Basic Kit for Turtle 2WD SKU:ROB0118

Function Introduction

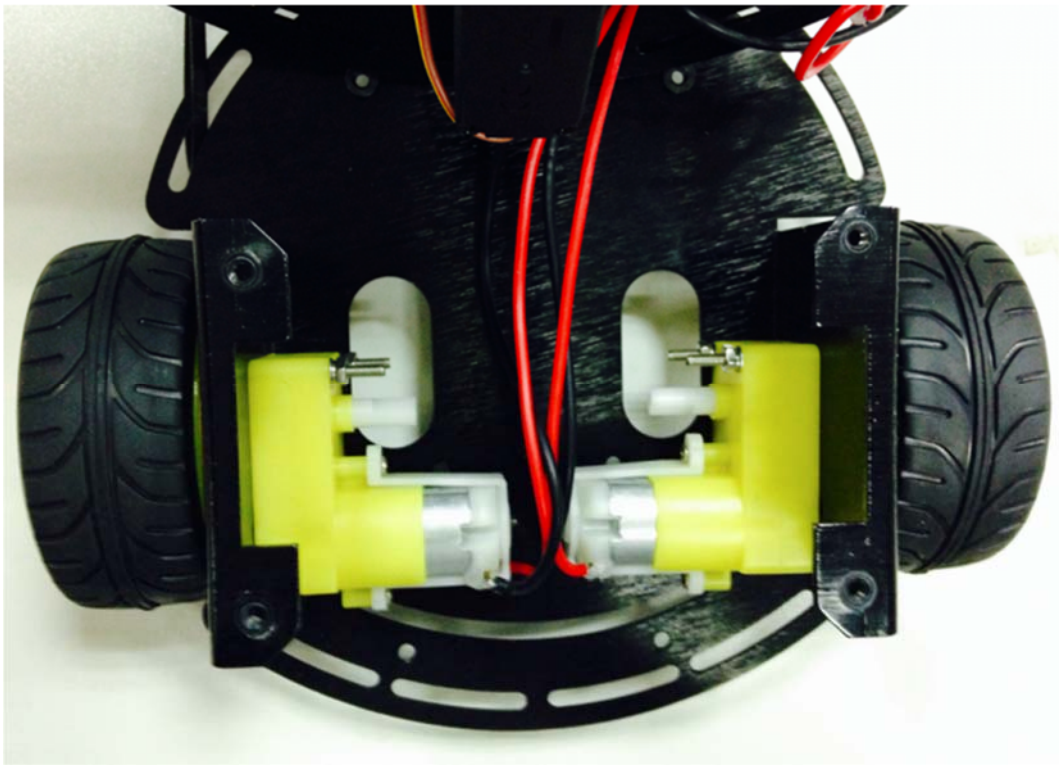
This Kit will teach you how to build a automatic obstacle - avoidance robot which is achieved on the platform of the Turtle Robot,based on ultrasonic sensor as distance measuring device,and combined with servo.

