《机器人编程基础》
实验报告

实验题目：基于 Arduino 的机器人编程

姓名、学号：成子谦，201730681303

学院、班级：软件学院 2017 级 1 班

华南理工大学软件学院

二〇一九年十一月

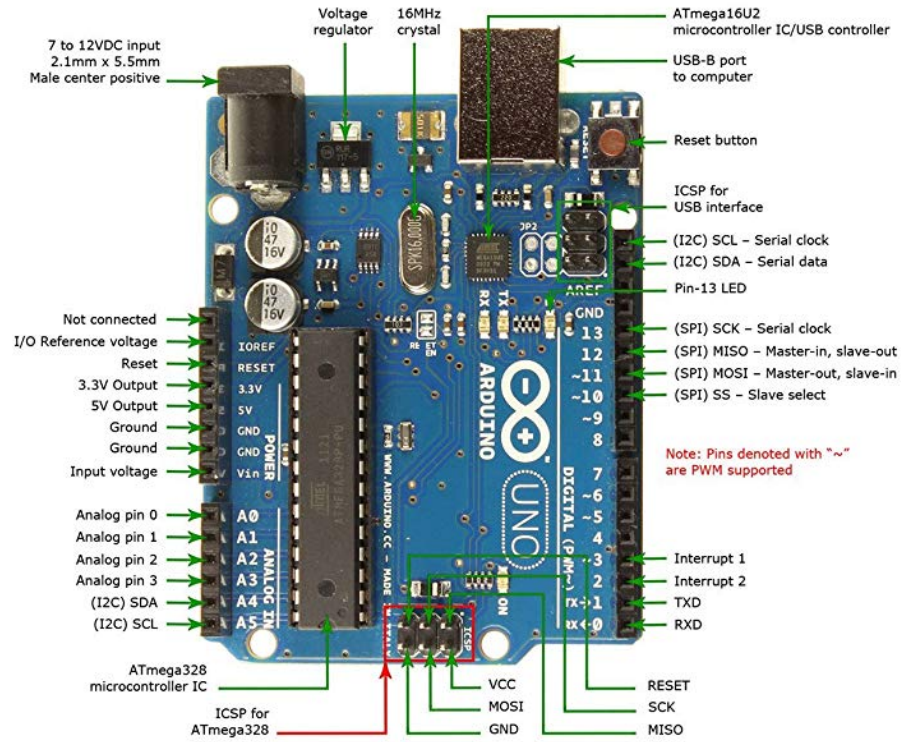
目　录

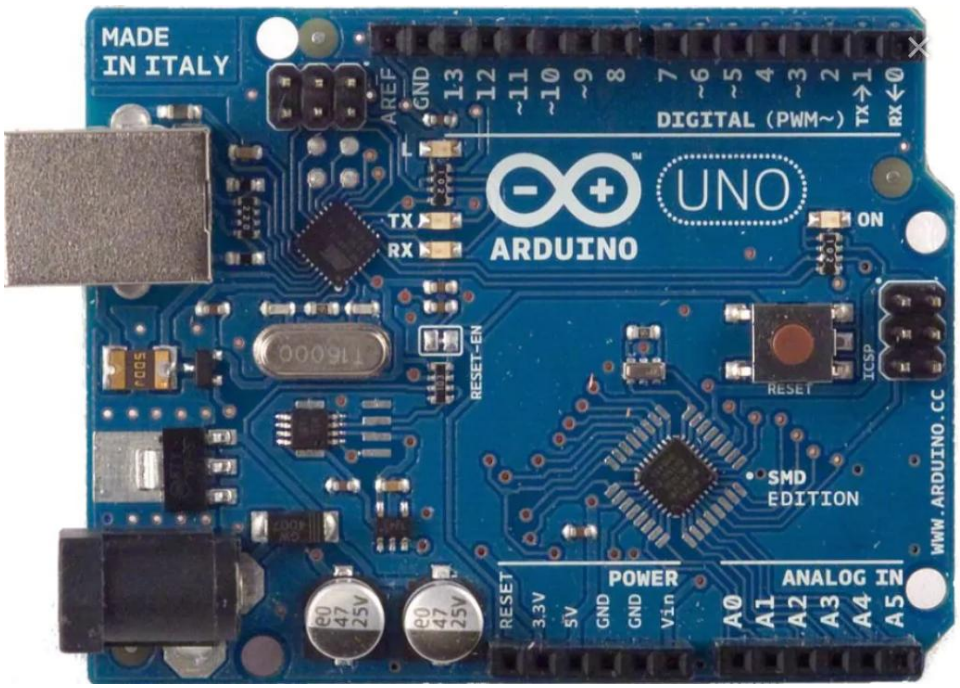
1. 实验目的

1. 掌握机器人的硬件组成
2. 掌握 Arduino 编程

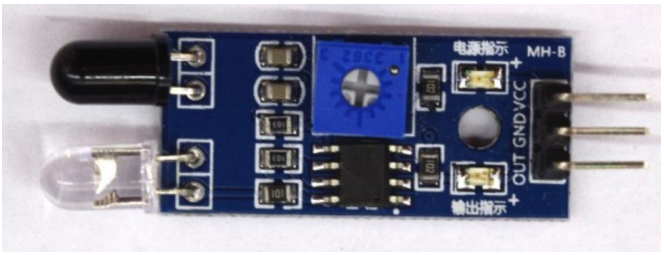
2. 实验材料

2.1. Arduino UNO 控制板

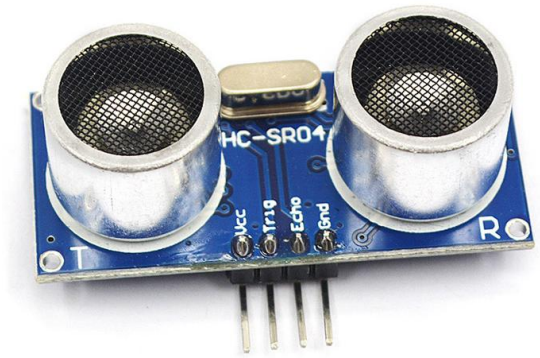




2.2. 红外避障模块



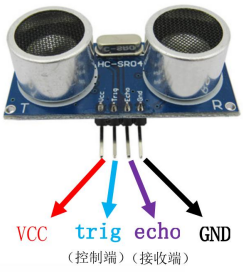
