Improving the Firefighting Experience Through Data

Nikki Colton, Anna Przybyl, Manas Biju, Tanmay Thakur, Manav Dogra, Paul Wei

Abstract

Our goal for this project was to create a backpack that is radio-linked to a base station and provides useful information on firefighting conditions and firefighter vitals. We designed three backpacks, complete with printed circuit boards (PCBs) that each include an Arduino Mega 2560 microcontroller, multiple sensors, and Long Range (LoRa) radio communication modules. The data acquisition program we designed and radio communication allows the backpacks to send information on firefighting conditions and firefighter biometrics to the base station. The idea is that the data can be monitored at the base station to keep track of environmental conditions and if necessary, alert the firefighters of any possible danger.

1. Background and Introduction

Firefighters work in an extremely dangerous environment - buildings on fire. They encounter life-threatening conditions including smoke, oxygen deficiency, high temperatures, and poisonous atmospheres regularly. Personal protective equipment (PPE) like face masks with oxygen and fire-proof suits are commonly used to protect firefighters from these dangers. However, it never hurts to be extra cautious; there is the possibility that the PPE kit fails, a firefighter could get hit with collapsing structures, etc. To prepare for such scenarios and due to the low visibility conditions inside a building filled with smoke, we propose a data acquisition system (DAQ), so that the necessary rescue and evacuation procedures go as smoothly as possible.

The sensors in the backpack carried by the firefighter collect real-time environmental data including temperature, hazardous gas concentrations, and particulate matter concentrations in the air; it also records the firefighter's condition, such as heart rate, location, and velocity. The data are sent back to the command center through radio communication. Each firefighter's status is monitored continuously through the data, so that when they encounter danger, the command center will be informed from the data and take the appropriate action. For instance, if a firefighter's location remains stationary for an abnormal amount of time, they are very likely to have lost their ability to move, and rescue can be sent to them in a timely manner. The device also helps the commander receive information about the environment so that they can perceive the situation inside the building and make data-driven decisions accordingly without going into the field.

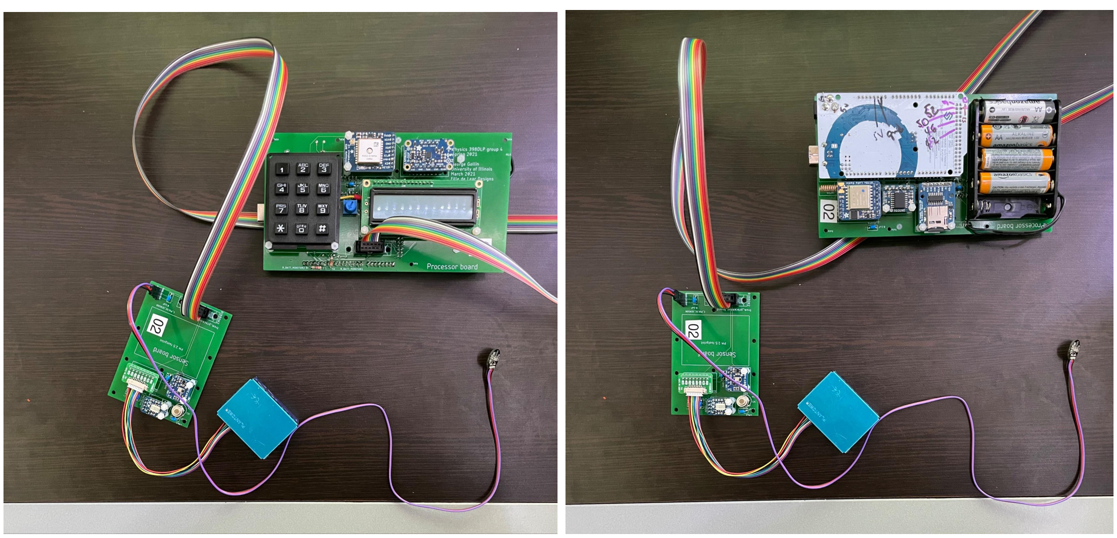
2. Methods and Procedures

We connected the sensors on our breadboards to make sure they worked when we were working on our data acquisition program. Part of the group was working on integrating sensors, and others worked on making radio communications work using the LoRa radios and creating a base station. After we made sure it all worked together, Professor Gollin designed PCBs that included all the components. The backpacks consisted of processor boards connected to sensor boards via a ribbon cable. The base station was separate, and the backpacks could communicate with the base stations using the LoRa radios.

2.1 Hardware

Our project is composed of two major components, the backpacks and the base station. The backpack, which will be on the firefighters, is designed to be responsible for collecting and recording different kinds of data, sending relevant data to the base station and sending as well as receiving simple SOS messages if required. The base station receives live data from the firefighter's backpacks and outputs useful information to a crew chief, enabling better communication and awareness of potential health and safety hazards. In case of emergency situations, it also relays an SOS message so that the necessary actions may be taken.

2.2 THE BACKPACKS



Figures 1 and 2: Final Backpack PCB

As mentioned above, firefighters will be equipped with the backpack while firefighting and hence will need to have the devices required in order to collect the data and communicate with the base station. As mentioned in Section 1, the data which we aim to collect in real-time is environmental data including the temperature, hazardous gas particle concentrations, particulate matter concentrations, heart rate of the firefighters, and their locations. In order to get these readings, we have used the various sensors and the Global Positioning System (GPS) module provided to us for the course. After collecting the essential data and storing it on an SD card using an SD Card reader module, the backpacks are designed to transmit all the relevant data to the base station for both present and future use. This includes the recorded anomalies and the SOS alerts which are also relayed to other backpacks in cases when immediate assistance is required. Since this will be a fairly large range, we used LoRa radios for the backpacks to communicate with the base stations.

For our project we designed a two-backpack system to illustrate their communication with the base station without any interference. In order to avoid any alteration of the message, we designed the backpacks to send messages at different fixed time intervals in a periodic fashion, so that the base station only has one message to receive at any given time. For this we used a Real Time Clock (RTC) for each of the backpacks and set them to the current time using the satellite data received by the GPS module.

In order to assimilate, manage and add logic to the sensors, the SD card reader module, the GPS module, and the RTC and to serve the purpose of a control unit for the backpacks, we have used a MicroController Unit (MCU). The backpacks are supposed to be with the firefighters and hence are designed to be mobile. As an output interface, we have used a Liquid Crystal Display (LCD) to show warnings and error messages received by the base station and a keypad for the firefighters to send SOS alerts. Since the GPS fails to find satellites in the case of overhead obstruction, it is not feasible to use the GPS to determine the location of a firefighter if they are indoors. Hence, we used an LSM breakout to calculate the acceleration and change in direction of the motion and integrated the acceleration readings over time once to find their velocity and twice to find their position.

The backpack can also be used to provide the firefighter with information in real time. There is an LCD on the backpack, and the DAQ prints temperature from the IR sensor as well as heart rate on the display. We printed these measurements as opposed to others because we thought they were the most important. A firefighter might want to know the temperature of an object, such as a door, before opening it or touching it in any way. All they have to do is point the IR thermometer at the object, and instantaneously they can read the measurement on the liquid crystal display. Similarly, a firefighter might want to keep track of their heart rate to make sure it is within normal ranges. Measurements from other sensors are also important, but we thought these two would be most useful to firefighters in real time. We chose not to scroll through other measurements, so that whenever a firefighter wants to know the temperature of an object or wants to know their heart beat, they don't have to wait for that information to show up on the display. The figure below shows the LCD display.

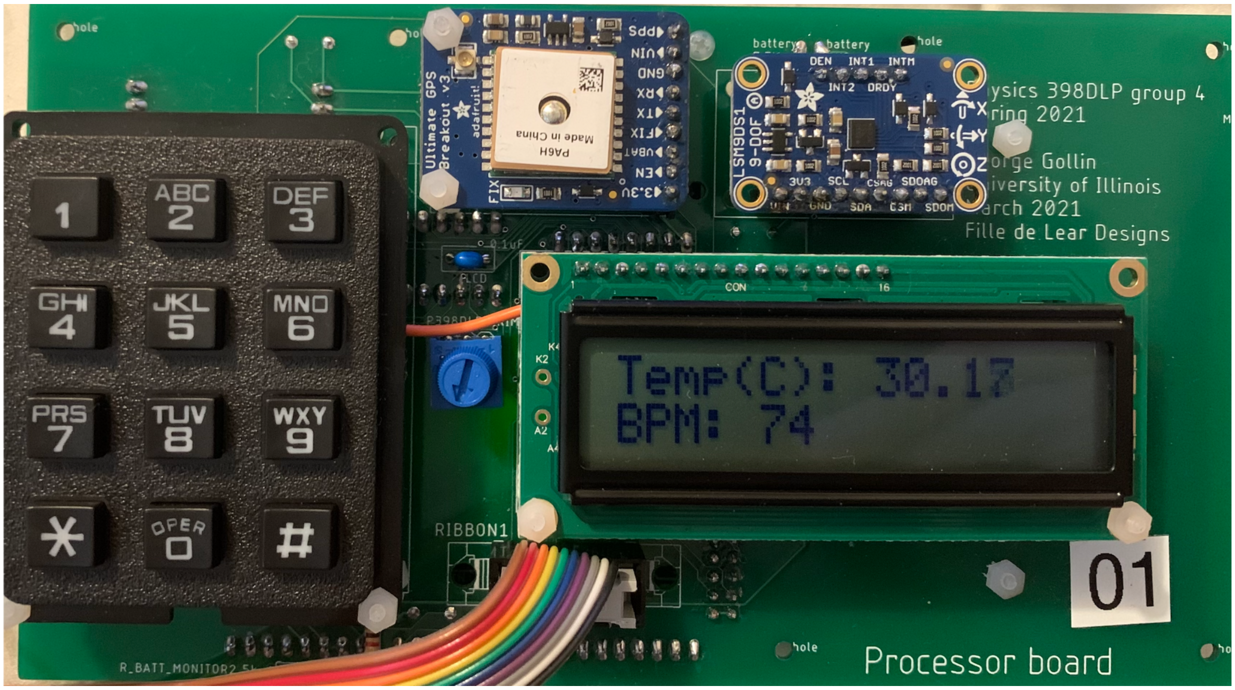


Figure 3: LCD Display on the PCB

DEVICES USED	MODEL	APPLICATION	DEVICE CATEGORY
Bosch Environment Sensor	BME 680	Temperature, Humidity, Pressure Volatile Organic Compounds (VOC)	Sensor
Carbon Monoxide Sensor	MiCS5524	CO, alcohol, VOC gas sensor	Sensor
Airborne Particulate Monitor	PM 2.5 PMS5003	Measure concentration of Particulate matter	Sensor
Pulse Sensor	Pulse Sensor Amped	Measure heart rate	Sensor
IMU sensor	LSM9DS1	Accelerometer Magnetometer Gyroscope	Sensor

LCD (16 x 2)	CFAH1602B-NGG-JTV	Display alerts and anomalies in readings	I/O
3 x 4 Keypad	3x4 Keypad #3845	Used as an input by firefighter to send SOS alerts	I/O
GPS	Ultimate GPS Breakout Board #746	Used to get the current time and date and sync RTCs with each other.	Breakouts
MicroSD card editor	MicroSD card breakout	Writes and reads on a microSD card	Breakouts
Long Range Radio	Adafruit RFM95W	Used to send and receive messages to and from the base station	Breakouts
Arduino MCU	Arduino Mega 2560	Microcontroller for overall coordination	Microcontroller Unit
Real Time Clock	DS 32321	For the periodic transmission of new messages from all backpacks one at a time.	Generates Synchronous Signals

Table 1: Devices on the Base Station PCB

2.3 Backpack Sensors

2.3.1 Arduino Mega 2560

The Arduino Mega 2560 is a microcontroller based on the ATmega2560. It has 54 digital Input/Output (I/O) pins, 15 of which can be used as PWM outputs, 16 analog inputs, 4 UART hardware serial ports, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. We used the Arduino Mega 2560 as the MCU for all of our sensors and the radio communication program.

2.3.2 RFM9x Long-Range (LoRa) Radio

LoRa radios are a radio communication technology that provides long range, low power data transmission based on spread spectrum modulation techniques. An RFM9x long range radio chip is used in our project to transmit data from the backpacks.

2.3.3 DS3231 Real Time Clock (RTC)

The DS3231 RTC is an extremely accurate and reliable chip that keeps track of the current time. The module stores seconds, minutes, hours, day, date, month, and year information. It is used in our project because the GPS timestamps are unreliable. The RTC's battery allows the RTC to keep track of the correct time even when the power is cut off. Each backpack also sends data at a given interval, so the RTC ensures that the signals are only sent at the scheduled time so that they don't interfere with each other.

2.3.4 Global Positioning System (GPS)

GPS is a positioning service owned and operated by the United States. It consists of 31 satellites that send location information to the user, with at least 24 satellites available for the majority of the time. In our project, we employed a GPS to keep track of the location of the firefighters.

2.3.5 Secure Digit (SD) Card

An SD card is a memory card that can store up to several terabytes worth of information. We used an SD card to store the relevant data including temperature, hazardous gas particles, particulate matter concentrations in the air, heart rate of the firefighters, and their locations.

2.3.6 BME 680

The BME 680 is an integrated environmental sensor that is capable of sensing temperature, humidity, barometric pressure, and VOC. A BME 680 is used in our project to collect the crucial environmental data that are relevant to firefighting.

2.3.7 PMS5003

The PMS5003 particulate matter sensor uses light diffraction to sense particles in the air. It consists of a piece that plugs into the sensor board and is connected to a box by wire. This box has a fan that brings air into the box, uses a laser to diffract the particles in the air, and measures the concentration of particles greater than 1, 2.5, and 10 micrometers in diameter in units of micrograms per cubic meter based on the scattering of the particles. It also measures the amount of particles greater than .3, .5, 1, 2.5, 5, and 10 micrometers in diameter per .1 liters. We are using this sensor in our project to measure the concentrations and amount of particles in the firefighting environment.

2.3.8 LSM9DS1 9-DOF

The LSM9DS1 integrates three separate sensors together—an accelerometer, a magnetometer, and a gyroscope. Together, these sensors allow us to measure motion, direction, and orientation. The 9 DOF come from the fact that each sensor has 3 degrees of freedom. We use this sensor in our project to collect acceleration information on the firefighters, and to attempt to integrate to find velocity and position.

2.3.9 MLX90614 IR Thermometer

The MLX90614 is a non-contact thermometer that senses the IR energy emitted from an object and converts it into that object's temperature. Should a firefighter want to know the temperature of a specific object, this is more useful than a BME680, as they simply need to point it at the object to obtain its temperature. We use this sensor to record temperature, and also to display the temperature on the backpack's LCD should the firefighter want to know what the temperature of a specific object is.

2.3.10 Liquid Crystal Display (LCD)

The LCD allows us to display real time information to the firefighters, such as temperature measured by the MLX90614 IR thermometer and heart rate measured by the pulse sensor. There are LCDs on the backpacks as well as on the base station, where they can display vitals and alerts. The LCDs used have 2 rows that are both 16 characters wide. The maximum message size that can be displayed at any given time is 32 characters long.

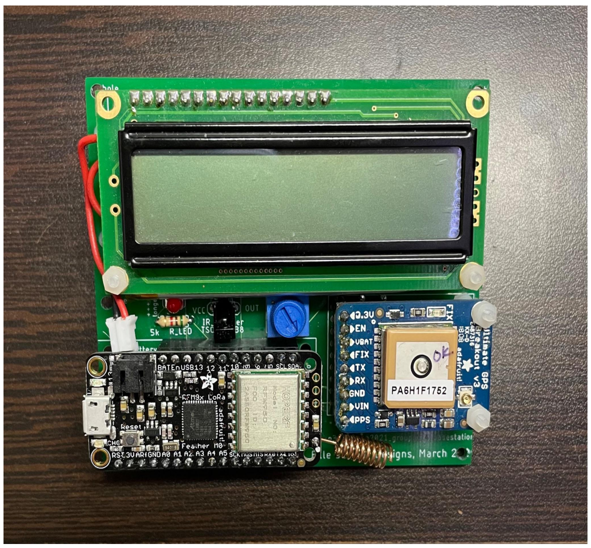
2.3.11 MiCS5524

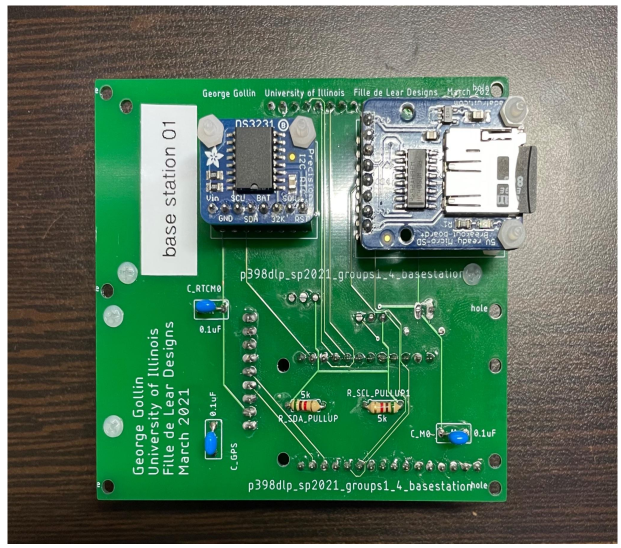
The MiCS5524 is a sensor that measures the amount of certain gasses in parts per million (ppm). The back of the sensor shows that it can measure carbon monoxide (CO) from 1 to 1000 ppm, ethanol (C_2H_5OH) from 10-500 ppm, hydrogen (H_2) from 1-1000 ppm, ammonia gas (NH_3) from 1-500 ppm, and methane gas (CH_4), but only at amounts greater than 1000 ppm. This sensor cannot tell you which gas is detected. We are using this sensor to measure the amount of these gases in the firefighting environment.

