Federal University of Technology of Paraná – UTFPR
Academic Department of Electronics – DAELN
Academic Department of Informatics – DAINF
Computer Engineering
Integration Workshop 3 (EEX23) – S71 – 2023/1

Technical Report
Vipper

Cássio K. Yamauchi yamauchi@alunos.utfpr.edu.br
Cesar A. P. Vial cesarvial@alunos.utfpr.edu.br
Gustavo H. Zeni gustavozeni@alunos.utfpr.edu.br
Paulo S. Á. Júnior pauloj.2018@alunos.utfpr.edu.br

June 2023

Abstract

The adaptability and versatility of soft robots has gathered attention due to their potential in solving problems in a myriad of areas. One often tackled problem is their utility in accessing hard-to-reach areas, since it could improve the chances of survival of victims in cases of individuals stuck in enclosed spaces like debris, for example. With this in mind, the Vipper was created in order to prototype a simple vine robot with the function of mapping, as there were only mentions of this in other articles, but no real implementation of mapping an unknown environment was done. This way, it could be possible to utilize the vine robot in places blocked off by events which only leave narrow entrances, reducing risks for both rescuer and survivor by utilizing the soft robot for exploring the region and identifying the victim, while mapping all of its course. Afterwards, it could be possible to locate the victims' location by reading the generated map, minimizing fruitless excavation, which could lead to further problems if done wrong.

1 Introduction

Landslides and earthquakes are common in many areas of the world, and they result in thousands of dollars in damages and bring risk to thousands of lives every year. Whenever a landslide or earthquake results in burial of persons, rescue teams are deployed in order to rescue the victims in a lengthy process that involves digging out parts of the resulting rubble very slowly in an attempt to find and remove the buried victims from under the debris.

Many projects have been created to try to ease this proccess, such as using a Pulse Sensor to detect the survivors' heartbeat under the debris [1] or Eletromagnetic Waves to detect breathing [2]. Most of these projects are theoretical in essence, and would be too expensive and complex to deploy in real rescue operations. A common solution to search and rescue problems is the use of robots to navigate the terrain and locate victims in hard to access places - in [3], a number of robots that could be used for fast rescue operations in mines are presented and the difficulties encountered discussed. In [4], Kamegawa presents the idea of a Snake Robot that was built by connecting multiple crawler robots serially for use in rescue ops performed on collapsed buildings.

A robot design that has gained much popularity recently is the Vine Robot, a self-growing robot that uses flexible materials to reduce friction with the ground by growing forward, allowing it to traverse highly perilous terrain that other kinds of robots wouldn't be able to manage. In [5], Blumenschein documents a number of designs for vine robots that have been used with varying degrees of success for many different purposes in the past, and in [6], Manoj presents a design for a vine robot to be used in upper gastrointestinal endoscopic imaging.

We propose a new system that uses a vine robot to enter small holes and explore the area underneath the rubble resulting from collapsed buildings, allowing the rescuers to explore in real time and communicate with any buried victims - Vipper, the Mapper Vine. We will give a full overview of the project in section 1.1, explaining the concept behind our solution. Then, in section 2, we will list the requirements that were elaborated to specify the scope of our solution. In section 3 we discuss the Mechanical, Hardware and Software design of the Vipper. Finally, in sections 4 and 5 we present the results of our project and our conclusions.

1.1 Overview

Figure 1 presents the block diagram of the Vipper project. The Vipper is divided in four main parts: Desktop App, Base, Body and Head. The user will only interact with the Desktop App and the Base. Each part will be explained below.

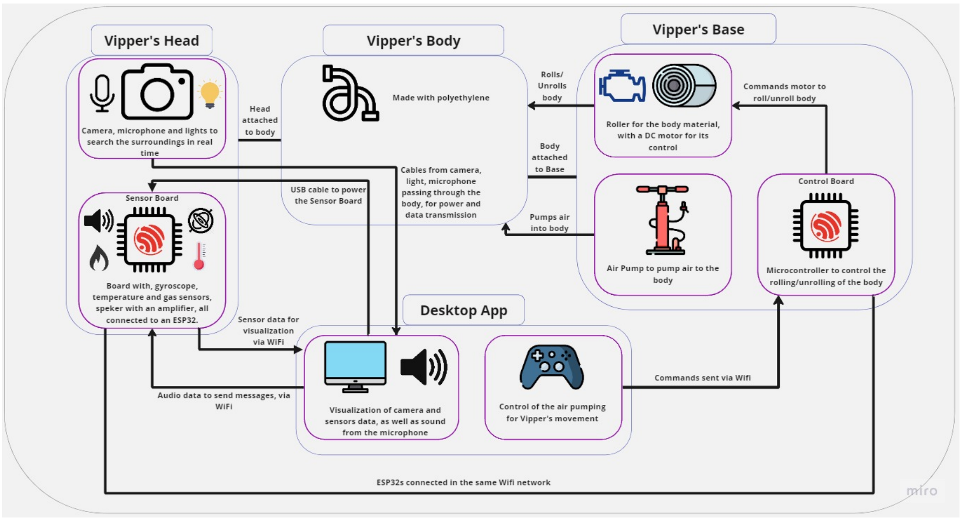
Vipper's Head
Camera, microphone and lights to search the surroundings in real time
Head attached to body
Sensor Board
Board with, gyroscope, temperature and gas sensors, speker with an amplifier, all connected to an ESP32.
Vipper's Body
Made with polyethylene
USB cable to power the Sensor Board
Cables from camera, light, microphone passing through the body, for power and data transmission
Rolls/ Unrolls body
Body attached to Base
Pumps air into body
Vipper's Base
Roller for the body material, with a DC motor for its control
Commands motor to roll/unroll body
Air Pump to pump air to the body
Control Board
Microcontroller to control the rolling/unrolling of the body
Desktop App
Sensor data for visualization via WiFi
Audio data to send messages, via WiFi
Visualization of camera and sensors data, as well as sound from the microphone
Control of the air pumping for Vipper's movement
Commands sent via Wifi
ESP32s connected in the same Wifi network
miro

