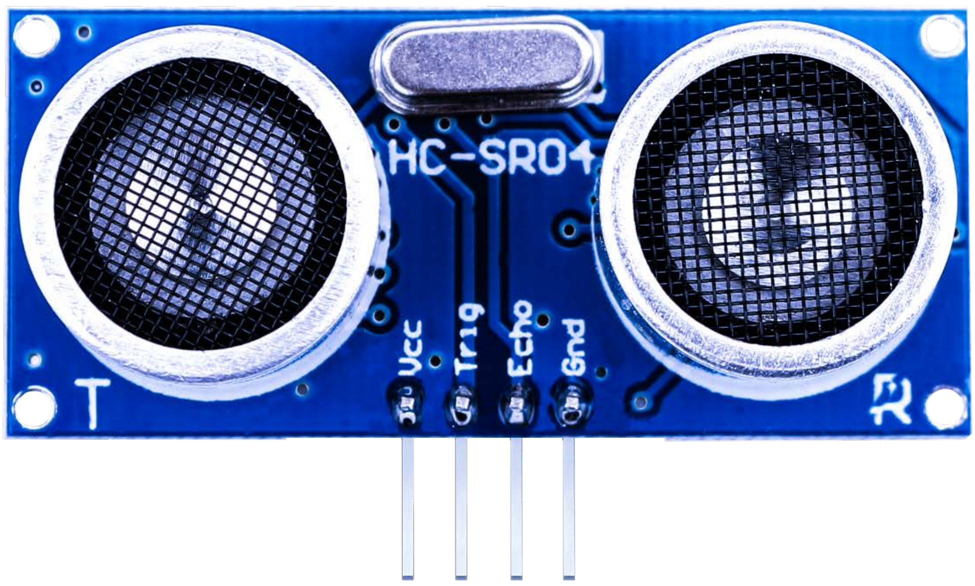
Lesson4 Obstacle avoidance car



Points of this section

The joy of learning, is not just know how to control your car, but also know how to protect your car. So, make you car far away from collision.

Learning parts:

- Learn how to assemble the ultrasonic module
- Be familiar with using steering
- Learn about the principle of car avoidance
- Use the program to make obstacle avoidance car come true

Preparations:

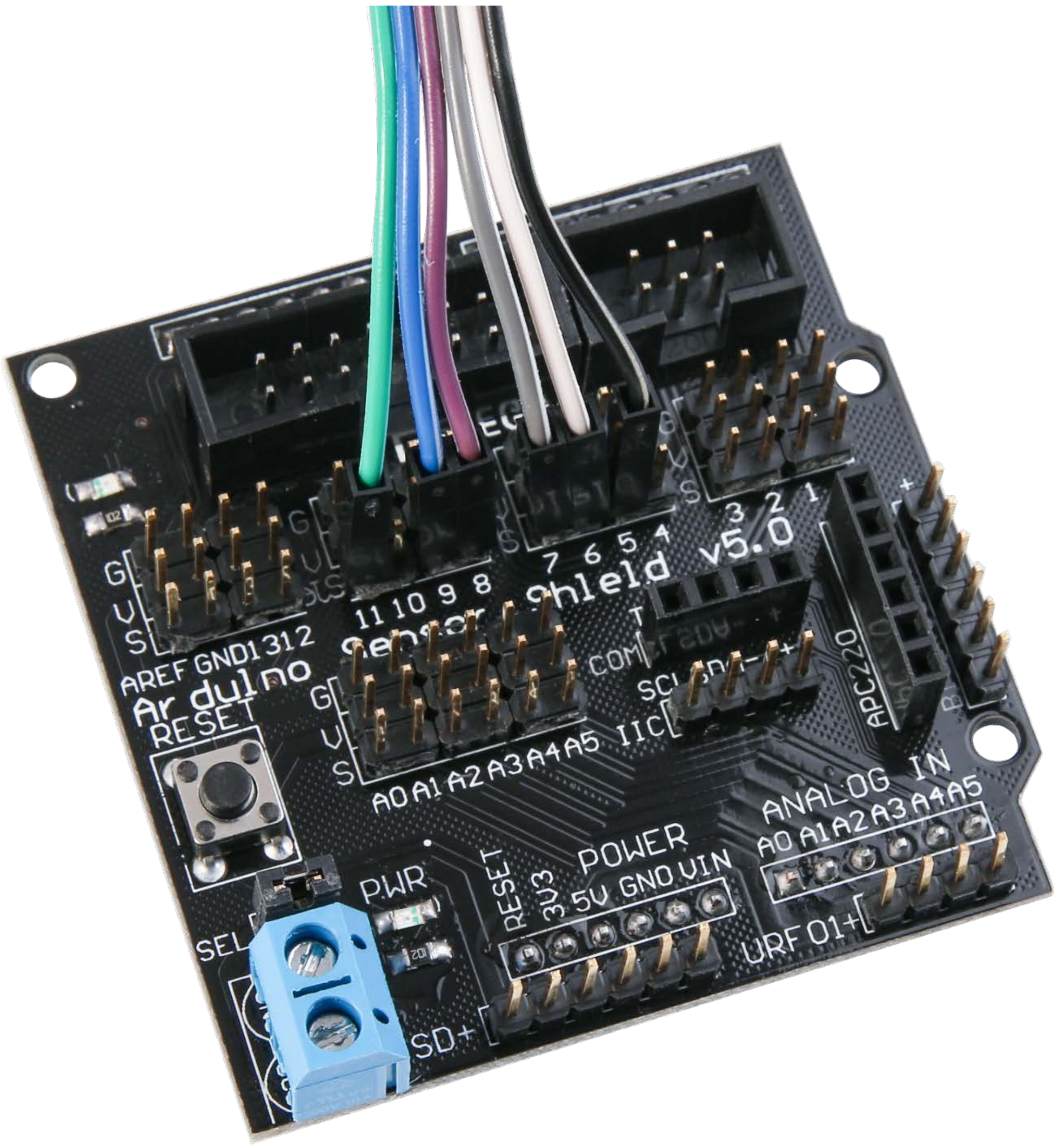
- A car (with battery)
- A USB cable
- A suit of ultrasonic cradle head