2.3.12 Pulse Sensor Amped

The pulse sensor amped is a sensor that consists of three long wires attached to a small, round sensor. This sensor can be attached to a finger using a finger strap, or to an ear using an ear clip (this is the method we are employing with the backpacks). We are using this sensor to measure the firefighter’s heart beat in beats per minute (bpm) and to alert the base station if this measurement is too low or too high.

2.4 The Base Station





Figures 4 and 5: Final Base Station PCB

The Base Station has been designed to be operated by the supervisor on the ground, within two kilometres of where the fire is happening. It is responsible for receiving readings and warnings from the backpacks and relaying alerts back to the backpacks. It has an LCD to display the warning messages and an SD card editor to store the readings for future analysis. Unlike the backpacks, the base station is equipped with a Feather M0 with a 915 MHz radio instead of a Mega 2560 and LoRa radio combination. The base station is not required to record any readings and thus has no sensors on it. The base station is also equipped with an RTC which is synced to the satellite time received by the GPS on the Base station PCB.

DEVICES USED	MODEL	APPLICATION	DEVICE CATEGORY
LCD (16 x 2)	CFAH1602B-NGG-JTV	Display alerts and anomalies in readings	I/O
Feather M0	Feather M0 with 915 MHz radio #3178	Used as an MCU as well as a radio to communicate with the backpacks	MCU + Radio
MicroSD card editor	MicroSD card breakout	Writes and reads on a microSD card	Breakouts
Real Time Clock	DS 32321	To transmit the message in a synchronous periodic fashion	Generates Synchronous Signals

Table 2: Devices on the Base Station PCB

2.5 Communication

As an initial model of communication between the backpacks and the base station, we decided to have every single backpack, as well as the base station listen in for a signal from the the previous backpack (with 'previous' being defined as the backpack with the ID number being one less), and send its data to the base station. Each backpack would not send its own signal until it hears from the previous backpack. The algorithm for the backpack communication would look something like the following:

1) Listen for a signal from the previous backpack. Identify a magic message, which is a special set of characters to recognise the following bits as a valid message, and the backpack ID of the received signal.
2) Once it is the current backpack's turn to send data, send the recorded data to the base station and all other backpacks, with the magic message and backpack ID attached to the front.

We recognised that this model of communication had some potential issues which are discussed in Section 5. We then came up with the following revised model of communication. Every backpack sends the data signal to the base station at a regular time interval, regardless of what the other backpacks are doing. The formula to calculate if the i*th* backpack sends a signal in a squad of size *n,* where time elapsed since the start of the program is *t*, is given by the following: if $\{(t) \bmod (n) == i\}$, send a signal to base.

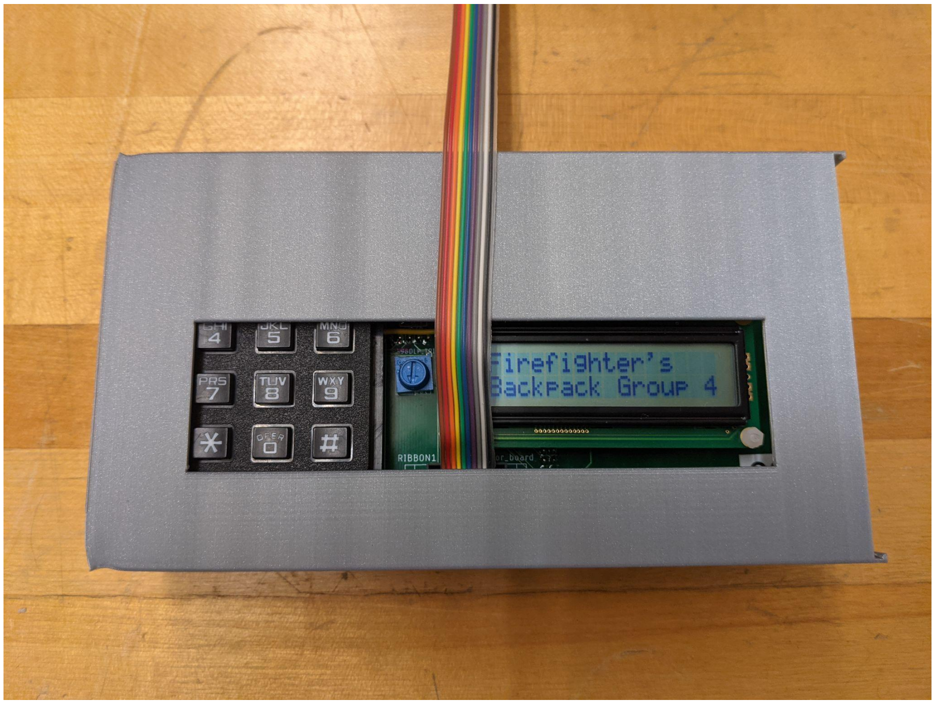


Figure 6: Backpack processor board in the 3D printed case

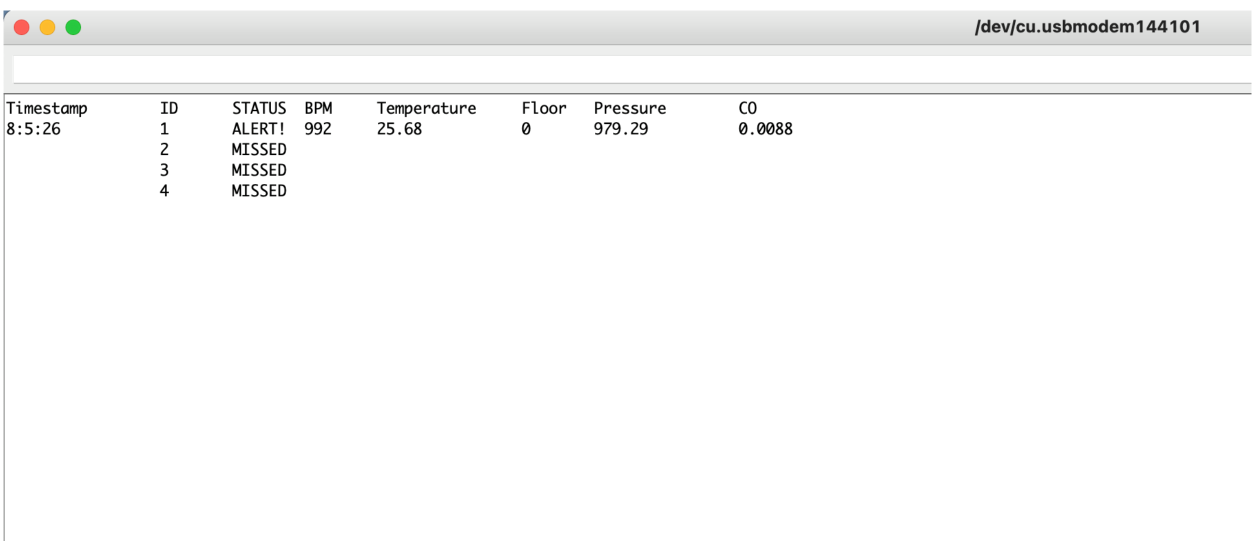


Figure 7: Output from the base station serial monitor screen

3. Data Acquisition

The data taken from each of these sensors was stored in CSV format. The top line contains the header names of each column. The data from each of the backpack's sensors was concatenated into one long string that was one row of the CSV data file. The string began with RTC and GPS date and time information separated by commas for each data field. Each time we added a new sensor to the DAQ the new data was concatenated onto this one string variable and separated by commas which we were able to send over radio back to the base station relatively easily. Because there was only one string variable to send for all of the backpack's sensors we simply had to allocate enough space in the radio transmission for a string variable of a few hundred characters. When the base station received a new string variable from the backpacks they were appended to the proper CSV data file. In the event that one or more of the sensors did not have valid data to concatenate onto the string variable zeros and commas were used in lieu of valid data to keep all of the CSV's rows the proper length. This occurred either when the GPS did not have a connection to enough satellites or when the timing of the data acquisition done by the particulate matter sensor was not in sync with the timing of the DAQ's loop function. A screenshot of the serial monitor output is included below in Figure 8. Our data CSVs consisted of 37 columns in total, some of which may have had zeros instead of valid data. The right side of the serial monitor output has been cropped out for readability.

```
rtcDate,rtcTime,gpsDate,gpsTime,Latitude (Degrees),Latitude Direction,Longitude (Degrees),Longitude Direction,Speed (m/s),Altitude (Meters)
2021-4-23,16:22:46,0,0,0,0,0,0,0,167.10,0,22.84,993.38,24.99,228.46,0,0.0117,2,3,3,534,146,17,2,0,0,1.82,-2.98,9.29,22.13,27.84,-19.38,0.01
2021-4-23,16:22:47,0,0,0,0,0,0,0,166.93,0,23.29,993.36,24.92,247.89,118,0.0108,2,3,3,531,143,18,2,0,0,1.76,-3.02,9.20,22.76,27.65,-20.04,0.
2021-4-23,16:22:49,0,0,0,0,0,0,0,166.93,0,23.39,993.36,24.76,255.19,110,0.5024,0,0,0,0,0,0,0,0,1.79,-2.97,9.16,22.19,27.59,-19.56,0.01,0.
2021-4-23,16:22:50,0,0,0,0,0,0,0,166.59,0,23.60,993.40,24.65,265.05,127,0.0117,1,2,2,531,148,22,2,0,0,1.75,-3.00,9.22,22.45,27.52,-20.15,0.
2021-4-23,16:22:51,0,0,0,0,0,0,0,166.93,0,23.72,993.38,24.50,270.93,176,0.0127,2,3,3,531,148,22,2,0,0,1.76,-3.00,9.23,21.81,27.40,-19.60,0.
2021-4-23,16:22:53,0,0,0,0,0,0,0,166.59,0,23.80,993.38,24.36,278.02,189,0.0117,2,3,3,546,153,22,2,0,0,1.73,-2.99,9.16,22.35,27.30,-18.69,0.
2021-4-23,16:22:54,0,0,0,0,0,0,0,166.76,0,23.80,993.38,24.20,280.63,187,0.0117,0,0,0,0,0,0,0,0,1.81,-2.97,9.26,21.97,27.92,-19.67,0.01,0.
2021-4-23,16:22:56,0,0,0,0,0,0,0,166.93,0,23.85,993.40,24.07,285.74,172,0.0117,2,3,3,534,149,17,2,0,0,1.76,-2.98,9.11,21.86,27.55,-20.18,0.
2021-4-23,16:22:57,0,0,0,0,0,0,0,166.76,0,23.89,993.40,23.97,288.75,162,0.0117,2,3,3,540,145,15,2,0,0,1.77,-3.01,9.23,22.77,27.81,-19.34,0.
2021-4-23,16:22:59,0,0,0,0,0,0,0,166.59,0,23.78,993.38,23.81,291.30,147,0.0117,0,0,0,0,0,0,0,0,1.79,-2.99,9.13,22.04,27.46,-19.25,0.01,0.
```

Figure 8: Serial Monitor Output

The date and timestamps from the RTC and the GPS were logged in our data files in order to plot the various sensor data on 2D graphs with sensor data on the y-axis with time on the x-axis. The latitude and longitude from the GPS was logged in our data files and the altitude in meters was calculated from the barometric pressure readout from the BME680. The latitude and longitude readouts were converted to meters using Equation (1). These three coordinates were then used to plot our sensor data on 3D scatterplots with a colormap and colorbar legend representing how the sensor data readouts varied with position.

$$\Delta L = 2\pi\Delta\theta/360^\circ \qquad (1)$$

Equation 1 is an exact equation for degrees longitude, but only an approximation for degrees of latitude. Fortunately the error from this approximation is only apparent on length scales of several kilometers. Because this device is not intended to be used on scales that large the approximation was left in. On the scales that a firefighter would be using our device this approximation when converting from degrees of latitude to meters is negligible.

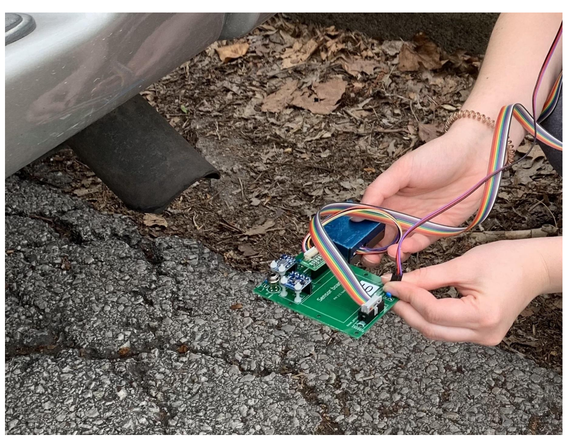
4. Results

4.1 Sensor Data Versus Time

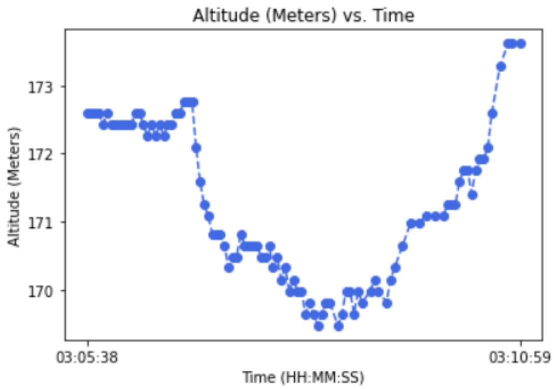
The following two sections plot various sensor data versus time under different circumstances that were intended to simulate firefighting conditions. We exposed our firefighter's backpacks to car exhaust and incense sticks which were relatively safer alternatives to the smoke of a typical house fire that a firefighter would encounter.

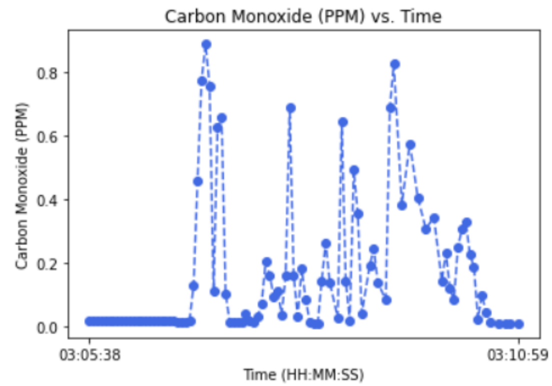
4.1.1 Car Exhaust

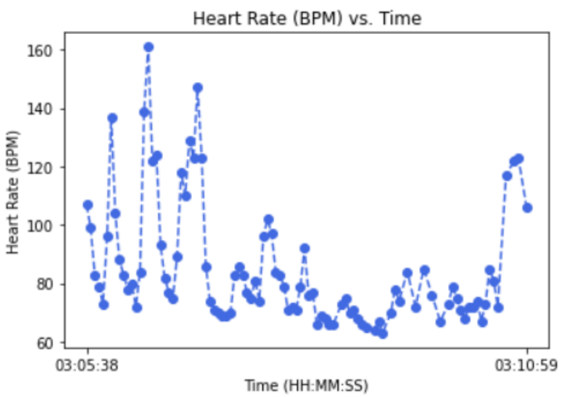
The next section shows how the atmospheric conditions changed due to car exhaust. The car had just been driven, and it stayed running during the test. The PCB was taken out of the passenger side of the car and brought to the exhaust pipe. It was a bit tough to make sure the particulate matter sensor and the gas sensor were taking in car exhaust and not just air outside, as it was a bit windy. The device was held about six inches away from the exhaust pipe. Figures 9 and 10 illustrate how this data was taken and figures 11-26 below show how our various sensors responded.

