About Arduino:

Arduino is an open-source hardware project platform. 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’s widely applied in robot field.

About keyestudio:

keyestudio is a newly developed brand of keyes, a company specialized in open-source hardware. We have professional teams dedicated in product R&D, manufacturing and selling.

We provide supports in both hardware & software for electronic enthusiasts around the world.In this open-source world, what we have, we would like to share.

Catalog

1.Introduction

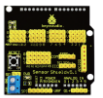
Mini tank robot is a learning application development system of microcontroller based on Arduino.
It has functions such as ultrasonic obstacle avoidance, Bluetooth remote control. This kitcontains many interesting programs.
It can also be expanded with external circuit modules to Have other functions.
This kit is designed to help you interestingly learn Arduino. You can learn Arduino MCU development ability while having fun.

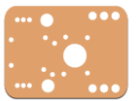
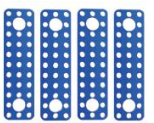
2. Parameters

1. Motor Parameters: 6V, 150rpm/min
2. Use L298P driver module for motor control.
3. Equipped with Ultrasonic module,
4. Equipped with Bluetooth wireless module, can remotely control the robot after pairing with mobile phone Bluetooth.
5. Can be connected to external 7 ~ 12V power supply; with various sensor modules, it can realize various functions.

3. Component List

No.	Name	QTY	Picture
1	keyestudio UNO R3 Controller	1	
2	keyestudio L298P Shield	1	

3	keyestudio V5 Sensor Shield	1	
4	HC-SR04 Ultrasonic Sensor	1	
5	keyestudio Bluetooth Module (HC-06)	1	
6	Plastic Platform (PC)	1	
7	Servo Motor	1	

8	Transparent Acrylic Board	1	
9	Metal Holder	4	
10	Tank Driver Wheel	2	
11	Tank Load-bearing Wheel	2	
12	Caterpillar Band	2	

13	Metal Motor	2	
14	Copper Coupler	2	
15	18650 2-cell Battery Case	1	
16	USB Cable (1m)	1	
17	Copper Bush	2	

18	Flange Bearing	4	
19	Hexagon Copper Bush (M3*10MM)	8	
20	Hexagon Copper Bush (M3*45MM)	4	
21	Round Screw (M3*6MM)	12	
22	M3*6MM Flat Head Screw	2	

23	Round Screw (M4*35MM)	4	
24	Inner Hexagon Screw (M3*8MM)	12	
25	Inner Hexagon Screw (M3*20MM)	6	
26	Inner Hexagon Screw (M3*25MM)	6	
27	Inner Hexagon Screw (M4*10MM)	6	
28	Inner Hexagon Screw (M4*50MM)	2	

29	M3 Nut	6	
30	M4 Self-locking Nut	2	
31	M4 Nut	15	
32	Connector Wire (150mm, black)	2	
33	Connector Wire (150mm, red)	2	
34	F-F Dupont Wire (20CM, 4Pin)	1	

35	Supporting Part (27*27*16MM, Blue)	2	
36	Winding Wire (12CM)	1	

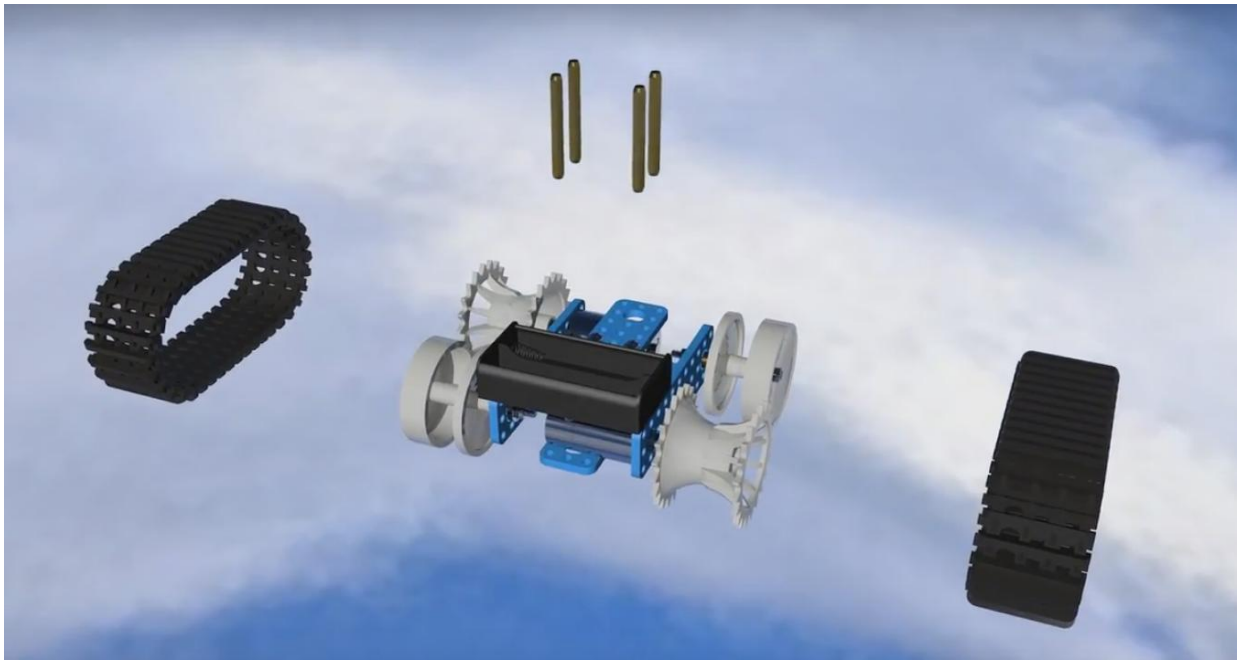
****

* Self-prepare Parts

1	18650 Rechargeable Battery	2	
2	18650 Charger	1	

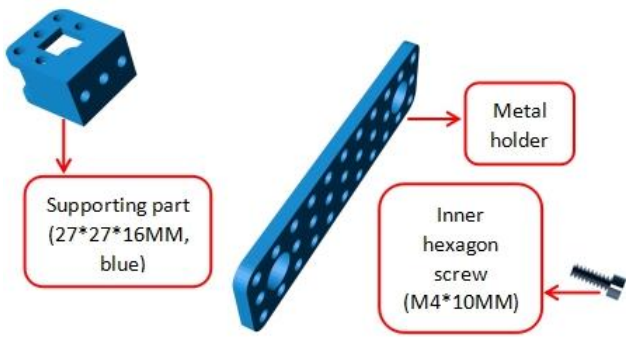
4. Assembly Video Address

https://arduino-ua.com/



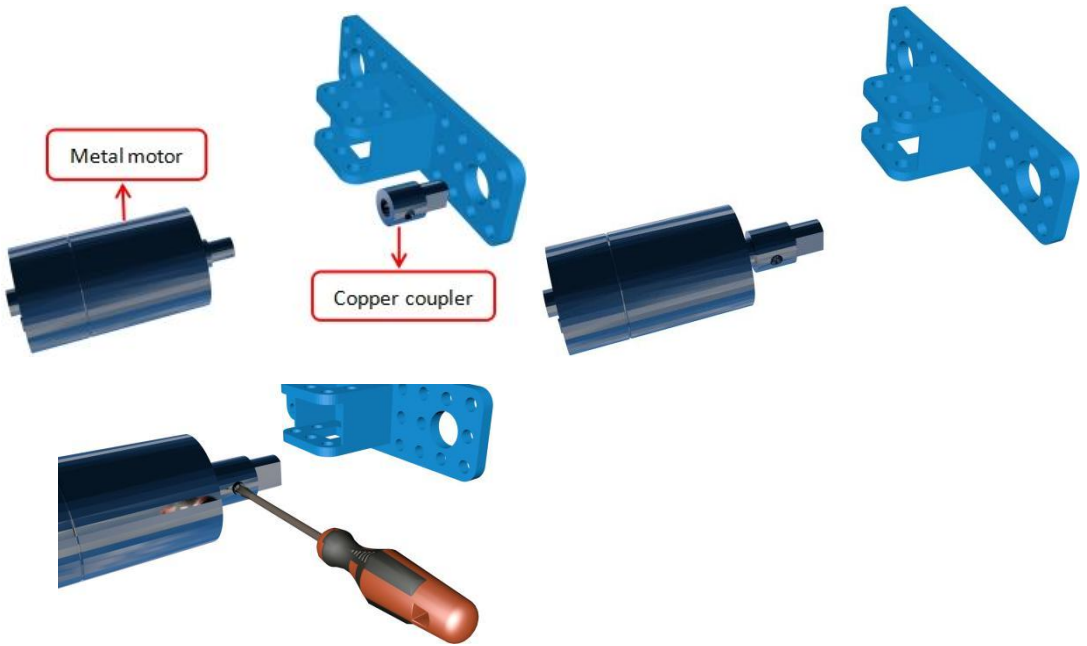
5.Installation Method

1. Attach the supporting part to metal holder with screws and nuts.

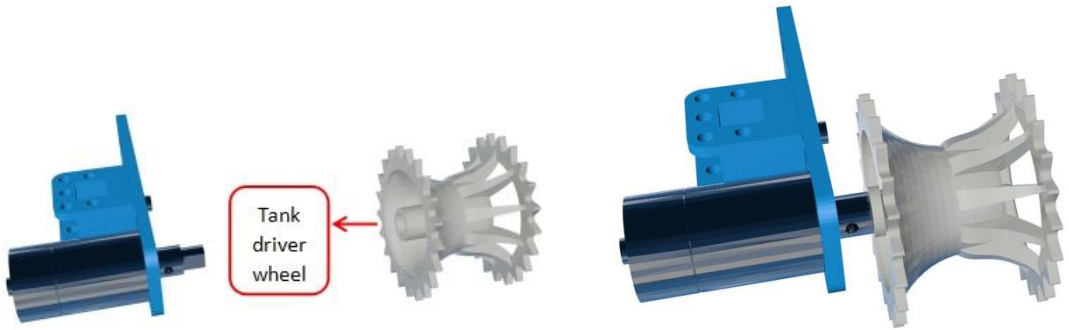


2. Screw nuts to the screws.

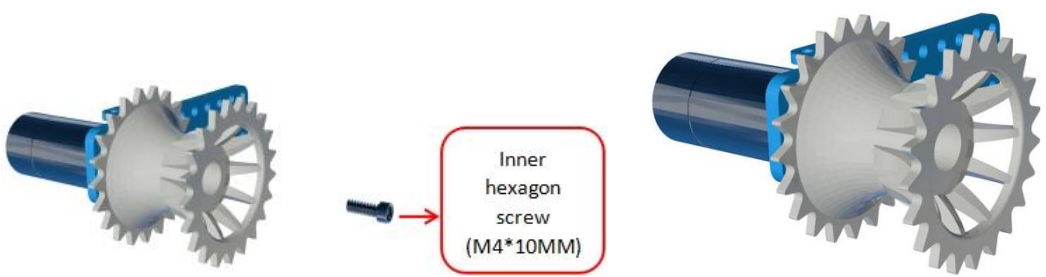
3. Plug the coupler into the motor and screw with a screwdriver.



4. Plug the driver wheel into the motor.

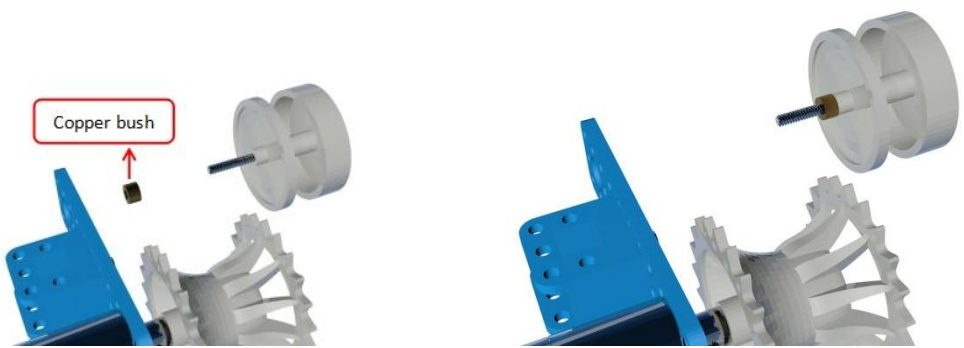


5. Fix the driver wheel with a screw.



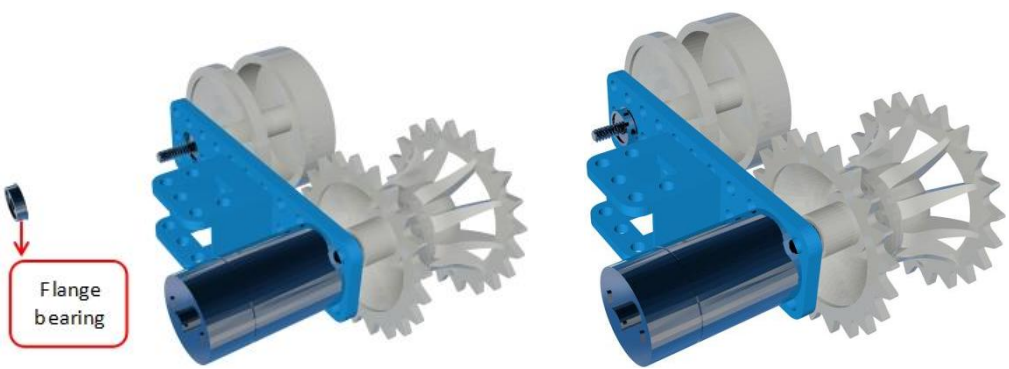
6.Insert a screw into the hole of load-bearing wheel.