Ⅰ. Connection

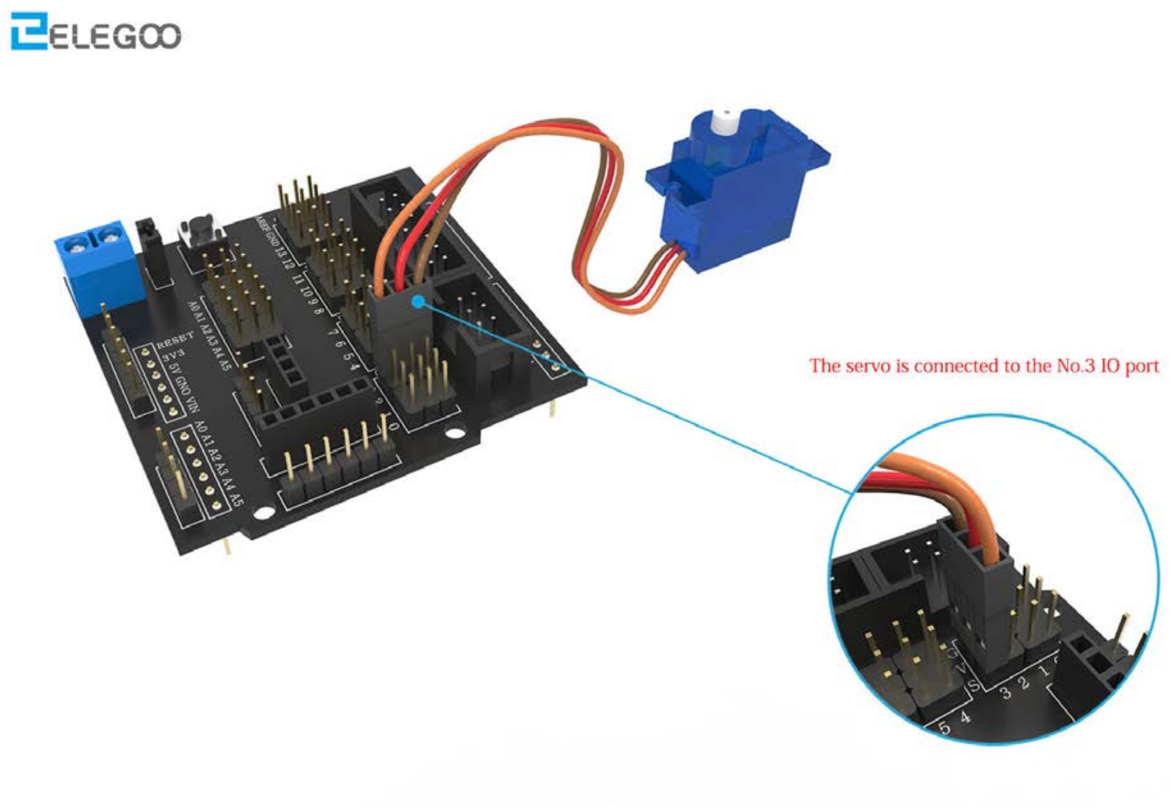
L298N

The Servo library supports up to 12 motors on most Arduino boards and 48 on the Arduino Mega. On boards other than the Mega, use of the library disables analogWrite() (PWM) functionality on pins 9 and 10, whether or not there is a Servo on those pins. On the Mega, up to 12 servos can be used without interfering with PWM functionality; use of 12 to 23 motors will disable PWM on pins 11 and 12.

