ScentBot

Authors: Aditti Ramsisaria, Caroline Pang, Eshita Shrawan
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—Anosmia (the loss of sense of smell) has been a major focus of public health in recent years due to the COVID-19 pandemic. In order to combat this issue, we have designed a ScentBot to autonomously detect and classify potentially hazardous domestic scents. ScentBot uses a sensor array and machine learning algorithm to autonomously traverse a map, locate, and classify a scented object. Due to the scope of this course, we have developed a scaled-down testing environment with one scented object in a square map, and have limited the classified scents to alcohol and paint thinner. Object detection and classification are typically done with computer vision, but these algorithms are not easily able to distinguish water from other clear hazardous chemicals. We are designing a system that can overcome this challenge by using scent detection algorithms.**

***Index Terms*—Design, Edge Computing, Motion Control, Robotics, Scent Classification, Sensor Fusion, TinyML**

1 INTRODUCTION

Over 19.1% of adults suffer from anosmia, with this figure being over 80% for adults over the age of 75 [1]. Common household chemicals like spirits and paint thinner fumes pose an increased risk to people with anosmia. Exposure to paint thinner fumes can cause eye, throat, and skin irritations [2, 3], while spirits can be a fire hazard. Currently, most identification systems rely on computer vision capabilities, which are unable to differentiate between different colorless liquids like isopropyl alcohol, paint thinner, and water. Additionally, scent detection methods like ion mobility spectrometry are limited to industrial applications, are immobile, and cost upwards of $15000, which makes such capabilities inaccessible to the normal adult suffering from anosmia. In this capstone project, we aim to build a mobile scent classification system that can map and locate the source of the odor to aid proper mitigation of present hazards. Particularly we focus on domestic settings with a subset of scents.

2 USE-CASE REQUIREMENTS

Given our focus on hazard detection in a domestic setting, we have identified three main use case requirements: (1) accurate classification, (2) collision-free navigation and location detection, and (3) accessibility. Our main considerations in deciding these use-case requirements were to positively contribute to public health and safety by developing a safe and reliable system to mitigate hazards.

2.1 Accurately classify different scents

Our first use-case requirement is derived from the user's need to be informed of the type of scent in order to effectively formulate a plan to mitigate the hazard. In order to satisfy this requirement, ScentBot must be able to detect and differentiate 2 types of scents from unscented objects: paint thinner and isopropyl alcohol. We have chosen these scents not only as they are easy to obtain and test with, but also because of their high TVoC and carbon content, hence presenting as hazards because of their toxic fumes. Moreover, ScentBot should be able to identify these scents with an accuracy of **over 95%**. Since gaseous sensors and distributions also vary due to environmental conditions like temperature and humidity, ScentBot must still be able to adapt to these changes and detect the correct scent.

2.2 Collision-free navigation and location detection

ScentBot aims to be an autonomous mobile system to aid those who cannot recognize potential hazards, so it is of the utmost importance that it must navigate to objects to classify them and must do so without causing any further damage and/or spills. We also want to let the user know that a hazardous object has been detected so that they can properly handle it. The sub-requirement ScentBot must achieve in order for this is obstacle avoidance. ScentBot must also spend 1.5s per step for data routing, collection and inference.

2.3 Accessibility

Our biggest motivation is for ScentBot to become as accessible and scalable for different hazards as possible. Hence, an important use-case requirement is that ScentBot has to be mobile. It should have the ability to autonomously traverse a map to identify hazards to remove as much human effort from the process of detecting hazards as possible. As such, we have also set a budget of $150 for our sensor array system, for which we will utilize I2C compatible sensors for easy interfacing. It should also give a clear visual indicator of the hazard type to the user when it is confident of the presence of a scented object.

3 ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

Our system implementation consists of five main subsystems hosted entirely on an Arduino Mega 2560. These subsystems include: (1) a sensing system for scent and obstacle detection (2) a motion system to issue motor commands based on Runge-Kutta odometry, (3) a scent localization system that determines the path of the robot, (4) a classification system for multi-class scent classification and (5) a visual alerting system comprising an LCD and Neopixel display. Fig. 1 provides a high-level overview of the interactions between these subsystems.

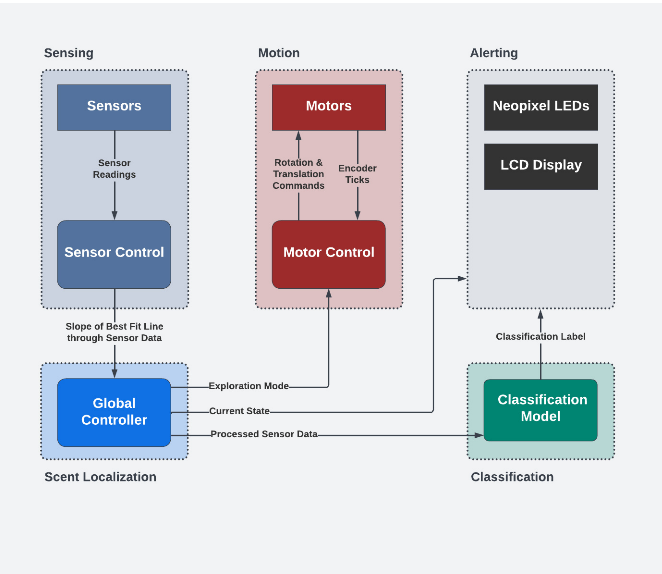

Figure 1: System Overview

The initial goal of the robot is to find and localize a scent using a combination of random exploration and targeted tracking based on gas concentrations. First, while a scent has not been detected yet, the local path planner computes a pseudo-random coordinate and orientation for the robot to navigate to. The motion system then issues motor commands to achieve this configuration using Runge-Kutta odometry for localization and a positional controller for precise navigation. If at any point an obstacle is detected by one of the three ultrasonic sensors on the robot, the motion system also contains logic to avoid the obstacle and reposition itself. As the robot moves, the sensing system samples the air flowing into the chamber and uses the sensor data to determine if the current sensor readings are indicative of a known scent in the environment. Upon detecting an increasing gradient of scent concentration during traversal, the robot enters a scan mode during which it stops and samples the air in 30-degree sectors spanning a total of 180 degrees. The robot then follows the path of maximum increase in scent concentration until the values reach a confirmation threshold. If a scent is confirmed, the embedded ML classifier is used to determine which scent it is out of three classes: alcohol, paint thinner, or an unknown scent. Fig. 2 illustrates the envisioned movement of the robot across the map for scent localization.

Our hardware is completely contained on the robot which comprises an MCU for computation, a gas sensor array for collecting data, three ultrasonic sensors for obstacle avoidance, two DC motors for movement with magnetic encoders for localization, batteries for power, and a fan to ensure continuous airflow. Our sensor array contains a multichannel gas sensor for measuring VOC, ethanol, NO2, and CO content, a sensor for measuring TVOC and eCO2 content, and a sensor for measuring the temperature, humidity, and pressure of the environment.

4 DESIGN REQUIREMENTS

We have developed a series of design requirements based on our use-case requirements and have divided them according to the three major subsystems: Navigation, Sensing & Classification, and Motor Control.

4.1 Navigation

A robust path planning system is an essential design requirement to meet our use-case requirement of collision-free navigation and location detection. The path planning system should generate a path that enables the robot to efficiently traverse the map and locate a scented object.

Specifically, the robot should be able to confirm the location of a scented object within a 2x2 meter map in under 3 minutes. The efficiency of the system in locating a hazardous scent is important because prolonged exposure to these scents could pose health risks. The hyperparameters of the random path generation algorithm, such as the ranges defined for minimum and maximum distances traveled in a single direction and the scent localization logic should be tuned in order to meet this three-minute metric.

Additionally, the path planning system should ensure that there are zero collisions between the robot and any obstacles. This is to avoid potentially spreading or spilling any hazardous compounds in an environment which would exacerbate the situation. The path planning system should take into account readings from ultrasonic sensors to avoid running into obstacles.

