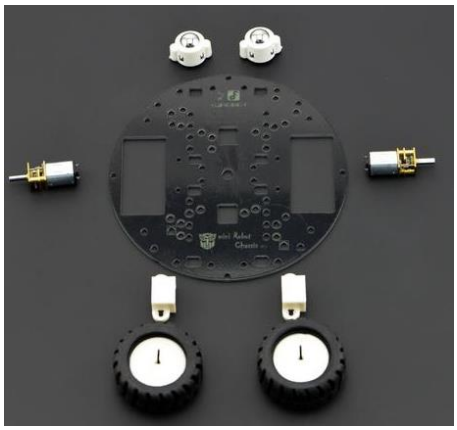
The Application for Car Obstacle Avoidance with TFmini and Servo

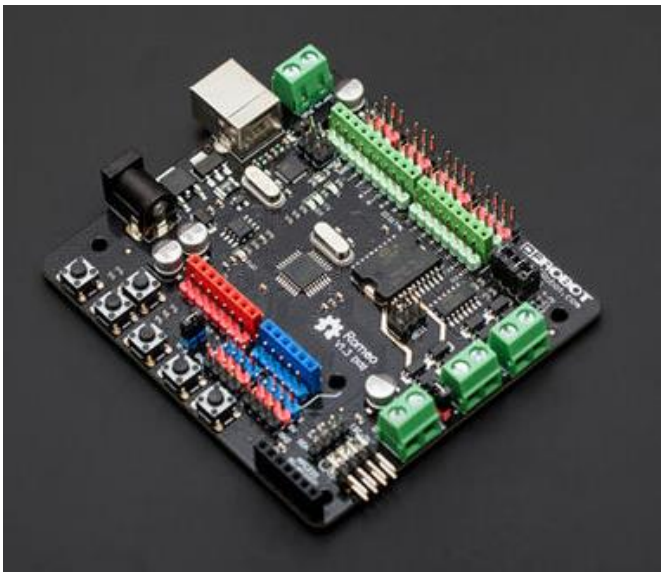
1. Experimental equipment and wiring

1.1 Experimental equipment

- MiniQ Desktop Robot Chassis

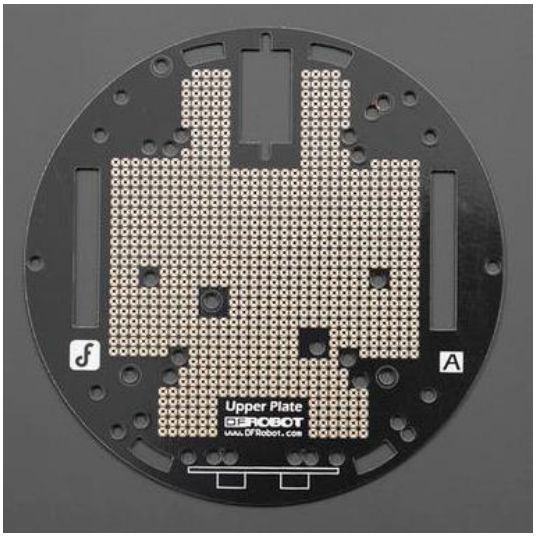


  - Chassis Diameter： 122mm
  - Wheel Diameter： 42mm
  - Chassis Height： 15mm
  - Compatible with Arduino Standard Board and Romeo Controller Fixed Hole
  - Motor parameters：
    - N20 Motor Voltage： 3-9V
    - Load-free speed： 13000rpm
    - 50： 1 Reducer260rpm@6V
    - 40mA@6V
    - 360mA @6V
    - 10O ounce inch torque@6V
- Romeo Trinity Arduino Compatible Controller



- Using Atmel Atmega328 MCU
- Arduino UNO bootloader
- Fully compatible port layout for Aruduino UNO
- Integrated APC220 wireless data transmission and DF-Bluetooth V3 (SKU: TEL0026) Bluetooth module interface
- Five I2C Bus Interfaces Supported
- Supporting two-way motor drive, peak current 2A, 4 control ports using jumper switching
- External input voltage range： 6V~20V
- Detailed description of the parameters can be found in the appendix's web address.