7. Twist the bush with the screw inserted into the wheel.

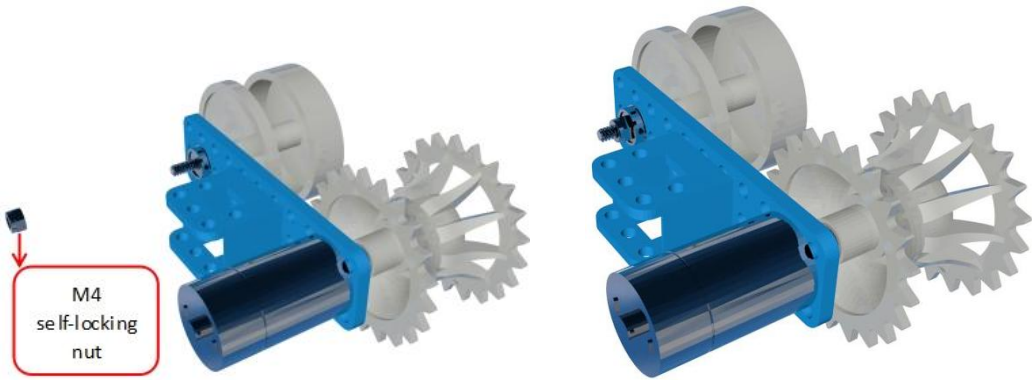


8. Add a flange bearing to the screw inserted into the wheel.

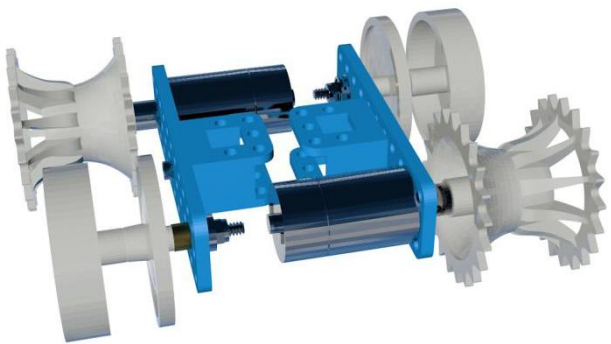
9. Plug the whole load-bearing wheel into the metal holder.



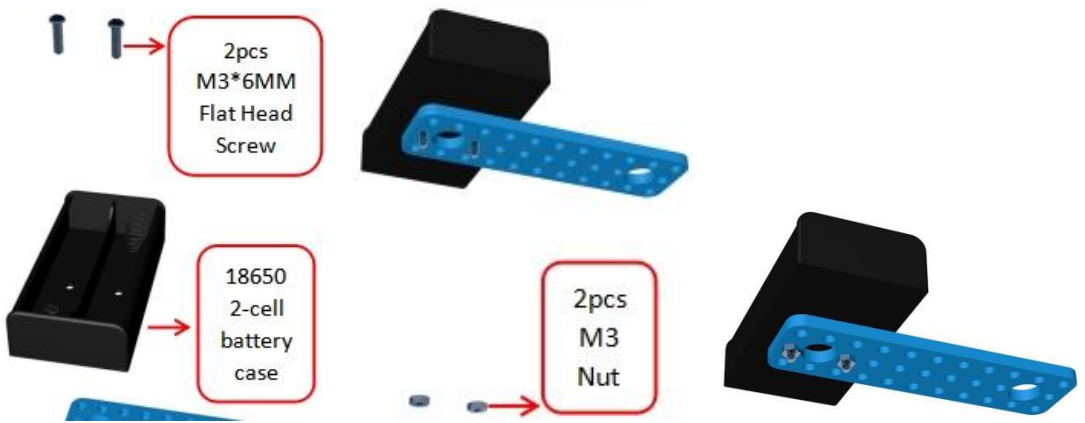
10. Fix the load-bearing wheel with nuts.



11. Assembly another according above steps.

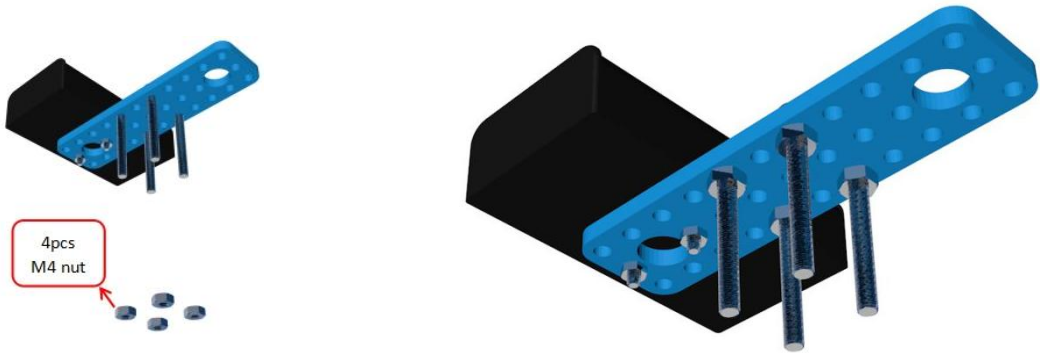


12. Attach battery case to metal holder with screws and nuts.



13. Screw 4 screws in the metal holder beside the battery case.

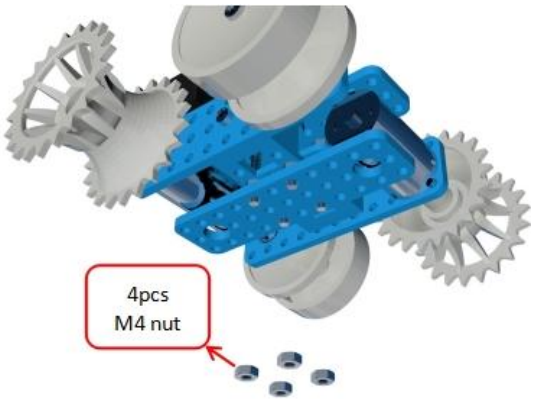
14. Twist 4 nuts to the screws.

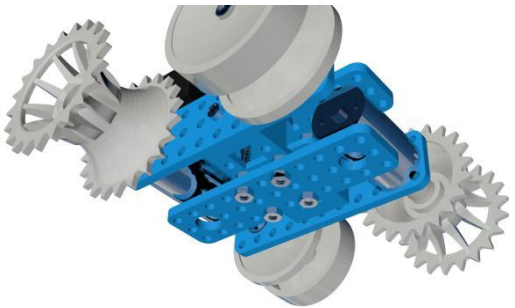


15. Put all installed component together as shown in below figure.

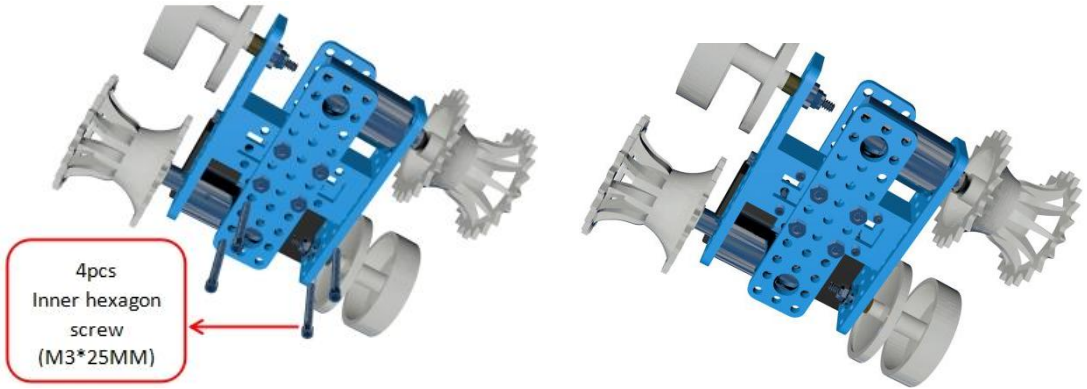
16. Fix components with nuts and screws.

17. Attach another piece of metal holder to them with nuts.

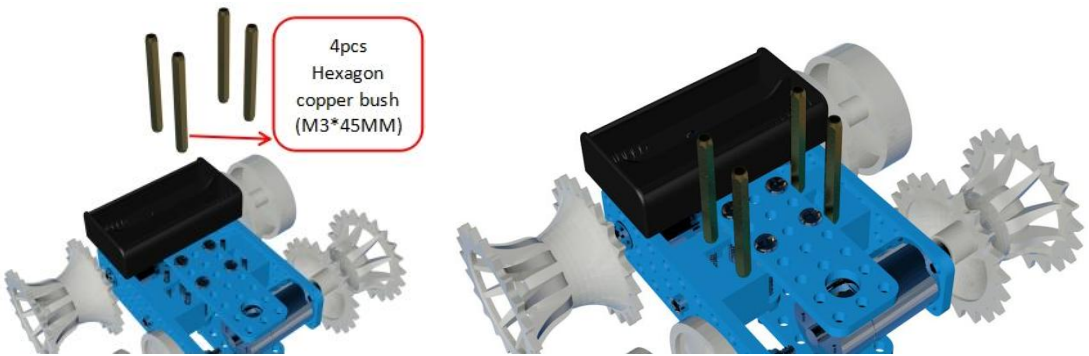




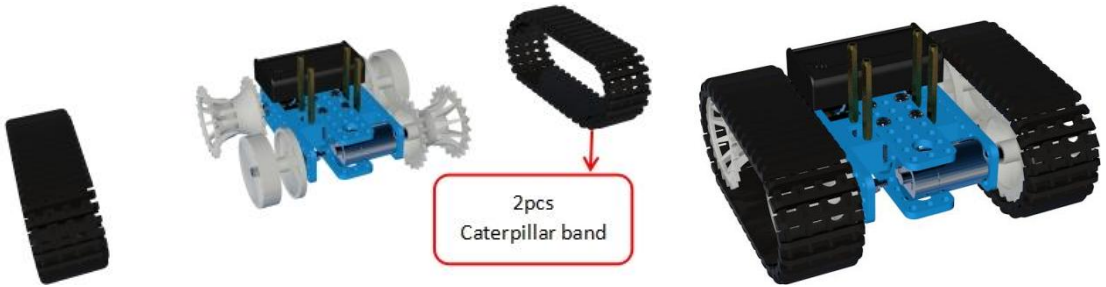
18. Plug 4 screws into the metal holder.



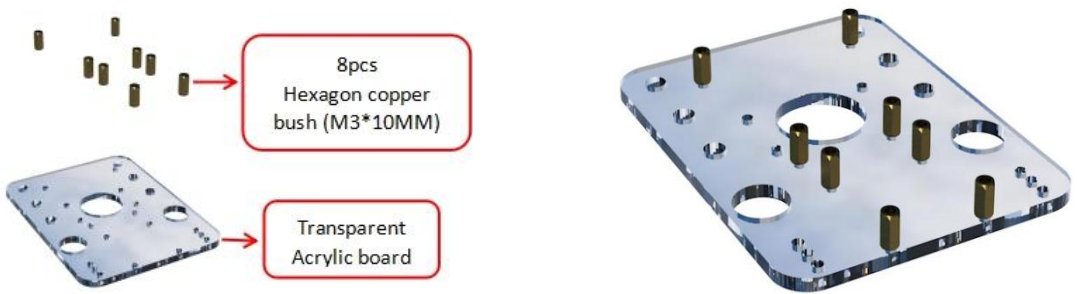
19. Slot the screws into the copper bush.



