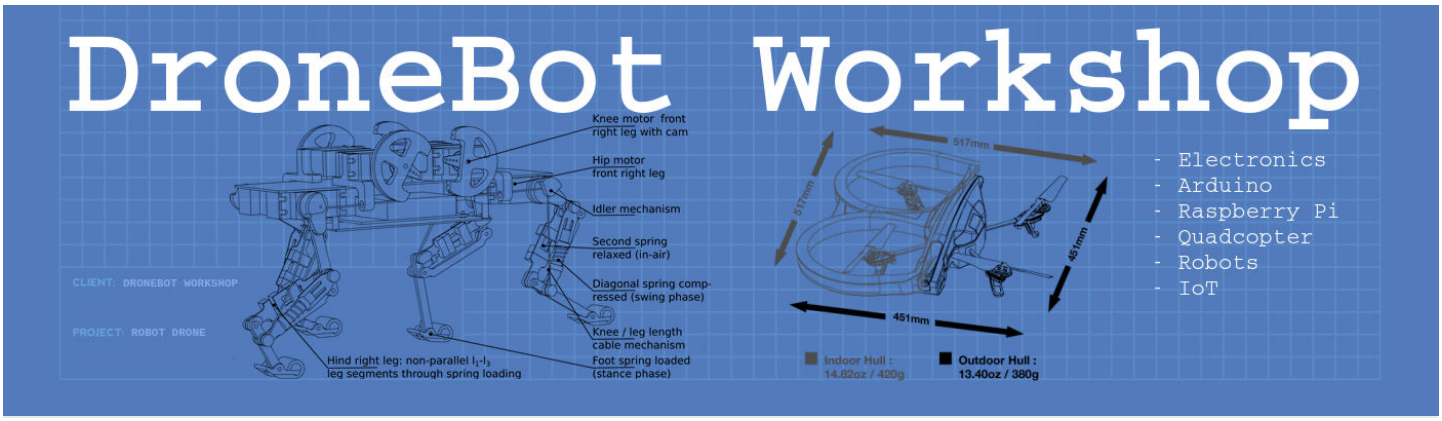


Taller DrobeBot

Construye tu propio Drones, Quadcopters y Robots - ¡Bienvenidos al Taller!

Uso del sensor de distancia ultrasónico HC-SR04 con Arduino

Uno de los sensores más útiles para proyectos de robótica es un sensor de distancia. El HC-SR04 es un sensor de distancia ultrasónico de bajo costo que puede ayudar a su robot a navegar por una habitación. Con un poco de cuidado y un componente adicional también se puede utilizar como un dispositivo de medición. En este artículo aprenderás todo lo que necesitas saber para usar este pequeño dispositivo maravilloso con un Arduino.

Using the HC-SR04 Ultrasonic Distance Sensor wit...

Artículos

Arduino

Electrónica

IoT

QuadCopter

Frambuesa Pi

Robots

Tabla de contenido

El Sensor de Distancia Ultrasónica HC-SR04

El Sensor de Distancia Ultrasónica HC-SR04 es un dispositivo barato que es muy útil para robótica y proyectos de equipos de prueba. Este pequeño sensor es capaz de medir la distancia entre sí y el objeto sólido más cercano, que es realmente buena información para tener si usted está tratando de evitar conducir a una pared!

El HC-SR04 puede ser conectado directamente a un Arduino u otro microcontrolador y funciona a 5 voltios. También se puede utilizar con el Raspberry Pi, sin embargo, dado que el HC-SR04 requiere una lógica de 5 voltios, necesitará un par de resistencias para conectarlo con el puerto GPIO de 3,3 voltios del Pi.

Este sensor de distancia ultrasónica es capaz de medir distancias entre 2 cm a 400 cm (es decir, de una pulgada a 13 pies para aquellos de ustedes que no hablan "métrico"). Es un dispositivo de corriente baja por lo que es adecuado para dispositivos con pilas. Y como un bono incluso se ve bien, como un par de Wall-E Robot ojos para su última invención robótica!

Así que sigue leyendo y te mostraré cómo conectar y usar el HC-SR04 Ultrasonic Distance Sensor. También lo haremos a través de algunas pruebas para ver lo exacto que es y veremos cómo podemos mejorar esta precisión. Y por supuesto voy a tener algunos ejemplos de código y proyectos para que usted pruebe.

¡Empecemos!

Cómo funciona el HC-SR04

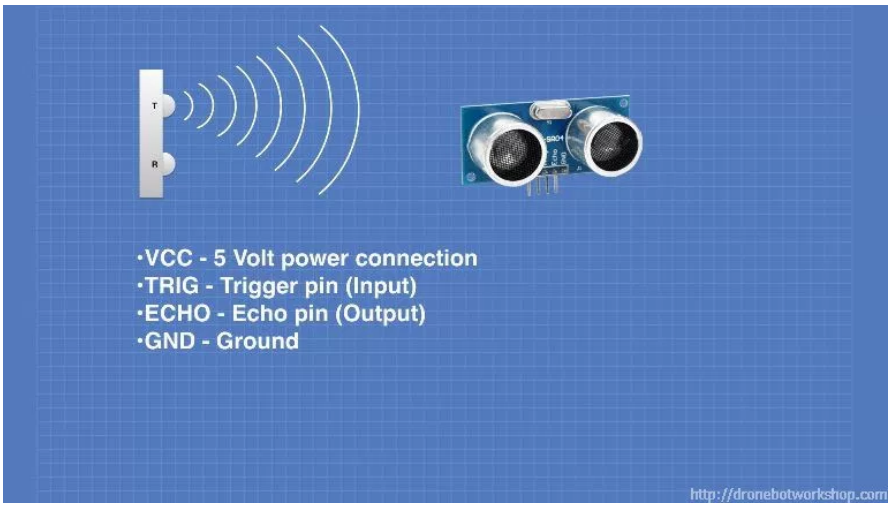
Los sensores ultrasónicos de la distancia utilizan pulsos del sonido ultrasónico (sonido sobre la gama del oído humano) para detectar la distancia entre ellos y los objetos sólidos cercanos. Los sensores se componen de dos componentes principales:

