The Development of a Low-Cost Air Quality Sensor Array for Mobile Platforms

Justin Taylor
Concordia University - Portland, justaylor@cu-portland.edu

Follow this and additional works at: https://commons.cu-portland.edu/theses
Part of the Biology Commons

CU Commons Citation
https://commons.cu-portland.edu/theses/187

This Open Access Thesis is brought to you for free and open access by CU Commons. It has been accepted for inclusion in Undergraduate Theses by an authorized administrator of CU Commons. For more information, please contact libraryadmin@cu-portland.edu.
The Development of a Low-Cost Air Quality Sensor Array for Mobile Platforms

A senior thesis submitted to
The Department of Math-Science
College of Arts & Sciences

By

Justin Taylor

Faculty Supervisor
Dr. Matthew Wise
Date

Department Chair
Dr. Mihail Iordanov
Date

Dean, College of Arts & Sciences
Dr. Michael Thomas
Date

Provost
Dr. Michelle Cowing
Date

Concordia University
Portland, Oregon
April, 2019
# TABLE OF CONTENTS

Title Page.................................................................................................................................1

Table of Contents....................................................................................................................2

Acknowledgements..................................................................................................................3

Abstract....................................................................................................................................4

Introduction...............................................................................................................................5

Methods.....................................................................................................................................8
  
  *Sensor Array Design & Fabrication*..................................................................................8
  
  *CO₂ Sensor Testing*..........................................................................................................10
  
  *Humidity Sensor Testing*..................................................................................................11
  
  *Real World Testing*.........................................................................................................11

Results....................................................................................................................................13

  *CO₂ Sensor Testing*..........................................................................................................13
  
  *Humidity Sensor Testing*................................................................................................16
  
  *Real World Testing*.........................................................................................................19

Discussion...............................................................................................................................21

  *Sensor Array Design & Fabrication*...............................................................................21
  
  *CO₂ Sensor Testing*........................................................................................................21
  
  *Humidity Sensor Testing*................................................................................................22
  
  *Real World Testing*.........................................................................................................22

Conclusion...............................................................................................................................23

Appendix A...............................................................................................................................25

References...............................................................................................................................26
Acknowledgments:

I want to thank Concordia University. Concordia has provided me with the experience and schooling necessary for me to excel in my future endeavors. I have been given many opportunities at Concordia that I would never have anywhere else. Concordia has been my home and always will be. Thank you to all the professors that have supported me during my time at Concordia.

I owe everything to Dr. Matthew Wise for giving me this opportunity. He believed in me and invited me to help on this research project. I struggled through his inorganic chemistry class, but he was always available to ask questions. With his constant help, I was able to excel in his class. I then started to research with him on this project. I gained an excellent mentor as well as a friendship. I wish him the best with his new career in Colorado and Concordia will sincerely miss him.

I always want to say thanks to Nicole Patterson. She has been a constant help to my research. Without her, this science program would be greatly negatively impacted. She has always been there for me and the other research students anytime we needed. Thank you for doing so much for us and the science program.

Lastly, but not least, I want to thank my family. My parents, Jeff and Jennifer, have always been incredibly supportive of my goals. They have helped me through school and always believed in me. My parents are my biggest fans. I could not have accomplished anything without their love. Also, I would like to thank my sister, Jenessa. She went to Concordia before me and showed me how to be successful. Jenessa has been my role model by working hard at everything in her life.
The Development of a Low-Cost Air Quality Sensor Array for Mobile Platforms

Justin Taylor, Concordia University, Portland, OR
Math and Science Department, Concordia University, Portland, OR

Abstract:

The traditional method of measuring air quality uses large immobile sensors. These sensors can cost hundreds of thousands of dollars. If small, low-cost sensors are proven reliable then they can be attached to a drone for air quality management. The K30 Carbon Dioxide sensor from CO2Meter.com was used for testing. The BME280 sensor from Adafruit was used to measure humidity and temperature. A Raspberry Pi was used as an electronic controller and the programming was written in Python. After testing was completed, the K30 Carbon Dioxide sensor was found inadequate for atmospheric testing. The BME280 was determined effective for air quality sensing as long as the relative humidity was below 75%.
Introduction:

Air quality and pollution are some of the most challenging public health dangers (Figure 1). There were 3.7 million deaths attributed to air pollution in 2012 (“WHO: Burden of Disease from Ambient Air Pollution,” 2012). Air pollution not only poses a risk to individuals but to the whole planet — air pollution is causing an increase in the average temperature of the earth. It is otherwise known as enhanced climate change. Carbon dioxide is important to measure because it acts as a greenhouse gas. Greenhouse gases can be dangerous if concentrations are too high because they absorb infrared radiation from the sun. This leads to an increase in the overall global temperature. There is a necessity for a new system of testing air quality so that greenhouse gases like CO\textsubscript{2} can be widely measured.