20. Install the caterpillar band on the wheels.

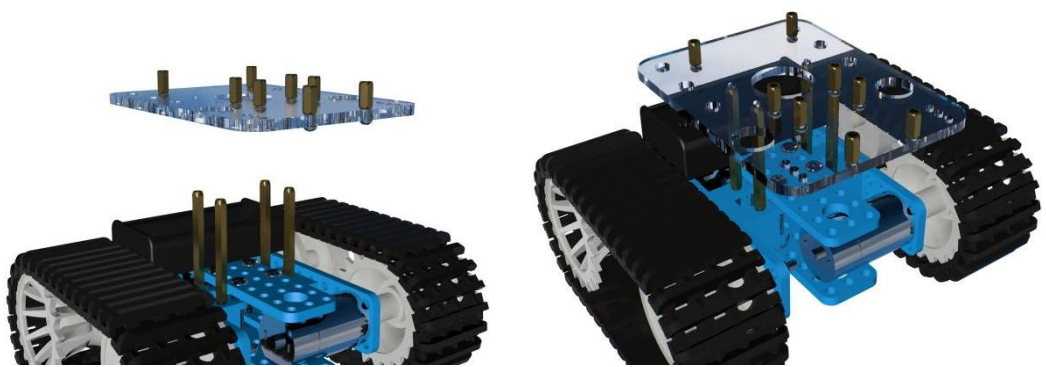


21. Place 8 copper bushes on Acrylic board.



22.Slot 8 screws into 8 copper bushes.

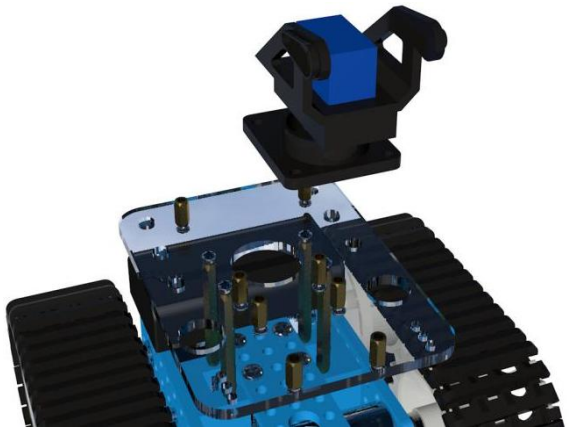
23. Then put the Acrylic board upon 4 copper bushes.

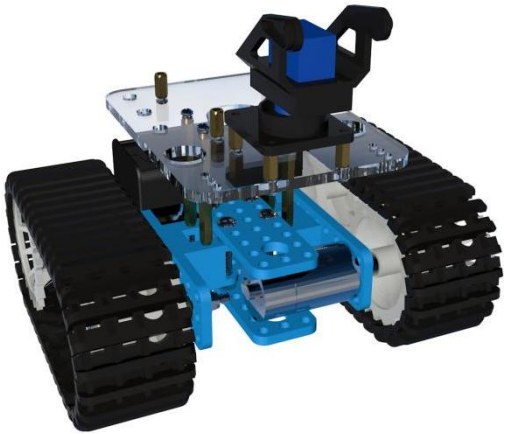


24. Fix the Acrylic board with screws.

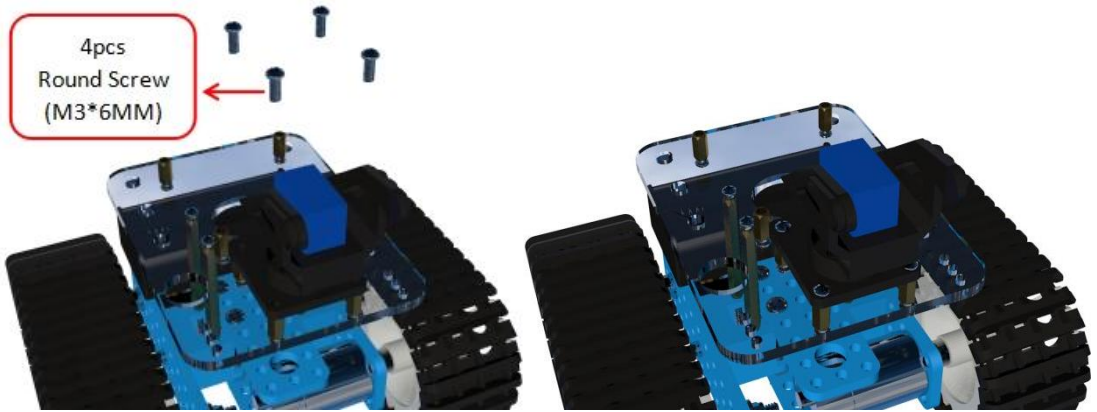
25. Install platform steering engine, and first place a plastic cross into the component.

26. Then put the whole steering engine upon 4 copper bushes.



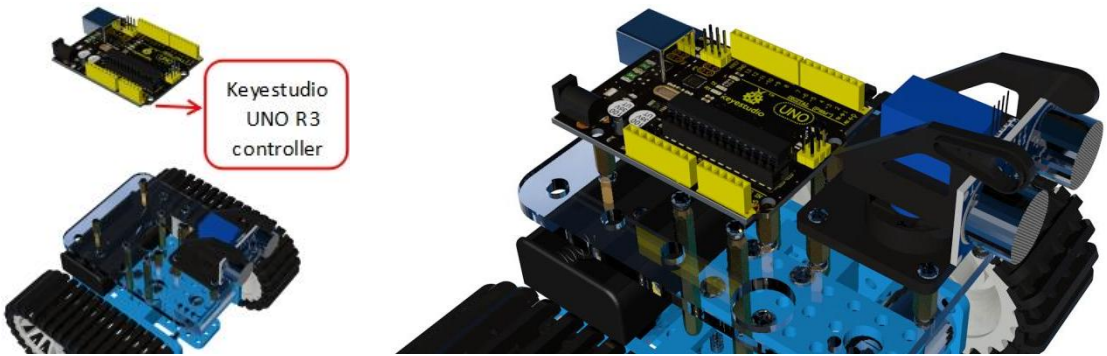


27. Fix the steering engine with 4 screws.

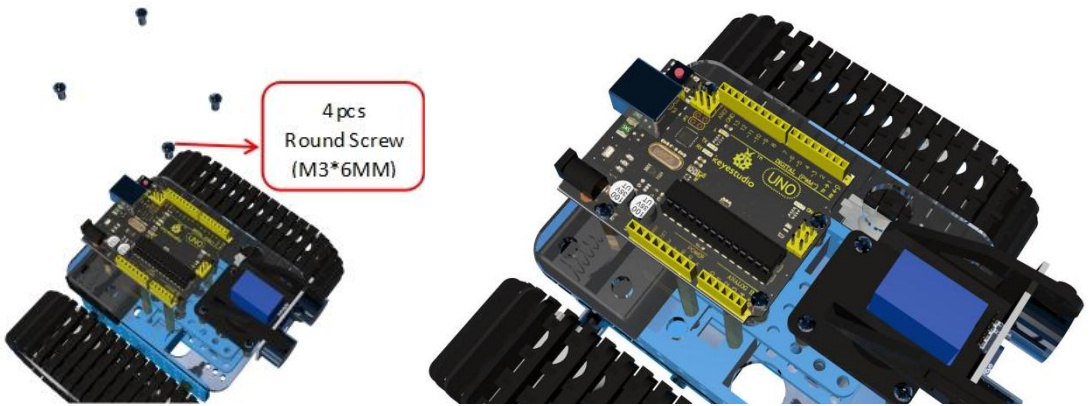


28. Attach the ultrasonic sensor to steering engine with winding band.

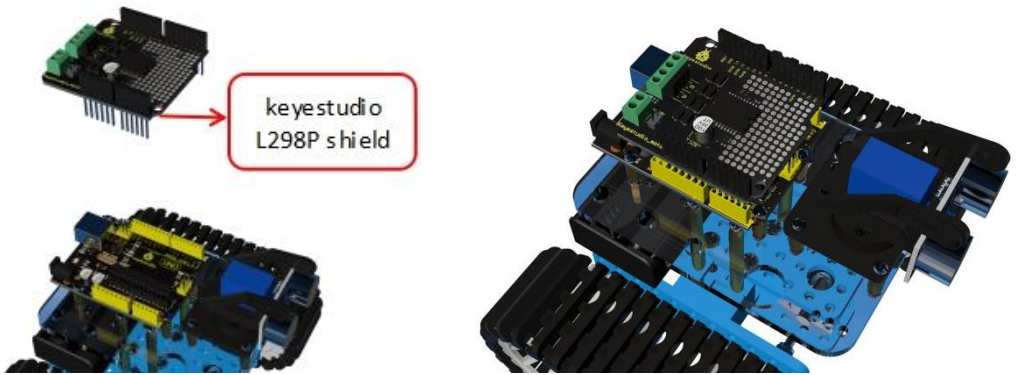
29. Place UNO R3 controller above 8 copper bushes.



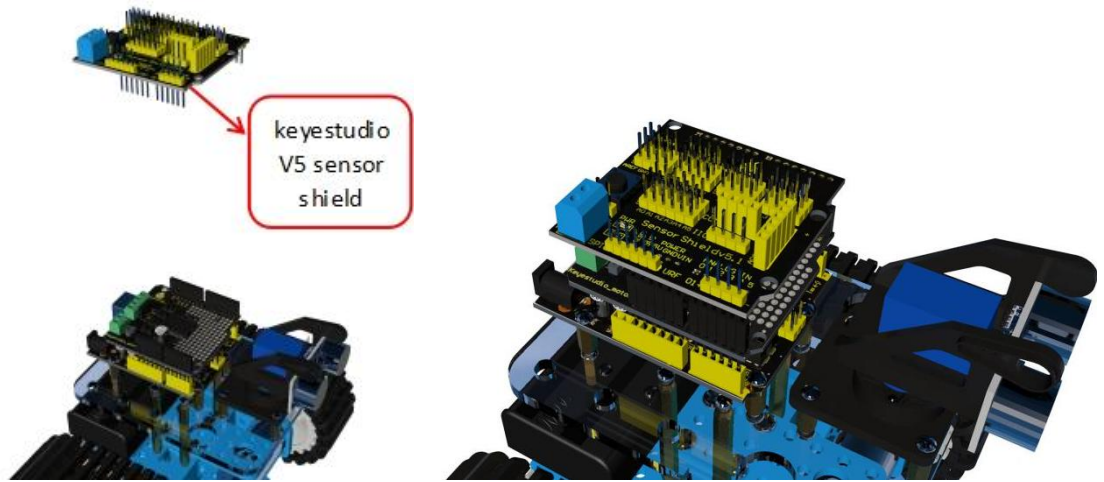
30. Then fix the UNO R3 with screws.



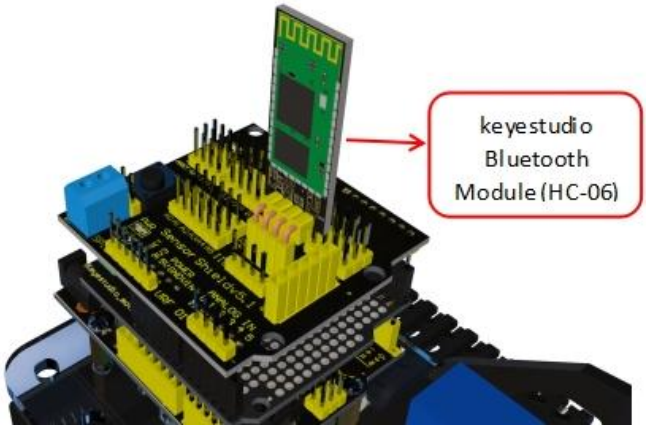
31. Plug the L298P shield into UNO R3.