- **Un Transmisor Ultrasónico** - Transmite los impulsos ultrasónicos de sonido, opera a 40 KHz

- **Un receptor ultrasónico** - El receptor escucha los impulsos transmitidos. Si los recibe, produce un impulso de salida cuya anchura puede usarse para determinar la distancia recorrida por el impulso.

El HC-SR04 tiene las siguientes cuatro conexiones:

- **VCC** - Esta es la fuente de alimentación positiva de 5 voltios.
- **Trig** - Este es el pin "Trigger", el accionado para enviar los pulsos ultrasónicos.
- **Echo** - Éste es el pin que produce un pulso cuando se recibe la señal reflejada. La longitud del impulso es proporcional al tiempo que tardó en detectarse la señal transmitida.
- **GND** - Este es el pin de tierra.

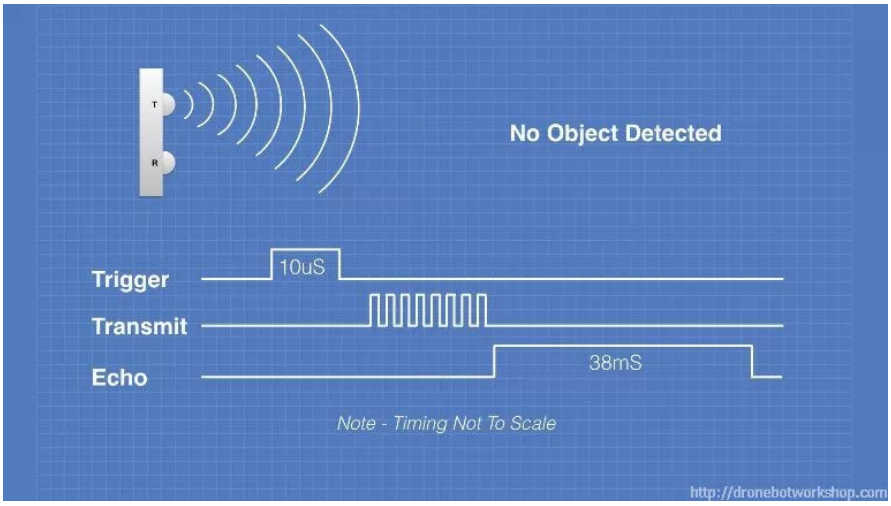


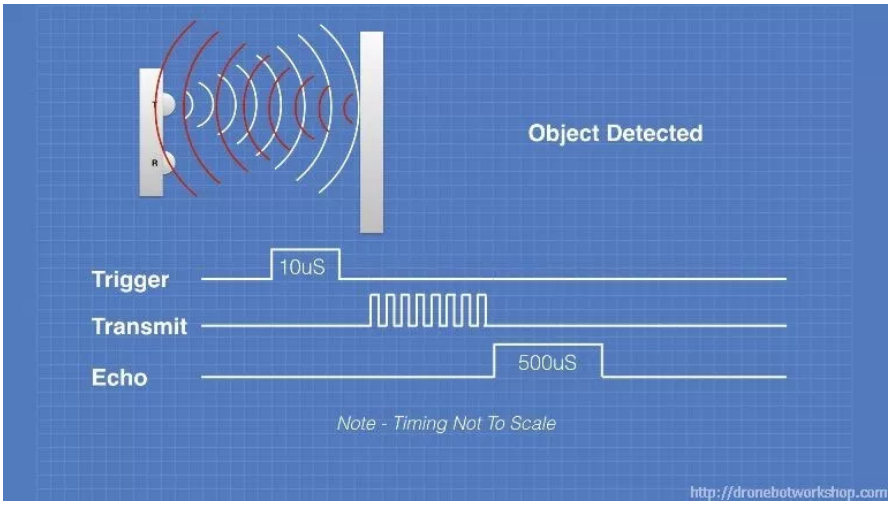
El dispositivo funciona de la siguiente manera:

1. Un pulso de 5 voltios de por lo menos 10 uS (10 microsegundos) de duración se aplica al pin del gatillo.
2. El HC-SR04 responde transmitiendo una ráfaga de ocho impulsos a 40 KHz. Este patrón de 8 pulsos hace que la "firma ultrasónica" del dispositivo sea única, permitiendo al receptor discriminar entre el patrón transmitido y el ruido de fondo ultrasónico.
3. Los ocho impulsos ultrasónicos viajan a través del aire lejos del transmisor. Mientras tanto, el pin Echo se pone alto para comenzar a formar el comienzo de la señal de eco-retroceso.
4. Si el pulso en NOT se refleja de nuevo, la señal de eco se apagará después de 38 mS (38 milisegundos) y volverá a la baja. Esto produce un pulso de 38 mS que indica que no hay obstrucción dentro del rango del sensor.
5. Si el pulso es reflejado hacia atrás, el pin Echo se pone bajo cuando se recibe la señal. Esto produce un pulso cuya

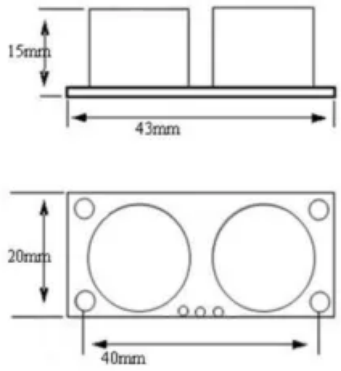
anchura varía entre 150 uS y 25 mS, dependiendo del tiempo que tomó para que la señal sea recibida.