STEP 1: Assemble Robot

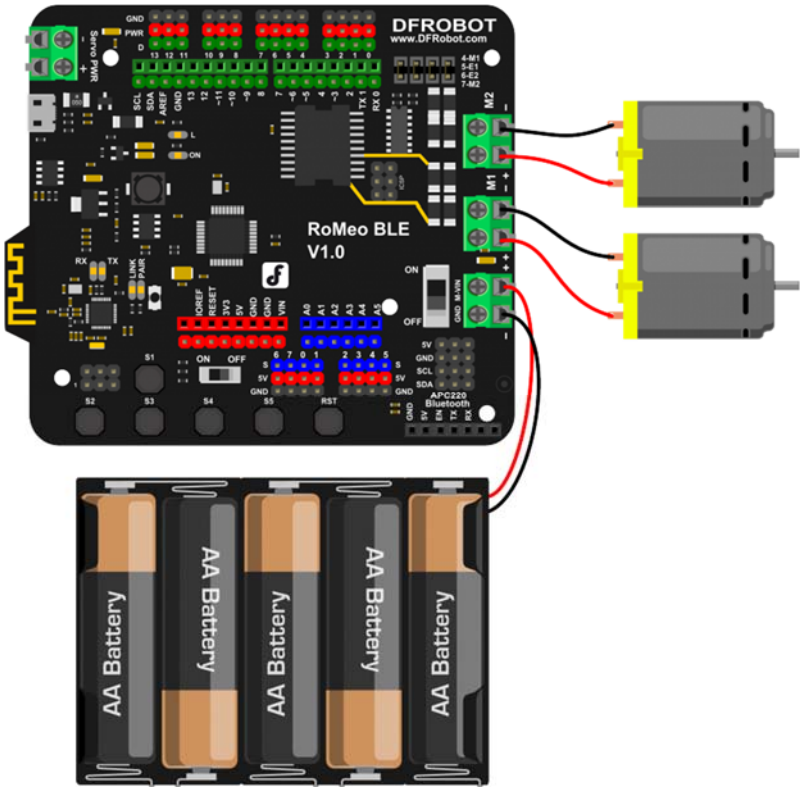
Refer to Instruction Manual
http://www.dfrobot.com.cn/image/data/ROB0005/CN/ROB0005%20InstructionManual%20V1.1.pdf

Precautions:

1. Instruction Manual says nothing about how to deal with motor wire. The next pictrue tells it.



2. Finished the robot chassis.Then following the connection diagram to wire the hardware.



Motor Connection

Physical map

STEP 2: Debug Motor

Download the Code

```
int speedPin_M1 = 5;     //M1 Speed Control
int speedPin_M2 = 6;     //M2 Speed Control
int directionPin_M1 = 4;     //M1 Direction Control
int directionPin_M2 = 7;     //M1 Direction Control

void setup(){

}

void loop(){
    carAdvance(100,100);
    delay(1000);
    carBack(100,100);
    delay(1000);
    carTurnLeft(250,250);
    delay(1000);
    carTurnRight(250,250);
    delay(1000);
}

void carStop(){                 //  Motor Stop
  digitalWrite(speedPin_M2,0);
  digitalWrite(directionPin_M1,LOW);
  digitalWrite(speedPin_M1,0);
  digitalWrite(directionPin_M2,LOW);
}

void carAdvance(int leftSpeed,int rightSpeed){         //Move forward
  analogWrite (speedPin_M2,leftSpeed);              //PWM Speed Control
```

```
  digitalWrite(directionPin_M1,HIGH);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,HIGH);
}

void carBack(int leftSpeed,int rightSpeed){       //Move backward
  analogWrite (speedPin_M2,leftSpeed);
  digitalWrite(directionPin_M1,LOW);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,LOW);
}

void carTurnRight(int leftSpeed,int rightSpeed){           //Turn Right
  analogWrite (speedPin_M2,leftSpeed);
  digitalWrite(directionPin_M1,LOW);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,HIGH);
}
void carTurnLeft(int leftSpeed,int rightSpeed){          //Turn Left
  analogWrite (speedPin_M2,leftSpeed);
  digitalWrite(directionPin_M1,HIGH);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,LOW);
}
```

