Team 33
Project Proposal

Air Pollution Mapping Bands

Chirag Nanda
Vedant Agrawal
Vatsin Shah

Table of Contents

1. Introduction

1.1 Problem

As air pollution has increased globally, the need for pollution tracking has increased in tandem. Today, most cities take readings using satellites as well as sensors scattered around the city to collect an aggregate reading of city-wide air quality based[1]. While this may give a good estimate of the air pollution over a city-wide area, the air quality of individual localities and streets may differ vastly.

Air pollution can change over the course of a day. A variety of factors including traffic, population density, the operation of office buildings, and factories can influence the air quality. A more dynamic calculation of air quality can help people decide which routes to take and which places to avoid. Some cities like Barcelona and Chicago have tried implementing IOT based air pollution trackers embedded into city-wide infrastructure to aid in this effort. Google has even tried to fit street view cars with sensors to track pollution levels[2]. Nonetheless, these devices are often extremely expensive. For instance, the sensor nodes used in Chicago cost around five thousand dollars per node.[3] Additionally, the sensors are often spread far apart, preventing accurate locality-centric/streetwise data collection of pollution.

1.2 Solution

Our solution to this problem is to create a cheap wearable band and an accompanying mobile app that will continuously monitor the air quality around the user. The broader idea is to have thousands of users wear this band to help contribute to a city-wide map that everyone can access. Nevertheless, within the time constraints of the course we plan to first create a proof of concept of the band and a simple application that gives alerts to the user about their general vicinity. The app can keep a personal record of air pollutant levels of the places they visited on a map.

We aim to keep track of carbon dioxide and carbon monoxide. Additionally, since this band will be portable, it has the potential to be useful as a warning device in indoor spaces. Hence, we also wish to sense propane as it is a common flammable gas. The band can then help find poorly ventilated areas and even warn users of potential gas leaks in places like warehouses and storage rooms. For our project we only plan to build one band. However, we plan to have multiple profiles on our app to test how multiple users can update the same map with the pollution data they collect.

1.3 Visual aid

A mockup of our idea is shown below. All sizes are approximate for now. We picked 3cm for the thickness of the band because the height of the tallest component we plan to use is 2.5cm. We will try to keep the size of our final device within these approximations if possible.