Historically, air quality is tested using large, immovable sensors that cost hundreds of thousands of dollars. These instruments are accurate and do an excellent job measuring different concentrations of gases such as CO\textsubscript{2}. The instruments can also measure particulates, and other variables like temperature and relative humidity. The problem with these devices is that they cannot be quickly moved. Also, they are only able to measure the air immediately surrounding them. Air quality can drastically change depending on the location. The data from these instruments can only be used as “guidelines”. For example, Bullseye Glass

Figure 1: An example of what smog and air pollution can look like it heavily affected areas.
Company in Portland, was able to release arsenic and other heavy metals into the air. The toxins were released through smokestacks during glass production. The toxins then scattered over the neighborhood which poisoned the soil and many people. Bullseye Glass was able to let this happen because monitoring stations were too far away to measure the pollutants (Guevarra, 2019).

The goal of this project is to create an air quality sensor that can be as accurate as the large sensors but is free to move on a drone. The drone that is ideal for the project is an octocopter. An octocopter will have enough lift to be able to carry the sensor array and can be maneuvered easily around structures. If this sensor array and drone can be created, then the pollutant producer like Bullseye Glass Company can be found early and accurately measured. The challenge is to find sensors that are small but can accurately measure low concentrations of gases and particulates in the air. The sensors that will work for this project are electrochemical sensors.

Electrochemical sensors have been used in the workplace to monitor air quality and carbon monoxide, carbon dioxide, chlorine or methane gases. If the concentrations get too high, then an alarm will be triggered. Modifying these sensors to work in an open environment may prove challenging. Moreover, they may not be accurate enough to provide research level data. In order to find out if these sensors are adequate, they need to be tested in the lab using concentrations similar to the environment.
The components of an electrochemical sensor are a working electrode, a counter electrode, and a reference electrode (Figure 2). The electrodes are encased in a housing that is filled with a liquid electrolyte. The working electrode is covered by a permeable membrane that allows the specific gas to diffuse through. The gas diffuses through the membrane and comes into contact with the working electrode. This then causes an electrochemical reaction. The reaction will either cause electrons to flow from the working electrode to the counter electrode or vice versa. The direction that the electrons flow depends on whether the reaction is an oxidation or reduction. The flow of electrons creates a circuit. The voltage varies depending on the concentration of the gas that is being measured. There are other electronics associated with the electrochemical sensor that helps to amplify and read the concentrations of the gas (“Electrochemical Sensors,” n.d.). The electrochemical sensor must be combined with an electric controller to record the concentrations.

Several options can be used to read the data coming from the electrochemical sensor. The first and easiest option is to buy a pre-built gas sensing unit. An example is the pSense Portable CO₂ Meter made by CO₂Meter.com. This device includes the electrochemical sensor and the electronics that display the results without any effort by the user. These devices cost hundreds or thousands of dollars, so they are not feasible for this project. The next popular method is to use what is called Arduino or Raspberry Pi.

Arduino and Raspberry Pi are very similar platforms. The Raspberry Pi is a credit card sized computer that only costs 35 dollars (“Teach, Learn, and Make with Raspberry Pi,” n.d.). It can be wired to a monitor, keyboard and mouse to be used as a fully functioning computer. The Pi also runs on Python which is a powerful programming language. Many popular programs and businesses including YouTube, Google, and Instagram rely on Python to run their applications.
The Raspberry Pi also boasts a 1.4GHz 64-bit processor and a 1GB RAM ("Teach, Learn, and Make with Raspberry Pi," n.d.). The power of the Raspberry Pi is nowhere near a modern computer. However, it is more powerful than most smartphones and will work for the air quality sensor application. The Raspberry Pi is the processing platform that was chosen for the project because it has more capabilities, runs on Python, has faster processing than Arduino, and is tiny in size.
**Methods:**

**Sensor Array Design & Fabrication:**

The first step was to acquire the necessary materials for fabrication. The Raspberry Pi 3 model B+ was purchased for only 35 dollars. There are two primary sensors used for the project. The BME280 sensor from Adafruit only cost 20 dollars and was used to test the atmospheric temperature, and relative humidity. The K30 CO₂ sensor sold by CO₂Meter.com for 95 dollars and was the most expensive component. In order to wire the sensors to the Raspberry Pi, an Adafruit Perma-Proto Pi HAT was purchased for 5 dollars. Other miscellaneous items were needed to fabricate the Sensor Array. This includes wires, soldering iron, solder, a portable battery, and a sensor box. In total, the air quality sensor array was built for under 250 dollars.