Figure 1: Block diagram of the Vipper project.

The Desktop App is the only interface with the user. It is implemented in Python, and allows the user to command the base to roll and unroll the body, to view the current temperature and presence of harmful gas in the head, to view the current path the Vipper went through, to see and hear what the webcam is capturing in real time, and to send messages to be played in the head. The communication between the Desktop App and the boards is made via Wi-Fi. The communication between the Desktop App and the webcam is made via a 5 meter usb cable.

The Base is where the Control Board is located. The Control Board consists of an ESP32 connected to a H-bridge that controls a DC motor. The DC motor is connected to a spool where the body is rolled. In the base, there is also a 12V air compressor with an air compressor regulator to pump air into the body for its expansion. The air compressor regulator must be manually controlled by the user.

The body consists of a polyethylene roll. When the body is unrolled in the base and air is pumped inside, the body grows forward. In the body there are also two 5 meter usb cables, connecting the Desktop App with the head. One cable is used to power the webcam and transmit data from the webcam to the App, the other cable is only used to power the Sensor Board.

In the head, there is a webcam with microphone and lights integrated, and the Sensor Board. The Sensor Board's central component is an ESP32, that is used to gather data from a temperature sensor, a gas sensor, and a gyroscope, and send this data to the Desktop App via Wi-Fi. The Sensor Board can also receive audio data to be played, using an amplifier and a speaker. With the microphone and the speaker, the Vipper has two-way real time voice communication between the head and the operator of the Desktop App.

The four parts of the Vipper and their respective components and functionalities are going to be explained in detail in the Development section (Section 3) of this report.

2 Project Specification

The requirements of the Vipper project are listed in the subsections 2.1 and 2.2.

2.1 Functional requirements

- **RF01**: The Vipper's body must be able to move through an area by using an air pump.
- **RF01.1**: The Vipper's body must be able to expand, going forward, by pumping air into and unrolling it.
- **RF01.2**: The Vipper's body must be able to retract, going backwards, by re-rolling it.

- **RF02**: The Vipper's body must be able to bypass obstacles.
- **RF02.1**: The Vipper's body must be able to bypass rocks of a maximum diameter of 10cm.
- **RF02.2**: The Vipper's body must be able to squeeze through holes of a minimum diameter of 12cm.
- **RF03**: The system must collect images from the Vipper head's trajectory.
- **RF04**: The Vipper's head must reproduce messages from the user.
- **RF05**: The system must be able to map its explored trajectory.
- **RF06**: The system must collect data from the Vipper head's surroundings.
- **RF06.1**: The system must measure the temperature in the Vipper's head surroundings.
- **RF06.2**: The system must detect the presence of harmful gas, specifically Nitric Oxide and Carbon Dioxide, in the Vipper's head surroundings.
- **RF06.3**: The system must capture sound from the Vipper's head surroundings.
- **RF07**: The data collected from the environment by the MQ135 gas sensor and the MPU6050 (gyroscope, accelerometer, temperature sensor) must be sent to a desktop app.
- **RF08**: The desktop app must be able to save the collected data to a file.
- **RF09**: The desktop app must show to the user the images captured from the Vipper's head with a delay of less than 5 seconds from the image's capture.
- **RF10**: The desktop app must save the images captured from the Vipper's head. The camera must have internal illumination and a microphone.
- **RF11**: The desktop app must be able to show to the user the system's mapped path. An MPU6050, 3 axis accelerometer and 3 axis gyroscope, should be used as sensor.
- **RF12**: The desktop app must allow the user to control the Vipper (extending/retracting the body, visualizing the image seen by the head, hearing the sounds close to the head, and sending audio signals to be reproduced by a speaker in the head).

