Chapter8 红外避障

实验目的

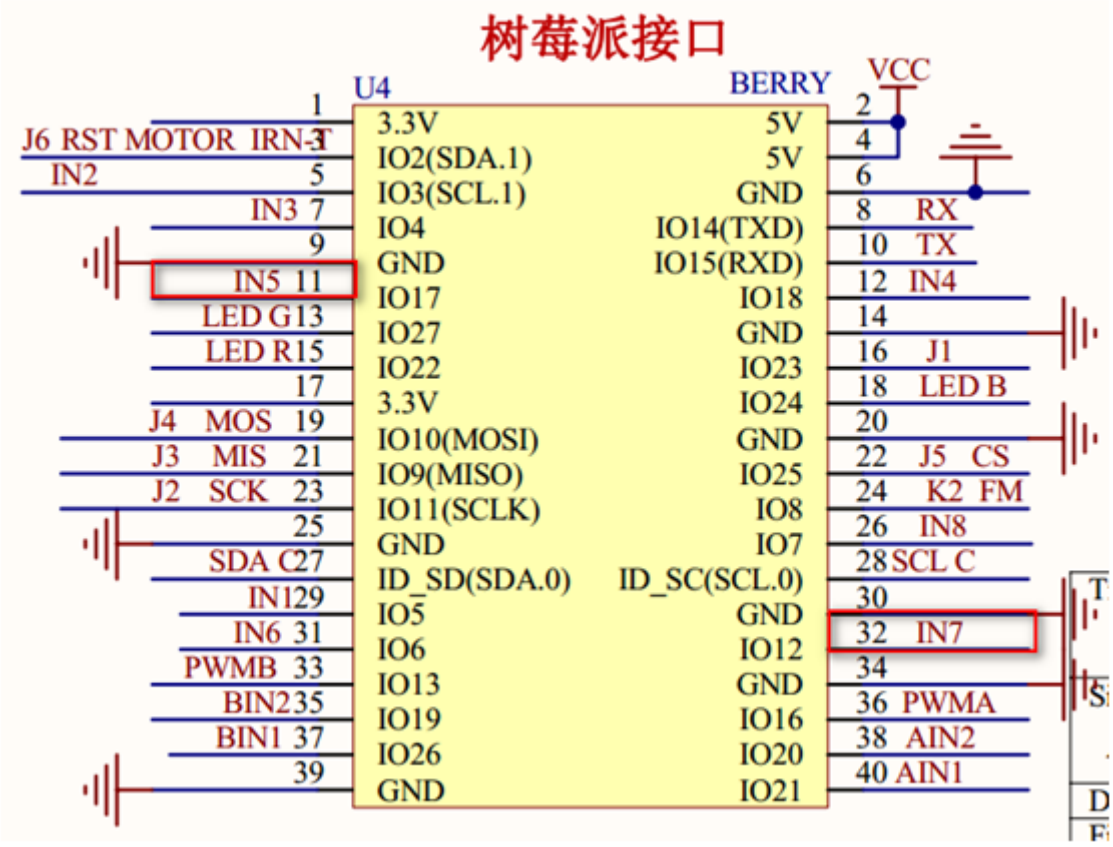
远程登录树莓派系统之后，运行程序，按下按键K2，小车启动红外避障功能，当前方有障碍物时，向相反的方向转向避障。