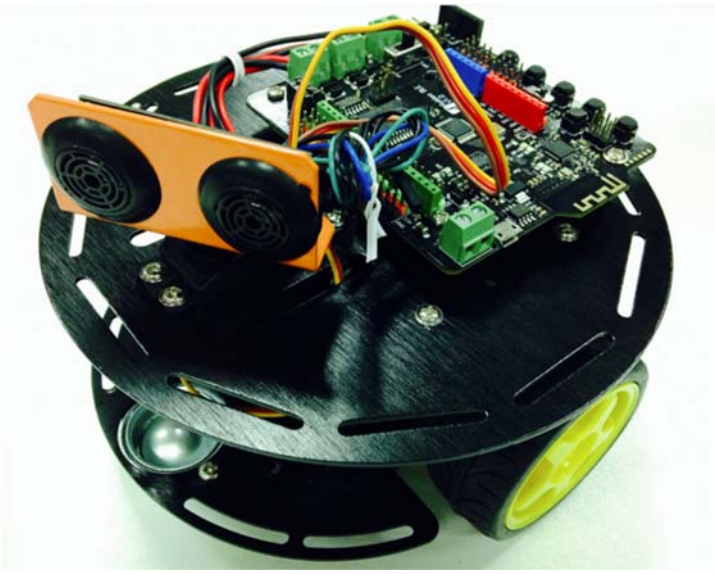
STEP 3：Install Upper Plate

1. Prepare the Materials

2. Fixed Ultrasonic Sensor Position

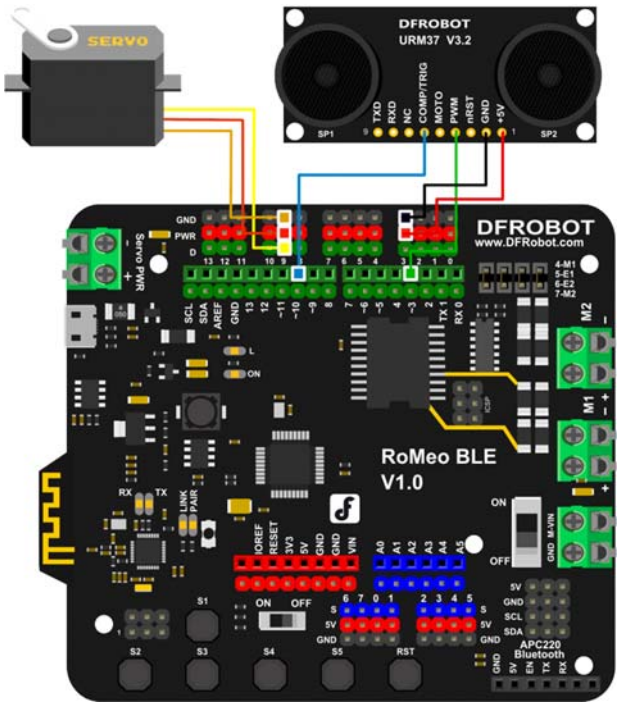
Please see the Installation Manual http://www.dfrobot.com.cn/images/upload/File/20141030183325g7lofm.pdf

