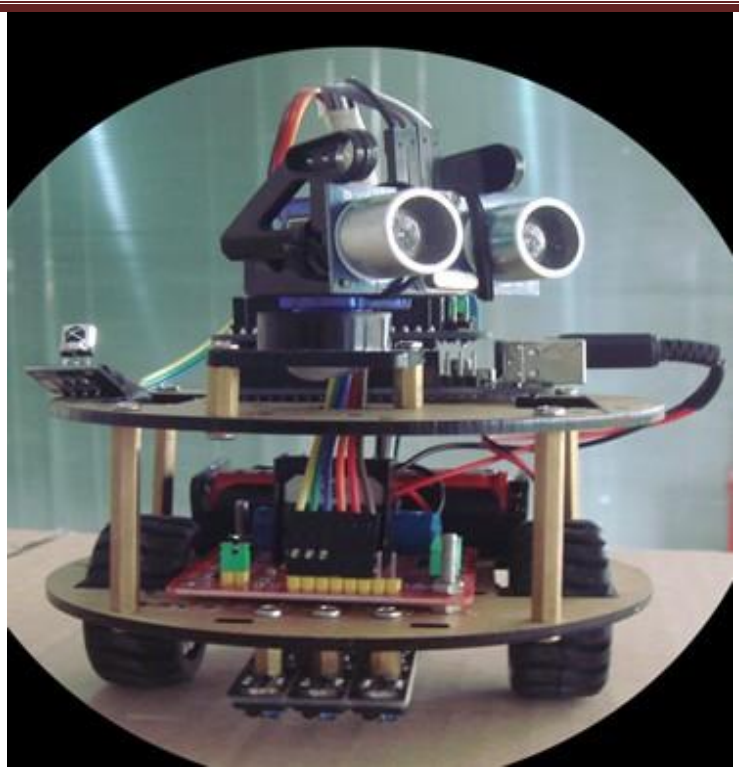
****

Arduino program for tracking car:

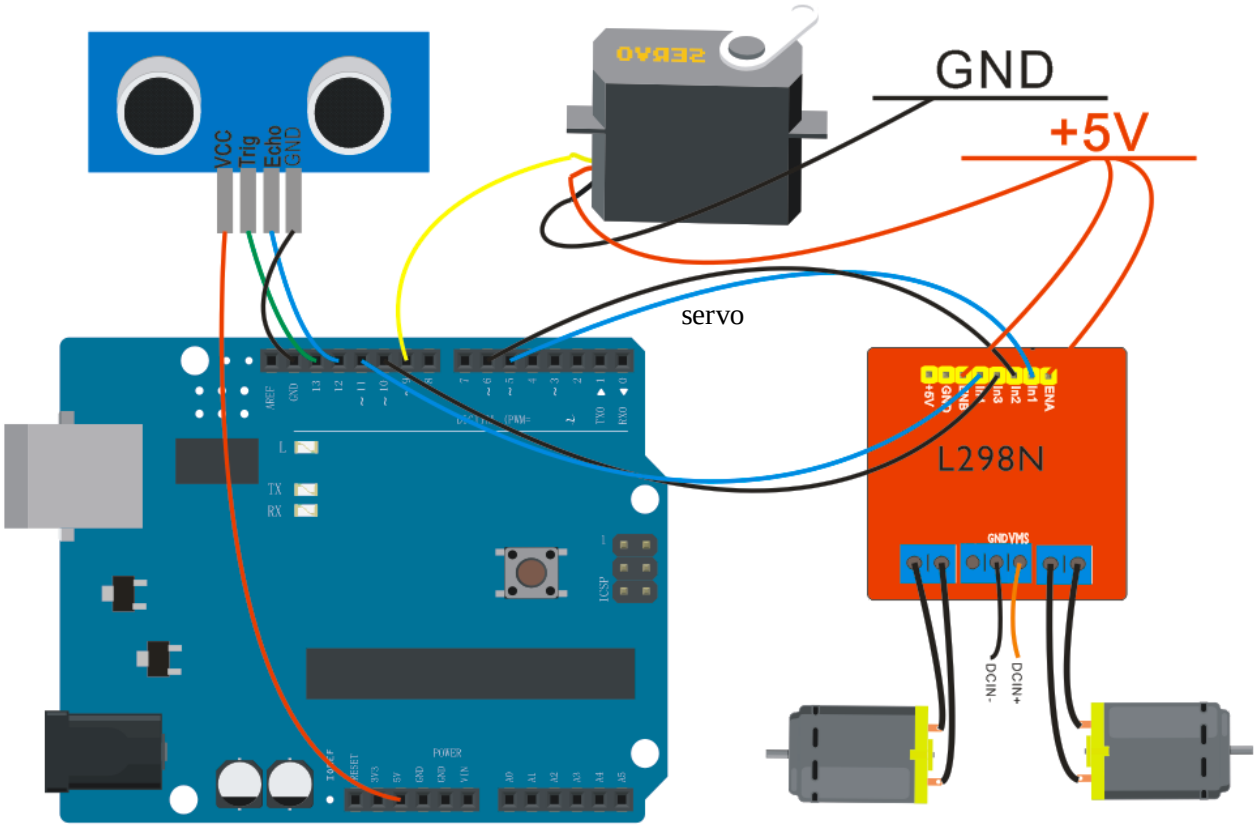
```
int MotorRight1=5;
int MotorRight2=6;
int MotorLeft1=10;
int MotorLeft2=11;
const int SensorLeft = 7;          // pin for sensor on the left
const int SensorMiddle= 4 ;// pin for sensor in the middle
const int SensorRight = 3;// pin for sensor on the right
int SL;          // left sensor status
int SM;          // middle 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 sensor
pinMode(SensorMiddle, INPUT);// define middle sensor
pinMode(SensorRight, INPUT); // define right sensor
}


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

if (SM == HIGH)// middle sensor in black area
{
if (SL == LOW & SR == HIGH) // black on left, white on right, turn left
{
digitalWrite(MotorRight1,LOW);
digitalWrite(MotorRight2,HIGH);
analogWrite(MotorLeft1,0);
```

```
analogWrite(MotorLeft2,80);
}
else if (SR == LOW & SL == HIGH) // white on left, black on right, turn right
{
analogWrite(MotorRight1,0);// turn right
analogWrite(MotorRight2,80);
digitalWrite(MotorLeft1,LOW);
digitalWrite(MotorLeft2,HIGH);
}
else // white on both sides, going forward
{
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 // middle sensor on white area
{
if (SL == LOW & SR == HIGH)// black on left, white on right, turn left
{
digitalWrite(MotorRight1,LOW);
digitalWrite(MotorRight2,HIGH);
digitalWrite(MotorLeft1,LOW);
digitalWrite(MotorLeft2,LOW);
}
else if (SR == LOW & SL == HIGH) // white on left, black on right, turn right
{
digitalWrite(MotorRight1,LOW);
digitalWrite(MotorRight2,LOW);
digitalWrite(MotorLeft1,LOW);
digitalWrite(MotorLeft2,HIGH);
}
else // all white, stop
{
digitalWrite(MotorRight1,HIGH);
digitalWrite(MotorRight2,LOW);
digitalWrite(MotorLeft1,HIGH);
digitalWrite(MotorLeft2,LOW);;
}}}
```