实验原理

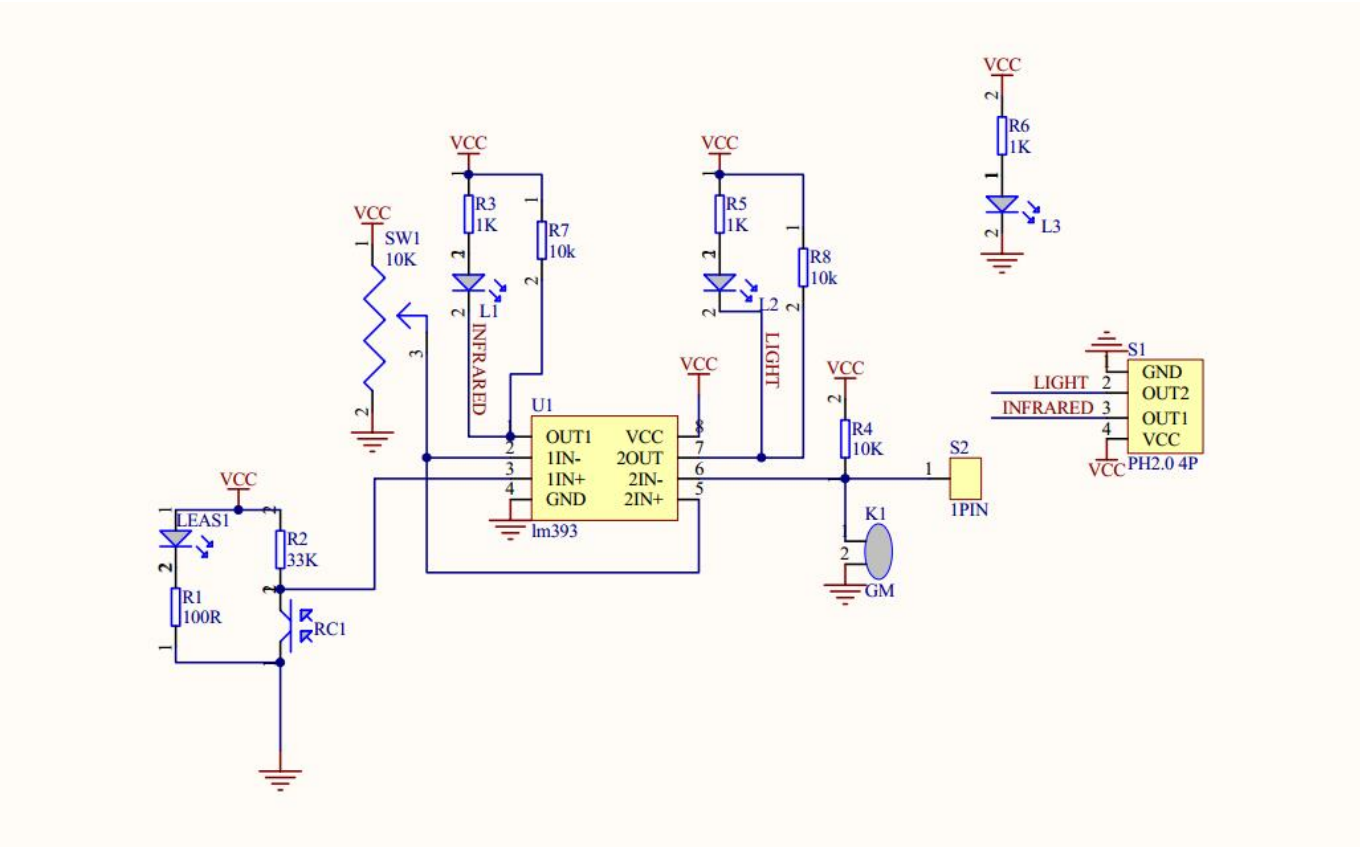
红外传感器原理

红外传感器具有一对红外发射与接收管，发射管发射出一定频率的红外线。红外传感器避障的基本原理是利用物体的反射性质，在一定范围内，如果没有障碍物，发射出去的红外线，因为传播的距离越来越远而逐渐减弱，最后消失。如果有障碍物，红外线遇到障碍物，会被反射到达传感器接收头。可通过红外传感器模块上的电位器设置红外避障的距离。