3. Fixed Servo Position

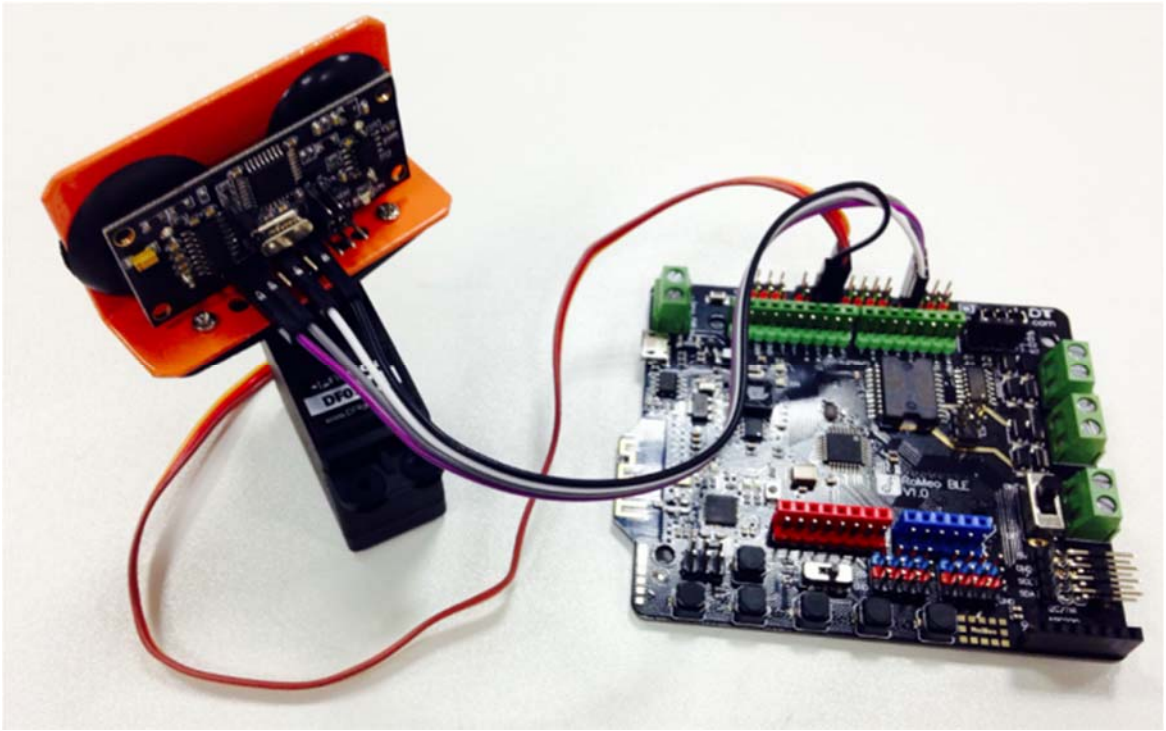


STEP4: Debug Ultrasonic Sensor and Servo

1. Hardware Connection



Ultrasonic Sensor and Servo Controll



2. Download Code

Install the library firstly. Metro libray http://www.dfrobot.com.cn/images/upload/File/20141031110246wu4065.rar

```
#include <Servo.h>
#include <Metro.h>
Metro measureDistance = Metro(50);
Metro sweepServo = Metro(20);

unsigned long actualDistance = 0;

Servo myservo;  // create servo object to control a servo
int pos = 60;
int sweepFlag = 1;



int URPWM = 3; // PWM Output 0—25000US，Every 50US represent 1cm
int URTRIG= 10; // PWM trigger pin
uint8_t EnPwmCmd[4]={0x44,0x02,0xbb,0x01};    // distance measure command

void setup(){                                 // Serial initialization
  myservo.attach(9);
  Serial.begin(9600);                         // Sets the baud rate to 9600
  SensorSetup();
}

void loop(){
 if(measureDistance.check() == 1){
      actualDistance = MeasureDistance();
//      Serial.println(actualDistance);
//      delay(100);
 }

 if(sweepServo.check() == 1){
      servoSweep();
```

```
 }


}


void SensorSetup(){
  pinMode(URTRIG,OUTPUT);                     // A low pull on pin COMP/TRIG
  digitalWrite(URTRIG,HIGH);                  // Set to HIGH
  pinMode(URPWM, INPUT);                      // Sending Enable PWM mode comm
and
  for(int i=0;i<4;i++){
      Serial.write(EnPwmCmd[i]);
   }
}


int MeasureDistance(){        // a low pull on pin COMP/TRIG  triggering a se
nsor reading
    digitalWrite(URTRIG, LOW);
    digitalWrite(URTRIG, HIGH);               // reading Pin PWM will output
pulses
    unsigned long distance=pulseIn(URPWM,LOW);
    if(distance==50000){              // the reading is invalid.
      Serial.print("Invalid");
    }else{
      distance=distance/50;           // every 50us low level stands for 1cm
    }
    return distance;
}


void servoSweep(){
  if(sweepFlag ){
     if(pos>=60 && pos<=120){
        pos=pos+1;                                  // in steps of 1 degree
        myservo.write(pos);                         // tell servo to go to po
sition in variable 'pos'
    }
```

```
      if(pos>119)  sweepFlag = false;                          // assign the var
iable again
  }else {
      if(pos>=60 && pos<=120){
        pos=pos-1;
        myservo.write(pos);
      }
      if(pos<61)  sweepFlag = true;
   }
}
```