4.2 Sensing and Classification

The design requirements of our classification system are driven by the use-case requirement of accurately detecting and classifying scents. First, our scent classification model should have an accuracy of over 95 % when distinguishing between the fumes of alcohol, paint thinner, and ambient scent, and a false negative rate of less than 1%; the model should not incorrectly predict that there is no hazard when there actually is one present. This is important because it is more harmful if users are unaware of an existing hazard, as opposed to falsely being notified of a hazard. The model

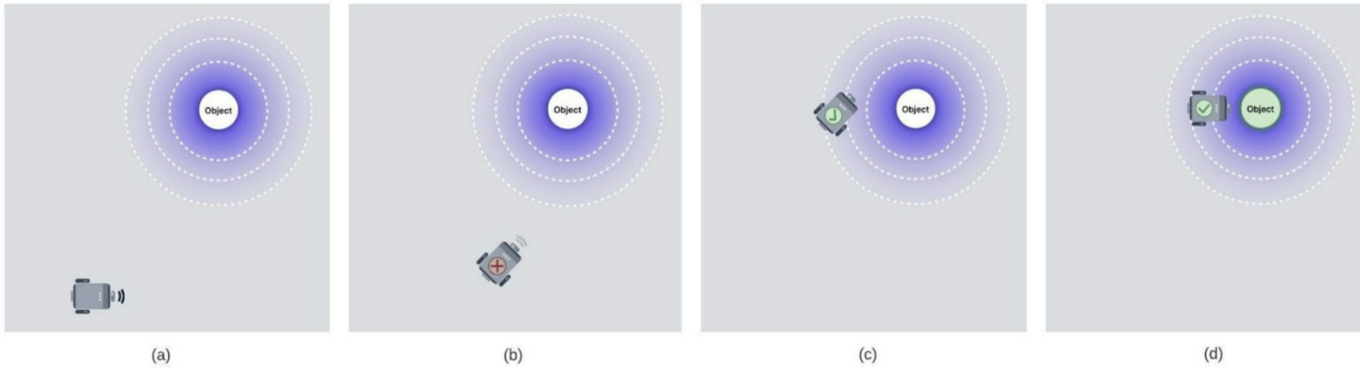

Figure 2: (a) Map for random exploration, robot moves pseudo-randomly in the test arena while sampling (b) Robot does not detect scent (c) Robot continues exploration, scent detected (d) Robot moves towards scent source for confirmation

should also be robust to slight variations in temperature and humidity as would be expected in an indoor environment. To meet this design requirement, we must generate a dataset for these scents in at least three different indoor environments with varying temperature and humidity levels.

The design requirements of our sensing system are guided by the use-case requirements for classification accuracy and efficient and collision-free traversal of the space. In order to balance these requirements for accuracy and efficiency, we define a lower and upper bound for the amount of time that the system should take for data collection and inference at a single step. The time allotted for data collection should be at least one second so that there is enough time for the air to be circulated through the robot chamber and over the sensor array for an accurate reading. Additionally, the time for inference at each step should be less than 0.5 seconds so that the system can be efficient. Altogether, the time at each step for data collection from the sensors should be less than 1.5s, and the classification that occurs when ScentBot is confident of the presence of a scented object should occur in less than 0.5s.

The robot should also be able to detect a scented object in front of it within 0.5 m for it to efficiently locate an object. At this point, it does not have to be able to accurately classify the scent, rather, it should not be predicting ambient scent with a confidence over 50%. Then, the robot can switch from random exploration to a more strategic approach which allows it to quickly track down the object.

Another overall design requirement of this system is that it should be able to accurately classify scents with over 95% confidence when an object is less than 5 cm away from the front of the robot/ sensor array. We define the metric of 5 cm based on our research into similar projects and the sensitivity of our sensors. We also think that this distance is a safe distance to maintain from the object while getting close enough to detect any potential fumes.

4.3 Motor Control

Based on the weight of our robot and components, our motors should be able to support and propel a load of at least 0.5kg.

The robot should be able to navigate within 5 cm of each waypoint within the determined path. This distance is less than the width of the robot and should therefore not affect the requirement of collision avoidance. For rotation, the robot should be able to orient itself within 10 degrees on either side of the desired orientation angle.

5 DESIGN TRADE STUDIES

5.1 Robot Design

5.1.1 Custom vs Off-the-shelf

Scents and odors are detected based on gaseous compounds, which are diffused based on principles of fluid dynamics. Odor molecules released from their source are spread by being carried off by a turbulent airflow field in the given environment [4]. To account for this, we needed a hollow, chamber-like structure with modifiable inlets and outlets to ensure that we can get a directional airflow over the sensors. We also needed to ensure that the motors on the robot had encoders for localization. Due to these constraints, off-the-shelf robots like the iRobot Create 2, BrickPi3, Pololu 3pi+ 32U4 OLED Robot, and ELEGOO UNO R3 were eliminated from our design considerations, and we opted to build a custom robot to fit our requirements.

5.1.2 Motor Selection

We decided to use brushless DC motors over servo motors or stepper motors. This is because of the precise control over rotational speed offered by these motors in addition to their high durability. Our design requires motors that are able to accommodate a range of speeds and torques and maintain reasonable efficiency, which is why stepper motors are unsuitable. Due to the constant feedback of servo motors to correct any drift, twitching may occur when trying to hold a steady position, which would be unsuitable for getting consistent samples due to the directional dependency. Moreover, we required motors with built-in encoders for localizing to the robot's position and using that as feedback to the control system. We chose the Adafruit N20 DC Motor with two hall effect sensors (magnetic encoders) with a 1:100 gear ratio, as it required a nominal 6V power supply, and easy control using PWM signals. Having a higher gear ratio implied better torque/acceleration usage.

5.1.3 Sensor Selection

Based on the chemical compositions of each of our target odors, we wanted to be able to detect specific gases - namely, VOCs, CO2, C2H5OH, and NH3 [9]. We wanted to ensure that the sensors were less temperature sensitive and more sensitive to lower concentrations to be able to robustly detect trace amounts of the compound present in the air. Based on prior research into different types of gas sensors such as MOX, IR, electrochemical sensors, etc. [5], we decided to use MOX sensors. A MOX sensor is a heated surface of a metal oxide that changes its electrical resistance depending on the oxygen content on its surface. Oxidizing gases like NOx (providing more oxygen than ambient air) increase the resistance, whereas reducing gases like VOCs (consuming oxygen by being combusted on the metal oxide surface) reduce the resistance, and this change in resistance can be measured [6]. As a result, these sensors are more sensitive in detecting compounds such as CO, H2S, aldehydes, alcohols, and ketones. Due to our accessibility requirement, we also wanted our sensors to be small and portable, which is why MEMS-based sensors were particularly appealing. The last consideration was cost, as we wanted a cost-effective sensor array. Due to these considerations, we decided to use the Grove Multichannel Gas Sensor V2 sensor which has four measuring units for VOCs, CO, C2H5OH, and NO2. Additionally, we decided to use the ENS160, a MOX sensor for measuring eCO2 and TVOC content. Since gas concentrations and sensor measurements are sensitive to environmental conditions - temperature, humidity, pressure - we also decided to include a BME280 in our sensor array.

5.1.4 MCU Selection

Our primary contenders for a microcontroller/processor were the Arduino Mega, Arduino Uno, NodeMCU, Raspberry Pi, and the Wio Terminal. We needed enough GPIO pins to be able to connect to a motor driver and have interrupts for reading from the hall effect sensors on the motors, as well as I2C ports for connecting multiple sensors. Additionally, we needed the ability to supply 5V power to the sensors.

The NodeMCU would give us enough GPIO pins and an inbuilt ESP8266 with well-documented usage for IoT applications. The Raspberry Pi would also have enough GPIO pins, along with wireless LAN and Bluetooth connectivity, with the added ease of using Python. However, the Raspberry Pi and the NodeMCU would be unable to serve as a 5V power supply for encoders and other sensors, and would also require additional boards for motor control. The Wio terminal allows for enough computing power to run a machine learning model on the edge, with easy-to-use interfaces using Edge Impulse, but we would be limited to only using Grove sensors. The Arduino Uno is able to supply 5V of power, but does not have enough memory capability to support our system. Therefore, we decided to use the Arduino Mega 2560 microcontroller which has sufficient system memory, GPIO pins, and a 5V output pin for powering our sensors.