3. Ultrasonic Obstacle Avoidance Smart Car



Ultrasonic obstacle avoidance is easy to realize, simple in calculation. It is easy to control it in real time with practical measuring accuracy. Therefore, it has become a common method for obstacle avoidance. For the application method of ultrasonic, please refer to “Arduino ultrasonic ranging instruction”. Below is the connection diagram for ultrasonic obstacle avoidance:



1. Connection of motor

Motor 1 to L298N MOTOA

Motor 2 toL298N MOTOB

2. Power supply of L298N

Use 1 contact of battery case of 6 cells of AA batteries to supply power for L298N motor driver module, another contact for Arduino main board. The + of the power supply for L298N motor driver module is connected to the VMS of L298N; the - to the GND. + 5V interface of L298N is not connected to anything.

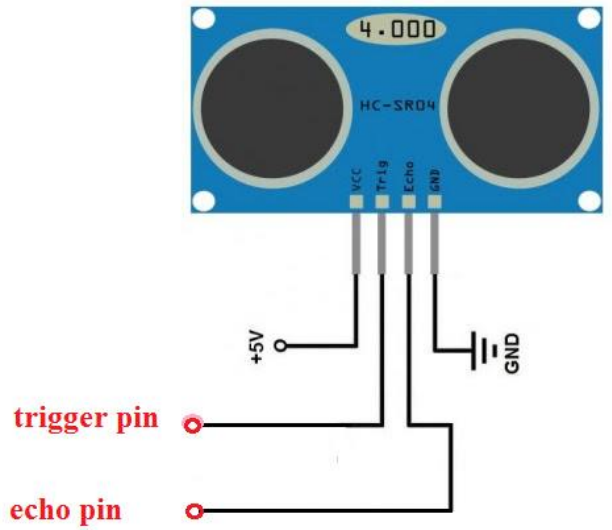
3. The enable and turning function of the motor (with program)

```
int pinLB=5;      //  define pin 5 for left and back, connected to pin PWM5of the controller
int pinLF=6;      // define pin 6 for left and front, connected to pin PWM6 of the controller board
int pinRB=10;     // define pin 10 for right and back, connected to pin PWM10 of the controller board
int pinRF=11;     // define pin 11 for right and front, connected to pin PWM11 of the controller board
```