- MiniQ Car Upper Installation Plate

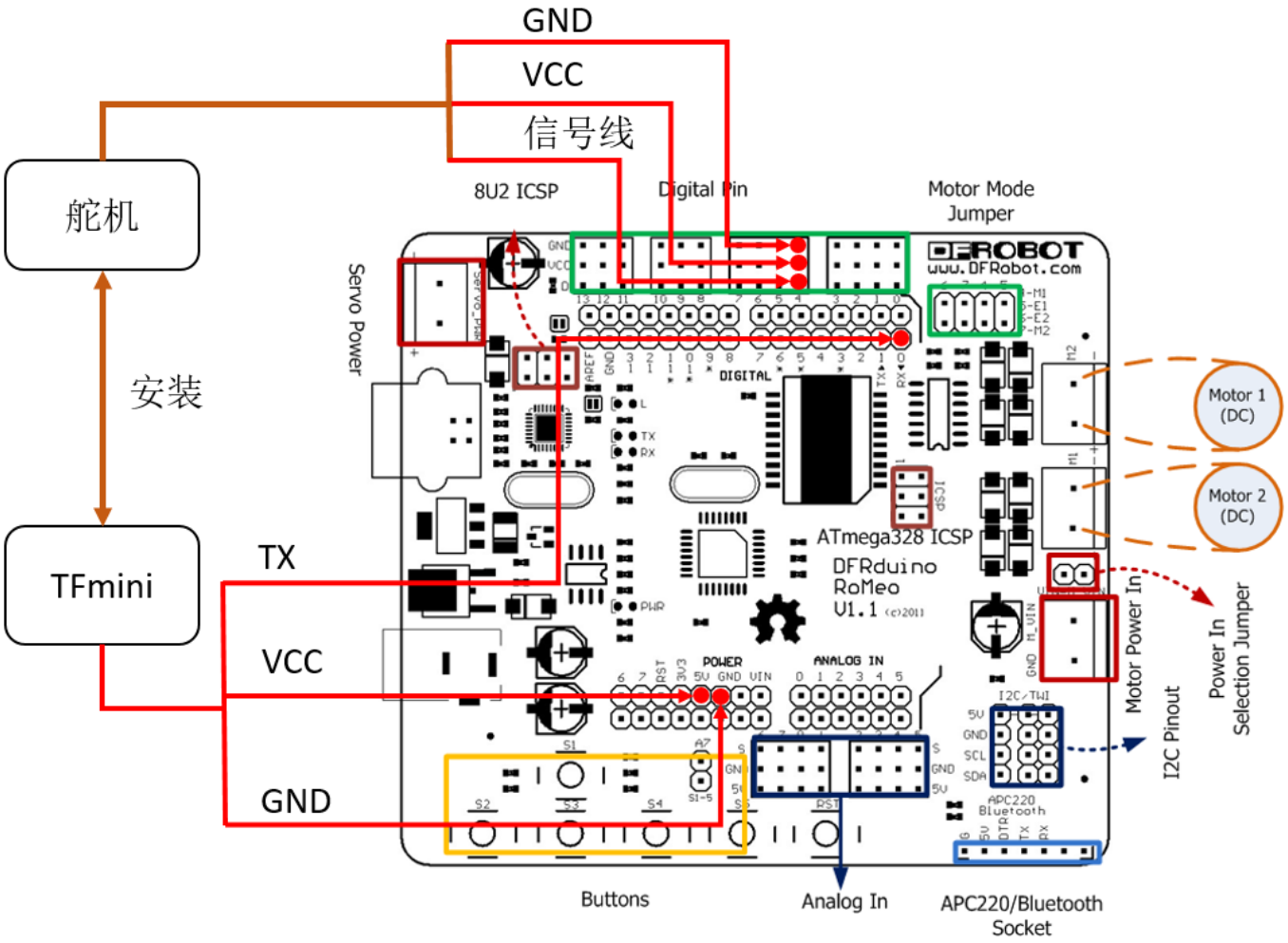


- Benewake TFmini

Detailed parameters of TFmini can be found in the instructions of TFmini.

