

**Estimation of a Plume With a UTV**

A Major Qualifying Project Report

Submitted to the Faculty of the


WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

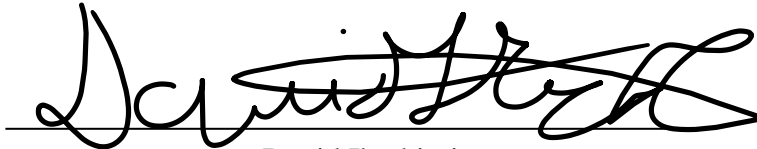
In Aerospace Engineering

by

  
John McCarthy



Vincent Mitala



David Ibrahimi

March 4, 2022

Approved by:

\_\_\_\_\_  
Professor Michael A. Demetriou, Advisor  
Aerospace Engineering Department

WPI

## **Abstract**

This project consisted of researching and designing a gas sensing system onboard of a ClearPath® Robotics Husky. The gas sensing system determines the spatial gradient of the gas so the mobile robot can locate the gas source. The project also consisted of developing upon the work of previous teams to build a gas dispersion setup which was used to test the system. The Carbon Dioxide sensor system that was optimized by a previous team was placed on a 3D-printed stand. The work that this project team does will help advance the field of autonomous vehicles, especially relating to gas sensing.

## **Acknowledgments**

We would like to thank the Aerospace Engineering Department at Worcester Polytechnic Institute and others who assisted us throughout this Major Qualifying Project. This appreciation also goes toward Ms. Tina Stratis who helped our team with project purchases. We also express our gratitude to Professor Demetriou for his advice throughout the entirety for this Major Qualifying Project.

## Table of Authorship

Section	Authorship
<b>Introduction</b>	John McCarthy
1.1	Vincent Mitala
1.1.1	Vincent Mitala
1.1.2	Vincent Mitala
1.1.3	Vincent Mitala
1.1.4	Vincent Mitala
1.1.5	Vincent Mitala
1.1.6	Vincent Mitala
<b>Societal Impact</b>	Vincent Mitala
<b>Robot/Stand/Sensors</b>	David Ibrahimi
3.1	David Ibrahimi
3.2	David Ibrahimi
3.3	David Ibrahimi
<b>Plume Generation</b>	Vincent Mitala
4.1	Vincent Mitala
4.2	Vincent Mitala
4.3	Vincent Mitala
4.4	Vincent Mitala
4.5	Vincent Mitala
4.6	Vincent Mitala

<b>Control</b>	John McCarthy
5.1	John McCarthy
5.2	John McCarthy
5.3	John McCarthy
5.4	John McCarthy
<b>Summary</b>	John McCarthy
<b>Conclusion</b>	John McCarthy
<b>Recommendations</b>	John McCarthy
<b>References</b>	Vincent Mitala, David Ibrahimi, John McCarthy
<b>Appendix</b>	Vincent Mitala, David Ibrahimi, John McCarthy
10.1	John McCarthy
10.2	David Ibrahimi
10.3	Vincent Mitala

# Table of Contents

Acknowledgements .....	ii
Table of Authorship .....	iii
Table of Figures .....	vii
Table of Tables .....	viii
1. Introduction .....	1
1.1. Summaries of Previous Major Qualifying Projects .....	2
1.1.1 Summary of NAG 1501 .....	2
1.1.2 Summary of MAD 1601 .....	2
1.1.3 Summary of MAD 1702 .....	3
1.1.4 Summary of MAD 1802 .....	3
1.1.5 Summary of MAD 1902 .....	4
1.1.6 Overall Project Comparisons and Objectives .....	5
2. Societal Impact .....	6
3. Robot Platform and Sensors .....	8
3.1 Older Robots .....	9
3.2 New Robot .....	10
3.3 Sensor and Base .....	12
4. Plume Generation .....	16
4.1 Plume Generation .....	17
4.2 Regulator .....	18
4.3 Flow Meter .....	18
4.4 Diffuser Use (Point Source) .....	20

4.5 Designs of Diffusers .....	20
4.6 Isentropic Properties .....	22
5. Control .....	23
5.1 Sensor Configurations .....	24
5.2 MATLAB Testing Environment .....	27
5.3 Onboard Computer .....	28
5.4 Kalman Filter Design .....	29
6. Summary .....	30
7. Conclusions .....	31
8. Recommendations .....	33
9.. References .....	34
10. Appendix .....	35
10.1 Appendix A – MATLAB CODING. ....	35
10.2 Appendix B – Sensor Configuration Arduino Coding + Plots. ....	39
10.3 Appendix C – Drawing for Diffuser Design With Measurements. ....	42

## Table of Figures

Figure 1: 2019 MQP (MAD 1902) Robot with Sensors . . . . .	4
Figure 2: Sensor Stand Robot Model from 2018 MQP (MAD 1802) Project . . . . .	4
Figure 3: Khepera IV & iRobot Comparison . . . . .	9
Figure 4: ClearPath HUSKY Unmanned Ground Vehicle . . . . .	10
Figure 5: Figure 3: Powerful CIM Motors . . . . .	11
Figure 6: Sensor Stand Design . . . . .	13
Figure 7: Sensor Structural Stability . . . . .	14
Figure 8: Sensor Stand Displacement and Stress Testing . . . . .	15
Figure 9: Fully-Printed Sensor Stand . . . . .	15
Figure 10: Cozir CO <sub>2</sub> Sensor. . . . .	16
Figure 11: Plume Generation System with CO <sub>2</sub> Tank . . . . .	17
Figure 12: Harris® Model 9296 Regulator that will generate the plume. . . . .	18
Figure 13: Sierra© SmartTrak C100L Mass Flow Rate Meter . . . . .	19
Figure 14: 2018 MQP Diffuser . . . . .	20
Figure 15: 2019 MQP Diffuser (Point Source Sparger) . . . . .	20
Figure 16: One Half of the Diffuser Design and Sphere for Diffusion . . . . .	21
Figure 17: 3D Printed Diffuser . . . . .	21
Figure 18: A design of the four-sensor configuration based on the MA 1902 project . . . . .	24
Figure 19: Shows the Comparison between the Arc Length of a Circle and the Percent Length of a Straight Line at Different Values of Theta. . . . .	25
Figure 20: Shows How the New Sensor Configuration Will Function.. . . . .	26
Figure 21: Raspberry Pi-4 Used For Controls . . . . .	29



Figure 22: A Sample Plot of the Concentration Distribution Created By the Time-Invariant Simulation ..... 31

Figure 23: Measurement Sketch For Diffuser Design ..... 42

**Table of Tables**

Table 1: A, B, C- TERM Timetable ..... 7

Table 2: Budget List ..... 8

# 1. Introduction

Gas is a state of matter that always surrounds us. Gasses such as oxygen and nitrogen are vital to life whereas carbon monoxide and chlorine gas are very toxic gases that people can encounter in their daily lives. Complications from toxic gas exposures can include hallucination, dizziness, and suffocation. In many cases people can tell when a dangerous gas is in the air as the gas either has some color to it or a smell is added to the gas to alert people that there is a cause for concern. However, in many cases, especially that of carbon monoxide, there is no color or smell to the gas which makes it difficult for people to detect it. This lack of detectability makes carbon monoxide dangerous because it replaces oxygen in the lungs, meaning the danger increases over time even for minute exposure. For this reason, the project team developed an autonomous vehicle that can locate the source of a gas plume. The autonomous vehicle ensures that people will not be placed in hazardous situations trying to locate the source of dangerous gases. Due to the danger of carbon monoxide the project team used carbon dioxide is used as a stand-in as it is similar while also being much safer to work with. This project consisted of a point source that was used to generate a plume of carbon dioxide in a concentrated area while a robot fixed with concentration sensors measured the amount of carbon dioxide in the air at three points and used that to determine a direction of travel. The sensors were placed on top of a stand that was attached to the surface of a ClearPath Husky. This plume generation system also shows how mobile robots can be used to navigate hazardous environments in the future by locating dangerous areas for people to avoid.

## **1.1 Summaries of Previous Major Qualifying Projects**

Results from previous major qualifying projects from WPI (Worcester Polytechnic Institute) were used to inform decisions on this project.

### **1.1.1 Summary of NAG 1501**

In Christopher Clark et al, [4] this MQP team presented the idea of creating a concentrated system of plume generation and creating a robot that will detect the gas that served as the basis for several projects including this one. NAG 1501 selected carbon dioxide as the gas to generate the plume which set the precedent for future projects.

### **1.1.2 Summary of MAD 1601**

For Nicholas Christie et al, [3] this MQP team created a sensor mount design which used a four-legged aluminum 6061 Alloy Weld Design for rigidity. An important parameter of the plume estimation is that the gas concentration readings were measured from a set distance above the ground to account for in-flow disturbances from the floor. The group used the Khepera IV robot instead of other considerations due to its small size and similar capabilities from two previous robots. The group started using CO<sub>2</sub> Meter COZIR Gas Sensors. Once the team implemented and defined discrete-time kinematic equations of motion, they described how discretization is a process of estimating a continuous-time system that helps evaluate the state of the robot as an iterative process. The team then extended Kalman Filter for eliminating sensor noise in a dynamic system.

### **1.1.3 Summary of MAD 1702**

In Eric Fast et al, [5] this MQP team used the Computer for Autonomous System Execution (CASE) that's utilized for all gas information to be received. The gas information was then turned into a map of the concentration gradient and then instructed the robot wireless how to move to the next area of interest. A Universal Serial Bus (USB) camera was used to analyze the validity of an extended Kalman Filter. The Kalman Filter was used for filtering out Gaussian noise of a system by using the sensors and fusing them together. The 2017 MQP group then decided to use an extended Kalman Filter for state estimation. An array of four CO<sub>2</sub> COZIR sensors were used and arranged as in a cross shape. The stands were positioned to avoid floor bias.