5.1.5 Path Planning Approach

Due to the high dependency of our project on the range of our sensors, considered two different approaches for path planning. Ultimately, we based our decision on what fits better for our use case and the complexity of implementation. One approach is to use random exploration. Here the robot pseudo-randomly computes a distance within a fixed range along with an orientation and as its next configuration, until it can start detecting a scent and start localizing towards it. In this case, the dependency on our sensor sensitivity is high, as we don't want the robot to have too much random motion till it can get within detection distance of the object, as this would be incredibly inefficient. This means that the sensors must be sensitive enough to be able to detect a scent from a substantial distance. However, this allows for motion control to be less rigid and precise, as we can rely on the robot reorienting itself at every step to achieve the desired configurations as it makes its way toward the object.

5.1.6 Classification & Model Selection

From prior research into building e-noses, we determined that using simple arithmetic or ratio-based differentiation would not be enough to distinguish between different odor compounds. In testing our sensor readings with different scents, we discovered that all channels of gases go up in some quantity, over which it is not easy to differentiate based solely on a high threshold. Consequently, we require a pattern recognition model to classify our use-case scents. We decided on trying several different machine-learning approaches for this multi-class classification problem such as SVM, decision trees, and neural networks.

5.1.7 Choice of Model

In discovering the kind of model to use, we primarily explored Google Cloud Platform's Neuton AI platform and MicroMLgen by Eloquent Arduino. Neuton AI creates a TinyML C library based on a dataset provided to it. It will create library functions to initialize and run a convolutional neural network on a microcontroller. MicroMLgen has the capability to turn a Support Vector Classification (SVC) model into a C library. When exploring the performance of both platforms, we observed a SVC model had an F-1 accuracy of over 90% on a train-test split of our dataset, while the convolutional neural network had an accuracy of just over 50%. Moreover, Neuton AI did not provide support for the Arduino Mega, which is why we decided to move ahead with our model on MicroMLgen.

5.1.8 Integration of Model

The integration of where the model should perform inference had multiple tradeoffs. We decided to explore inference at every translation step of the robot, or to classify a scent only after it observe high enough sensor readings. Upon testing our assembled robot, we discovered the sensors have false positives, and are affected by the amount of people in a room, along with various other geographical considerations. Based on this, conducting inference at every step with our model could lead to a high number of false positives. We hence decided to introduce a hard thresholded value of sensor readings across various channels to account for the different scents ScentBot would encounter.

5.2 Networking approach

5.2.1 Cloud Computing

Various networking approaches were considered during our design to process data received from our sensors, and the ability to host a multi-classification model. Cloud computing was a desirable approach for us because of the seamless ML pipeline we could integrate. There are several cloud provisioning services that have IoT-specific platforms. The primary ones we considered in our implementation were ThingSpeak, AWS IoT, and Azure IoT Central. ThingSpeak offers the capability to integrate MATLAB analysis with the IoT data that is collected. While ThingSpeak interested us because of the intensive documentation that was present with its usage with the ESP8266 Wi-Fi module we had chosen for our design, the only way to analyze any data was through manually exporting it as a .CSV file to MATLAB. This would hamper our design requirement of total exploration time and our use-case requirement of accessibility through the autonomous nature of the robot. Azure IoT offered varied hardware drivers when compared with AWS. Our initial design was therefore a combination of using Azure IoT with the ESP8266 NodeMCU module. To communicate data over Wi-Fi, we investigated AT commands, which would communicate with a cloud instance using MQTT. While this provided updated data to a web server, there was a considerable delay and lack of integration with either AWS or Azure Cloud. Moreover, we discovered limitations with communicating between microcontrollers, which would add high latency per step to our robot. We explored both Serial UART communication and I2C Master-Slave communication, and edge computing provided us with the highest locality and lowest latency for ScentBot's user requirements, further discussed in the next subsection.

5.2.2 Edge Computing

When we discovered the high-speed control loop our robot needs to maintain in order to traverse and localize correctly, we started considering hosting the entire system locally. Edge computing would offer us the ability to reduce our latency to almost instantaneous. We considered various platforms and methods to generate a multi-class classification model as described in Section 5.1.8, deciding on MicroMLgen. With MicroMLgen, we could observe the performance of the model in Python and make changes without having to perform a lengthy upload process on the Arduino until we were confident in its performance.

6 SYSTEM IMPLEMENTATION

6.1 Robot Design

Our robot is custom built and consists of a microcontroller, sensor system, motor system, alerting system, and laser-cut chassis. The CAD drawing for our robot chassis is depicted in Fig. 10. The layout of the internal components is shown in Fig. 3, and the full schematic can be found in Fig. 9.

6.1.1 Microcontroller

We use an Arduino Mega 2560 as our microcontroller because of its versatility and compatibility with our sensing and motor subsystems. The additional storage offered by the Arduino Mega enables us to control the multiple software subsystems (Fig. 1) of the robot and use edge computing to compute classification of different scents.

6.1.2 Batteries & Power Regulation

The system contains two separate 9V power supplies. One line is used to power the Arduino Mega, which then provides power to the sensors, encoders, and fan via its 5V and 3.3V output pins. The other 9V line supplies power to the L298 motor driver which then powers the two 6V DC motors.

6.1.3 Sensors

Our sensing system hardware comprises six sensors for obstacle avoidance and scent detection. It also includes a fan for air circulation over the gas sensors.

For obstacle avoidance, the robot has three HC-SR04 ultrasonic sensors, with one sensor located at the front of the robot to prevent it from translating into obstacles and one located on each side of the robot to prevent it from rotating into obstacles.

The remaining three sensors are used for scent detection and classification. This includes a Grove Multichannel

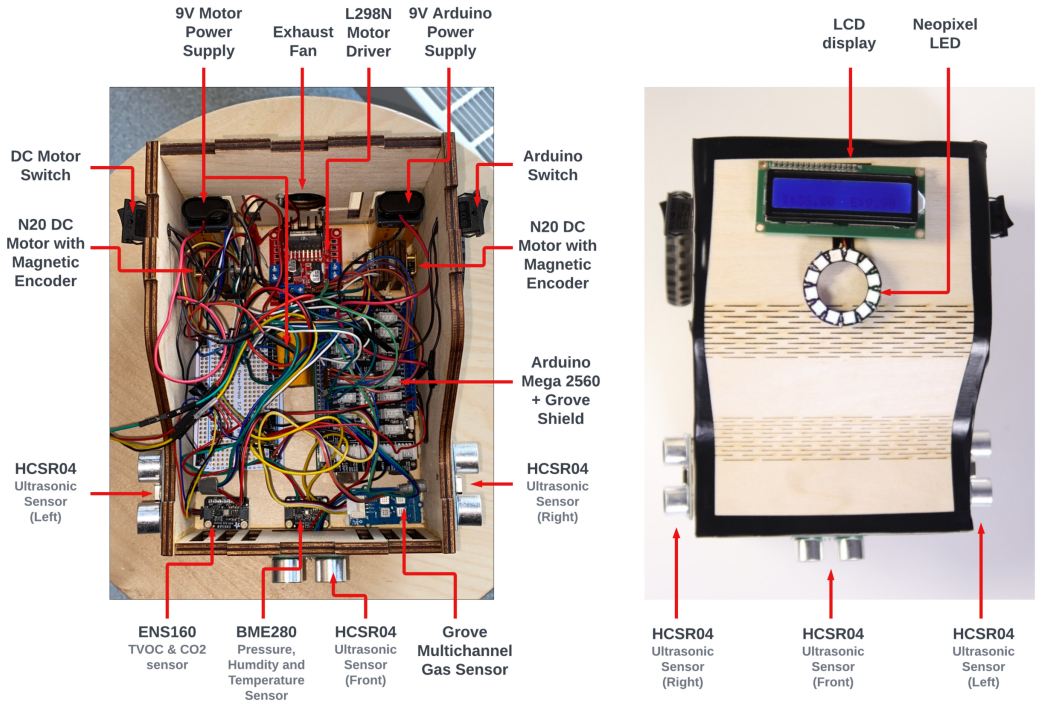

Figure 3: Robot Interior (left) and Exterior (right) Top View

Gas Sensor, ENS160 TVOC and eCO2 sensor, and BME280 temperature, pressure, and humidity sensor. The gas sensors measure the concentrations of various chemical compounds and VOCs found in the scents we aim to classify. The BME280 is also necessary to measure the temperature, pressure, and humidity of the environment since these factors can affect scent classification. These three sensors are located inside the chassis of the robot so that the sensor readings are less affected by external air movement and currents. A small fan located at the back of the robot is used to pull air through the robot chamber and over the sensors for efficient air circulation and directional readings.

