Proceedings of the 2024 Virginia Tech
Mechanical Engineering Senior Capstone Exposition
VTMECAP-2024
April 26, 2024, Blacksburg, Virginia, USA

VTMECAP-2024-Team 44

FUNCTIONAL ROBOT FOR ID AND RETRIEVAL

Nathan Carpenter[1], Mohamed Eldirdiri[1], Wang Guen Lee[1], Joshua Menezes[1], Taylor Pippen[1], Sam Wood[1],
Kaveh Akbari Hamed[3]

[1]Department of Mechanical Engineering, Virginia Tech, Blacksburg, VA
[3]Department of Mechanical Engineering, Virginia Tech, Blacksburg, VA

ABSTRACT

Orchestrated autonomy is a concept that facilitates the collaboration of interconnected machines, enhancing their collective capabilities by allowing them to perform as if they have trained together extensively. This interoperability allows for effective robotic coordination that would be impossible without advanced knowledge of other localized actions [1]. This report focuses on the design and development process of an independent robotic system aimed at assessing toxic gas concentrations in high-risk areas. The system demonstrates orchestrated autonomy by operating autonomously while leaving room for the integration of other robots, such as a mapping robot, to achieve common objectives. It employs multi-terrain locomotion to navigate diverse surfaces presenting a reliable solution for environmental gauging.

Keywords: Autonomy, Orchestrated Autonomy, Robotics, Path Planning, Chassis, Control Systems, Carbon Monoxide, Arduino, Python, A*

NOMENCLATURE

CO	Carbon Monoxide
Kp	proportional gain
Kd	derivative gain
PWM	Pulse Width Modulation
UDP	User Datagram Protocol
A*	A Star

1. INTRODUCTION

The project is being undertaken by Team 44 from Virginia Tech under the guidance of Noblis to advance the concept of orchestrated autonomy through the development of an autonomous robot designed for toxic gas detection in hazardous environments. This initiative addresses critical inefficiencies and dangers associated with human operated toxic gas detection and hazardous zones, which tend to be slow, risky, as well as constrained by human limitations and environmental conditions.

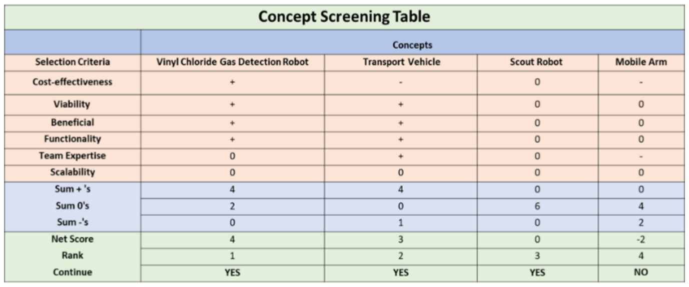
Previous literature highlights the risk to human operators [2] and inefficiencies of manual toxic gas detection [3], which advocates for technological solutions that enhance safety and data collection accuracy. The project's objectives are linked to the robot’s capabilities in path planning, obstacle detection, real-time information exchange, and carbon monoxide detection. Each concept is extremely crucial for orchestrated autonomy. Path planning and obstacle detection are binary in nature, as the robot either successfully navigates and recognizes obstacles or it does not, impacting its ability to operate autonomously in complex environments. Information exchange is also binary, involving the successful or unsuccessful real-time transmission and receipt of data necessary for coordinated operation among multiple robots. Lastly, carbon monoxide detection is binary, either having or lacking the ability to detect changes in carbon monoxide. A full list of requirements can be found in the appendix. These capabilities prove integral to orchestrated autonomy allowing for the collective functionality of robotic systems in dynamic environments.

2. MATERIALS AND METHODS

2.1 CONCEPT SELECTION

After establishing the project requirements and objectives it was necessary to develop a design concept. The process for concept generation was as follows: team members individually created rough sketches of potential concepts and selected their top three concepts to present to the rest of the team. Some of the concepts proposed included a robot transport vehicle, an autonomous scout robot, and a mobile arm. The Pugh concept selection method, a five-step process, was utilized to evaluate and eliminate concepts until three finalists were determined for optimal selection. The first step is to identify the selection criteria. The team picked cost effectiveness, viability, beneficial, functional, team expertise, and scalability as the selection criteria. The selection criteria were chosen based on the needs of Noblis and ease of evaluation. The second step is to enter plausible solutions. All concepts created by team members were considered totaling fourteen different concepts. Each concept was entered into Table 1 shown below. The third step in the screening process is to choose one plausible solution as the benchmark solution. The fourth step is to rate each plausible solution relative to the benchmark solution. If a solution outperforms the benchmark in any of the criteria, it will receive a "+," for a neutral performance, it will be rated as "0," and for a performance worse than the benchmark, it will receive a "-." After each solution has been judged on every criterion, its net score will be calculated, and the solution will be assigned a rank based on its standing. All solutions with a negative score will be eliminated, and all solutions with a positive score will stay for the next round of screening. The fifth step is to repeat steps three and four until there are three solutions remaining. Before the concept screening, there were fourteen concepts: afterwards, only three remained. The finalists include the Vinyl Chloride gas detection robot, the transport vehicle, and the scout robot.

