**Year 10**

**Student Research Project**

14

**St Joseph’s College Hunters Hill**

The Affect of Temperature and Relative Humidity (and Air Pressure) on the Observed Distance by a Sonar Distance Sensor.

Jordan Joseph Lewis

This document is the full report on my research and experimentation on the affect of temperature and relative humidity on the accuracy of a sonar distance sensor, using a Raspberry Pi.

08

**Fall**

Table of Contents

Topic Selection 4

Introduction 4

Background Research 4

The Basics of Sound 4

The Sound Wave 4

The Speed of Sound 5

The Speed of Sound in Numbers 5

Sound through Air 5

Other Factors that Affect the Speed of Sound through Air 5

Aim 6

Hypothesis 6

Equipment 7

Method 8

Notes – Must Read Before Proceeding 8

Do the Following in Order 8

Install Raspbian 9

Connect to the Raspberry Pi Remotely 10

Reboot / Shutdown the Raspberry Pi 10

To Reboot 10

To Shutdown 10

Update and Upgrade the Raspberry Pi 10

Set The Date 10

Install TightVNC and Set It Up 11

Start the TightVNC Server and Connect To It 11

To Start the VNC Server 11

To Connect to the VNC Server 11

Launch the File Manager 11

Copy the Project Files to the Raspberry Pi 11

Install the MySQL Server and Python Connector 12

Install Libraries and Dependencies + Some Configuration 12

Login to MySQL 12

Create a Database and Table 13

Empty Data in Table 13

Wire Up the Electronics 13

Test if the Electronics Work 13

Starting the Program on Boot 13

Let the Program Run 14

How to Check into the Database 14

Export the Data in the Database as CSV and Copy to USB 15

Graph the Data 16

Variables 17

Independent Variable(s) 17

Dependent Variable(s) 17

Controlled Variable(s) 17

Risk Assessment 17

Diagram(s) 18

Results 19

Observations 19

How The Observations Were Made 19

The Observations (No Pressure) 19

The Observations (With Pressure) 19

Discussion 20

Overview 20

How My Program Works 20

Other Important Notes 20

Results 20

Deployments 1 through to 4 (Without Air Pressure) 20

Deployment 1 20

Deployment 2 21

Deployment 3 21

Deployment 4 21

Summarizing Deployments 1 through to 4 21

Deployments 5 thought to 8 (With Air Pressure) 21

Deployment 5 22

Deployment 6 22

Deployment 7 22

Deployment 8 22

Summarizing Deployments 5 through to 8 22

The Reliability of the Results 23

Future Experiments 23

Have all of the sensors from the beginning 23

Have an accurate set distance 23

Use a flat wall 23

Sonar distance sensor should be parallel to ground and perpendicular to wall 23

Use a different Temperature and Humidity Sensor, or fix the program 24

Have the program run for longer 24

Have a Control 24

Conclusion 25

Goals/Aims 25

Results 25

What I was looking for 25

What my results tell me 25

Without Air Pressure (Deployments 1-4) 25

With Air Pressure (Deployments 5-8, observed distances are ignored in 5) 25

Deployment 5 25

Deployment 6-8 25

The Conclusion on the Results 26

Social and Scientific Significance – Real World Uses 26

The Verdict 26

Bibliography 27

# Topic Selection

The topic of my research task is the affect of temperature and its relative humidity on the observed distance of a sonar distance sensor. This topic is a mainly electronics and programming based experiment and will be required to run for extended periods of time to collect a lot of data on the atmosphere and observed distance by the sonar distance sensor. This experiment is significant in both the scientific and social, as sonar distance sensors are used everywhere from small DIY robots to large-scale projects. This topic has its feet set in the physics department of science and will most likely show patterns based on the background research. This experiment, to the best of my knowledge and some research on the Internet has not been done the way I plan to do it, so all/most of the code and method are my own brainchildren. This should be good fun.

# Introduction

The speed of sound changes as the density of a material changes, in a way that as the medium gets denser; the speed of sound will increase. In this research project, I collected large amounts of data using several sensors and a Raspberry Pi. I focused on the affect of temperature and its relative humidity on the accuracy or the change in measurements of a stationary sonar distance sensor. This data included sonar (distance) measurements, temperature (Celsius) measurements, and humidity (%) measurements. The above data was used to create graphs to visualize the change in the sonar distance measurements as time goes on through the day and as temperature and humidity changes.

# Background Research

## The Basics of Sound

Sound is a wave caused by a disturbance in the medium such as air, often caused by your vocal chords or speakers. Sound travels by air molecules colliding into each other and transferring energy as they collide. Ultimately the wave transfers energy and not matter, so it is not the air molecules travelling themselves but rather the sound energy.

## The Sound Wave

In a gas or liquid, sound travels in compression waves. Compression waves travel in something like a sweeping motion, colliding particles into other particles near or in front of it to transfer energy. Due to sound travelling in compression waves, the density and the rigidity affects how fast and how far sound can travel in that medium. (Wikipedia 2014)

## The Speed of Sound

When talking about the speed of sound, you are talking about the amount of time it takes for the energy from the initial disturbance to a certain distance, say from the vibrations of your vocal chords to your ears. The speed of sound is determined by the medium or material it travels through. There are two properties that determine the speed of sound, or how rapidly sound is transferred, which are the density of the medium, and its rigidity or elastic modulus. (Brennan 2014)

### The Speed of Sound in Numbers

The speed of sound is calculated to be about 343metres/second (1125feet/second) in dry air at 20 degrees Celsius (68 degrees Fahrenheit). This is 1,234 kilometers per hour (666 kn; 767 mph), or about a kilometer in three seconds or a mile in five seconds. (Wikipedia 2014)

## Sound through Air

Humidity does not affect the rigidity of air, but rather affects the density of air. As the humidity increases, so too does the ratio/percentage of air molecules there are to water molecules. Water molecules are smaller than oxygen, nitrogen, or carbon dioxide molecules, which means that the more water molecules there are in the air, the less dense the air becomes. Sound travels faster through lower densities than high-density mediums, which translates to sound travelling faster in high humidity air. The increase in the speed of sound through air is very minimal though air with a higher humidity. In room temperature air at sea level, for example, sound travels about 0.35 percent faster in 100 percent humidity (very humid air) than it does in 0 percent humidity (completely dry air). (Brennan 2014)

## Other Factors that Affect the Speed of Sound through Air

The effect of humidity on the speed of sound is slightly greater in lower air pressures, like what you experience in higher altitudes. At about 6,000 meters (20,000 feet) above sea level, the difference between the speed of sound in room temperature dry air at 0 percent humidity and the same air at 100 percent humidity is about 0.7 percent. Increasing temperature also magnifies the effect of humidity on the speed of sound in air, although again the increase is relatively modest.