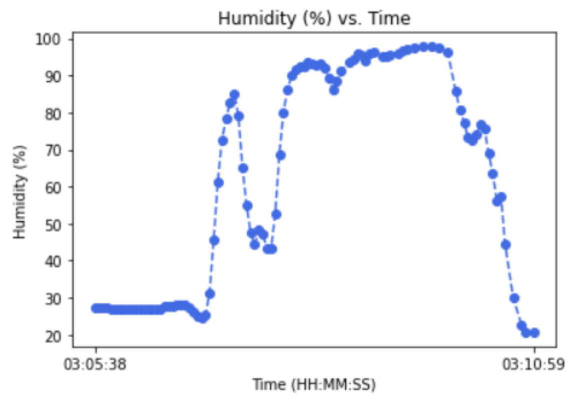


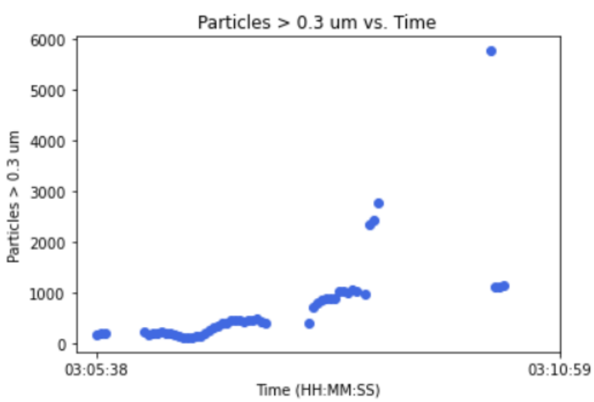
Figures 9 and 10: Car Exhaust Data

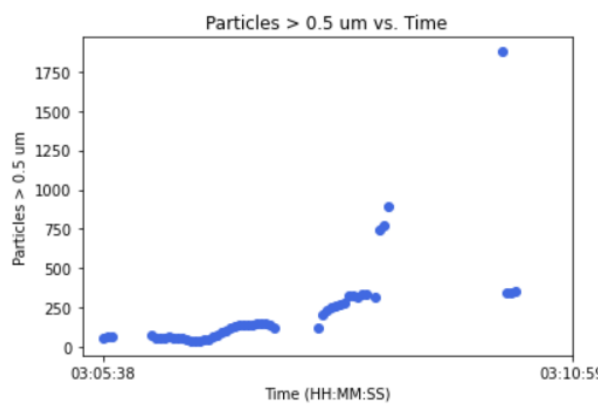

Altitude (Meters) vs. Time
Altitude (Meters)
173
172
171
170
03:05:38
03:10:59
Time (HH:MM:SS)

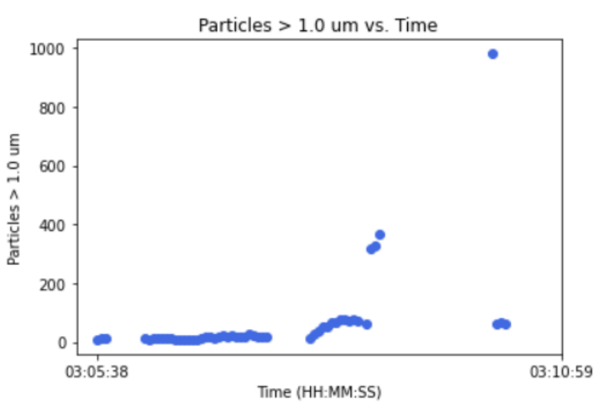

Carbon Monoxide (PPM) vs. Time
Carbon Monoxide (PPM)
0.8
0.6
0.4
0.2
0.0
03:05:38
03:10:59
Time (HH:MM:SS)

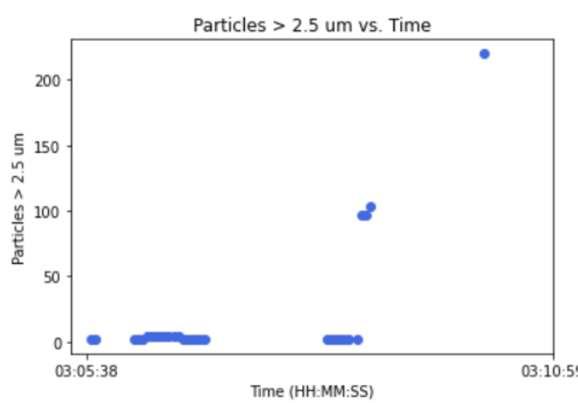

Heart Rate (BPM) vs. Time
Heart Rate (BPM)
160
140
120
100
80
60
03:05:38
03:10:59
Time (HH:MM:SS)

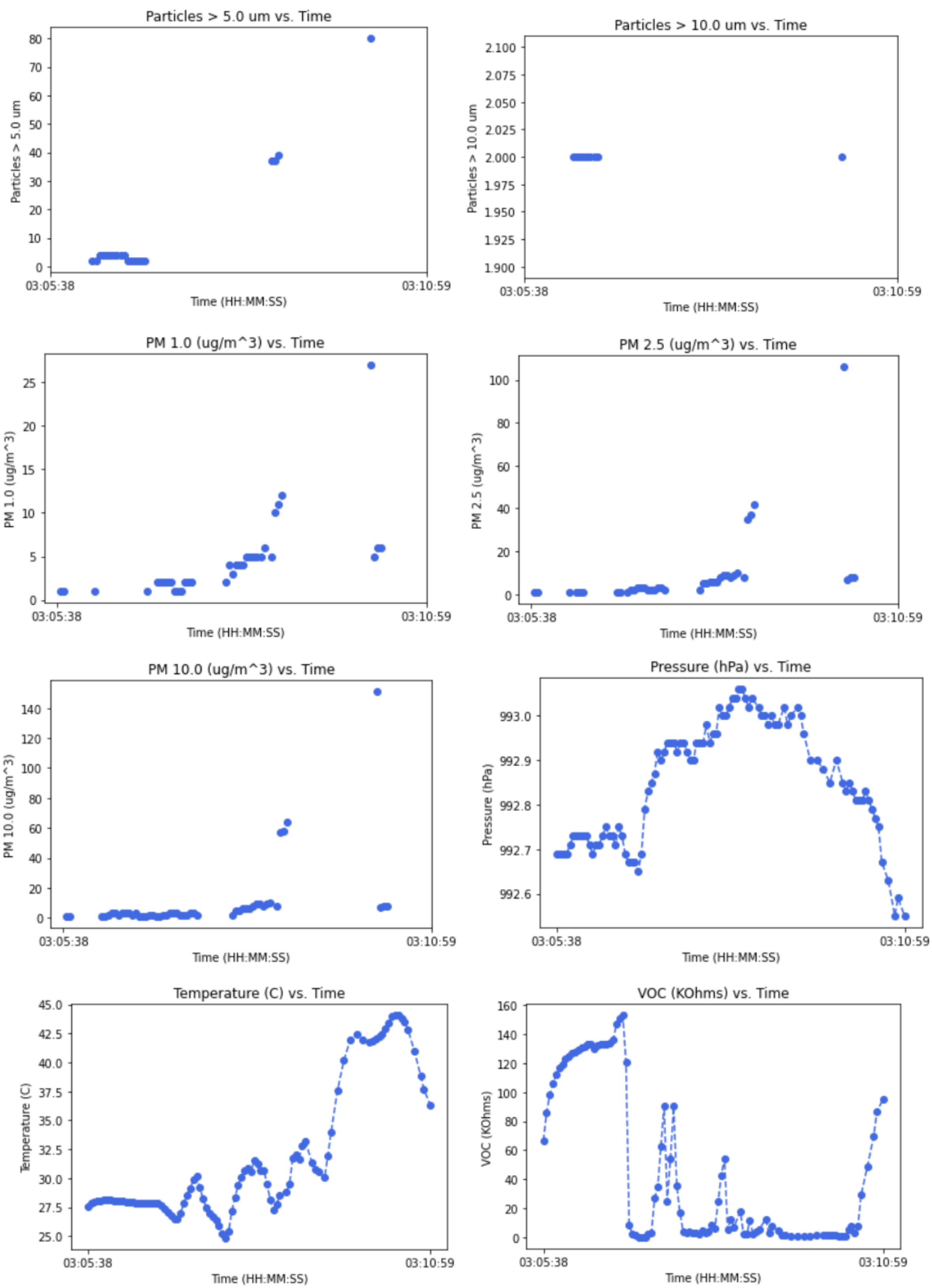

Humidity (%) vs. Time
Humidity (%)
100
90
80
70
60
50
40
30
20
03:05:38
03:10:59
Time (HH:MM:SS)


Particles > 0.3 um vs. Time
Particles > 0.3 um
6000
5000
4000
3000
2000
1000
0
03:05:38
03:10:59
Time (HH:MM:SS)


Particles > 0.5 um vs. Time
Particles > 0.5 um
1750
1500
1250
1000
750
500
250
0
03:05:38
03:10:59
Time (HH:MM:SS)


Particles > 1.0 um vs. Time
Particles > 1.0 um
1000
800
600
400
200
0
03:05:38
03:10:59
Time (HH:MM:SS)


Particles > 2.5 um vs. Time
Particles > 2.5 um
200
150
100
50
0
03:05:38
03:10:59
Time (HH:MM:SS)



Figures 11-26: Car Exhaust Sensor Data

The particulate matter sensor had a few periods where it did not output any data. Instead zeros were logged in our data file in lieu of valid sensor data. For clarity when reading the graphs these zeros have not been plotted. All the particle sizes slowly increased at the beginning of each graph, and then have two spikes. In between these two spikes is when the sensor stopped working. However, it is still easy to see that both concentrations of particulate matter and temperature increased. The count of particulate matter started off relatively low, and rose to 5,763 for particles of diameter greater than .3 micrometers. The concentration of particles 10 micrometers or less in diameter rose from 1 to 151. The measurement of carbon monoxide in ppm is highly variable, which might have resulted from the fact that measurements were taken outside and some wind was present, which could be interfering with the sensor measuring car exhaust. VOC makes sense, as it decreased in the presence of the exhaust. Altitude makes sense, because the PCB was brought down from the passenger seat to the curb and the car is quite tall.

For the heart rate sensor data, nothing strenuous was being done, nor was there any anxiety, so the heart rate should be near normal levels rather than 160 beats per minute. Since the sensor connects via an ear clip, it is possible that hair touching the clip could have somehow impacted the measurements. However, an additional problem that was causing this irregular, not smooth heartbeat value was the threshold variable. The pulse sensor is a simple device, with only three wires. One goes to power, the other goes to ground, and the third goes into an analog pin on the Mega 2560 (in our case it was analog pin 1). Since the pulse sensor works through an analog pin, the input value will be a value between 0 V and 5 V, and this will be converted to an integer value between 0 and 1023. The threshold value determines what input value should be considered a “heartbeat”. The manufacturers created a library for the pulse sensor, so it easily finds the heartbeat in Beats Per Minute (BPM). To determine this threshold value, they also developed a demo piece of code that helps to determine the threshold value. For the sensors wired up to the breadboard, the code showed that this threshold value was 435. We kept this threshold value the same when we integrated the pulse sensor into the DAQ, only to later realize it was different for the pulse sensor in the PCB. This is probably due to the fact that the PCB had a lot more items wired up than the breadboard did. When we ran the demo code again, a new threshold value of 500 was found, and later tests showed this value to provide a more accurate measurement of heart rate. The reason a threshold value of 435 gave such an erratic and high heart rate compared to 500 is that the analog input value hit 435 more frequently than 500, so it counted heartbeats more frequently than it should have.

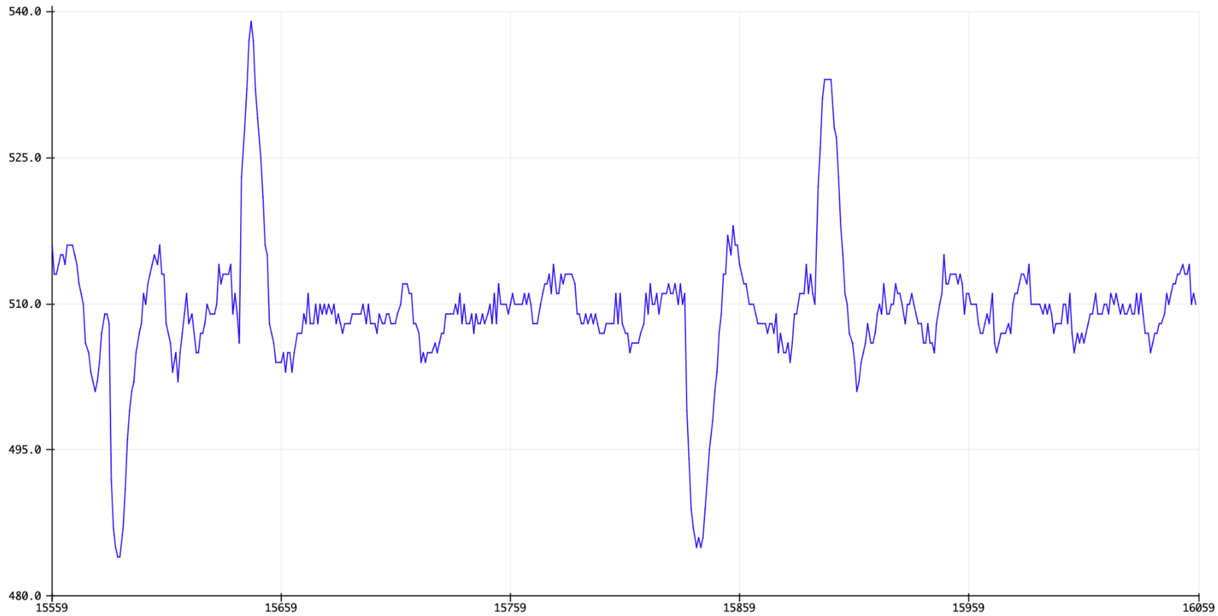


Figure 27: Pulse Sensor Analog Input

The figure above is the analog output on the serial plotter, showing a heartbeat. The threshold value of 500 means that every time the analog input hits 500, it counts a heartbeat. This happens at the beginning of the heartbeat. There is a spike downward, where it hits 500. After the program identifies a heartbeat, it delays for a very short amount of time so that the program does not count a heartbeat on the way up from the spike. The input does not go down to 435 when the pulse sensor is on the PCB, hence the inaccurate reading when the threshold value is 435.

Figure 28: Heart Rate Output

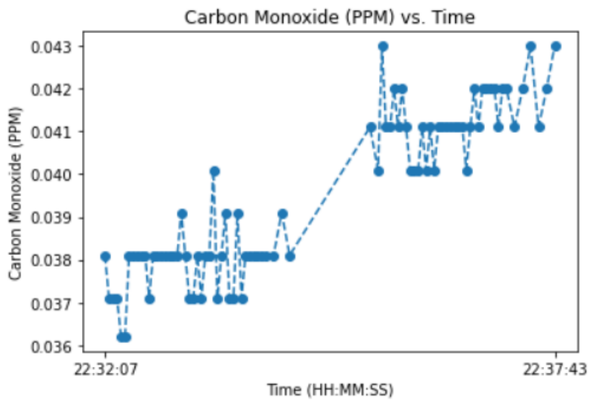
When the threshold value makes sense (in this case it is 500), heartbeats are counted correctly and are not irregular, and the screenshot of the serial monitor heartbeat output above shows this.

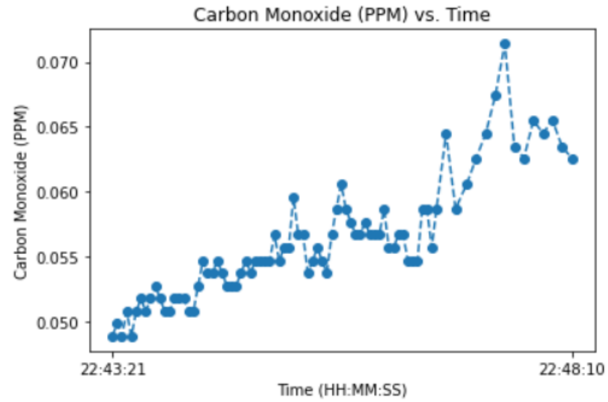
4.1.2 Incense

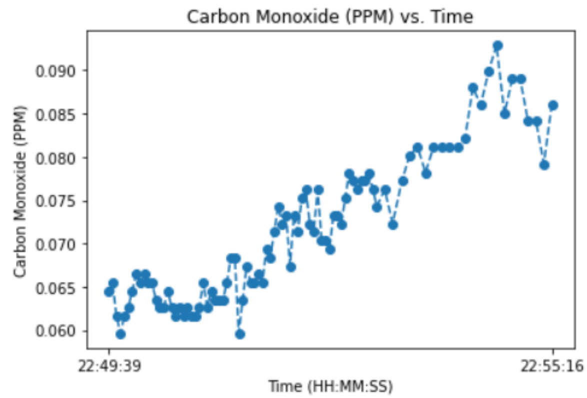
The next section shows how the atmospheric conditions changed due to burning Egyptian Jasmine incense in a confined space. The space measured (5 ft)(5 ft)(8 ft) = 200 ft^3. The air vent to the room was closed to prevent the incense smoke from dispersing. Data acquisition was done after lighting 1, 3, and 5 incense sticks. The three pictures, Figures 29-31, and the corresponding graphs, Figures 32-73, show how the various sensors responded to 1 incense stick (top left), 3 incense sticks (top right), and 5 incense sticks (bottom middle).