The fabrication is straight forward and can be constructed using the wiring diagram attached (Figure 3). All the wires are soldered together to provide strong connections.

![BME280 temperature/relative humidity sensor](image)

**Figure 3:** Wiring schematic for Sensor Array

The sensor box holds all of the electrical components. The sensor box allows the sensors to actively sample air using a pump and protect the sensors from damage. An 8” x 4.7” x 2.2” electrical box was used to create the sensor box (Figure 4). Three holes were drilled into the box.
Two ¼ inch holes were drilled on either end of the box. Then Swagelok tube fittings were screwed and glued into the holes. The Swageloks provide the ability to attach plastic tubes to the sensor array. The plastic tubes allow air to be pulled through the sensor box. Also, one ½ inch hole was drilled in the middle. This hole was used for calibrating the BME280 sensor by inserting the Omega HH314A Humidity Temperature Meter sensor through the ½ inch hole.

**Figure 4:** The sensor box used for testing purposes.

A computer program was written in code so that the sensors would work on the Raspberry Pi. The program was written in python. The sensor array the program can be downloaded from GitHub (https://github.com/t-dot/Raspberry-Pi-Sensor-Array.git) or rewritten from Appendix A.
CO₂ Sensor Testing:

The first step was to connect the gas tube from calibration gas to the K30. The calibration gas came in three different quantities of carbon dioxide; 200, 400, and 800 ppm. Nitrogen gas was used to calibrate the sensor for a 0 ppm concentration of carbon dioxide.

The CO₂ sensor was calibrated using a carbon dioxide test gas. The “Sensor Tube Cap Adapter” was installed onto the K30 CO₂ sensors (Figure 5). The adapter allows direct flow of gas to the sensor through an attached tube. There was a total of five different K30 CO₂ sensors tested which helps to provide an adequate population for statistical analysis.

The calibration of the CO₂ sensor was initiated by flowing 400 ppm calibration gas through the sensor. The gas flowed for 30 seconds to purge the normal atmospheric air. Then for another 30 seconds, while the gas is flowing, din1 must be short-circuited for at least 8 seconds. Din1 is a terminal on the sensor’s board. When Din1 is shorted it triggers the sensor to reset to 400 ppm. This will calibrate the sensor to 400 ppm.

After each sensor was calibrated, they were then tested to determine efficacy. The tube adapter was installed onto the sensor. Gas from the nitrogen tank was allowed to flow at 0.5 liters per minute. After a 30 second purge, the test was started. The data was to be collected for one minute and then averaged to find the measured concentration. This process was repeated for
each sensor, and then repeated for each carbon dioxide concentration. After each sensor has been calibrated then the test gas was drawn over the sensor to test if it read the correct concentration.

**Humidity Sensor Testing:**

The BME280 humidity sensor cannot be mechanically zeroed. The sensor has no method to tell itself where 0% relative humidity is unlike the CO₂ sensor. The way it was calibrated was by comparing its data to the Omega HH314A Humidity Temperature Meter. The Omega sensor was used because it is NIST certified. The Omega’s probe was inserted into the hole cut into the side of the sensor box. In order to keep an airtight seal, the probe was hot glued into place. A ¼ inch tube was inserted into one of the Swagelok’s of the sensor box. This tube was then connected to the Blaustein Atomizer (BLAM) unit. The BLAM utilizes compressed gas and a liquid to efficiently generate aerosols. But in this experiment, the BLAM is creating a humidified nitrogen steam of gas. The BLAM was filled with approximately 50 ml of distilled water before compressed nitrogen being pumped through at 4 liters per minute. The nitrogen is connected to the BLAM using ¼ inch tubing.

The sensor array being tested is placed into the sensor box and attached to a power source. The lid to the sensor box is screwed into place. Then the test is started allowing the nitrogen gas to flow through the BLAM into the sensor box. Data is taken every 10 seconds for the first three minutes and then once every minute after that. The trial is run for 20 minutes total to ensure the relative humidity leveled out. The test was repeated for each sensor array.