# Aim

To observe the affect of changing temperature and relative humidity on the distance measurements of an sonar distance sensor using a Raspberry Pi as the controller assuming that sound travels at 340m/s. Then, if possible find a formula to return more accurate / consistent sonar distance measurements.

# Hypothesis

I think that as the temperature rises the sonar distance sensor will return shorter distances because when the temperature rises the air becomes less dense meaning sound would travel faster through it, hence returning a shorter distance measurement.

# Equipment

1 \* Laptop with an SSH Client Software Installed

1 \* Raspberry Pi Model B

1 \* 5v 2.0A USB Power Supply

1 \* USB to Micro USB Cable

1 \* SD Card (Minimum 8GB)

1 \* 640 hole breadboard (300 holes will do)

5 \* Resistors

1 \* 640 Ohm 1%

1\* 1k Ohm 1%

3\* 2k Ohm 1%

1 \* DHT22 Sensor

1 \* BMP180 (Pressure Sensor)

1 \* HC-SR04 (Sonar Distance Sensor)

1 \* Wi-Fi Router

12 \* Male to Female Leads

3 \* Red

2 \* Orange (1 can be Green)

2 \* Brown

1 \* Green

2 \* Yellow

2 \* Blue (1 can be Yellow)

2 \* Female to Female Leads

1 \* Yellow

1 \* Green

1 \* RJ-45 Cable (Ethernet Cable)

2 \* 3mm LEDs

1\* Red

1 \* Yellow

17 \* Bridges

Internet Access

# Method

## Notes – Must Read Before Proceeding

Jordan Lewis does not take any responsibility for damaged electronics or any other form of damaged caused by doing this experiment.

There are several images attached inside the Method Help folder inside the Images folder inside the Project Resources folder. Have a look before proceeding.

You MUST remember the following before continuing

All code that must be typed in will be in italics.

e.g. ssh pi@192.168.1.3

Any phrase inside <> are instructions for the specific command

e.g. ssh pi@<enter the Raspberry Pi’s IP Address>

You would put something like 192.168.1.3 where the <> are.

It should become ssh pi@192.168.1.3

Console Output will be in bold.

e.g. **mysql>**

or **password>**

Any instruction given to you will have an underline.

e.g. Wait 1 minute

or Now do something

Comments or notes meant for you will be placed inside 3 dashes.

e.g. --- Files are provided in the Resources Folder ---

## Do the Following in Order

1. Install Raspbian
2. Connect to the Raspberry Pi Remotely
3. Update and Upgrade the Raspberry Pi
4. Install TightVNC and Set It Up
5. Start the TightVNC Server and Connect To It
6. Launch the File Manager
7. Copy the Project Files to the Raspberry Pi
8. Install the MySQL Server and Python Connector
9. Install Libraries and Dependencies + Some Configuration
10. Connect to the Raspberry Pi Remotely
11. Login to MySQL
12. Create a Database and Table
13. Wire up the Electronics
14. Test if the Electronics Work
15. Starting the Program on Boot
16. Let the Program Run
17. Check into the database (Optional)
18. Export the Data in the Database as CSV and Copy to USB
19. Graph the Data

## Install Raspbian

You can use the following steps below or use this website: <https://www.andrewmunsell.com/blog/getting-started-raspberry-pi-install-raspbian>

1. Download the latest Raspbian from either of the links below.
   1. [http://downloads.raspberrypi.org/raspbian\_latest](%22)
   2. [http://www.raspberrypi.org/downloads/](%22)
2. Connect the SD Card to your computer
3. Launch Disk Utility in /Applications/Utilities/ then do the following below
   1. Select the SD Card
   2. Click the Partition tab
   3. Under Partition Layout click the dropdown menu and select ‘1 Partition’.
   4. Set the format of the partition to MS-DOS (FAT)
   5. Click Apply and Yes
   6. Once the partitioning process finishes click on the child section of the SD Card (probably named UNTITLED 1)
   7. Click on the First Aid tab
   8. Click the Verify Disk button
   9. Once the process finishes, there should be text on the window that includes something similar to /dev/rdisk2s1. Take note of the number after disk and before s, in the above case it would be 2.
   10. Now click on the Unmount button at the top of the window to unmount the SD Card.
4. Launch Terminal in /Applications/Utilities/ then do the following
   1. Type the following

sudo dd bs=1m if=<drag the raspbian img file here, or path to Raspbian image> of=/dev/diskn

--- The n after disk must be replaced with the number noted down above ---

* 1. It should prompt you for your password

**Password:** <enter your password (must be an admin on the computer)>

1. Once installed you should get something like

**2825+0 records in**

**2825+0 records out**

**2962227200 bytes transferred in 1057.961051 secs ( 2799940 bytes/sec)**

The SD Card should also appear on your desktop with the name boot.