STEP 5: Debugging Robot

```
#include <Servo.h>
#include <Metro.h>
Metro measureDistance = Metro(50);
Metro sweepServo = Metro(20);

int speedPin_M1 = 5;     //M1 Speed Control
int speedPin_M2 = 6;     //M2 Speed Control
int directionPin_M1 = 4;     //M1 Direction Control
int directionPin_M2 = 7;     //M1 Direction Control
unsigned long actualDistance = 0;

Servo myservo;  // create servo object to control a servo
int pos = 60;
int sweepFlag = 1;

int URPWM = 3; // PWM Output 0—25000US，Every 50US represent 1cm
int URTRIG= 10; // PWM trigger pin
uint8_t EnPwmCmd[4]={0x44,0x02,0xbb,0x01};    // distance measure command

void setup(){                                  // Serial initialization
  myservo.attach(9);
```

```
  Serial.begin(9600);                           // Sets the baud rate to 9600
  SensorSetup();
}

void loop(){

 if(measureDistance.check() == 1){
      actualDistance = MeasureDistance();
//      Serial.println(actualDistance);
//      delay(100);
 }

 if(sweepServo.check() == 1){
      servoSweep();
 }

 if(actualDistance <= 30){
            myservo.write(90);
            if(pos>=90){
//                  carBack(100,100);
////                  Serial.println("carBack");
//                  delay(100);
                  carTurnRight(150,150);
//                  Serial.println("carTurnRight");
                  delay(100);
               }else{
//                   carBack(100,100);
////                   Serial.println("carBack");
//                   delay(100);
                   carTurnLeft(150,150);
//                   Serial.println("carTurnLeft");
                   delay(100);
               }
    }else{
```

```
                    carAdvance(70,70);
//                    Serial.println("carAdvance");
                    delay(100);
     }
//        carBack(150,150);
}



void SensorSetup(){
  pinMode(URTRIG,OUTPUT);                     // A low pull on pin COMP/TRIG
  digitalWrite(URTRIG,HIGH);                  // Set to HIGH
  pinMode(URPWM, INPUT);                      // Sending Enable PWM mode comm
and
  for(int i=0;i<4;i++){
      Serial.write(EnPwmCmd[i]);
   }
}



int MeasureDistance(){  // a low pull on pin COMP/TRIG  triggering a sensor r
eading
    digitalWrite(URTRIG, LOW);
    digitalWrite(URTRIG, HIGH);               // reading Pin PWM will output
pulses
    unsigned long distance=pulseIn(URPWM,LOW);
    if(distance==50000){              // the reading is invalid.
      Serial.print("Invalid");
    }else{
      distance=distance/50;           // every 50us low level stands for 1cm
    }
    return distance;
}


void carStop(){                 //  Motor Stop
  digitalWrite(speedPin_M2,0);
  digitalWrite(directionPin_M1,LOW);
```

```
  digitalWrite(speedPin_M1,0);
  digitalWrite(directionPin_M2,LOW);
}

void carAdvance(int leftSpeed,int rightSpeed){         //Move forward
  analogWrite (speedPin_M2,leftSpeed);              //PWM Speed Control
  digitalWrite(directionPin_M1,HIGH);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,HIGH);
}

void carBack(int leftSpeed,int rightSpeed){       //Move backward
  analogWrite (speedPin_M2,leftSpeed);
  digitalWrite(directionPin_M1,LOW);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,LOW);
}

void carTurnRight(int leftSpeed,int rightSpeed){           //Turn Right
  analogWrite (speedPin_M2,leftSpeed);
  digitalWrite(directionPin_M1,LOW);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,HIGH);
}

void carTurnLeft(int leftSpeed,int rightSpeed){          //Turn Left
  analogWrite (speedPin_M2,leftSpeed);
  digitalWrite(directionPin_M1,HIGH);
  analogWrite (speedPin_M1,rightSpeed);
  digitalWrite(directionPin_M2,LOW);
}
void servoSweep(){
  if(sweepFlag){
     if(pos>=60 && pos<=120){
```

```
        pos=pos+1;                                  // in steps of 1 degree
        myservo.write(pos);                         // tell servo to go to po
sition in variable 'pos'
    }
      if(pos>119)  sweepFlag = false;                       // assign the var
iable again
  }
   else {
      if(pos>=60 && pos<=120){
        pos=pos-1;
        myservo.write(pos);
      }
      if(pos<61)  sweepFlag = true;
    }
}
```

Your own car was born!