**Real World Testing:**

The sensor arrays were placed outside in order to test the reliability of all the carbon dioxide sensors in a “real-world” application. The sensor arrays were all powered by the same electrical outlet so that any electrical variances were eliminated. A motorized fan was used to
blow air a continuous stream of air over the sensors. This creates an active sampling environment. The test was started, and data was taken autonomously using the python program (Figure 5). The program takes a data point every second for approximately six days.

**Figure 6:** Raspberry Pi’s recording data outdoors for the “Real World” test.

**Figure 7:** Raspberry Pi’s “Real World” test from above including the fan.
Results:

CO₂ Sensor Testing:

The CO₂ sensors provided results that were close to one another. They were each slightly different except for at 400 ppm. The exact values for measured carbon dioxide concentration are displayed in Table 1. All of the sensors have a calibration curve graph which is displayed below (Figures 8-12). The error bars represent the ± 30 ppm accuracy of the CO₂ sensors.

<table>
<thead>
<tr>
<th>CO₂</th>
<th>Sensor 1</th>
<th>Sensor 2</th>
<th>Sensor 3</th>
<th>Sensor 4</th>
<th>Sensor 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 ppm</td>
<td>7 ppm</td>
<td>9 ppm</td>
<td>7 ppm</td>
<td>1 ppm</td>
<td>7 ppm</td>
</tr>
<tr>
<td>200 ppm</td>
<td>202 ppm</td>
<td>203 ppm</td>
<td>205 ppm</td>
<td>206 ppm</td>
<td>208 ppm</td>
</tr>
<tr>
<td>400 ppm</td>
<td>400 ppm</td>
<td>400 ppm</td>
<td>400 ppm</td>
<td>400 ppm</td>
<td>400 ppm</td>
</tr>
<tr>
<td>(Calibrated)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>800 ppm</td>
<td>784 ppm</td>
<td>790 ppm</td>
<td>794 ppm</td>
<td>791 ppm</td>
<td>800 ppm</td>
</tr>
</tbody>
</table>

Table 1: This table shows the measured concentrations of the CO₂ sensors. The first column, CO₂, is the concentration of the calibration gas.

Figure 8: Shows the calibration curve for Sensor 1.
Figure 9: Shows the calibration curve for Sensor 2.

Figure 10: Shows the calibration curve for Sensor 3.
Figure 11: Shows the calibration curve for Sensor 4.

Figure 12: Shows the calibration curve for Sensor 5.
Humidity Sensor Testing:

The humidity sensors all varied in their responses to the increasingly humid environment. The relative humidity of the sensors are compared to that of the Omega Humidity sensor in the below Figures 13-17. The test lasted a total of 1200 seconds. The humidity varied from about 40% to 95%.

**Figure 13:** Shows the difference in relative humidity of sensor 1 versus the Omega sensor.
**Figure 14:** Shows the difference in relative humidity of sensor 2 versus the Omega sensor.

**Figure 15:** Shows the difference in relative humidity of sensor 3 versus the Omega sensor.
Figure 16: Shows the difference in relative humidity of sensor 4 versus the Omega sensor.

Figure 17: Shows the difference in relative humidity of sensor 5 versus the Omega sensor.
Real World Testing:

The CO$_2$ concentration for all five sensors, for the most part, stayed between 400 ppm and 600 ppm. All five sensors varied and did not produce similar results. Sensor five stopped working after only 2 hours. Then sensor 2 stopped recording data after 1 day. The remaining sensors worked until the experiment was over. The data is displayed in Figures 18 and 19.

**Figure 18:** Plots CO$_2$ concentrations from all five sensors.
Figure 19: Plots the same data from Figure 18, but is focused on the first two hours of the experiment.
Discussion:

Sensor Array Design & Fabrication:

The sensor array went through several prototypes before the final product was created. A breadboard was used in the early stages in order to fabricate the sensor array. This allowed for variability when designing. The final product was more compact than the prototype. The final product needs to be compact so that it could fit easily on a drone. The total cost of the sensor array was under 250 dollars. This is significantly cheaper than the traditional methods of monitoring air quality.

CO₂ Sensor Testing:

All of the carbon dioxide sensors produced varying results which were disappointing. They were all the same type of sensor and used the same Raspberry Pi yet did not give similar results. None of the sensors were able to measure an exact 0 ppm. The tank is filled with pure nitrogen, and therefore the reading should be 0 ppm. The closest was sensor 4 which read 1 ppm. There were no sensors that correctly measured the 200 ppm concentration gas. Sensor 1 was the closest with 202 ppm. All of the sensors measured the 400 ppm gas. They were calibrated at 400 ppm which is the reason why they correctly measured this concentration. The sensors had a tough time measuring the 800 ppm test gas. The results varied from as low as 784 ppm by sensor 1 to 800 ppm by sensor 5.