### **1.1.4 Summary of MAD 1802**

In Sylvester Halama et al, [7] this MQP team decided to use the Dagu Wild Thumper 4WD All-Terrain Chasis because it is sturdy and possesses stability. This team created a new sensor stand to counteract oscillations from previous designs. The stand was made out of acrylonitrile butadiene styrene (ABS plastic) for reduced weight while also having strength to support the sensors. It also contributed to a lower center of mass. There were errors of previous COZIR sensors. The settling time length prevented an efficient experiment. The 2018 group found this would only make the robot continuously turn or change direction instead of focusing on the CO<sub>2</sub> source. This team instead decided to use SprintIR WR 20% CO<sub>2</sub> sensors because it had a higher capability to sense carbon dioxide.

## 1.1.5 Summary of MAD 1902

For Theresa Bender et al, [1] this MQP team decided to model their robot based on the iRobot Create and Khepera IV robots to simplify the equations of motion. By simplifying the equations, there were fewer assumptions to make about the robot's motion. The robot was created with plywood and the large base was utilized to remove the possibility to tipping with a four-legged stand attached on top. For specific dimensions, the group used a laser cutter. The robot used a Raspberry Pi 3B dual motor controller because of its wireless communication. Python 2.7 programmed Raspberry Pi because of its relatively easy language. The 2019 MQP also used the SprintIR WR 20% CO<sub>2</sub> sensors. For its plume generation system, it used CO<sub>2</sub> tank pressurized to approximately 700 psi. After a dual state regulator decreased the pressure to 100 psi, the CO<sub>2</sub> passed through a flow meter to control the mass flow rate. For robot movement used the CO<sub>2</sub> sensors and the Python script sorting all the CO<sub>2</sub> sensor data.



Figure 1: 2019 MQP (MAD 1902) Robot with Sensors.



Figure 2: Sensor Stand Robot Model from 2018 MQP (MAD 1802) Project.

## 1.1.6 Overall Project Comparisons and Objectives

The 2015 MQP creates the same type of conceptual plume generation system. Like the current and previous reports, this was to introduce and lay the groundwork for new types of experiments to detect all types of gases in a room in the future. A difference between our project and the 2016 MQP project on plume generation used a Khepera IV robot with four gas concentration sensors and a wind velocity sensor for plume detection.

The sensors needed to be placed at a distance so that the eddies in the flow would not cause notable errors in the sensors. This was a problem that was addressed by previous MQP teams by placing the sensors at distances greater than twice the diameter of the eddies which are measured by the integral scale. So far, the distance that the sensors need to be spaced is at least 10 cm apart. The larger size of the Husky allows for a much larger weight limit than in past years which means that the computations can be done onboard the husky as opposed to from a base station as they were in the past. Further, the height of the tower needs to be notably less than in previous years because the tower now only needs to exceed the boundary layer formed by the Husky's motion, and since the Husky will be stationary while the measurements are settling the boundary layer will be small.

The team met three times a week to discuss what to do and to attempt to troubleshoot problems with the robot. The team also met with Professor Demetriou weekly to update him on our progress as well as ask any questions we had relating to the project. In terms of the breakdown of the work, one person focused on the fluid dynamics aspect, such as the diffuser; another focused on the structural dynamics aspect, such as the tower and sensors; and one person focused on the programming aspect, especially the controls.

The objectives of this project were to have a HUSKY UGV ClearPath robot with attached sensors be able to detect carbon dioxide and locate the source of the plume. The team will design a sensor stand to attach the sensors to the Husky. We will design a point source to diffuse carbon dioxide from a CO<sub>2</sub> canister tank. For the diffuser, the objective is to design a diffuser and test that it is safely dispersing CO<sub>2</sub> at levels, rates, and at a distance where there is no hazard to humans. Another objective is to make sure the HUSKY can drive to the point source without any human help. To do that the information from the sensors will be transferred to a computer which will determine inputs to get the Husky to the point source.

In terms of the broader impact of the project, detection of a plume of carbon dioxide or other gas by a vehicle or robot-like structure will allow people to detect gas leaks that they would not necessarily find. Future possibilities include having vehicles that can detect different gases in areas where there is a concern about said gas leaking. This can be useful, especially in places that deal and work with many gases and pollutants.

## **2. Societal Impact**

Our project can impact society by presenting a new way for people to either avoid or locate sources of hazardous gases. As mentioned before, the goal of this project has been to program the ClearPath HUSKY vehicle to drive towards a generated carbon dioxide plume. Many gases that people can interact with daily can cause health issues. As noted earlier the potential health effects of carbon monoxide inhalation include drowsiness lethargy suffocation and eventually death. Due to the nature of carbon monoxide, people should avoid it as much as possible. This autonomous vehicle can be easily modified to avoid areas of high concentrations of specific gases which can be very helpful for people evacuating during disasters. Carbon monoxide displaces oxygen in the lungs resulting in less oxygen in the blood which can be very

harmful to humans. Less available oxygen can also cause permanent organ damage to the heart and brain. Meaning that the ability to avoid hazardous gases such as carbon monoxide can make a major difference to the health and well-being of people who are trying to traverse hazardous areas

	week 1	week 2	week 3	week 4	week 5	week 6	week 7	week 8	week 9	week 10	week 11	week 12	week 13	week 14	week 15	week 16	week 17	week 18	week 19	week 20	week 21
background research	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█					
design the diffuser	█	█	█	█	█	█															
familiarize ourselves with robot operation	█	█	█																		
asses the sensors	█	█	█																		
simulate plume expansion			█	█	█	█	█														
design the sensor stand				█	█	█	█														
create a space to test the control algorithm							█														
create CAD models for the diffuser and sensor stand						█	█	█	█	█	█	█	█								
research the mass flow meter								█	█	█	█	█	█	█	█	█					
develop a control algorithm in MATLAB							█	█	█	█	█	█	█	█	█	█					
develop the Kalman filter to reduce errors											█	█	█	█	█	█	█				
3D print the sensor stand and diffuser												█	█	█	█	█	█	█	█	█	█
connect to the robot																	█	█	█	█	█
converted the MATLAB code to python																		█			
Assemble the diffuser and sensor stand																		█	█		
modify python code due to unexpected errors																		█	█	█	
install necicary programs to the Raspberri Pi																					█
perform tests to see how well the Husky performs																					█

Table 1: A, B, C-TERM Timetable.



## Budget List

ITEM	COST	LINK
USB to Wi-Fi Adapter	\$23.00	<a href="https://www.amazon.com/Wireless-USB-WiFi-Adapter-PC/dp/B07V4R3QHW">https://www.amazon.com/Wireless-USB-WiFi-Adapter-PC/dp/B07V4R3QHW</a>
Gorilla Dual Temp Mini Hot Glue Gun Kit with 30 Hot Glue Sticks, (Pack of 1)	\$8.49	<a href="https://www.amazon.com/Gorilla-8401509-Hot-Glue-Sticks/dp/B07K791YRP?th=1">https://www.amazon.com/Gorilla-8401509-Hot-Glue-Sticks/dp/B07K791YRP?th=1</a>
Raspberry Pi	\$109.95	<a href="https://www.canakit.com/raspberry-pi-4-starter-kit.html">https://www.canakit.com/raspberry-pi-4-starter-kit.html</a>
USB 3.0 Extension Cable - A-Male to A-Female Adapter Cord- 9.8 Feet (3.0 Meters) (2X)	\$24.63	<a href="https://www.amazon.com/AmazonBasics-Extension-Cable-Male-Female/dp/B00NH12O5I?th=1">https://www.amazon.com/AmazonBasics-Extension-Cable-Male-Female/dp/B00NH12O5I?th=1</a>
UGREEN USB to USB Cable, USB 3.0 Male to Male Type A to Type A Cable for Data Transfer Compatible with Hard Drive, Laptop, DVD Player, TV, USB 3.0 Hub, Monitor, Camera, Set Up Box and More 1.5FT	\$20.85	<a href="https://www.amazon.com/UGREEN-Transfer-Enclosures-Printers-Cameras/dp/B00P0E3954/ref=sr_1_1_sspa?keywords=Male%2Bto%2BMale%2BUSB%2BCable&amp;qid=1642784715&amp;sr=8-1-spons&amp;spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyMFdlSVkzSVAYNTZBjMvuy3J5cHRIZEIkPUewMDMzNTM5MTMzQ1BPV01RS1M4WCZLbmNyeXB0ZWRBZEIkPUewMzYxNDc4MTdJRjIGNzE0OVBOXCZ3aWRnZXROTW1IPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU&amp;th=1">https://www.amazon.com/UGREEN-Transfer-Enclosures-Printers-Cameras/dp/B00P0E3954/ref=sr_1_1_sspa?keywords=Male%2Bto%2BMale%2BUSB%2BCable&amp;qid=1642784715&amp;sr=8-1-spons&amp;spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyMFdlSVkzSVAYNTZBjMvuy3J5cHRIZEIkPUewMDMzNTM5MTMzQ1BPV01RS1M4WCZLbmNyeXB0ZWRBZEIkPUewMzYxNDc4MTdJRjIGNzE0OVBOXCZ3aWRnZXROTW1IPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU&amp;th=1</a>

Table 2: Budget List.

### 3. Robot Platform and Sensors

The Husky's objective is to locate the highest concentration of gas in the room. This was achieved using three sensors that read carbon dioxide concentration. Previous MQPs used robots such as Khepera, iRobot, or an in-house built robot. All of the previous robots were roughly one foot in diameter and a few inches tall, for comparison the Husky stands three feet tall and has a

two and a half by two and a half foot platform on top of it. The Husky solved many of the issues of previous teams such as loading limits and stability considerations however the Husky presented its own challenges due to its large size and the difficulty interfacing with it.