硬件原理



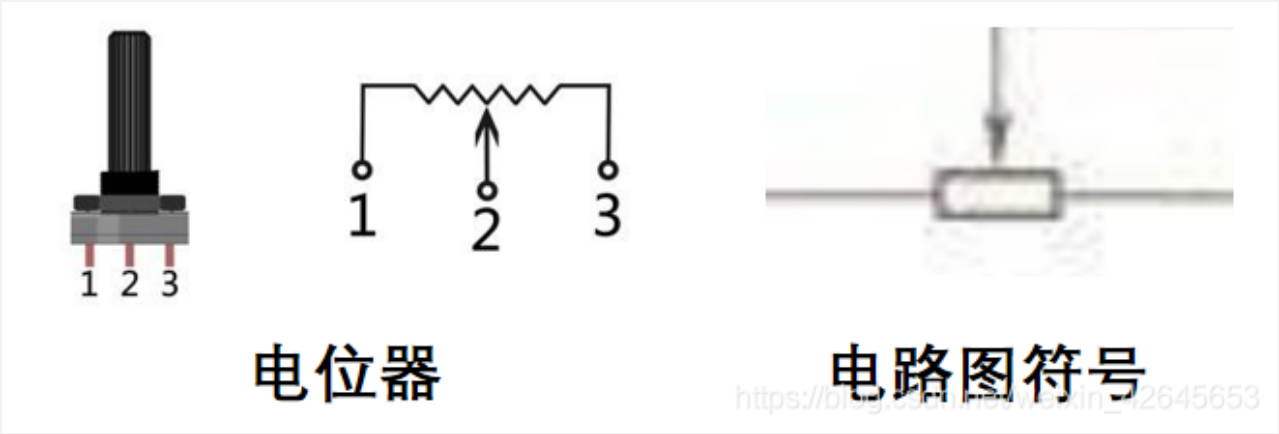
图一 树莓派接口



图二 红外传感器模块接口及电路

传感器模块提供了四个引脚，其中OUT1控制红外传感测距，OUT2控制循光。

我们来看电路图，可以看到**1IN-**和**2IN+**引脚都接到了同一个电路模块，该器件叫做电位器(类似于滑动变阻器)。可以用来调整红外测距时电平发生改变的距离。

> 电位器属于**无极性**器件，**可变电阻**的一种，三个触点，通过旋转旋钮改变2号脚的位置，从而改变阻值的大小，1脚和3脚分别接5V和GND，2脚接模拟输入引脚。
>
> 
>

代码实现

```
#-*- coding:UTF-8 -*-
import RPi.GPIO as GPIO
import time

#小车电机引脚定义
```

```
AIN1 = 21
AIN2 = 20

BIN1 = 26
BIN2 = 19

PWMA = 16
PWMB = 13

#小车按键定义
key = 8

#红外避障引脚定义
AvoidSensorLeft = 12
AvoidSensorRight = 17

#设置GPIO口为BCM编码方式
GPIO.setmode(GPIO.BCM)

#忽略警告信息
GPIO.setwarnings(False)

#电机引脚初始化为输出模式
#按键引脚初始化为输入模式
#红外避障引脚初始化为输入模式
def init():
    global pwm_PWMA
    global pwm_PWMB
     #初始化引脚
    GPIO.setup(PWMA,GPIO.OUT,initial=GPIO.HIGH)
    GPIO.setup(AIN2,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(AIN1,GPIO.OUT,initial=GPIO.LOW)

    GPIO.setup(PWMB,GPIO.OUT,initial=GPIO.HIGH)
    GPIO.setup(BIN2,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(BIN1,GPIO.OUT,initial=GPIO.LOW)

    GPIO.setup(key,GPIO.IN)
    GPIO.setup(AvoidSensorLeft,GPIO.IN)
    GPIO.setup(AvoidSensorRight,GPIO.IN)
    #设置pwm引脚和频率为2000hz
    pwm_PWMA = GPIO.PWM(PWMA, 2000)
    pwm_PWMB = GPIO.PWM(PWMB, 2000)
    pwm_PWMA.start(0)
    pwm_PWMB.start(0)

#小车前进
def run():
    pass
#小车后退
def back():
    pass
#小车左转
def left():
```

```
    pass

#小车右转
def right():
    pass

#小车原地左转
def spin_left():
    pass

#小车原地右转
def spin_right():
    pass

#小车停止
def brake():
    pass

#按键检测
def key_scan():
    pass

#延时2s
time.sleep(2)

#try/except语句用来检测try语句块中的错误,
#从而让except语句捕获异常信息并处理。
try:
    init()
    key_scan()
    while True:
        #遇到障碍物,红外避障模块的指示灯亮,端口电平为LOW
        #未遇到障碍物,红外避障模块的指示灯灭,端口电平为HIGH
        LeftSensorValue  = GPIO.input(AvoidSensorLeft)
        RightSensorValue = GPIO.input(AvoidSensorRight)

        if LeftSensorValue == True and RightSensorValue == True:
            run()         #当两侧均未检测到障碍物时调用前进函数
        elif LeftSensorValue == True and RightSensorValue == False:
            spin_left()   #右边探测到有障碍物，原地向左转
            time.sleep(0.002)
        elif RightSensorValue == True and LeftSensorValue == False:
            spin_right()  #左边探测到有障碍物，原地向右转
            time.sleep(0.002)
        elif RightSensorValue == False and LeftSensorValue == False :
            spin_right()  #当两侧均检测到障碍物时,刹车
            time.sleep(0.002)

except KeyboardInterrupt:
    pass

pwm_PWMA.stop()
pwm_PWMB.stop()
GPIO.cleanup()
```