07a Ultrasonic Sensing Introduction

Nicholas Bruzzese

January 5, 2025

Objective

Learn how to measure distances using the HC-SR04 ultrasonic sensor with a Raspberry Pi. This guide will introduce the sensor's workings, the necessary setup, and the Python code required to get it operational.

What is the HC-SR04 Ultrasonic Sensor?

The HC-SR04 is an ultrasonic sensor that measures distance using sound waves. It calculates the time taken by a sound pulse to travel to an object and reflect back.

Key Features:

- **Trigger Pin:** Initiates the sound pulse.
- **Echo Pin:** Receives the reflected sound pulse.
- **Range:** 2 cm to 400 cm.
- **Accuracy:** ± 3 mm.

How Ultrasonic Sensing Works

1. The sensor emits an ultrasonic pulse from the Trigger pin.
2. The pulse travels until it hits an object.
3. It bounces back to the Echo pin.
4. The time between sending and receiving is used to calculate distance:

$$\text{Distance (cm)} = \frac{\text{Time (s)} \times 34300}{2}$$

5. 34300 cm/s is the speed of sound.

Hardware Required

- Raspberry Pi (e.g., Raspberry Pi 4 Model B or similar).
- HC-SR04 Ultrasonic Sensor.
- Breadboard.
- Jumper Wires.
- 330Ω and 470Ω resistors for level shifting.

Setup and Circuit Diagram

Connections:

- HC-SR04 Trigger → Raspberry Pi GPIO18 (Physical Pin 12).
- HC-SR04 Echo → Voltage Divider → Raspberry Pi GPIO24 (Physical Pin 18).
- VCC → 5V Power on Raspberry Pi.
- GND → Ground on Raspberry Pi.

Wiring Diagram

This is a simple setup:

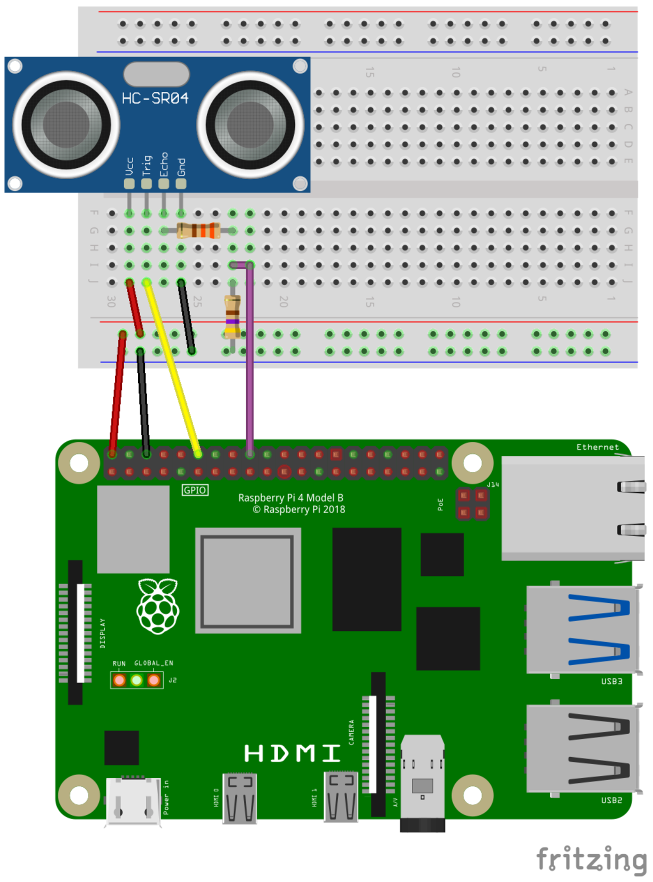

Figure 1: Wiring Diagram

Voltage Divider for Echo Pin:

- Use a resistor voltage divider: Connect a 470Ω resistor between the Echo pin and ground, and a 330Ω resistor between the Echo pin and GPIO24.

Python Code to Measure Distance

Here's the Python script for the project. The code uses the `RPi.GPIO` library to interact with the GPIO pins.

```
#Libraries
import RPi.GPIO as GPIO
import time

#GPIO Mode (BOARD / BCM)
GPIO.setmode(GPIO.BCM)

#set GPIO Pins
GPIO_TRIGGER = 18
GPIO_ECHO = 24

#set GPIO direction (IN / OUT)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

def distance():
   # set Trigger to HIGH
   GPIO.output(GPIO_TRIGGER, True)

   # set Trigger after 0.01ms to LOW
   time.sleep(0.00001)
   GPIO.output(GPIO_TRIGGER, False)

   StartTime = time.time()
   StopTime = time.time()

   # save StartTime
   while GPIO.input(GPIO_ECHO) == 0:
   StartTime = time.time()

   # save time of arrival
   while GPIO.input(GPIO_ECHO) == 1:
   StopTime = time.time()

   # time difference between start and arrival
   TimeElapsed = StopTime - StartTime
   # multiply with the sonic speed (34300 cm/s)
   # and divide by 2, because there and back
   distance = (TimeElapsed * 34300) / 2

   return distance

if __name__ == '__main__':
   try:
      while True:
           dist = distance()
           print ("Measured Distance = %.1f cm" % dist)
           time.sleep(1)

   # Reset by pressing CTRL + C
   except KeyboardInterrupt:
       print("Measurement stopped by User")
       GPIO.cleanup()
```

Running the Code

1. Save the script as `ultrasonic_distance.py`.
2. Run the script in your terminal:

   ```
   python3 ultrasonic_distance.py
   ```

Expected Output

The terminal will display the measured distance in centimeters, updating every second:

```
Measured Distance = 35.4 cm
Measured Distance = 36.0 cm
...
```

Troubleshooting Tips

- **No Output:**
  - Check GPIO connections and ensure the voltage divider is correctly set up.
- **Incorrect Readings:**
  - Ensure no objects obstruct the sensor.
  - Verify the HC-SR04 is not angled.

Applications

- Obstacle detection for robots.
- Parking assistance systems.
- Proximity alarms.