### 3.1 Older Robots

The Khepera IV which was used by the MAD 1601 team was selected for its small size, processor power, and extension options. Khepera IV is a very flexible robot featuring two differential drive wheels. The biggest benefit to this robot was the eight infrared sensors evenly spaced around the robot and five ultrasonic sensors facing toward the front which were used for measuring distances. This meant that the MAD 1601 team had a built-in system to gather information on the robot's position. The tiny base contributed to a very unstable ground unit which is why future groups moved away from the Khepera IV

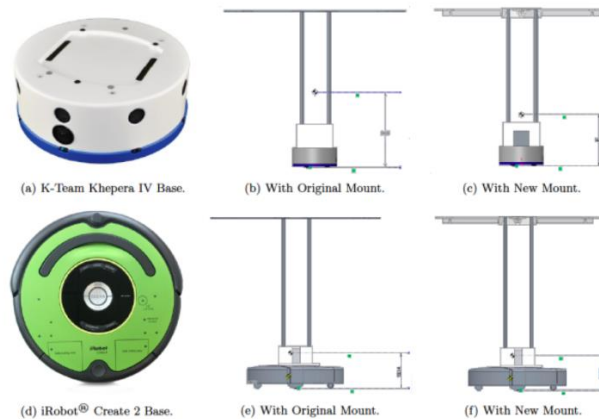


Figure 3: Khepera IV & iRobot Comparison.

The following MQP decided to switch their robot to iRobot Create 2 (MAD 1702). The iRobot is wider and heavier than Khepera IV; above, shown in Figure 1 you can see the comparison between the center of gravity of these robots which was developed by the MAD

1702 team. The base is much lower, which contributes to a more stable robot. The only downgrade to the iRobot was that there are no sensors regarding distance and measuring its surroundings. Although its stability was better than Khepera IV, the weight of the sensors and the stand managed to sway the robot's natural motion.

The MAD 1802 and MAD 1902 teams both built robots from scratch using the information from previous teams as a basis for their design. The teams primarily attempted to address the persistent stability issue to increase success. The MAD 1802 and MAD 1902 teams made use of an Arduino and a Raspberry Pi respectively to read the inputs from the sensors while using a base station to perform calculations. The motivation for this was primarily limitations in weight and computing power. The size of the Husky eliminates these concerns allowing the team to develop a robot that has no stability concerns and makes use of an onboard computer for calculations.

### **3.2 New Robot**

The team was supplied a HUSKY UGV ClearPath robot to use for this project. It was designed to be modifiable, with an internal bay to place sensitive components, a variety of accessible power leads, and a top plate covered in M5 screw holes designed for mounting custom parts.



Figure 4: ClearPath HUSKY Unmanned Ground Vehicle.

This plate on top of the Husky as well as the large size simplified the sensor stand design. The Husky is an off-the-shelf, 110lb robotic drive base intended to be used in projects that require navigation of outdoor or indoor terrain. The Husky contains two CIM motors shown in Figure 5.



Figure 5: Powerful CIM Motors.

The HUSKY takes velocity or position commands through a serial connection to a Linux PC. Although it is designed to traverse outdoor terrain, the HUSKY has only a couple of inches of ground clearance, and with only about 160 watts of power, it has trouble with steep slopes. This is not a concern because the tests will occur on an indoor flat surface. It can travel at a maximum speed of 2 meters per second which is much faster than the robots other teams have used in the past.

### 3.3 Sensor and Base

For the CO<sub>2</sub> sensors to properly function, the previous years' teams designed cross-like sensor stands, which were attached to the robot. The sensor stand was designed to be at a certain height to keep the sensors away from the ground. This plays an essential role in the sensor readings, as sensors can get false readings due to plume recirculation close to the ground or floor. The sensor stand included a set of three CO<sub>2</sub> COZIR™ sensors, which stood at the end of each arm. These sensors were very light, so the PLA material used to 3D print the arms did not have any trouble carrying the weight.

There was a significant change in this year's sensor stand; it only has three arms, meaning this experiment was performed with three sensors, unlike the previous years' teams who used four sensors. The arms are equally distant from one another so that the readings are independent without putting a computational strain on the onboard computer. As shown in Figure 4 below, the sensor stand was designed to have a simple structure and to not be as tall as the previous years' MQP teams due to the height of the Husky.

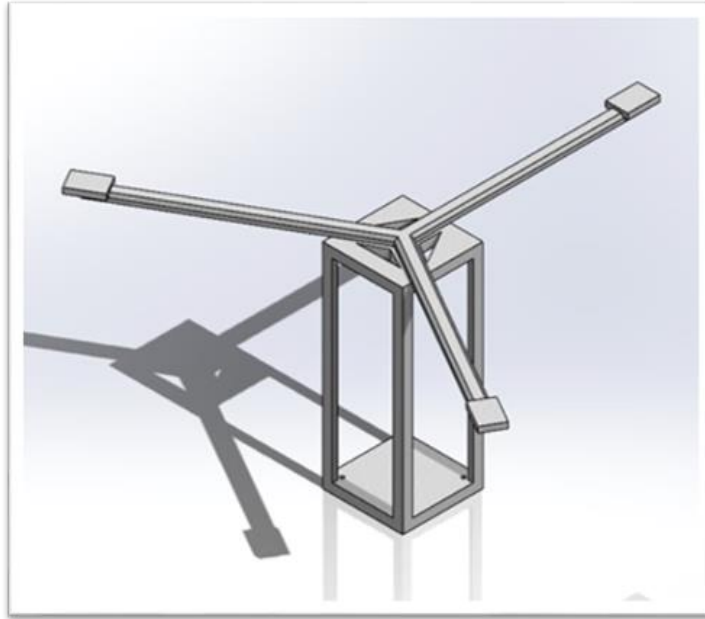


Figure 6: Sensor Stand Design.

The PLA material used to 3D print the stand is very light, totaling the weight of the stand at 0.565 kg, which the HUSKY robot can easily handle. At the bottom of the stand, there are four holes for the team to screw the stand to the top platform of the HUSKY robot, attaching them as one piece. Since there were three arms and three sensors performing this experiment, the distance of each arm from the center was 14 inches. This was enough distance for the sensors to get readings that would not be affected by turbulent airflow.

Before the stand was 3D printed, the team ran a set of tests on the stand to test its stability and strength using the SOLIDWORKS program. As shown in Figure 5 below, the first test was done when the HUSKY robot went from 2 m/s to an instant stop, the diagram below showed the team the stand's structural stability and displacement. When arrived at a stop, the top half of the stand(color change dark to light blue) was displaced 0.0013 mm, and the displacement increased slightly the higher you moved. For example, the arm leading the robot did not experience any

significant displacement (0.065mm), while the two arms in the back experienced a displacement of 0.156 mm as the angle of experiencing this momentum was 60 degrees.

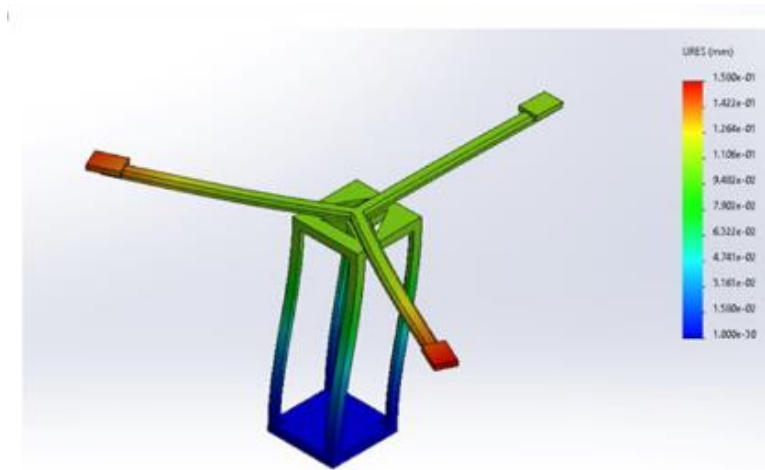


Figure 7: Sensor Structural Stability.

Overall, the displacement created by the robot when it arrived at a complete stop is within the predicted margin. The following test focused on the stand's arms, particularly where the COZIR sensors will be placed. The team wanted to see the effects of the weight of the sensors (2.5 grams/sensor) on the arms. Although the weight of the COZIR sensors was minimal, the impact on each arm would also be small to negligible ( $\sim 0.025$  N/arm). Although, as shown in Figure 6 below, the amount of force per arm was more than 0.025 Newtons during the tests, this was because the team wanted to test the arms limits before they were 3D printed.

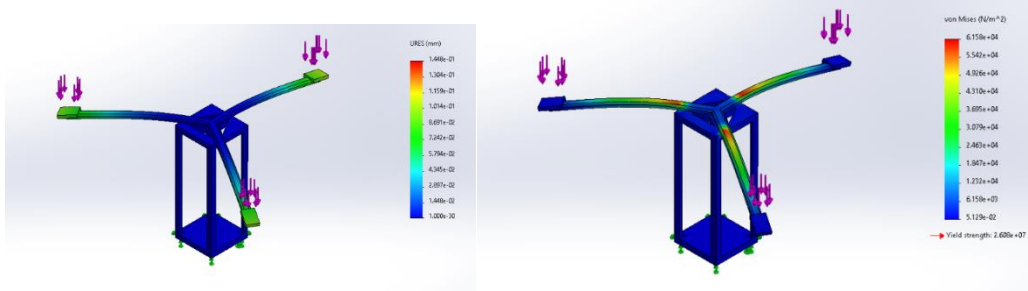


Figure 8: Sensor Stand Displacement and Stress Testing.