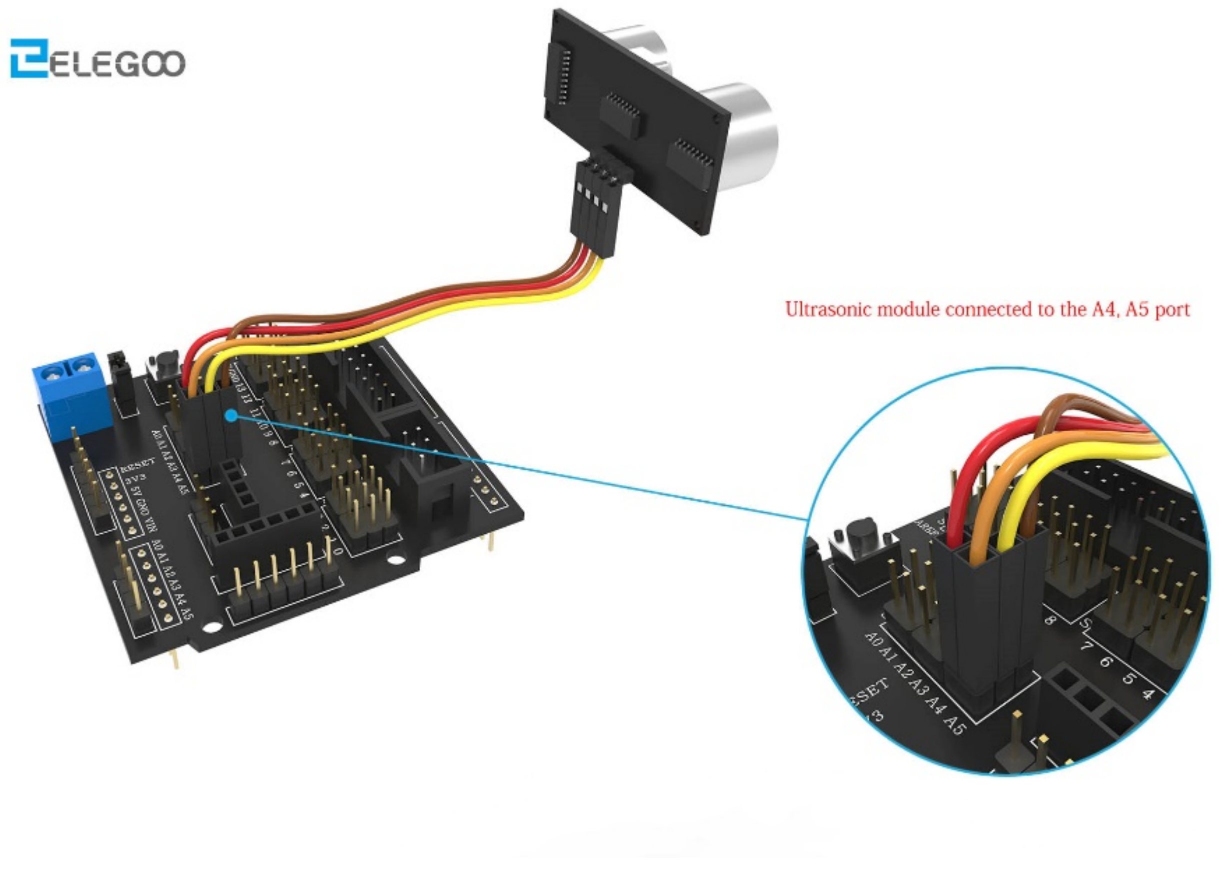
So you have to change the ENA Enable from Digital Pin 10 to Digital Pin 11