TABLE 1: THIS CHART SHOWS THE RESULTS OF THE FINAL ROUND OF CONCEPT SCREENING



Concept Screening Table				
	Concepts			
Selection Criteria	Vinyl Chloride Gas Detection Robot	Transport Vehicle	Scout Robot	Mobile Arm
Cost-effectiveness	+	-	0	-
Viability	+	+	0	0
Beneficial	+	+	0	0
Functionality	+	+	0	0
Team Expertise	0	+	0	-
Scalability	0	0	0	0
Sum + 's	4	4	0	0
Sum 0's	2	0	6	4
Sum -'s	0	1	0	2
Net Score	4	3	0	-2
Rank	1	2	3	4
Continue	YES	YES	YES	NO

After narrowing down our solutions to three concepts, a dedicated power point was created for each of the three finalists. During the next meeting with Noblis, we presented our final concepts. Noblis provided us with feedback for each concept and discussed ways they could be implemented into their existing fleet. Ultimately, Noblis did not want to influence the concept selection process. And instead opted to let the team make their own decision regarding which concept to select. The Vinyl Chloride detection robot was chosen as the project concept. The concept does an excellent job meeting the project requirements outlined by Noblis. However, one major change was made to the concept to allow the team to test the robot safely. The robot will detect Carbon Monoxide instead of Vinyl Chloride. This is because sensors can pick up carbon monoxide gas long before it becomes deadly for humans [5]. This will allow the team to test their robot without compromising safety.

2.2 CHASSIS DEVELOPMENT & ABS COMPONENTS

The robot chassis is primarily built from 6061 T6 Aluminum plate and square tube. Aluminum was selected for the frame material due to its material strength and ability to be easily machined [6]. The robot has a height of twelve inches, a length of twenty inches, and a width of twelve inches. The larger size was selected to contain all of the components necessary to function, as well as fulfill the ground clearance requirement. To develop the chassis all the parts were modeled and assembled in SolidWorks 2023 as shown in Fig. 1 This allowed the team to easily experiment with different designs and make quick changes when needed. Once the design was settled on the team developed a comprehensive packet of shop drawings to be used to create parts in the applied lab.

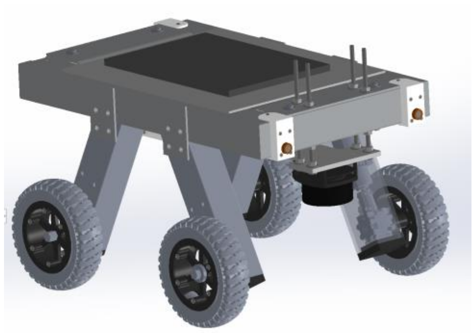


FIGURE 1: COMPLETE CAD MODEL CREATED IN SOLIDWORKS 2023

To create all the metal parts for the robot the team utilized Virginia Tech's Applied Lab. The first step in machining the chassis parts was to use a band saw to cut a twelve-foot piece of square aluminum tube into eight pieces that would make up the upper frame and legs. The second step was to use a CNC machine to add features to the square tube parts such as holes and angled cuts. The third step was to cut out the sheet metal parts from a 1/8" aluminum plate using a plasma cutter. These parts included the upper frame support plate and eight support brackets. The fourth step was to bend the support plate into shape using a press break. The completed set of aluminum parts can be seen in Fig. 2.

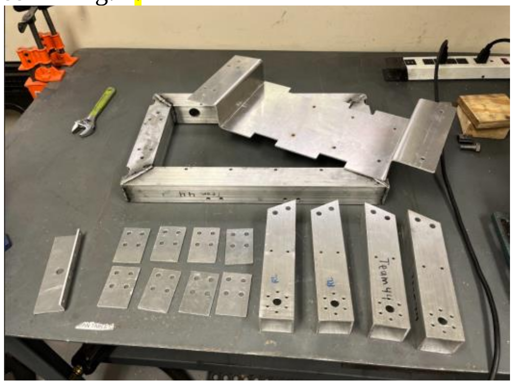


FIGURE 2: COMPLETE SET OF ALUMINUM PARTS CREATED IN THE VIRGINIA TECH APPLIED LAB

The fifth step was to weld four tube pieces together to make up the upper chassis. The final step was to deburr all the parts and assemble the frame. The motor assembly consists of four main parts and is shown in Fig. 3. The motor is fastened to the leg frame by a clamping mount designed for the exact model. A coupler is used to convert the 6 mm output d-shaft of the motor to a 5 mm hex shaft. That hex shaft is connected to a 1:1 ratio right angle gearbox, which allows the wheels to be mounted perpendicular to the legs.

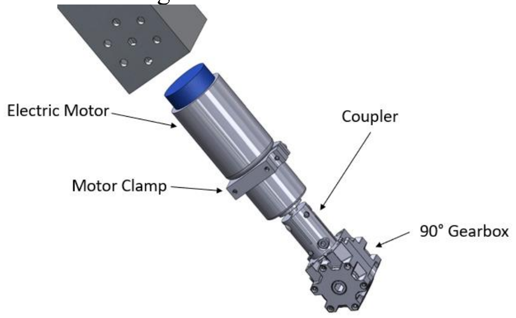


FIGURE 3: MOTOR ASSEMBLY SHOWING THE CLAMP, COUPLER, AND GEARBOX.

A custom shaft, shown in Fig. 4, was machined that converts the gearbox output to a 0.5-inch hex shaft needed to attach the wheels. The team utilized the Virginia Tech Applied lab to fabricate the hex shafts. The first step was to cut a piece of six foot long one inch diameter aluminum rod into four pieces using a band saw. The pieces were then machined using a manual CNC mill to form the hexagon shape.