As mentioned above, the arms were tested at a force of 0.1 N/arm. Moreover, the results of this test give an equal displacement of 0.072 mm, which was the result the team and the stand itself can accept. In Figure 6 to the right side is the stress analysis of the sensor stand. During the constant loading of the arms, there is a stress increase at each of the arms. Solidworks records the highest performed stress on the arm during the test, reaching 6,150 N/m<sup>2</sup> per arm. Each of the arms can handle these readings.

Although the arms were 3D printed using PLA material, they also had to be split into three arms, and all are identical. This was due to limitations on the size of the 3D printers. The team 3D printed each of the arms one by one, then later glued them together at an angle of 120 degrees.



Figure 9: Fully-Printed Sensor Stand.



The team decided to use the COZIR CO<sub>2</sub> sensor because upon research it was the most reliable and simple to set up. It is a low power, high performance CO<sub>2</sub> sensor that can measure up to 100% CO<sub>2</sub> levels. It is suitable for measuring high concentrations of CO<sub>2</sub> in closed-loop sampling applications or battery operation in portable sampling instruments. Below is appendix 10.2, which shows the Arduino code used to gather sensor readings.



Figure 10: COZIR CO<sub>2</sub> Sensor.

## 4. Plume Generation

The team made use of an existing plume generation stand which contained all the necessary parts to create and monitor a carbon dioxide plume. The team improved upon the stand by developing a diffuser that could sit on the stand and function as a point source. Similarly, to the sensor stand the diffuser was 3D printed to ensure it met specifications.

## 4.1 Plume Generation

The plume generation system comprises a pressurized carbon dioxide canister and is controlled via LabView to produce plumes of known CO<sub>2</sub> profiles in an indoor environment. The plume generation can be fixed in space representing a stationary source, or onboard a terrain vehicle to represent a moving source. The tube from the tank connects to a Harris® Model 9296 Regulator, a flow meter, and into a diffuser.



Figure 11: Plume Generation System with CO<sub>2</sub> Tank.

## 4.2 Regulator

There is a two-stage regulator called the Harris® Model 9296 that has the purpose of decreasing the high pressure from the carbon dioxide tank to a smaller constant pressure. The dual-stage regulator is connected to the CO<sub>2</sub> tank that is pressurized at 700 pounds-per-square inches (psi). During testing the regulator was adjusted to 100 psi to ensure a safe diffusion system. Figure #8 shows the regulator.



Figure 12: Harris® Model 9296 Regulator that will generate the plume.

## 4.3 Flow Meter

The third part of the plume generation is the Sierra©SmartTrakC100L which set the CO<sub>2</sub> mass flow rate. The device displays accurate flow rate readings by monitoring the temperature variation as it moves through a hot wire which is also connected to a PID-controlled valve. This

reference information was useful when designing the diffuser. Previous MQPs experimented and determined that 500 mg/s would be the most useful mass flow. However, this specific tool or brand only measures up to 459 mg/s. For testing and safety purposes, we set the mass flow rate to about 450 mg/s. This flow rate of carbon dioxide did not result in unsafe conditions as the average level of carbon dioxide concentration was around 400-1000 parts per minute (ppm) or (mg/L) in concentrated areas, where the diffusion system will be tested. This value for the concentration is less than the range of dangerous CO<sub>2</sub> levels (1000-2000 ppm) where there are symptoms of drowsiness and poor air quality. The area immediately around the point source does contain dangerous levels of CO<sub>2</sub> to avoid these dangerous levels during testing the team remained at least 5 to 6 feet distance away from the point source of the carbon dioxide diffusion preferring to be outside the room during testing.



Figure 13: Sierra© SmartTrak C100L Mass Flow Rate Meter.

## 4.4 Diffuser Use (Point Source)

The purpose of the diffuser for this experiment was to evenly diffuse the carbon dioxide across the specific concentrated area by acting as a point source. This prevents the carbon dioxide from ejecting in a single direction at a high velocity which would throw off the sensors. Previous MQP groups designed diffusers with a box-shaped figure with a sphere with spiked holes for the CO<sub>2</sub> to pass through. Another design consisted of a sparger connected to a 4-inch piece of metal tubing. A benefit of the sparger design was that it is smaller and is easier to manipulate. 2-3 designs were created to attempt to improve on the sparger.

### PREVIOUS DESIGNS



Figure 14: 2018 MQP Diffuser.



Figure 15: 2019 MQP Diffuser  
(Point Source Sparger).

## 4.5 Design of Diffuser

The diffuser had three potential designs which were all compared mathematically and the best of the three was attached to the 2019 MQP's sparger. The dimensions were taken of the metallic piece base of the sparger. It has a hexagonal shape, with each side length being 0.45 inches. The final diffuser design was created in CAD (Computer-Aided Design) as two identical pieces which were attached together by a hot glue gun. The full length is 6.20 inches, and the

height is 6.20 inches. The whole width is 2.5 inches. Like the 2018 MQP diffuser, the design has a sphere with holes where the carbon dioxide will diffuse evenly. The sphere will have a 0.60-inch diameter opening for the CO<sub>2</sub> to enter with much smaller holes for CO<sub>2</sub> to diffuse out of.

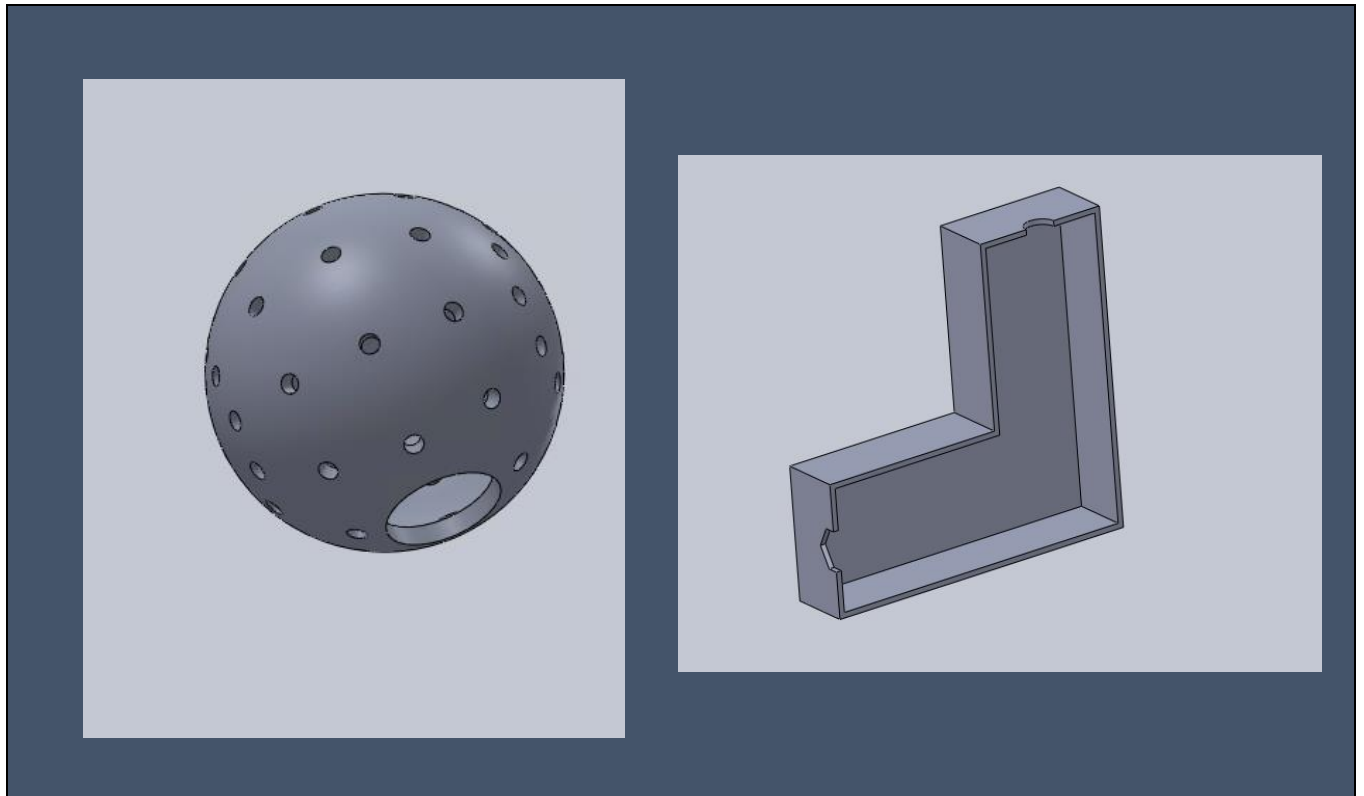


Figure 16: One Half of the Diffuser Design and Sphere for Diffusion.

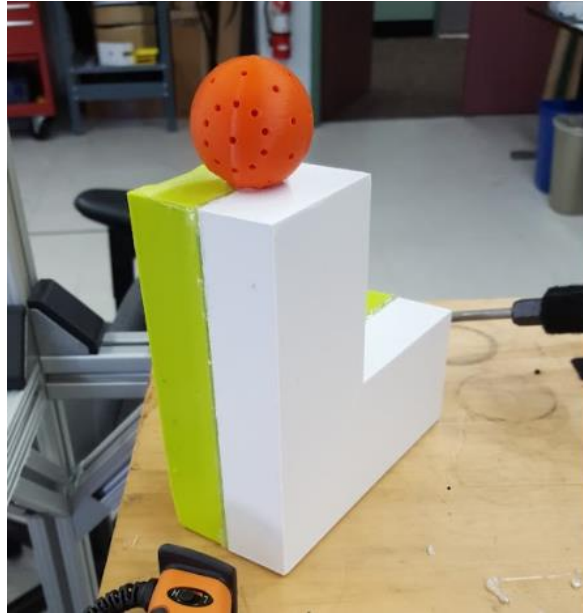


Figure 17: 3D Printed Diffuser.