1. Eject the SD Card
2. Now plug in the SD Card into the Raspberry Pi
3. Connect the following in order to the Raspberry Pi
   1. HDMI Cable, which is connected to a display
   2. USB Keyboard
   3. USB Mouse
   4. Ethernet Cable (Connect to the Internet) (Optional at the moment)
   5. Micro USB cable (connected to a 5 Volt 2.0 Amp USB Power Supply)
4. If everything worked properly the screen should start showing scrolling text as the Raspberry Pi boots up. Once it has finished booting up for the first time, it will show you a screen with a blue background and options. Here do the following below; this will all be done using the USB Keyboard.
   1. Select the first ‘Expand Filesystem’ option and press enter.
   2. Select the eight ‘Adavanced Options’ option and press enter.
      1. Select the fourth ‘SSH’ option and press enter.
      2. Enable SSH
   3. Select finish and reboot
5. If everything worked properly it should reboot and show the scrolling text again, this time at the end it will show **raspberrypi login:** now do the steps below
   1. Type in pi and press enter
   2. It will prompt you to enter the password like **Password:** enter raspberry as the password (The password will not appear as you are typing it for security reasons, just be very sure of what you type in.) and press enter. --- Note: It will be a good idea to remember the user name and password ‘pi’ and ‘raspberry’ respectively.
6. If you have logged in successfully, the bottom of the screen should now show **pi@raspberrypi ~ $**. --- From here you will enter a lot of the commands required, the steps for logging in are the same whether it is wirelessly or on the Raspberry Pi itself.

## Connect to the Raspberry Pi Remotely

1. Launch a SSH Client on your Computer
2. Connect to the same network as the Raspberry Pi
3. Type into the terminal ssh pi@<enter the Raspberry Pi’s IP Address>
4. **password>** raspberry

## Reboot / Shutdown the Raspberry Pi

### To Reboot

Type in: sudo reboot

### To Shutdown

Type in: sudo shutdown now

## Update and Upgrade the Raspberry Pi

1. Type in: sudo apt-get update
2. Type in the following when the previous has finished*:* sudo apt-get upgrade
3. Enter y or yes when required

## Set The Date

Type in: sudo date --set “dd/MMMM/yyyy HH:mm:ss”

--- Where dd is the date (15), MMMM is the month in letters (August), yyyy is the year (2014), HH is the hour in 24 hour time (14), mm is minutes (35), ss is seconds (57).

The above would be sudo date --set “15/August/2014 14:35:57”

## Install TightVNC and Set It Up

1. Type in: sudo apt-get install tightvncserver
2. Enter y or yes when required
3. Type in: tightvncserver
4. Type in: tightvnc as the password
5. Verify the password
6. Type in: nano startvnc.sh
7. Type in the following into the terminal:

#!/bin/sh

vncserver :0 -geometry 1920x1080 -depth 24 -dpi 96

1. Press Control-Out, then Enter, then Control-X
2. Type in: chmod +x startvnc.sh

## Start the TightVNC Server and Connect To It

### To Start the VNC Server

Type in: ./startvnc.sh

### To Connect to the VNC Server

1. Type in the IP Address of the Raspberry Pi as the server.
2. Type in: tightvnc as the password when prompted

## Launch the File Manager

When on the desktop of the Raspberry Pi, at the bottom, there should be something like a dock. Click on the item at the bottom right to reveal a menu. Then go to Accessories and open File Manager.

## Copy the Project Files to the Raspberry Pi

1. Copy the project files to a USB
2. Plug the USB into the Raspberry Pi (the Raspberry Pi may reboot when the USB is plugged in, wait until it finishes.)
3. Login to the Raspberry Pi
4. Start the TightVNC Server and Connect to it
5. Launch the File Manager
6. Select the USB and copy the contents of the Raspberry Pi Files folder to a folder named pi (otherwise known as the home folder of the user pi). Now the pi folder should have the following plus other system files:
   1. Adafruit\_DHT\_Driver\_Python (Folder)
   2. Adafruit\_Python\_BMP-master (Folder)
   3. collectdata.py (File)
   4. Tests (Folder)

Eject the USB and unplug it (The Raspberry Pi may reboot when you physically unplug the USB).

## Install the MySQL Server and Python Connector

1. Type in the following when the previous has finished: sudo apt-get install mysql-server python-mysqldb
2. Enter y or yes when required
3. When greeted with a blue screen prompting you to enter a password, enter *raspberry*
4. Re-enter the raspberry when required

## Install Libraries and Dependencies + Some Configuration

1. Type in: sudo apt-get install python-smbus
2. Enter y or yes when required
3. Type in: sudo apt-get install i2c-tools
4. Type in: sudo nano-/etc/modprobe.d/raspi-blacklist.conf
5. The screen should now show something like what the below.

**#blacklist spi and i2c by default (many users don’t need them)**

**blacklist spi-bcm2708**

**blacklist i2c-bcm2708**

Add #s in front of the last 2 lines to make it look like

**#blacklist spi and i2c by default (many users don’t need them)**

**#blacklist spi-bcm2708**

**#blacklist i2c-bcm2708**

1. Press Control-Out, then Enter, then Control-X
2. Type in: sudo nano /etc/modules
3. Check for the following lines in the file, otherwise add at the end:

i2c-bcm2708

i2c-dev

1. Once the lines are there press Control-Out, then Enter, then Control-X
2. Type in: cd Adafruit\_Python\_BMP-master
3. Type in: sudo python setup.py install
4. Type in: cd
5. Type in: sudo apt-get install python-dev
6. Type in: sudo apt-get install python-rpi.gpio
7. Type in: sudo apt-get install git-core build-essential
8. Type in: cd Adafruit\_Python\_BMP-master
9. Type in: sudo python setup.py install
10. Type in: cd
11. Type in: sudo reboot

## Login to MySQL

Type in:mysql –u root –p

**password>**raspberry

**mysql>** <any SQL commands will be entered here>

--- To leave the MySQL terminal, type in *exit* into the terminal ---

## Create a Database and Table

1. Login to MySQL
2. Type in create database <database name>;

Replace <database name> with a database name, in this case use deployment1.

1. Type in use <database name>;

Replace <database name> with the name you used to create the database.