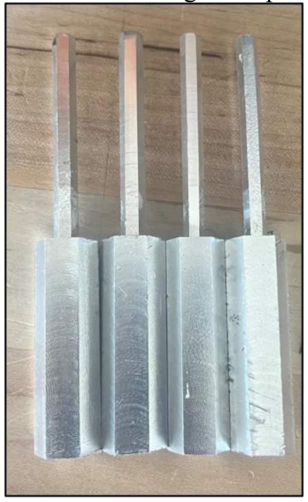


FIGURE 4: CUSTOM HEX SHAFTS FABRICATED IN THE VIRGINIA TECH APPLIED LAB.

In addition to the aluminum parts the robot also has many ABS components. ABS was chosen due to its widespread use in 3D printing, allowing for the creation of complex parts. These components include two battery support brackets, four cover plates, a lidar sensor mount, and four CO sensor mounts. Each of these parts were fabricated using a Stratasys F120 3D Printer provided by Virginia Tech.

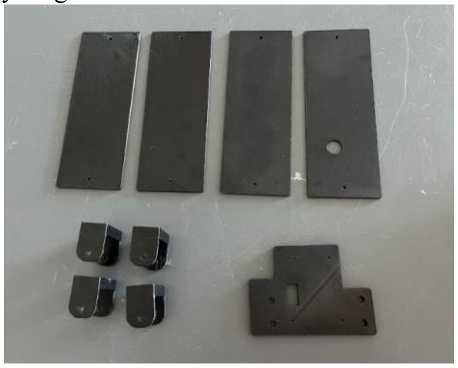


FIGURE 5: ABS PARTS FABRICATED USING A STRATASYS F120 3D PRINTER.

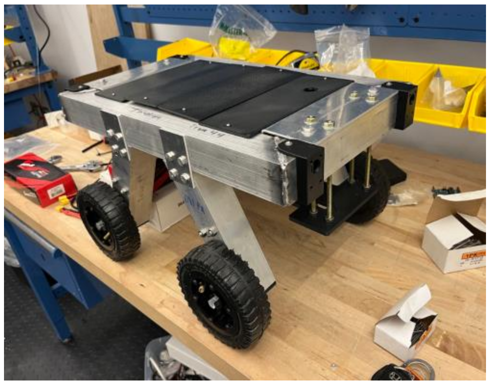


FIGURE 6: COMPLETED ROBOT CHASSIS INCLUDING ABS COMPONENTS.

2.3 ELECTRICAL CIRCUIT AND COMPUTER HARDWARE

In the robot's motor circuit design (Fig. 7), wire gauging plays an integral role in ensuring the efficient and safe transmission of electrical currents during its drive mode. Considering critical factors in motor circuits such as voltage drops, conductor material, and ambient conditions, the team identified the maximum current which each wire needs to carry for optimal motor control. The Arduino Giga handles the low-level control of motors and reads the object detection status determined by the connected Arduino Mega via an on/off digital pin. We selected the 12-V motors that cover the voltage range of 6V to 12V and supply speed of 118±12 RPM. The motors also come with encoders attached. The pulse amplitude of the encoder's output square wave signal is dependent on the voltage supplied to the sensor. For example, providing 6V to sensor, the channels will have pulse amplitude at the matching 6V. Use of electronics generate heat over time which can lead to deficiency and short lifespan of surrounding components. To account for the excessive heat and control the temperature of the internal components, a DC cooling fan was installed and connected to the Giga board. This fan is powered to facilitate steady-state airflow and regulate heat transfer in and out of the electrical components. Giga board helps to control the RPM of the fan to set fluid velocity and rate of heat transfer at an optimal level.

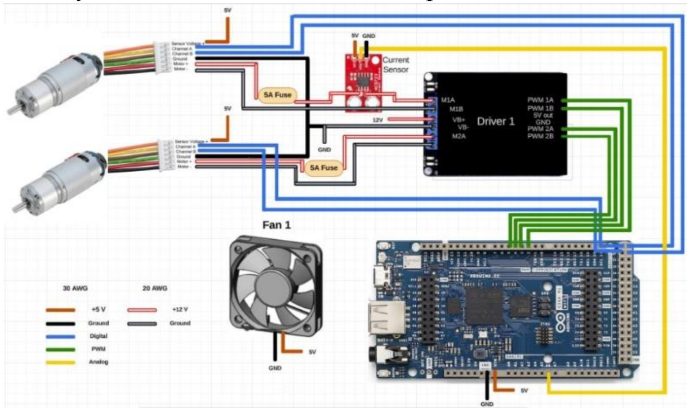


FIGURE 7: MOTOR CIRCUIT DIAGRAM

In Figure 8, the power supply diagram displays the connection between power supplies and PCB board. Two 12-V LiPo batteries are wired in parallel to PCB board to optimize the current flows and supplies equivalent voltage to maintain the operation of the vehicle without overheating one power supply over time. LiPo battery was selected over other power supply options such as lithium-ion battery to ensure there are higher and more sufficient voltage under high amp draw applications while maintaining lower overall temperatures under high discharge. Two LiPo batteries were used to account for their generally shorter battery life compared to runtime of lithium-ion batteries.