32. Then plug V5 sensor shield into L298P shield.

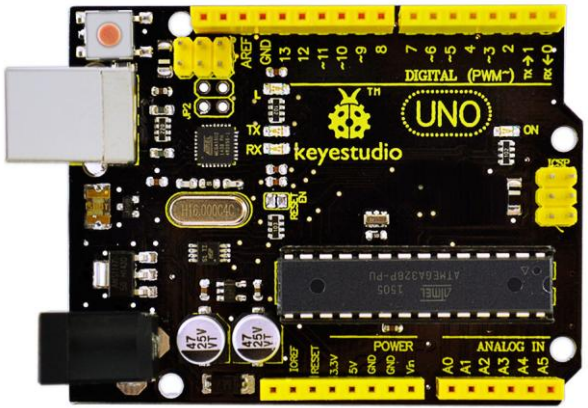


33. Plug Bluetooth module into V5 sensor shield.



6.UNO R3 Video Address

https://arduino-ua.com

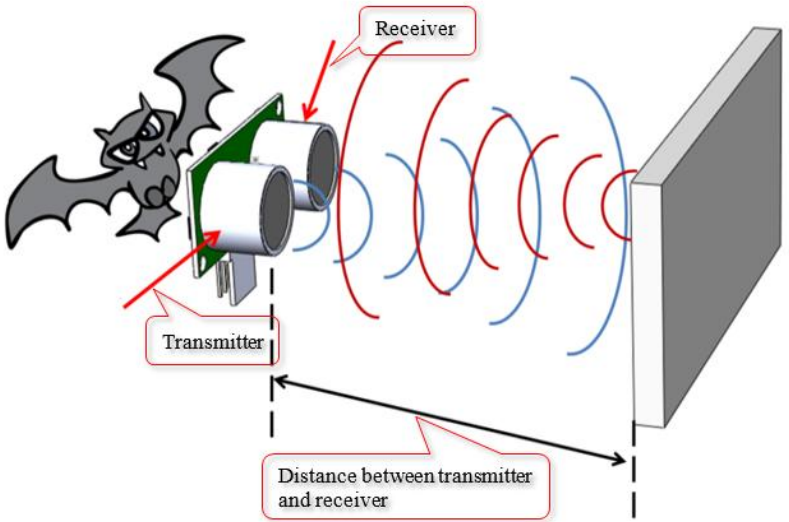

DIGITAL (PWM~)
UNO
keyestudio
POWER
ANALOG IN

7. Project Details

Project 1:Ultrasonic Sensor

1. Introduction

The HC-SR04 Ultrasonic Sensor is a very affordable proximity distance sensor that has been used mainly for object avoidance in various robotics projects. It essentially gives your Arduino eyes spacial awareness and can prevent your robot from crashing or falling off a table. It has also been used in turret applications, water level sensing, and even as a parking sensor. This simple project will use the HC-SR04 sensor with an Arduino and a Processing sketch to provide a neat little interactive display on your computer screen.



2.Specification

Working Voltage: DC 5V

Working Current: 15mA

Working Frequency: 40Hz

Max Range: 4m

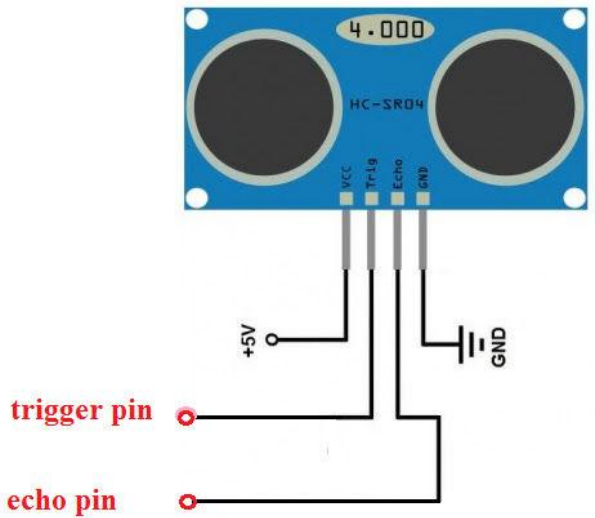
Min Range: 2cm

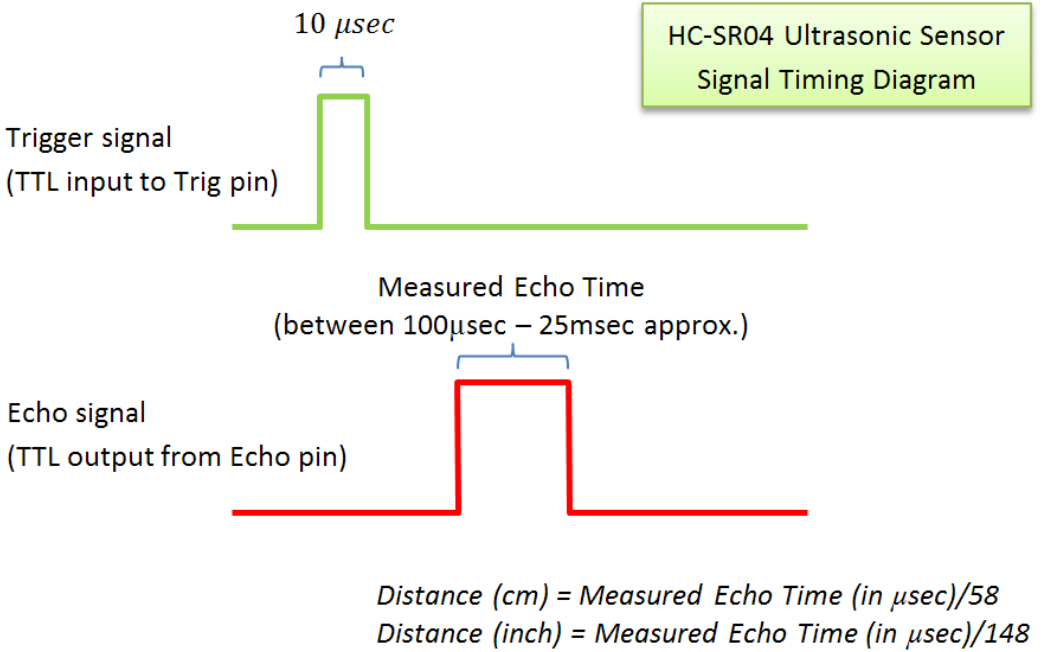
Measuring Angle: 15 degree

Trigger Input Signal: 10μS TTL pulse

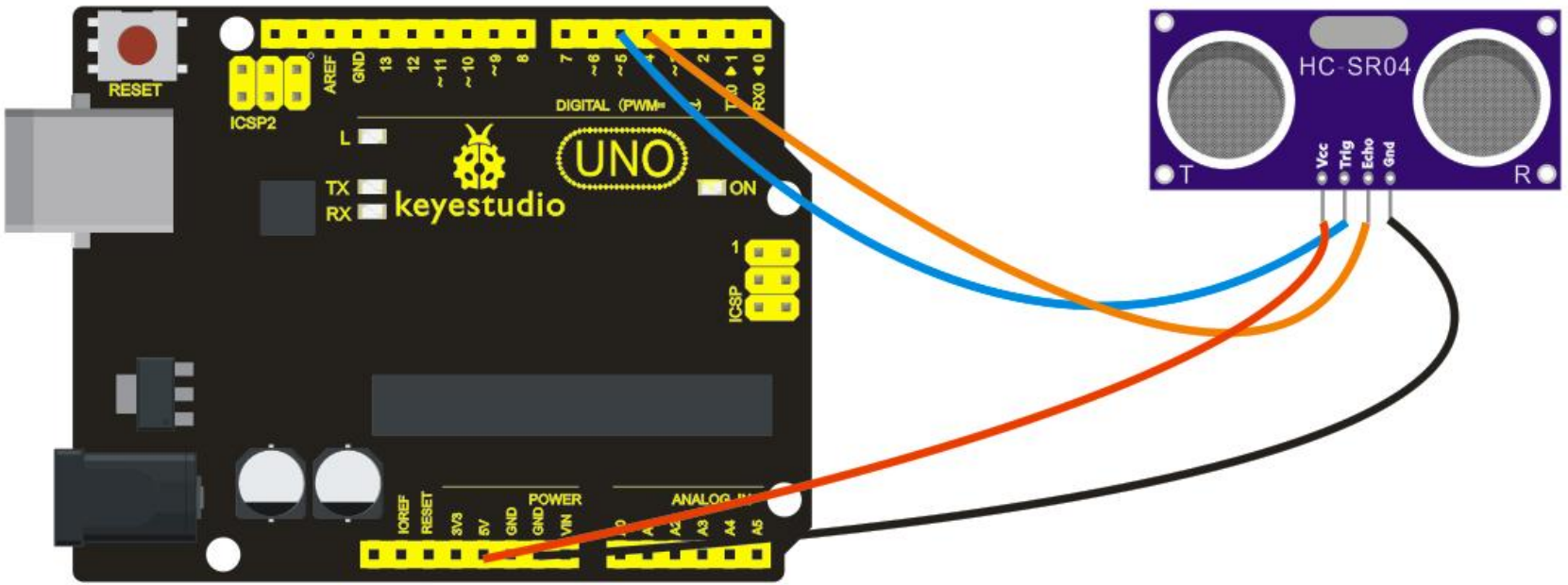
Echo Output Signal Input TTL lever signal and the range in proportion

3.Connection Diagram





Distance (cm) = Measured Echo Time (in µsec)/58
Distance (inch) = Measured Echo Time (in µsec)/148


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=
L
TX
RX
keyestudio
UNO
ON
1
ICSP
POWER
IOREF
RESET
3V3
5V
GND
GND
VIN
A3
A4
A5
HC-SR04
Vcc
Trig
Echo
Gnd
T
R

4.Sample Code

```
  VCC to Arduino 5v
  GND to Arduino GND
  Echo to Arduino pin 4
  Trig to Arduino pin 5
  ************************************************************************************************************************************
#define echoPin 4 // Echo Pin
#define trigPin 5 // Trigger Pin
#define LEDPin 13 // Onboard LED
int maximumRange = 200; // Maximum range needed
int minimumRange = 0; // Minimum range needed
long duration, distance; // Duration used to calculate distance

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(LEDPin, OUTPUT); // Use LED indicator (if required)
}
```

```
void loop() {
/* The following trigPin/echoPin cycle is used to determine the
  distance of the nearest object by bouncing soundwaves off of it. */
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);

  //Calculate the distance (in cm) based on the speed of sound.
  distance = duration/58.2;

  if (distance >= maximumRange || distance <= minimumRange){
  /* Send a negative number to computer and Turn LED ON
  to indicate "out of range" */
  Serial.println("-1");
```

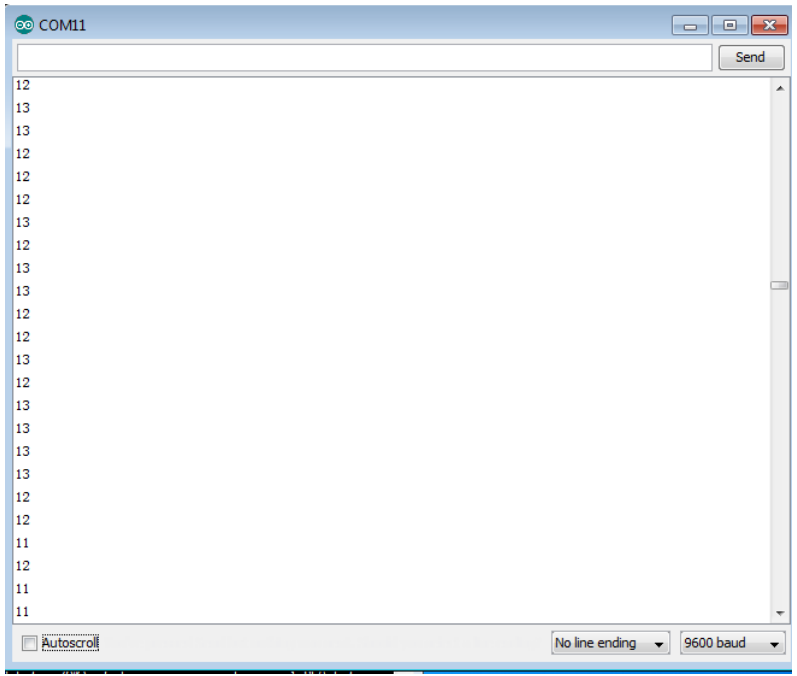
```
  digitalWrite(LEDPin, HIGH);
  }
  else {
  /* Send the distance to the computer using Serial protocol, and
  turn LED OFF to indicate successful reading. */
  Serial.println(distance);
  digitalWrite(LEDPin, LOW);
  }

  //Delay 50ms before next reading.
  delay(50);
}
```

****

5. Result

After connection and uploading, when ultrasonic sensor senses obstacle within sensing area, it is measuring the distance between itself and obstacle and the value of distance is displayed on serial monitor as shown in bellow figure.