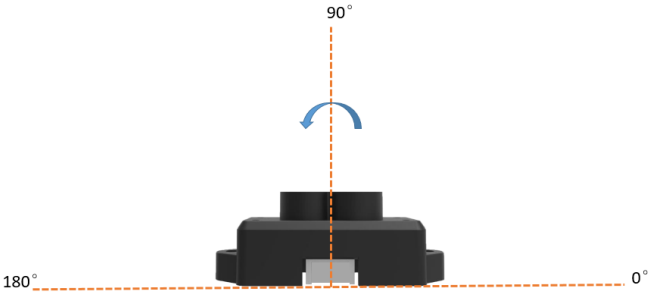
- 9g Steering engine

1.2 Wiring



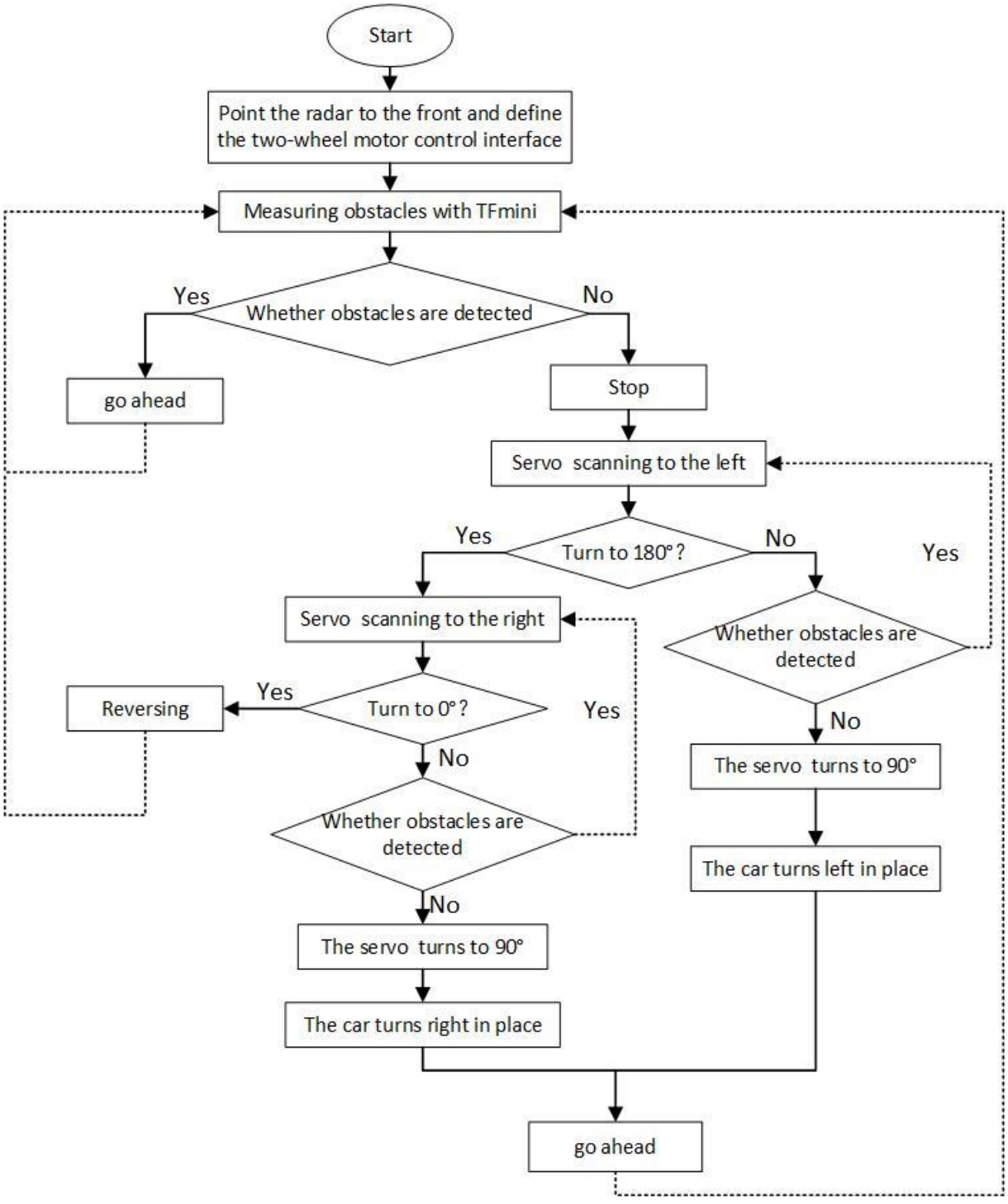
2. principle

When the car started, it began to move forward. When the radar detects obstacles within the forward threshold, the car stops moving and begins to scan left and right. TFmini on the steering gear scans from 90 degrees to 180 degrees, and then from 180 degrees to 0 degrees.



When there is no obstacle in the scanning direction, the car steers in this direction and the steering gear returns to 90 degrees. If there is no traveling route in the left to right scan circle, the car will retreat and the steering gear will be corrected.

The logical flow chart is as follows:



3. considerations

- Current models are only used to explore the feasibility of using TFmini to avoid obstacles, and can not be widely applied to large-scale business scenarios. If necessary, the code of professional software developers should prevail.

- When the external power supply is too heavy, it will affect the friction force of the wheels of the car. Maybe the speed of the two wheels is not the same, resulting in the car can not follow the trajectory.
- Car wheels may cause idling on smooth ground, leading to the car can not go straight.
- If the external power supply to TFmini is supplied separately, the external power supply and the control board should be grounded simultaneously.
- If you are carrying a more complex program, you need to consider the chip's ability. The current development board has found that there will be a Karton phenomenon when running the program.