## 4.6 Isentropic Properties

It is important to keep the Mach number as low as possible to get the maximum exit pressure and to ensure that the flow is turbulent when it reaches the end of the diffuser. To find the desired area for the opening where the carbon dioxide will immediately enter, an intermediate area of the choked condition  $A^*$  was used to relate the area of the flow through the sparger to the area at the desired Mach number. Then the two ratios are multiplied to determine the desired flow area of the diffuser. The subsonic Mach number the team selected was 0.4. The specific heat ratio of carbon dioxide is 1.28. Once the area ratio was calculated, the Reynolds number of the flow was calculated to ensure that it would be turbulent under those conditions.

$$\gamma = 1.28$$

$$A^* = 4 \text{ in}$$

$$A = \text{Desired Area}$$

$$M = 0.4$$

$$\text{Atmospheric Temperature (T)} = 273.15 \text{ K}$$

$$\text{Universal Gas Constant (R)} = 8314 \text{ J/kmol-K}$$

$$\text{Molecular Mass (MM) of Carbon Dioxide} = 44.01 \text{ g/mol}$$

$$R_{\text{specific}} = \frac{\bar{R}}{MM} = \frac{8314}{44.01} = 188.911 \quad (1)$$

$$a = \sqrt{\gamma RT} = \sqrt{(1.28)(188.9116)(273.15)} = 257.0010591 \frac{m}{s} \quad (2)$$

$$\frac{A}{A^*} = \left(\frac{\gamma+1}{2}\right)^{\frac{-\gamma+1}{2(\gamma-1)}} \frac{\left(1 + \frac{\gamma-1}{2} M^2\right)^{\frac{\gamma+1}{2(\gamma-1)}}}{M} \quad (3)$$

$$A = \left(\frac{1.28+1}{2}\right)^{\frac{-1.28+1}{2(1.28-1)}} \frac{\left(1 + \frac{1.28-1}{2} (0.4)^2\right)^{\frac{1.28+1}{2(1.28-1)}}}{0.4} * 4$$

$$\mathbf{A = 6.419 \text{ in}^2}$$

## 5. Control

The control algorithm utilized by the team differed primarily due to the three-sensor configuration. While previous teams used four sensors to keep the calculations simple the team opted instead to use slightly more complex guidance equations which made use of only three sensors.



## 5.1 Sensor Configurations

The Husky determines which way to move by measuring the local gas concentration at each of the gas sensors and then using the law of cosines to determine the gradient direction. To more efficiently use the sensors that we had the team decided to use three sensors so that the fourth sensor can be placed near the gas source and read as a reference value during testing. The configuration uses the three-sensor configuration shown in figure 4 and a control scheme which was developed based on information from the MAD 1902 project. The three-sensor configuration uses two front and a rear sensor placed one hundred twenty degrees apart, the program which models the three-sensor configuration. The primary difference between the two is that the three-sensor configuration always uses the readings from all its sensors as opposed to three out of the four which are used by the four-sensor configuration.

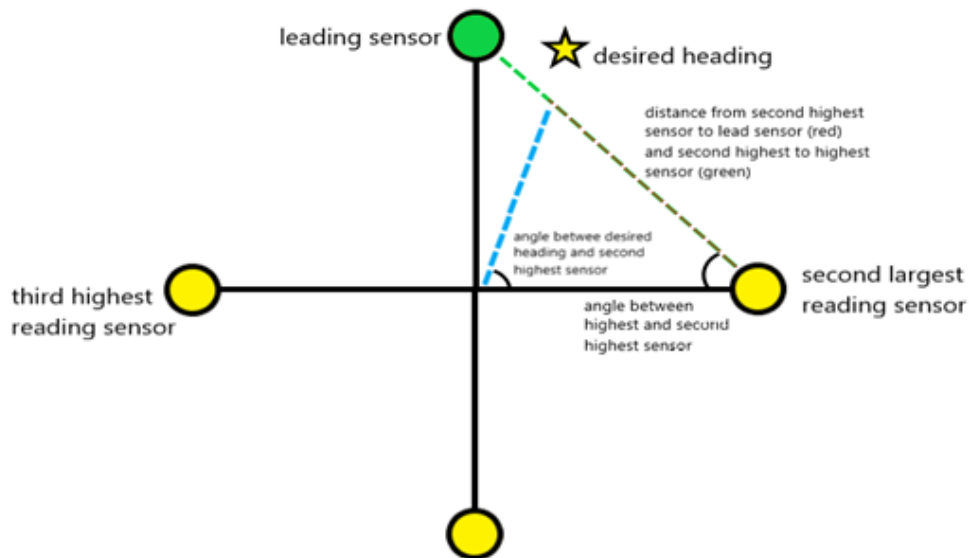


Figure 18: A design of the four-sensor configuration based on the MAD 1902 project.

The four-sensor configuration works by defining the sensor with the largest reading as a “leading sensor.” The second-largest reading is used to calculate the angle that the robot needs to turn to face in the direction of the greatest increase. The third highest sensor is used as a reference sensor and is subtracted from the highest and second-highest sensor to account for the potentially large sensor readings. This means that the fourth sensor is not used to determine the direction of travel and thus only three sensors are needed.

The three-sensor configuration works by considering the positive x-axis in the body-fixed frame as the forward direction, which is the same in the four-sensor model but there is no sensor along this axis. The three sensors are then given fixed relations which are used to determine the desired heading. To determine the desired heading the sensors which are on the same side as the front of the robot are compared, the robot will then turn in the direction of the larger sensor. The amount of the turn is then determined by subtracting the reading from sensor three from sensor one and sensor two. If the results are negative for one, then the resultant angle is then subtracted from one hundred and twenty degrees. If both the results from sensor one and sensor two are negative, then the sign of the desired angle is switched and added to one hundred and eighty degrees.

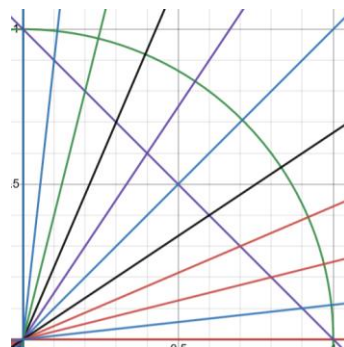


Figure 19: Shows the Comparison between the Arc Length of a Circle and the Percent Length of a Straight Line at Different Values of Theta.

Since the sensor configuration that we are using now has a larger angle between the sensors it is important to realize that the change in the length of the arcs is larger than the change in the length of the line as the angles approach forty-five degrees. As such we need to adjust the equations so that we get the length along the arc and this arc length is then converted into an angle as opposed to utilizing the law of cosines to determine an angle based on the distance along the straight line. This would normally be ignored due to the small angle assumption. However, the initial turn of the Husky cannot be assumed to be small due to computational limitations preventing small discrete time intervals.

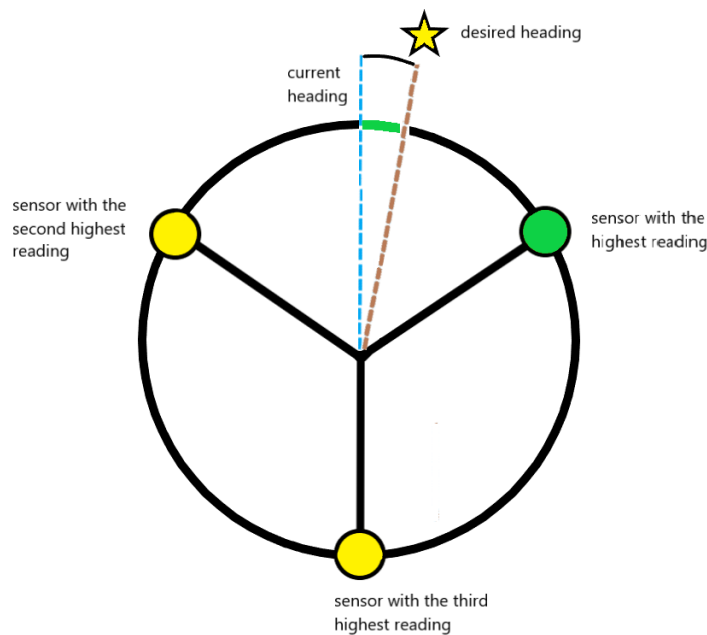


Figure 20: Shows How the New Sensor Configuration Will Function.

This figure shows how the new turn angle will be determined. This method is both more accurate and can be done using fewer calculations which works well with the onboard computing configuration.

$$Ratio = \frac{R2 - R3}{R1 - R3 + R2 - R3} \quad (4)$$

The above ratio describes the relationship between the sensor readings with R1 corresponding to the reading for sensor one, R2 corresponding to the reading for sensor two, and R3 corresponding to the reading for sensor three. The Sign correction term noted in equation (5) is used to differentiate if the robot is going to turn clockwise or counterclockwise. Arc length divided by the total circumference can just be expressed as a radian value from zero to two times pi over three as that is the radial value of the distance between the sensors. The sector correction term is a term that uses the location of the sensors relative to the heading angle to correct the change in the arclength while only using two readings. The sensor angle correction term accounts for the difference between the heading angle and the positions of the sensors because there is no sensor directly in line with the direction of travel.

$$\frac{Change\ in\ Desired\ Heading\ Angle}{Heading\ Angle} = \frac{Sign\ Correction * Ratio\ of\ Readings * Arc\ Length}{Total\ Circumference} + Sector\ Correction + Sensor\ Angle\ Correction \quad (5)$$

## 5.2 MATLAB Testing Environment

To compare the three and four-sensor configurations to ensure that there was no notable difference between the two the plume expansion simulation code was combined with the controls code. This allowed the team to assess how the two different configurations would perform in several different situations. The two versions of the testing environment were created a time-varying concentration case and a time-invariant concentration case. The two versions were used

for two distinct aspects, the time-invariant case was used for the troubleshooting of the code because the consistent nature of the concentration distribution made it easier to check that the simulated robot was doing what it was supposed to be doing. The time-variant case was used to test the accuracy and to compare the two configurations. The goal was to determine which of the two configurations would be more desirable. The two codes also function as a fine and a coarse way to improve the guidance algorithm, as it is easier to see the general effect of a change in the time-invariant case while the time-variant case can be used to see how much if at all the change improves the time it takes for the robot to locate the source of the gas.