COM11
Send
12
13
13
12
12
12
13
12
13
13
12
12
13
12
13
13
13
13
12
12
11
12
11
11
Autoscroll
No line ending
9600 baud

Project 2: Bluetooth Module

1.Introduction

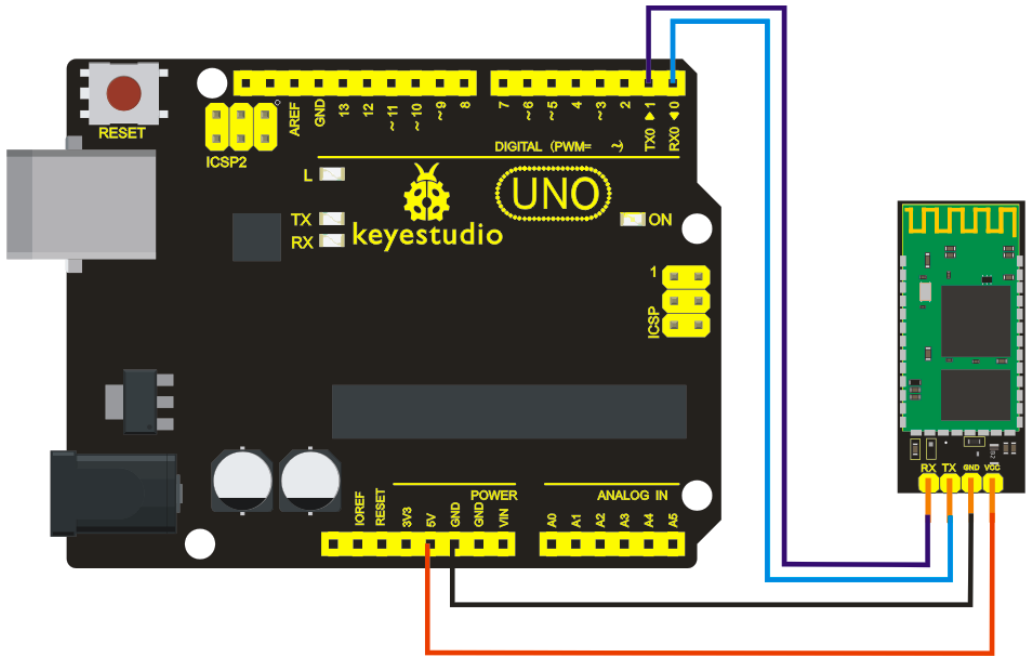
This Bluetooth module can easily achieve serial wireless data transmission. Its operating frequency is among the most popular 2.4GHz ISM frequency band (i.e. Industrial, scientific and medical). It adopts Bluetooth 2.1+EDR standard. In Bluetooth 2.1, signal transmit time of different devices stands at a 0.5 seconds interval so that the workload of bluetooth chip can be reduced substantially and more sleeping time can be saved for bluetooth. This module is set with serial interface, which is easy-to-use and simplifying overall design/development cycle.

2.Specification

Bluetooth Protocol: Bluetooth 2.1+ EDR standard
USB Protocol: USB v1.1/2.0
Operating Frequency: 2.4GHz ISM frequency band
Modulation Mode: Gauss frequency Shift Keying
Transmit Power: ≤ 4dBm, second stage
Sensitivity: ≤-84dBm at 0.1% Bit Error Rate
Transmission Speed: 2.1Mbps(Max)/160 kbps(Asynchronous)； 1Mbps/1Mbps(Synchronous)
Safety Feature: Authentication and encryption

Supported Configuration: Bluetooth serial port (major and minor)
Supply Voltage: +3.3 VDC 50mA
Operating Temperature: -20 to 55℃

3.Connection Diagram



4.Sample Code

****

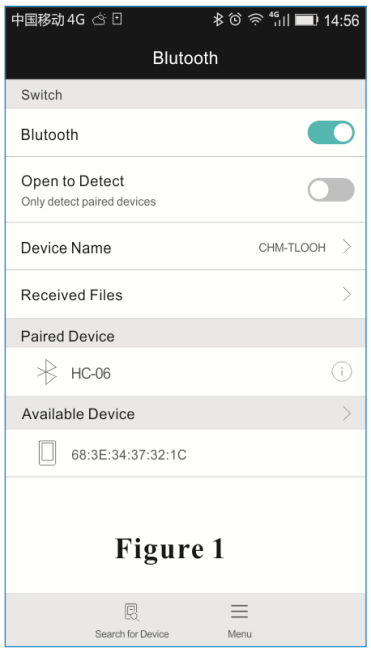
```
  int val;
  int ledpin=13;
  void setup()
  {
  Serial.begin(9600);
  pinMode(ledpin,OUTPUT);
  } void loop()
  { val=Serial.read();
  if(val=='a')
{
digitalWrite(ledpin,HIGH);
delay(250);
digitalWrite(ledpin,LOW);
delay(250);
  Serial.println("keyestudio");
}}
```

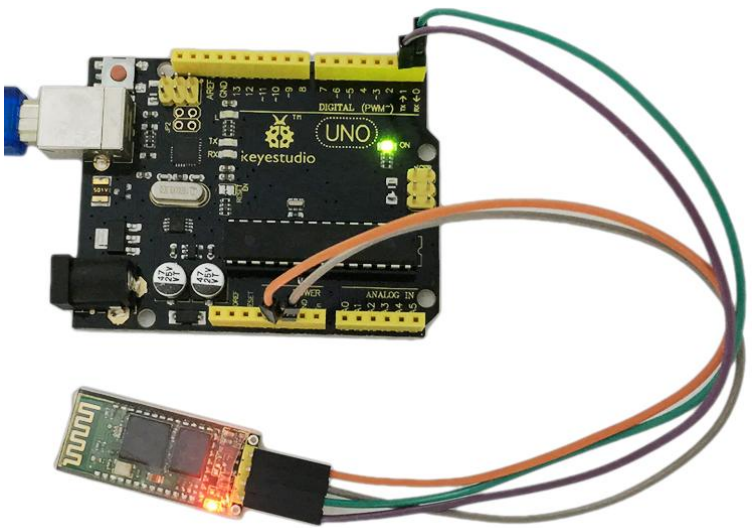
****

5. Result

After power-on, power indicator D1 is on, and LED on Bluetooth module is blinking; open Bluetooth on mobile phone, pair them, input 1234, and finish pairing

as shown in **Figure 1** ; open APP—Bluetooth serial communication assistant, connect it to Bluetooth, select normal mode, complete connection, and LED on Bluetooth module is on as shown in **Figure 2**; input an “a” in the assistant, and display “keyesdudio” in it as shown in **Figure 3**.





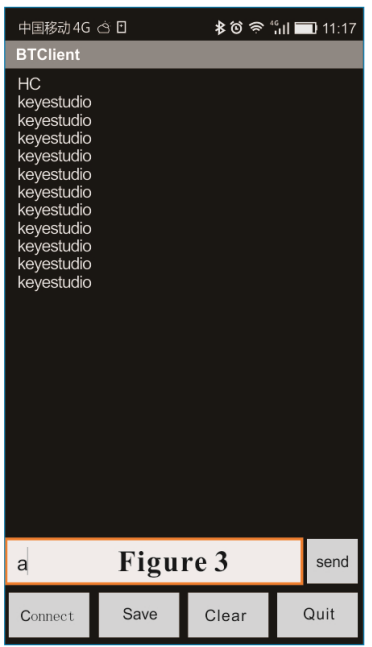
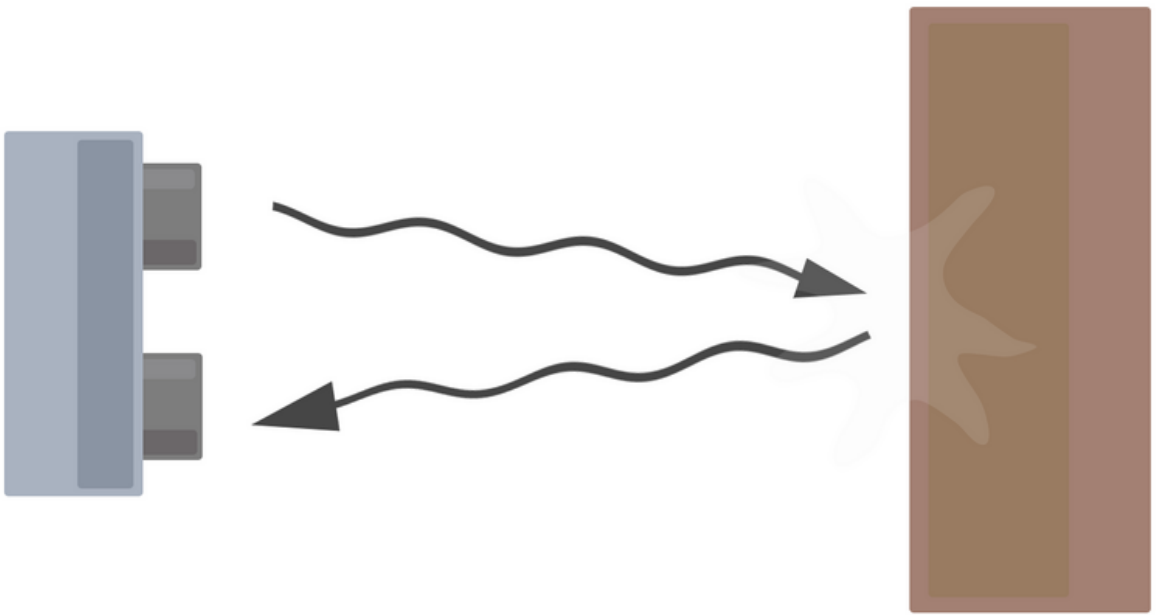


Figure 1 **Figure 2** **Figure 3**

Project 3: Obstacle-avoidance Tank



1.Introduction

This project is a simple obstacle avoidance tank robot system based on Arduino, including the software and hardware design. The controller part is a UNO board. Ultrasonic sensor and servo motors are used to detect whether there are obstacles ahead, and feedback the signal to UNO. UNO will analyze the signal to determine and control the motors movement to adjust Tank moving direction. Therefore the tank robot can automatically avoid obstacles.