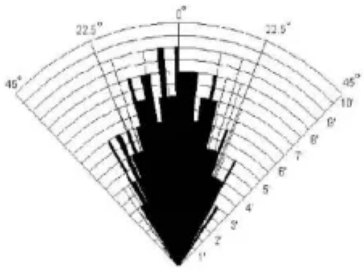
6. La anchura del impulso recibido se utiliza para calcular la distancia al objeto reflejado. Recuerde que el pulso indica el tiempo que tomó para que la señal sea enviada y reflejada de nuevo para obtener la distancia que necesitará para dividir su resultado a la mitad.





La siguiente ilustración muestra las dimensiones del sensor de distancia ultrasónico HC-SR04, así como el ángulo de funcionamiento efectivo. Como puede ver, el sensor es más preciso cuando el objeto que se va a detectar está directamente frente a él, pero obtiene una respuesta de los objetos dentro de una "ventana" de 45 grados. La documentación recomienda limitar esa ventana a 30 grados (15 grados en ambos lados) para lecturas precisas.

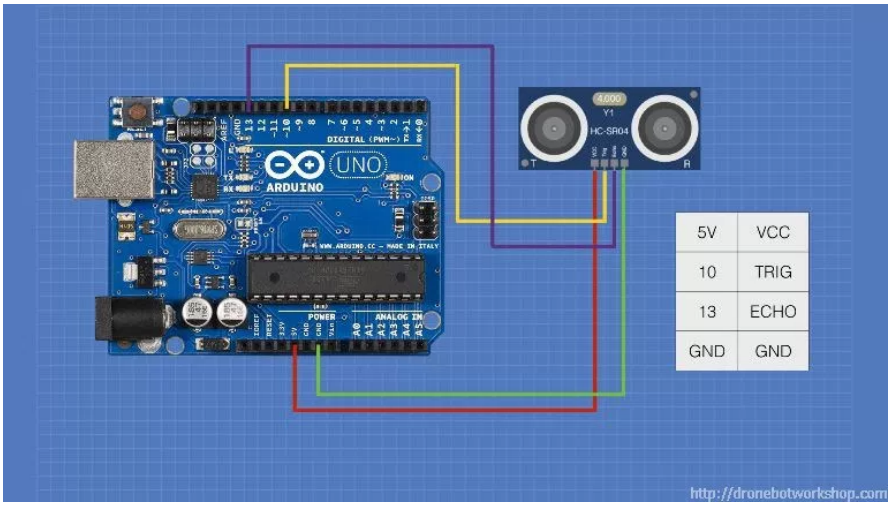




Practical test of performance,
Best in 30 degree angle

Conexión del HC-SR04

Conectar el HC-SR04 al Arduino es bastante fácil. Necesitará un par de puertos digitales de E / S y una conexión a los pines de 5 Voltios y tierra de Arduino.



En realidad, si usted no tiene pasadores, puede conectar los pines de gatillo y Echo del HC-SR04 a un solo pin de E / S digital en el Arduino y usar el código para cambiar el pin entre la salida (para enviar el pulso de 10 uS) Y entrada (para recibir el impulso Eco). Algunos sensores ultrasónicos en realidad tienen sólo un pin que hace tanto el gatillo como el eco. Voy a discutir esto y dar un ejemplo más abajo, así que sigue leyendo.

La mayoría de los ejemplos que voy a mostrar aquí utilizan el método más convencional de dos pines. Cualquier Arduino y cualquier pines digitales de E / S que estén libres pueden usarse así que si desea conectarlo a un conjunto diferente de pines de E / S simplemente cambie los bocetos para reflejar esos

cambios. Para nuestra demostración utilizaré un Arduino Uno y pin 10 para el Trigger y pin 13 para el Echo.

Las notas de la aplicación para el HC-SR04 hacen hincapié en que usted necesita tener el pin de tierra conectado antes de conectar VCC (5 voltios), por lo que si está experimentando "en vivo" en una placa de soldadura sin soldadura que tal vez quiera tener en cuenta .

Tan ahora que hemos conectado nuestro sensor ultrasónico de la distancia es hora de escribir un cierto código y de probarlo hacia fuera.

Demostración básica de Arduino

En nuestra primera demostración probaremos simplemente el sensor para ver si está funcionando. El boceto es bastante simple y utiliza el Monitor Serial para mostrar la distancia que detecta, medida en centímetros. Vamos a mirarlo en detalle.

Con el fin de probar la precisión del sensor de distancia ultrasónica, configuro una placa de prueba con un sensor montado en un extremo (he usado velcro para montarlo). Puse un palo de 1 metro en el tablero para poder probar el sensor en el rango de 2 a 100 cm. Puede ver el video asociado a este artículo si desea ver los resultados de esta y de las otras demostraciones.

Si desea mostrar los resultados en pulgadas en lugar de centímetros, hay dos maneras de hacerlo:

1. Utilice el valor Imperial para la velocidad del sonido en lugar de la métrica. Al nivel del mar a 20 grados Celsius (68 grados Fahrenheit) el sonido viaja a 343 metros por segundo, que es 1,125 pies por segundo o 13500 pulgadas por segundo.
2. Guarde el código tal como está pero convierta a pulgadas al final. Hay 2.54 centímetros en una pulgada. Personalmente, esto es lo que haría esto, ya que me permitiría mostrar los resultados en valores tanto Imperial como Metric.