2.2 Anti-requirement

- **AR01**: The system will not be steerable.

3 Development

The development of the Vipper can be separated in three parts, the Mechanical, Hardware and Software designs. These parts are explained in subsections 3.1, 3.2 and 3.3, respectively.

3.1 Mechanical Design

The mechanical design can be divided in two parts, the base and the head. Both parts are explained in the sections 3.1.1 and 3.1.2.

3.1.1 Base Design

The main mechanical design of the project was the project of the Vipper's Base. The Pre-Formed Vine Robot version, from Stanford University [1], was used as reference for the base project, because of budget constraints. They also made available for everyone the 3D modeling of the base, which can be seen separately in figure 2, and mounted in the figure 3. Some of these parts were also 3D printed to be used, and they can be seen in the figure 4.

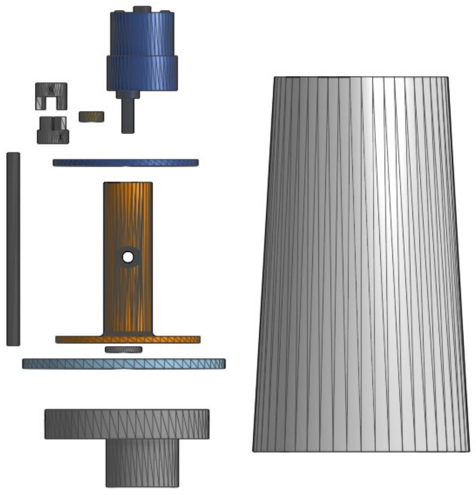
Figure 2: 3D modeling with all separated parts.

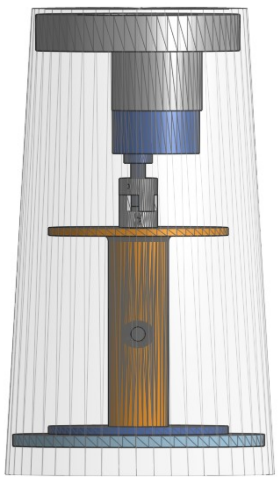
Figure 3: 3D modeling with the full mounting.

[1] P.-F. V. Robot, "https://www.vinerobots.org/build-one/pre-formedvine-robot/."

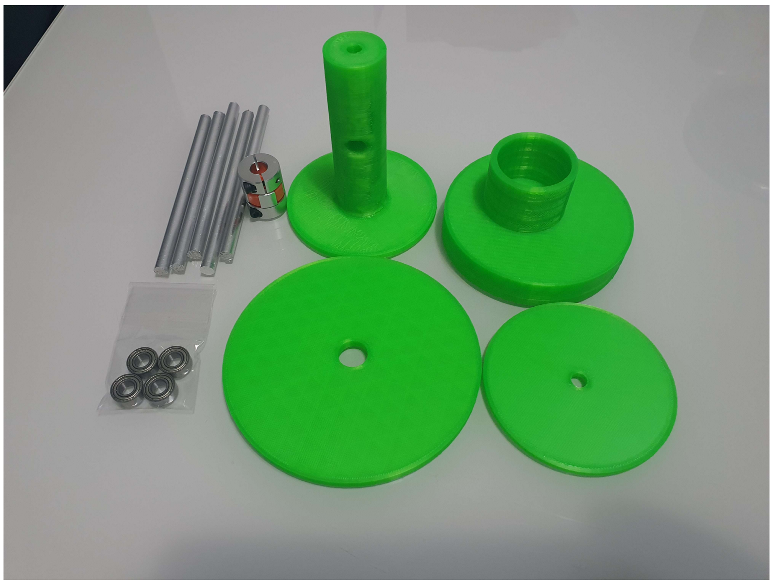

Figure 4: Printed parts with metal bar, spider coupler and bearings.

The end product was not exactly as planned, because the team could not acquire the exact materials used by the Pre-Formed Vine Robot [2]. Only two parts are different from planning: the plastic recipient and the DC Motor. Because of these different materials, another way to hold the DC motor for its operation was used, adding a closed plastic material to the top of the base. The group also added a wooden board to put the base, the control board and the 12V power source together, making it easier to transport and setup for operation. Figures 5 and 6 shows the base of the Vipper seen from the top, and a close to show the inside.

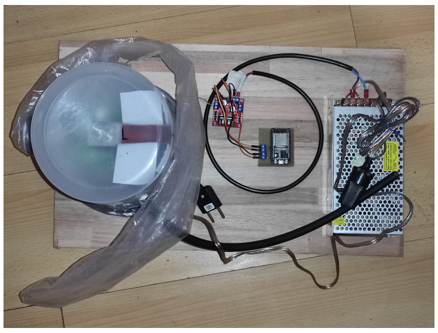

Figure 5: Vipper's base seen from top.