6.1.4 Motors

We use two N20 DC motors with magnetic encoders and a 1:100 gear ratio for propelling the robot. The gear ratio and rated torque of the motors ensure that we are able to propel a load of at least 0.5kg, which is less than the weight of our robot. The encoders on these motors have a resolution of 14 ticks per rotation which allows us to precisely track and control the distance traveled by the robot. Our motors are connected to 65 mm diameter wheels which translates to a resolution of around 14 mm.

6.1.5 Alerting

We have an LCD screen and Neopixel LED at the top of our robot to indicate the current state of the robot as well as the classification results. The Neopixel color key is shown in Fig. 4. The LCD screen displays text corresponding with each of the Neopixel states.

Figure 4: Neopixel Color Key

6.1.6 Robot Chassis

Fig. 10 contains the dimension specifications of the CAD for our robot. We designed our chassis to be constructed from 6mm and 3mm laser-cut plywood. This material is both light and flexible which makes it the optimal choice for our robot.

We defined the base dimension to be 192x144mm in order to accommodate all of our hardware components. We designed the layout specifically such that the sensor array could be placed at the front of the robot and the motors could be placed at the back. The front and back panels of the robot contain slats to allow airflow through the robot and over the sensors. The base plate contains a custom contraption to help stabilize the motors and secure the ball caster. Finally, the top and side panels are designed to have a sloped profile to make the robot more aerodynamic.

6.2 Scent Localization

A global controller hosted on the robot MCU determines the localization strategy of the robot. There are two main path planning strategies depending on the state of the robot: Random Exploration and Scanning. A state diagram can be seen in Fig. 5.

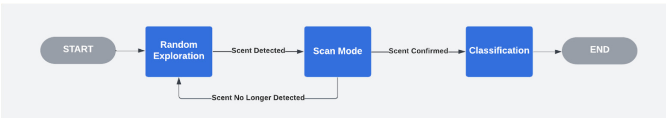

Figure 5: Robot State Diagram

6.2.1 Random Exploration

While a scent has not been detected, the planner will use a random exploration strategy to traverse the space. The planner will compute random distances and orientations within a fixed range for the robot to move to using Arduino's random number functions. This information is then relayed to the motion system to execute the corresponding motor commands.

6.2.2 Scent Detection

While the robot is randomly exploring a space, it is constantly reading from the gas sensors at a frequency of 10Hz. For scent detection, we look at the ethanol channel of the Grove sensor and TVOC channel of the ENS160 sensor in particular because both of these values rise when exposed to the chemicals we aim to classify. Additionally, using values from two separate sensors for detection helps prevent false positives and false negatives. The raw sensor values are aggregated over a 1 second sampling period and a best fit line through the readings is also computed. If the slope of the best fit line surpasses a set threshold, we determine that a scent has been detected, and the robot transitions into Scanning mode.

6.2.3 Scanning

In Scanning mode, the robot will attempt to localize the source of the scent by following the path of increasing concentration gradients until the sensor readings surpass a confidence threshold. The robot will first conduct a 180 degree scan of the environment centered around the direction in which the scent was detected consisting of 6 scans at 30 degree increments. The duration of each scan is 5 seconds to allow time for air to flow into the robot chamber and to account for delays in the sensor readings. After the 180 degree scan, the robot will rotate to the direction which had the maximum scent concentration and translate forwards for a fixed distance. It will repeat this scanning process until the confidence threshold is reached and then perform classification.

We have also included logic to exit out of the scanning mode in case of a false positive detection. This can sometimes occur due to random air currents or noise in the sensor readings. The robot will return to random exploration from scanning mode if one or both of the two gas sensors report consistently low values during the six scans.

6.2.4 Obstacle Avoidance

The robot also contains logic to avoid obstacles using readings from the three ultrasonic sensors. We define a 10cm threshold to determine whether an object has been detected by one of the sensors. The robot's corrective action depends on whether the obstacle is detected by the front or side sensors, and whether the robot is in Random Exploration or Scanning mode.

During random exploration, if an obstacle comes into the view of the front ultrasonic sensor, the robot will halt its current motor commands, reverse, and then compute a new pseudo-random coordinate in the 180 degree range behind it. If one of the side ultrasonic sensors is triggered, the robot will turn 90 degrees away from the direction of the obstacle detected.

When in Scanning mode, the behavior is different. This is to ensure that the robot can get as close to the potentially scented object as possible and avoid navigating away if it is nearby. If one of the side ultrasonic sensors is triggered, the robot will reverse slightly to avoid colliding with the object while rotating to its next scan position. After reversing, the robot will continue with its 30 degree rotations. The front ultrasonic sensor will not trigger during scanning since only rotation is being performed.

6.3 Robot Control

6.3.1 Odometry

For robot localization we will be using odometry information computed using encoder ticks. Using the two hall effect sensors on each motor, we use a hardware interrupt on one encoder and read from the other encoder on every rising edge. From this we get the number of encoder ticks. Using this information and the Runge-Kutta method, we get the x, y, and θ position of the robot as follows:

$$\omega_{wheel} = \frac{\pi}{180}((n_c - n_p)/\Delta T) \tag{1}$$

$$v_{wheel} = \omega_{wheel} \cdot r \tag{2}$$

$$\omega_{robot} = (v_r - v_l)/L \tag{3}$$

$$x = v_{robot} \cdot \cos\theta, y = v_{robot} \cdot \sin\theta \tag{4}$$

Where ω_{wheel} is the angular wheel velocity, n_c is current encoder ticks, n_p is previous encoder ticks, Δ T is timestep between encoder ticks, v_{wheel} is the tangential velocity of point of contact on the wheel, r is the radius of the wheel, L is the length of wheelbase and w_{robot} and v_{robot} are the angular and linear velocity of the robot respectively.

This system of non-linear equations can be solved by Runge-Kutta Methods as follows:

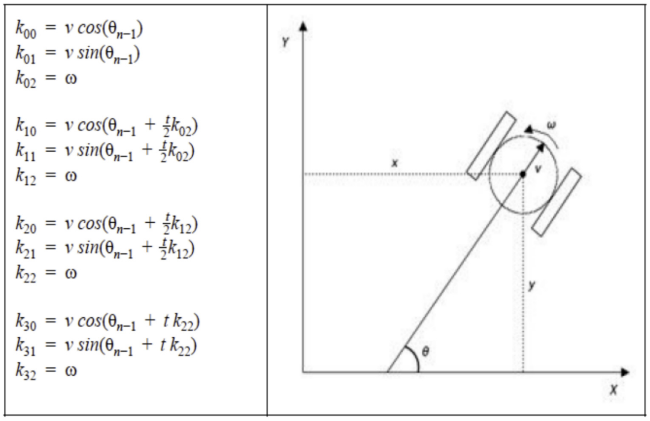

Figure 6: Runge-Kutta Estimation for Differential-Drive Robot

Using these equations, the configuration (x, y, θ) of the robot at time n is given by:

$$\begin{bmatrix} x_n \\ y_n \\ \theta_n \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \\ \theta_{n-1} \end{bmatrix} + \frac{t}{6} \begin{bmatrix} k_{00} + 2(k_{10} + k_{20}) + k_{30} \\ k_{01} + 2(k_{11} + k_{21}) + k_{31} \\ k_{02} + 2(k_{12} + k_{22}) + k_{32} \end{bmatrix} \tag{5}$$

6.3.2 Feedback Control

Because the motors we are using are not perfectly calibrated, the robot will sometimes overshoot the desired position despite receiving the correct signals. Therefore, we added error correction logic using positional feedback control. We obtain the robot's absolute position and orientation in a global coordinate system using odometry. If at any point the distance between the robot and target position begins to increase, the robot enters a "hard stop" condition which terminates the current motor commands. Then, new commands are computed using the updated current position and the same desired position until the robot reaches the target coordinates.

6.4 Classification

Our classification system uses a linear SVC model to classify scents based on a feature vector generated from the raw sensor data. We sample at 10Hz from each channel of each sensor to generate feature vectors. After preprocessing and generating our feature vector, it is passed into our classification model.

6.4.1 Dataset Generation

Our team generated a dataset of over 10000 samples per scent (alcohol, ambient, and paint thinner scent) by placing an object at varying distances in front of the robot to create the model. For scented object data collection, we placed 3 milliliters of liquid on a cotton ball and varied its orientation and distance in a range of 0.5m over the 180-degree environment in front of the robot. We collected this data over two days to ensure different temperature, humidity, and pressure conditions can be collected to improve the performance and robustness of the machine learning model.