1. Type in create table data (time DATETIME, sonar DOUBLE PRECISION, temperature DOUBLE PRECISION, humidity DOUBLE PRECISION, pressure DOUBLE PRECISION);
2. You have successfully created a new database with a table with columns
   1. time
   2. sonar
   3. temperature
   4. humidity
   5. pressure

## Empty Data in Table

1. Login to MySQL
2. Type in: use <database name>;
3. Type in: truncate table <table name (most likely named data)>;

## Wire Up the Electronics

Follow the circuit diagram attached with this document to wire up the circuits for the sensors. Note that each row up to the divider is connected. i.e. **1 a->e** is one line and **1 f->j** is another, therefore wiring a resistor from **1a** to **1e** will do nothing as the electric current will choose the path of least resistance and simply travel underneath it, which is why resistors should be wired across lanes. Images of my circuit have also been attached to act as a guideline.

## Test if the Electronics Work

To run a script to test if the electronics work, follow the below steps:

1. Type in: cd Tests
2. Then: sudo python <script name (below)>
   1. rangefinder.py - To test the HC-SR04 Sensor.
   2. dhtread-test.py - To test the DHT22 Sensor.
   3. bmp-test.py - To test the BMP180 Sensor.
3. Once finished type in: cd

## Starting the Program on Boot

1. Type in: sudo nano /etc/rc.local
2. Scroll down using the arrow keys
3. Add this: sudo python /home/pi/collectdata.py &
4. Press Control-Out, then Enter, then Control-X

If Connected to a screen, it should output something like the below:

**================================================================**

**Measurement number: 13**

**Got Sonar: 308.523178101mm.**

**Got DHT22: Temperature = 18.1000003815 Celsius. Humidity = 50.9000015259%.**

**Got BMP180(085): Temperature = 17.0 Celsius. Pressure = 101880 pa.**

**================================================================**

## Let the Program Run

1. Reboot the Raspberry Pi
2. Login to the Raspberry Pi
3. Set the date on the Raspberry Pi
4. Login to MySQL
5. Empty the data in the table
6. Exit MySQL
7. Type in: logout

The program is now running in the background and collecting information, you can now leave and do other work, just make sure that the yellow LED is blinking (about every 2 seconds), if it ever stops you can start a new database and keep collecting data:

* 1. Login to the Raspberry Pi
  2. Login to MySQL
  3. Create a new database and table, name the database deployment<previous + 1>
  4. Exit MySQL
  5. Type in: nano collectdata.py
  6. Scroll down and change the section, which says deployment<1 or any other number> to correspond with the new database name.

e.g.

***db = MySQLdb.connect("localhost", "root", "raspberry", "deployment1")***

becomes

**db = MySQLdb.connect("localhost", "root", "raspberry", "deployment2")**

* 1. Press Control-Out, then Enter, then Control-X
  2. Repeat from Step 1.

## How to Check into the Database

1. Login to the Raspberry Pi
2. Login to MySQL
3. Type in: use <database name>
4. Type in: select \* from data; This will display a table of all the data gathered and saved to that table in the database.

Something like (Text has been made smaller to fit this document):

**mysql> select \* from data;**

**+-----------------------------+----------------------+-----------------------+--------------------+-------------+**

**| time | sonar | temperature | humidity | pressure |**

**+-----------------------------+----------------------+-----------------------+--------------------+-------------+**

**| 2014-08-29 15:31:05 | 305.406332016 | 17.4416664759 | 51.086666743 | 101873 |**

**+---------------------+------------------------------+-----------------------+--------------------+-------------+**

**1 row in set (0.01 sec)**

1. Exit MySQL
2. Type in: logout

## Export the Data in the Database as CSV and Copy to USB

1. Plug a USB into the Raspberry Pi
2. Login to the Raspberry Pi
3. Login to MySQL
4. Type in: use <database name>;
5. Type in: select time, sonar, temperature, humidity, pressure from data into outfile “/tmp/<name of file>.csv” fields terminated by “,” lines terminated by “\n”;
6. A good idea for the name of file is the name of the database.
7. Repeat steps 4 and 5 for different databases, remember to change the file name to something different every time.
8. Start and Connect to the TightVNC Server
9. Launch the File Manager
10. At the top of the File Manager window, type in /tmp into the location bar.
11. Drag the CSV file(s) to the USB
12. Eject the USB and unplug it (The Raspberry Pi may reboot when you physically unplug the USB).

## Graph the Data

This will be done on your computer not the Raspberry Pi.

1. Copy the CSV files from your USB to your computer.
2. Open them in Microsoft Excel
3. If there are blank rows in between rows with data, do the following:
   1. Select the whole table and press both the **fn** and **f5** keys at the same time.
   2. Click Special
   3. Select Blanks on the left hand column
   4. Click Ok
   5. Then right click a selected cell
   6. Click Delete…
   7. Select the Entire row option
   8. Click Ok
4. Now open TextEdit
5. Create 4 blank text files
6. For each press **Command-Shift-T** at the same time to make them plain text documents.
7. Now in Excel select a column and copy and paste it into a text file (one column per text file) and save the text file with the correct name. --- Note: the columns go from left to right in the order of time, sonar, temperature, humidity, and pressure ---
8. Launch Plot2
9. Press **Command-N**
10. Press **Command-Shift-A**
11. Save the file with the appropriate name.
12. Make sure each graph window takes up the whole screen
13. Press **Command-Option-E** and save each graph with an appropriate name as an EPS file. --- Note: EPS files are scalable and can be used later for comparing data ---
14. Repeat steps 9 through to 13 for each text file. Now you should have 4 EPS files for each set of data (column), use these to compare the changes in sonar distance measurements as the temperature, humidity, and pressure changes.

# Variables

## Independent Variable(s)

The temperature.

## Dependent Variable(s)

The relative humidity to the temperature in %, the temperature in Celsius, the air pressure (last few experiments), the distance between a sonar distance sensor and a stationary object.