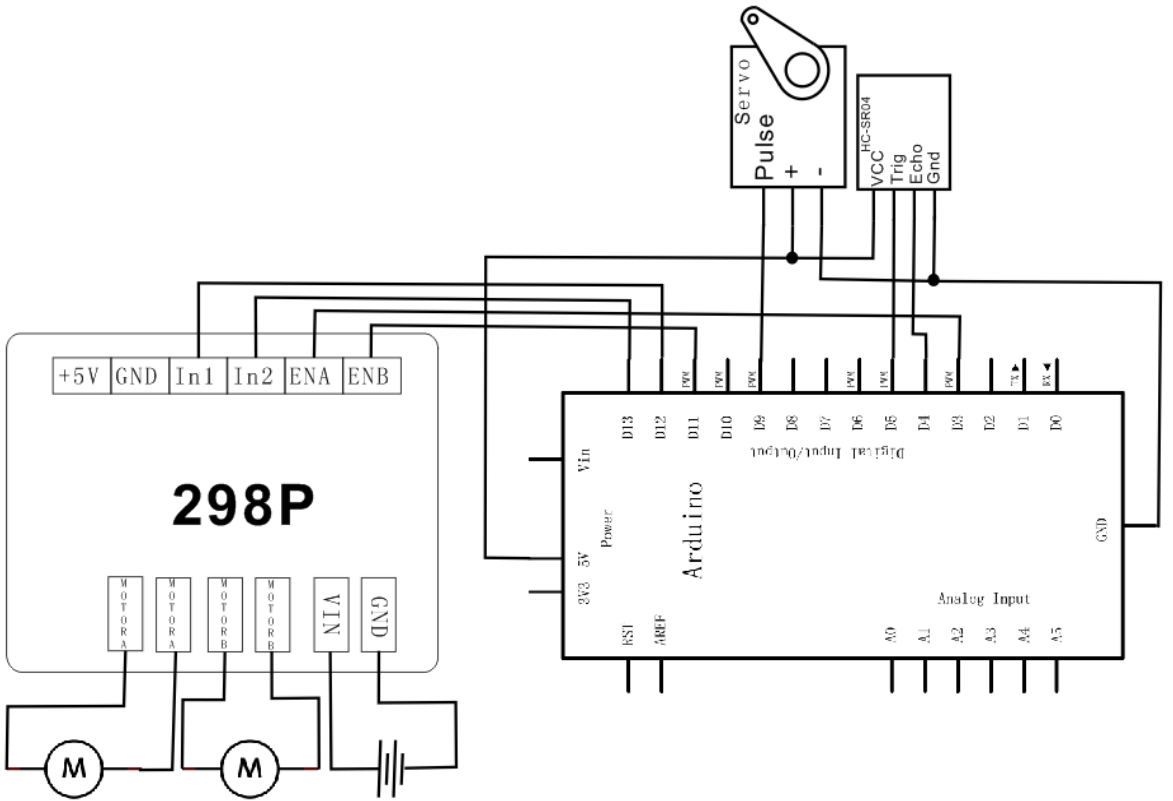
2.Working Principle

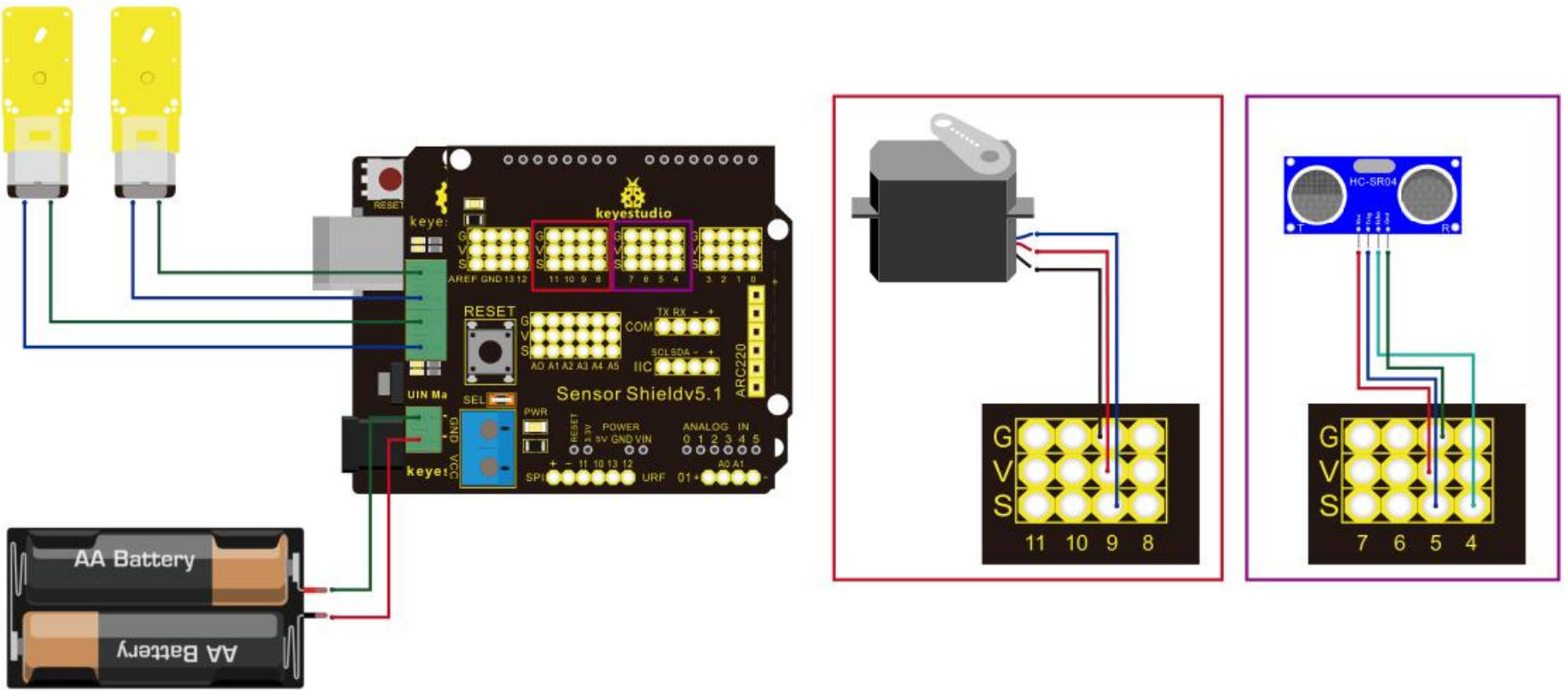
Ultrasonic ranging: the controller sends out a a high level signal of more than 10μs, when the output pin receives the high level signal, the timer will be on; when the signal changes to low level, we can read the time period of the timer, which is the time used for this ultrasonic wave transceiving. Together with its transmission speed, we can calculate the distance.

After we use the ultrasonic sensor to detect the distance from an obstacle, we can control the movement of the Tank according to the data.

If the distance from the obstacle is < 10cm, the Tank moves backward; if the distance is >=25cm, the Tank moves forward; if the distance is <25cm, we control the movement of the servo motors to measure the distance of the left and right. If both the distance are <10cm, the Tank moves backward; if the distance are both >= 10cm, and distance on the left is more than the distance on the right, the Tank moves to the left; if distance on the left is <= the distance on the right, the Tank moves to the right.

3.Schematic and Connection Diagram


Servo
Pulse
+
-
HC-SR04
VCC
Trig
Echo
Gnd
+5V
GND
In1
In2
ENA
ENB
298P
MOTOR A
MOTOR A
MOTOR B
MOTOR B
VIN
GND
M
M
Arduino
Digital Input/Output
Analog Input
Power
Vin
5V
3V3
RST
AREF
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
A0
A1
A2
A3
A4
A5


keyestudio
Sensor Shieldv5.1
RESET
POWER
ANALOG IN
ARC220
AA Battery
AA Battery
HC-SR04
G
V
S
11 10 9 8
7 6 5 4

4.Arduino Code

```
/*
L = Left        R = Right        F = forward       B = backward
*****************************************************************************************************************************
#include <Servo.h>
int pinLB = 12;          // define pin 12
int pinLF = 3;          // define pin 3
int pinRB = 13;        // define pin 13
int pinRF = 11;        // define pin 11
////////////////////////////////
int inputPin = 4;        // define pin for sensor echo
int outputPin =5;        // define pin for sensor trig

int Fspeedd = 0;           // forward speed
int Rspeedd = 0;           // right speed
int Lspeedd = 0;           // left speed
int directionn = 0;      // forward=8 backward=2 left=4 right=6
Servo myservo;               // set myservo
int delay_time = 250; // settling time after steering servo motor moving B
```

```
int Fgo = 8;                // Move F
int Rgo = 6;                // move to the R
int Lgo = 4;                // move to the L
int Bgo = 2;                // move B
void setup()
  {
    Serial.begin(9600);         // Define motor output pin
    pinMode(pinLB,OUTPUT); // pin 12
    pinMode(pinLF,OUTPUT); // pin 3 (PWM)
    pinMode(pinRB,OUTPUT); // pin 13
    pinMode(pinRF,OUTPUT); // pin 11 (PWM)
    pinMode(inputPin, INPUT);        // define input pin for sensor
    pinMode(outputPin, OUTPUT);    // define output pin for sensor
    myservo.attach(9);        // Define servo motor output pin to D9 (PWM)
  }
void advance()          // move forward
       {
       digitalWrite(pinLB,LOW);        // right wheel moves forward
digitalWrite(pinRB, LOW);    // left wheel moves forward
       analogWrite(pinLF,255);
```

```
      analogWrite(pinRF,255);
      }
void stopp()                  // stop
      {
        digitalWrite(pinLB,HIGH);
        digitalWrite(pinRB,HIGH);
        analogWrite(pinLF,0);
        analogWrite(pinRF,0);
      }
void right()                  // turn right (single wheel)
      {
    digitalWrite(pinLB,HIGH);    // wheel on the left moves forward
    digitalWrite(pinRB,LOW); // wheel on the right moves backward
    analogWrite(pinLF, 255);
    analogWrite(pinRF,255);
      }
void left()                   // turn left (single wheel)
      {
    digitalWrite(pinLB,LOW);    // wheel on the left moves backward
    digitalWrite(pinRB,HIGH); // wheel on the right moves forward
```

```
    analogWrite(pinLF, 255);
    analogWrite(pinRF,255);
     }

void back()                  // move backward
     {
      digitalWrite(pinLB,HIGH);    // motor moves to left rear
      digitalWrite(pinRB,HIGH);    // motor moves to right rear
      analogWrite(pinLF,255);
      analogWrite(pinRF,255);
     }
void detection()             // measure 3 angles (0.90.179)
     {
       int delay_time = 250;      // stabilizing time for servo motor after moving backward
       ask_pin_F();                 // read the distance ahead
      if(Fspeedd < 10)              // if distance ahead is <10cm
       {
       stopp();                     // clear data
       delay(100);
       back();                      // move backward for 0.2S
```

```
delay(200);
}
if(Fspeedd < 25)              // if distance ahead is <25cm
{
   stopp();
   delay(100);                  // clear data
   ask_pin_L();                 // read distance on the left
   delay(delay_time);          // stabilizing time for servo motor
   ask_pin_R();                 // read distance on the right
   delay(delay_time);          // stabilizing time for servo motor

   if(Lspeedd > Rspeedd)      // if distance on the left is >distance on the right
   {
      directionn = Lgo;          // move to the L
   }

   if(Lspeedd <= Rspeedd)      // if distance on the left is <= distance on the right
   {
     directionn = Rgo;          // move to the right
   }
```

```
            if (Lspeedd < 10 && Rspeedd < 10)      // if distance on left and right are both <10cm
           {
            directionn = Bgo;           // move backward
           }}
        else                                  // if distance ahead is >25cm
        {
           directionn = Fgo;              // move forward
        }}
void ask_pin_F()      // measure the distance ahead
      {
        myservo.write(90);
        digitalWrite(outputPin, LOW);      // ultrasonic sensor transmit low level signal 2μs
        delayMicroseconds(2);
        digitalWrite(outputPin, HIGH);    // ultrasonic sensor transmit high level signal10μs, at least 10μs
        delayMicroseconds(10);
        digitalWrite(outputPin, LOW);       // keep transmitting low level signal
        float Fdistance = pulseIn(inputPin, HIGH);    // read the time in between
        Fdistance= Fdistance/5.8/10;             // convert time into distance (unit: cm)
        Fspeedd = Fdistance;                        // read the distance into Fspeedd
      }
```

```
  void ask_pin_L()      // measure distance on the left
      {
         myservo.write(5);
         delay(delay_time);
         digitalWrite(outputPin, LOW);      // ultrasonic sensor transmit low level signal 2μs
         delayMicroseconds(2);
         digitalWrite(outputPin, HIGH);    // ultrasonic sensor transmit high level signal10μs, at least 10μs
         delayMicroseconds(10);
         digitalWrite(outputPin, LOW);        // keep transmitting low level signal
         float Ldistance = pulseIn(inputPin, HIGH);    // read the time in between
         Ldistance= Ldistance/5.8/10;              // convert time into distance (unit: cm)
         Lspeedd = Ldistance;                          // read the distance into Lspeedd
      }
void ask_pin_R()      //    measure distance on the right
      {
         myservo.write(177);
         delay(delay_time);
         digitalWrite(outputPin, LOW);      // ultrasonic sensor transmit low level signal 2μs
         delayMicroseconds(2);
         digitalWrite(outputPin, HIGH);    // ultrasonic sensor transmit high level signal10μs, at least 10μs
```

```
    delayMicroseconds(10);
    digitalWrite(outputPin, LOW);       // keep transmitting low level signal
    float Rdistance = pulseIn(inputPin, HIGH);   // read the time in between
    Rdistance= Rdistance/5.8/10;            // convert time into distance (unit: cm)
    Rspeedd = Rdistance;                     // read the distance into Rspeedd
  }
  void loop()
  {
  myservo.write(90);   // home set the servo motor, ready for next measurement
  detection();             // measure the angle and determine which direction to move
if(directionn == 2)   // if directionn= 2
{
   back();
   delay(800);                              //   go backward
   left() ;
   delay(200);                      // Move slightly to the left (to prevent stuck in dead end)
}
if(directionn == 6)                  // if directionn = 6
{
   back();
```

```
    delay(100);
    right();
    delay(600);                    // turn right
  }
  if(directionn == 4)              // if directionn = 4
  {
    back();
    delay(600);
    left();
    delay(600);                    // turn left
  }
  if(directionn == 8)              // if directionn = 8
  {
   advance();                      // move forward
   delay(100);
  } }
```

****

5. Result

After power-on, the car runs and will avoid obstacle automatically when catching obstacle.

Project 4: Bluetooth Control Tank Robot

1.Introduction

This project is a tank robot system based on Bluetooth communication, including software and hardware design. The controller part is a UNO board. A Bluetooth module is used to receive the Bluetooth signal from the cellphone and feedback the signal to the UNO. UNO will analyze the signal to determine and control the motors movement to adjust car moving direction. Therefore the tank robot can be controlled by cellphone.

