

SOUTHERN ILLINOIS UNIVERSITY  
CARBONDALE

---

ECE 296L-002 Intro to Software & Robotics Lab

## Lab 2: Ultrasonic Sensor Distance Detection Circuit

Performed by:

Chase Lotito  
856093429

Instructor: Dr. James Phegley  
TA: Muhammad Zia Hameed

February 15, 2024

## Table of Contents

Introduction.....	3
Assessment of Design .....	3
Conclusion .....	4
References.....	6
Appendix A: Hardware Schematic .....	7
Appendix B: Code for the Software Developed .....	8

## Introduction

The goal of this experiment is to set up a circuit that can detect the distance of an object from itself. We will later use this circuit to track the distance of several objects at different angles from our robot.

We can achieve range detection with the ultrasonic sensor that comes in the kit with our Raspberry Pi's. The device works by sending out a chirp (ultrasonic soundwave pulse) which will reflect off an object, and the sensor will detect when the chirp comes back.

We can use python logic to determine the transmit time and receive time, then calculate the distance using simple kinematics.

## Assessment of Design

The design is inspired and guided by the design provided by Dr. James Phegley in the Lab 2 manual [1].

The physical circuit is implemented as shown in Figure 1. We have the sensor's echo pin connected to pin 20 on the Raspberry Pi,  $V_{cc}$  connected to  $+5V_{DC}$ , GND connected to GND, and the trigger pin is connected to a resistive network coming out of pin 16.

The ultrasonic sensor can only handle  $3.3V_{DC}$  and the GPIO pins on the Raspberry Pi are configured to output  $5V_{DC}$ . So, connect three  $10k\Omega$ -resistors in series, which creates a voltage divider that has our desired voltage of  $3.3V_{DC}$  at the node of the second resistor---this is shown in detail in Appendix A.

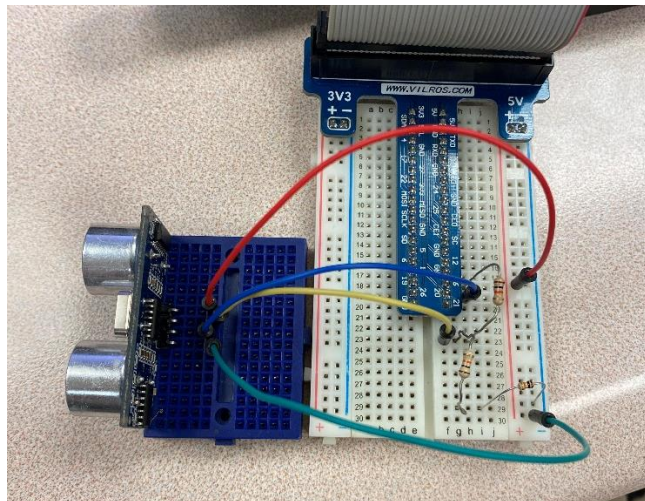


Figure 1: Ultrasonic Sensor Circuit

In python, we have our logic living inside a forever while loop, unless a keyboard interrupt triggers us to move to the finally block. Inside that code, we send a pulse from the Raspberry Pi to the ultrasonic sensor's trigger pin, which is our way of telling the sensor to send a chirp.

We start then start a while loop that repeats until the echo pin goes high sending our chirp. Inside this loop we are continuously logging the time, and the logged time when the block breaks is the transmit time  $t_{tx}$ .

Once this happens, we enter a second loop that waits for the chirp to be received, which happens when our echo pin falls back to a false state. Again, we are continuously logging the time, and when the loop breaks the time that is logged is the receive time  $t_{rx}$ .