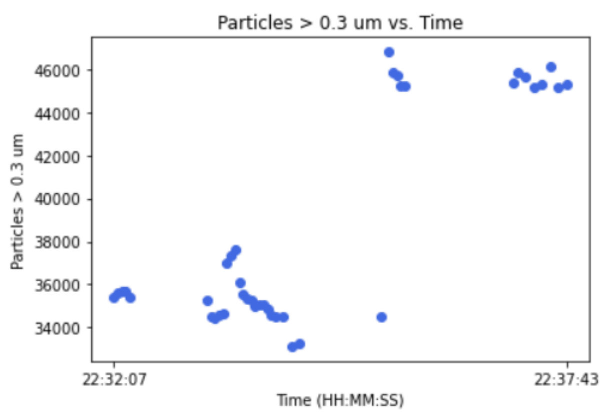


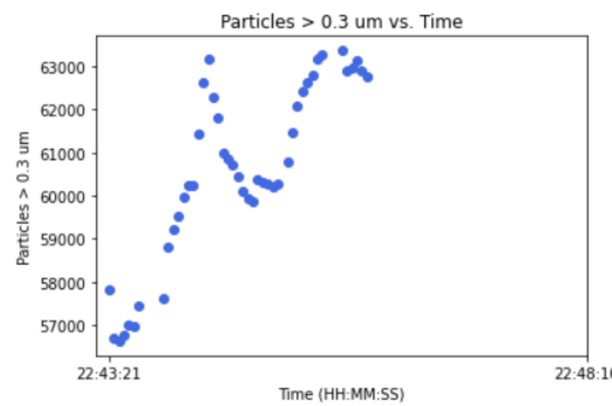
Figures 29-31: 1, 3, and 5 Egyptian Jasmine Incense Sticks

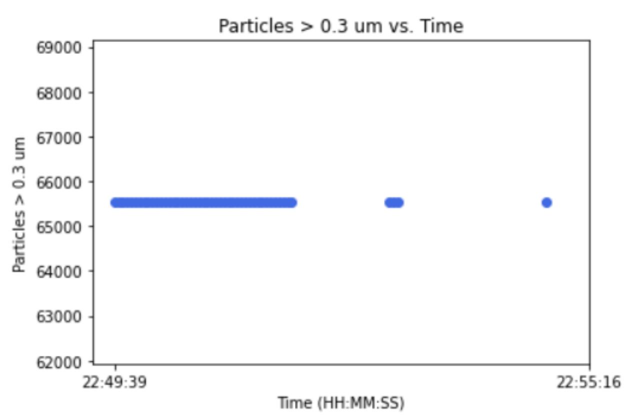

Carbon Monoxide (PPM) vs. Time
Carbon Monoxide (PPM)
0.043
0.042
0.041
0.040
0.039
0.038
0.037
0.036
22:32:07
22:37:43
Time (HH:MM:SS)

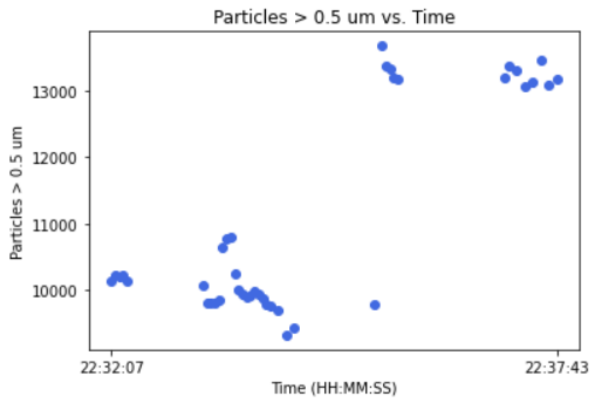

Carbon Monoxide (PPM) vs. Time
Carbon Monoxide (PPM)
0.070
0.065
0.060
0.055
0.050
22:43:21
22:48:10
Time (HH:MM:SS)

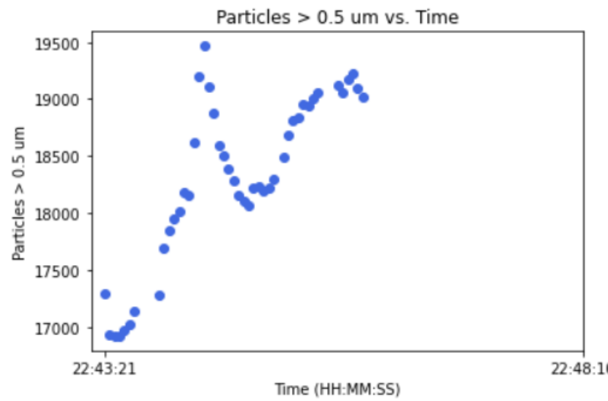

Carbon Monoxide (PPM) vs. Time
Carbon Monoxide (PPM)
0.090
0.085
0.080
0.075
0.070
0.065
0.060
22:49:39
22:55:16
Time (HH:MM:SS)

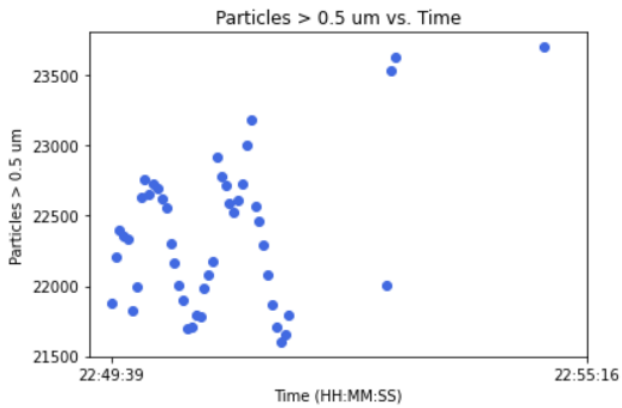

Particles > 0.3 um vs. Time
Particles > 0.3 um
46000
44000
42000
40000
38000
36000
34000
22:32:07
22:37:43
Time (HH:MM:SS)

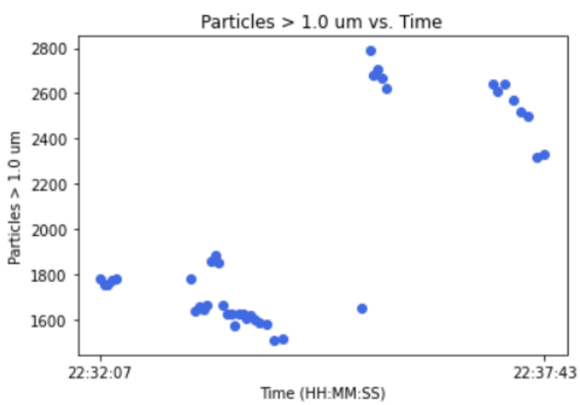

Particles > 0.3 um vs. Time
Particles > 0.3 um
63000
62000
61000
60000
59000
58000
57000
22:43:21
22:48:10
Time (HH:MM:SS)

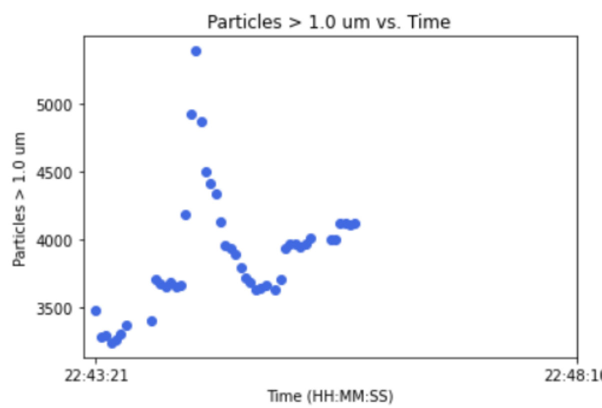

Particles > 0.3 um vs. Time
Particles > 0.3 um
69000
68000
67000
66000
65000
64000
63000
62000
22:49:39
22:55:16
Time (HH:MM:SS)

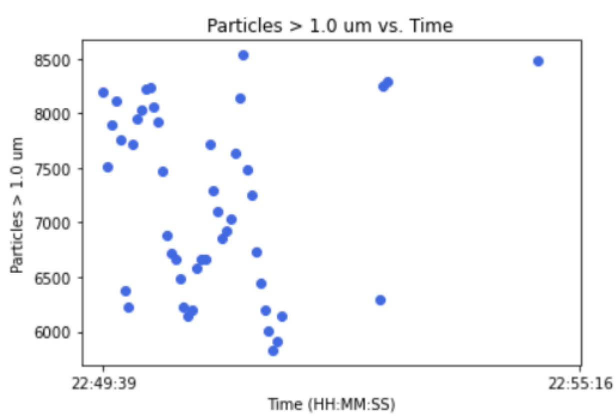

Particles > 0.5 um vs. Time
Particles > 0.5 um
Time (HH:MM:SS)
22:32:07
22:37:43
13000
12000
11000
10000

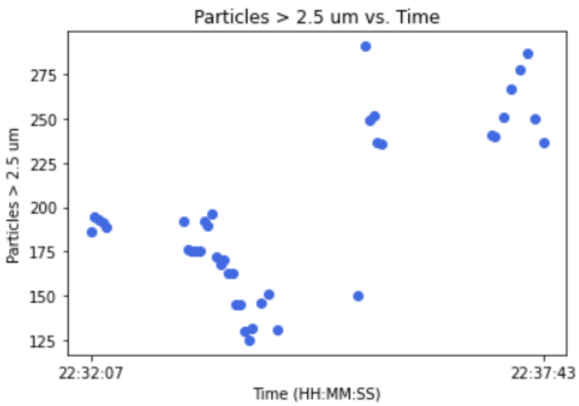

Particles > 0.5 um vs. Time
Particles > 0.5 um
Time (HH:MM:SS)
22:43:21
22:48:10
19500
19000
18500
18000
17500
17000

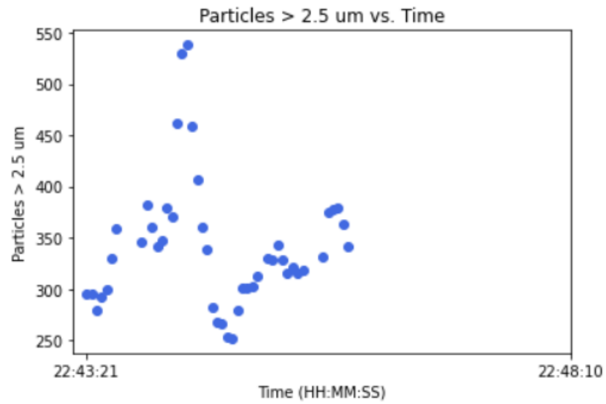

Particles > 0.5 um vs. Time
Particles > 0.5 um
Time (HH:MM:SS)
22:49:39
22:55:16
23500
23000
22500
22000
21500

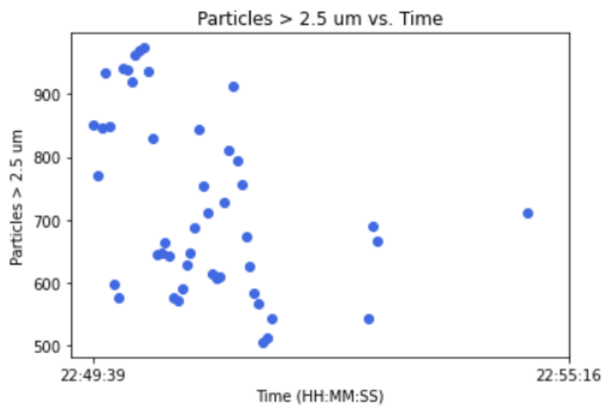

Particles > 1.0 um vs. Time
Particles > 1.0 um
Time (HH:MM:SS)
22:32:07
22:37:43
2800
2600
2400
2200
2000
1800
1600

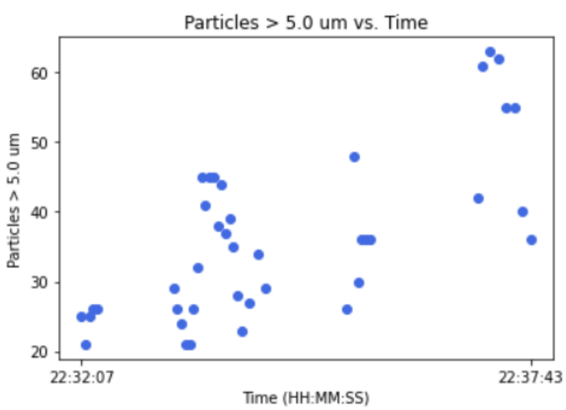

Particles > 1.0 um vs. Time
Particles > 1.0 um
Time (HH:MM:SS)
22:43:21
22:48:10
5000
4500
4000
3500

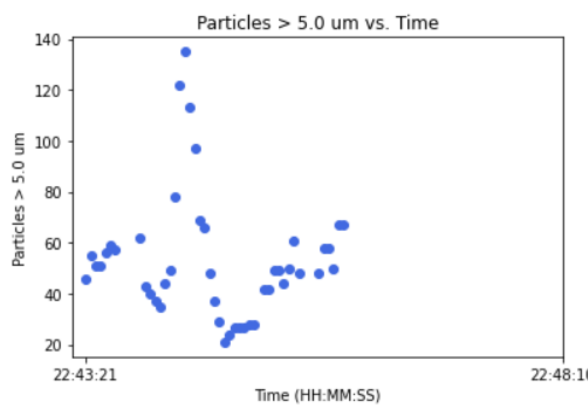

Particles > 1.0 um vs. Time
Particles > 1.0 um
Time (HH:MM:SS)
22:49:39
22:55:16
8500
8000
7500
7000
6500
6000