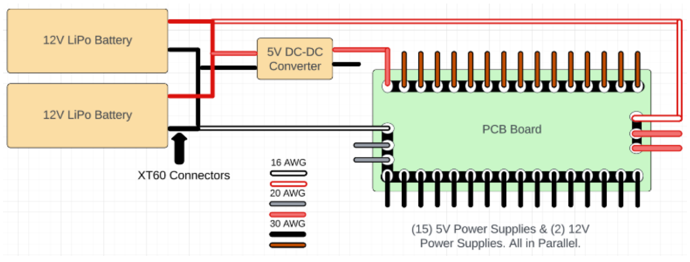


FIGURE 8: POWER SUPPLY DIAGRAM

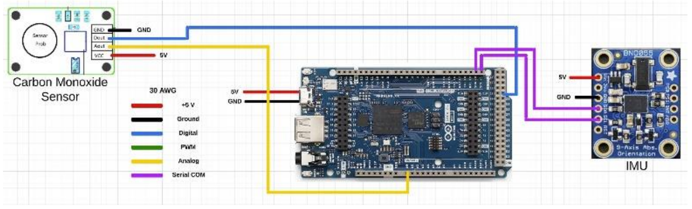


FIGURE 9: SENSOR TO ARDUINO CONNECTION DIAGRAM

Figure 9 displays the integration of the Carbon Monoxide and IMU sensors into the robot circuitry. This is pivotal for its operation. There is a potential for four CO sensors to be positioned strategically at each corner of the robot to maximize the detection coverage. This enhances the robot's ability to identify hazardous gas concentrations accurately. We are utilizing an MQ 7 carbon monoxide Arduino sensor which is operable in conditions between 10°C and 50°C and is able to detect the presence as well as concentration of carbon monoxide up to 10,000 parts per million. The sensor contains a 4-pin male header with the voltage, ground, digital output, and analog output pins are utilized. It is supplied with the necessary 5 volts to power itself from the Arduino and requires 30 minutes of preheating.

The Inertial Measurement Unit (IMU) integrated into the robot plays a crucial role in its navigation and orientation capabilities. This component combines accelerometer, gyroscope, and magnetometer data to provide precise real-time positioning and motion tracking. This improvement is vital for maintaining the robot's trajectory and operational integrity in complex and dynamic environments.

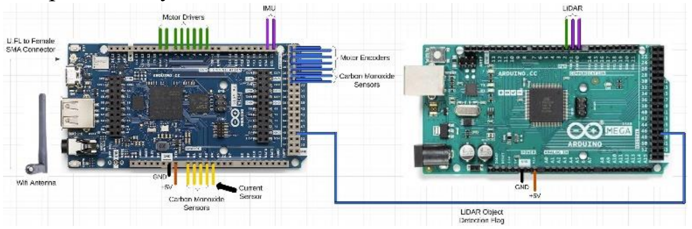


FIGURE 10: ARDUINO GIGA TO ARDUINO MEGA CONNECTION DIAGRAM

As per Fig.10, the robot employs dual Arduino Microcontrollers (MCUs) to manage distinct yet critical operations. The older Arduino Mega is utilized for the sake of compatibility. The utilized LiDAR has a data library not compatible with the Arduino Mega, hence the need for an additional MCU. This configuration optimizes task distribution and enhances system reliability and efficiency, data handling, and autonomous decision making. The ability for these controllers to communicate effectively ensures that the robot can maintain real time data syncing and operational coordination. The coordinated autonomy idea depends on the robust and real-time data interchange these MCUs provide through a dedicated server mechanism. Wi-Fi and serial connections are used to implement this communication, which supports data interchange using UDP message formats. This configuration not only guarantees smooth operation coordination between the robot's control (motor and sensors) and communication (data transfer and autonomy algorithms) components, but it also improves the robot's capacity to integrate and work in a fleet. The robustness and adaptability of this MCU communication method allows it to handle updates and modifications in operational parameters without affecting the robot's responsiveness or performance. The team is using the A1M8 model of the Slamtec A1 RPLiDAR which is a low cost 360° laser range scanner. This system also requires 5 volts which is supplied from the Arduino. This system can operate between 0°C and 40°C with an 8K sampling frequency. Most importantly, it has a measuring range of 0.15 meters to 12 meters.

2.4 SOFTWARE IMPLEMENTATION

To handle path planning protocol, the team utilized the A* path planning algorithm to calculate the most efficient path in a grid-like environment. The vehicle uses only vertical and horizontal movements due to its skid steering capabilities, and implementation of this level of autonomy was more achievable under the time restraints of this project. The algorithm works by considering nodes touching the start point and assigns a movement cost associated with those squares. Then, it assigns a separate cost for the distance between that square and the goal point. The node with the lowest total traversal cost is selected and then the process restarts by evaluating the squares touching that point. Once the end node is evaluated, the destination has been reached and the path is returned as a list of nodes to travel through. The path is then converted into a list of commands that will be wirelessly sent to the robot in an iterative loop, taking into account the vehicle's orientation in space.

2.4.1 WIRELESS NETWORK COMMUNICATION

Wireless transmission of data to and from the robot was achieved through User Datagram Protocol (UDP). UDP was chosen because of its simplicity and the speed at which information can be sent [7]. Figure 11 shows an overview of how UDP communication is utilized in the system. A smartphone's wireless hotspot was established as the server, which acts as an internet router handling communications between two clients. For this system, the two clients are the Arduino GIGA R1 and a laptop running the path planning code. Each client is connected to the hotspot's wireless network and is given an IP address. A UDP port number is defined in each of the clients' code. The laptop sends a data packet to the server containing its IP address and port number, a motor command, and the IP address and port number of the targeted receiver. Once the server receives the packet, it is then directed to the Arduino GIGA R1. The Arduino parses the packet and retrieves the motor command. The GIGA sends a return message in a similar fashion using the sender's IP address and port number from the original packet. The packet contains a signal indicating if the path is clear or not, followed by an integer value of the carbon monoxide ppm reading at that location. This process is repeated for each motor command that is sent.