Top View

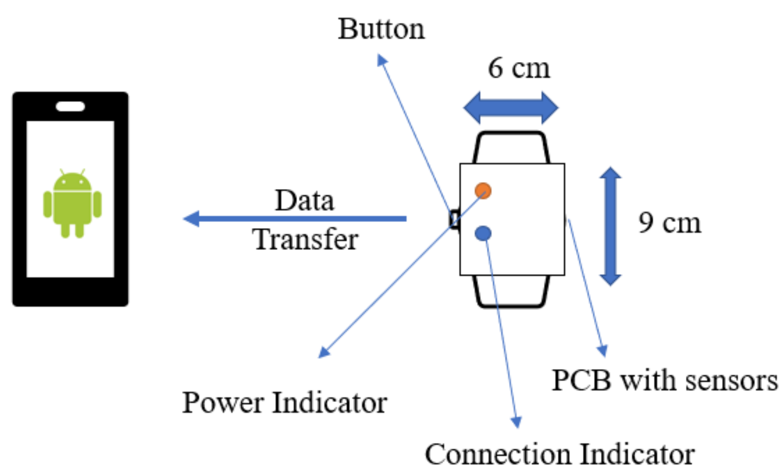

Figure-1: Top view of band

Side View

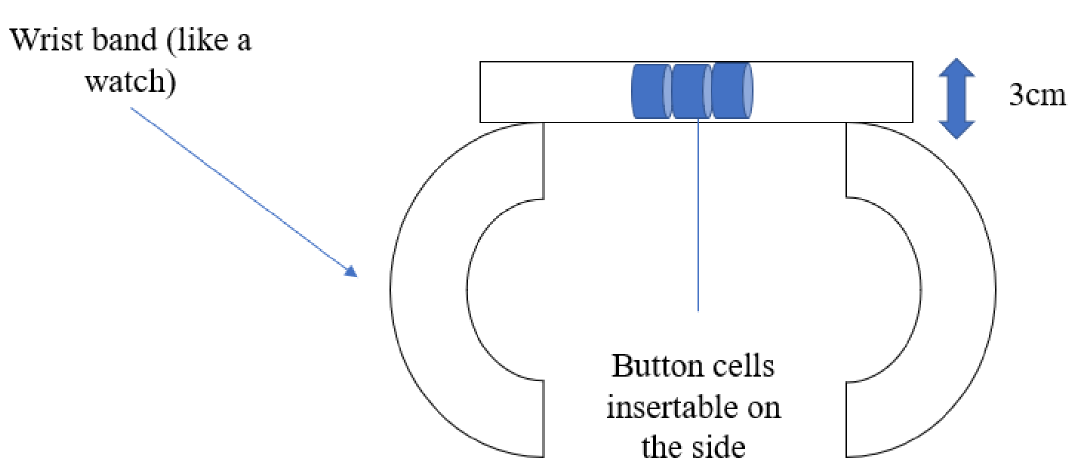

Figure-2: Side view of band

1.4 High-level Requirements

1. The propane sensor should be able to simply detect the presence of propane. The carbon monoxide sensor should be able to detect up to 200 ppm. The carbon dioxide sensor should be able to detect upto 10000 ppm. We have picked these values based on USDA determined values of dangerous exposure. [4] [5]

2. The app will need to be able to take pollution data from the band and update the map accordingly. Since the data will be a continuous stream, as a design choice we will update the app at a specific period of time as pollutant values will not be changing continuously. The app must also be able to warn the user in the form of a notification or sound if the pollution level is not safe or if the propane was detected (since it is flammable).

3. Our band needs to be wearable and must have around 1-3 hours battery life to be able to track pollution data when a person makes their commute. We plan to create the housing for our circuitry using a 3D printer

2. Design

2.1 Block Diagram

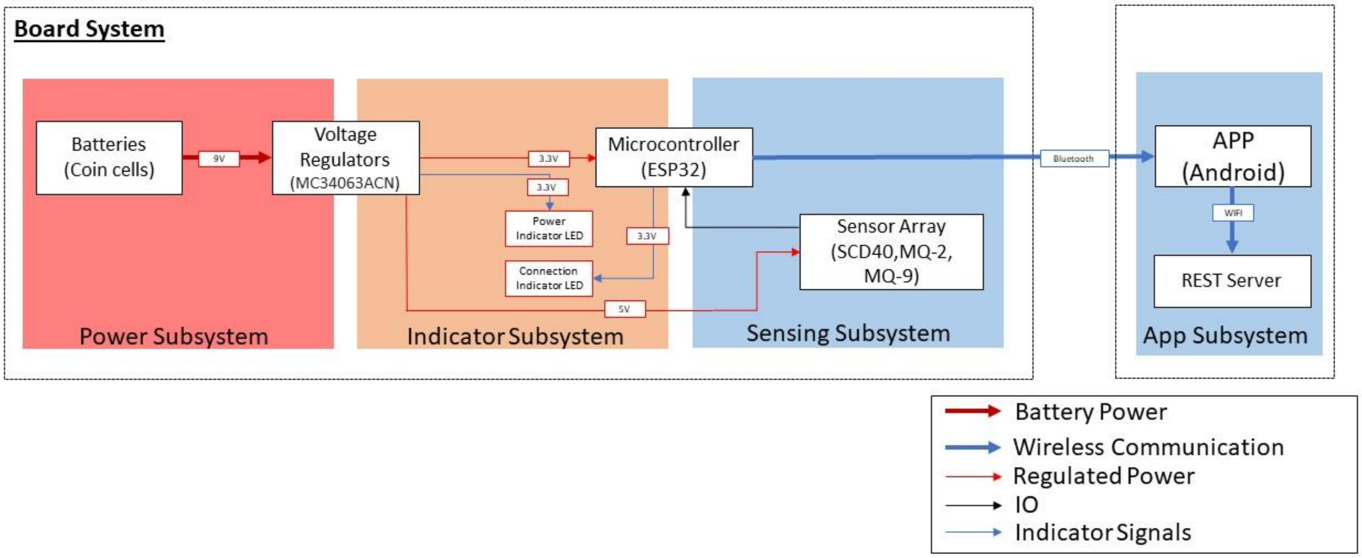

Figure-3: Block diagram of the project

2.2 App Subsystem

2.2.1 Overview

We intend on designing an Android application to produce human-readable values of the output of the sensors and use these values in the decision model to alert the user. The application would also serve the purpose of displaying the heat map, alerting the user if they enter a contaminated region, and an interface to interact with a server to indicate contaminated regions. This subsystem aims to solve the second high-level requirement of interacting with the sensor data and update the shared map with those values to handle user alerts

2.2.2 Requirements

1. Connect to the bluetooth of ESP32 chip and periodically receive bluetooth packets and unpack them without losing any data. The exact time period of this data exchange will be determined later and will possibly be configurable.