4. Connection of the servo motor

```
myservo.attach(9);    // set servo motor output as pin 9(PWM)
```

5. Connection of the ultrasonic sensor



4 pins for this sensor

VCC to +5V

TRIQ signal input

ECHO signal output

GND to GND

```
int inputPin = 13;  //
define receiving pin
for ultrasonic signal
int outputPin =12;  // define sending pin for ultrasonic signal
**************************************************************************
Ultrasonic obstacle avoidance smart car program (ARDUINO)
        L = left
        R = right
        F = front
        B = back
    *
```

```
/
    #include <Servo.h>
    int pinLB=5;   // define pin 6 as left and back
    int pinLF=6;   // define pin 9 as left and front
    int pinRB=10;            // define pin 10 as right and back
    int pinRF=11;// define pin 11 as right and front
    int inputPin = 13;  // define receiving pin for ultrasonic signal
    int outputPin =12; // define sending pin for ultrasonic signal
    int Fspeedd = 0;  // speed going forward
    int Rspeedd = 0;  // speed going right
    int Lspeedd = 0;    // speed going left
    int directionn = 0;  //F=8 B=2 L=4 R=6
    Servo myservo;          // set  myservo
    int delay_time = 250; // settling time for the servo motor moving backwards

    int Fgo = 8;       // going forward
    int Rgo = 6;       // going right
     int Lgo = 4;   // going left
    int Bgo = 2;   // going backwards


    void setup()
    {
    Serial.begin(9600);     // define motor output
    pinMode(pinLB,OUTPUT); // pin  5 (PWM)
    pinMode(pinLF,OUTPUT); // pin  6 (PWM)
    pinMode(pinRB,OUTPUT); // pin 10 (PWM)
    pinMode(pinRF,OUTPUT); // pin  11 (PWM)

    pinMode(inputPin, INPUT);      // define receiving pin for ultrasonic signal
    pinMode(outputPin, OUTPUT); // define sending pin for ultrasonic signal
    myservo.attach(9);//  set servo motor output as pin 9(PWM)
    }
    void advance(int a)     // going forward
    {
    digitalWrite(pinRB,LOW); // motor move right and back
    digitalWrite(pinRF,HIGH);
    digitalWrite(pinLB,LOW); // motor move to left and back
    digitalWrite(pinLF,HIGH);
    delay(a * 100);
    }


    void right(int b)          // turn right(1 wheel)
    {
    digitalWrite(pinRB,LOW);  //motor move right and back
    digitalWrite(pinRF,HIGH);
    digitalWrite(pinLB,HIGH);
    digitalWrite(pinLF,HIGH);
    delay(b * 100);
    }
    void left(int c)            // turn left(1 wheel)
    {
    digitalWrite(pinRB,HIGH);
    digitalWrite(pinRF,HIGH);
    digitalWrite(pinLB,LOW);  // motor move left and back
    digitalWrite(pinLF,HIGH);
    delay(c * 100);
    }
    void turnR(int d)         // turn right( 2 wheels)
    {digitalWrite(pinRB,LOW); // motor move right and back
     digitalWrite(pinRF,HIGH);
    digitalWrite(pinLB,HIGH);
    digitalWrite(pinLF,LOW); //motor move left and front
    delay(d * 100);
    }
    void turnL(int e)// turn left(2 wheels)
    {digitalWrite(pinRB,HIGH);
    digitalWrite(pinRF,LOW);  // motor move right and front
    digitalWrite(pinLB,LOW);  // motor move left and back
    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)// going backwards
{digitalWrite(pinRB,HIGH); // motor move right and back
digitalWrite(pinRF,LOW);
digitalWrite(pinLB,HIGH); // motor move left and back
digitalWrite(pinLF,LOW);
delay(g * 100);
}
void detection()// measure 3 angles(0.90.179)
{
int delay_time = 250;  // settling time for the servo motor moving backwards
ask_pin_F();// read the distance upfront
if(Fspeedd < 10)// if distance less than  10cm
{
stopp(1);// clear output information
back(2);// going backwards for  0.2 second
}
if(Fspeedd < 25)// if distance less than 25cm
{
 stopp(1);// clear output information
ask_pin_L();// read the distance on the left
delay(delay_time);// settling time for the servo
ask_pin_R();// read the distance on the right
delay(delay_time);// settling time for the servo

if(Lspeedd > Rspeedd) // if distance on the left is more than that on the right
{
directionn = Rgo;// going right
}
if(Lspeedd <= Rspeedd) // if distance on the left is less than that on the right
{
directionn = Lgo;// going left
}


if (Lspeedd < 10 && Rspeedd < 10)  // if both distance are less than 10cm
{
directionn = Bgo;// going backwards
}
}
else// if the distance upfront is more than 25cm
{
directionn = Fgo;// going forward
}


}
void ask_pin_F()  // measure the distance upfront
{
myservo.write(90);
digitalWrite(outputPin, LOW);  // ultrasonic sends out low voltage 2μs delayMicroseconds(2);
digitalWrite(outputPin, HIGH); // ultrasonic sends out high voltage  10μs, at least 10μs
delayMicroseconds(10);
digitalWrite(outputPin, LOW);// maintain low voltage sending
float Fdistance = pulseIn(inputPin, HIGH); // read the time difference
Fdistance= Fdistance/5.8/10;// convert time into distance(unit: cm)
Serial.print("F distance:");// output distance in cm
Serial.println(Fdistance);// display distance
Fspeedd = Fdistance;// read the distance data into  Fspeedd
}
void ask_pin_L()// measure the distance on the left
{
myservo.write(9);


delay(delay_time);
digitalWrite(outputPin, LOW); // ultrasonic sends out low voltage 2μs
delayMicroseconds(2);
digitalWrite(outputPin, HIGH);// ultrasonic sends out high voltage  10μs, at least 10μs
delayMicroseconds(10);
digitalWrite(outputPin, LOW);// maintain low voltage sending
```

```
float Ldistance = pulseIn(inputPin, HIGH); //  read the time difference
Ldistance= Ldistance/5.8/10;// convert time into distance(unit: cm)
Serial.print("L distance:");        //output distance in cm
Serial.println(Ldistance);// display distance
Lspeedd = Ldistance;// read the distance data into  Lspeedd
}
void ask_pin_R()//  measure the distance on the right
{
myservo.write(177); delay(delay_time);
digitalWrite(outputPin, LOW);  // ultrasonic sends out low voltage 2μs
delayMicroseconds(2);
digitalWrite(outputPin, HIGH); // ultrasonic sends out high voltage  10μs, at least 10μs
delayMicroseconds(10);
digitalWrite(outputPin, LOW); // maintain low voltage sending
float Rdistance = pulseIn(inputPin, HIGH); //read the time difference
Rdistance= Rdistance/5.8/10;                // convert time into distance (unit: cm)
Serial.print("R distance:");       // output distance in cm
Serial.println(Rdistance);// display distance
Rspeedd = Rdistance;// read the distance data into    Rspeedd
}


void loop()
{
myservo.write(90); // reset the servo motor and prepare it for the next measurement
detection();  // measure the angle and decide which direction to move
if(directionn == 2) //if  directionn = 2
{
back(8);// going backwards
turnL(2);// slightly move to the left to avoid stuck in the dead end
Serial.print(" Reverse ");  // display direction (backwards)
}
if(directionn == 6)// if  direction = 6
{
back(1);
turnR(6);// turn right
Serial.print(" Right ");// display direction(right)


}
if(directionn == 4)//if  direction = 4
{
back(1);
turnL(6);// turn left
Serial.print(" Left ");// display direction(left)
}
if(directionn == 8)//if  direction= 8
{
advance(1);// going forward
Serial.print(" Advance ");  //display direction(forward)
Serial.print("  ");
}
}
```