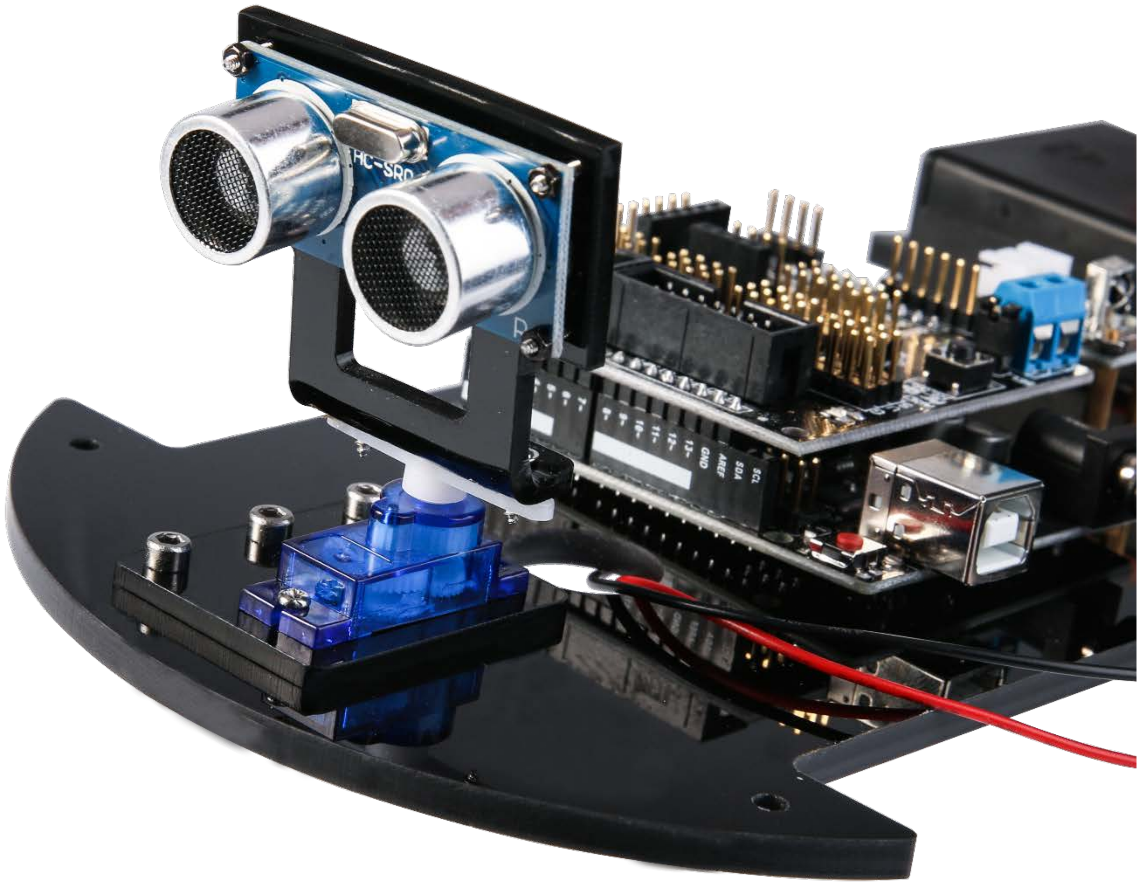

11 10 9 8
7 6 5 4
3 2 1
Sensor Shield v5.0
AREF GND 13 12
Arduino
RESET
G
V
S
A0 A1 A2 A3 A4 A5
COM
SCL
IIC
APC220
ANALOG IN
A0 A1 A2 A3 A4 A5
POWER
RESET 3V3 5V GND VIN
PWR
SEL
SD+
URF 01+