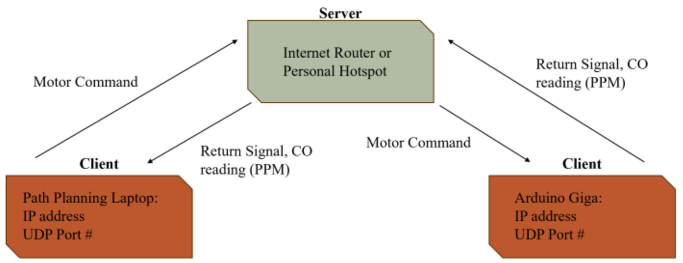

Server
Internet Router or Personal Hotspot
Motor Command
Return Signal, CO reading (PPM)
Return Signal, CO reading (PPM)
Motor Command
Client
Path Planning Laptop: IP address UDP Port #
Client
Arduino Giga: IP address UDP Port #

FIGURE 11: OVERVIEW OF SYSTEM SPECIFIC UDP COMMUNICATION

The purpose of the motor control is to execute grid-like movements with enough accuracy and precision that the robot can follow its desired path without absolute positioning feedback. The motor control also needs to ensure that when driving on varied surfaces and gradients, the motors do not exceed their nominal torque, which would cause overheating with prolonged operation. The maneuverability method involves four basic commands: forward, 90° left turn, 90° right turn, and emergency stop. The first three commands allow the robot to navigate the grid map generated by the path planning algorithm. The emergency stop command is used both to stop the robot when an object is detected and as a remote safety stop for the operator to use. During the design phase, simulations were completed to show that controlling the rate of acceleration and deceleration of the robot would be sufficient to limit applied torque on the desired surfaces and gradients of the robot's operability [8]. Therefore, correctly tuned speed control can be used to limit applied torque without being able to measure torque or current directly and accurately. This means that the robot will need to perform speed control and position control when executing a forward command. The rate of acceleration is controlled using a closed-loop velocity control loop with a setpoint that increases linearly with time (*APPENDIX 1*). This only requires a simple proportional controller to follow the desired speed and the time rate of change of the desired speed is what controls the rate of acceleration. Then, once the robot has reached top speed, a closed-loop position control loop, with Kp and Kd, is executed (*APPENDIX 2*). Here, the setpoint is the motor encoder counts associated with the desired grid size, assuming no slip. In this control loop the ratio between Kp and Kd controls the rate of deceleration, while Kp ensures that the robot approaches its desired stopping point. A theoretical approximation of the motion resulting from these sequential control loops is given in *APPENDIX 3*. It is important to note that the feedback in both control loops is truly motor position and motor speed because we don't have reliable, real-time positioning available on this robot, so we are approximating the robot's position and speed by assuming no slip.

A 2-D LiDAR sensor was implemented to fulfill the requirement of obstacle detection. The LiDAR code was programmed on the Arduino MEGA using libraries supplied from the manufacturer. As the sensor rotates, it returns data corresponding to a relative angle in degrees and a distance in millimeters. From this data, the closest object at every angle can be determined. Each data point that is collected is processed to determine if the object is in front of the robot. If the returned angle is between 0° and 90° or 270° and 359°, it is added to a buffer with its corresponding distance. This process omits any objects behind the robot, which may include the robot itself. Once the buffer is full, secondary post-processing begins. The robot is programmed to only move forward to the grid space directly in front of it. Therefore, any buffer data corresponding to objects in diagonal or distant grid spaces must be disregarded. To minimize computing cost, trigonometric functions were not used to break down the objects into x and y locations. Instead, 3 search areas were calculated to increase the coverage area within the targeted grid space. Figure 12 shows a graphical representation of this design. Each search area has angle interval and maximum distance allowed. The search area with the largest distance coincides with the furthest side of the targeted grid space at an angle of 0°. Consequently, this search area has the smallest angle interval to ensure objects in diagonal grid spaces are not recognized. As the maximum distance decreases, the angle interval is allowed to increase. If a buffer data point is within one of the search areas, a variable corresponding to the number of obstacles is incremented by 1. After all of the buffer data has been processed, if the number of obstacles is greater than a specified threshold, the path is deemed to be not clear. Otherwise, the targeted grid space is assumed to be traversable. Based on this assessment, the MEGA will change the state of the pin that is connected to the GIGA. Once finished, the buffer data will be erased, and data collection starts again. The number of search areas can be increased to cover a greater percentage of the targeted grid space, but this will cause the time between pin state updates to increase, potentially resulting in an incorrect reading by the GIGA. The 3-search area method was chosen because it optimized these two parameters.

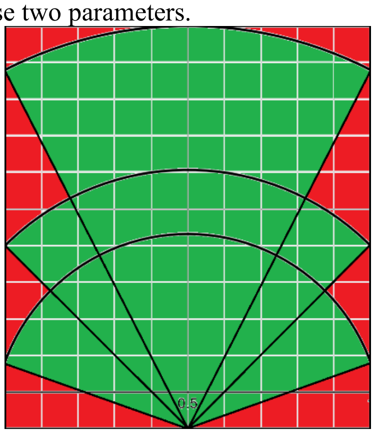


FIGURE 12: GRAPHICAL REPRESENTATION OF LIDAR SEARCH AREAS

A function was programmed on the Arduino GIGA to return an integer value of the carbon monoxide ppm at the robot's location. The function begins by collecting ten ppm values from the carbon monoxide sensors and averaging them together. The raw value is then normalized by a specified calibration value of 105 ppm. The reading is then linearly mapped to atmospheric levels and returned to the main loop for transfer to the path planning laptop.