De lo contrario, sólo use el sistema métrico, que se utiliza en todo el mundo y (lo que es más importante) que se utilizó en Star Trek La próxima generación. Si es lo suficientemente bueno para el capitán Picard, entonces es lo suficientemente bueno para mí!

Así que al bosquejo. Este es el bosquejo básico de HC-SR04, para un desglose detallado de cómo funciona basta con mirar el video o simplemente leer los comentarios en el código.

```
/*
  HC-SR04 Basic Demonstration
  HC-SR04-Basic-Demo.ino
  Demonstrates functions of HC-SR04 Ultrasonic Range Find
  Displays results on Serial Monitor

  DroneBot Workshop 2017
  http://dronebotworkshop.com
*/

// This uses Serial Monitor to display Range Finder dista

// Hook up HC-SR04 with Trig to Arduino Pin 10, Echo to A

#define trigPin 10
#define echoPin 13

float duration, distance;

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {

  // Write a pulse to the HC-SR04 Trigger Pin

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the response from the HC-SR04 Echo Pin

  duration = pulseIn(echoPin, HIGH);

  // Determine distance from duration
  // Use 343 metres per second as speed of sound

  distance = (duration / 2) * 0.0343;

  // Send results to Serial Monitor

  Serial.print("Distance = ");
  if (distance >= 400 || distance <= 2) {
     Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
    delay(500);
  }
  delay(500);
}
```

Bibliotecas de código Arduino

En nuestro primer boceto no usamos bibliotecas de código, simplemente usamos el comando *delayMicrosecond* de Arduino para crear nuestro pulso de 10 uS para activar y el comando *pulseIn* para medir el ancho de pulso de la señal recibida. Sin embargo, hay otras maneras de hacerlo utilizando bibliotecas de código especial. Hay bastantes de ellos disponibles, el más versátil es uno llamado "NewPing".

Si no has tenido ninguna experiencia usando bibliotecas en tus bocetos de Arduino es una habilidad que realmente necesitas aprender. Las bibliotecas proporcionan funciones de código para tareas específicas y hay literalmente cientos de bibliotecas disponibles para el Arduino para tareas y para soportar todo tipo de hardware externo.

El sitio web de Arduino tiene instrucciones para instalar nuevas bibliotecas en su IDE de Arduino .

La biblioteca de NewPing fue escrita por Tim Eckel y reemplaza a la biblioteca Ping más antigua que fue escrita por Caleb Zulawski y diseñada principalmente para el sensor de ultrasonidos Parallax Ping (aunque funcionará con el HC-SR04 si lo usa en modo de 3 pines) .

La biblioteca NewPing es muy avanzada y mejora considerablemente la precisión de nuestro diseño original. También soporta hasta 15 sensores ultrasónicos a la vez y puede salir directamente en centímetros, pulgadas o duración.

Aquí está nuestro bosquejo reescrito para usar la biblioteca NewPing:

```
/*
  HC-SR04 NewPing Library Demonstration
  HC-SR04-NewPing.ino
  Demonstrates functions of NewPing Library for HC-SR04 U
  Displays results on Serial Monitor

  DroneBot Workshop 2017
  http://dronebotworkshop.com
*/

// This uses Serial Monitor to display Range Finder dista

// Include NewPing Library
#include "NewPing.h"

// Hook up HC-SR04 with Trig to Arduino Pin 10, Echo to A
// Maximum Distance is 400 cm

#define TRIGGER_PIN  10
#define ECHO_PIN     13
#define MAX_DISTANCE 400

```

```
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

float distance;

void setup() {
  Serial.begin (9600);
}

void loop() {

  distance = sonar.ping_cm();

  // Send results to Serial Monitor
  Serial.print("Distance = ");
  if (distance >= 400 || distance <= 2) {
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
    delay(500);
  }
  delay(500);
}
```

El dibujo anterior es simple y funciona bien, pero sólo tiene una resolución de hasta un centímetro. Si desea recuperar los valores del punto decimal, puede utilizar NewPing en modo de duración en lugar de en modo de distancia. A continuación, podemos utilizar la duración para calcular la distancia como lo hicimos en el primer bosquejo que observamos.

Aquí está nuestro bosquejo NewPing reescrito para usar la duración en lugar de la distancia.

```
/*
  HC-SR04 NewPing Duration Demonstration
  HC-SR04-NewPing-Duration.ino
  Demonstrates using Duration function of NewPing Library
  Displays results on Serial Monitor

  DroneBot Workshop 2017
  http://dronebotworkshop.com
*/

// This uses Serial Monitor to display Range Finder dista

// Include NewPing Library
#include "NewPing.h"

// Hook up HC-SR04 with Trig to Arduino Pin 10, Echo to A
// Maximum Distance is 400 cm

#define TRIGGER_PIN  10
#define ECHO_PIN     13
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

float duration, distance;

void setup() {
  Serial.begin (9600);
}

```

```
31 void loop() {
32
33   duration = sonar.ping();
34
35   // Determine distance from duration
36   // Use 343 metres per second as speed of sound
37
38   distance = (duration / 2) * 0.0343;
39
40   // Send results to Serial Monitor
41   Serial.print("Distance = ");
42   if (distance >= 400 || distance <= 2) {
43     Serial.println("Out of range");
44   }
45   else {
46     Serial.print(distance);
47     Serial.println(" cm");
48     delay(500);
49   }
50   delay(500);
51 }
```

Otra función de NewPing es "iteraciones" Para iterar significa pasar por encima de algo más de una vez, y eso es precisamente lo que hace el modo de iteración. Requiere muchas mediciones de duración en lugar de sólo una, descarta las lecturas no válidas y luego promedia las restantes. Por defecto toma 5 lecturas pero usted puede especificar realmente cuántas como usted desea.