The reason for varying measurements was due to sensors accuracy. By manufacturer standards, the sensors have an accuracy of ± 30 ppm. This accounts for the variance seen in the measurements. There was not one sensor that exceeded the 30 ppm variance. When dealing with atmospheric changes, 30 ppm is a substantial change. This leads to the conclusion that the K30 CO₂ sensor is not adequate for atmospheric air quality testing.
Humidity Sensor Testing:

All of the sensors were more accurate at lower relative humidity percentages. Once the humidity reached around 80%, the Omega and Raspberry Pi sensors stopped reading similar percentages. The Omega is NIST certified and will give accurate results. The Omega gave more accurate results and responded to changes in humidity much quicker than the BME280. The BME280 was reliable around lower percentages. Therefore, it is recommended to use for air quality purposes as long as the relative humidity is below 80%.

The temperature data was corrupted during data transferring. Unfortunately, there are no graphs available. However, the BME280 and the Omega were within a couple of degrees from each other. The BME280 would work reliably as a temperature sensor for air quality purposes.

Real World Testing:

The sensors were placed in the open air in a backyard located in NE Portland and three blocks from Concordia University. A small electric fan was softly blowing over the sensors. Also, a canopy was covering the sensors, so they did not get wet from the rain. There were a couple of complications with the experiment. The first being that the main sensor array (sensor 5) stopped working after only 2 hours. The second complication was that sensor 2 stopped recording after one day. These errors were not caught until the data was being processed after the experiment was conducted. The sensors stopped working due to a bug that arose in the code. The bug only caused a problem when there was too much data, and then the program crashed. The problem has been fixed by increasing the time between each data recording. The rest of the sensors recorded for about six days.

The largest CO$_2$ peak was a little above 900 ppm. This unbelievably high spike in CO$_2$ levels lasted only a couple of seconds and then went back to normal levels. One explanation for
this could be that someone went to look at the sensors. Their breath would then cause the CO₂ to
spike abruptly. Another reason is that squirrels caused the spike in CO₂. A walnut was found with
the sensors (Figure 20). A canopy was covering the sensors so this nut could only have gotten
there if it was placed there by a squirrel or another animal.

The sensors were all giving different readings. They each were within at least 100 ppm of
each other but never followed a similar trend. For example, if sensor 3 went down, then sensor 1
might go up, and then sensor 4 might stay the same. The sensors would have proved more
reliable if they followed similar trends, meaning they all went up or went down at the same time.
Since the sensors were not accurate, they are not recommended to be used in atmospheric air
quality testing.

**Conclusion:**

The carbon dioxide sensor produced by CO₂Meter.com will not be adequate for precise
air quality testing. The variability has too wide of range (± 30 ppm) to use to measure small
changes in atmospheric CO₂. The BME280 is accurate enough to measure changes in temperature
and relative humidity for our purposes. The Raspberry Pi was the right choice and works great
for this project. The fabricated sensor array also worked well. It was small and efficient enough
to be suitable for mobile platforms.

The next step for this project would be to find and test a carbon dioxide sensor that has a
better accuracy range. A smaller accuracy range would allow for more precise measurements of
changes in atmospheric CO₂ concentrations. Also, the “Real World” test should be conducted
indoors to remove any outdoor variability. It could prove to have better results. The sensors
should be put into a container with an airtight seal, then have a pump pull a consistent airflow
through the box. This would help eliminate the inconsistent air flow produced by the fan.
Figure 20: Walnut placed inside the box during the real world outdoor test by a squirrel or another animal.
Appendix A:

```python
# Drone

import serial
import time
from Adafruit_BME280 import *
import datetime
# import Adafruit_ADS1x15
import smbus
import gps

#adc = Adafruit_ADS1x15.ADS1115()
#GAIN = 4

bus = smbus.SMBus(1)

ser = serial.Serial("/dev/ttyAMA0")
ser.flushInput()

session = gps.gps("localhost", "2947")
session.stream(gps.WATCH_ENABLE | gps.WATCH_NEWSTYLE)

sensor = BME280(p_mode=BME280_OSAMPLE_8, t_mode=BME280_OSAMPLE_2, h_mode=BME280_OSAMPLE_1, filter=BME280_FILTER_16)
print "Serial Connected!"
time.sleep(1)

with open("OutsideTest.csv", "a") as log:
    while True:

        report = session.next()

        ser.write("\x80\x00\x00\x02\x25")
        resp = ser.read(7)
        high = ord(resp[3])
        low = ord(resp[4])
        co2 = (high*256) + low

        # Sets variables from BME280
        celsius = sensor.read_temperature()
        fahrenheit = celsius * 1.8 + 32
        pascals = sensor.read_pressure()
        torr = pascals * 0.00750062
        #hectopascals = pascals / 100
        humidity = sensor.read_humidity()
        # To get pressure in inches of Hg
        inches = pascals * 0.0002953
        # To get Altitude from pressure
        sealevel_pa = 101325  # Enter Sea Level Pressure for that Day#
        altitude = 44330.0 * (1.0 - pow(pascals / sealevel_pa, (1.0/5.255)))
```