## Controlled Variable(s)

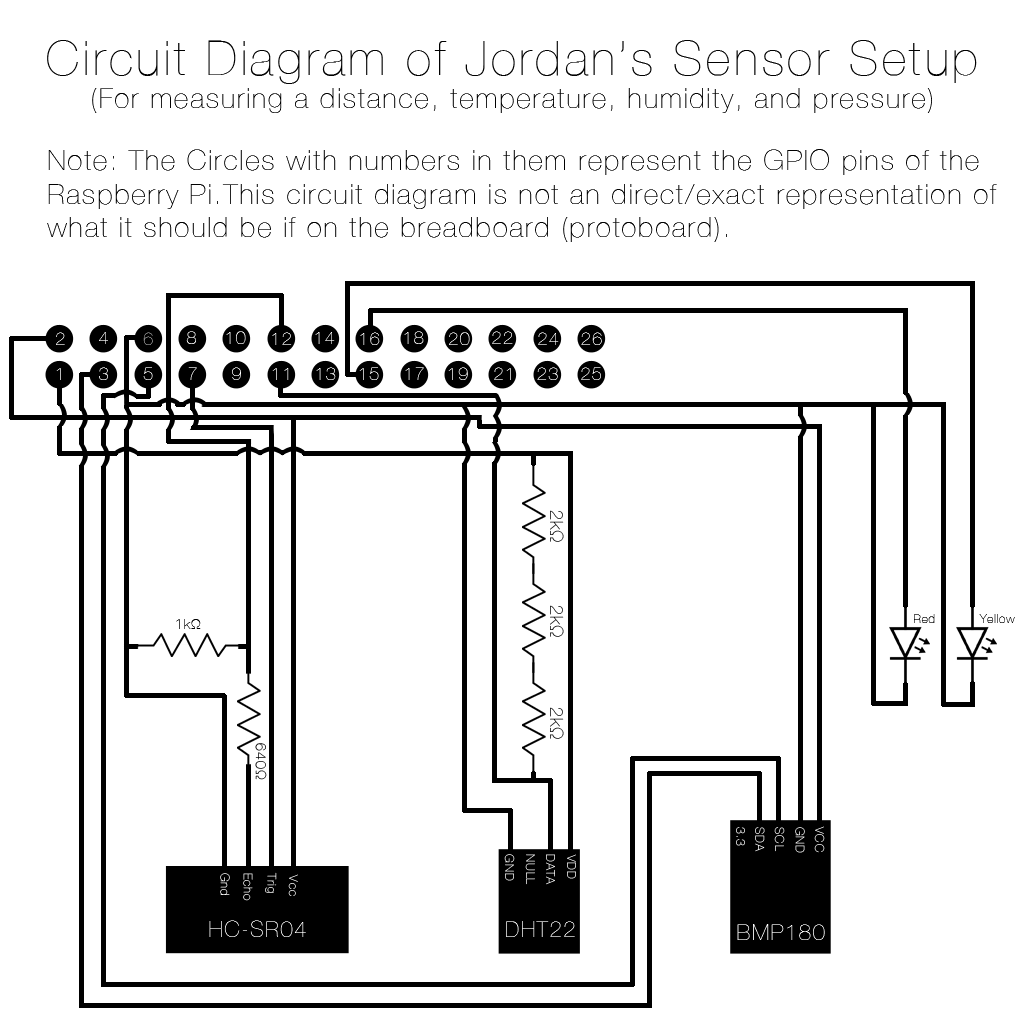
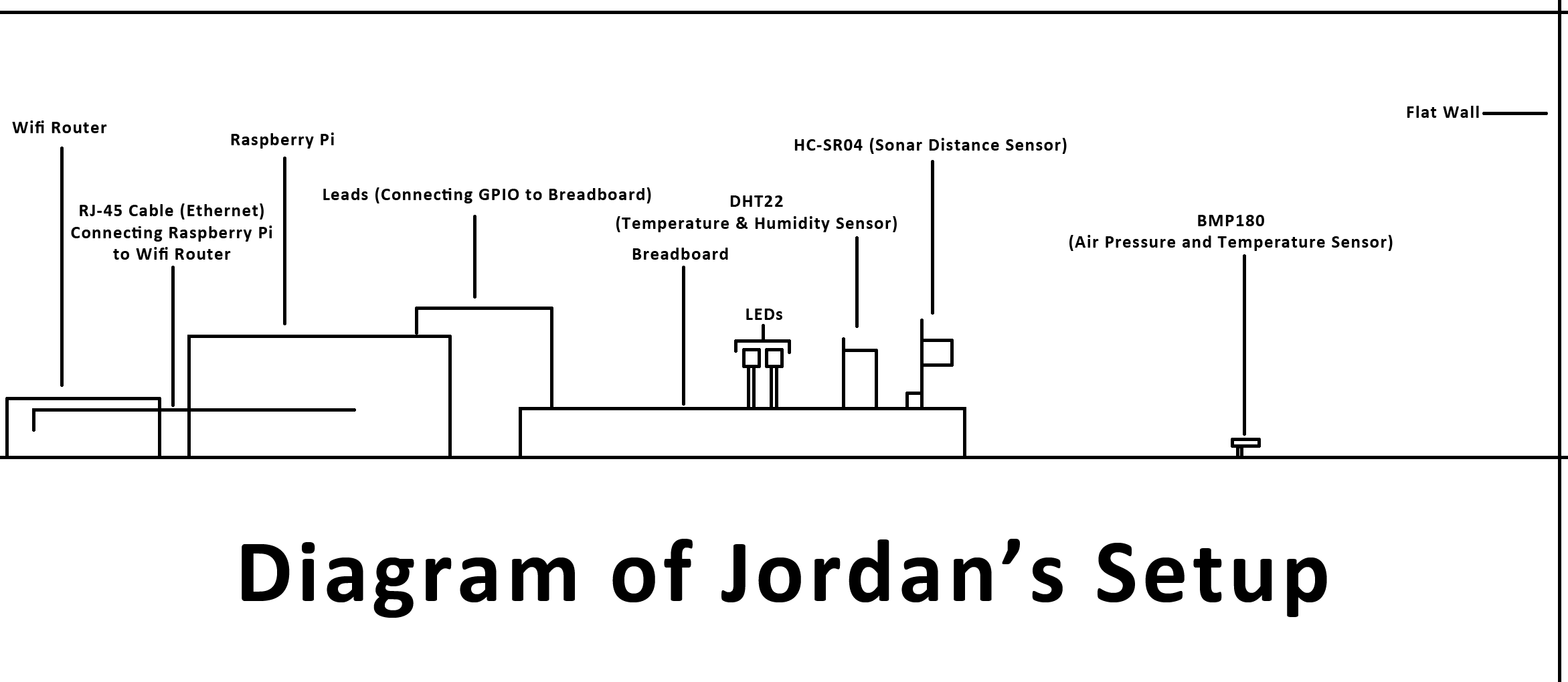
The distance between the sonar distance sensor and the wall of a box (about 300mm).