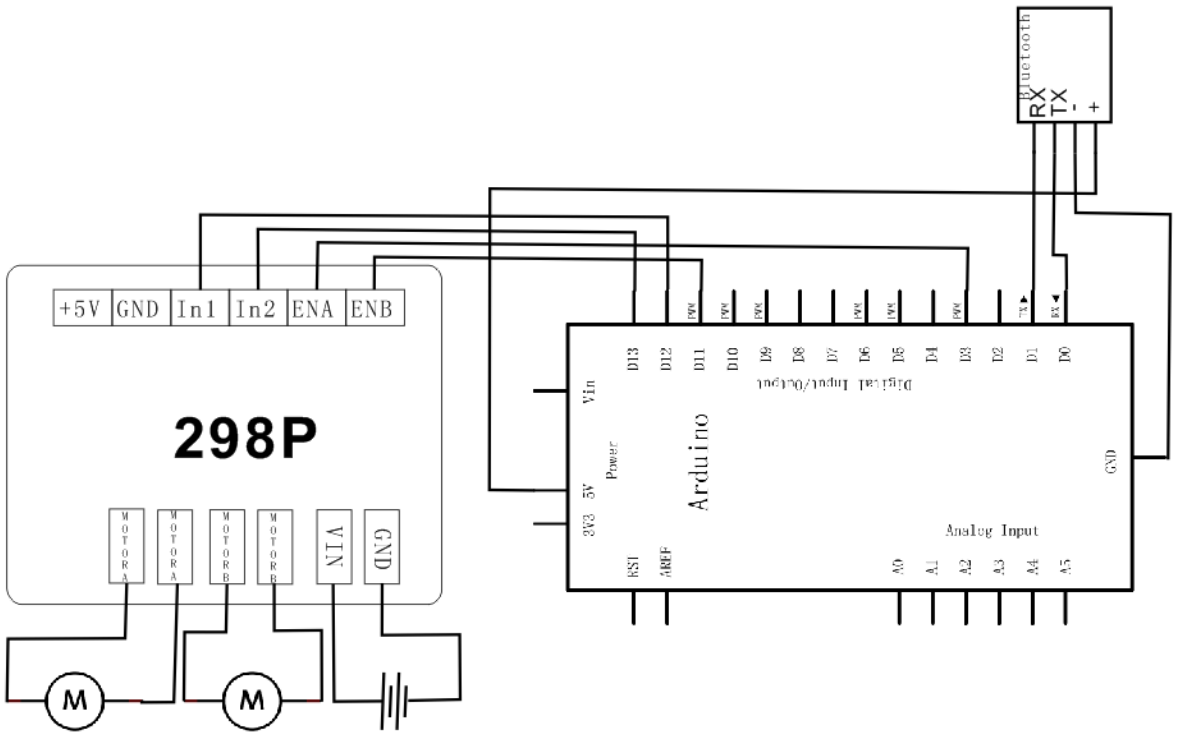
2.Working Principle

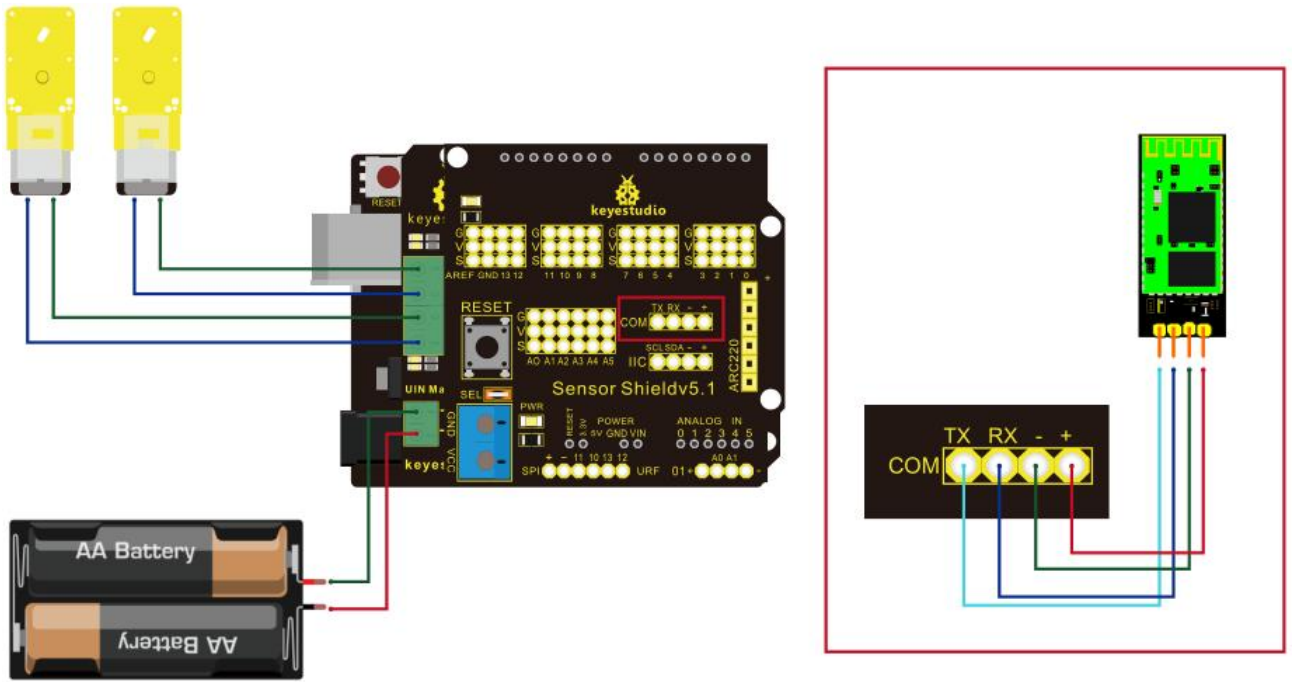
1. The Bluetooth module is connected to UNO; the module communicates with cell phone through a Bluetooth APP.
2. The Bluetooth APP on the cell phone will pass information of “U”“D”“L”“R”“S” to the Bluetooth module.
3. The Bluetooth module will pass the information to the UNO, so the UNO can determine car movement according to the information received.
4. When the UNO receives a “U”, the car goes straight forward; when it receives a “D”, the car goes backward; “L” for turning left; “R” for turning right; and “S” for stop.

3.Bluetooth Usage

Connect main board +5V to Bluetooth VCC, GND to Bluetooth GND, TX to Bluetooth RX and RX to Bluetooth TX.
Remember to open the Bluetooth on your phone; when you open the Bluetooth APP, it will remind you.Pair up Bluetooth device on your phone, search and pair.
Pair up device, PIN No. is 1234.
Open Bluetooth APP and pair up Bluetooth device. After it’s paired, the Bluetooth module can communicate with cell phone.

4.Schematic and Connection Diagram


Bluetooth
RX
TX
-
+
+5V
GND
In1
In2
ENA
ENB
298P
MOTORA
MOTORA
MOTORB
MOTORB
VIN
GND
M
M
Arduino
Power
Digital Input/Output
Analog Input
Vin
5V
3V3
RST
AREF
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
A0
A1
A2
A3
A4
A5



5. Arduino Code for Bluetooth Control Tank Robot

/*

```
      L = left,      R = right ,    F = forward,      B = backward.
************************************************************************************************************************************
int pinLB = 12;          // define pin 12
int pinLF = 3;          // define pin 3
int pinRB = 13;        // define pin 13
int pinRF = 11;        // define pin 11
int val;
void setup()
  {
    Serial.begin(9600);          // define pin for motor output
    pinMode(pinLB,OUTPUT); // pin 12
    pinMode(pinLF,OUTPUT); // pin 3 (PWM)
    pinMode(pinRB,OUTPUT); // pin 13
    pinMode(pinRF,OUTPUT); // pin 11 (PWM)
  }
void advance()          // move forward
       {
       digitalWrite(pinLB,LOW);        // right wheel moves forward
       digitalWrite(pinRB, LOW);    // left wheel moves forward
       analogWrite(pinLF,255);
```

```
      analogWrite(pinRF,255);
      }
void stopp()                  // stop
      {
        digitalWrite(pinLB,HIGH);
        digitalWrite(pinRB,HIGH);
        analogWrite(pinLF,0);
        analogWrite(pinRF,0);
      }
void right()                  // turn right (single wheel)
      {
    digitalWrite(pinLB,HIGH);    // left wheel moves forward
    digitalWrite(pinRB,LOW); // right wheel moves backward
    analogWrite(pinLF, 255);
    analogWrite(pinRF,255);
      }
void left()                   // turn left (single wheel)
      {
    digitalWrite(pinLB,LOW);    // left wheel moves forward
    digitalWrite(pinRB,HIGH); // right wheel moves backward
```

```
    analogWrite(pinLF, 255);
    analogWrite(pinRF,255);
      }
  void back()                  // move backward
      {digitalWrite(pinLB,HIGH);    // motor moves to left rear
       digitalWrite(pinRB,HIGH);    // motor moves to right rear
       analogWrite(pinLF,255);
       analogWrite(pinRF,255);
      }
void loop()
      { val=Serial.read();
      if(val=='U')advance();
      if(val=='D')back();
      if(val=='L')left() ;
      if(val=='R')right();
if(val=='S')stopp();
}
```

****

6. Result

Connected to Bluetooth APP, the movement of the car is controlled by the APP.

Project 5: Ultrasonic Ranging Tank Robot

1.Introduction

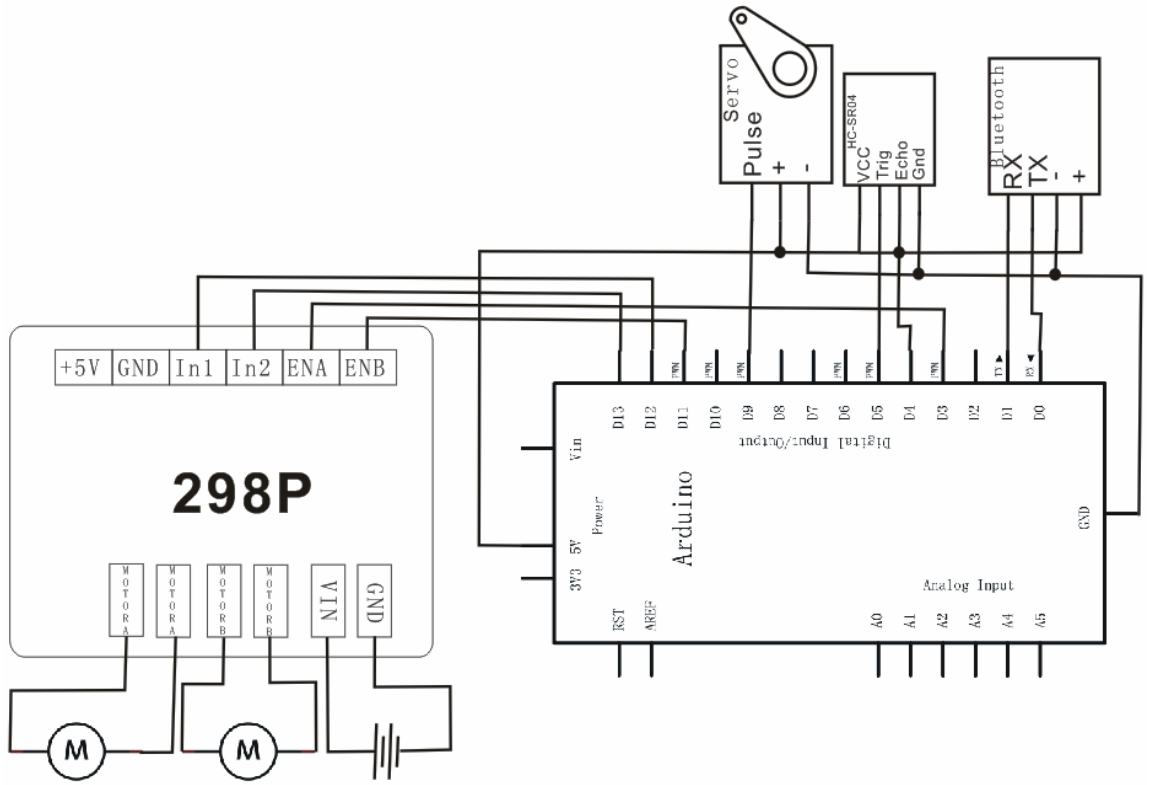
In project 3, we use the ultrasonic sensor module for the tank to realize obstacle avoidance function. In project 4, we use a HC-06 Bluetooth module for the tank, so the tank can be controlled form a cellphone terminal.