2.3. 超声波测距模块及其支架



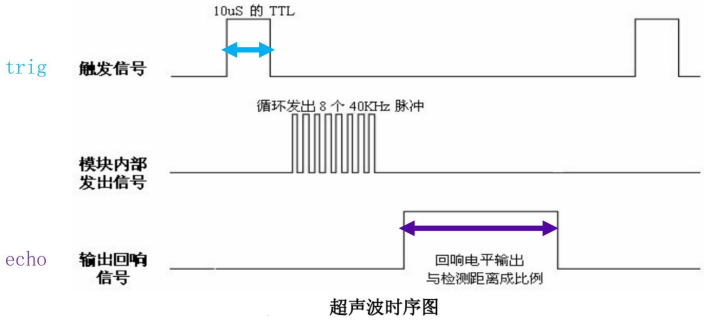
超声波模块

- 功能：测量距离

距离 = 344 m/s × 时间 / 2

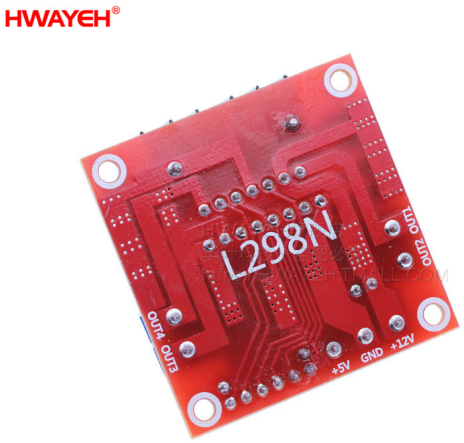


超声波时序图：



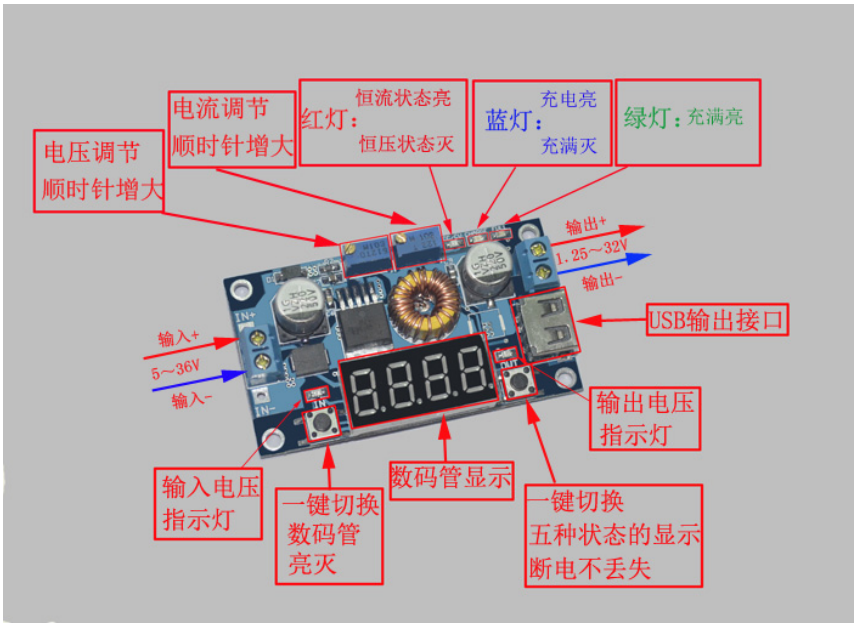
超声波时序图

2.4. 电机驱动模块 L298N



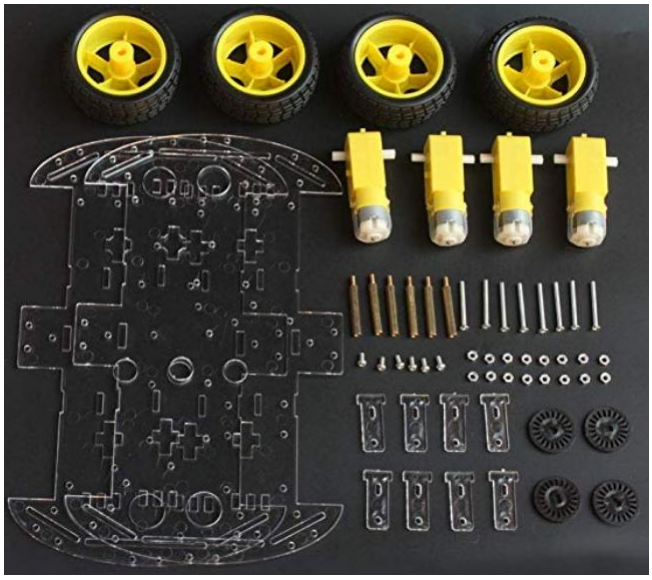
ENA	IN1	IN2	直流电机状态
0	X	X	停止
1	0	0	制动
1	0	1	正转
1	1	0	反转
1	1	1	制动

2.5. 电源稳压模块



调节电压，输出 5v。 一路输出（USB），连接控制板 UNO 的 USB； 另一路输出，连接面包板，提供电源给其他设备。

2.6. 车体及电机



2.7. 面包板

每一组的五孔是联通的，组组之间不连通。

2.8. 杜邦线

杜邦线的两端是公头或母头。

2.9. 电池

2 节 18650 锂电池，单节 3.7v。

2.10.电池盒

两节电池串联，电池盒带开关。

放入电池前，确保开关处于关闭状态，防止短路。

3. 实验内容

3.1. 软件模块

编写各个模块的测试程序，包括：

1. 板载 LED。提示：使用 digitalWrite 函数。
2. 串口通信。提示：使用 Serial 对象。
3. 红外避障。提示：使用 digitalRead 函数。
4. 超声波测距。提示：单片机的一个输出管脚接 Trig，一个输入管脚接 Echo。使用 pulseIn 函数测量 Echo 的脉冲宽度。
5. 电机驱动。提示：使用 PWM 控制电机速度。使用 analogWrite 输出 PWM。

