Ultrasonic Ranging

Overview

In this project, we use ultrasonic ranging module to measure distance, and print out the data in the terminal.

Experimental Materials:

Raspberry Pi *1

GPIO Expansion Board & Wire x1

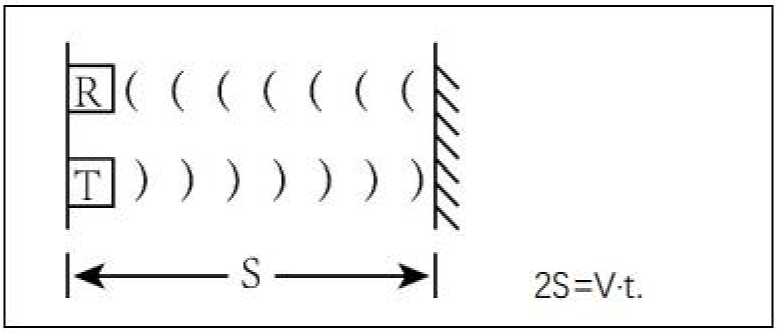
Breadboard*1

Some DuPont lines

Ultrasonic Ranging

Product description：

Ultrasonic ranging module use the principle that ultrasonic will reflect when it encounters obstacles. Start counting the time when ultrasonic is transmitted. And when ultrasonic encounters an obstacle, it will reflect back. The counting will end after ultrasonic is received, and the time difference is the total time of ultrasonic from transmitting to receiving. Because the speed of sound in air is constant, and is about v=340m/s. So we can calculate the distance between the model and the obstacle: s=vt/2.



Ultrasonic module integrates a transmitter and a receiver. The transmitter is used to convert electrical signals (electrical energy) into sound waves (mechanical energy) and the function of the receiver is opposite.

Pin description

VCC	power supply pin
Trig	trigger pin
Echo	Echo pin
GND	GND

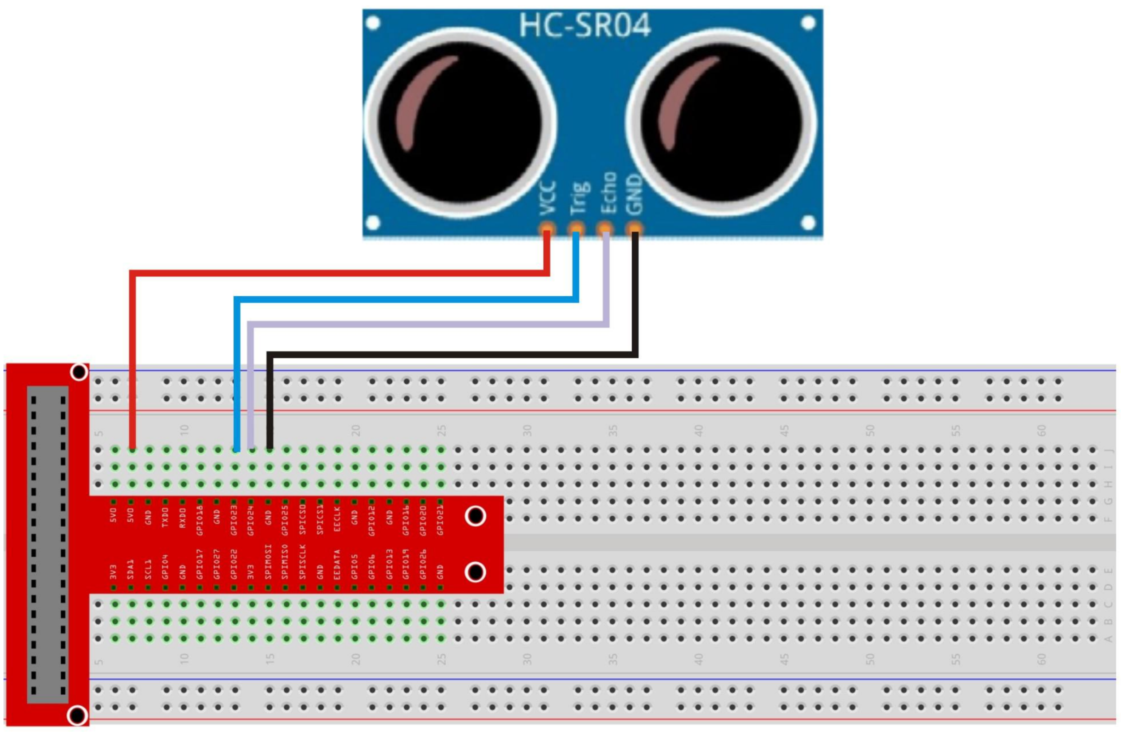
Technical Parameters:

Working voltage: 5V Working current: 12mA

Minimum measured distance: 2cm Maximum measured distance: 200cm

Instructions for use: output a high-level pulse in Trig pin lasting for least 10uS. Then the module begins to transmit ultrasonic. At the same time, the Echo pin will be pulled up. When the module receives the returned ultrasonic, the Echo pin will be pulled down. The duration of high level in Echo pin is the total time of the ultrasonic from transmitting to receiving, s=vt/2.