### **5.3 Onboard Computer**

Due to the size of our robot, we decided to attach a computer directly onto the robot to perform directional calculations. To perform the calculations for this project we decided to use a Raspberry Pi-4. This was done because a new model Raspberry Pi-4 or RPi-4 has similar capabilities to the computer in the lab so even if we used the lab computer as a base station, we would not have any difference in computing power (Raspberry pi website). The onboard computer provides a few benefits, the most significant of which is the reduction in the time it takes for the computer to communicate with the Husky. Additionally, the changing MAC address of the RPi-4 is not an issue when it is directly connected to the Husky. Further due to the Husky being able to provide power to the RPi-4 the computer will always power on at the same time the Husky does, meaning that a person will not need to interact with the computer directly to execute the experiments.



Figure 21: Raspberry Pi-4 Used for Controls.

## 5.4 Kalman Filter Design

The Kalman filter functions by taking a set of uncertain information bounded by some equations and it includes reference data as well as the level of uncertainty of that reference compared to the uncertainty of state. This information is then used to get a more accurate measure of the state. The use of the Kalman filter, in this case, is to continually compare the calculated state to the observed state to ensure that the error in the calculated state remains bounded.

The team decided that due to time constraints it was beyond the scope of the project to implement the Kalman filter. The largest reason for this was the difficulty utilizing the camera in the lab. This would only work some of the time, this faultiness made it difficult to test the camera

and to try to implement the position reference measurements which are required for the Kalman Filter.

This bounding of the error is not significant for the purposes of the Husky reaching the location of the highest concentration of the gas as that only depends on the readings of the sensors. However, the Husky also needs to determine the location once it reaches it. This means that the Husky needs to keep the error bound. The reference information that would have been used to bound the error would have been obtained using a camera mounted to the ceiling of the lab and April Tags software.

## **6. Summary**

The team set out to utilize a ClearPath Robotics Husky to locate the source of a plume of carbon dioxide. The team designed and 3D printed a new sensor stand to allow for effective gas concentration measurement. The team wrote MATLAB code to model the expansion of the gas. This was useful both for the visualization of the diffusion of the gas as well as the creation of an algorithm that found the source of the plume. The team created a new diffuser which allowed for a more accurate simulation of a point source, which improved the effectiveness of the CO<sub>2</sub> dispersion testing stand. The team developed MATLAB code to model the motions of the robot with three and with four sensors. The comparison of the three and four sensor configurations highlighted the need for only three sensors in the searching algorithm.

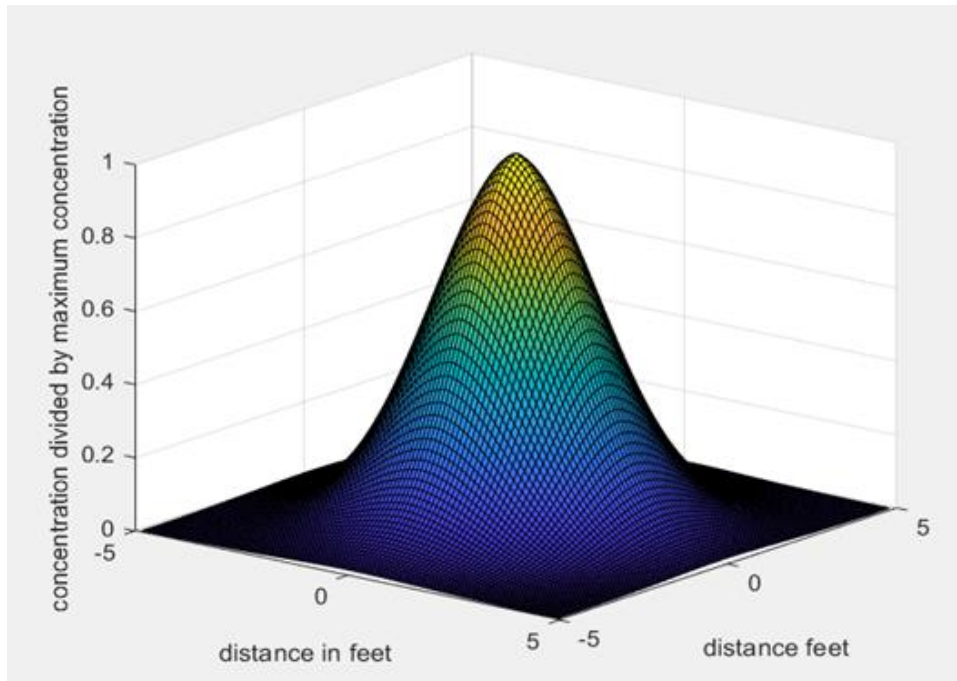


Figure 22: A Sample Plot of the Concentration Distribution Created by the Time-Invariant Simulation.

Additionally, the team acquired a microprocessor in the form of an RPi-4 which was used to perform the onboard directional calculations for the Husky. The system was able to properly sense the gas concentration gradient in the room and reached the source of the plume successfully. The location data that was produced by the husky had very small errors due to only making four measurements of the gas concentration before reaching the source.

## 7. Conclusions

As expected, the team faced numerous challenges upon taking on this project. Additionally, there was a lot of information that the group needed to review to continue where previous groups had left off. This was beneficial because although it initially took some time to



research and to decipher past work, it served as a starting point to build from which made the generation of current ideas much more streamlined.

The team learned about the importance of beginning steps such as 3D printing a few weeks prior to when they are needed as the process took twice as long as we initially planned which delayed our project. The team also familiarized itself with Linux and the utilization of ROS (Robot Operating System) software. This was done through a textbook that explained basic Linux operations and tutorials on how to utilize ROS software.

Further, the team adapted to changing conditions as one of the sensors seemed to be faulty at first which forced a redesign of the sensor stand that would allow it to operate just as well with three sensors as it had with four. The team developed a more complicated python code to take the sensor readings into the RPi-4 and to not only scale the values so that they measured the same amount in similar situations, but also to perform the calculations for the robot motion as soon as possible with the sensor information. The speed of the calculations was important because the sensors had a long settling time which significantly limited the speed at which the robot could accurately travel towards the point source.

Another challenge faced by the team was the difficulty connecting the Raspberry Pi to the Husky as the Raspberry Pi needed to have certain software specifically for the husky which the team did not realize until nearly the end of the project. Once the subparts were finished, there was no issue putting it together. The project is now accomplished as the team built a robot that can locate a gas source based on the readings given by three gas sensors, which directs the robot towards the gas source.

## 8. Recommendations

This section will include all recommendations for any future teams who are working on similar projects. Future teams hoping to take this project further could develop a sensor stand that has variable arm lengths that would allow the sensors to check to see if they are at the point source or if the system is just on a level surface of gas concentration. Additionally, a future team could try to utilize small drones to sense gas concentrations though that would be difficult as the drones themselves disturb the airflow significantly.

An additional recommendation that the team determined would be useful is to create a system so that the testing stand can be operated remotely. This will allow for more safety testing if the team wants to move to potentially more hazardous gases. This could be done by making use of a second Raspberry Pi. Another recommendation is a future team could build out a sensing system to determine the robot's location so that a Kalman filter can be developed without the need for April Tags.

## 9. References

1. Bender, T., Burris, K., Herrington, S., & Weber, D. (2020, January 16). *Detection and Estimation of a Plume with an Unmanned Terrain Vehicle*. File | MAD\_1902\_Plume\_Detection.pdf | ID: 4b29b775h | Digital WPI. Retrieved February 26, 2022, from <https://digital.wpi.edu/show/4b29b775h>
2. *Carbon dioxide*. Wisconsin Department of Health Services. (2021, June 3). Retrieved February 26, 2022, from <https://www.dhs.wisconsin.gov/chemical/carbondioxide.htm#:~:text=The%20levels%20of%20CO2%20in,of%20drowsiness%20and%20poor%20air.>
3. Christie, N., and Colfer, J. (2016, March 24). *Plume Estimation Using a Gas-Sensing Mobile Robot*. Digital WPI. Retrieved February 26, 2022, from <https://digital.wpi.edu/pdfviewer/g158bj979>
4. Clark, C., Greene, M., and Seigle, M. (2015, March 23). *Design of Plume Generation and Detection Systems*. Digital WPI. Retrieved February 26, 2022, from <https://digital.wpi.edu/pdfviewer/dn39x317q>
5. Fast, E., Harnais, S., and Wiesenber, R. (2017, March 21). *Plume Analysis and Detection*. Digital WPI. Retrieved February 26, 2022, from <https://digital.wpi.edu/pdfviewer/wm117q38v>
6. Government of Canada, C. C. for O. H. and S. (2022, February 25). *Carbon dioxide : Osh answers*. Canadian Centre for Occupational Health and Safety. Retrieved February 26, 2022, from [https://www.ccohs.ca/oshanswers/chemicals/chem\\_profiles/carbon\\_dioxide.html](https://www.ccohs.ca/oshanswers/chemicals/chem_profiles/carbon_dioxide.html)
7. Halama, S., Kasapis, S., Kontopygros, M., McGrath, O., & Preston, B. (2018, March 20). *Estimation of a Plume with an Unmanned Terrestrial Vehicle*. Digital WPI. Retrieved February 26, 2022, from <https://digital.wpi.edu/pdfviewer/vm40xt28t>
8. NASA. (n.d.). *Isentropic flow equations*. NASA. Retrieved February 26, 2022, from <https://www.grc.nasa.gov/www/k-12/airplane/isentrop.html>
9. Raspberry Pi. (n.d.). *Teach, learn, and make with Raspberry Pi*. Raspberry Pi. Retrieved February 26, 2022, from <https://www.raspberrypi.org/>