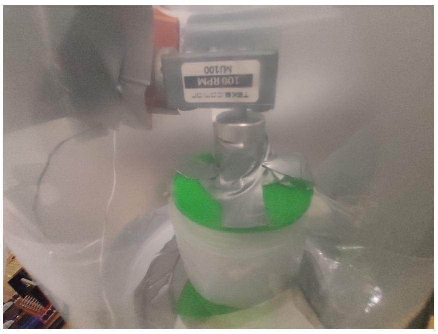

Figure 6: Close on the inside of the Vipper's base.

[2]P.-F. V. Robot, "https://www.vinerobots.org/build-one/pre-formedvine-robot/."

3.1.2 Head Design

For the head design, it was necessary to attach the webcam in front and the sensor board behind it, for it to be able to capture the images from the front. It needs to have necessary space for the body to be able to push it forward, and two usb cables should pass through to power the webcam and the head.

During development, the cables where not passed through the body, the team just connected the cables in the desktop and in the head. The main problem was space in the spool and in the base, since both cables would need to be inside the body rolled in the spool. Another problem was maintenance, when the body is rolled back, it does not stay perfectly like the first mounting, and it would be way worse if the cables needed to roll together with the body, and changing the roll of the body in case of severe damage would also need more work.

Figures 7 and 8 show the head seen from the front, showing the webcam, and from the side, showing the sensor board, respectively.

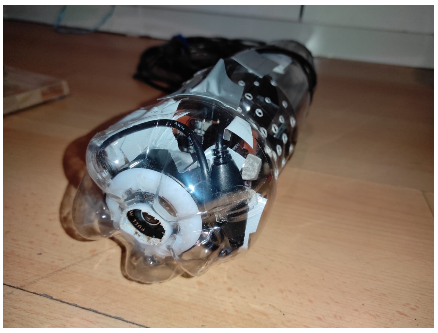
Figure 7: Vipper's head seen from the front.

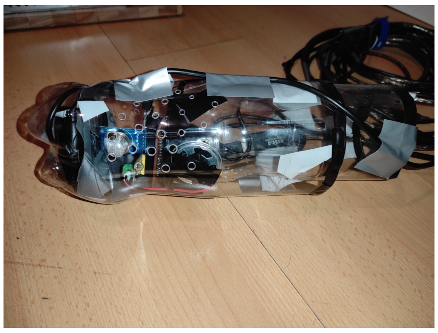
Figure 8: Vipper's head seen from the side.

3.2 Hardware Design

The hardware design is divided into three main parts: the webcam, the control board and the sensor board. The webcam is utilized for capturing audiovisual data and emitting light to the environment. The control board is used to connect the microcontroller to the H-bridge driver module. The sensor board is very compact, due to the head size limitations, consisting of another microcontroller, sensors for reading surrounding data and an amplifier with a speaker for communicating with the victims. In Figure 9, it is displayed the general connection among the components. The following three sections detail more the purpose and each of the components utilized in the project.

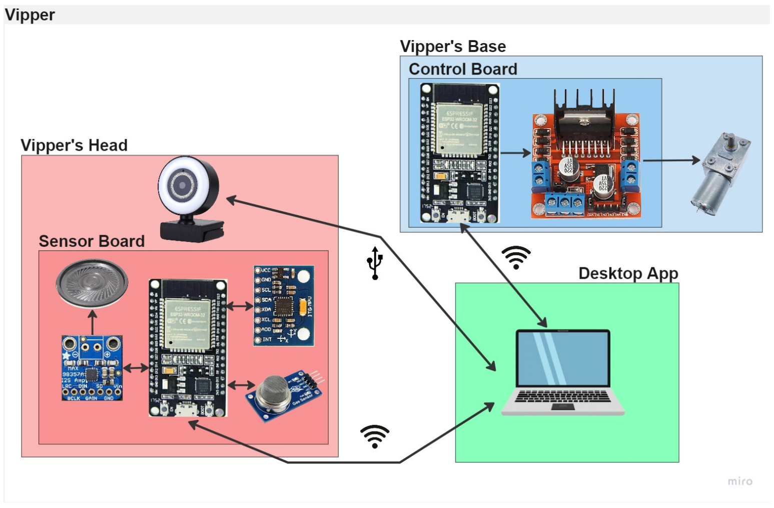

Figure 9: All hardware components and their connections.

3.2.1 Control Board

The control board was designed to connect the microcontroller responsible for the logical signals that command the rotation of the DC motor with the H-bridge driver capable of handling the current and direction of the said motor. Due to its simplicity, it was implemented with a universal PCB. The board's connections can be seen in Figure 10. In this board, the chosen microcontroller was the ESP32 due its embedded Wi-Fi module and good cost-benefit, while also having reasonable memory and processing performance to handle a Wi-Fi server point.