6.4.2 Data Pre-processing

First, we normalize and standardize the data such that every value is between 0 and 1. This is done since the units of each measurement channel are different. We then compute the average, minimum, maximum, root-mean-square, and standard deviation of the gas sensor readings over a period of one second and concatenate them to generate our feature vector. Since there is not much variability in the temperature, humidity, and pressure, we only use the average over a one-second time period for these sensors. Additionally, in our data collection process, we observed that the ENS160 sensor has periods where its readings drop to 0, hence we also remove any such data points in the preprocessing. Overall, we use 4 channels from the Grove sensor, 2 channels from the ENS160, and 1 channel from the BME280 which results in a $5 * 7 = 35$ dimensional feature vector.

6.4.3 Classification Algorithm

The classification algorithm is a Support Vector Classification model. The SVC model is a hyperplane optimization for linearly classifiable patterns and extends them to unseen data points by the transformation of the original data into a new space utilizing a kernel function. While creating the decision surface for classification, there exist training data points that are the most difficult to classify, and hence have a direct bearing on the formation of the hyperplane. The decision function is hence decided by this subset of training points called the support vectors of the model.

The model outputs a set of weights in the form of a linear equation which will predict the value of the target variable, in this case, the scent label of "paint thinner," "alcohol," "unidentified" and "ambient." In training the model, we implemented a train-test split across our dataset of 60% of the dataset being used for training, and 40% being used to test the performance of the model.

The goal of the SVC model is to determine the optimal placement of a margin for classification. For example, consider the feature vector defined by $\mathbf{x}$ and the target vector defined by $\mathbf{y}$. Consider the hyperplanes where for all data points $\mathbf{w} * x_i + b \geq 1$ for $y_i = 1$ and $\mathbf{w} * x_i + b \leq -1$ for $y_i = -1$. The two hyperplanes hence become $H_1 = \mathbf{w} * x_i + b = 1$ and $H_2 = \mathbf{w} * x_i + b = -1$. The margin is defined as the area that separated the closest positive point and closest negative point of the two hyperplanes.

The optimal placement for a margin would be to find the biggest margin possible, to account for all data points. The SVC model creation is hence a constrained optimization problem to minimize $||\mathbf{w}||$.

ScentBot's SVC model was created using sklearn. A linear SVC was decided upon due to its high performance for our set of scents and the low memory that it utilizes on the Arduino. An SVC in general also offers greater dimensionality reduction due to only being determined by the support vectors, which are a subset of the original data points. Using MicroMLgen, this was ported into a C library, which performs the methodology described above by storing all the constants and weights inside the C library file. It computes the needed dot products at the time of inference, as described in the next subsection.

6.4.4 ScentBot's inference methodology

When ScentBot encounters high sensor readings as determined by our thresholds, the most pre-processed sample of data is run through the SVC model. 162 float kernels are created by the linear model by computing dot products of the feature vector with each support vector of the model, as defined by the following: $k_i = w \cdot u + b$. The model then computes 6 linear equations of kernels in a one-vs-one voting model of classification. A one-vs-one classification implies that each class will be pitted against the other. In our dataset, 0 is the value for paint thinner, 1 is the value for alcohol, 2 is the value for unidentified scents and 3 is the value for ambient scent. Here, the voting will be between 0-1, 0-2, 0-3, 1-2, 1-3, and 2-3 for a total of 6 votes produced for determining the label of the data feature vector.

7 TEST & VALIDATION

7.1 Testing Setup

For our testing arena, we have fixed some environmental conditions described for our approaches below.

1. The arena is a walled arena of **2m x 2m x 0.5m**. A walled arena was chosen to test the obstacle avoidance of ScentBot and to also limit external unpredictable airflow to the robot.

2. The arena contains **one singular scented marker** placed anywhere on the arena, which is assumed to have a radial distribution to aid our robot in searching for the object.

We define one trial by our design requirement of the time of completion of 3 minutes. The specifications of each trial were defined as shown in Fig 7. below. One scented object was placed in one of 9 positions shown in Fig 7 (a). The robot was then placed at each of the four corners of the arena, making sure the obejct is always at least 1m away from the robot as shown in Fib 7 (b). We measured the distance from which ScentBot conducts its first true positive scan (Fig 7. (c)), the total time of convergence, and the classification label returned by ScentBot. (Fig 7. (d)). This generated a total of 64 trials that we conducted in Techspark, 32 trials with paint thinner and alcohol each.

7.2 Results for Accurate hazard classification

Per our use-case requirement of accurate hazard classification, we placed that the robot must be able to classify scents with a True Positive Rate of over 95% and a False Negative Rate of ¡1%. Since we only classify based on a high sensor threshold, over our trial runs, ScentBot converged onto the scented object in all trials with a classification **accuracy of 98.4%**. Observing the results, our predictions were matched with our implementation. However, this involved a tuning process of when to calculate the inference, which depends on the sensitivity of our sensor array. We went through several iterations of thresholds for when to begin inference for our robot, as we only want to trigger when the robot is highly confident that there is something hazardous present in front of it.

7.3 Results for Safe Navigation

As defined in our use-case and design requirements, we wanted ScentBot to travel within 5cm of the randomly generated waypoint, with the ability to self-correct itself, while avoiding all obstacles in its traversal. This was unit-tested through coordinates and values given back by the encoder to determine when the self-correction was triggered and that it executed correctly. In our design process, we discovered several blind spots in the design of the robot and the placement of the ultrasonic sensor. We introduced two ultrasonic sensors to the sides of the robot for the final design. While this improved our obstacle detection, there were still some limitations in meeting this use-case requirement of collision-free navigation. In our trials, ScentBot made contact with the object **32% of the time**.

These limitations were due to the object placement in the gaps where the sensors cannot detect an obstacle. Here, we observed a trade-off between complexity and the need for a domestic use case with using ultrasonic sensors or moving to a more complicated obstacle-sensing system like Light Detection and Ranging (LiDaR). We wanted ScentBot to be an easily calibrated mobile system, and we envision that with more sensor tuning and data collected for a classification algorithm, the detection distance of such a system can be increased, hence reducing the risk of making contact with the object. The goal for ScentBot to converge onto the object was given a higher priority in our design process to show the capability of a scent detection system.

7.4 Results for Efficient navigation & low latency

According to our design requirement of finding a scented object in a 2m x 2m space in under 3 minutes (180 s), we tested for the total time taken to converge for ScentBot. On average, ScentBot takes 2.5 minutes (161.6s) to converge onto the object based on different configurations. Our requirement for low latency (1.5s per step) was satisfied as the robot samples at 10Hz while continuously traversing,

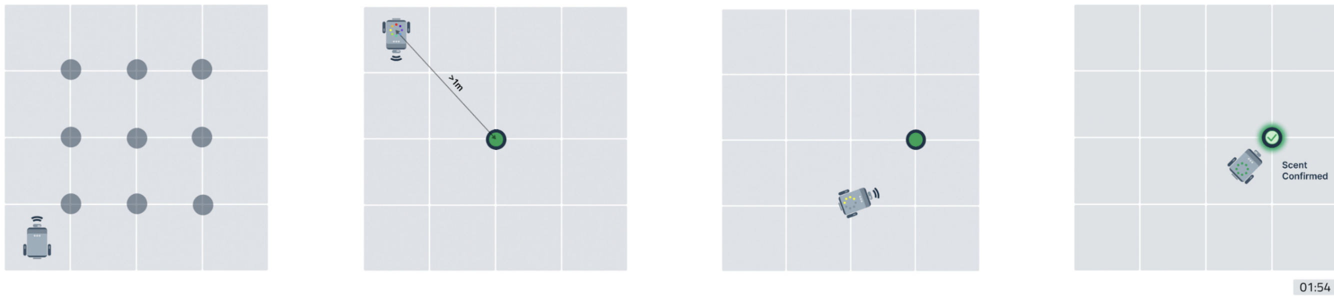

Figure 7: (a) Map for testing, showing 9 configurations of object placement (b) Robot is always placed at least 1m away from the scented object (c) Testing for first true positive scan made by robot (d) Testing for correct classification label and total time taken for convergence

and inference from the classification model was observed to be almost instantaneous. In our design and testing process, we observed a trade-off between random exploration and a planned path for the robot. While the robot can effectively traverse any space because of random exploration, it was observed that it takes longer to explore in some configurations than others as a result of this.