Aquí vamos de nuevo con un bosquejo NewPing escrito para usar iteraciones. Como se puede ver, es virtualmente idéntico al boceto anterior, todo lo que se ha agregado es una variable para especificar el número de iteraciones.

```
1  /*
2    HC-SR04 NewPing Iteration Demonstration
3    HC-SR04-NewPing-Iteration.ino
4    Demonstrates using Iteration function of NewPing Librar
5    Displays results on Serial Monitor
6
7    DroneBot Workshop 2017
8    http://dronebotworkshop.com
9  */
10
11 // This uses Serial Monitor to display Range Finder dista
12
13 // Include NewPing Library
14 #include "NewPing.h"
15
16 // Hook up HC-SR04 with Trig to Arduino Pin 10, Echo to A
17 // Maximum Distance is 400 cm
18
19 #define TRIGGER_PIN  10
20 #define ECHO_PIN     13
21 #define MAX_DISTANCE 400
22
23 NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
24
25 float duration, distance;
26
27 int iterations = 5;
28
```

```
29 void setup() {
30   Serial.begin (9600);
31 }
32
33 void loop() {
34
35   duration = sonar.ping_median(iterations);
36
37   // Determine distance from duration
38   // Use 343 metres per second as speed of sound
39
40   distance = (duration / 2) * 0.0343;
41
42   // Send results to Serial Monitor
43   Serial.print("Distance = ");
44   if (distance >= 400 || distance <= 2) {
45     Serial.println("Out of range");
46   }
47   else {
48     Serial.print(distance);
49     Serial.println(" cm");
50     delay(500);
51   }
52   delay(500);
53 }
```

Mejora de la precisión

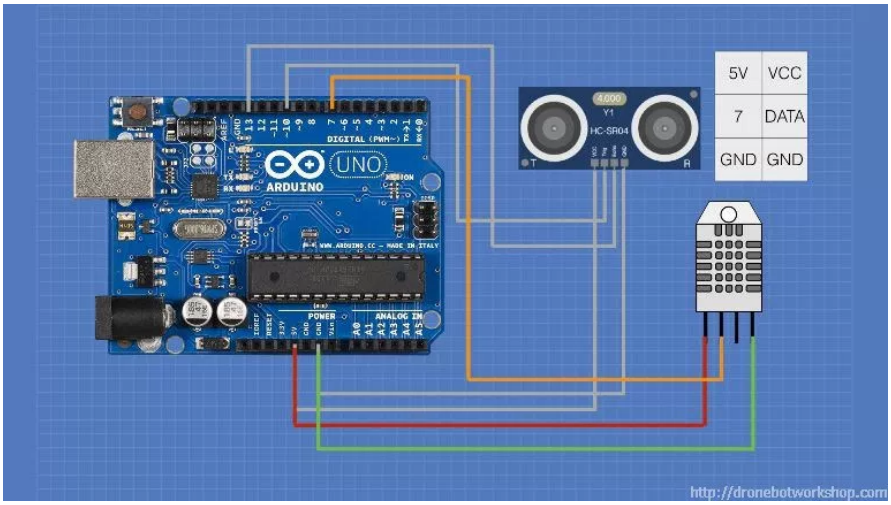
El HC-SR04 es razonablemente preciso, en su forma básica es muy útil para robots, detección de intrusos o alarmas de proximidad. Pero hay ocasiones en las que puede que desee un poco más de precisión, por ejemplo, puede estar construyendo una herramienta de medición o puede estar utilizando su robot para mapear el perímetro de una habitación. Si es así, hay algunas cosas que puede hacer para mejorar la precisión del HC-SR04.

Como mencioné en la última sección, la biblioteca NewPing ha implementado una serie de técnicas internas para mejorar la precisión del sensor. En la mayoría de los casos esto es todo lo que necesita para mejorar sus lecturas.

Si está diseñando un dispositivo que se va a utilizar al aire libre o en un ambiente inusualmente caliente o frío, es posible que desee tener en cuenta el hecho de que la velocidad del sonido en el aire varía con la temperatura, la presión del aire y la humedad. Dado que la velocidad de los factores de sonido en nuestro cálculo de distancia HC-SR04 esto podría afectar nuestras lecturas si la temperatura era mucho más caliente o más fría que la temperatura ambiente.

Con el fin de factor en la temperatura y la humedad, decidí utilizar el sensor DHT22, es relativamente barato, pero muy preciso. También podría usar el DHT-11 menos costoso para

hacer este experimento con una precisión ligeramente menor. Aquí está cómo lo enganché para arriba:



El DHT22 requiere un par de bibliotecas de código para que funcione, Adafruit tiene dos bibliotecas que funcionarán bien con DHT22 y DHT11. La biblioteca Adafruit AM2315 y la biblioteca Adafruit Unified Sensor se pueden instalar directamente en el IDE de Arduino utilizando el Administrador de bibliotecas.