2. GPS data can be collected using the device's inbuilt GPS

3. Use the collected values of carbon dioxide and carbon monoxide to check against the threshold to determine if the user should be alerted. Any detection of propane should be

notified. Additionally, periodically send the collected data along with gps data to the central server using a post request via TCP protocol. We are yet to decide whether we will host the rest server on localhost or online. We intend to keep a running average of the gas level values within the update time frame to reduce noise in our data.

4. On the central server, we will maintain a ledger of all the values we get against the gps coordinates. We plan on rounding off the gps coordinates to a certain degree that would allow us to group the values in 100m radius together on the map. We have chosen this value based on a research paper by REVIHAAP that suggests that the gasses can be in dangerous concentration upto 100 m away from a leak[6]. If a value for the given coordinates already exists in the ledger, update the value with an average of the existing and current value. Otherwise, just make a new entry.

5. On the android app, we should be able to perform a get request to the central server and get this ledger of gps coordinates against the gas level values. These values will be visually displayed on a map using the google maps API. Additionally, check if the rounded down version of your own gps coordinates exists in the ledger. If it exists, check if the values exceed the threshold values and alert the user of a nearby leak.

2.3 Board System

Our board system consists of the power subsystem, indicator subsystem, and sensing subsystem. The board system meets our first high level requirement by being responsible for powering our sensors and microcontroller to monitor and send pollutant data to the app. To maintain our third high level requirement, we plan to keep the design minimal and ensure that our pcb is as small as possible so that the band is light and wearable.

2.3.1 Power Subsystem

2.3.1.1 Overview

This subsystem takes in battery voltage (~9V) from 3 3V cells and steps it down to 5V for powering the sensors and 3.3V for powering the ESP32. The voltage regulator also connects to the indicator subsystem to indicate that the device has been powered on.

2.3.1.2 Requirements

1) Batteries: The 9V input must come from 3 3V button cells to keep the design compact. Also, in order to improve battery life (to meet high level requirement 3) when the band is not being used, there needs to be a switch between the battery and voltage regulators.

Battery life will be tested manually and maximized to ensure that the battery doesn't need replacement often.

2) Voltage Regulator: The power subsystem needs to step voltage down while wasting minimum possible energy in order to maintain high battery life. The ideal way to do this would be a buck converter for each voltage while keeping the same 9V input for both. Less energy wastage in step down means higher battery life over time. The microcontroller operates on 3.3V and sensor array operates on 5V, so both of those stepdowns will be required.

2.3.2 Indicator Subsystem

2.3.2.1 Overview

The purpose of this subsystem is to indicate whether the band is powered up, if the system is ready to be connected to, or if some device is already connected to the system.

2.3.2.2 Requirements

There need to be two LEDs, one for indicating power and one for indicating the status of connection.

1) Power Indicator LED: For power indication, this LED will be directly connected to the voltage regulator to indicate whether the band is powered on or not.

2) Connection Indicator LED: The LED indicating the status of connection needs to be a multi-color LED as it needs to indicate that the system is ready to be connected to and that a device is connected to it. This LED will be controlled by the status of connection of the microcontroller to an external device. So this will be powered by the microcontroller directly.

2.3.3 Sensing Subsystem

2.3.3.1 Overview

The sensing subsystem involves the microcontroller (ESP32) and an array of sensors. The purpose of this subsystem is to measure pollution levels and send the data over to the connected phone using bluetooth or WiFi. The Carbon Monoxide and Propane sensors connect to the ESP32 via the analog input pins available on the microcontroller. The Carbon Dioxide sensor is a digital sensor that connects via I2C communication protocol. At a specific time period, the sensor data converted to PPM measurements is sent to the connected phone via chosen wireless communication protocol. This subsystem specifically meets our first high level requirement.