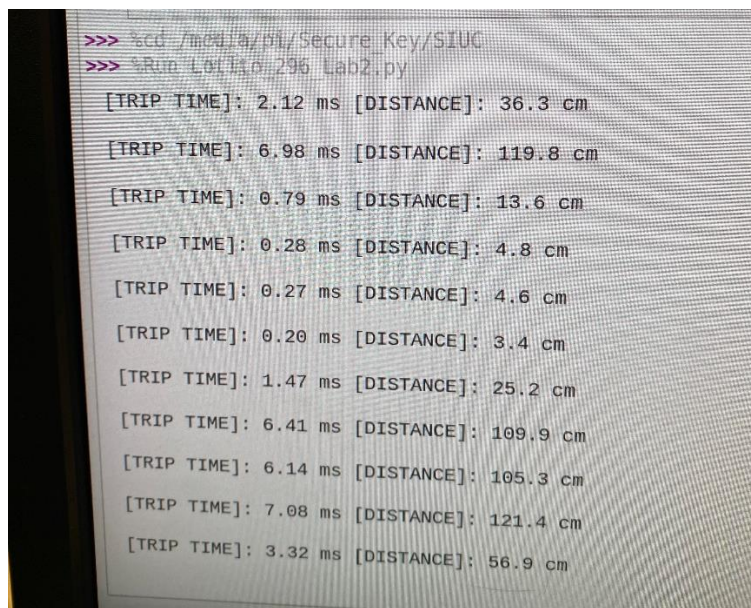
The `time()` function returns the time since the epoch, which is not useful unless we take the difference between the transit time and receive time to get the total trip time.

$$t_{trip} = t_{rx} - t_{tx}$$

From this we can use the fact that the speed of sound in air is 343 m/s to calculate the distance the object is from the ultrasonic sensor---remembering that we only need one half of this distance as the trip time accounts for travel there and back.

$$distance = 343(t_{trip} * 0.5)$$

Then we multiply this value by 100 to convert from meters to centimeters. We also multiply the  $t_{trip}$  by 1000 to convert from seconds to milliseconds. These values are then outputted to the console to verify the design works, show in Figure 2.



```
>>> %cd /media/pi/Secure_Key/SIUC
>>> %Run Lab2.py
[TRIP TIME]: 2.12 ms [DISTANCE]: 36.3 cm
[TRIP TIME]: 6.98 ms [DISTANCE]: 119.8 cm
[TRIP TIME]: 0.79 ms [DISTANCE]: 13.6 cm
[TRIP TIME]: 0.28 ms [DISTANCE]: 4.8 cm
[TRIP TIME]: 0.27 ms [DISTANCE]: 4.6 cm
[TRIP TIME]: 0.20 ms [DISTANCE]: 3.4 cm
[TRIP TIME]: 1.47 ms [DISTANCE]: 25.2 cm
[TRIP TIME]: 6.41 ms [DISTANCE]: 109.9 cm
[TRIP TIME]: 6.14 ms [DISTANCE]: 105.3 cm
[TRIP TIME]: 7.08 ms [DISTANCE]: 121.4 cm
[TRIP TIME]: 3.32 ms [DISTANCE]: 56.9 cm
```

Figure 2: Console Log from Circuit

## Conclusion

This lab was the foundation before integrating the ultrasonic sensor into a object-tracking robot. I learned a lot in how to deal with the quirks of the device, like the chirp being sent out at a random time. But I also needed to bring my knowledge of soundwaves and simple math to