Después de que instalé las bibliotecas de Adafruit escribí un bosquejo rápido de la prueba para estar seguro que mi DHT22 trabajaba correctamente.

```
/*
  DHT22 Basic Demonstration
  DHT22-Basic-Demo.ino
  Demonstrates functions of DHT22 Digital Temperature & H
  Displays results on Serial Monitor

  DroneBot Workshop 2017
  http://dronebotworkshop.com
*/

// Include DHT Libraries from Adafruit
// Dependant upon Adafruit_Sensors Library

#include "DHT.h";

// Define Constants

#define DHTPIN 7     // DHT-22 Output Pin connection
#define DHTTYPE DHT22   // DHT Type is DHT 22 (AM2302)

// Initialize DHT sensor for normal 16mhz Arduino

DHT dht(DHTPIN, DHTTYPE);


// Define Variables

float hum;  //Stores humidity value
float temp; //Stores temperature value

```

```
31 void setup()
32 {
33   Serial.begin(9600);
34   dht.begin();
35 }
36
37 void loop()
38 {
39     delay(2000);  // Delay so sensor can stabalize
40
41     hum = dht.readHumidity();  // Get Humidity value
42     temp= dht.readTemperature();  // Get Temperature valu
43
44     // Print temperature and humidity values to serial mo
45
46     Serial.print("Humidity: ");
47     Serial.print(hum);
48     Serial.print(" %, Temp: ");
49     Serial.print(temp);
50     Serial.println(" Celsius");
51
52 }
```

Y finalmente aquí está un bosquejo que los factores en temperatura y humedad para mejorar la exactitud usando el DHT22.

```
1  /*
2    HC-SR04 with Temp and Humidity Demonstration
3    HC-SR04-Temp-Humid-Demo.ino
4    Demonstrates enhancements of HC-SR04 Ultrasonic Range F
5    With DHT22 Temperature and Humidity Sensor
6    Displays results on Serial Monitor
7
8    DroneBot Workshop 2017
9    http://dronebotworkshop.com
10 */
11
12 // Include DHT Libraries from Adafruit
13 // Dependant upon Adafruit_Sensors Library
14 #include "DHT.h";
15
16 // Include NewPing Library
17 #include "NewPing.h"
18
19 // Define Constants
20
21 #define DHTPIN 7       // DHT-22 Output Pin connection
22 #define DHTTYPE DHT22   // DHT Type is DHT 22 (AM2302)
23 #define TRIGGER_PIN  10
24 #define ECHO_PIN     13
25 #define MAX_DISTANCE 400
26
27 NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
28
29 // Define Variables
30
31 float hum;    // Stores humidity value in percent
32 float temp;   // Stores temperature value in Celcius
33 float duration; // Stores HC-SR04 pulse duration value
34 float distance; // Stores calculated distance in cm
35 float soundsp;  // Stores calculated speed of sound in M/
36 float soundcm;  // Stores calculated speed of sound in cm
37 int iterations = 5;
38
39 // Initialize DHT sensor for normal 16mhz Arduino
40
41 DHT dht(DHTPIN, DHTTYPE);
```

```
42
43 void setup() {
44   Serial.begin (9600);
45   dht.begin();
46 }
47
48 void loop()
49 {
50
51   delay(2000);  // Delay so DHT-22 sensor can stabalize
52
53     hum = dht.readHumidity();  // Get Humidity value
54     temp= dht.readTemperature();  // Get Temperature valu
55
56     // Calculate the Speed of Sound in M/S
57     soundsp = 331.4 + (0.606 * temp) + (0.0124 * hum);
58
59     // Convert to cm/ms
60
61     soundcm = soundsp / 10000;
62
63   duration = sonar.ping_median(iterations);
64
65   // Calculate the distance
66   distance = (duration / 2) * soundcm;
67
68   // Send results to Serial Monitor
69
70     Serial.print("Sound: ");
71     Serial.print(soundsp);
72     Serial.print(" m/s, ");
73     Serial.print("Humid: ");
74     Serial.print(hum);
75     Serial.print(" %, Temp: ");
76     Serial.print(temp);
77     Serial.print(" C, ");
78     Serial.print("Distance: ");
79
80     if (distance >= 400 || distance <= 2) {
81     Serial.print("Out of range");
82     }
83     else {
84     Serial.print(distance);
85     Serial.print(" cm");
86     delay(500);
87     }
88
89   Serial.println(" ");
90 }
```

En mis pruebas en el banco de trabajo, encontré que efectivamente mejoraba la precisión de las lecturas.

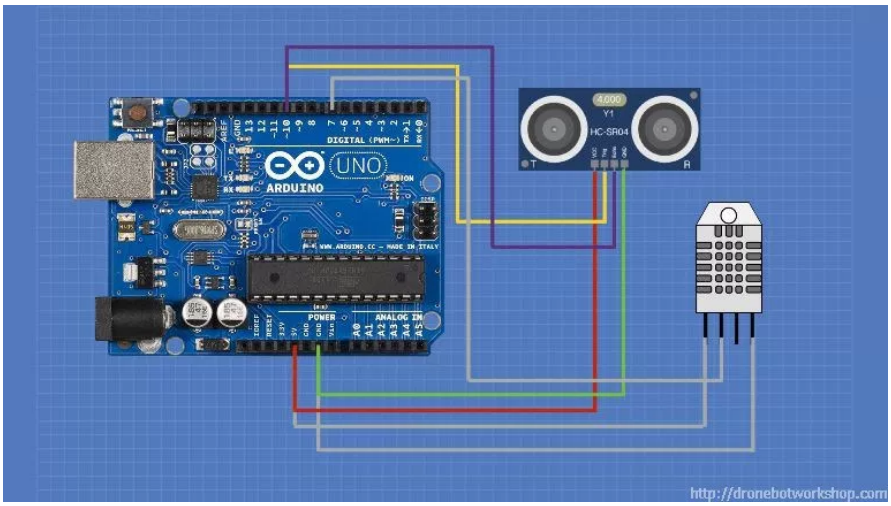
Uso del modo de 3 cables

Como ya he aludido antes, es posible usar el HC-SR04 en "Modo de 3 cables". En este modo sólo necesitará una conexión a un único pin de E / S digital de Arduino. Hay otros sensores de ultrasonidos que sólo funcionan en el modo de 3 cables.