3.2. 组装机器人

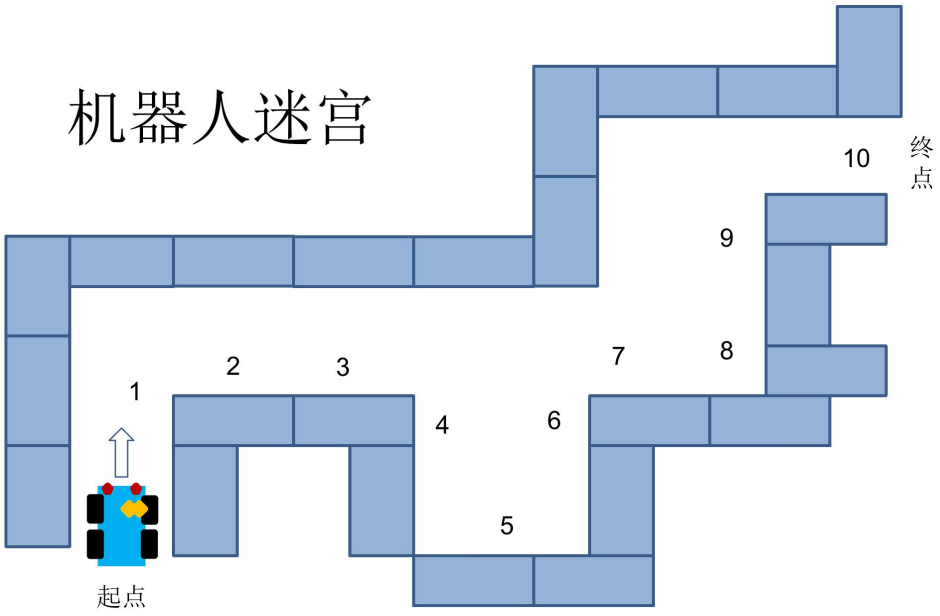
将机器人底盘、轮子、电池、控制器等组合为一个完整机器人，用于机器人走迷宫。

机器人前方安装两个红外避障传感器，右侧安装一个超声波测距传感器。

3.3. 机器人走迷宫

编写走迷宫机器人程序。

提示：机器人控制策略是一直靠右墙走。



附录 A： Arduino UNO 功能测试源代码

```
#define Trig 2
#define Echo 3
float cm, temp;
void setup() {
    // LED 测试
    //   pinMode(LED_BUILTIN, OUTPUT);
    // 红外测试
    //   pinMode(2, INPUT);
    //   pinMode(LED_BUILTIN, OUTPUT);
    // 超声波测试
    Serial.begin(115200);
    pinMode(Trig, OUTPUT);
    pinMode(Echo, INPUT);
}
void loop() {
    // LED 测试
    //   digitalWrite(LED_BUILTIN, HIGH);
    // 串口测试
    //   SoftwareSerial mySerial(2,3);
    //   if (mySerial.available())
    //       Serial.write(mySerial.read());
    //   if (Serial.available())
    //       Serial.write(Serial.read());
    // 红外测试
    //   digitalWrite(LED_BUILTIN, digitalRead(2));
    // 超声波测试
    //   digitalWrite(Trig, LOW);
    //   delay(2);
    digitalWrite(Trig, HIGH);
```

```
    delayMicroseconds(10);
    digitalWrite(Trig, LOW);
    temp = float(pulseIn(Echo, HIGH));
    cm = temp * 17.0 / 1000.0;
    Serial.print("Echo = ");
    Serial.print(temp);
    Serial.print(" | Distance = ");
    Serial.print(cm);
    Serial.println("cm");
    delay(100);
    //    Serial.println("test");
}
```