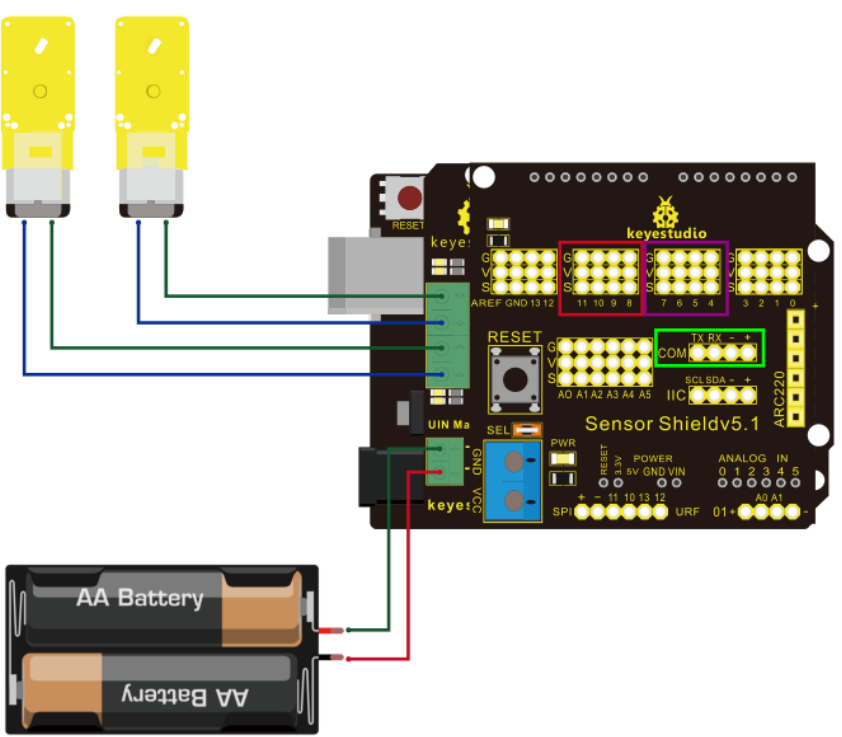
This project is based on project 3 and project 4. Here, we apply echo sounding method for the ranging. The trig end of ultrasonic sensor will launch ultrasonic wave in a specific direction. At the same time, the timer begins to count; when the ultrasonic wave encounters an obstacle, it's reflected back; when the echo end receives the signal, the timer stops the count. With the traveling speed of the wave and the traveling time, we can calculate the distance between the launching point and the obstacle.

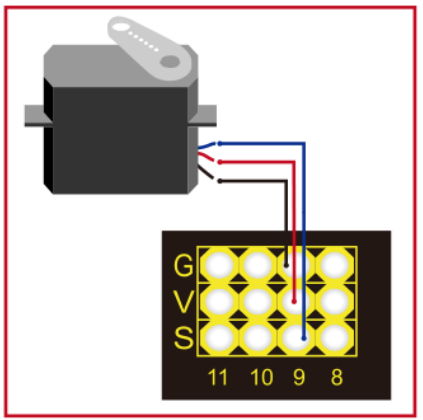
Also,we will use the HC-06 Bluetooth module for the Tank to communicate with the terminal. The tank will send the measured distance to the terminal, so we can see clearly the distance between the tank and the obstacle while the tank is on.

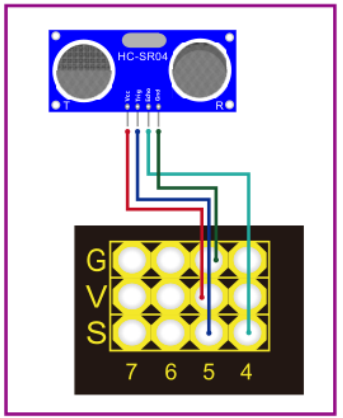
This robot can be used in outdoor exploration or terrain exploration in a confined space.

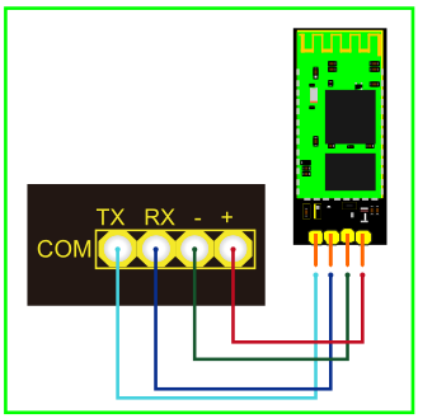
2.Schematic and Connection Diagram


Servo
Pulse
+
-
HC-SR04
VCC
Trig
Echo
Gnd
Bluetooth
RX
TX
-
+
+5V
GND
In1
In2
ENA
ENB
298P
MOTORA
MOTORA
MOTORB
MOTORB
VIN
GND
M
M
Arduino
Digital Input/Output
Analog Input
Power
Vin
5V
3V3
RST
AREF
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
A0
A1
A2
A3
A4
A5


keyestudio
Sensor Shieldv5.1
RESET
POWER
ANALOG IN
AA Battery


G
V
S
11 10 9 8


HC-SR04
G
V
S
7 6 5 4


TX RX - +
COM

3.Arduino Code for Ultrasonic Ranging Tank Robot

```
/*
  L = Left,             R = Right ,             F = forward ,          B = backward
*********************************************************************************************************************************
#include <Servo.h>
int pinLB = 12;          // define pin 12
int pinLF = 3;          // define pin 3
int pinRB = 13;         // define pin 13
int pinRF = 11;         // define pin 11
int inputPin = 4;       // define pin for sensor echo
int outputPin =5;       // define pin for sensor trig
int Fspeedd = 0;           // forward speed
int Rspeedd = 0;           // right speed
int Lspeedd = 0;           // left speed
int directionn = 0;      // forward=8 backward=2 left=4 right=6
Servo myservo;               // set myservo
int delay_time = 250; // settling time after steering servo motor moving B
int Fgo = 8;                // Move F
int Rgo = 6;                // move to the R
```

```
int Lgo = 4;                // move to the L
int Bgo = 2;                // move B
void setup()
  {
    Serial.begin(9600);         // Define motor output pin
    pinMode(pinLB,OUTPUT); // pin 12
    pinMode(pinLF,OUTPUT); // pin 3 (PWM)
    pinMode(pinRB,OUTPUT); // pin 13
    pinMode(pinRF,OUTPUT); // pin 11 (PWM)
    pinMode(inputPin, INPUT);       // define input pin for sensor
    pinMode(outputPin, OUTPUT);    // define output pin for sensor
    myservo.attach(9);       // Define servo motor output pin to D9 (PWM)
  }
void advance()          // move forward
       {
       digitalWrite(pinLB,LOW);       // right wheel moves forward
       digitalWrite(pinRB, LOW);    // left wheel moves forward
       analogWrite(pinLF,255);
       analogWrite(pinRF,255);
       }
```

```
void stopp()                  // stop
      {
        digitalWrite(pinLB,HIGH);
        digitalWrite(pinRB,HIGH);
        analogWrite(pinLF,0);
        analogWrite(pinRF,0);
      }
void right()                  // turn right (single wheel)
      {
    digitalWrite(pinLB,HIGH);    // wheel on the left moves forward
    digitalWrite(pinRB,LOW); // wheel on the right moves backward
    analogWrite(pinLF, 255);
    analogWrite(pinRF,255);
      }
void left()                   // turn left (single wheel)
      {
    digitalWrite(pinLB,LOW);    // wheel on the left moves backward
    digitalWrite(pinRB,HIGH); // wheel on the right moves forward
    analogWrite(pinLF, 255);
    analogWrite(pinRF,255);
```

```
    }

void back()                 // move backward
    {
     digitalWrite(pinLB,HIGH);    // motor moves to left rear
     digitalWrite(pinRB,HIGH);    // motor moves to right rear
     analogWrite(pinLF,255);
     analogWrite(pinRF,255);
    }
void detection()            // measure 3 angles (0.90.179)
    {
      int delay_time = 250;      // stabilizing time for servo motor after moving backward
      ask_pin_F();                 // read the distance ahead
     if(Fspeedd < 10)             // if distance ahead is <10cm
      {
      stopp();                    // clear data
      delay(100);
      back();                      // move backward for 0.2S
      delay(200);
      }
```

```
if(Fspeedd < 25)                // if distance ahead is <25cm
{
   stopp();
   delay(100);                   // clear data
   ask_pin_L();                  // read distance on the left
   delay(delay_time);           // stabilizing time for servo motor
   ask_pin_R();                  // read distance on the right
   delay(delay_time);           // stabilizing time for servo motor

   if(Lspeedd > Rspeedd)      // if distance on the left is >distance on the right
   {
      directionn = Lgo;          // move to the L
   }
    if(Lspeedd <= Rspeedd)      // if distance on the left is <= distance on the right
   {
    directionn = Rgo;           // move to the right
   }
   if (Lspeedd < 10 && Rspeedd < 10)      // if distance on left and right are both <10cm
   {
    directionn = Bgo;           // move backward
```

```
        } }
      else                                  // if distance ahead is >25cm
      {
        directionn = Fgo;               // move forward
      }}
void ask_pin_F()      // measure the distance ahead
    {
      myservo.write(90);
      digitalWrite(outputPin, LOW);      // ultrasonic sensor transmit low level signal 2μs
      delayMicroseconds(2);
      digitalWrite(outputPin, HIGH);    // ultrasonic sensor transmit high level signal10μs, at least 10μs
      delayMicroseconds(10);
      digitalWrite(outputPin, LOW);        // keep transmitting low level signal
      float Fdistance = pulseIn(inputPin, HIGH);    // read the time in between
      Fdistance= Fdistance/5.8/10;             // convert time into distance (unit: cm)
      Fspeedd = Fdistance;                        // read the distance into Fspeedd
  Serial.print("Fspeedd = ");
  Serial.print(Fspeedd );
  Serial.println("    cm");
    }
```

```
  void ask_pin_L()      // measure distance on the left
      {
         myservo.write(5);
         delay(delay_time);
         digitalWrite(outputPin, LOW);      // ultrasonic sensor transmit low level signal 2μs
         delayMicroseconds(2);
         digitalWrite(outputPin, HIGH);    // ultrasonic sensor transmit high level signal10μs, at least 10μs
         delayMicroseconds(10);
         digitalWrite(outputPin, LOW);        // keep transmitting low level signal
         float Ldistance = pulseIn(inputPin, HIGH);    // read the time in between
         Ldistance= Ldistance/5.8/10;              // convert time into distance (unit: cm)
         Lspeedd = Ldistance;                          // read the distance into Lspeedd
    Serial.print("Lspeedd = ");
    Serial.print(Lspeedd );
         Serial.print("    cm    ");
      }
void ask_pin_R()      //    measure distance on the right
      {
         myservo.write(177);
         delay(delay_time);
```

```
        digitalWrite(outputPin, LOW);      // ultrasonic sensor transmit low level signal 2μs
        delayMicroseconds(2);
        digitalWrite(outputPin, HIGH);    // ultrasonic sensor transmit high level signal10μs, at least 10μs
        delayMicroseconds(10);
        digitalWrite(outputPin, LOW);       // keep transmitting low level signal
        float Rdistance = pulseIn(inputPin, HIGH);    // read the time in between
        Rdistance= Rdistance/5.8/10;             // convert time into distance (unit: cm)
        Rspeedd = Rdistance;                        // read the distance into Rspeedd
   Serial.print(" Rspeedd = ");
   Serial.print(Rspeedd );
        Serial.println("    cm");
      }
void loop()
   {
     myservo.write(90);    // home set the servo motor, ready for next measurement
     detection();               // measure the angle and determine which direction to move
    if(directionn == 2)    // if directionn= 2
    {
      back();
      delay(800);                                //    go backward
```

```
    left() ;
    delay(200);                    // Move slightly to the left (to prevent stuck in dead end)
  }
  if(directionn == 6)              // if directionn = 6
  { back();
    delay(100);
    right();
    delay(600);                    // turn right
  }
  if(directionn == 4)              // if directionn = 4
  { back();
    delay(600);
    left();
    delay(600);                    // turn left
  }
  if(directionn == 8)              // if directionn = 8
  { advance();          // move forward
   delay(100);
  }}
```

****

4. Result

After connection and power-on, the car is connected to Bluetooth module. Open and connect to APP, and the car will go and avoid obstacle automatically. We can see the data of distance between the car and obstacle in APP.