4. Infrared remote control smart car

Before the experiment:

1. Place the function library IRremote into the Arduino libraries directory.

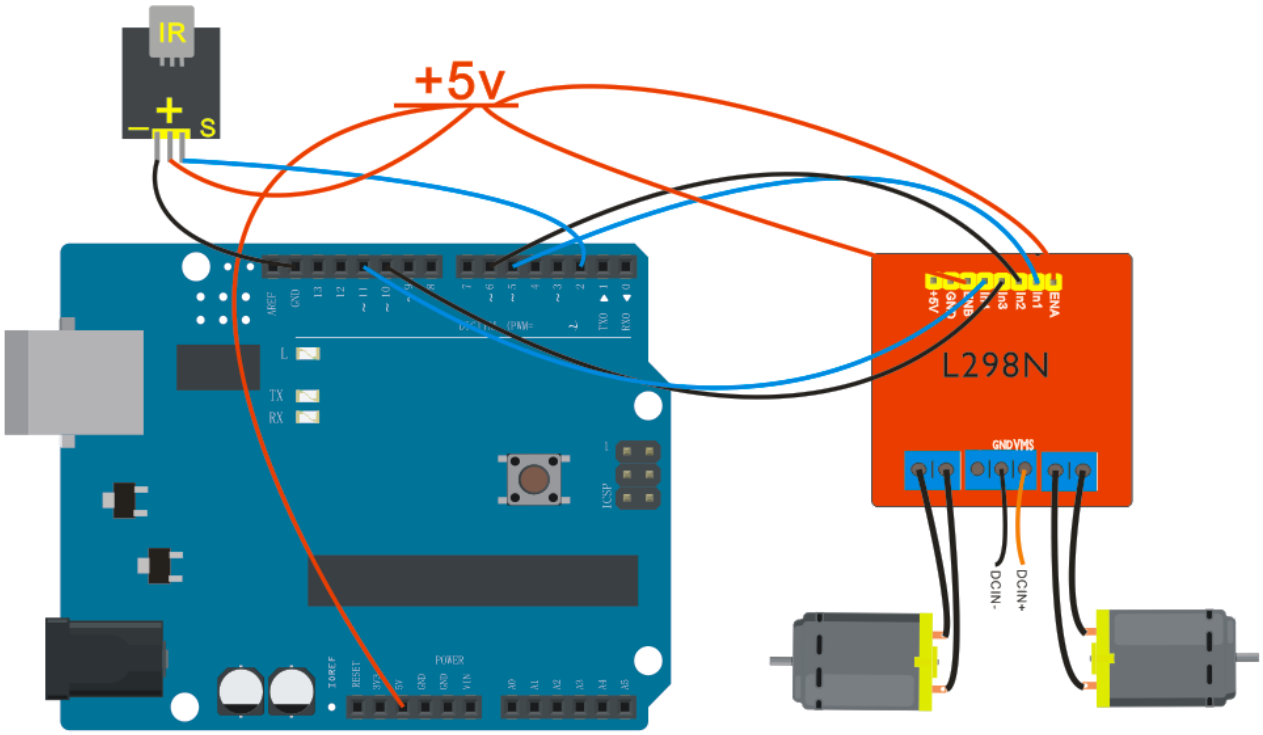
2. Open IrReceive.pde to acquire the code for your infrared remote control (IRcode will be displayed in Serial Monitor), write down IRcode and modify it in your IR code in the program.

```
/*
 * IRRemote code test
 *   Example 1.2: display the type of IR protocol such as NEC, Sony SIRC, Philips RC5, Philips RC6
 */
        #include <IRremote.h>          // call IRRemote function library
        const int irReceiverPin = 2;     // set pin 2 as IR receiver signal  OUTPUT


        IRrecv irrecv(irReceiverPin);              // set  IRrecv to receive IR signal
        decode_results results;          //  decode result will be put into the variable of the result in the
        decode_results


        void setup()
        {
        Serial.begin(9600);// set communication rate at  9600 bps
        irrecv.enableIRIn();// start IR decoding
        }


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


        // determine  the type of IR protocol
        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");
}


// serial print IR code to Serial port Serial.print(", irCode: ");
Serial.print(results->value, HEX);          // IR code
Serial.print(", bits: ");
Serial.println(results->bits);    //  IR code bit
}


void loop()
{
if (irrecv.decode(&results)) {   //  finish decoding, receive IR signal
showIRProtocol(&results);// display the type of IR protocol
irrecv.resume();// continue receiving IR signal coming next
}
}
```