# Risk Assessment

|  |  |
| --- | --- |
| The Risk | The Protection |
| Having 5V go back into the GPIO pin from the sonar distance sensor to the Raspberry Pi, causing an electric overload, which causes the CPU to heat up and over heat until it has a little explosion and fails. (Maximum damage - Smoke, Up to $75 in damages. Raspberry Pi: $35-40, sensors $10 for both, leads $2-5, nearby Wi-Fi router (very unlikely) $30). | Use the appropriate resistors.  Be careful and double check the wiring. |
| Electric Shock | Be careful when handling the electronics, especially when plugging into the wall, make sure that the Raspberry Pi is off when wiring. |

# 

# Diagram(s)



# Results

As it would not work to import all of the results files into this document, I have attached them in a separate folder named results.

# Observations

## How The Observations Were Made

The experiment recorded distance (mm), temperature (Celsius), and humidity (%) about every 2 seconds. After a minute (30 recordings) it would average these recordings and store the average into a database. After I received a pressure sensor from Mr. Clark, the experiment also recorded pressure.

Once the deployment was over, I would export the data in the database as a CSV (instructions in the Method), copy the CSV onto my computer and open it in Microsoft Excel, and save it as a spreadsheet. After which, I would split up the columns into separate text files. These text files are then used as a data source for a program on OSX called Plot 2.200.

Once imported it will create a line graph, these line graphs will be used for the following observations.

## The Observations (No Pressure)

I found that the observed distance across about 300mm rises when the temperature drops and the relative humidity rises. This was a common observation throughout the Deployments 1 to 4.

## The Observations (With Pressure)

I found that the observed distance across about 300mm was more constant once the pressure sensor was added, but spiked a little bit more often. This was common throughout Deployments 6 to 8. Deployment 5 seemed to have run into an issue, which cause the observed distance to plummet from around 300mm to 5mm.

In a closer examination of Deployments 6 to 8, I found that there is a very slight rise in the observed distance from the sonar distance sensor when the temperature is slowly decreasing, and the humidity and air pressure drops. This pattern occurs in Deployments 6 to 8.

# Discussion

## Overview

In this experiment, a sonar distance sensor, temperature and humidity sensor, and a barometric pressure sensor were use to continuously gather data over a set period of time. This data was then graphed and used to visually identify the affect of the temperature and its relative humidity on the observed distance from a stationary sonar distance sensor. The graphs are placed one above the other to line up the changes in the observed distances from the sonar distance sensor with the respective temperature, humidity and pressure measurements.

### How My Program Works

The program I wrote to collect the measurements takes 30 measurements for each sensor and averages it out to attempt to eliminate any one off errors. The program also ignores all other measurements if one fails to return data, which means that all of the data was taken and averaged at the same time. This means that about every minute, there is an average measurement recorded and stored into a database.

### Other Important Notes

Unfortunately, for the first few Deployments (1-3), there was a single temperature measurement in each, which had dropped to around -90 degrees Celsius, which meant when graphing, any program would make the range very large, causing the majority of measurements to be rendered into something close to a flat line. To get this issue, I went and averaged the temperature measurement from before and after the rogue temperature measurement. This rogue measurement was due to the something going wrong in the DHT22 temperature and humidity sensor.

## Results

### Deployments 1 through to 4 (Without Air Pressure)

By looking and comparing graphs, I can see that in the Deployments 1 through to 4, that there is a relationship between the sonar distance, temperature, and humidity measurements. The relationship being, that as the temperature drops and the humidity rises, the observed distance by the sonar distance sensor is greater when the temperature drops and humidity rises.

#### Deployment 1

In Deployment 1, the graphs show certain patters, such as when the observed distance is greater when the temperature drops and humidity rises, that the humidity rises when the temperature drops, and that the humidity drops when the temperature rises. From just this data set, it could be said that the temperature has an inverse affect on the humidity.

#### Deployment 2

In Deployment 2, some of the parts of a pattern shown in Deployment 1 are visible. An example would be at the end when the temperature drops and the observed distance rises. This set of data does not show the patterns where the humidity rises when the temperature drops, or when the humidity drops when the temperature rises.

#### Deployment 3

In Deployment 3, there seemed to be some issues with the sonar distance sensor when taking measurements, but it is evident that when the temperature rose to about 16 to 18 degrees Celsius, the sonar distance sensor suddenly became more consistent in its measurements. Then when the temperature dropped, the sonar distance sensor became less consistent again. I am unsure what made the sonar distance sensor inconsistent, but this tells us that the temperature has an affect on the observed distance from the sonar distance sensor, or the speed of sound.

#### Deployment 4

In Deployment 4, the patterns shown in Deployment 1 are visible again. The most evident is near the beginning of the graphs when the observed distance spikes when the temperature plummets and humidity jumps. This is also faintly visible near the end of the graphs when there is a slight lump/rise in the observed distance by the sonar distance sensor. Here the temperature drops and the humidity is rising.

#### Summarizing Deployments 1 through to 4

From the results in Deployments 1 through to 4, I think I can safely state that the temperature does have an affect on the observed distance of a sonar distance sensor. In Deployments 1,2, and 4, the temperature has an inverse affect on the observed distance, which means that as the temperature changes in one direction, the observed distance will do the opposite.