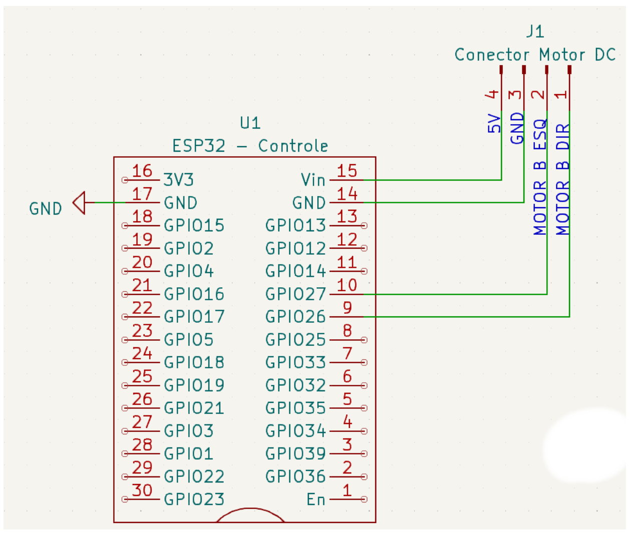

Figure 10: Schematic of Control Board

3.2.2 Sensor Board

The sensor board was designed to handle all of the sensors' data gathering and transmission, allow communication of the user with the Vipper's head neighboring, while also being as compact as possible for minimal interference with the size restrictions of the head. The sensor board also has additional circuitry to allow it to accept the backup plan, in case the power supply by USB cable failed. As such, the sources (+9V) in the schematic represented in Figure 11 are representing a situation where the batteries are utilized. In the current state, the USB port of the ESP32 was sufficient to power the whole system. The utilized sensors were: a MPU6050 module for measuring temperature and for its accelerometer and gyroscope function, and a MQ135 for detecting dangerous gases, e.g., ammonia gas, sulfide, benzene series steam and carbon dioxide. The audio amplifier utilized was the MAX98357A, paired with a 0.5W 8Ω speaker.

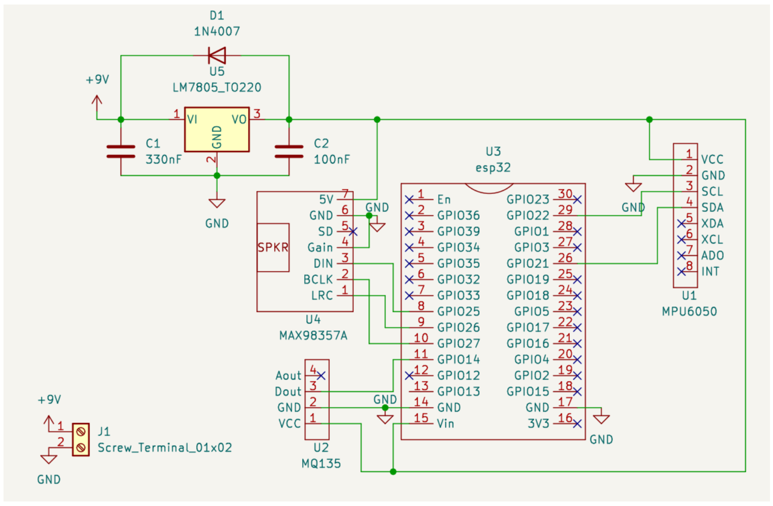

Figure 11: Schematic of Sensor Board

In Figure 12, it is possible to see the finished sensor board.

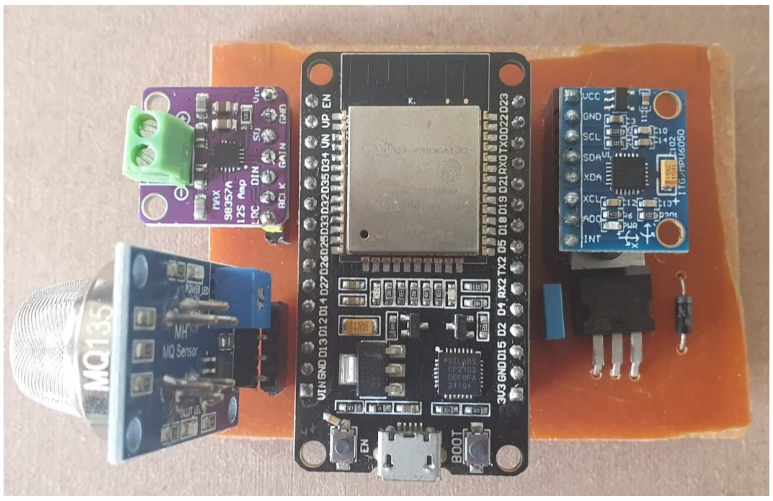

Figure 12: Sensor Board with components

3.2.3 Webcam

The webcam is utilized to light the environment and provide the user with audiovisual data. This webcam was chosen due to its size, integrated lights and round shape, being ideal for meeting many requirements at once with as few resources as possible. The webcam is connected to the user's laptop by a 5m USB cable. This way, it is possible to monitor and transfer data from the webcam in the Vipper's head to the user.