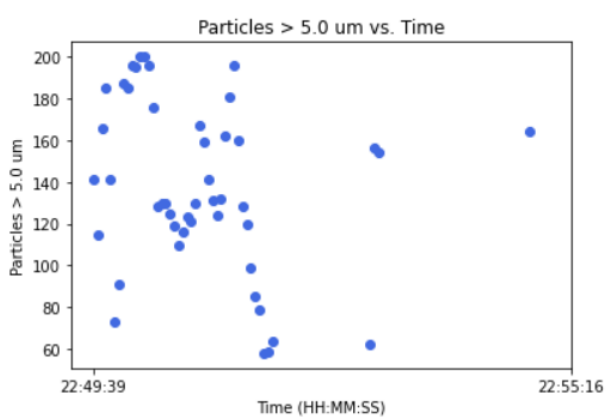

Particles > 2.5 um vs. Time
Particles > 2.5 um
Time (HH:MM:SS)
22:32:07
22:37:43

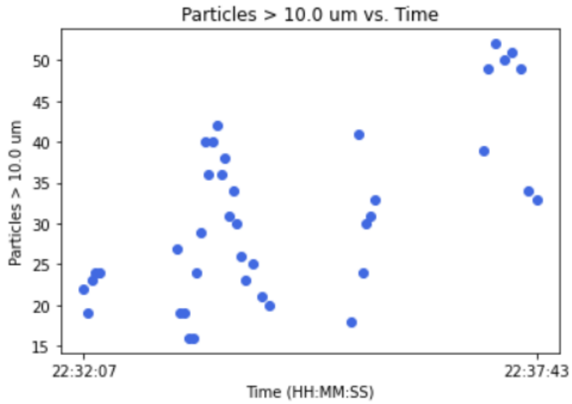

Particles > 2.5 um vs. Time
Particles > 2.5 um
Time (HH:MM:SS)
22:43:21
22:48:10

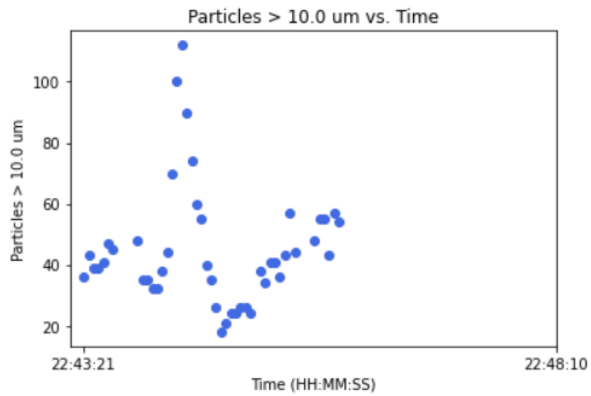

Particles > 2.5 um vs. Time
Particles > 2.5 um
Time (HH:MM:SS)
22:49:39
22:55:16

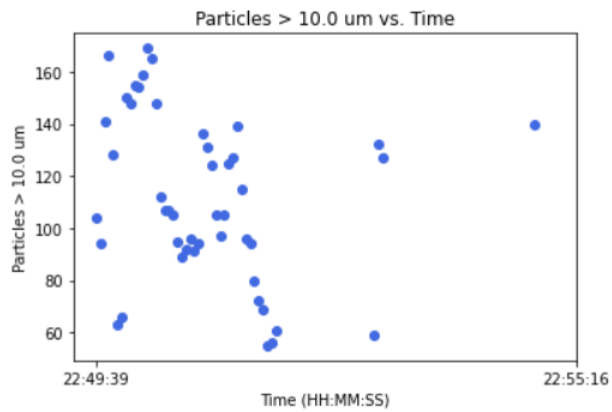

Particles > 5.0 um vs. Time
Particles > 5.0 um
Time (HH:MM:SS)
22:32:07
22:37:43

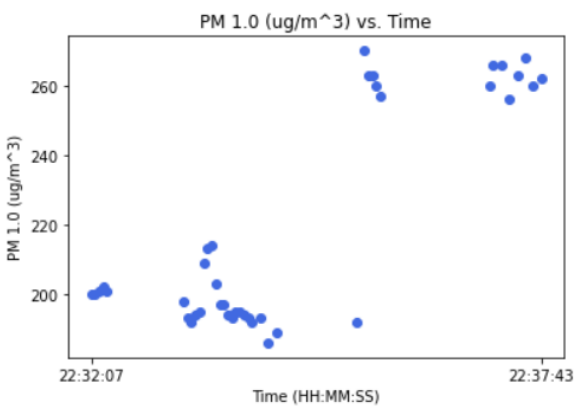

Particles > 5.0 um vs. Time
Particles > 5.0 um
Time (HH:MM:SS)
22:43:21
22:48:10

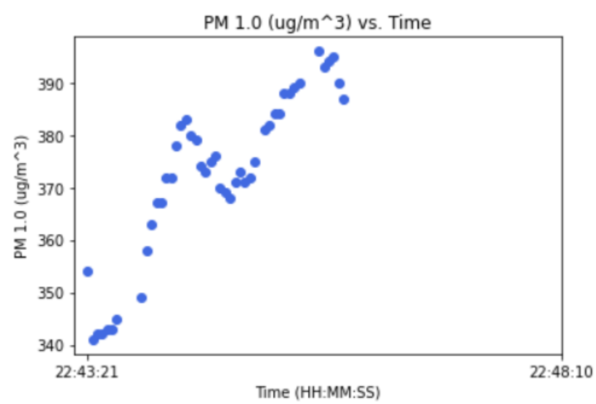

Particles > 5.0 um vs. Time
Particles > 5.0 um
Time (HH:MM:SS)
22:49:39
22:55:16

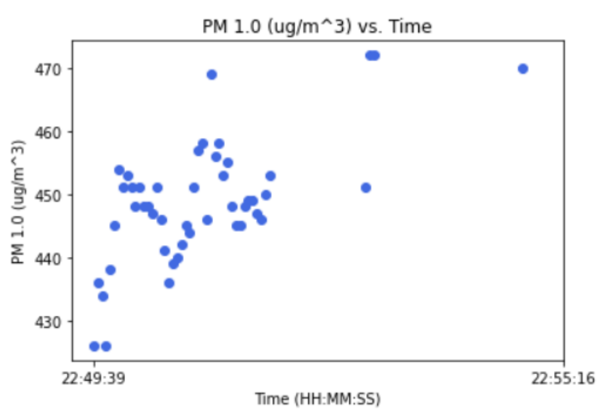

Particles > 10.0 um vs. Time
Particles > 10.0 um
50
45
40
35
30
25
20
15
22:32:07
22:37:43
Time (HH:MM:SS)

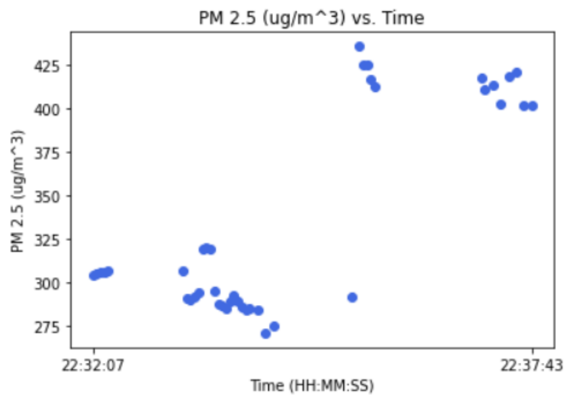

Particles > 10.0 um vs. Time
Particles > 10.0 um
100
80
60
40
20
22:43:21
22:48:10
Time (HH:MM:SS)

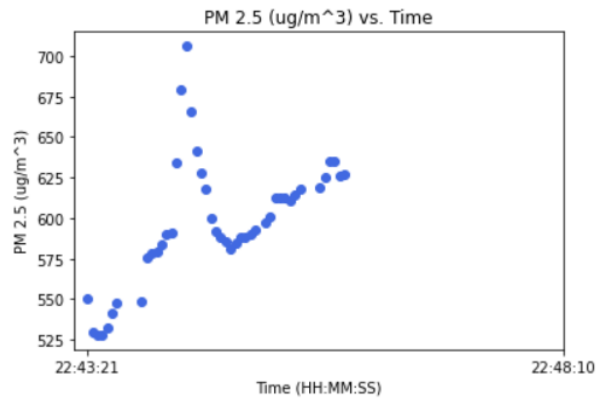

Particles > 10.0 um vs. Time
Particles > 10.0 um
160
140
120
100
80
60
22:49:39
22:55:16
Time (HH:MM:SS)

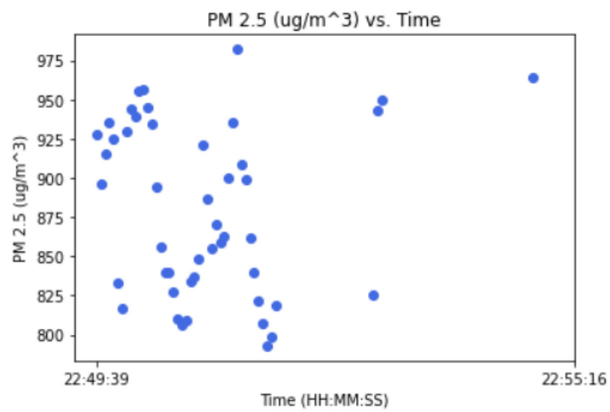

PM 1.0 (ug/m^3) vs. Time
PM 1.0 (ug/m^3)
260
240
220
200
22:32:07
22:37:43
Time (HH:MM:SS)

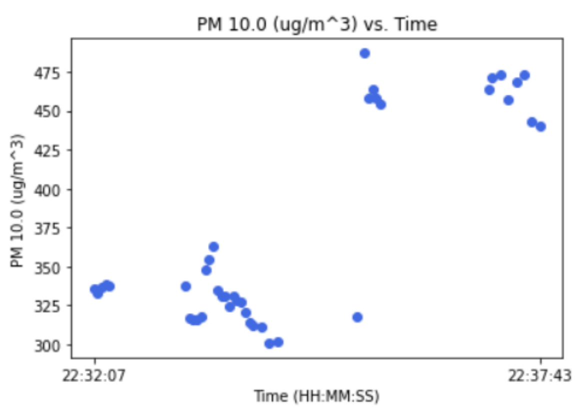

PM 1.0 (ug/m^3) vs. Time
PM 1.0 (ug/m^3)
390
380
370
360
350
340
22:43:21
22:48:10
Time (HH:MM:SS)

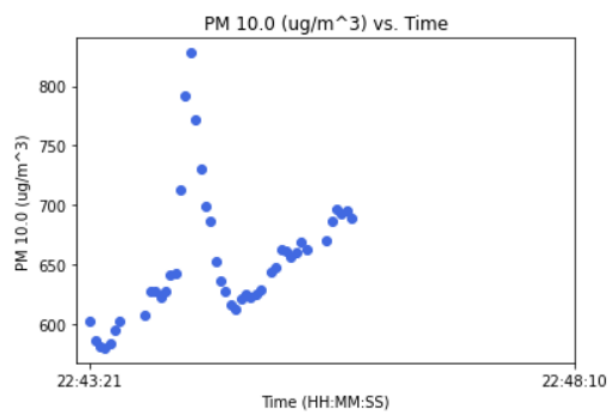

PM 1.0 (ug/m^3) vs. Time
PM 1.0 (ug/m^3)
470
460
450
440
430
22:49:39
22:55:16
Time (HH:MM:SS)

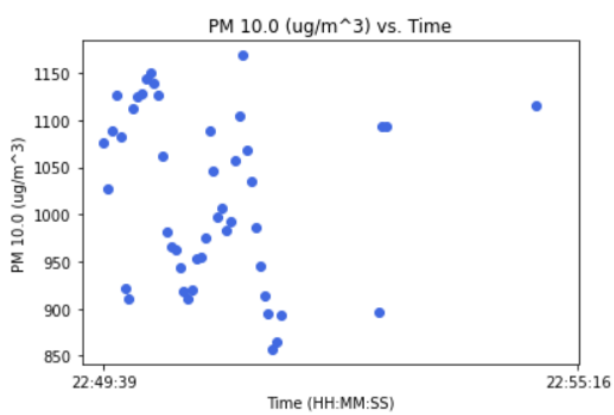

PM 2.5 (ug/m^3) vs. Time
PM 2.5 (ug/m^3)
425
400
375
350
325
300
275
22:32:07
22:37:43
Time (HH:MM:SS)

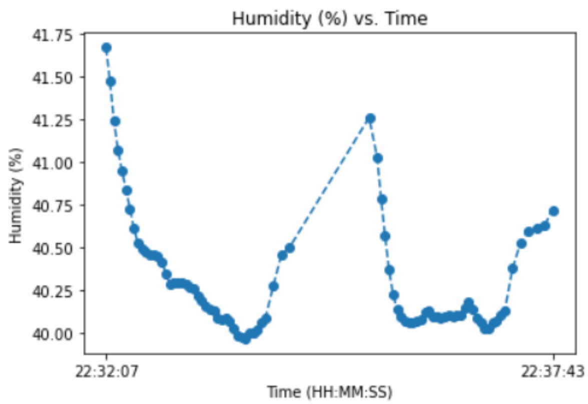

PM 2.5 (ug/m^3) vs. Time
PM 2.5 (ug/m^3)
700
675
650
625
600
575
550
525
22:43:21
22:48:10
Time (HH:MM:SS)

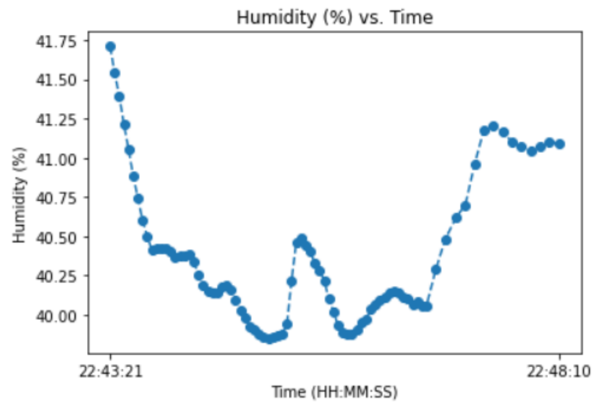

PM 2.5 (ug/m^3) vs. Time
PM 2.5 (ug/m^3)
975
950
925
900
875
850
825
800
22:49:39
22:55:16
Time (HH:MM:SS)

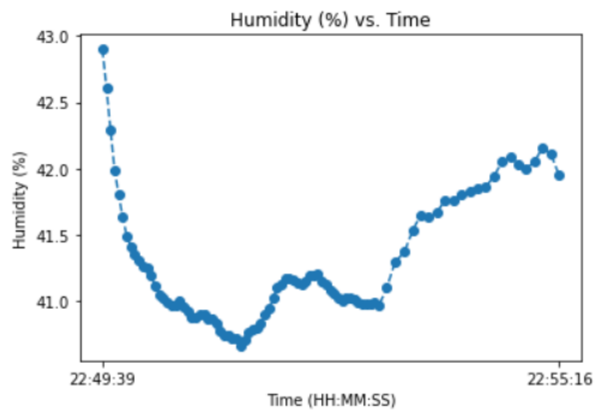

PM 10.0 (ug/m^3) vs. Time
PM 10.0 (ug/m^3)
475
450
425
400
375
350
325
300
22:32:07
22:37:43
Time (HH:MM:SS)

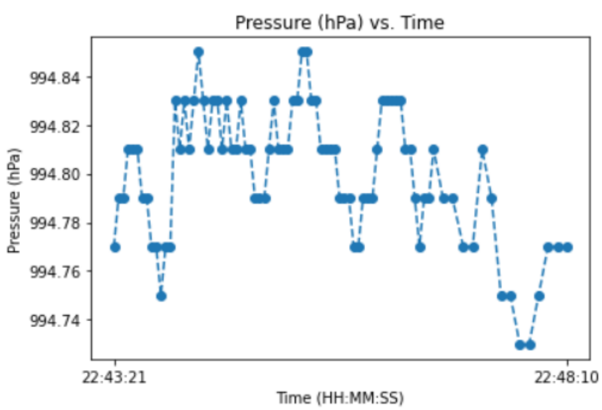

PM 10.0 (ug/m^3) vs. Time
PM 10.0 (ug/m^3)
800
750
700
650
600
22:43:21
22:48:10
Time (HH:MM:SS)

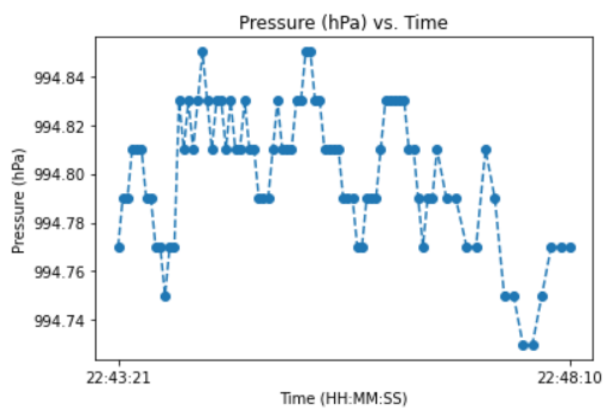

PM 10.0 (ug/m^3) vs. Time
PM 10.0 (ug/m^3)
1150
1100
1050
1000
950
900
850
22:49:39
22:55:16
Time (HH:MM:SS)