### Deployments 5 thought to 8 (With Air Pressure)

I found that the measurements recorded from Deployments 5 through to 8 had more consistent observed distances, excluding 5, which seemed to have had an error while testing. According to Mr. Clark, the reasoning for the more consistent results was likely due to the BMP180 air pressure sensor regulating the clock on the Raspberry Pi better than the Raspberry Pi itself. This is because the Raspberry Pi calculates the distance according to the time it takes for the sonar distance sensor to return a ping after sending one out. If the clock is regulated better by the BMP180, the observed distances would be more accurate because the time that is taken to get a return ping is more accurate than without the BMP180.

#### Deployment 5

Deployment 5 was a failure; the sonar distance sensor seemed to have run into some sort of issue at some point, causing it to return very short ping distances. This may have been caused by dust getting stuck in the wire mesh covering the ping speaker and receiver, causing the very short distance of about 5 millimeters to be returned. I speculate this because after noticing the sudden change of measurements, I stopped, unplugged the sonar distance sensor and blew into both of the speakers and receivers, which fixed the problem for the next 3 Deployments.

#### Deployment 6

In Deployment 6, a faint connection could be made if looking closely at the graphs, between the observed distances, the humidity and the air pressure. There is also a very faint similarity in graph shape at about the 800 mark in the observed distances and temperatures. Here the graphs have a very similar shape, not inversed, which collides with the analysis of the Deployments 1 through to 4.

#### Deployment 7

In Deployment 7 there was only one very faint link I could make with the observed distance that was that as the air pressure drops, the observed distance increases. This is very faintly visible at both the beginning of the graphs and from around the 600 mark to the 875 mark. This also works in the other way, such as at the very end when the air pressure rises, the observed distance drops. There is another link between the temperature and humidity, where as the temperature rises the humidity drops. This is especially evident at the end where the temperature jumps and the humidity drops.

#### Deployment 8

The relationship between the observed distance and air pressure is also evident in Deployment 8. This is at around the 525 mark to the 775 mark where there is a slight rise in the observed temperature as the air pressure slightly drops. Otherwise, the temperature and humidity relationship is still existent, especially at the end where the temperature rises and humidity drops and vice versa.

#### Summarizing Deployments 5 through to 8

Deployment 5 will be omitted from the following summary due to the technical errors, hence useless data. From the results from Deployments 6 through to 8, it is hard to pull out a concrete relationship between datasets. What is faintly there is the inverse relationship between the observed distances and the air pressure. This is the only relationship that exists in Deployments 6, 7, and 8. Otherwise, the inverse relationship between the temperature and humidity exists in both deployments 5, 7 and 8, 5 will be included here as there was nothing wrong with the temperature and humidity sensor.

### The Reliability of the Results

I believe that the results obtained in this experiment are reliable, as the experiment has been run several times, each time collecting hundreds of sets of data, some having a couple thousand sets of data. This experiment was not just run many times, but the program I wrote averaged every 30 measurements at a rate of a measurement every 2 seconds, which means it saves a reliable result every minute. In addition to this, every 2 seconds, there were two temperature measurements, one from the DHT22 and one from the BMP180 at two different locations, these measurements where averaged every two seconds have a better sense of the surrounding temperature. This adds another layer of reliability to my experiment. This is why I believe that the results I obtained in this experiment are reliable.

## Future Experiments

If this experiment was to be redone in the future, there are some improvements that could be made.

### Have all of the sensors from the beginning

In my experiment, I thought that having a temperature and humidity sensor would be enough but as research progressed, it became evident that an air pressure sensor would also be required to have more accurate and relevant datasets. It also means that there will be one less unexpected variable of clock regulation.

### Have an accurate set distance

In my experiment, although the distance was static at all times, it changed slightly in every deployment due to changing setups or adjusting sensors. To make the deployments more consistent, it may be a good idea to have a set distance of say about 300mm to 1000mm for each deployment. This would ensure that any changes in each measurement can be accurately graphed as the set distance is known.

#### Use a flat wall

In my experiment, I setup my equipment in a white plastic container, which had its walls coming out from the bottom pointing out (walls that slanted outwards). This is an issue that should be fixed for future experiments, as it means that the would-be set distance would change if the height of the sonar distance sensor changed.

#### Sonar distance sensor should be parallel to ground and perpendicular to wall

It was hard to get the sonar distance sensor parallel to the ground in my experiment, again causing slight changes in the observed distance due to the angle of the sound waves hitting the wall being not quite perpendicular to it. By ensuring that the sonar distance sensor is always parallel to the ground and perpendicular to the wall, it would ensure better, and more consistent measurements.

### Use a different Temperature and Humidity Sensor, or fix the program

The DHT22 was the main cause of having to start new deployments as I call it. This was because at some point while the program was running, the DHT 22 would simply stop responding to any requests from the Raspberry Pi. Mr. Clark says that this may be due to water droplets forming/making its way into the sensor, and a way around it would be to temporarily heat up the DHT22 using some trick and forcing the water to evaporate. This could be a viable way of getting past the issue of the DHT22 not responding, but I believe that there are better ways, such as pausing the program and waiting for it to respond. An even better alternative would be to use a whole different sensor that is not affected by moisture, although this would mean having to rewrite a small section of the program.

### Have the program run for longer

By having the program run for longer, it is possible to see the changes through out the day. A good length of time would be about a week of continuous recording data and rerunning the experiment at least 3 times. Although this would mean doing the above of replacing the temperature and humidity sensor or rewriting the program, it would come to a huge benefit when scanning though the graphs, as rises and falls of temperatures and humidity becomes more evident and the changes in the observed distance by the sonar distance sensor can be easily tied to other data sets’ changes.

### Have a Control

It would be a great idea to have a control experiment for this, where the temperature always stays the same. I was unable to this due to only having enough equipment for one setup and keeping the temperature the same in the science labs would be very tricky. By having a control, we can compare the contrast the differences between the two datasets to see if there is a change in the observed distance by the sonar distance sensor regardless of the change in temperature.