2.3.3.2 Requirements

1) Sensor array: The sensor array will consist of 3 sensors that will respond to changes in pollution levels for Carbon Dioxide, Carbon Monoxide, and Propane. These sensors are 2.5cm in height including connection pins, so they have a low profile, allowing us to fit them in a bracelet (to meet high level requirement 3). The sensitivity needs to be adjusted using resistors such that we can detect small changes in Carbon Monoxide and Propane since small levels of these gasses can cause a lot of damage. The analog data from these sensors needs to be converted to PPM values accurately in order to find the appropriate pollution levels in the area. We will be using the following sensors to create our sensor array:
    a) MQ-2 Semiconductor Sensor for Combustible Gas[7]
    b) MQ-9 Semiconductor Sensor for Carbon Monoxide[8]
    c) Semiron's SCD4x Carbon Dioxide Sensor[9]
    All the sensors mentioned above use 5V power each.

2) Microcontroller (ESP 32): The microcontroller should be able to send the recorded PPM values to the app over the chosen wireless transfer protocol every 5 minutes or whenever a dangerous amount of any gas is detected. We need 3.3V to power our microcontroller, as per the datasheet of the ESP 32.[10]

2.4 Tolerance Analysis

For each sensor we have calculated the approximate error using graphs from their datasheet that catalog the relative readings based on various temperatures and humidity. Since we will be testing our bands in the spring in Champaign, we have assumed that the temperature will be between 15-25°C and the humidity will be around 55%[11]. To calculate the percentage error, we make use of the following formula:

$$\boldsymbol{\delta} = \left|\frac{(measured\ value - absolute\ value)}{absolute\ value}\right| * 100$$

1) **MQ-2:**

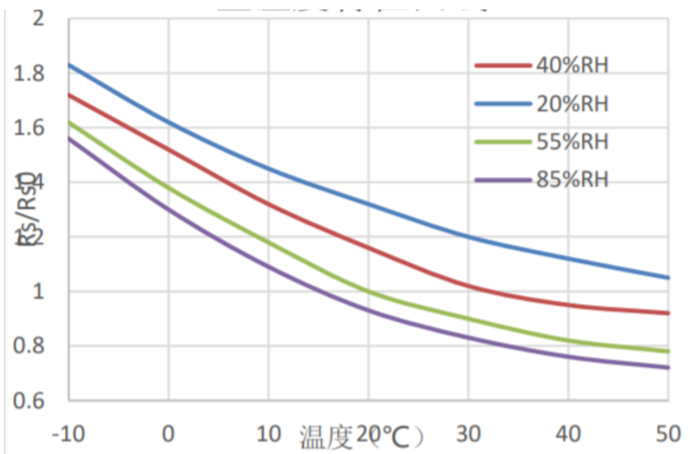

Figure-4: Graph of MQ-2's relative temperature/humidity characteristics[7]

The y-axis of this graph represents the ratio of Rs/Rso and the x-axis represents temperature. Rs (the measured value) is the resistance of the sensor in 2000ppm of propane in various temperatures and pressures. Rso (the absolute value) is the resistance of the sensor in 2000ppm propane under 20°C/55% relative humidity. Relative Humidity can be assumed to be 55% (the green curve) and operating temperature is 15-25°C.

<u>Relative error at 15°C:</u>

$$\left|\frac{R_s - R_{so}}{R_{so}}\right| * 100$$

$$= \left(\left|\frac{R_s}{R_{so}} - 1\right|\right) * 100$$

= (1.1-1)*100

= 10%

<u>Relative error at 20°C:</u>

$$\left|\frac{R_s - R_{so}}{R_{so}}\right| * 100$$
$$= (\left|\frac{R_s}{R_{so}} - 1\right|) * 100$$
$$= (|1 - 1|)*100$$
$$= \mathbf{0\%}$$

<u>Relative error at 25°C:</u>

$$\left|\frac{R_s - R_{so}}{R_{so}}\right| * 100$$
$$= (\left|\frac{R_s}{R_{so}} - 1\right|) * 100$$
$$= (|0.95 - 1|)*100$$
$$= \mathbf{5\%}$$

2) **MQ-9**:

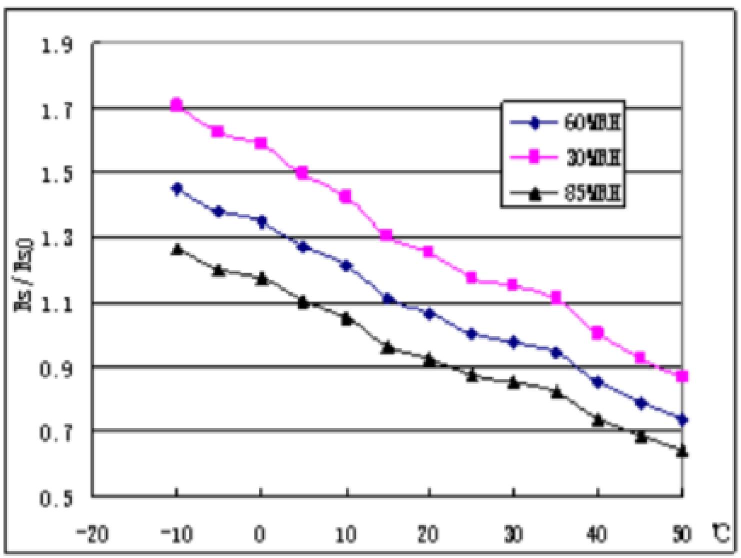

Figure-5: Graph of MQ-9's relative temperature/humidity characteristics[8]

The y-axis of this graph represents the ratio of Rs/Rso and the x-axis represents temperature. Rs (the measured value) is the resistance of the sensor in 150ppm of CO in various temperatures and pressures. Rso (the absolute value) is the resistance of the sensor in 150ppm CO under 20°C/55% relative humidity. Relative Humidity can be assumed to be 55% (the green curve) and operating temperature is 15-25°C. Relative Humidity can be assumed to be 60% (closest to 55%) (the blue curve) and operating temperature is 15-25°C.

Relative error at 15°C:

$$\left|\frac{R_s - R_{so}}{R_{so}}\right| * 100$$

$$= \left(\left|\frac{R_s}{R_{so}} - 1\right|\right) * 100$$

$= (|1.1 - 1|)*100$

= 10%

Relative error at 20°C:

$$\left|\frac{R_s - R_{so}}{R_{so}}\right| * 100$$

$$= \left(\left|\frac{R_s}{R_{so}} - 1\right|\right) * 100$$

$= (|1.07 - 1|)*100$

= 7%

Relative error at 25°C:

$$\left|\frac{R_s - R_{so}}{R_{so}}\right| * 100$$

$$= \left(\left|\frac{R_s}{R_{so}} - 1\right|\right) * 100$$

$= (|1 - 1|)*100$

= 0%

3) **SCD40:**
   At high accuracy, ±(40ppm + 5%) should be expected error. The range of high accuracy measurements is 400 - 2000ppm. At higher ppm, the inaccuracies increase, though the datasheet does not quantify the exact amount[9].

We have also highlighted possible faults in our system and our proposed solutions to them:

Pain point: Due to some external or internal factors, the sensors can occasionally transmit faulty values. These faulty values can severely affect the accuracy of our data.

Solution: We will try to mitigate this issue by maintaining a running average of the collected values and pushing this average to the server. By maintaining a running average, we can smooth out the inaccurate spikes in our data.

Pain point: Faulty bracelets will always be transmitting inaccurate data to the server which affect the accuracy of the data for other users.

Solution: We will try to minimize the effect of faulty bracelets by averaging the received value with the current value on the server side. The hope is that with enough non-faulty functioning bracelets in the region, the error can be offset easily as the average would be closer to the accurate values.

Pain point: The power expected by sensors and the microcontroller is very precise. Sensors expect a 5V input with a tolerance of ±0.1V and the ESP32 expects input of 3.3V but can accept 2.2 to 3.6V.

Solution: Using buck converters, which have a feedback loop each, we should be able to precisely control the voltage step down from ~9V from the battery to the desired 5V and 3.3V irrespective of the load.