7.5 Results for Maximum detection distance

Per our design requirements for scent localization, we wanted the robot to be able to detect a scent from a radial distance of 0.5m. This metric was decided on assumptions made about the radial distribution of the scent. In our testing, ScentBot was able to begin localization toward a scent from an average distance of **0.22m radially.** We observed that sensor sensitivity and directionality are affected by the natural airflow inside a room. The radial distribution of the scent did not hold true in our test setup as a result. This combined with the small concentrations of paint thinner and isopropyl alcohol we tested limited ScentBot's performance toward this metric.

7.6 Results for Accessibility

For our use-case requirement of accessibility, we set a budget of \$150 for the sensor array system. Our bill of materials, attached to this report, is an indicator that we have completed this requirement, as our sensor array costs **\$66.89.** Additionally, we wanted to clearly communicate to users the kind of scent detected through different mediums while keeping in mind user accessibility. We came up with a color-coded neo-pixel key for users, along with the LCD message to cater to the kind of output preferred by the user.

8 PROJECT MANAGEMENT

8.1 Schedule

The schedule is shown in Fig. 11. As an overview, from our initial design report, we faced additional challenges in networking across the cloud. Hence, we moved data set generation to Week 11 and the making of the classification model to Week 12. Tuning and thresholding of the sensors moved testing to our initial 2-week slack period.

8.2 Team Member Responsibilities

Aditti	• Robot navigation and control • Robot assembly • Scent localization
Caroline	• Sensor system assembly • Robot assembly • Scent localization & obstacle avoidance
Eshita	• Classification algorithm • Alerting System • Sensor system assembly • Dataset Generation
All Members	• Hardware integration • Software integration • Field Construction • Scenario design & testing

8.3 Bill of Materials and Budget

See Fig. 8 for the full BoM. For the sensor array, we are utilizing the BME280, Grove Multichannel Gas Sensor, 3 HC-SRO4 ultrasonic sensors, and the ENS160: Air Quality Multi-Gas Sensor. For the robot, we have laser-cut wood frames for our design and have used a fan and N20 DC motors to make it mobile. From our design report, we had not planned for 2 more ultrasonic sensors, the Arduino Mega and its corresponding Grove Shield, along with additional quantities of batteries needed for testing. We also obtained cardboard for our new testing setup from our personal inventory.

8.4 Risk Management

During the ideation phase, we decided to go ahead with a research idea that we were all curious about but at the same time could contribute our prior experience to. Aditti had taken robotics courses and control courses and knew about scent classification work and signal processing for sensor fusion. Caroline had experience in CAD modeling and hardware systems, and Eshita had experience in building software solutions as well as machine learning algorithms.

We identified certain risks with the project quite early on in the planning and scheduling phase. The biggest risk factor was the sensitivity of the sensors. We focused on a random exploration approach for the navigation module of ScentBot, but in case the sensors were not sensitive enough to meet our use case requirements and could not converge, we had design approaches in place to implement a vision-based path-planning approach to make ScentBot travel close to each object on the map for testing. We accounted for this possibility in our schedule early on and made sure to allocate time and resources to first identify if our planned approach was feasible or not.

Apart from the sensor sensitivity, the risk was primarily in integrating different software components with our self-designed and assembled robot. We planned our schedule to make sure that we could get through a few iterations of the robot design if we needed to without compromising the software stack development. Therefore, we focused on developing the hardware prototype for the robot early on in the semester.

In terms of budget, given how we defined our use-case requirements, we needed to ensure that we did not overspend on sensors. However, we also wanted to make sure that there were no hindrances in the development of the robot due to hardware reliability issues. For this, we ordered copies of our sensors, as well as additional servo motors in case the DC motors did not work out. We made sure to balance our costs accordingly and relied as much on inventory items as we could. The one issue that we ran into during this process that we did not account for earlier on was the flash memory available on our initially chosen microcontroller, the Arduino Uno R3. However, due to our careful usage of monetary resources and a more modular development process, we were able to adapt to this setback quickly.

9 ETHICAL ISSUES

ScentBot aims to contribute to the improvement of public health, safety, and welfare. The system is designed to be accessible and user-friendly, while also providing accurate and timely detection of harmful fumes by machine learning with cost-effective hardware. Our target audience comprises individuals who suffer from anosmia or have a diminished sense of smell. This could include people with a range of medical conditions such as age-related anosmia, head injuries, nasal diseases, or genetic disorders affecting the sense of smell. Additionally, people who are sensitive to certain smells or suffer from allergies could use this technology to identify and avoid certain odors that may cause adverse reactions.

However, we acknowledge certain ethical considerations for the project and its usage in public settings. Firstly, a certain amount of technical know-how is required to operate the device and interpret results. Changes in air currents might affect odor distribution and could lead to inaccurate results. Users who lack knowledge about the substances being detected may not fully understand the implications of the results. This lack of understanding could lead to misapplication of the technology or failure to take appropriate action when necessary. False positives and negatives must be taken into consideration as they could lead to unnecessary alarm or exposure to harmful substances. The device is intended to be used as an early-detection device rather than a verification system. We do not recommend relying solely on the scent classification system to detect hazardous fumes and neglecting precautions like proper ventilation, personal protective equipment, and other safety measures necessary for the safe handling of hazardous substances.

Our proposed solution includes the use of safe and reliable hardware and software components and rigorous testing to ensure the system's reliability and safety in various environments. Furthermore, we prioritize privacy and data protection laws to ensure that user information is secure and not compromised. We harp on the intended use-case for ScentBot to be only within domestic environments and not used solely as a way for identifying harmful substances. Within the use-case scenario, we envision positively contributing to Accountability, Trust and Responsibility. We also acknowledge through our reflections that expansion and use of the product can expand to industries and applications with higher criticality and can be used maliciously regardless of our intention.

10 RELATED WORK

The purpose of this design is to contribute to the increasing awareness around an "AI Nose," where everyday household scents can be recognized using TVoC sensors [7].

The main difference here is that their classification algorithm and setup can only detect scents from a close distance and has to be manually placed near the scent in order to smell it. Moreover, household scents like coffee and citrus do not fit a use-case requirement that would help mitigate hazards for people suffering with anosmia. There are also a variety of industrial gas leak detection robots [8]. These robots work off of a host of sensor systems, including thermal gas detection cameras and spectroscopy to detect gas leaks. This solution, while ideal for detecting hazards, is meant for big factories and facilities and is inaccessible to the common user. Our solution differs from these existing solutions in that we are developing a domestic solution like the AI nose, but with the capabilities of detecting hazards like the gas leak detection robot would do.

11 SUMMARY

We have built a mobile scent classification system that can detect and locate the source of different odors to help prevent hazards. We built this solution with the goal of positively contributing to public health and safety with our robot. We used our own design to assemble the robot which utilizes a sensor array to measure various chemical compounds that are emitted by scented objects like TVOCs, ethanol, ammonia, and CO2. We read and processed this data using an Arduino Mega, which is a self-contained system hosting our entire codebase including a TinyML model for classification, scent localization logic, and motor control algorithm. With this project, we hope to contribute to the field of robotics and bring a focus to hazard prevention in domestic settings. The system is designed to be accessible and user-friendly, while also providing accurate and timely detection of harmful fumes by machine learning with cost-effective hardware.

11.1 Future work

We believe this project has a lot of potential for future research and expansion with an array of applications. With the scalability of our sensor array and capability of our TinyML classification to handle multiple classes, our project can be adapted to domestic use solutions catered to recognizing different hazardous scents, as well as potential industrial applications - in environments with high concentrations of odorous substances, such as factories, chemical plants, or laboratories, as it could help them detect and avoid hazardous fumes. Making the system mobile and autonomous makes it suitable for search and rescue operations where it might be dangerous to send in human assets. Emergency responders such as firefighters or hazmat teams could also benefit from this technology as it could help them identify potentially dangerous chemicals or gases quickly and accurately, allowing them to take the necessary measures to ensure public safety.

We would like to extend our project by introducing networking. This will give us the ability to store historical data and alert users if a scent is classified in situations where the robot is unmonitored. By communicating with a local server or introducing IoT components, we can also improve on our alerting system and make the robot remotely operable.