Replace the IR code in the IR control part program with the code from the test result

IR remote control smart car program




```
//******IR remote control smart car program*******


#include <IRremote.h> int RECV_PIN = 2;

int pinLB=5;// define pin for  I1

int pinLF=6;// define pin for  I2

int pinRB=10;// define pin for  I3

int pinRF=11;// define pin for  I4

//******IR control part******** long advence = 0x00EF807F; long back = 0x00EFA05F;

long stop = 0x00EF906F;
```

```
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);// motor going right
digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,LOW);// motor going left
digitalWrite(pinLF,HIGH);}


if (results.value == back )



{digitalWrite(pinRB,HIGH);// motor going right and BACK
digitalWrite(pinRF,LOW);}


if (results.value == left )
{ digitalWrite(pinRB,LOW);// motor going right and STOP
digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,HIGH);// motor going left
digitalWrite(pinLF,LOW);}



if (results.value == right )
{ digitalWrite(pinRB,HIGH);// motor going right
digitalWrite(pinRF,LOW);
digitalWrite(pinLB,HIGH);// motor going left and STOP
digitalWrite(pinLF,HIGH);}


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

```
{
digitalWrite(pinRB,HIGH);// motor going right and STOP
digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,HIGH);// motor going left and STOP
digitalWrite(pinLF,HIGH);


}



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

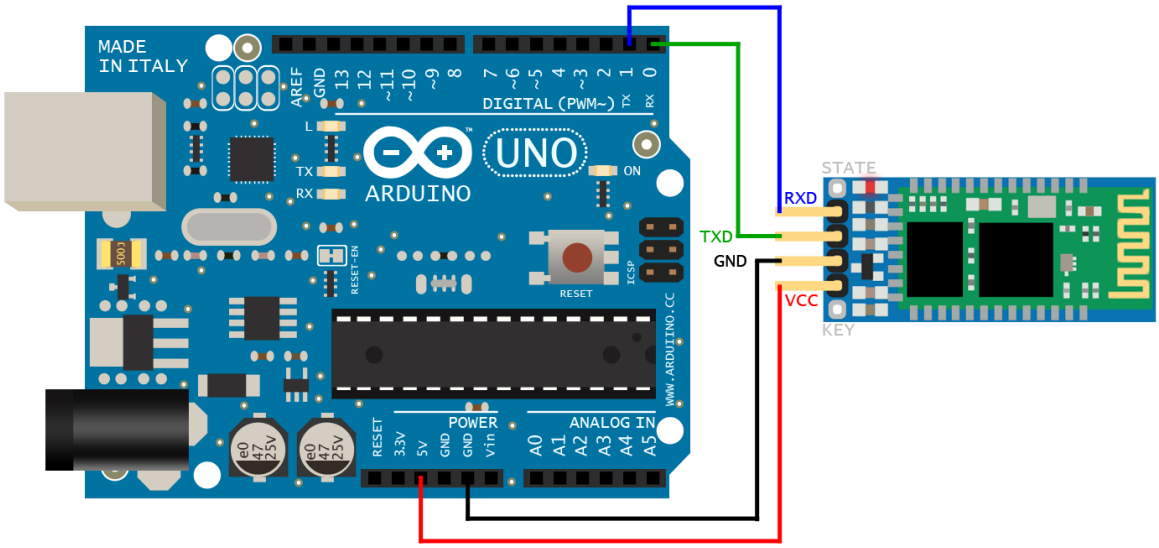
Arduino communication by bluetooth

The term “bluetooth” is derived from a Denmark king’s name in 10th century. The king is named Harald Blatand, while “Blatand” in English means bluetooth.

The so-called bluetooth technology is actually a kind of short distance wireless transmission technology. We can use “bluetooth” to effectively simplify the communication between terminal devices such as laptop and mobile phone. It can also simplify communication between these devices and the Internet, thus improving the speed and efficiency of data transmission between them, widening the application scope of wireless communication.

Since this is our first interaction with bluetooth module, we’ll do a small project of communication

between Arduino and PC. First is the wire connection. Connect main board +5V to bluetooth VCC, main board GND to bluetooth -GND, main board TX to bluetooth RX and RX to bluetooth TX. When the bluetooth module is successfully connected to PC, the power indicator of the bluetooth module will flicker and green connection indicator will be light up.