servo



Ultrasonic module





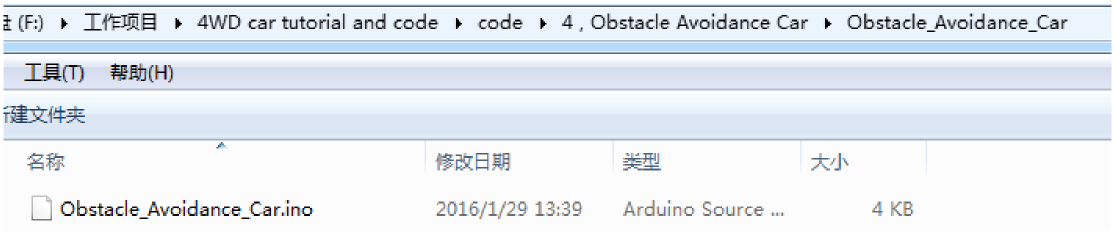
Ⅱ. Upload program

```
#include <Servo.h> //servo library
Servo myservo; // create servo object to control servo
int Echo = A4;
int Trig = A5;
int in1 = 9;
int in2 = 8;
int in3 = 7;
int in4 = 6;
int ENA = 5;
int ENB = 11;
int ABS = 130;
int rightDistance = 0,leftDistance = 0,middleDistance = 0 ;
void _mForward()
{
 analogWrite(ENA,ABS);
 analogWrite(ENB,ABS);
  digitalWrite(in1,HIGH);//digital output
  digitalWrite(in2,LOW);
  digitalWrite(in3,LOW);
  digitalWrite(in4,HIGH);
 Serial.println("go forward!");
}

void _mBack()
{
 analogWrite(ENA,ABS);
 analogWrite(ENB,ABS);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,HIGH);
  digitalWrite(in4,LOW);
 Serial.println("go back!");
}

void _mleft()
{
 analogWrite(ENA,ABS);
 analogWrite(ENB,ABS);
  digitalWrite(in1,HIGH);
```

```
   digitalWrite(in2,LOW);
   digitalWrite(in3,HIGH);
   digitalWrite(in4,LOW);
 Serial.println("go left!");
}

void _mright()
{
 analogWrite(ENA,ABS);
 analogWrite(ENB,ABS);
   digitalWrite(in1,LOW);
   digitalWrite(in2,HIGH);
   digitalWrite(in3,LOW);
   digitalWrite(in4,HIGH);
 Serial.println("go right!");
}
void _mStop()
{
   digitalWrite(ENA,LOW);
   digitalWrite(ENB,LOW);
   Serial.println("Stop!");
}
 /*Ultrasonic distance measurement Sub function*/
int Distance_test()
{
   digitalWrite(Trig, LOW);
   delayMicroseconds(2);
   digitalWrite(Trig, HIGH);
   delayMicroseconds(20);
   digitalWrite(Trig, LOW);
   float Fdistance = pulseIn(Echo, HIGH);
   Fdistance= Fdistance/58;
   return (int)Fdistance;
```

```
}

void setup()
{
   myservo.attach(3);// attach servo on pin 3 to servo object
   Serial.begin(9600);
   pinMode(Echo, INPUT);
   pinMode(Trig, OUTPUT);
   pinMode(in1,OUTPUT);
   pinMode(in2,OUTPUT);
   pinMode(in3,OUTPUT);
   pinMode(in4,OUTPUT);
   pinMode(ENA,OUTPUT);
   pinMode(ENB,OUTPUT);
   _mStop();
}

void loop()
{
      myservo.write(90);//setservo position according to scaled value
      delay(500);
      middleDistance = Distance_test();
      #ifdef send
      Serial.print("middleDistance=");
      Serial.println(middleDistance);
      #endif

      if(middleDistance<=20)
      {
         _mStop();
         delay(500);
         myservo.write(5);
         delay(1000);
         rightDistance = Distance_test();

         #ifdef send
         Serial.print("rightDistance=");
         Serial.println(rightDistance);
         #endif

         delay(500);
           myservo.write(90);
         delay(1000);
         myservo.write(180);
         delay(1000);
         leftDistance = Distance_test();

         #ifdef send
         Serial.print("leftDistance=");
         Serial.println(leftDistance);
         #endif

         delay(500);
         myservo.write(90);
         delay(1000);
         if(rightDistance>leftDistance)
         {
             _mright();
             delay(360);
           }
           else if(rightDistance<leftDistance)
           {
             _mleft();
             delay(360);
           }
           else if((rightDistance<=20) | | (leftDistance<=20))
           {
             _mBack();
             delay(180);
           }
           else
           {
             _mForward();
           }
       }
       else
             _mForward();
   }
```

Open the file Obstacle_Avoidance_Car\Obstacle_Avoidance_Car.ino


盘 (F:) ▸ 工作项目 ▸ 4WD car tutorial and code ▸ code ▸ 4 , Obstacle Avoidance Car ▸ Obstacle_Avoidance_Car

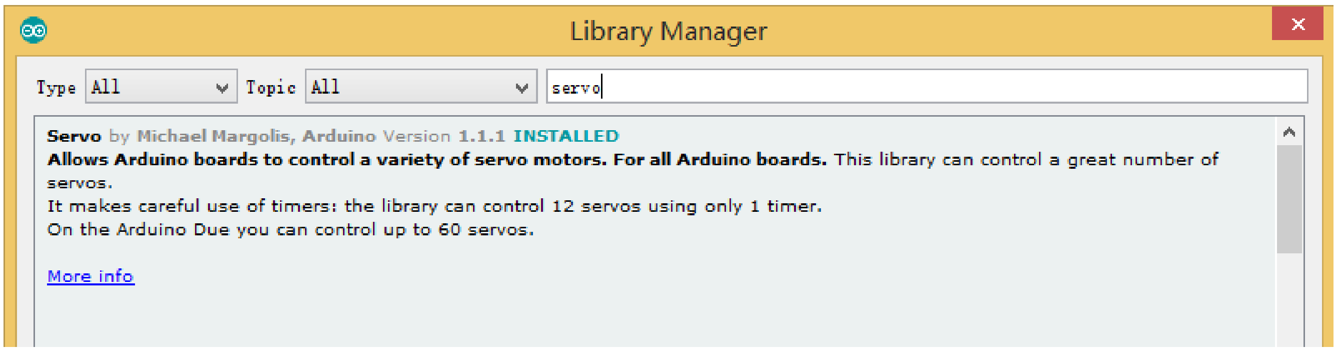
工具(T) 帮助(H)

新建文件夹

名称	修改日期	类型	大小
Obstacle_Avoidance_Car.ino	2016/1/29 13:39	Arduino Source ...	4 KB

Because the program uses the library <servo.h>, so we need to install the library at first.