On the research front, through this project we propose an alternate scent-based navigation mechanism that can be particularly useful in low visibility situations. Exploring this field can lead to an additional input modality for safer, more efficient navigation.

11.2 Lessons Learned

For a scent classification system, there are various considerations from the working of scent classification, robot hardware tuning, and the design and integration of components. The sensors we chose are sensitive to various environmental factors like crowded spaces, natural airflow, temperature, and weather dependencies. We spent increased effort toward thresholding and determining the correct transitional states for the robot to go through in order to localize to the scented object correctly. Additionally, the choice of a classification algorithm was limited due to our particular scents. There are hazards that our sensor array cannot recognize, highlighting the tradeoff of the performance and specificity of machine learning applications on a larger scale. With our robot design, tuning wheel speeds and self-correcting from the positional controller required a lot of tuning due to the weight the wheels had to propel, along with the surface the robot had to run on. Lastly, our team decided to set a Slack time for 2 weeks in our scheduling, which we actually utilized to test and integrate our components together. Observing ScentBot required a lot of time and attention to debugging edge cases in its performance. An important lesson learned is hence to set time aside for the integration and testing of different subsystems together.

Glossary of Acronyms

Include an alphabetized list of acronyms if you have lots of these included in your document. Otherwise define the acronyms inline.

- TVOC - Total Volatile Organic Compounds
- VOC - Volatile Organic Compounds
- CO2 - Cardon Dioxide Gas
- CO - Carbon Monoxide Gas
- C2H5OH - Ethanol Gas
- NH3 - Ammonia Gas
- NO2 - Nitrogen Dioxide Gas
- MOX - Metal Oxide
- IR - Infrared

- IoT - Internet of Things
- MCU - Microcontroller Unit
- I2C - Inter-Integrated Circuit Protocol
- SDA - Serial Data Line
- SCL - Serial Clock Line
- SVC - Support Vector Classification
- PWM - Pulse Width Modulation
- UART - Universal Asynchronous Receiver-Transmitter

12 References

1. Hopkins, C., Kelly, C. Prevalence and persistence of smell and taste dysfunction in COVID-19; how should dental practices apply diagnostic criteria?. BDJ In Pract 34, 22–23 (2021). https://doi.org/10.1038/s41404-021-0652-4

2. "Department of Health," The Facts About Ammonia. [Online]. Available: https://www.health.ny.gov/environmental /emergency/chemical_terrorism/ammonia_tech.htm#

3. "Osha infosheet - Occupational Safety and Health Administration." [Online]. Available: https://www.osha.gov/sites/default/files /publications/OSHA3646.pdf.

4. H. Matsukura, "Fluid Dynamic Considerations for Realistic Odor Presentation Using Olfactory Display," in Presence, vol. 19, no. 6, pp. 513-526, 1 Dec. 2010, doi: 10.1162/pre_a_00019.

5. K. R, "Types of gas sensors in the world of IOT," Intuz. [Online]. Available: https://www.intuz.com/blog/types-of-iot-gas-sensor. [Accessed: 03-Mar-2023].

6. "What is a metal oxide (MOX) sensor? - sensirion." [Online]. Available: https://sensirion.com/media/documents/ 0083CDF4/6294DFEA/Info_Note_MOX_sensor.pdf. [Accessed: 04-Mar-2023].

7. "Artificial Nose Technology," Microsoft AI. [Online]. Available: https://www.microsoft.com/en-us/ai/ai-lab-artificial-nose. [Accessed: 03-Mar-2023].

8. "Gas leak inspection robot," SMP Robotics - Autonomous mobile robot, 02-Mar-2023. [Online]. Available: https://smprobotics.com/ products_autonomous_ugv/gas-leak-inspection-robot/. [Accessed: 03-Mar-2023].

Status	System	Part Name	Quantity	Manufacturer	Cost
	Sensor Module				
Recieved		Grove - Gas Sensor V2(Multichannel)	1	Seeed Studio	29.99
Recieved		Adafruit ENS160 MOX Gas Sensor - S	1	Adafruit	21.95
Recieved		Temperature, Humidity, Pressure Sens	1	Adafruit	14.95
Recieved		Ultrasonic Sensor	3	Adafruit	3.95
Recieved		Grove Arduino Shield	1	Seeed Studio	6
Total:					84.74
	Robot				
Recieved		Arduino Mega	1	DigiKey	48
Recieved		Fan	1	Adafruit	13.99
Recieved		External Power Supply	1	Amazon	9.59
Recieved		Motors	2	Adafruit	12.5
Recieved		Wheels	2	Adafruit	1.95
Recieved		Ball caster	1	Digikey	2.5
Recieved		Battery Casing	1	Digikey	0.55
Recieved		L298N Motor Controller	1	Qunqi	6.99
Total:					110.52
	Testing				
Recieved		Cardboard	4	Inventory	0
Recieved		Isopropyl Alcohol		Inventory	0
Not Ordered		Paint Thinner/Varnish		Inventory	0
Total Cost					195.26