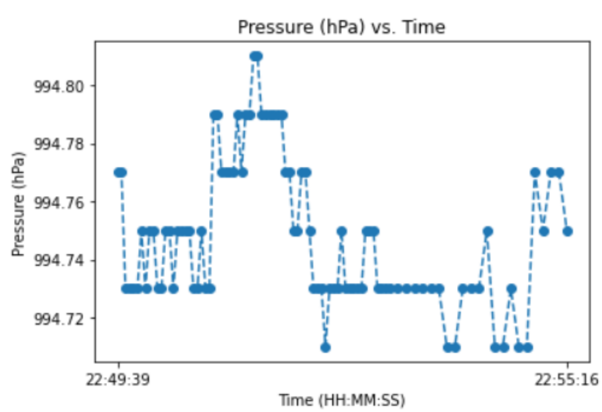

Humidity (%) vs. Time
Humidity (%)
Time (HH:MM:SS)
22:32:07
22:37:43

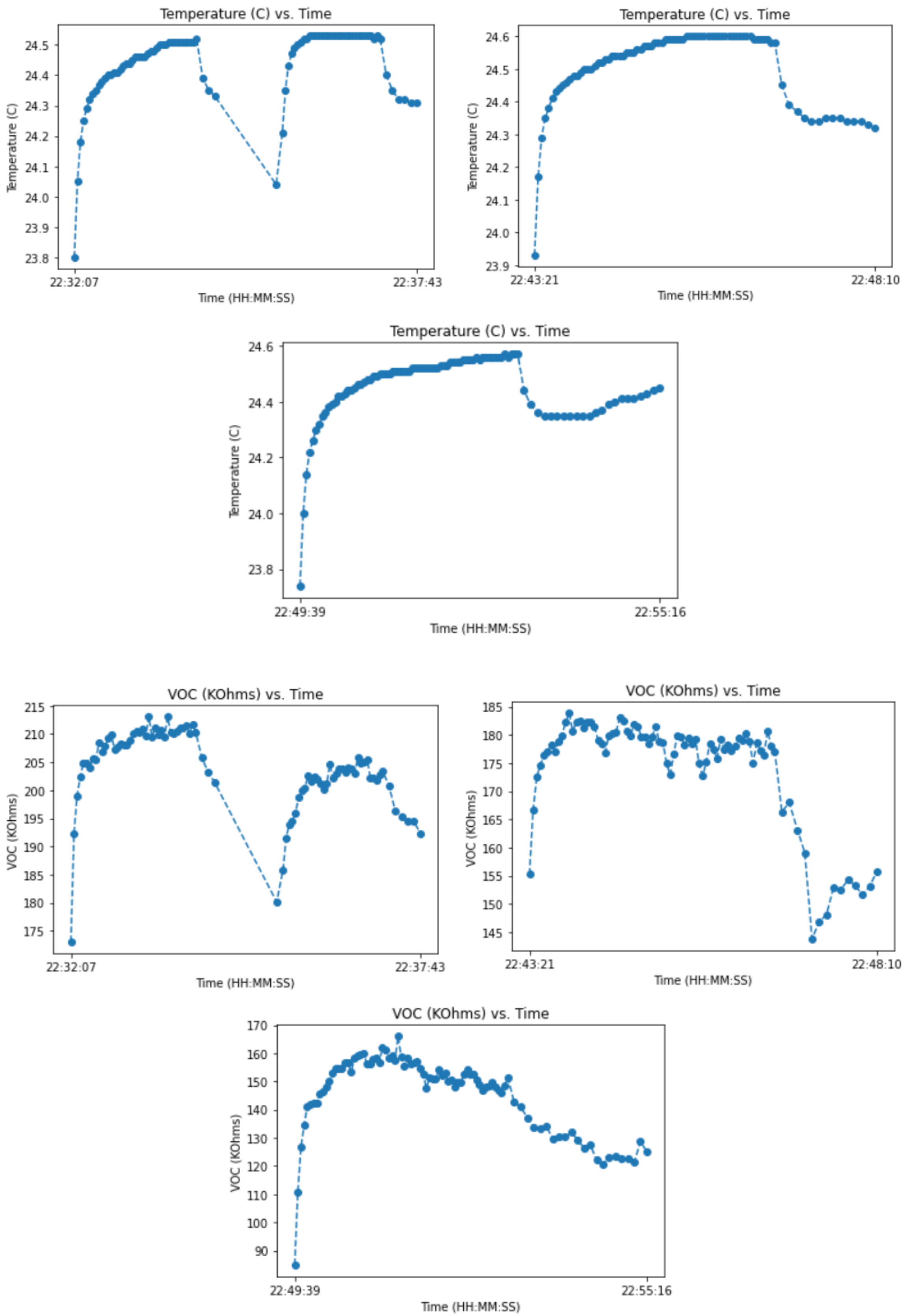

Humidity (%) vs. Time
Humidity (%)
Time (HH:MM:SS)
22:43:21
22:48:10


Humidity (%) vs. Time
Humidity (%)
Time (HH:MM:SS)
22:49:39
22:55:16


Pressure (hPa) vs. Time
Pressure (hPa)
Time (HH:MM:SS)
22:43:21
22:48:10


Pressure (hPa) vs. Time
Pressure (hPa)
Time (HH:MM:SS)
22:43:21
22:48:10


Pressure (hPa) vs. Time
Pressure (hPa)
Time (HH:MM:SS)
22:49:39
22:55:16



Figures 32-73: Incense Sensor Data

The concentration of carbon monoxide increased from roughly 0.036 PPM to 0.09 PPM. The concentrations of particulate matter increased for all size categories that the PMS5003 could measure. By the time the 5th incense stick was lit the sensor's count of particulate matter with a diameter greater than 0.3 micrometers maxed out at 2^{16}-1=65,535. There was more particulate matter in the air than that, but our sensor did not have the capability of measuring a concentration greater than 65,535 particles per 0.1 liters. There were a couple of outlying data points from the particulate matter sensor. This could have been due to the incense sticks which were not held in the censer but were slowly waved throughout the room to spread the smoke as much as possible. Humidity, pressure, and temperature all stayed relatively constant around 41%, 994 hPa, and 24 °C respectively. Something that we do not have a good explanation for is that VOC decreased from 1 incense stick to 3 incense sticks and from 3 incense sticks to 5 incense sticks.

4.1.3 MLX90614 IR Thermometer

If the BME680's temperature sensor should fail there is a backup MLX90614 IR thermometer on the backpack. The main differences between these two sensors is that the BME680 measures environmental temperature regardless of which orientation it is held at, while the MLX90614 measures the temperature of the objects that it is pointed at. To demonstrate the functionality of the MLX90614 in our project the device measured the ambient temperature of the room it was in. A hand was then placed over the sensor and then removed.

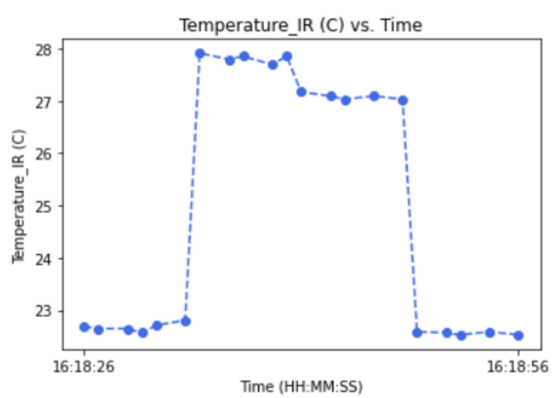


Figure 74: MLX90614 IR Thermometer

Data acquisition from the MLX90614 in Figure 74 above shows that the ambient temperature of the room was about 23.5 °C and the temperature of the hand was about 27.5 °C. The response time of the sensor was extremely quick.

4.2 Sensor Data vs GPS Position

The next 2 sections plot various sensor data versus position on 3D graphs. The latitude and longitude from the GPS sensor was converted to meters and then used for the x and y coordinates. For the z coordinate barometric pressure from the BME680 was used to calculate the altitude above sea level in meters. A color map was used to show how the various sensor data varied with position. The starting position for all of the 3D scatterplots is at the position (x = 0, y = 0). Because the 3D scatterplots are represented as a 2D Portable Network Graphics (PNG) image, the depth of the plots is a little difficult to interpret correctly. In order to view the 3D scatterplots from a different perspective the data files could be run through our data analysis program a second time while representing the 3D scatterplots with a different azimuthal angle and distance.

4.2.1 Walking

The next section of plots was taken while walking around Loomis. Sensor data begins on the 4th floor near the southwest stairwell. The firefighter's backpack was taken down to the first floor and out the front door, then around the patio and up towards the northwest entrance on Goodwin Avenue. Then the device was taken back around the front of Loomis and into the parking lot on the east side of the building. This walking route is illustrated in Figure 75.

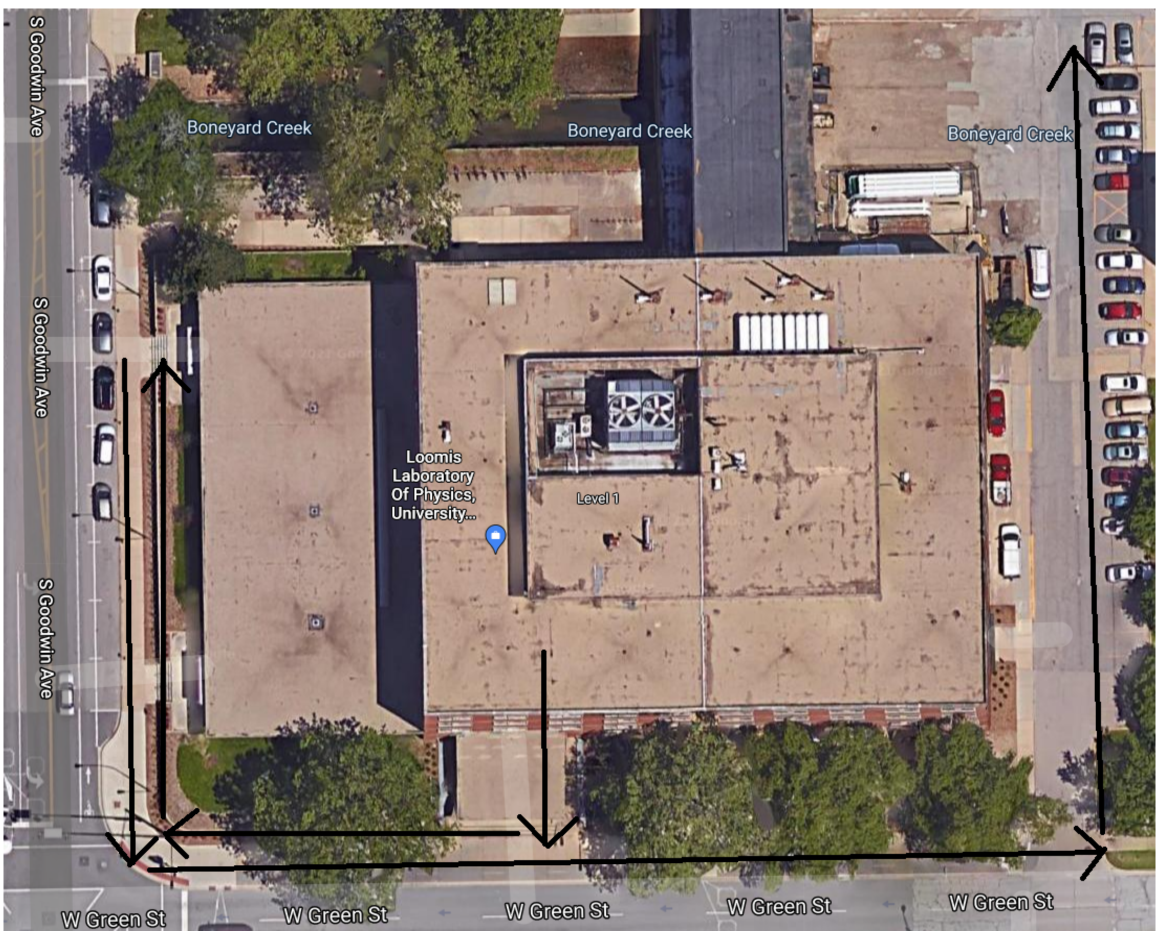
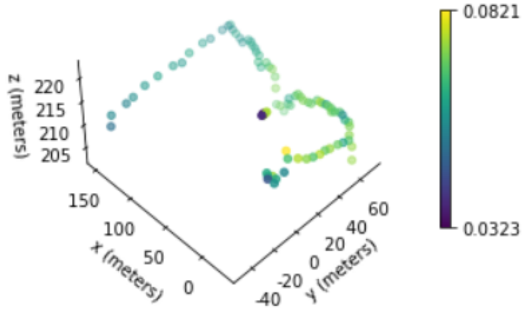
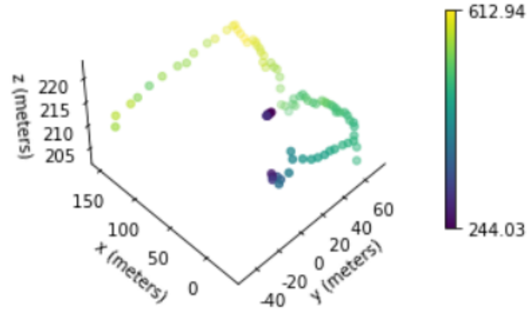