bus.write_byte_data(0x39, 0x00 | 0x80, 0x03)
bust.write_byte_data(0x39, 0x01 | 0x80, 0x02)
data = bus.read_i2c_block_data(0x39, 0x0C | 0x80, 2)
data1 = bus.read_i2c_block_data(0x39, 0x0E | 0x80, 2)
ch0 = data[1] * 256 + data[0]
ch1 = data[1] * 256 + data[0]

# value = adc.read_ack_difference(0, gain=GAIN)
# value1 = adc.read_ack_difference(3, gain=GAIN)
# value2 = abs((value) - (value1))
# total = value2 / 750

if report['class'] == 'TPV':
    kmh = (report.speed * 1.852)
    print(time.strftime('%H:%M:%S %m/%d/%Y'))
    print("Latitude: {0:.2f}".format(report.lat))
    print("Longitude: {0:.2f}".format(report.lon))
    print("Speed: {0:.2f} km/h".format(kmh))
    print("Altitude: {0:.2f} meters ",format(report.alt))
    print("Temp: {0:.2f} °C",format(celsius))
    print("Humidity: {0:.2f} %",format(humidity))
    print("Pressure: {0:.2f} torr",format(torr))
    print("Altitude: {0:.2f} meters",format(altitude))
    print("Full Spectrum(IR + Visible)= %d lux" %ch0)
    print("Infrared Value= %d lux" %ch1)
    print("Visible Value= %d lux" % (ch0 - ch1))
    print("CO2= {0:.2f} ppm \n",format(co2))
    # print("NO= {0:.2f} ppb \n",format(total))

    log.write(time.strftime('%H:%M:%S %m/%d/%Y \n'))
    log.write("Latitude= {0:.2f} \n",format(report.lat))
    log.write("Longitude= {0:.2f} \n",format(report.lon))
    log.write("Speed= {0:.2f} km/h \n",format(report.speed))
    log.write("Altitude= {0:.2f} \n",format(report.alt))
    log.write("Temp= {0:.2f} °C \n",format(celsius))
    log.write("Humidity= {0:.2f} % \n",format(humidity))
    log.write("Pressure= {0:.2f} torr \n",format(torr))
    log.write("Altitude= {0:.2f} \n",format(altitude))
    log.write("Full Spectrum(IR + Visible)= %d \n%lux \n" %ch0)
    log.write("Infrared Value= %d \nlux \n" %ch1)
    log.write("Visible Value= %d \nlux \n" % (ch0 - ch1))
    log.write("CO2= {0:.2f} ppm \n",format(co2))
    # log.write("NO= {0:.2f} ppb \n",format(total))
    log.flush()

    time.sleep(5)
References:


15. Teach, Learn, and Make with Raspberry Pi. (n.d.). Retrieved from

    Overview of Small Unmanned Aerial Vehicles for Air Quality Measurements: Present
    Applications and Future Prospectives. Sensors, 16(7), 1072. doi:10.3390/s16071072

17. Villa, T., Salimi, F., Morton, K., Morawska, L., & Gonzalez, F. (2016). Development and
    Validation of a UAV Based System for Air Pollution Measurements. Sensors, 16(12),
    2202. doi:10.3390/s16122202

18. WHO: Burden of disease from Ambient Air Pollution for 2012, World Health
    Organization, Geneva,
    http://www.who.int/entity/phe/health_topics/outdoorair/databases/AAP_BoD_results_Ma

    System for Atmospheric Carbon Dioxide Concentration Using a Small Unmanned Aerial
    Vehicle. Journal of Atmospheric and Oceanic Technology, 23(5), 700-710.
    doi:10.1175/jtech1866.1

    Remote Sensing and Scientific Research: Classification and Considerations of