附录 B：小车主逻辑代码

```
#define TRIG 2
#define ECHO 3
#define WHEEL_RIGHT_1 5
#define WHEEL_RIGHT_2 6
#define WHEEL_LEFT_1 9
#define WHEEL_LEFT_2 10
#define INFRARED_1 11
#define INFRARED_2 12
#define FORWARD_LIMIT 50
#define EPS 1e-6
#define SPEED 120

void setup() {
    // serial init
    Serial.begin(115200);

    // built-in led init
    pinMode(LED_BUILTIN, OUTPUT);

    // ultrasonic wave init
    pinMode(TRIG, OUTPUT);
    pinMode(ECHO, INPUT);

    // wheel init
    pinMode(WHEEL_LEFT_1, OUTPUT);
    pinMode(WHEEL_LEFT_2, OUTPUT);
    pinMode(WHEEL_RIGHT_1, OUTPUT);
    pinMode(WHEEL_RIGHT_2, OUTPUT);
```

```
    // rightNotBlocked init
    pinMode(INFRARED_1, INPUT);
    pinMode(INFRARED_2, INPUT);
}

void stateInit() {
    clearWheelState();
}

void moveForward() {
    analogWrite(WHEEL_LEFT_1, 0);
    analogWrite(WHEEL_LEFT_2, SPEED);
    analogWrite(WHEEL_RIGHT_1, 0);
    analogWrite(WHEEL_RIGHT_2, SPEED);
}

void clearWheelState() {
    analogWrite(WHEEL_LEFT_1, 0);
    analogWrite(WHEEL_LEFT_2, 0);
    analogWrite(WHEEL_RIGHT_1, 0);
    analogWrite(WHEEL_RIGHT_2, 0);
}

void moveBackward() {
    analogWrite(WHEEL_LEFT_1, SPEED);
    analogWrite(WHEEL_LEFT_2, 0);
    analogWrite(WHEEL_RIGHT_1, SPEED);
    analogWrite(WHEEL_RIGHT_2, 0);
}
```

```
void turnRight() {
	clearWheelState();
	analogWrite(WHEEL_RIGHT_1, SPEED);
	analogWrite(WHEEL_RIGHT_2, 0);
	delay(500);
	clearWheelState();
	analogWrite(WHEEL_LEFT_1, 0);
	analogWrite(WHEEL_LEFT_2, SPEED);
	delay(500);
	clearWheelState();
	moveForward();
	delay(200);
}

void turnLeft() {
	clearWheelState();
	analogWrite(WHEEL_LEFT_1, SPEED);
	analogWrite(WHEEL_LEFT_2, 0);
	delay(500);
	clearWheelState();
	analogWrite(WHEEL_RIGHT_1, 0);
	analogWrite(WHEEL_RIGHT_2, SPEED);
	delay(500);
	clearWheelState();
	moveForward();
	delay(200);
}

int rightBlocked() {
	digitalWrite(TRIG, HIGH);
```

```
    delayMicroseconds(10);
    digitalWrite(TRIG, LOW);

    float temp = float(pulseIn(ECHO, HIGH));
    float cm = temp * 17 / 1000.0;
    Serial.print("Echo = ");
    Serial.print(temp);
    Serial.print(" | Distance = ");
    Serial.print(cm);
    Serial.println("cm");
    if (cm - FORWARD_LIMIT < EPS) {
        return 1;
    } else {
        return 0;
    }
}

int frontTest() {
    int leftNotBlocked = digitalRead(INFRARED_1);
    int rightNotBlocked = digitalRead(INFRARED_2);
    if (leftNotBlocked && rightNotBlocked) return 1;
    else if (leftNotBlocked && !rightNotBlocked) return 2;
    else if (!leftNotBlocked && rightNotBlocked) return 3;
    else return 4;
}

void loop() {
    // car state init
    stateInit();
    int res = frontTest();
```

```
    if (res == 1) {
        moveForward();
        delay(300);
    } else if (res == 2 || res == 3) {
        if (!rightBlocked()) turnRight();
        else turnLeft();
    } else if (!rightBlocked()) {
        turnRight();
    } else turnLeft();
}
```

提交文件说明

|-report.doc

|-report.pdf

|-src

    |-function_test.ino

    |-car_logic.ino

|-image

    |-Arduino 小车.jpg

|-video

    |-Arduino 小车.mp4

    |-红外测试.mp4

    |-超声测试.mp4