Figure 8: Bill of Materials

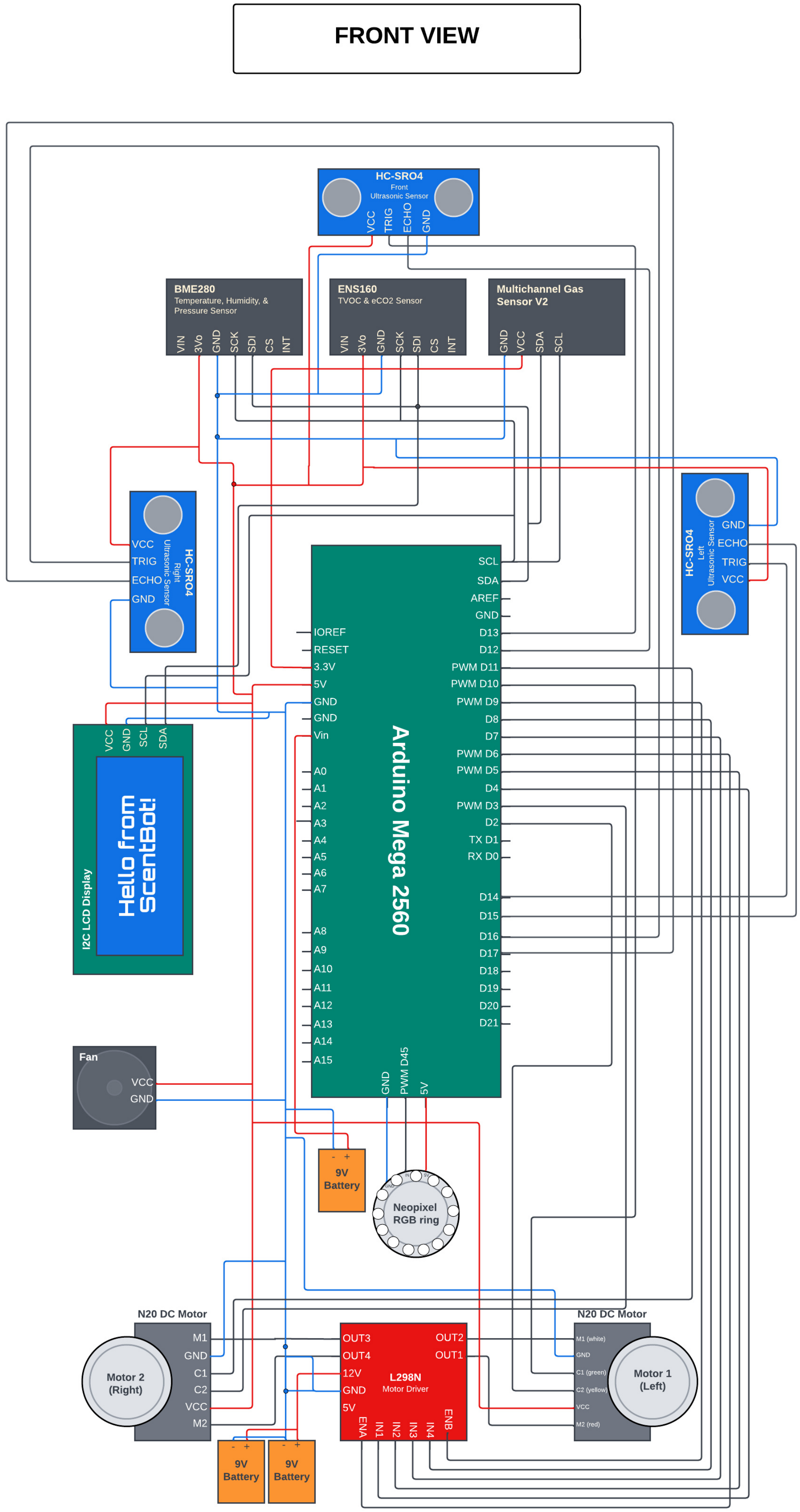

Figure 9: Robot Schematic Diagram

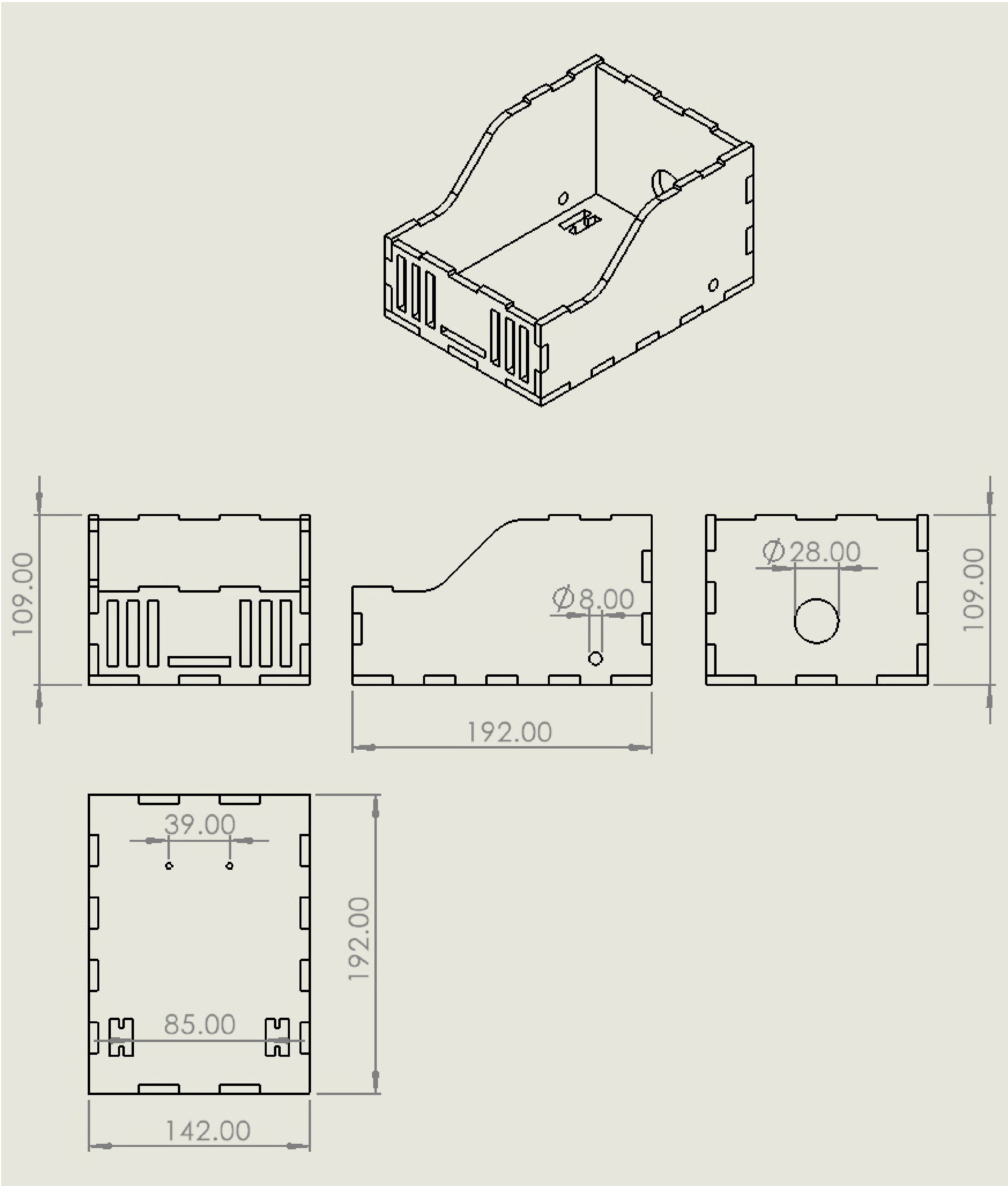

Figure 10: Robot CAD Drawing (mm)

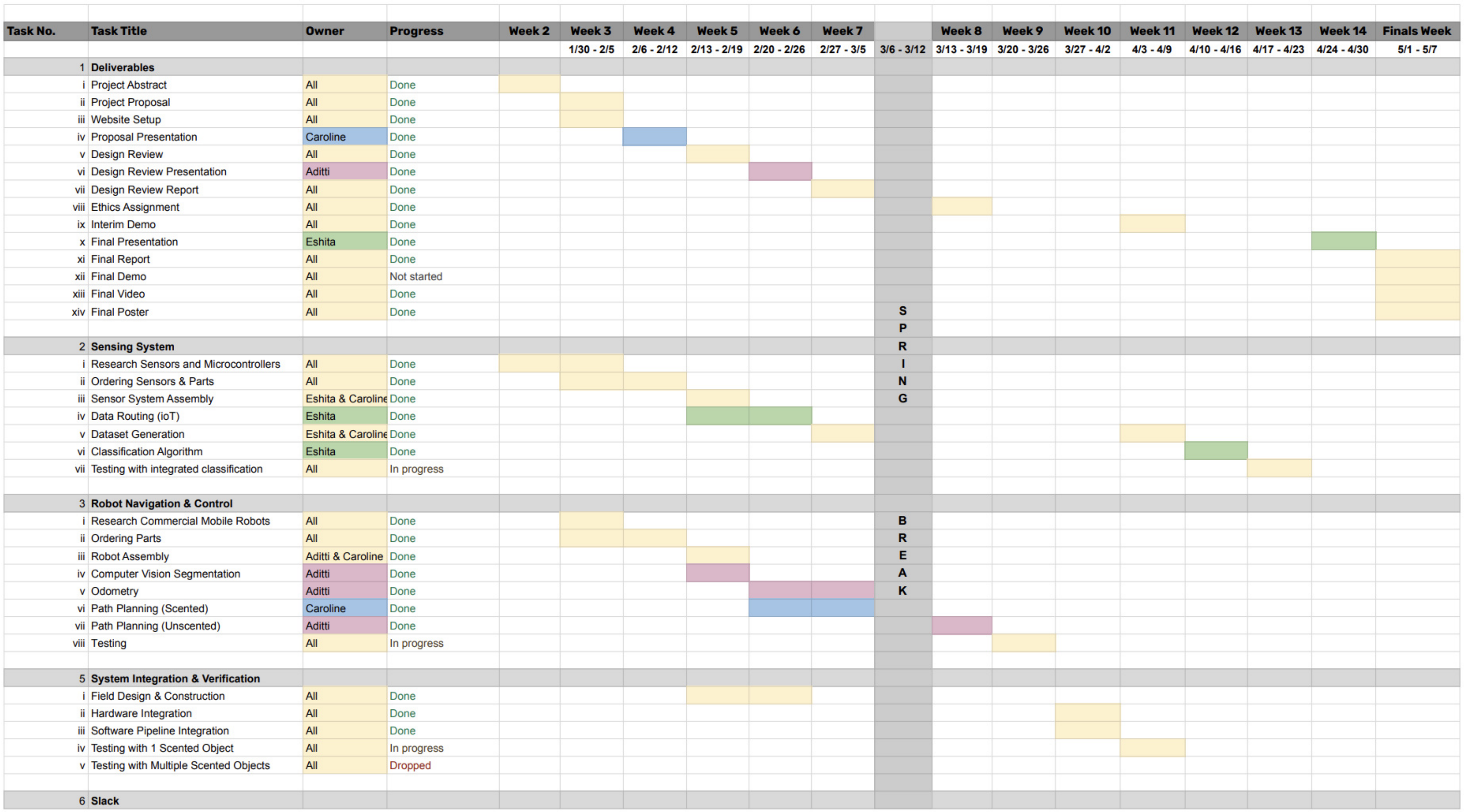

Figure 11: Gantt Chart