Figure 75: GPS Walking Route

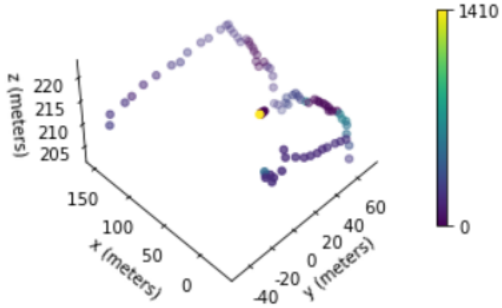
Carbon Monoxide (PPM) vs. Position



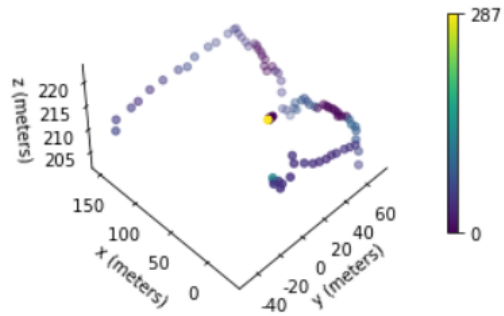
VOC (KOhms) vs. Position



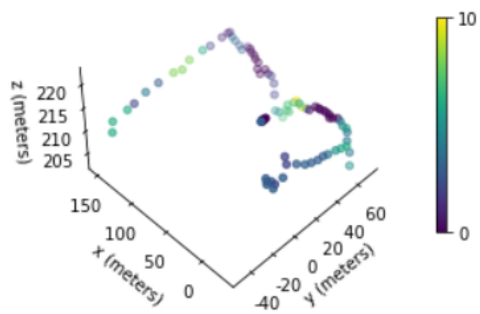
Particles > 0.3 um vs. Position



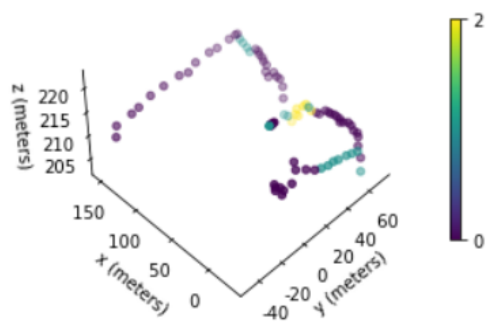
Particles > 0.5 um vs. Position



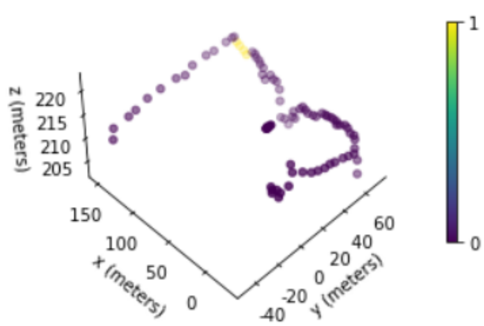
Particles > 1.0 um vs. Position



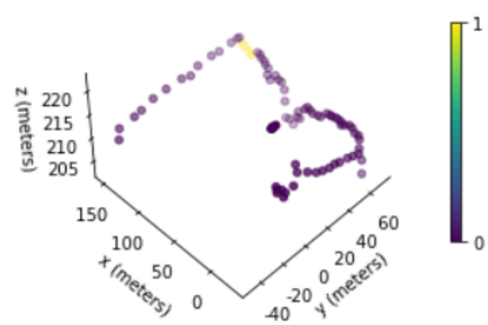
Particles > 2.5 um vs. Position



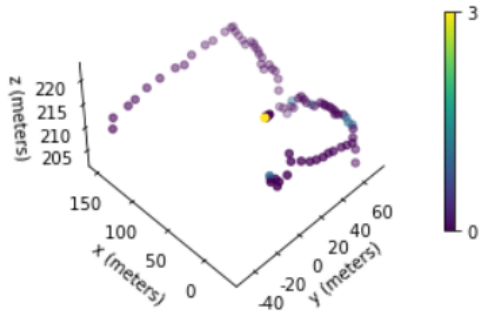
Particles > 5.0 um vs. Position



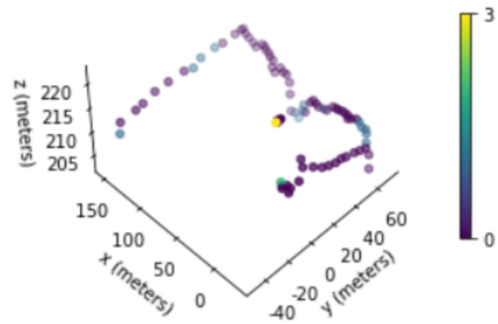
Particles > 10.0 um vs. Position



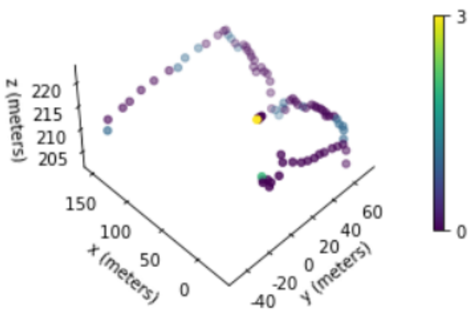
PM 1.0 (ug/m^3) vs. Position



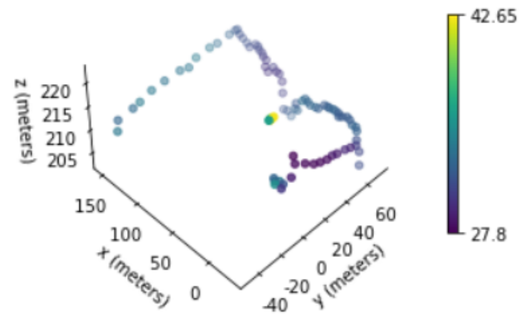
PM 2.5 (ug/m^3) vs. Position



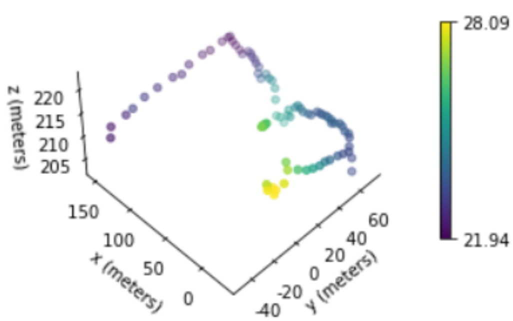
PM 10.0 (ug/m^3) vs. Position



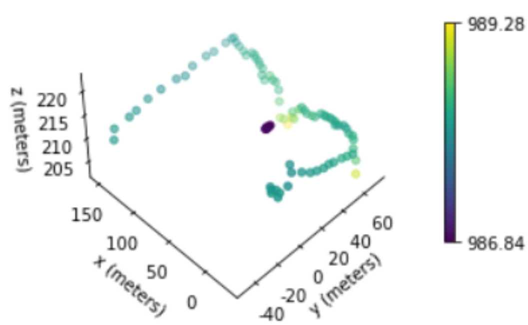
Humidity (%) vs. Position



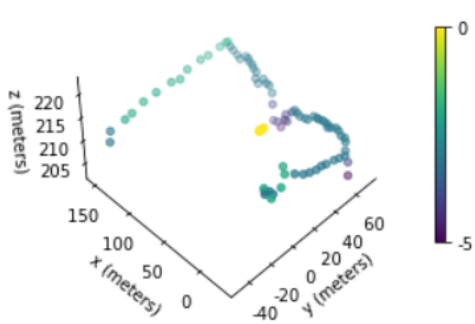
Temperature (C) vs. Position



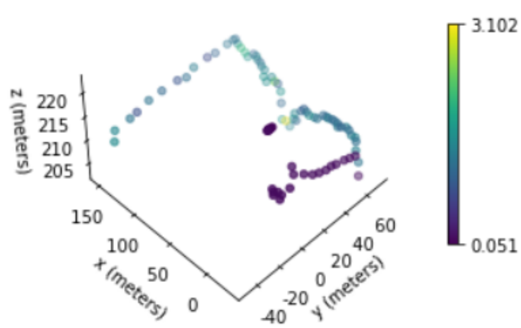
Pressure (hPa) vs. Position

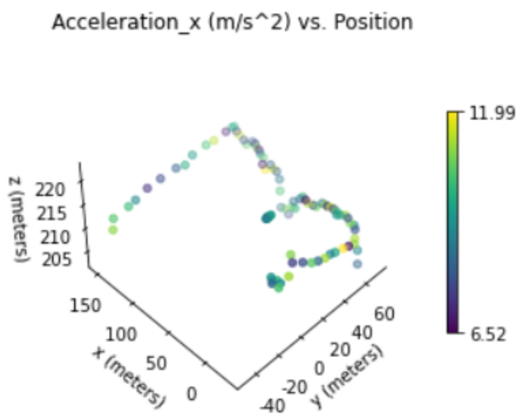
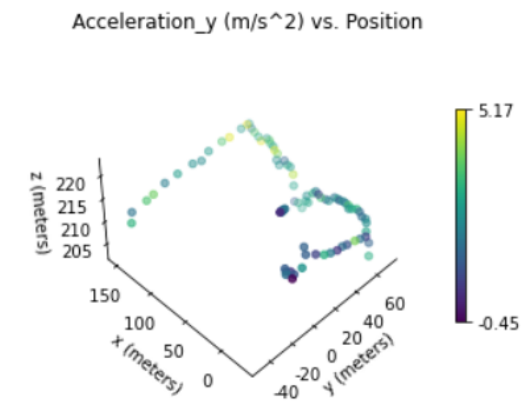
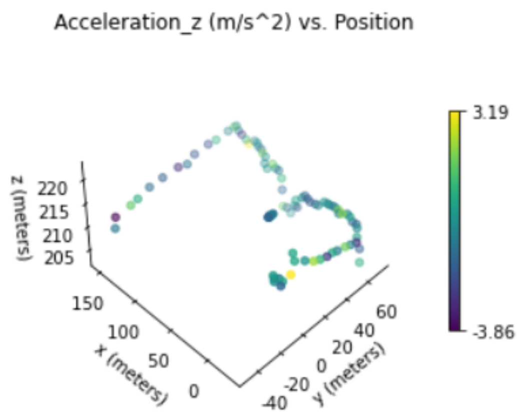
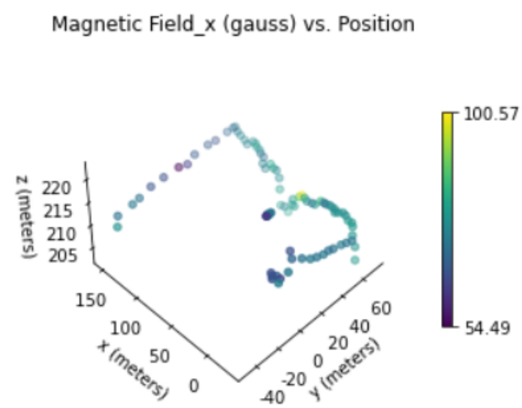
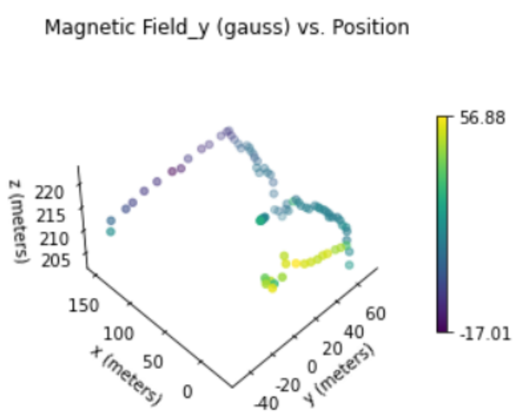
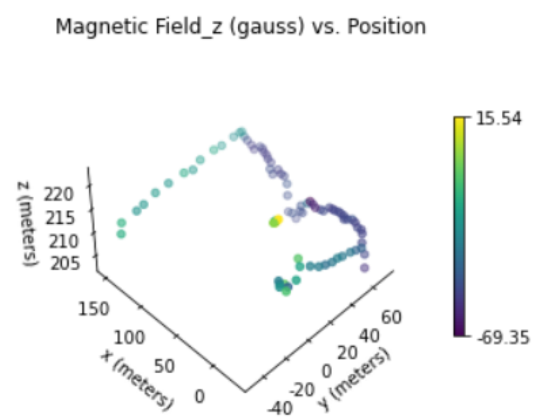
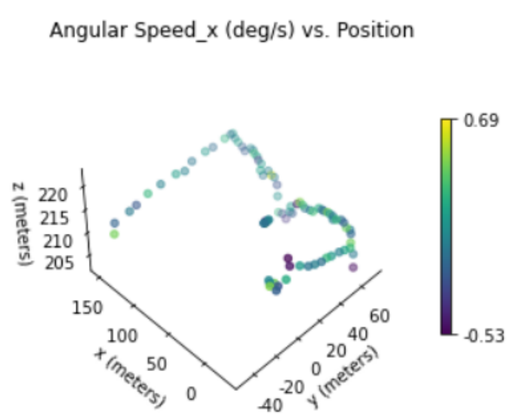
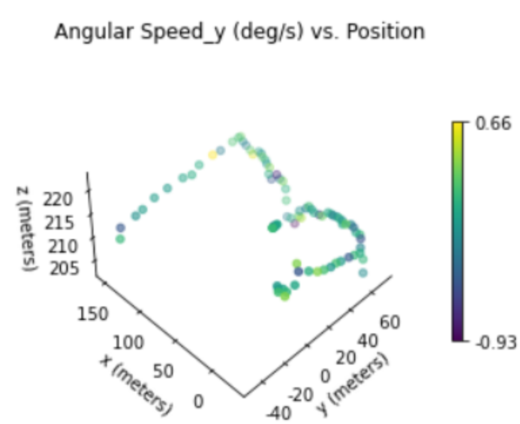


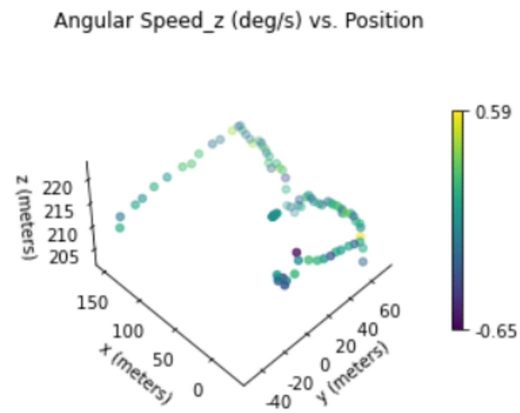
Floor vs. Position



Speed (m/s) vs. Position




Acceleration_x (m/s^2) vs. Position
11.99
6.52
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Acceleration_y (m/s^2) vs. Position
5.17
-0.45
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Acceleration_z (m/s^2) vs. Position
3.19
-3.86
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Magnetic Field_x (gauss) vs. Position
100.57
54.49
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Magnetic Field_y (gauss) vs. Position
56.88
-17.01
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Magnetic Field_z (gauss) vs. Position
15.54
-69.35
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Angular Speed_x (deg/s) vs. Position
0.69
-0.53
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)

Angular Speed_y (deg/s) vs. Position
0.66
-0.93
z (meters)
220 215 210 205
150 100 50 0
x (meters)
-40 -20 0 20 40 60
y (meters)



Figures 76-100: Walking GPS and Sensor Data

Figures 76-100 show how our various sensors responded to changing atmospheric conditions. There was a higher concentration of carbon monoxide and VOC outside the building than inside. Atmospheric pressure was also higher outside than inside. Humidity was lower outside than inside. The floor of a building was approximated by using the barometric pressure at the beginning of data acquisition to determine ground level and subtracting that from the barometric pressure during data acquisition divided by 3.5 meters per floor and finally truncating that value to the nearest integer. This is represented by Equation (2) below.

$$\lfloor(\text{current altitude - starting altitude})/3.5 \text{ meters}\rfloor = \text{current floor} \quad (2)$$

There was an error of ± 1 floor using this method due to arbitrary starting and ending heights.

4.2.2 Driving

The next set of data was taken while driving. The car was driving from Oberweis, located on route 45, to Lincoln Avenue. The car was driving roughly north (on route 45, so it was a little north east), then turned east onto Kirby Avenue, and then north again on Lincoln Avenue. The route is shown on figure 100 below. The speed limit on those roads is about 40 miles per hour (approximately 17 m/s), so that is about how fast the car was going.

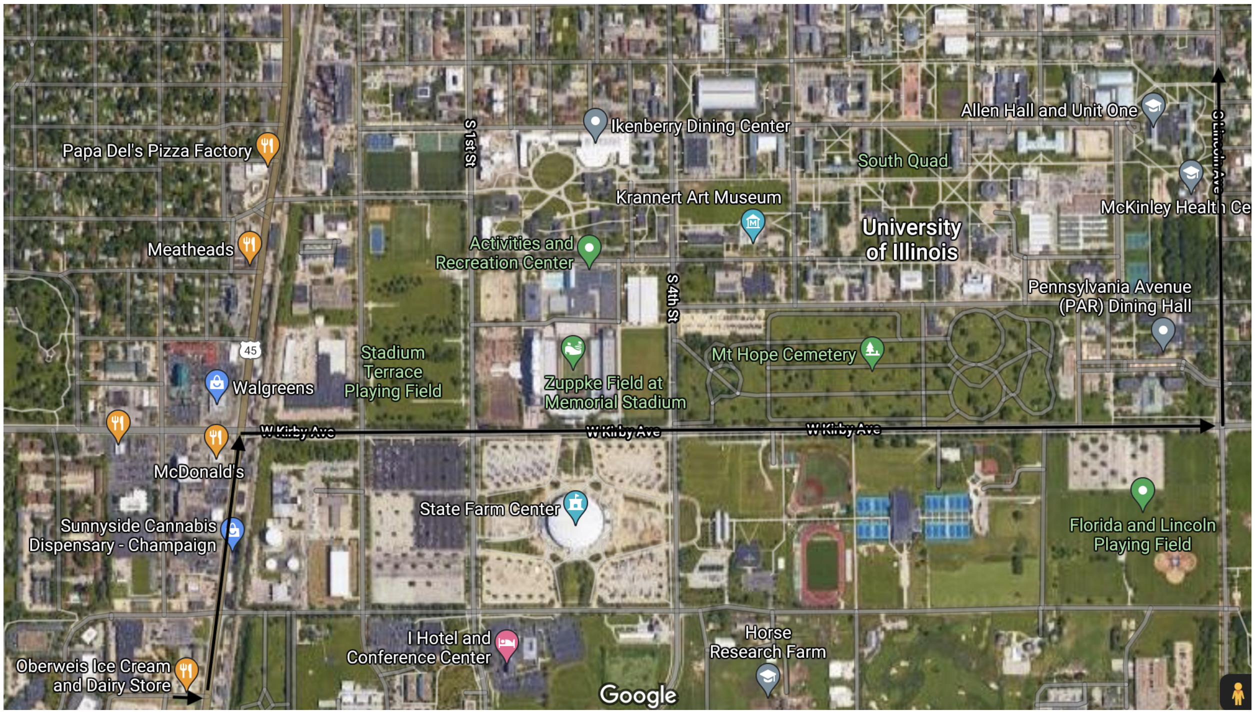
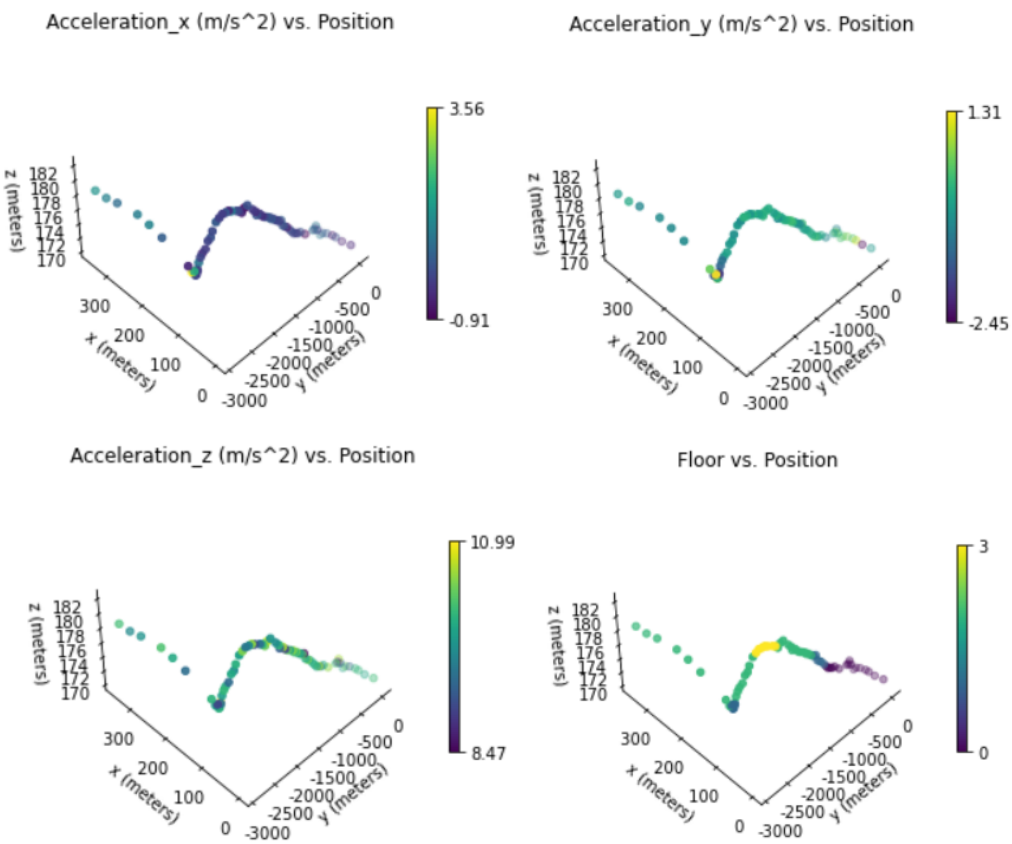
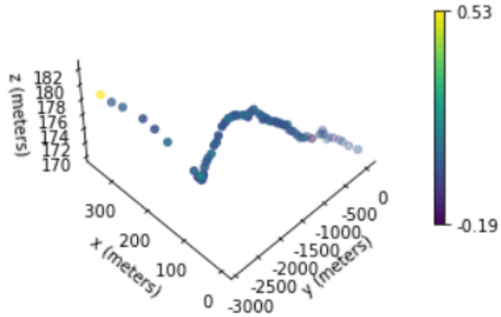