Open the Sketch---Include Library---Manage Libraries


Library Manager

Type All Topic All servo

Servo by **Michael Margolis, Arduino** Version **1.1.1** **INSTALLED**
Allows Arduino boards to control a variety of servo motors. For all Arduino boards. This library can control a great number of servos.
It makes careful use of timers: the library can control 12 servos using only 1 timer.
On the Arduino Due you can control up to 60 servos.

More info

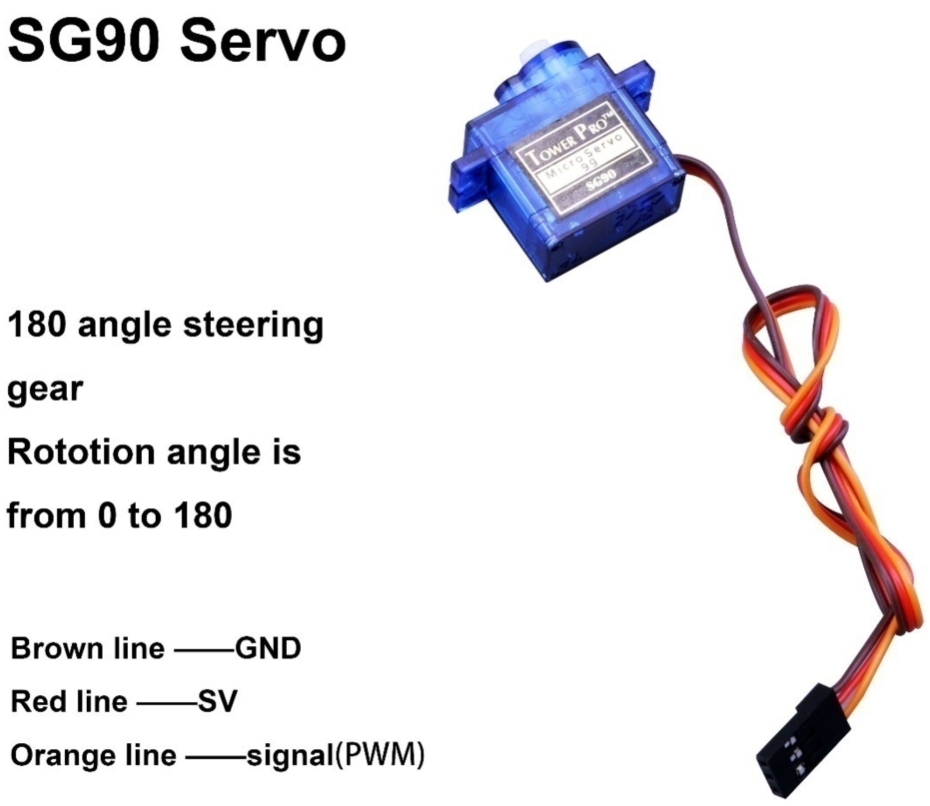
Search servo and then install the newest version.

After uploading the program to the UNO control board, disconnect the cable, put the vehicle on the ground and switch on the power supply.

You will see that the vehicle will move forward and the cloud platform keeps rotating to make the distance measuring sensors operate continuously. If there are obstacles ahead, the cloud platform will stop and the vehicle will change its direction to bypass the obstacle. After bypassing the obstacle, the cloud platform will keep rotating again and the vehicle will also move on.

Ⅲ. Introduction of principle

First of all, let's learn about the SG90 Servo:



Classification: 180 steering gear

Normally the servo has 3 controlling line: power supply, ground and sign.

Definition of the servo pins: brown line——GND, red line——5V, orange——signal.

How does servo work:

The signal modulation chip in the servo receives signals from the controller board then the servo will get the basic DC voltage. There is also a reference circuit inside the servo which will produce a standard voltage. These two voltages will compare to each other and the difference will be output. Then the motor chip will receive the difference and decide the rotational speed, direction and angel. When there is no difference between the two voltages, the servo will stop.

How to control the servo:

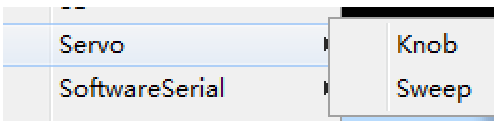
To control the servo rotation, you need to make the time pulse to be about 20ms and the high level pulse width to be about 0.5ms~2.5ms, which is consistent with the angle limited of the servo.