2.4.2 WHILE-LOOP ITERATION

While loop iterations are adopted and embedded across the autonomous vehicle for the continuous cycle of motion control, path planning and gas concentration data recording, and wireless connectivity. Continuous motor control utilizes a while loop iteration to accelerate to a specific threshold and come to a complete stop after moving by one unit of the mapping grid. The motherboard is programmed to take in motion commands at any given real time while the robot's drive mode is on. Given a command "forward", the robot supplements 25 inches of step size to move a single unit of grid. This velocity increases in a fraction of real time until it hits meets threshold and begins to de-accelerate until coming to a complete stop at one unit of the mapping grid. With autonomous functions applied, the commands are automated based on recognition of obstacles along the path and iterates through binary decision-making process to execute skid steering turn upon encountering obstacles and resume with the optimal path possible. While in motion, iteration applies in similar way for measuring gas concentration in which the wired carbon monoxide sensors continuously iterate quantitative data recording upon each stop of the robot. All these functions are possible by establishing a steady wireless network with the off-site controller or laptop which receives feedback on the robot's navigation route decision, recognition of underlying objects in the path, and CO measurements.

3. RESULTS AND DISCUSSION

A battery life test was conducted to determine if the robot met the requirement of a 30-minute minimum operating time on one charge. Furthermore, Li-Po batteries will be damaged if their voltage drops below a certain value. This minimum threshold is 9 Volts for the batteries used. The robot was driven on flat ground in a 3 by 3 square grid pattern for 30 minutes. At the beginning of the test each battery was at a full charge of 12.6 Volts. The voltage of each battery was recorded after 15, 25, and 30 minutes of driving. The voltages were measured using a dual channel smart charger with an accuracy of ±0.01 Volts. The results of the test can be found in Table 2. After 30-minutes of driving, each battery had a voltage of around 12.2 Volts. These results are better than anticipated from theoretical calculations, which assumed constant power draw with peak motor current. This is because the robot was designed to stop after each command and check sensor data. Additionally, the current draw never reached its maximum value on the testing terrain. The battery life test concluded that the required operating time had been met and the batteries are not in danger of being damaged during the operation cycle.

TABLE 2: BATTERY LIFE TEST RESULTS

	Time (min)	0	15	25	30
Battery 1	Voltage (V)	12.6	12.48	12.26	12.21
Battery 2	Voltage (V)	12.6	12.47	12.26	12.2

To test the robot's top speed, the motor control code was modified, increasing the step size to 10 feet, because the final iteration which uses step size of 25 inches doesn't give the robot enough time to reach full speed—this is not a problem for its own sake, but only for the sake of testing the top speed the robot is physically capable of. The encoders and a hardware interrupt at 500 Hz were used to calculate average rpm of the motors in real-time at 0.002 second intervals, which were sent over serial port to the computer display. Testing was done on the rubber-matted lab floor, so it was a 0% gradient and high traction. Assuming no-slip then, collecting to top recorded motor speed in 5 trials we find an average top speed of 87.61 rpm. With 6-inch diameter wheels, that converts to 0.6992 m/s for the robot. This is less than our requirement, but in line with what we expected, because we had a supply chain issue that forced us to buy different motors, which were one size down in rpm from the ones we designed for.

The motor torque tests were performed on varied terrain and gradients that could be found around the Gilbert Lab building. Motor speed, calculated as in the top speed test, and applied PWM signal were collected and output at 500 Hz. Using the given properties of our motors, we can calculate applied torque at each instance by using the Speed-Torque curve, based on motor properties, and the input voltage, where applied motor voltage is proportional to PWM value, and output speed, in rpm. A sample data set of real-time calculated motor torque is given in *APPENDIX 4*. Our objective was to drive on grass at 5% gradient as well as small bumps without exceeding nominal torque as well. The surfaces and gradients tested are shown in Table 3. The results show that the motor control implementation as well as proper motor sizing provided a very robust system capable of driving on rougher terrain than the objective, including 2-inch-tall curbs, while maintaining under nominal torque.

TABLE 3: MOTOR TORQUE TEST RESULTS

Surface	Gradient (%)	Peak Torque (kgf-cm)
Rubber Mat	0	8.95
Grass	0	8.97
Sidewalk	8	8.93
Grass	12	8.92
Curb	1 inch	9.05
Curb	2 inches	9.07

In a critical test of the robot's functionality, it was programmed to navigate a straight-line path within our test environment. At the third of five designated boxes in our grid space, a controlled introduction of carbon monoxide was executed to simulate a leak utilizing a lighter. The sensors successfully detected a spike in Co levels, demonstrating the robot's gas sensing capabilities. Throughout the test, the robot demonstrated path planning in autonomous operation while continuing the data collection process. This test confirmed the robot's ability to operate autonomously and adaptively while collecting CO data, which validates its design and operational objectives.