## 10. Appendix

### 10.1 Appendix A – MATLAB CODING

This function created variables that are used to determine which of the sensors acts as the lead sensor, which is the sensor that will be used to determine the angle correction, and which sensor will be used as the reference sensor.

```
function [lead,correction,ref,sign] = sensor_logic(sensor_1,sensor_2,sensor_3,sensor_4)
sorted=sortrows([sensor_1,sensor_2,sensor_3,sensor_4]');
if sorted(3)==sorted(4)
    correction=0;
end
if sorted(2)==sorted(3)
    ref=0;
end
for i =1:4
    if sorted(i)==sensor_1
        s_1=i;
    end
    if sorted(i)==sensor_2
        s_2=i;
    end
    if sorted(i)==sensor_3
        s_3=i;
    end
    if sorted(i)==sensor_4
        s_4=i;
    end
end
sensors=[s_1,s_2,s_3,s_4];
for j=1:4
    if sensors(j)==4
        lead=j;
    end
    if sensors(j)==3
        correction=j;
    end
    if sensors(j)==2
        ref=j;
    end
end
if ((lead>ref)&&(lead>1.5)) || (lead-ref==2)
    sign=1;
else
    sign=-1;
end
```

```
end
end
```

This function uses the above function to determine the desired heading angle based on the measurements for the sensors.

```
function [theta_desired] = motion_3(sensor_1,sensor_2,sensor_3,theta)
%distance between is characterised by the difference between the sensors
sensors=[sensor_1,sensor_2,sensor_3];
[lead,correction,ref,sign]=sensor_logic(sensor_1,sensor_2,sensor_3,0);
%speed max u_l=1
%speed max u_r=1
if correction==0
    theta_desired=theta;
else
    if ref==0
        theta_desired=-theta;
    else
        reading_1=sensors(lead);
        reading_2=sensors(correction);
        refrence=sensors(ref);
        r_2=reading_2-refrence;
        r_1=reading_1-refrence;
        l_1=0.155;
        l_2=0.155;
        H=(l_1^2+l_2^2-2*l_1*l_2*cos(2*pi/3))^(1/2);
        a=(H*r_1)/(r_2+r_1);
        B=pi/6;
        c=l_2;
        b=(a^2+c^2-a*c*cos(B))^(1/2);
        A=acos((-a^2+b^2+c^2)/(2*b*c));
        theta_desired=theta+sign*(A+(2*pi/3*(correction+lead-3))-pi/3);
    end
end
while theta_desired>=(2*3.1415926535)
    theta_desired=theta_desired-(2*3.1415926535);
end
while theta_desired<=(0)
    theta_desired=theta_desired+(2*3.1415926535);
end
end
end
```

This script ran a simulation of the three-sensor configuration using a non-time varying set of concentrations.

```
clc
clear
dt=1;%time step
time=0;
time_2=0;
timefinal=120;%final time in seconds
H=10;%total length
dy=1.1;%linear differetiation element
Uo=20000;%source concentration
%create plot of concentration over time
delta_x=rand*4.75;
delta_y=rand*4.75;
sigma_x=1.6667;
sigma_y=sigma_x;
mew_x=0;
mew_y=0;
rho=0;%correlation
room_concentration_static= two_d_steady_plot(sigma_x,sigma_y,mew_x,mew_y,rho);
```

```

%% motion of the 3 sensor robot
stop=0;
x=[delta_x,delta_y,0];%[x,y,theta]
x_record=zeros(3,timefinal/dt);
t_record=zeros(1,timefinal/dt);
x_record(:,1)=x;
sensor_record_x=zeros(3,timefinal/dt);
sensor_record_y=zeros(3,timefinal/dt);
sensor_readings_record=zeros(3,timefinal/dt);
u=[0,0];
column_number=0;
while stop<1&&time<timefinal
sensor_1=room_concentration_static(round((x(1)+.155*cos(x(3)+pi/3))/dy+50),round((x(2)+.155*sin(x(3)+pi/3))/dy+50));%concentration at x+.155*cos(theta-pi/3),y+.155*cos(theta-pi)
sensor_2=room_concentration_static(round((x(1)+.155*cos(x(3)-pi/3))/dy+50),round((x(2)+.155*sin(x(3)-pi/3))/dy+50));%concentration at x+.155*cos(theta+pi/3),y+.155*cos(theta+pi)
sensor_3=room_concentration_static(round((x(1)+.155*cos(x(3)-pi))/dy+50),round((x(2)+.155*sin(x(3)-pi))/dy+50));%concentration at x-.155*cos(theta-pi),y+.155*cos(theta-pi)
sensor_positions_x=[x(1)+.155*cos(x(3)+pi/3),x(1)+.155*cos(x(3)-pi/3),x(1)+.155*cos(x(3)-pi)];
sensor_positions_y=[x(2)+.155*sin(x(3)+pi/3),x(2)+.155*sin(x(3)-pi/3),x(2)+.155*sin(x(3)-pi)];
sensor_readings=[sensor_1,sensor_2,sensor_3];
[theta_desired]=motion_3(sensor_1,sensor_2,sensor_3,x(3));
w_max=2/.555;
t_turn_min=abs(theta_desired-x(3))/w_max;
w=(theta_desired-x(3))/2;
if w==0
t_turn=0;
else
t_turn=2;
end
t=0;
if (sensor_1||sensor_2||sensor_3)>0&&(sensor_1||sensor_2||sensor_3)<.0001
t_final=32+2;
else
if (sensor_1||sensor_2||sensor_3)>.0001&&(sensor_1||sensor_2||sensor_3)<.001
t_final=32/2+2;
else
if (sensor_1||sensor_2||sensor_3)>.001&&(sensor_1||sensor_2||sensor_3)<.01
t_final=16/2+2;
else

```

```

t_final=16/8+2;
end
end
end
while t<t_turn&&t<t_final
u_l=-w*.555/2;
u_r=w*.555/2;
u=[u_l,u_r];
k1 = dt*dx_robot(x,u);
k2 = dt*dx_robot(x + (1/2)*k1,u);
k3 = dt*dx_robot(x + (1/2)*k2,u);
k4 = dt*dx_robot(x + k3,u);
x_plus_dt= x + (1/6)*k1 + (1/3)*k2 + (1/3)*k3 + (1/6)*k4;

t=t+dt;
x=x_plus_dt;
time=time+dt;
column_number = column_number + 1;
t_record(:,column_number)=t;
x_record(:,column_number)=x;
sensor_readings_record(:,column_number)=sensor_readings;
sensor_record_x(:,column_number)=sensor_positions_x;
sensor_record_y(:,column_number)=sensor_positions_y;
sensor_positions_x=[x(1)+.155*cos(x(3)+pi/3),x(1)+.155*cos(x(3)-pi/3),x(1)+.155*cos(x(3)-pi)];
sensor_positions_y=[x(2)+.155*sin(x(3)+pi/3),x(2)+.155*sin(x(3)-pi/3),x(2)+.155*sin(x(3)-pi)];
end
while t>=t_turn&&t<=t_final
u_l=.1;
u_r=.1;
u=[u_l,u_r];
k1 = dt*dx_robot(x,u);
k2 = dt*dx_robot(x + (1/2)*k1,u);
k3 = dt*dx_robot(x + (1/2)*k2,u);
k4 = dt*dx_robot(x + k3,u);
x_plus_dt= x + (1/6)*k1 + (1/3)*k2 + (1/3)*k3 + (1/6)*k4;

t=t+dt;

```

```

x=x_plus_dt;
time=time+dt;
column_number = column_number + 1;
t_record(:,column_number)=t;
x_record(:,column_number)=x;
sensor_readings_record(:,column_number)=sensor_readings;
sensor_record_x(:,column_number)=sensor_positions_x;
sensor_record_y(:,column_number)=sensor_positions_y;
sensor_positions_x=[x(1)+.155*cos(x(3)+pi/3),x(1)+.155*cos(x(3)-pi/3),x(1)+.155*cos(x(3)-pi)];
sensor_positions_y=[x(2)+.155*sin(x(3)+pi/3),x(2)+.155*sin(x(3)-pi/3),x(2)+.155*sin(x(3)-pi)];
end
if abs(x(1))<=.1&&abs(x(2))<=.1
stop=1;
end
end
%% plot

x_range=-4.9:dy:4.9;
y_range=-4.9:dy:4.9;
figure(1)
surf(x_range,y_range,room_concentration_static)
view(2)
hold on
pause
four_sensors=plot3(0,0,0,'g.','0,0,0','g.','0,0,0','g.','0,0,0','g.'):
three_sensors=plot3(0,0,0,'r.','0,0,0','r.','0,0,0','r.'):
arms_4=plot3(0,0,0,'g.','0,0,0','g.','0,0,0','g.','0,0,0','g.'):
arms_3=plot3(0,0,0,'r.','0,0,0','r.','0,0,0','r.'):
for k=1:length(x_record)
figure(1)
hold on
plot3(x_record(1,k),x_record(2,k),1.01,'R.'):
delete(three_sensors);
three_sensors=plot3(sensor_record_x(1,k),sensor_record_y(1,k),1.1,'R.',sensor_record_x(2,k),sensor_record_y(2,k),1.01,'R.',sensor_record_x(3,k),sensor_record_y(3,k),1.01,'R.'):
pause(.1)
xlim([-5,5]);
ylim([-5,5]);