Taking 180 angle servo for example, corresponding control relation is as below:

0.5ms	0 degree
1.0ms	45 degree
1.5ms	90 degree
2.0ms	135 degree
2.5ms	180 degree

The program:

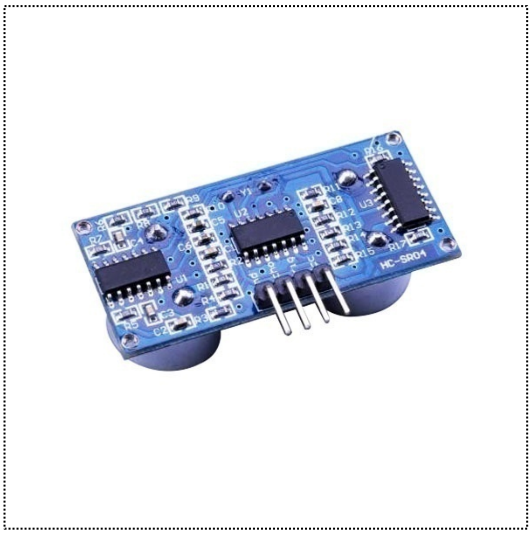
Arduino has library file.<Servo.h>



```
Servo myservo;   // create servo object to control servo
 myservo.attach(3);   // attach servo on pin 3 to servo object
 myservo.write(90); //set servo position according to scaled value
```

You can drive steering gear in 9 words.

Next, let’s have a look at the ultrasonic sensor module.



Feature of the module: testing distance, high precision module.

Application of the products: robot obstacle avoidance、object testing distance、liquid testing、public security、parking lot testing.

Main technical parameters

(1)：voltage used: DC---5V

(2)：static current: less than 2mA

(3)：level output: higher than 5V

(4)：level output: lower than 0

(5)：detection angle: not bigger than 15 degree

(6)：detecting distance: 2cm-450cm

(7)：high precision: up to 0.2cm

Method of connecting lines: VCC, trig (the end of controlling), echo (the end of receiving), GND

How does the module work:

(1)Apply IO port of TRIG to trigger ranging, give high level signal, at least 10us one time;

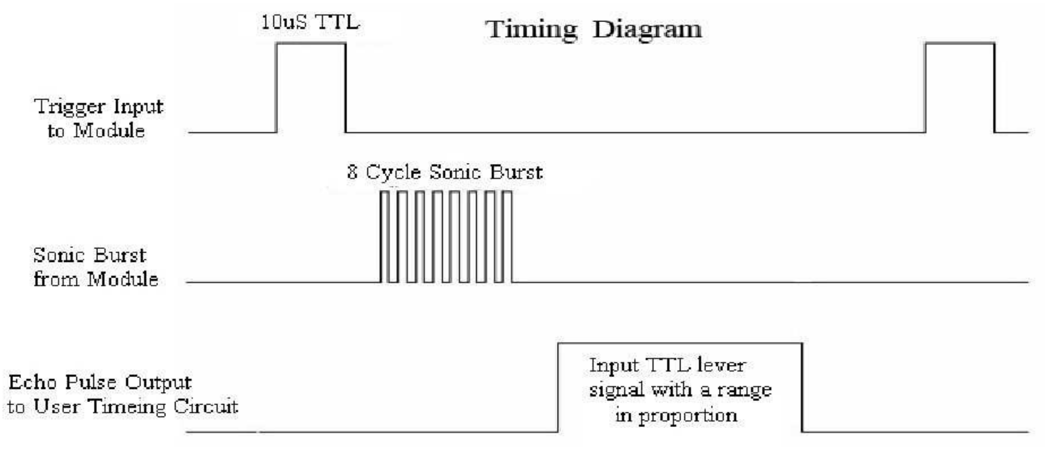
(2)The module sends 8 square waves of 40kz automatically, tests if there are signals returned automatically;

(3)If there are signals received, the module will output a high level pulse through IO port of ECHO, the duration time of high level pulse is the time between the wave sending and receiving. So the module can know the distance according to the time.