3.3 Software Design

The software design is divided in three major components: the Control Board firmware, the Sensor Board firmware and the Desktop App. The Control Board firmware is run by the Control Board in the base of the Vipper, and it receives commands from the Desktop App to activate the rolling and unrolling mechanism in the base of the Vipper in order to allow the Vipper to grow or retract. The Sensor Board firmware is run by the Sensor Board in the head of the Vipper in order to collect data from the robot's sensors and send them via Wi-Fi to the Desktop App. The Desktop App is run in a computer, and its main function is to serve as a Human-Machine Interface for the Vipper, allowing a human operator to control the robot's movement - unroll to go forward and roll to go backwards - and visualize the data being collected by it.

The following subsections contain more detailed explanations about how each of the software components were designed and implemented.

3.3.1 Control Board Firmware

The Control Board Firmware has one function: to implement the growing and retraction function of the vipper. It is composed of a TCP Socket that receives commands from the Desktop App and two GPIO outputs that activate an H-Bridge connected to the DC motor in the Vipper's base, as illustrated in figure 13. When the Control Board receives a command to grow, it enables the DC motor to rotate the base's spool, unrolling some of the Vipper's body and allowing it to grow. When it receives a command to retract, it rotates the base's spool the other way, rolling some of the Vipper's body back into the base.

The Control Board Firmware was written in C++ using ArduinoIDE. It runs on the ESP32 installed in the Vipper's base. The code is composed of a single State Machine that implements connecting to the desktop app and interpreting the commands received, that may be one of either FORWARD or BACKWARD.

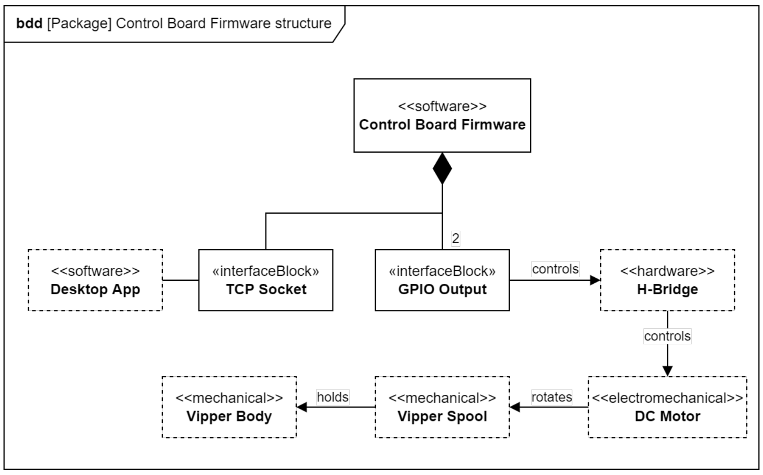
bdd [Package] Control Board Firmware structure
<<software>> Control Board Firmware
2
<<software>> Desktop App
«interfaceBlock» TCP Socket
«interfaceBlock» GPIO Output
controls
<<hardware>> H-Bridge
controls
<<mechanical>> Vipper Body
holds
<<mechanical>> Vipper Spool
rotates
<<electromechanical>> DC Motor

Figure 13: Control Board firmware block diagram

3.3.2 Sensor Board Firmware

The Sensor Board Firmware implements reading the gyroscope, thermometer and gas presence sensors' data, sending the data collected to the Desktop App and playing voice messages received from the Desktop App through a speaker. It works by periodically reading data from the sensors and sending it to the Desktop App through a TCP Socket, as illustrated in Figure 14. Whenever the Sensor Board receives a voice message from the desktop app, it stops reading the sensors and plays back the message with the board's amplifier module via I2S.

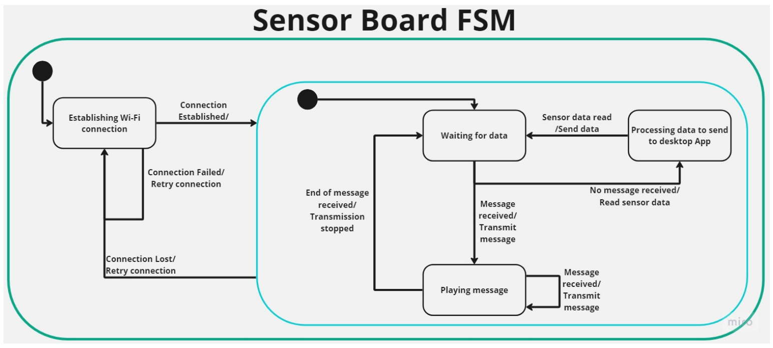
Sensor Board FSM
Establishing Wi-Fi connection
Connection Established/
Connection Failed/ Retry connection
Connection Lost/ Retry connection
Waiting for data
Sensor data read /Send data
Processing data to send to desktop App
No message received/ Read sensor data
End of message received/ Transmission stopped
Message received/ Transmit message
Playing message
Message received/ Transmit message
miro

Figure 14: Sensor Board state machine diagram

The Sensor Board Firmware interfaces with the MPU6050 (accelerometer, gyroscope and thermometer) module via I2C, with the MAX98357 Amplifier Module via I2S and with the MQ135 gas sensor via GPIO input, as can be seen in Figure 15. It was written in C++ using ArduinoIDE and runs on the ESP32 installed in the Vipper's head.