```

## 10.2 Appendix B – Sensor Configuration Arduino Coding + Plots

```

CalibratedCode
#include <SoftwareSerial.h>
#include "cozir.h"

|
#define co2_1_rx 3
#define co2_1_tx 2

#define co2_2_rx 7
#define co2_2_tx 5

#define co2_3_rx 13
#define co2_3_tx 12

SoftwareSerial co2_1 = SoftwareSerial(co2_1_rx, co2_1_tx);
SoftwareSerial co2_2 = SoftwareSerial(co2_2_rx, co2_2_tx);
SoftwareSerial co2_3 = SoftwareSerial(co2_3_rx, co2_3_tx);

const float Slope1 = 1.4214; //(TRUE_015) //0.8182; //(TRUE_738) //1; //(TRUE_954) //1.3792; //(TRUE_015) //6.5051; //(TRUE_B10) //C
const float Offset1 = 178.75; // -40.959; //0; //175.44; //326.53;

const float Slope2 = 1.000; //(TRUE_015) //0.589; //(TRUE_738) //0.7039; //(TRUE_954) //1.000; //(TRUE_015) //4.6557; //(TRUE_B10) //Co
const float Offset2 = 0; // -145.17; // -127.23; //0; // -637.21;

const float Slope3 = 1.6662; //(TRUE_015) //1; //(TRUE_738) //1.2218; //(TRUE_954) //1.6662; //(TRUE_015) //7.8904; //(TRUE_B10) //C
const float Offset3 = 245.78; //0; //90.382; //253.78; //477.46;

void setup() {
Serial.begin(9600);
delay(4000);
//czr.SetOperatingMode(CZR_POLLING);
//czr.SetOperatingMode(CZR_STREAMING);
//czr.CalibrateFreshAir();
// czr.SetDigiFilter(64);
}

```

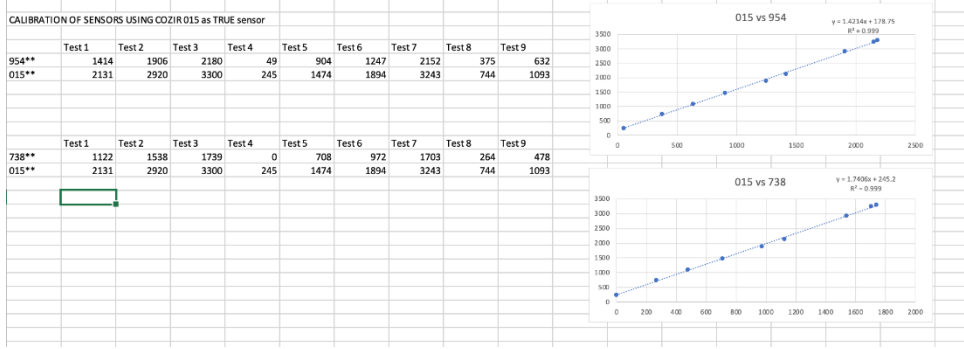
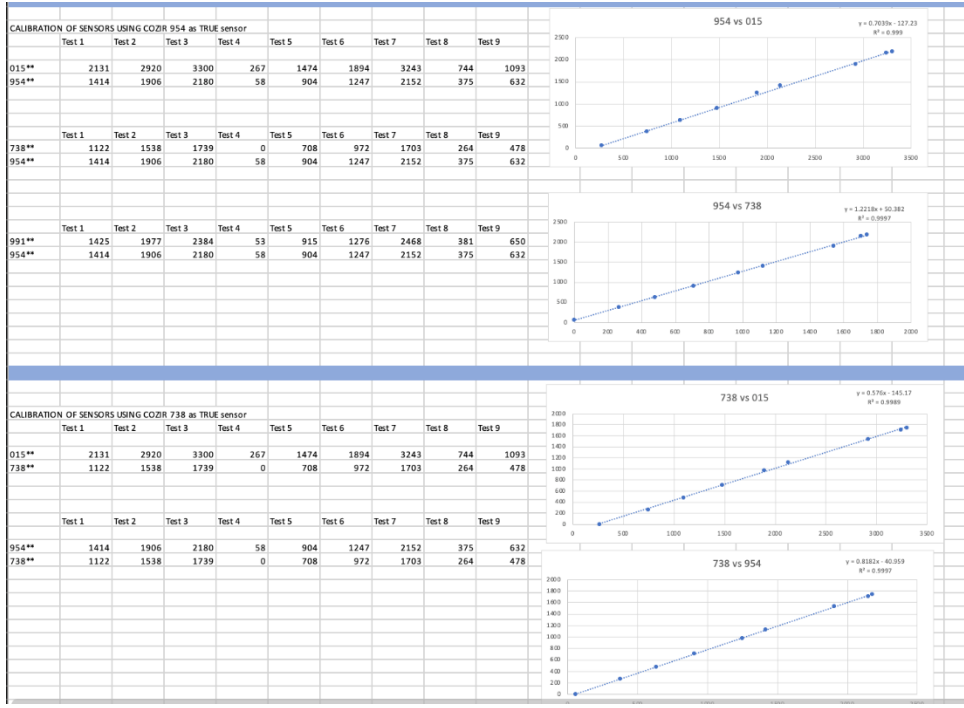
We used three sensors that were calibrated with a fourth sensor from the BIO labs. It was the better and most accurate reading sensor (TRUE\_BIO). The calibration took the form of  $y = mx + b$  and used multiple data points of the readings to get the best fit line equation.

Realized it was best to use one of the three COZIR sensors as our TRUE sensor and calibrate



around that sensor. Pictures are shown below. Some adjustments were made to slope and y-intercept.

```
CalibratedCode §  
}  
  
void loop(){  
  
  COZIR czr_1(co2_1);  
  co2_1.listen();  
  float c1 = czr_1.CO2();  
  
  COZIR czr_2(co2_2);  
  co2_2.listen();  
  float c2 = czr_2.CO2();  
  
  COZIR czr_3(co2_3);  
  co2_3.listen();  
  float c3 = czr_3.CO2();  
  
  delay(4000);  
  float CO2_cal1 = (c1 * Slope1 + Offset1) ; // The calibration curve here.  
  float CO2_cal2 = (c2 * Slope2 + Offset2) ;  
  float CO2_cal3 = (c3 * Slope3 + Offset3) ;  
  |  
  Serial.print("[1-954]CO2 : ");  
  Serial.print(c1);  
  Serial.print(" [1] CO2 calibrated : ");  
  Serial.println(CO2_cal1);  
  
  Serial.print("[2-015]CO2 : ");  
  Serial.print(c2);  
  Serial.print(" [2] CO2 calibrated : ");  
  Serial.println(CO2_cal2);  
  
  Serial.print("[3-738]CO2 : ");  
  Serial.print(c3);  
  Serial.print(" [3] CO2 calibrated : ");  
  Serial.println(CO2_cal3);  
  
  Serial.println( ' ');  
}
```



# 10.3 Appendix C – Drawing for Diffuser Design With Measurements

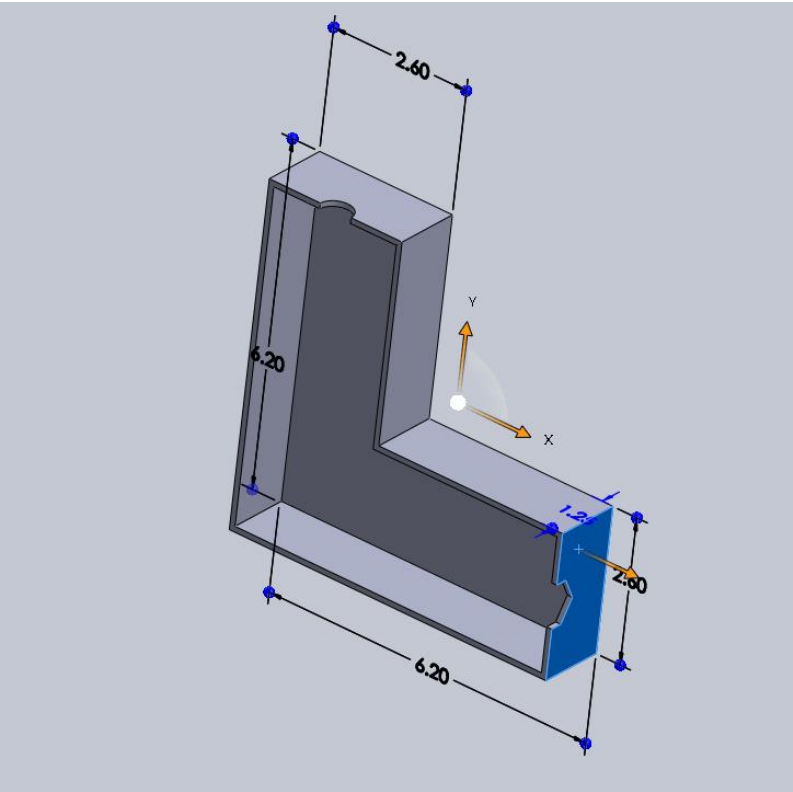


Figure 23: Measurement sketch for diffuser design.