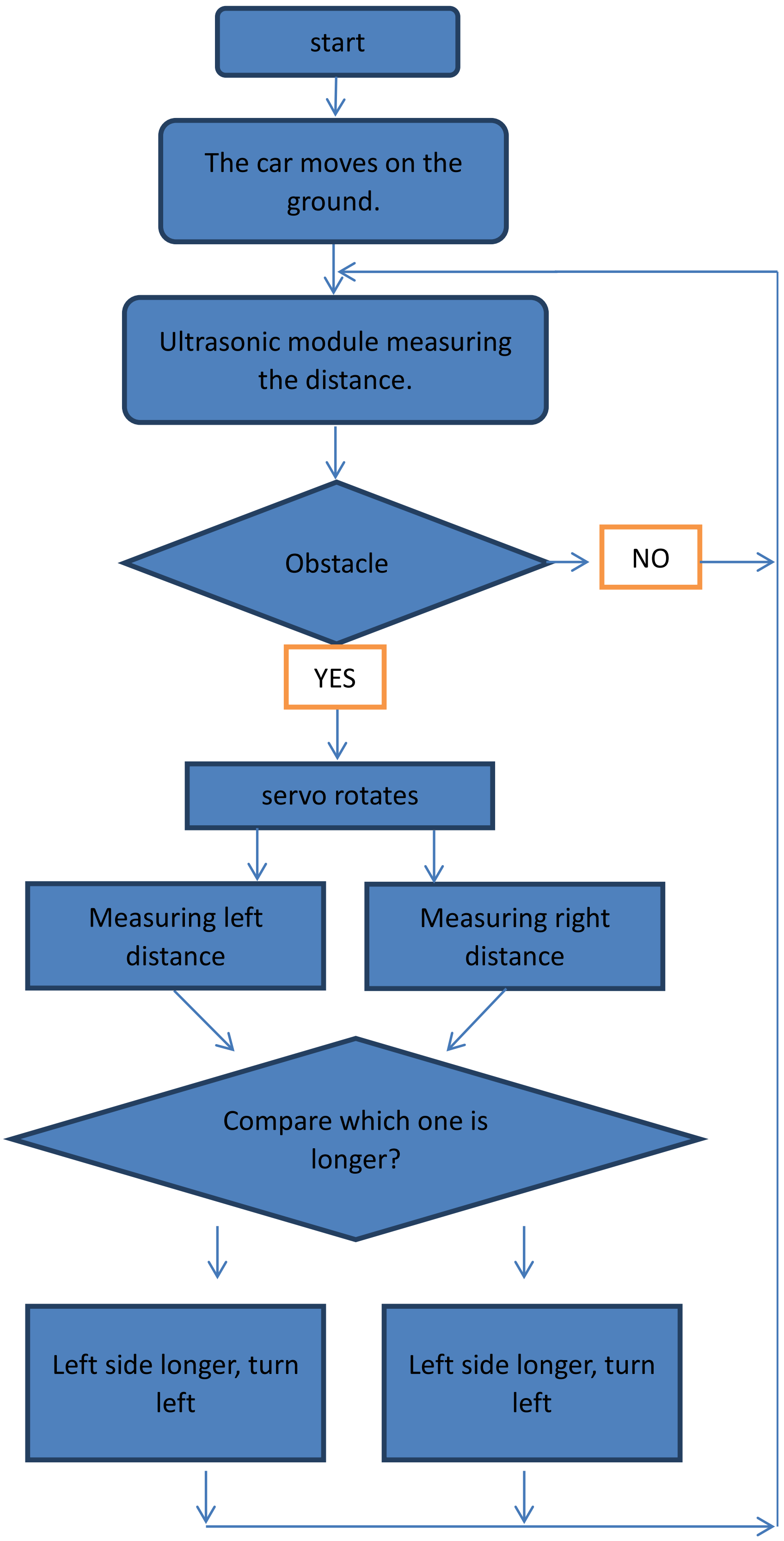
Testing distance= (high level time* velocity of sound (340M/S))/2);

Actual operation:

The Timing diagram is shown below. You only need to supply a short10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.


10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timeing Circuit
Input TTL lever signal with a range in proportion

```
/*Ultrasonic distance measurement Sub function*/
int Distance_test()
{
   digitalWrite(Trig, LOW);
   delayMicroseconds(2);
   digitalWrite(Trig, HIGH);
   delayMicroseconds(20);
   digitalWrite(Trig, LOW);
   float Fdistance = pulseIn(Echo, HIGH);
   Fdistance= Fdistance/58;
   return (int)Fdistance;
}
```


start
The car moves on the ground.
Ultrasonic module measuring the distance.
Obstacle
NO
YES
servo rotates
Measuring left distance
Measuring right distance
Compare which one is longer?
Left side longer, turn left
Left side longer, turn left

From above picture, we can see that the principle of obstacle avoidance car is very simple. The ultrasonic sensor module will detect the distance between the car and the obstacles again and again and sending the data to the controller board, then the car will stop and rotate the servo to detect the left side and right side. After compared the distance from the different side, the car turn to the side which has longer distance and move forward. Then the ultrasonic sensor module detects the distance again.

```
if(rightDistance>leftDistance)
        {
          _mright();
          delay(360);
         }
         else if(rightDistance<leftDistance)
         {
          _mleft();
          delay(360);
         }
         else if((rightDistance<=20) | | (leftDistance<=20))
         {
          _mBack();
          delay(180);
         }
         else
         {
          _mForward();
         }
     }
     else
          _mForward();
}
```