En el modo de 3 hilos, el pin de E / S individual se utiliza como entrada y como salida. Esto es posible porque nunca hay un momento en el que se utilicen tanto la entrada como la salida. Al

eliminar un requisito de pin de E / S, podemos guardar una conexión a nuestro Arduino y usarla para otra cosa. También es útil cuando se utiliza un chip como el ATtiny85 que tiene un número limitado de pines de E / S.

Aquí es cómo conecté el HC-SR04 al Arduino.



Como se puede ver todo lo que hice fue conectar el gatillo y el eco a Arduino pin 10.

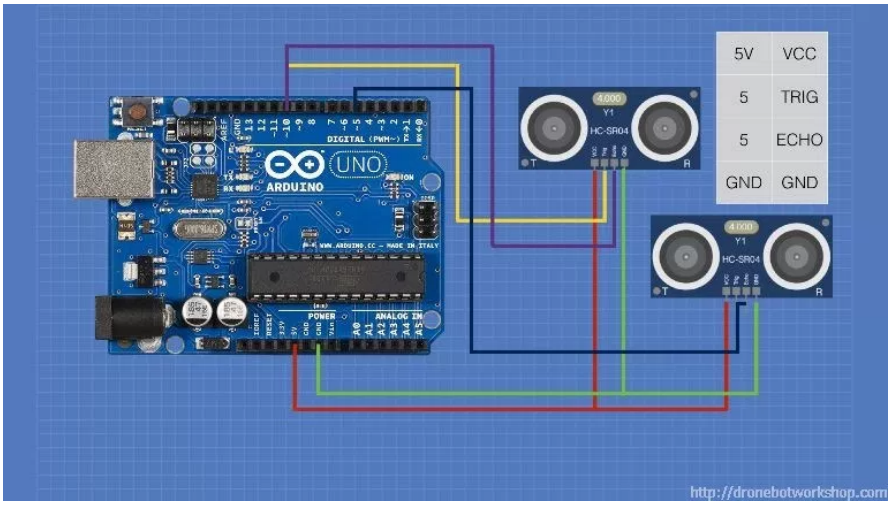
Y aquí está el bosquejo que escribí para usarlo. Tenga en cuenta que la única diferencia entre este boceto y el anterior es que he especificado pin 10 para los valores de gatillo y Echo pin. El resto del croquis es idéntico.

Uso de varios sensores HC-SR04

En muchas aplicaciones, querrá utilizar más de un sensor de ultrasonidos HC-SR04 en su diseño. Esto puede suceder cuando se desea supervisar la distancia a objetos externos desde diferentes lados de su robot o proyecto. Dos de ellos pueden ser usados para sensores delanteros y traseros, o conectar 6 de ellos y monitorear cada lado de un cubo - ¡depende de usted!

Cuando se utilizan varios sensores, una consideración obvia es que es necesario que la señal emitida por un sensor sea capturada y medida por otro sensor. La forma más sencilla de lograr esto es simplemente pulsar el disparador en un sensor y esperar hasta que reciba el eco antes de pasar al siguiente sensor. Podría ser prudente dejar un pequeño retraso entre las lecturas sólo en caso de que los sensores anteriores ondas de sonido siguen rebotando alrededor de la habitación!

Aquí es cómo usé dos HC-SR04 sensor ultrasónico con el Arduino. Tenga en cuenta que he cableado estos en el modo de 3 hilos, si lo desea también puede conectarlos en la moda convencional de 4 hilos. Si lo hace, simplemente modifique el boceto para especificar diferentes pines de gatillo y eco.