Wiring diagram:



C code:

```
#include <wiringPi.h>
#include <stdio.h>
#include <sys/time.h>

#define trigPin 4
#define echoPin 5
#define MAX_DISTANCE 220        // define the maximum measured distance
#define timeOut MAX_DISTANCE*60 // calculate timeout according to the maximum
measured distance
//function pulseIn: obtain pulse time of a pin
int pulseIn(int pin, int level, int timeout);
float getSonar(){   // get the measurement results of ultrasonic module,with
unit: cm
    long pingTime;
    float distance;
    digitalWrite(trigPin,HIGH); //trigPin send 10us high level
    delayMicroseconds(10);
    digitalWrite(trigPin,LOW);
    pingTime = pulseIn(echoPin,HIGH,timeOut);   //read plus time of echoPin
    distance = (float)pingTime * 340.0 / 2.0 / 10000.0; // the sound speed is
```

```
340m/s,and calculate distance
    return distance;
}

int main(){
    printf("Program is starting ... \n");
    if(wiringPiSetup() == -1){ //when initialize wiring failed,print messageto
screen
        printf("setup wiringPi failed !");
        return 1;
    }
    float distance = 0;
    pinMode(trigPin,OUTPUT);
    pinMode(echoPin,INPUT);
    while(1){
        distance = getSonar();
        printf("The distance is : %.2f cm\n",distance);
        delay(1000);
    }
    return 1;
}

int pulseIn(int pin, int level, int timeout)
{
   struct timeval tn, t0, t1;
   long micros;
   gettimeofday(&t0, NULL);
   micros = 0;
   while (digitalRead(pin) != level)
   {
      gettimeofday(&tn, NULL);
      if (tn.tv_sec > t0.tv_sec) micros = 1000000L; else micros = 0;
      micros += (tn.tv_usec - t0.tv_usec);
      if (micros > timeout) return 0;
   }
   gettimeofday(&t1, NULL);
   while (digitalRead(pin) == level)
   {
      gettimeofday(&tn, NULL);
      if (tn.tv_sec > t0.tv_sec) micros = 1000000L; else micros = 0;
      micros = micros + (tn.tv_usec - t0.tv_usec);
      if (micros > timeout) return 0;
   }
   if (tn.tv_sec > t1.tv_sec) micros = 1000000L; else micros = 0;
```

```
    micros = micros + (tn.tv_usec - t1.tv_usec);
    return micros;
}
```

Python code:

```
#!/usr/bin/env python3
import RPi.GPIO as GPIO
import time

trigPin = 16
echoPin = 18
MAX_DISTANCE = 220          #define the maximum measured distance
timeOut = MAX_DISTANCE*60   #calculate timeout according to the maximum
measured distance

def pulseIn(pin,level,timeOut): # function pulseIn: obtain pulse time of a pin
    t0 = time.time()
    while(GPIO.input(pin) != level):
        if((time.time() - t0) > timeOut*0.000001):
            return 0;
    t0 = time.time()
    while(GPIO.input(pin) == level):
        if((time.time() - t0) > timeOut*0.000001):
            return 0;
    pulseTime = (time.time() - t0)*1000000
    return pulseTime

def getSonar():     #get the measurement results of ultrasonic module,with unit:
cm
    GPIO.output(trigPin,GPIO.HIGH)      #make trigPin send 10us high level
    time.sleep(0.00001)     #10us
    GPIO.output(trigPin,GPIO.LOW)
    pingTime = pulseIn(echoPin,GPIO.HIGH,timeOut)    #read plus time of echoPin
    distance = pingTime * 340.0 / 2.0 / 10000.0      # the sound speed is 340m/s,
and calculate distance
    return distance

def setup():
    print ('Program is starting...')
    GPIO.setmode(GPIO.BOARD)       #numbers GPIOs by physical location
    GPIO.setup(trigPin, GPIO.OUT)   #
    GPIO.setup(echoPin, GPIO.IN)    #

def loop():
    GPIO.setup(11,GPIO.IN)
    while(True):
        distance = getSonar()
        print ("The distance is : %.2f cm"%(distance))
        time.sleep(1)

if __name__ == '__main__':     #program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt:  #when 'Ctrl+C' is pressed, the program will exit
        GPIO.cleanup()         #release resource
```

Experimental results:

In the directory where the code file is located, execute the following command

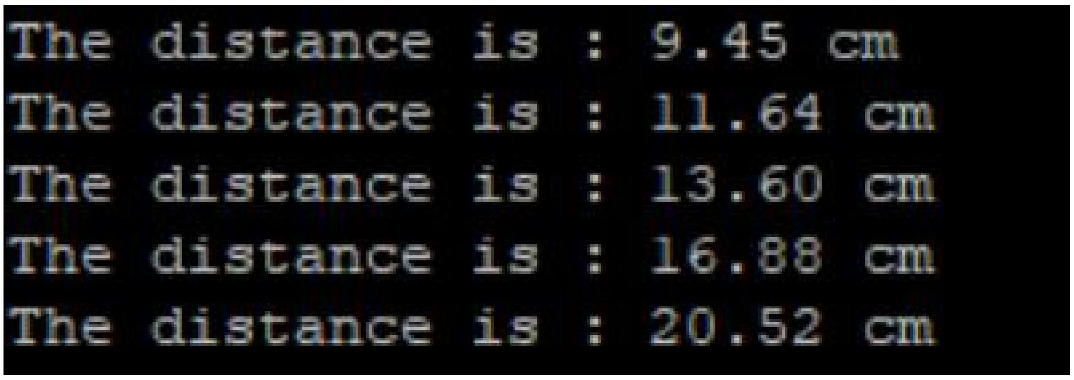
C:

```
gcc -Wall -o UltrasonicRanging UltrasonicRanging.c -lwiringPi
sudo ./UltrasonicRanging
```

Python:

```
python UltrasonicRanging.py
```

After the program is executed, make the detector of ultrasonic ranging module aim at the plane of an object, then the distance between the ultrasonic module and the object will be displayed in the terminal. As is shown below:



```
The distance is : 9.45 cm
The distance is : 11.64 cm
The distance is : 13.60 cm
The distance is : 16.88 cm
The distance is : 20.52 cm
```