4. Appendix

4.1 code

```
#include <Servo.h>

Servo myservo;

int pos=90; //Define servo angle
bool flag=true;//Define servo gear
float dist_f;//Define the direction distance of foward
float dist_s;//Define the direction distance of sideway
int E1=5; //Define M1 enablement
int E2=6; //Define M2 enablement
int M1=4; //Define M1 control
int M2=7; //Define M2 control
int temp_distance =0;

/**
  * Two-wheel stopping
  */
void brake(void){
```

```
    digitalWrite(E1,LOW);
    digitalWrite(E2,LOW);
}

/**
  * Two-wheel starting
  */
void advance(char a, char b){
    analogWrite(E1,a);
    digitalWrite(M1,LOW);
    analogWrite(E2,b);
    digitalWrite(M2,LOW);
}

/**
  * Two-wheel retreatting
  */
void back(char a, char b){
    analogWrite(E1,a);
    digitalWrite(M1,HIGH);
    analogWrite(E2,b);
    digitalWrite(M2,HIGH);
}

/**
  * turn left
  */
void turn_L(char a, char b){
```

```
    analogWrite(E1,a);
    digitalWrite(M1,LOW);
    analogWrite(E2,b);
    digitalWrite(M2,HIGH);
}


/**
  * turn right
  */
void turn_R(char a, char b){
    analogWrite(E1,a);
    digitalWrite(M1,HIGH);
    analogWrite(E2,b);
    digitalWrite(M2,LOW);
}

/**
  * Read data of TFmini measurements
  */
void getTFminiData(int* distance, int* strength) {
    static char i = 0;
    char j = 0;
    int checksum = 0;
    static int rx[9];
    if(Serial.available()) {
        rx[i] = Serial.read();
        if(rx[0] != 0x59) {
```

```
            i = 0;
        } else if(i == 1 && rx[1] != 0x59) {
            i = 0;
        } else if(i == 8) {

            for(j = 0; j < 8; j++) {
                checksum += rx[j];
            }
            /*
            if(rx[8] == (checksum % 256)) {
                *distance = rx[2] + rx[3] * 256;
                *strength = rx[4] + rx[5] * 256;
            }*/
            *distance = rx[2] + rx[3] * 256;
                *strength = rx[4] + rx[5] * 256;
            i = 0;
        } else {
            i++;
        }
    }
}


void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200);
    myservo.attach(4);
    brake();
```

```
  /*
   * The radar is facing the front
   */

  myservo.write(pos);

  /*
   * Set the tire motor output
   */
   pinMode(4,OUTPUT);
   pinMode(5,OUTPUT);
   pinMode(6,OUTPUT);
   pinMode(7,OUTPUT);
  delay(10);
}

void loop() {
/*
 * Reading once
 */
        int distance = 0;
        int strength = 0;
        getTFminiData(&distance, &strength);
        while(!distance) {
          getTFminiData(&distance, &strength);
          Serial.print("Distance: ");
          Serial.print(distance);
          Serial.print("cm         ");
```

```
            Serial.print("strength: ");
            Serial.println(strength);


        }
        /*
          * Set the    threshold is 30CM
          */
        if(distance <= 30 && distance > 0){
            temp_distance = distance;
        }
        delay(10);
/*
  * read the reading distance
  * If the distance is less than the threshold, stop and start scanning left to right until there is a gap to go. Then the wheel turns and the scanner returns.
  * If the reading distance is greater than the threshold, the car is starting
  */
          if(temp_distance <= 30 && temp_distance >= 0){
              brake();
              /*
                * Judging whether the current servo should turn left or right
                */
              if(flag){
                 if(pos<170){
                    pos=pos+45;
                 }else{
                   flag = false;
                 }
```

```
        /*
          * If the detection distance is greater than the threshold, the servo is facing the front and the car is turning
          */
        if(distance > 32){
                pos = 90;
                myservo.write(pos);
                delay(1200);
                //whether the car is facing the front or not
                if(pos >= 90){
                  turn_L(35,35);
                }else{
                   turn_R(35,35);
                }
                delay(250);
                temp_distance = distance;
        }
        /*
          * If the detection distance is less than the threshold, continue scanning
          */
        else{
           myservo.write(pos);
           delay(1200);
        }
      }else{
        if(pos>10){
           pos=pos-45;
        }else{
```

```
                flag=true;
            }
            /*
              * If the detection distance is greater than the threshold, the servo is facing the front and the car is turning
              */
            if(distance > 32){
                pos = 90;
                myservo.write(pos);
                delay(1200);
                //Judging Car Direction
                if(pos >= 90){
                  turn_L(35,35);
                }else{
                   turn_R(35,35);
                }
                delay(250);
                temp_distance = distance;
            }
            /*
              * If the detection distance is less than the threshold, continue scanning
              */
            else{
               myservo.write(pos);
               delay(1200);
            }
        }
    }
```

```
        /*
          * If there are no obstacles ahead, go straight.
          */
        else if(distance > 32 && distance < 1200){
          advance(35,35);
        }
        /*
          * If the radar breaks down, the car stops.
          */
        else if(distance == -3){
          brake();
        }
}
```