# Conclusion

## Goals/Aims

In this research task, I set out to observe the relationship of the speed of sound, temperature, humidity, and air pressure. My aim was to find and explain this, and possibly even have an equation that can calculate the speed of sound by using the temperature, humidity and air pressure. Unfortunately, I ran out of time before being able to do this due to hiccups with the DHT22 sensor. I would say, that I have only completed half of this research task.

## Results

The results that I obtained from this experiment tell me many similar things about the relationship between the speed of sound, the temperature, humidity, and air pressure.

### What I was looking for

According to my background research, although, extremely minimal, I was looking for a drop in the observed distance when either the air pressure dropped or the humidity rose. I was also looking for on the side, was as the temperature rose, the humidity would also rise.

### What my results tell me

My results tell me two similar yet different things when it comes to the relationship between the observed distance by a sonar distance sensor and the surrounding humidity and air pressure. The same applies for the relationship between the temperature and humidity.

#### Without Air Pressure (Deployments 1-4)

In deployments 1 through to 4, it can be seen that the observed distance rises as the surrounding humidity drops. It can also be seen that the temperature has a nearly inverse affect on the humidity, causing the humidity to rise when the temperature drops and vice versa.

#### With Air Pressure (Deployments 5-8, observed distances are ignored in 5)

In deployments 5 through to 8, it was harder to spot the patterns.

##### Deployment 5

In deployment 5, it can be seen that again the temperature has a nearly inverse affect on the humidity. Then the humidity has a nearly inverse affect on the air pressure.

##### Deployment 6-8

In deployments 6 through to 8, the same nearly inverse temperature and humidity relationship can be seen in deployments 7 and 8, except for 6. More importantly, there is a very faint relationship between the observed distance and the air pressure. This faint relationship is that as the air pressure drops, the observed distance rises. Also at the end of deployment 7 there is another key relationship where as the humidity drops, the air pressure rises.

### The Conclusion on the Results

I can say from the results gathered that the results to an extent, proves what the background research shows, where as the humidity rises and air pressure drops, the observed distance from the sonar distance sensor will decrease. Although it was broken up into two parts (1-4 and 5-8), the changes that were supposed to occur occurring to the background research said it would happen. I can also say that there is something close to an inverse relationship between the temperature and the humidity.

## Social and Scientific Significance – Real World Uses

This experiment has a significant social and scientific significance because sonar distance sensors are used in many industries to calculate distances, if the sensors are used to calculate distances over long distances, what would be a minute inaccuracy can be magnified to a point where there will be issues using the information for critical projects. By taking into account the temperature, humidity, and air pressure, dynamic and more accurate sonar distance sensors can be made to be used in industries where accuracy is critical but also on a budget. An example of uses for a dynamic and more accurate sonar distance sensors are in robots, where accurate information about its surrounding are required in a changing at the atmosphere can change quickly due to air conditioning or humidifiers. There are many other real world uses for what my experiment has proved.

## The Verdict

I cannot say this experiment was a success, but nor was it a failure. This experiment has proved my hypothesis and the background research that my hypothesis was based on. Proving a hypothesis might make any experiment a success, but in this experiment, there were many unexpected variables such as the better clock regulation with the BMP180, or not even having the BMP180 from the beginning, this made it very hard to draw relationships and spot patterns in the datasets. This to me is not good enough for an experiment of this level and would redo it if the opportunity presents itself. This means I have not come to a verdict yet, and I don’t think I will unless I fix up my mistakes and redo this experiment as according to my suggestions for future experiments.

# Bibliography

*BMP180, from Java* 2014, viewed 27 August 2014, <<http://www.lediouris.net/RaspberryPI/BMP180/readme.html>>.

Brennan, J 2014, *How Does Humidity Affect Speed of Sound?*, viewed 09 September 2014, <<http://science.opposingviews.com/humidity-affect-speed-sound-22777.html>>.

Bruno, C 2013, *NOBRU: TUTORIEL RAPSBERRY PI mesures météo*, viewed 27 July 2014, <<http://nobru54.blogspot.com.au/2013/07/rapsberry-pi-mesures-meteo.html>>.

CSGNetwork 2011, *Resistor Color Code Calculator*, viewed 26 July 2014, <<http://www.csgnetwork.com/resistcolcalc.html>>.

Eberhard, S 2014, *Calculation speed of sound in humid air and the air pressure humidity moist air water vapor density of water atmospheric pressure - sengpielaudio Sengpiel Berlin*, viewed 29 August 2014, <<http://www.sengpielaudio.com/calculator-airpressure.htm>>.

Engel, BA 2003, *ACS Articles - The Speed of Sound*, viewed 25 July 2014, <<http://www.allchurchsound.com/education/edart/soundspeed.html>>.

Gaven, M 2013, *Ultrasonic Sensor with the Raspberry Pi*, viewed 24 July 2014, <<https://www.youtube.com/watch?v=xACy8l3LsXI>>.

Mark, K 2014, *Tutorial: Raspberry Pi GPIO Pins and Python*, viewed 27 July 2014, <<http://makezine.com/projects/tutorial-raspberry-pi-gpio-pins-and-python/>>.

Michael, W 2014, *Plot2*, viewed 17 August 2014, <<http://plot.micw.eu/>>.

Steve, B 2013, *Raspberry Pi web server - Using MySQL on a Raspberry Pi* , viewed 26 July 2014, <<http://raspberrywebserver.com/sql-databases/using-mysql-on-a-raspberry-pi.html>>.

TutorialsPoint 2014, *Python Tuples*, viewed 27 July 2014, <<http://www.tutorialspoint.com/python/python_tuples.htm>>.

Wikipedia 2014, *Sonar*, viewed 21 July 2014, <<http://en.wikipedia.org/wiki/Sonar>>.

Wikipedia 2014, *Speed of sound*, viewed 09 September 2014, <<http://en.wikipedia.org/wiki/Speed_of_sound>>.

Wolfram Alpha 2014, *Sonar -- from Eric Weisstein's World of Physics*, viewed 23 July 2014, <<http://scienceworld.wolfram.com/physics/Sonar.html>>.