Now, let's move on to program. I'll enter "r" and after Arduino receives my command "r", the LED in PIN 13 will flicker and print character of "keyes". The program is 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");
 }
}
```

Now, let's figure out how to make programmable smart car to go forward, backward, turning left or right via arduino bluetooth remote control. There are two ways to control the movement of smart car, by PC or mobile phone (The phone operating system must support Android 2.3.7 or later version and PC must carry bluetooth). The phone should be paired with bluetooth car during initial use (no need to locate wireless device after first pairing). Please check the following steps for first paring:

1. Don't forget to turn on the bluetooth function of the phone. The software will remind users to turn on bluetooth function when the software is started.

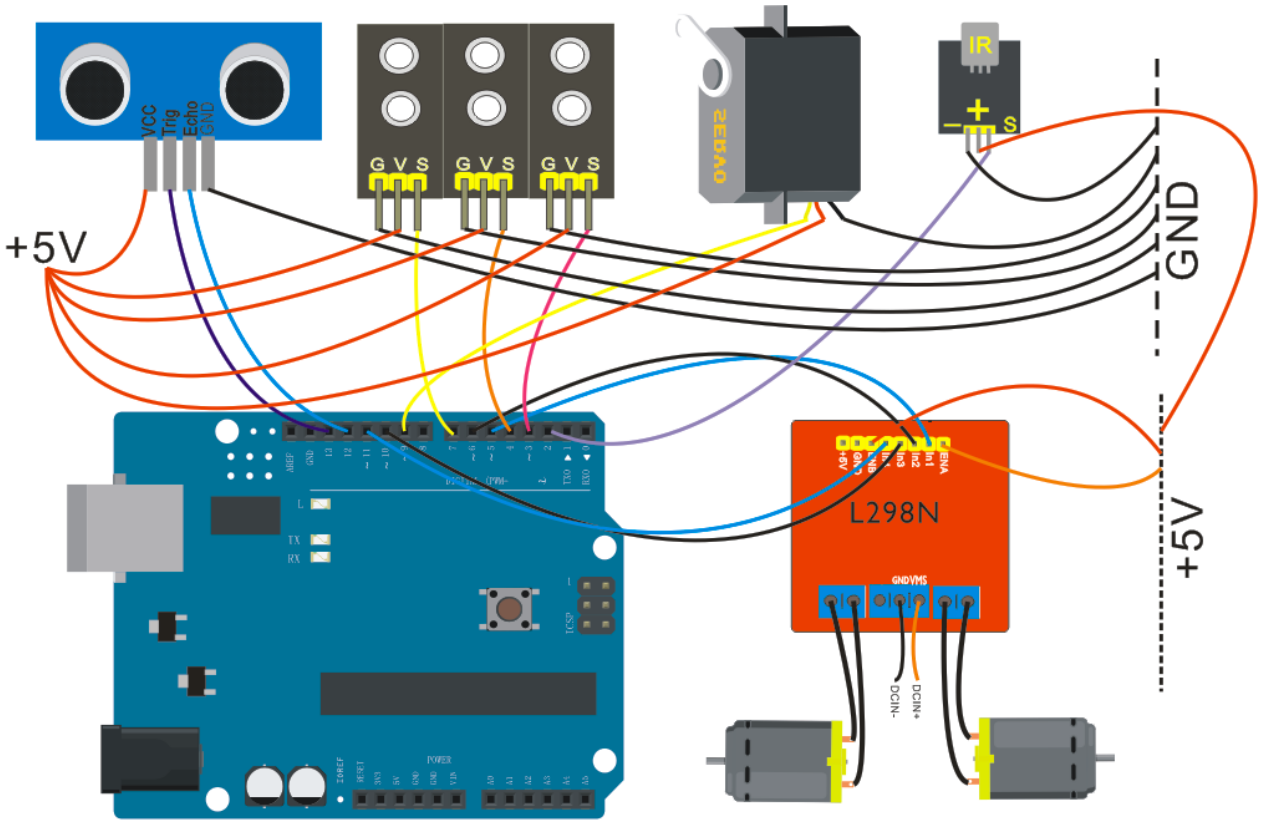
2. Connect bluetooth device according to the text instructions on the photo. Please scan and pair with bluetooth device, otherwise, you would fail to connect with smart car.
3. The code for pairing with the smart car is “1234”, go try it!

The program for Arduino bluetooth remote control programmable smart car is as follows:

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


void setup()
{
  Serial.begin(9600);
  pinMode(MotorRight1,  OUTPUT);  //  pin   5  (PWM)
  pinMode(MotorRight2,  OUTPUT);  //  pin   6  (PWM)
  pinMode(MotorLeft1,   OUTPUT);  //  pin   10  (PWM)
  pinMode(MotorLeft2,  OUTPUT);  // pin 11 (PWM)
}

void go()// 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() // go backwards
{
       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. 4 in 1(line tracking, obstacle avoidance, IR control, bluetooth control) program