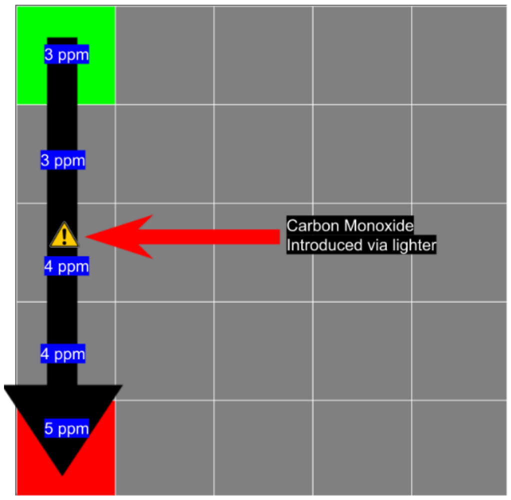


FIGURE 13: GRAPHICAL REPRESENTATION OF CARBON MONOXIDE TEST

The path planning algorithm testing for our autonomous vehicle as it traverses through real space using grid-like movements was a crucial part of the team's validation process. To validate the A* algorithm code, the team used a variety of software tests that tested basic and advanced configurations of the map to ensure navigational efficiency virtually. Some boundary conditions that were verified through testing included testing the map perimeter boundary, returning error messages when no viable path was found, and using mathematical checks on hardcoded maps to verify that the most efficient path was taken. Through using this approach, the team was able to validate the path returned from the A* function was correct. From here, the team transitioned to testing the efficacy of the solution by confirming the vehicle could reach the destination in a real-world scenario. The protocol for path planning testing was as follows: Generate a series of motor commands from the output of the A* function, considering parameters such as: the start and end point, the orientation of the vehicle in space, and the obstacles submitted in the map message. The list of commands then is transmitted wirelessly to the on-board Arduino Giga using a UDP connection. After validating that each command sent was completed in the physical world, the Giga sends a return message informing the base computer that the command was executed. Then, test facilitators could manually verify that the commands were being executed in real time. In the testing facility, the team then re-performed the earlier tests that were validated in software to ensure that the communication protocol was being executed properly. Through this approach, we were able to ensure proper coordination between the base computer and the Arduino Giga, validating our path planning and communication protocol requirements in the process.

The next phase of testing was targeted at verifying our autonomous capabilities by responding to unknown objects in the map and recharting a course to the goal point. This inclusive test in essence verified all autonomous subsystems of our vehicle: path planning (including renavigating), lidar obstacle detection, UDP protocol, and base computer to vehicle communication. These major capabilities are crucial for integration with Noblis' O.A. integrated robots, as they provide an interface for communication in a coordinated workplace. Therefore, it was crucial to test the ability to respond to unknown obstacles with error messages that can eventually be deciphered by a team of vehicles working in a response scenario. Our main focus for this test was to validate this communication by intentionally placing obstacles in the original path of the vehicle and verifying that the error protocols were accurately handled and communicated to the base computer. From there the base computer would accept the new obstacle information, create a new instance of the map and restart the process of sending commands to the vehicle from its current goal point. In Fig. 14 this basic scenario demonstrates logical differences in vehicle response to the same configuration of the map. In Case A, the three obstacles are placed around the goal node. Using this information, the shortest path available is outlined in blue. However, in Case B, the vehicle is initially unaware of points A and B, but aware of C. It initially calculates a path through points A and B before the lidar sensor detects obstacle A. It renavigates above and around before meeting obstacle B. Again, it recalculates around both obstacles and to the goal point. Note that there was an observed bias toward traversing upward on the map in both scenarios. While the path toward the bottom of the screen is also viable, the way that our program was set up was to start by considering squares in a clockwise order from the square above it. Refer the software implementation discussion in Section 2.4.

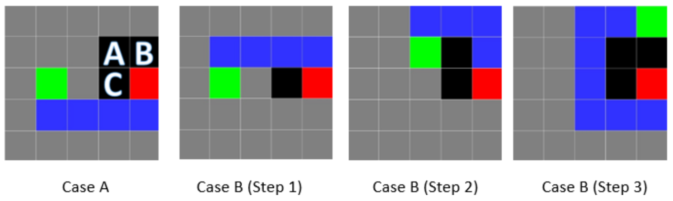


FIGURE #14: Difference in complete vs incomplete map state

This example is just a simple demonstration of the vehicle's ability to respond to unexpected navigational scenarios and how it communicates with the base computer to recalculate a path. From here, the team again tested various boundary cases, and handled situations where no path was found. The system was able to generate accurate status messages at all stages of the process, ensuring that it can properly be integrated with Noblis' robotic team, and validating our base level autonomy requirements.

In testing of the robot, a key structural failure related to fatigue design was noted. Custom hex shafts which join the motor gearboxes and wheels of different hex sizes underwent brittle fatigue failure due to accumulated stress concentration. The large stress concentration was found in the midpoint of the hex shaft at which different hex shaft sizes were met. From the result of finite element analysis in Fig. 15, the wheel end of the shaft experienced maximum amount of stress largely due to torque. To solve this issue, the team recommends the following approach: 1) designing a new shaft which tapers down to the smaller hex to reduce the stress concentration or, 2) using a material with a higher fatigue strength such as steel to increase the fatigue limit.

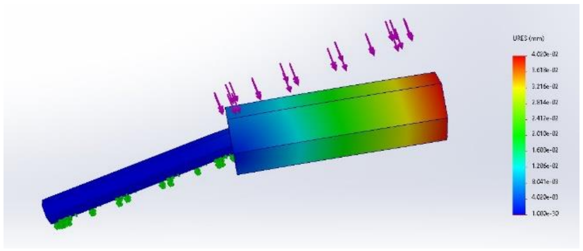


FIGURE #15: Deformation results of the hex shaft.

Working with limited space for the electrical circuit, the team observed that wires producing electrical noise and ultimately disrupting the motor control and IMU positioning signals. The space restrictions resulted in electromagnetic inferences among the connected wires running across the functioning boards. For future solutions, further reorganization of electrical circuit or soldered wires to the PCB board may be necessary to reduce the noise level due to electromagnetic interference and the chance of loose wires from motion vibrations.