3. Ethics and Safety

The biggest safety concern during the development of our bands lies in testing the bands for detection of harmful levels of gasses. To test carbon dioxide and carbon monoxide levels we plan on positioning the sensor at different distances from a lit flame (using a candle or bunsen burner). To ensure safety while doing these tests, we will only work in a well ventilated laboratory with a fire extinguisher. However, for testing propane detection we plan to purchase a propane tank. This is a safety issue as propane is flammable. Hence we will be testing with the tank only outdoors and ensure proper storage of the tank when we are not testing our project.

Since we are making a wearable band, we also need to make sure that all the circuitry is well insulated. We will create a proper housing for the pcb and batteries to ensure that no wires or circuitry is exposed. To create this enclosure we will make use of the machine shop or 3D print a suitable structure for our circuitry.

Since we will be tracking location data of the user, it could be a possible violation of IEEE[12] and ACM[13] privacy standards. To ensure privacy, we will log the location data but keep the user anonymous on our map. Our app will only associate pollutant data to the user's location and no trace of the user's identity will be recorded. Additionally, we only start using the user's gps location after appropriate in-app permissions are given.

References

1. “How Is Air Quality Measured?” NOAA SciJinks – All About Weather, https://scijinks.gov/air-quality/. Accessed 7 Feb. 2022.

2. “Some Google Street View Cars Now Track Pollution Levels | Colorado Public Radio.” Colorado Public Radio, Colorado Public Radio, 30 July 2015, https://www.cpr.org/2015/07/30/some-google-street-view-cars-now-track-pollution-levels/.

3. “How Cities Are Using the Internet of Things to Map Air Quality .” Data-Smart City Solutions, Harvard, https://datasmart.ash.harvard.edu/news/article/how-cities-are-using-the-internet-of-things-to-map-air-quality-1025. Accessed 7 Feb. 2022.

4. “Carbon Dioxide Health Hazard Information Sheet .” FSIS USDA, USDA, https://www.fsis.usda.gov/sites/default/files/media_file/2020-08/Carbon-Dioxide.pdf. Accessed 8 Feb. 2022

5. “What Are the Carbon Monoxide Levels That Will Sound the Alarm?” Kidde, 19 Aug. 2019, https://www.kidde.com/home-safety/en/us/support/help-center/browse-articles/articles/what_are_the_carbon_monoxide_levels_that_will_sound_the_alarm_.html.

6. “Proximity to Roads, NO2, Other Air Pollutants and Their Mixtures - Review of Evidence on Health Aspects of Air Pollution – REVIHAAP Project - NCBI Bookshelf.” National Center for Biotechnology Information, https://www.ncbi.nlm.nih.gov/books/NBK361807/. Accessed 9 Feb. 2022.

7. “MQ-2 Semiconductor Sensor for Combustible Gas .” Sparkfun, https://cdn.sparkfun.com/assets/3/b/0/6/d/MQ-2.pdf. Accessed 9 Feb. 2022.

8. “MQ-9 Semiconductor Sensor for CO/Combustible Gas.” Sparkfun, https://cdn.sparkfun.com/assets/d/f/5/e/2/MQ-9B_Ver1.4__-_Manual.pdf. Accessed 9 Feb. 2022.

9. “Sensors SCD4x Datasheet”. Sparkfun, https://cdn.sparkfun.com/assets/d/4/9/a/d/Sensirion_CO2_Sensors_SCD4x_Datasheet.pdf. Accessed 9 Feb. 2022.

10. “ESP 32 Series Datasheet.” Digikey, https://www.digikey.com/htmldatasheets/production/2845213/0/0/1/esp32-datasheet.html?utm_adgroup=xGeneral&utm_source=google&utm_medium=cpc&utm_campaign=Dynamic%20Search_EN_Product&utm_term=&utm_content=xGeneral&gclid=CjwKCAiA6Y2QBhAtEiwAGHybPRh4PBaR9pZXi6. Accessed 9 Feb. 2022.

11. “Weather in Champaign.” Champion Traveler, https://championtraveler.com/dates/best-time-to-visit-champaign-il-us/. Accessed 10 Feb. 2022.

12. “IEEE - IEEE Code of Ethics.” IEEE - The World’s Largest Technical Professional Organization Dedicated to Advancing Technology for the Benefit of Humanity., https://www.ieee.org/about/corporate/governance/p7-8.html. Accessed 9 Feb. 2022.

13. “Code of Ethics.” Association for Computing Machinery, https://www.acm.org/code-of-ethics. Accessed 9 Feb. 2022.