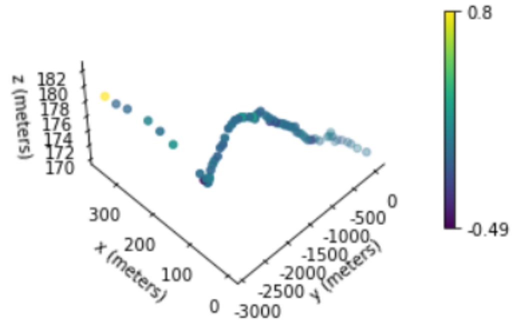
Figure 101: Driving GPS Route



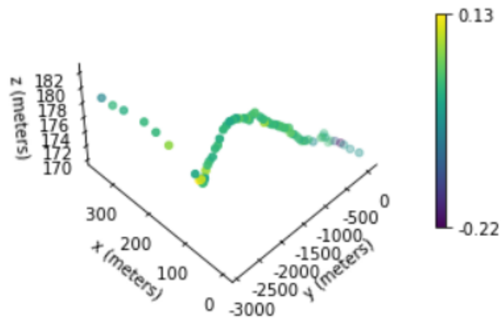
Angular Speed_x (deg/s) vs. Position



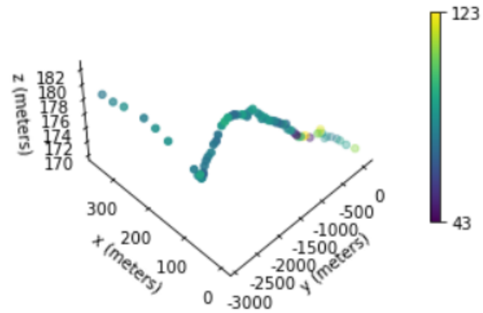
Angular Speed_y (deg/s) vs. Position



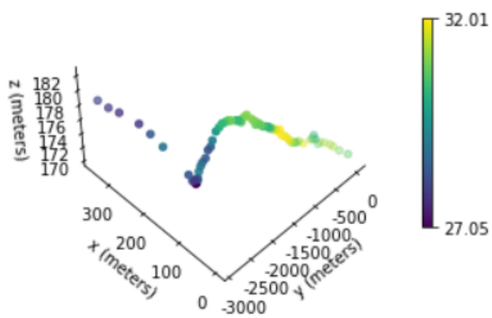
Angular Speed_z (deg/s) vs. Position



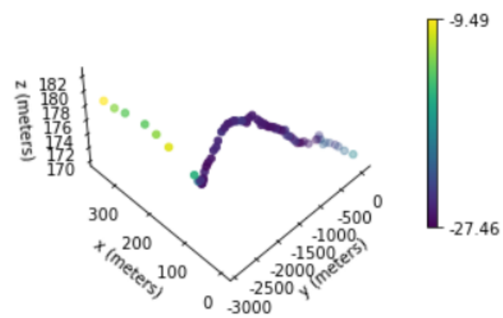
Heart Rate (BPM) vs. Position



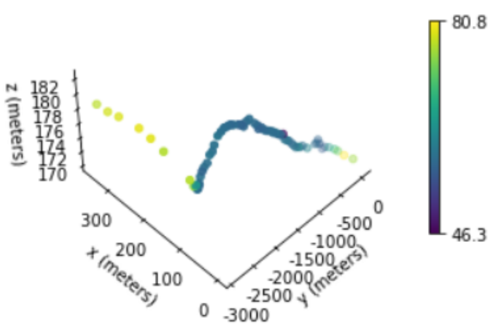
Humidity (%) vs. Position



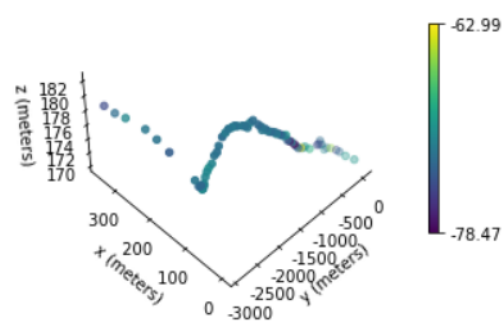
Magnetic Field_x (gauss) vs. Position

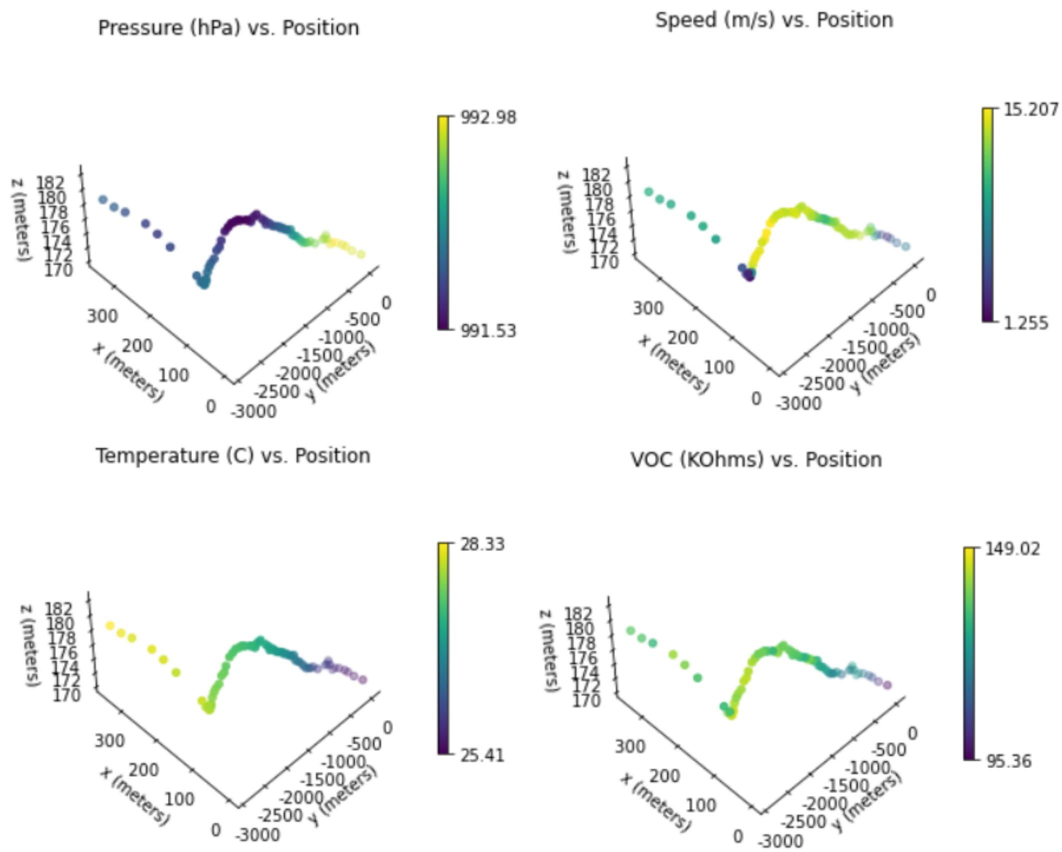


Magnetic Field_y (gauss) vs. Position



Magnetic Field_z (gauss) vs. Position





Figures 102-117: Driving GPS and Sensor Data

The GPS data reflects the direction of the drive. The path is traced on each figure, and the first part of the trip is clearly north east as opposed to simply north. The acceleration in all directions (x, y, and z) did not seem to change that much, so the car was moving at a fairly constant rate most of the time. However, when the car slowed down on Kirby Avenue before turning on Lincoln, the y acceleration (due to the orientation of the PCB in the car) decreased, and this instance can be seen on the graph as a yellow dot. This correlates with the speed data, which decreased greatly at the same time and can be observed by the purple dots on the graph. As the car warmed up, temperature increased by almost 3 degrees celsius.

5.1 Future Developments and Potential Improvements

The first and most obvious area needing further work and development is capturing location data. The backpack is intended to be used as a tool to gather data. This data is then sent to a central data processing hub (the base station), which ultimately allows for an observer to oversee operations, to make sure all of the incoming data is in order, and to take any necessary actions if a given parameter is not in a certain range. New data could potentially lead to revising the strategy of a firefighting operation (for example a redistribution of resources to secure a fire site based on toxic gas concentrations rather than temperature of fire). In order to obtain useful and sensitive data quickly we would need various high spatial-resolution maps. This would require more accurate and sophisticated positional data to be accumulated.

The GPS we used was the GlobalTop Technology FGPMMOPA6H GPS module. Given that this module does not work well inside a building (we tried obtaining a signal on multiple floors of different buildings at different times, to no avail), we would need to come up with a better way to track a given firefighter's position inside a building. While this module was simple enough to breadboard and connect to our backpack, it is not practical for use in real life fire fighting situations. To counter this issue, we propose the following solutions:

I. Use a better GPS module: This solution is simple and straightforward. Google maps has implemented this to some degree of success. One can quite easily pull out their mobile phone and use the 'indoor' version of maps. This indicates that the GPS module in our mobile phone is communicating with the wireless network inside the building and using it to transfer data to a satellite, mapping out one's real time coordinates to a relatively high degree of precision. This method of communicating with Wi-Fi for positioning is accurate to just a few feet, much improved over our current module. This would require us to get a GPS module with Wi- Fi compatibility. However, even this solution has its drawbacks, such as its dependency on Wi-Fi itself. It is unlikely that firefighters would receive a Wi-Fi signal inside a burning building, motivating the need for more alternatives.

II. Using an inertial sensor to measure acceleration: We can use an inertial device, such as an accelerometer to plot an acceleration vs. time graph. Using this and making a note of our initial position and velocity, we can integrate and plot position vs. time data as well. Most accelerometers today work by relying on the piezoelectric effect, which occurs when a voltage is generated across certain types of crystals as they are stressed. The acceleration of the test structure is transmitted to a seismic mass inside the accelerometer that generates a proportional force on the piezoelectric crystal. This external stress on the crystal then generates a high impedance, electric charge proportional to the applied force and thus proportional to the acceleration. A sample image is shown below. As the above process details, an accelerometer is a very sensitive piece of equipment, and any level of uncertainty will compound over time (especially as acceleration is being integrated twice to obtain position, with any errors eventually snowballing into huge disparities).

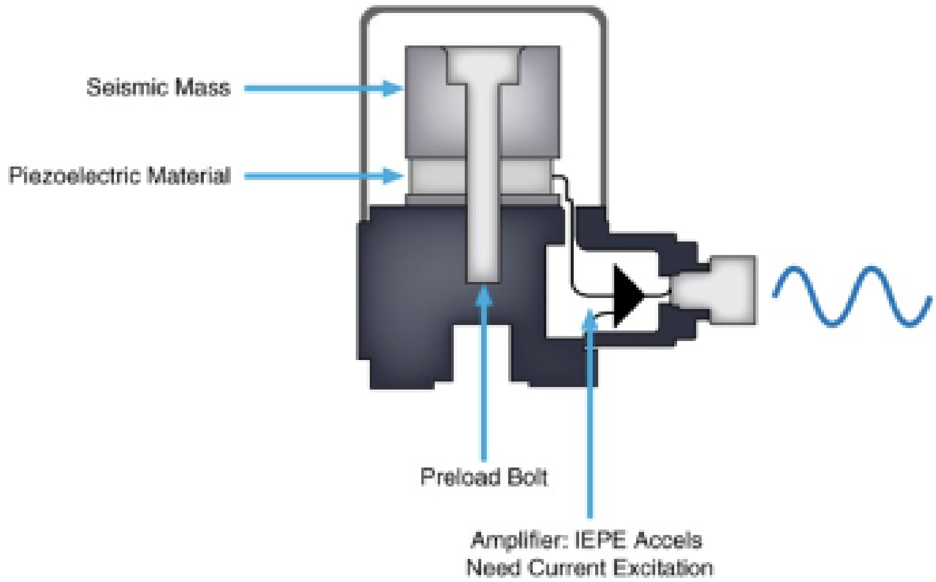


Figure 118: Example Accelerometer

III. The third and potentially best alternative that we came up with was the idea of triangulation/trilateration for indoor positioning. Both involve setting up three to four external transmitters outside the perimeter of the building, sending signals to each of the backpacks. Triangulation is a method for calculating position that relies on a known distance between two measuring apparatuses and the measured angles from those points to an object. This works by using the angle-side angle congruence theorem to find the location of an object. Trilateration is the more common of the two methods used for positional calculations. Trilateration uses the known distance from at least three fixed points in 2D space and four fixed points in 3D space (which is apt for our purposes) to calculate the position of an object. This works by finding the intersection of a series of circles (imagine a Venn diagram). While trilateration relies on signal strength for distance, triangulation relies on timing differences in the reception for tag's signals. As signals travel at the speed of light, the time differences in transmission are incredibly tiny, making such instruments quite expensive. Using signal strength could also be potentially problematic, as it is difficult to control for all the different surfaces it would have to travel through (such as those in a building; one position could require it to travel to more dense surfaces than others, leading to a weaker signal strength even from the same distance).

We also noticed some potential issues in our two proposed models of communication between the backpacks and the base station.

1) Model 1: Backpack ID of *n* will only send a signal when it hears from backpack *n-1,* and should something go wrong with backpack ID *n-1*, backpack *n* will never receive and thus never send a message, and the entire chain breaks down if even one link becomes compromised.

2) Model 2: The number of seconds between sending signals is equal to the number of backpacks there are. So in theory, as the number of firefighters becomes larger, say around 50, the base station would have to wait 50 seconds before it heard back from any given backpack. This could be problematic, as a firefighter could go quite a long time unaided if they need any assistance from the base. This essentially does not scale very well as there are more firefighters.

5.2 Inertial Positioning using LSM9DS1

As mentioned in the previous section, a potential solution to measure positioning of the firefighter would be to integrate acceleration measured by the LSM9DS1. We tried to do this, and it was harder than it seems to be at first glance. To do this, we had to integrate acceleration twice in each direction (x, y, and z) separately. The calculation follows the basic idea of integration. In our data acquisition program, we are using a timer function to record all of our measurements at the same time. We were able to use this timer function as the change in time in calculating position, because we knew the amount of time in between each datapoint. We multiplied the change in time by acceleration and added that to the previous velocity to find the velocity of the LSM9DS1. This same method was used again to calculate position. To use this method of integration, we needed to know the beginning values of velocity and displacement, so we assumed them to be 0 in each direction. Every time the program was run, the LSM9DS1 did start off at rest.

This method is not the most accurate. Since we are integrating discrete values rather than integrating continuously, some error will always be present in every calculation, and the errors will accumulate over time. The accelerometer could also have an error, which we think it does because it still outputs acceleration values of around 0.02 meters per second per second (m/s/s) when not moving. According to CH Robotics, an accelerometer with error of around .017 m/s^2 can be off in positioning by 6.12 kilometers after only 10 minutes. To correct for this error, various filters can be applied to the data. A Kalman filter is something commonly used to correct for errors when integrating acceleration twice. This filter, which works with a series of measurements taken over time, reduces noise in data and gives better estimates for the measurements. We were unable to include this in our project due to time constraints.

From the data taken with the accelerometer it appears as though using an accelerometer for positioning works better while in motion rather than staying stationary. Data was taken both while the accelerometer was stationary, and while the device was carried around the block. During our tests there was a lot more variability in the z values than in the x and y values. Because of the accumulation of errors, during an 8.5 minutes data acquisition the double integration of acceleration said that the LSM9DS1 had moved 7 meters in the x direction, 1.4 meters in the y direction, and 24.9 meters in the z direction. This was a total Euclidean distance of 25.9 meters, but in fact the LSM9DS1 had not moved at all. It is harder to check how accurate the walk around the block was, but it seems that acceleration is more accurate when the LSM9DS1 is in motion.

6. Conclusion

The goal behind the firefighter backpacks was to demonstrate how radio connected sensor backpacks could improve firefighting techniques. As mentioned in Section 5, there are some issues with our backpacks, but they provide real time environmental conditions to a base station. The base station observer can oversee the firefighters, and should something be wrong, they can be notified immediately and respond accordingly.

Section 4.1 shows plots generated through a data analysis program, and each of the different scenarios we ran to simulate extreme conditions provided expected results, with particle concentration, size, and amount increasing. Temperature increased as well, and the amount of carbon monoxide detected drastically rose, even though our situations were not nearly as hazardous as firefighting conditions.

Section 4.2 shows additional plots generated, this time in 3D. These plots show that while the GPS has a signal and can see satellites, it is able to provide location information very accurately. This, coupled with other sensors like the accelerometer, can map out how the backpack is moving. However, as discussed in Section 5, we found that the GPS signal was unreliable which is a downfall to the backpacks. This is a problem with many potential solutions, but due to time constraints we were not able to implement any in our project.

Firefighters work in extremely hazardous conditions. The topics discussed in Section 5 could be improved upon, however the backpacks are able to both collect and send data about firefighters and their conditions to a stationary observer at the base station. This data is then able to be analyzed, allowing observers to map the path the firefighters took, and see how the environment changed along the way. This could allow for faster response time should additional help be needed. The base station is also able to alert the observer if the firefighter has too low of a BPM, too high of a BPM, too high of a temperature, or if a backpack is lost. Our sensor backpacks, able to send data and alert the base station, would be able to aid in firefighting because they provide real time data on firefighting conditions to an observer who can monitor the situation from the base station.

7. References

[1] https://github.com/adafruit/RTClib/blob/master/examples/softrtc/softrtc.ino

[2] https://github.com/adafruit/Adafruit_BME680/blob/master/examples/bme680test/bme680test.ino

[3] https://github.com/arduino-libraries/SD/blob/master/examples/ReadWrite/ReadWrite.ino

[4] https://github.com/arduino-libraries/LiquidCrystal/blob/master/examples/HelloWorld/HelloWorld.ino

[5] https://courses.physics.illinois.edu/phys398dlp/sp2021/code/set_RTC_with_GPS.zip

[6] https://pulsesensor.com/pages/getting-advanced

[7] https://courses.physics.illinois.edu/phys398dlp/sp2021/code/PMS5003_reader.zip

[8] https://github.com/adafruit/Adafruit_LSM9DS1

[9] https://github.com/adafruit/Adafruit-MLX90614-Library

[10] https://forum.arduino.cc/t/integration-of-acceleration/158296/14

[11] http://www.chrobotics.com/library/accel-position-velocity

[12] https://www.adafruit.com/product/181

[13] https://www.adafruit.com/product/3199

[14] https://lastminuteengineers.com/pulse-sensor-arduino-tutorial/