4. CONCLUSION

The team has successfully reached the goal of general autonomy of desired task and compatibility with orchestrated autonomy despite many mechanical and software challenges related to the autonomous vehicle which the team had to overcome. The team's key achievements were the following: using finite element analysis to identify and mitigate structural strength and failures, discovering underlying known or unknown obstacles, overviewing quantitative mapping measurements to take recourse during navigation if necessary, recording quantitatively distinguishable data of toxic gas concentrations, and establishing steady wireless network for data information exchange across different platforms. However, certain features including advanced LiDAR mapping in 3-D and continuous motion and high-resolution path planning proved to be outside the project's scope due to time constraints. The robot's efficiency in completing its individual tasks may remain as a room for improvement. Overall, the autonomous vehicle in its current form has been showcased to Noblis and determined to be useful in testing the limits of orchestrated autonomy, exceeding the industry sponsor's technical expectations.

ACKNOWLEDGEMENTS

Place any acknowledgements here.

REFERENCES

[1] Williams, H. (2022, April 12). Noblis awarded patent for Autonomous Machine Coordination System. Noblis. https://noblis.org/patent-autonomy/

[2] Bashir, H., Mahalwar, G., & Henry, T. (2023, October 11). *The East Palestine Disaster: The potential toxic effects of vinyl chloride exposure on Cardiovascular Health*. Cureus. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10636715/

[3] Liu, X., Cheng, S., Liu, H., Hu, S., Zhang, D., & Ning, H. (2012). A survey on gas sensing technology. *Sensors (Basel, Switzerland)*, *12*(7), 9635–9665.

[4] R. Clark, "Lecture 2 - ME4015 - Clark," presented at the ME_4015,Virginia.[PowerPoint].Available: https://canvas.vt.edu/courses/178772/files/29523158?

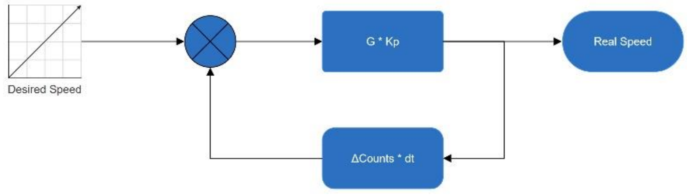
[5] "Interfacing MQ-7 Smoke Gas Sensor Module with Arduino,"Electropeak,Dec.08,2020.https://electropeak.com/learn/interfacing-mq-7-smoke-gas-sensor-module-with-arduino/ (accessed Oct. 26, 2023).

[6] "Specification for Aluminum-Alloy 6061-T6 Standard Structural Shapes, Rolled or Extruded (Metric)," American Society For Testing and Materials, Feb. 2021, Available: https://compass.astm.org/document/?contentCode=ASTM%7CB0308M-88%7Cen-US

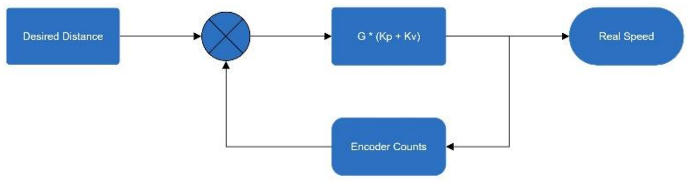
[7] Rosencrance, L., Lawton, G., & Moozakis, C. (2023, December 6). *User Datagram Protocol (UDP)*. Networking. https://www.techtarget.com/searchnetworking/definition/UDP-User-Datagram-Protocol

[8] Team 44, "Critical Design Review: Functional Robot for ID and Retrieval," Dec. 6, 2023. https://doi.org/10.3390/s120709635

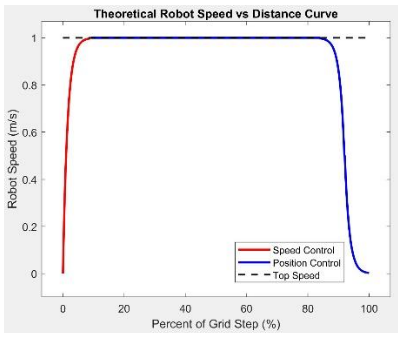
APPENDIX



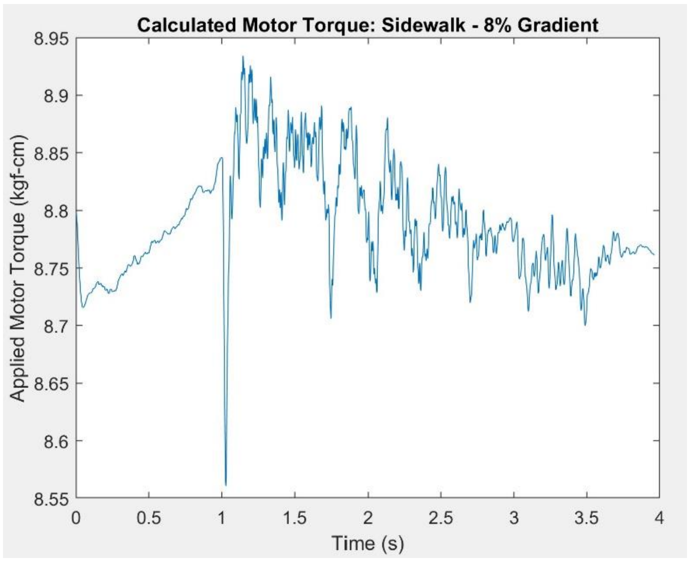
APPENDIX 1: Closed-Loop Speed Control of Motors



APPENDIX 2: Closed Loop Position Control of Motors



APPENDIX 3: Robot Speed During Forward Command



APPENDIX 4: Real-Time Calculated Motor Torque Data Sample