```
//*******************************
#include <IRremote.h>
#include <Servo.h>
//*********************** define motor pin **************************
int MotorRight1=5;
int  MotorRight2=6;
int  MotorLeft1=10;
int  MotorLeft2=11;
int counter=0;
const int irReceiverPin = 2; // set pin 2 as IR receiver signal OUTPUT

char val;
//************************set the IRcode from the test result**************************
long IRfront= 0x00FFA25D;          // code for going forward
long IRback=0x00FF629D;           // going backward
long IRturnright=0x00FFC23D;      // turn right
long IRturnleft= 0x00FF02FD;       // turn left
long IRstop=0x00FFE21D;           // stop
long IRcny70=0x00FFA857;  // CNY70 aoto-moving mode
long IRAutorun=0x00FF906F;  // ultrasonic aoto-moving mode
long IRturnsmallleft= 0x00FF22DD;
//************************* define pin CNY70*************************************
const int SensorLeft = 7;       // input pin for left sensor
const int SensorMiddle= 4 ;     // input pin for middle sensor
const int SensorRight = 3;     // input pin for right sensor
int SL;     // left sensor status
int SM;    // middle sensor status
int SR;     // right sensor status
```

```
IRrecv irrecv(irReceiverPin);  // set IRrecv to receive IR signal
decode_results results;                // decode result will be put into the variable of the result in the
decode_results
//************************ define pin for ultrasonic ******************************
int inputPin =13 ; //  define ultrasonic signal receiving pin rx
int outputPin =12; // define ultrasonic signal sending pin'tx
int Fspeedd = 0; // distance upfront
int Rspeedd = 0; // distance on the right
int Lspeedd = 0; // distance on the left
int directionn = 0; //  F=8 B=2 L=4 R=6
Servo myservo; //  set myservo
int delay_time = 250; // settling time for the servo motor moving backwards
int Fgo = 8;       // going forward
int Rgo = 6;       // going right
int Lgo = 4;  // going left
int Bgo = 2;  // going backwards
//*********************************************************************(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 IR decoding
  pinMode(SensorLeft, INPUT); //define left sensor
  pinMode(SensorMiddle, INPUT);// define middle sensor
  pinMode(SensorRight, INPUT); // define right sensor
  digitalWrite(2,HIGH);
  pinMode(inputPin, INPUT); // define receiving pin for ultrasonic signal
  pinMode(outputPin, OUTPUT); // define sending pin for ultrasonic signal
  myservo.attach(9); // set servo motor output as pin 5(PWM)


 }
//*********************************************************************(Void)
void advance(int a) // go forward
{
       digitalWrite(MotorRight1,LOW);
       digitalWrite(MotorRight2,HIGH);
       digitalWrite(MotorLeft1,LOW);
       digitalWrite(MotorLeft2,HIGH);  delay(a
       * 100);
}
void right(int b) //turn right(1 wheel)
{
      digitalWrite(MotorLeft1,LOW);
      digitalWrite(MotorLeft2,HIGH);
      digitalWrite(MotorRight1,LOW);
```

```
      digitalWrite(MotorRight2,LOW);
      delay(b * 100);
}
void left(int c) // turn left(1 wheel)
{
     digitalWrite(MotorRight1,LOW);
     digitalWrite(MotorRight2,HIGH);
     digitalWrite(MotorLeft1,LOW);
     digitalWrite(MotorLeft2,LOW); delay(c
     * 100);
}
void turnR(int d) // turn right(2 wheels)
{
     digitalWrite(MotorRight1,HIGH);
     digitalWrite(MotorRight2,LOW);
     digitalWrite(MotorLeft1,LOW);
     digitalWrite(MotorLeft2,HIGH);
     delay(d * 100);
}
void turnL(int e) // turn left (2 wheels)
{
     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) // go backwards
{
       digitalWrite(MotorRight1,HIGH);
       digitalWrite(MotorRight2,LOW);
       digitalWrite(MotorLeft1,HIGH);
       digitalWrite(MotorLeft2,LOW);; delay(g
       * 100);
}
void detection() // measure 3 angles(0.90.179)
{
   int delay_time = 250; // settling time for the servo motor moving backwards
   ask_pin_F(); //  read the distance upfront
```

```
  if(Fspeedd < 10) // if distance less than 10cm
 {
   stopp(1); // clear output information
   back(2); // oing backwards for 0.2 second
 }
  if(Fspeedd < 25) // if distance less than 25cm
 {
   stopp(1); // clear output information
   ask_pin_L(); // read the distance on the left
   delay(delay_time); // settling time for the servo
   ask_pin_R();       // read the distance on the right
   delay(delay_time);          // settling time for the servo

     if(Lspeedd > Rspeedd) //if distance on the left is more than that on the right

  {
    directionn = Lgo; // go left
  }

   if(Lspeedd <= Rspeedd) // if distance on the left is less than that on the right
   {
    directionn = Rgo; // going right
   }

     if (Lspeedd < 15 && Rspeedd < 15) //  if both distance are less than 10cm
  {
      directionn = Bgo; // going backwards
   }
  }
  else // if the distance upfront is more than 25cm
 {
     directionn = Fgo; // going forward
 }
}
//*************************************************************************
****
void ask_pin_F() //  measure the distance upfront
{
myservo.write(90);
digitalWrite(outputPin, LOW); // ultrasonic sends out low voltage 2µs
delayMicroseconds(2);
digitalWrite(outputPin, HIGH); // ultrasonic sends out high voltage 10µs, at least 10µs
delayMicroseconds(10);
digitalWrite(outputPin, LOW); // maintain low voltage sending
float Fdistance = pulseIn(inputPin, HIGH); // read the time difference
Fdistance= Fdistance/5.8/10; // convert time into distance (unit: cm)
Serial.print("F distance:"); //output distance in cm
Serial.println(Fdistance); // display distance
```