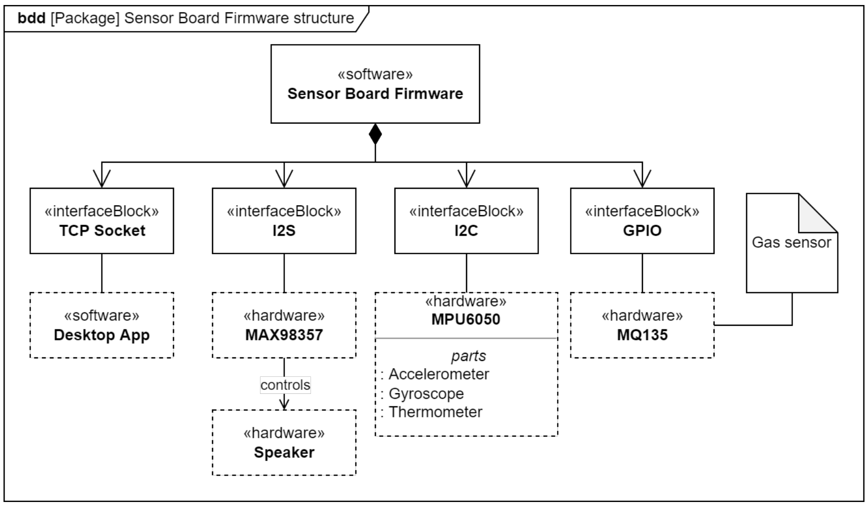

Figure 15: Sensor Board firmware block diagram

3.3.3 Desktop App

The Desktop App's main function is to serve as an interface that the Vipper's operator can use in order to control the robot's movement and read the incoming data. It is composed of two screens: the Mapping view and the Camera view, as shown in Figures 17 and 16, respectively. Both views allow the user to see the thermometer and gas presence data that is being received, hear the audio captured from the robot's head and control its forward and backwards movement. The main difference is the main component shown in the view - in the Mapping view, a 3D map of the Vipper's current trajectory is shown, while in the Camera view the operator can see what is directly in front of the robot. In the Camera view, the user can also send voice messages to be played in the head. A State Machine Diagram for the Desktop App can be seen in Figure 18.