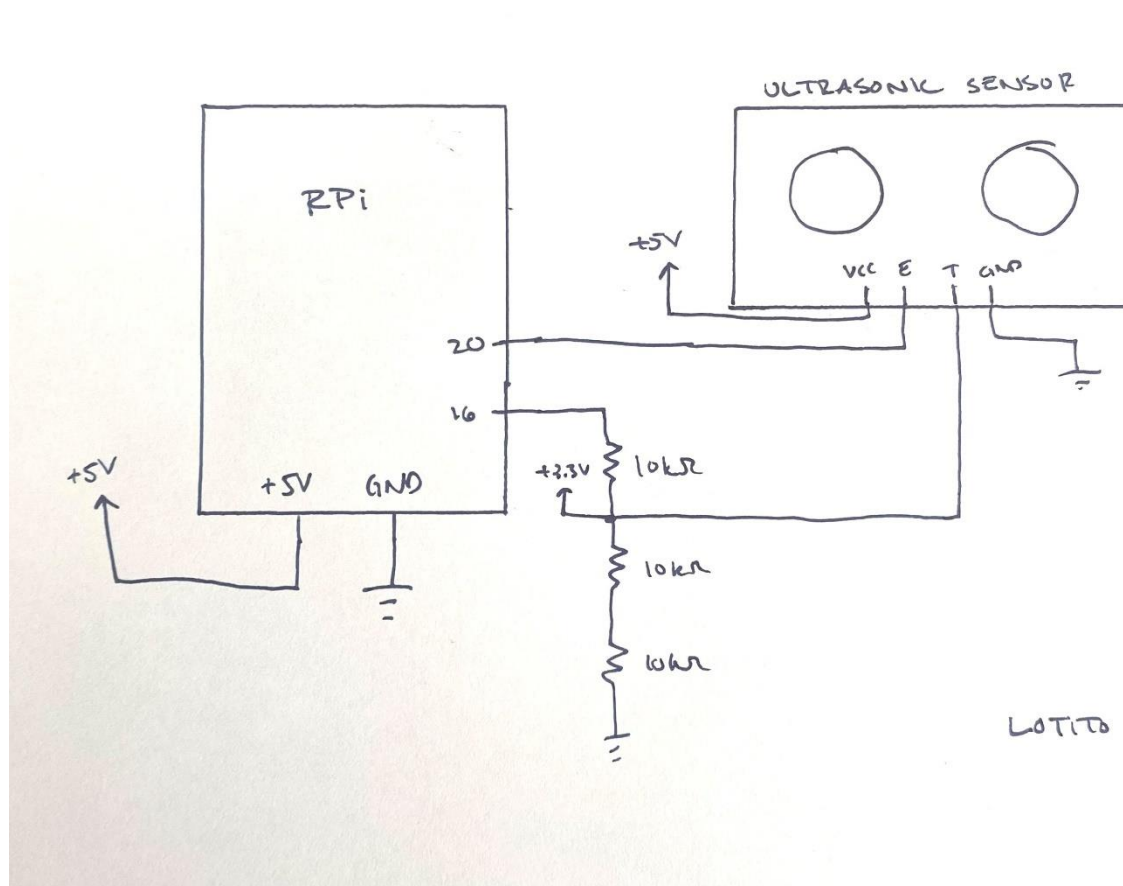
calculate for the proper times and distances with the appropriate values. At the end, the circuit was able to accurately write out the distance of an object from itself.

In regard to python, this project reinforced while loops.

## References

- [1] J. Phegley, "Raspberry Pi Lab Number 2," SIUC ECE Dept, Carbondale, 2024.

## Appendix A: Hardware Schematic



## Appendix B: Code for the Software Developed

```
'''
Chase Lotito - SIUC - SP2024
ECE296L - Dr. James Phegley
RPi Lab 2 - Code from Phegley
Ultrasonic Sensor
'''

import RPi.GPIO as GPIO
from time import sleep, time # time() returns time in sec since EPOCH

# RPi Pin Setups
TRIG_PIN = 16
ECHO_PIN = 20

GPIO.setmode(GPIO.BCM) # BROADCOM PINS
GPIO.setup(TRIG_PIN, GPIO.OUT)
GPIO.setup(ECHO_PIN, GPIO.IN)

try:
    while True:
        # Toggle the Trigger pin
        GPIO.output(TRIG_PIN, True)
        sleep(0.1)
        GPIO.output(TRIG_PIN, False)

        # Detect when ECHO CHIRP is sent
        while GPIO.input(ECHO_PIN) == False:
            t_tx = time()
        # Detect when ECHO CHIRP is recieved
        while GPIO.input(ECHO_PIN) == True:
            t_rx = time()

        trip_time = t_rx - t_tx
        distance = 343 * (trip_time / 2) * 100 # * 100 to convert m -> cm
        sleep(1)
        print('[TRIP TIME]: %.2f ms [DISTANCE]: %.1f cm \n' % (trip_time * 1000,
distance))
finally:
    GPIO.cleanup()
    print('PROGRAM TERMINATED')
```