```
Fspeedd = Fdistance; // read the distance data into Fspeedd
}
//*************************************************************************
***
void ask_pin_L() // measure the distance on the left
{
myservo.write(177);
delay(delay_time);
digitalWrite(outputPin, LOW); //  ultrasonic sends out low voltage 2μs
delayMicroseconds(2);
digitalWrite(outputPin, HIGH); //  ultrasonic sends out high voltage  10μs, at least 10μs
delayMicroseconds(10);
digitalWrite(outputPin, LOW); //  maintain low voltage sending
float Ldistance = pulseIn(inputPin, HIGH); // read the time difference
Ldistance= Ldistance/5.8/10; //convert time into distance(unit: cm)
Serial.print("L distance:"); //output distance in cm
Serial.println(Ldistance); //display distance
Lspeedd = Ldistance; // read the distance data into Lspeedd
}
//*************************************************************************
*
void ask_pin_R() // measure the distance on the right
{
myservo.write(5);
delay(delay_time);
digitalWrite(outputPin, LOW); // ultrasonic sends out low voltage 2μs
delayMicroseconds(2);
digitalWrite(outputPin, HIGH); // ultrasonic sends out high voltage  10μs, at least 10μs
delayMicroseconds(10);
digitalWrite(outputPin, LOW); //maintain low voltage sending
float Rdistance = pulseIn(inputPin, HIGH); //read the time difference
Rdistance= Rdistance/5.8/10; //convert time into distance(unit: cm)
Serial.print("R distance:"); //output distance in cm
Serial.println(Rdistance); //display distance
Rspeedd = Rdistance; // read the distance data into Rspeedd
}
//*************************************************************************
*(LOOP)
void loop()
{
    SL = digitalRead(SensorLeft); SM =
    digitalRead(SensorMiddle); SR =
    digitalRead(SensorRight);

    performCommand();
//*************************************************************************normal
remote control mode
  if (irrecv.decode(&results))
   {        // finish decoding, receive IR signal
```

```
/*********************************************************************/
    if (results.value == IRfront)// go forward
    {
     advance(10);// go forward
    }
/*********************************************************************/
    if (results.value ==  IRback)// go backward
    {
     back(10);// go backward
    }
/*********************************************************************/
    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);
  }
//********************************************************************** black
and white line mode
  if (results.value == IRcny70)
 {
   while(IRcny70)
   {
     SL = digitalRead(SensorLeft); SM =
     digitalRead(SensorMiddle); SR =
     digitalRead(SensorRight);

     if (SM == HIGH)// middle sensor in black area
     {
       if (SL == LOW & SR == HIGH) // black on left, white on right, turn left
       {
         digitalWrite(MotorRight1,LOW);
         digitalWrite(MotorRight2,HIGH);
         analogWrite(MotorLeft1,0);
         analogWrite(MotorLeft2,80);
       }
       else if (SR == LOW & SL == HIGH) // white on left, black on right, turn right
       {
```

```
      analogWrite(MotorRight1,0);// turn right
      analogWrite(MotorRight2,80);
      digitalWrite(MotorLeft1,LOW);
      digitalWrite(MotorLeft2,HIGH);
    }
   else  // white on both sides, going forward
    {
      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 //  middle sensor on white area
{
   if (SL == LOW & SR == HIGH)// black on left, white on right, turn left
  {
     digitalWrite(MotorRight1,LOW);
     digitalWrite(MotorRight2,HIGH);
     digitalWrite(MotorLeft1,LOW);
     digitalWrite(MotorLeft2,LOW);
  }
   else if (SR == LOW & SL == HIGH) // white on left, black on right, turn right
  {
    digitalWrite(MotorRight1,LOW);
    digitalWrite(MotorRight2,LOW); digitalWrite(MotorLeft1,LOW);
    digitalWrite(MotorLeft2,HIGH);
  }
   else // all 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;
  }
//********************************************************************** ultrasonic
auto-moving mode
 if (results.value ==IRAutorun )
    {
        while(IRAutorun)
      {
        myservo.write(90); // reset the servo motor and prepare it for the next measurement
        detection(); // measure the angle and decide which direction to move
         if(directionn == 8) // if  directionn = 8
        {
          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); // going forward
            Serial.print(" Advance "); // display direction(forward)
            Serial.print(" ");
         }
        if(directionn == 2) // if  directionn = 2
       {
         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); // going backwards
   turnL(3); // slightly move to the left to avoid stuck in the dead end
   Serial.print(" Reverse "); //  display direction (backwards)
}
 if(directionn == 6) // if  direction = 6
{
 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(right)
}
 if(directionn == 4) // if  direction = 4
{
  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(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;
     }
/*********************************************************************/
    else
   {
        digitalWrite(MotorRight1,LOW);

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


      irrecv.resume();    // continue receiving IR signal coming next
  }
}

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

}
```