Aquí hay un boceto que utiliza la biblioteca NewPing y dos sensores. Como con todos los otros bocetos que los resultados de salida al monitor en serie.

```
/*
  Dual HC-SR04 with Temp and Humidity Demonstration
  HC-SR04-Temp-Humid-Dual-Demo.ino
  Demonstrates enhancements of HC-SR04 Ultrasonic Range
  With DHT22 Temperature and Humidity Sensor
  Displays results on Serial Monitor

  DroneBot Workshop 2017
  http://dronebotworkshop.com
*/

// Include DHT Libraries from Adafruit
// Dependant upon Adafruit_Sensors Library
#include "DHT.h";

// Include NewPing Library
#include "NewPing.h"

// Define Constants

#define DHTPIN 7       // DHT-22 Output Pin connection
#define DHTTYPE DHT22   // DHT Type is DHT 22 (AM2302)
#define TRIGGER_PIN_1  10
#define ECHO_PIN_1     10
#define TRIGGER_PIN_2  5
#define ECHO_PIN_2     5
#define MAX_DISTANCE 400

NewPing sonar1(TRIGGER_PIN_1, ECHO_PIN_1, MAX_DISTANCE);
NewPing sonar2(TRIGGER_PIN_2, ECHO_PIN_2, MAX_DISTANCE);

// Define Variables

float hum;    // Stores humidity value in percent
float temp;   // Stores temperature value in Celcius
float duration1; // Stores First HC-SR04 pulse duration
```

```
float duration2; // Stores Second HC-SR04 pulse duration
float distance1; // Stores calculated distance in cm for
float distance2; // Stores calculated distance in cm for
float soundsp;  // Stores calculated speed of sound in M
float soundcm;  // Stores calculated speed of sound in c
int iterations = 5;

// Initialize DHT sensor for normal 16mhz Arduino

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin (9600);
  dht.begin();
}

void loop()
{

  delay(1000);  // Delay so DHT-22 sensor can stabalize

    hum = dht.readHumidity();  // Get Humidity value
    temp= dht.readTemperature();  // Get Temperature val

    // Calculate the Speed of Sound in M/S
    soundsp = 331.4 + (0.606 * temp) + (0.0124 * hum);

    // Convert to cm/ms

    soundcm = soundsp / 10000;

    // Measure duration for first sensor

  duration1 = sonar1.ping_median(iterations);

  // Add a delay between sensor readings

  delay(1000);

  // Measure duration for first sensor

  duration2 = sonar2.ping_median(iterations);

  // Calculate the distances for both sensors

  distance1 = (duration1 / 2) * soundcm;
  distance2 = (duration2 / 2) * soundcm;

  // Send results to Serial Monitor

    Serial.print("Distance 1: ");

    if (distance1 >= 400 || distance1 <= 2) {
    Serial.print("Out of range");
    }
    else {
    Serial.print(distance1);
    Serial.print(" cm ");
    }

    Serial.print("Distance 2: ");

    if (distance2 >= 400 || distance2 <= 2) {
    Serial.print("Out of range");
    }
    else {
    Serial.print(distance2);
    Serial.print(" cm");
    }
```

```
106
107    Serial.println(" ");
108  }
```

¡Sea bueno con los animales!

Una cosa que tiene en cuenta al diseñar un dispositivo en torno a cualquier emisor de ultrasonidos, como el HC-SR04, es que muchos animales pueden escuchar el sonido ultrasónico. Según WikiPedia esto incluye perros, gatos, jerbos y conejos.

Si usted tiene peludos 4-legged amigos vagando por su casa es posible que desee abstenerse de someterlos al sonido emitido por el sensor, ya que podría ser muy molesto para ellos. Imagine oír una serie aguda de "pitidos" cada segundo sin parar y usted puede simpatizar con lo que su mascota podría estar pasando.

Hay otros métodos de medir la distancia como la luz del IR y LIDAR que no implican sonido ultrasónico, así que usted puede ser que desee mirar en eso si los animales son una consideración en su hogar. ¡Por lo menos es probablemente una buena idea en general mantener Rover o Fifi fuera del taller!

Moviéndose

Como ya has visto, el Sensor de Distancia Ultrasónica HC-SR04 es un dispositivo económico pero útil que se puede utilizar para una gran variedad de aplicaciones. Si usted está en robots de construcción esto es un componente esencial en su juego de herramientas.

Esperamos que haya encontrado este artículo útil. Si tiene alguna pregunta sobre el HC-SR04 o los bocetos que he presentado aquí, por favor escríbalos en la sección de comentarios de abajo. Y si usted viene con un proyecto basado en el HC-SR04 me encantaría saberlo.

Así que vamos a tener la detección - yo te ping más tarde!

Recursos

HC-SR04 Esquemas Todos los bocetos utilizados en este artículo.

Arduino Sitio NewPing Artículo La biblioteca NewPing como se describe en el sitio web oficial de Arduino.

Biblioteca de NewPing en GitHub El repositorio de NewPing Library en GitHub. Ve aquí para la última versión.

Biblioteca Adafruit DHT22 La biblioteca Adafruit para DHT22 en GitHub. También puede instalarlo desde su Administrador de biblioteca en el IDE de Arduino.

Adafruit Unified Sensor Library La biblioteca Adafruit Unified Sensor en GitHub. Una vez más, sólo puede instalarlo desde su Administrador de biblioteca.

Compartir este:

Relacionado

Introducción Robots de construcción
6 de febrero de 2016
En "Robots"

Arduino 37 Sensores Parte 1 - Descripción general
27 de febrero de 2016
En "Arduino"

Conoce a la familia Arduino
17 de febrero 2016
En "Arduino"

Taller DroneBot 29 de julio de 2017 Arduino

No hay comentarios

← Control de Motores DC con el L-298N Dual H-Bridge y un Arduino

Deja una respuesta

Su dirección de correo electrónico no será publicada. Los campos obligatorios están marcados *

Comentario

Nombre *

Correo electrónico *

Sitio web

publicar comentario

Notifíqueme de comentarios consecuentes por email.

Notifícame de nuevas entradas por e-mail.

Proyectos

Proyectos de Arduino

Tutoriales

Tutoriales de Arduino

Qué hay de nuevo

Conexiones del taller

Proyectos Drone
Proyectos Electrónicos
Proyectos de Internet de Cosas
Raspberry Pi Proyectos
Proyectos de robots

Tutoriales de Drone
Tutoriales de electrónica
Tutoriales de Internet de Cosas
Tutoriales de frambuesa Pi
Tutoriales de Robots
Tutoriales de software

Uso del sensor de distancia ultrasónico HC-SR04 con Arduino
Soldar el Conector GPO Pi Zero de la Frambuesa
Control de Motores DC con el L-298N Dual H-Bridge y un Arduino
Montaje del brazo robótico MeArm
Arduino 37 Sensores Parte 1 - Información general
Conoce a la Familia Arduino
No obtenga Zapped por electricidad estática!

Casa Arduino Frambuesa Pi Drones Robots Electrónica IoT Comentarios Tutoriales Sobre nosotros Contáctenos Privacidad y Cookies