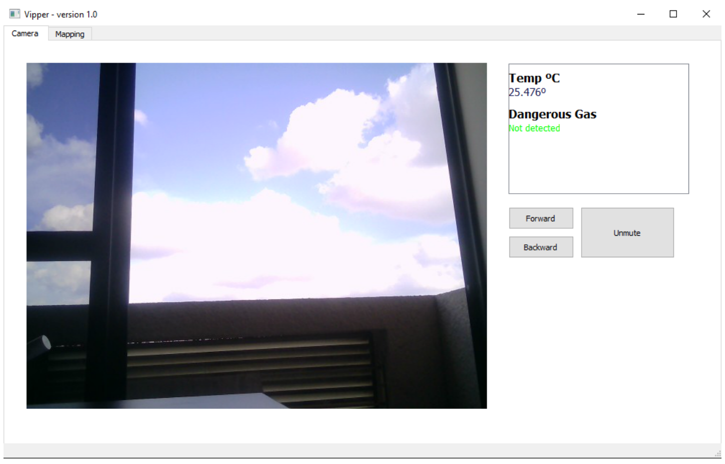

Figure 16: Desktop App's camera view

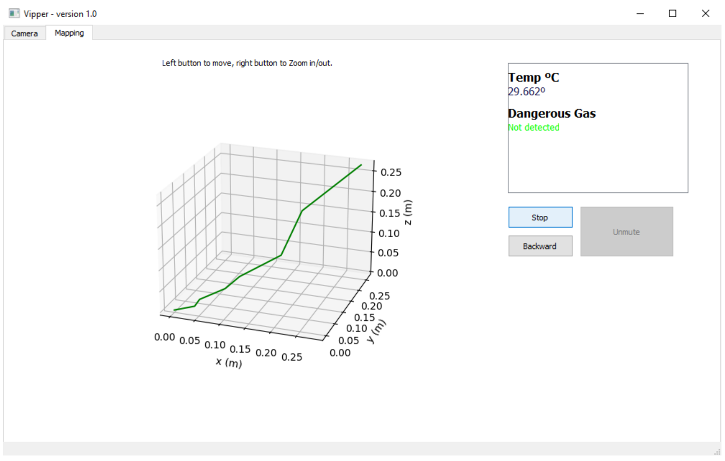

Figure 17: Desktop App's mapping view

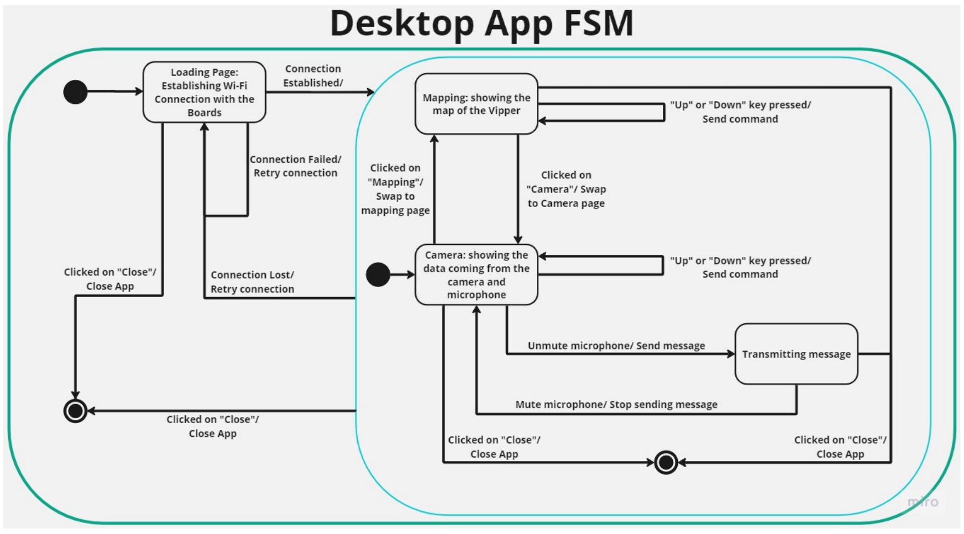
Desktop App FSM
Loading Page: Establishing Wi-Fi Connection with the Boards
Connection Established/
Connection Failed/ Retry connection
Clicked on "Close"/ Close App
Connection Lost/ Retry connection
Clicked on "Close"/ Close App
Mapping: showing the map of the Vipper
"Up" or "Down" key pressed/ Send command
Clicked on "Mapping"/ Swap to mapping page
Clicked on "Camera"/ Swap to Camera page
Camera: showing the data coming from the camera and microphone
"Up" or "Down" key pressed/ Send command
Unmute microphone/ Send message
Transmitting message
Mute microphone/ Stop sending message
Clicked on "Close"/ Close App
Clicked on "Close"/ Close App
miro

Figure 18: Desktop App's state machine

The app was written in Python using PyQT5[3]. It uses TCP sockets to communicate with the sensor and control boards via Wi-Fi. The mapping is drawn by estimating the distance traveled per click of the FORWARD button and drawing that path along some time while calculating the current orientation of the Vipper using data collected from its gyroscope. The mapping plot was implemented with MatplotLib[4]. All the audio processing was done using PyAudio[5].

4 Results

The results of the Vipper's execution were close to expected. All the requirements were fulfilled, with requirement **RF11** having some problems that will be discussed in the Difficulties subsection (Section 4.1).

In the end, the project successfully presented a functioning Vine-Robot, with the added functionalities specified, namely the environment being analyzed with temperature and gas sensors, together with the webcam, a two-way real time communication with the microphone and the speaker on the head, with a mapping which, although not with good accuracy, shows the approximate path of the Vipper, in three dimensions.

4.1 Difficulties faced

The biggest difficulties faced during the development of the project were related to mechanical assembly, audio data transmission and mapping.

[3]https://pypi.org/project/PyQt5/
[4]https://matplotlib.org
[5]https://pypi.org/project/PyAudio/

Starting with the mechanical assembly, the team had problems controlling the air pressure, mainly because of the air compressor acquired and the sealing of the recipient of the base. The first air compressor did not have enough pressure for the body to grow with the head, and with the base not being properly sealed, the robot would not move. Changing the mounting of the base with the same air compressor did not work, but after acquiring a more robust air compressor the robot started moving as expected.

The audio data transmission was harder than initially expected due to the project being changed during development. The original concept was to send pre-recorded messages with a MP3 module. This was later changed to making real-time voice transmissions. For this, the MAX98357A amplifier was used, and the team had some difficulties to process the data received from the Desktop App in the Sensor Board. The chosen solution was to send the data in small WAV files that encoded a second of voice recordings each.

The mapping was also harder than expected. In the beginning, the idea was to use gait tracking algorithms [6], adapting it for the Vipper's movement. This ended up not working for various reasons, such as the algorithm not being ideal for the robot's unpredictable movement and the precision of the accelerometer being too low. After other implementation attempts, the team considered the accelerometer data to be useless, because the movement of the Vipper was too slow and sporadic for it to measure any meaningful data.

In the end, the team opted for a simple approach that worked well, using the gyroscope data to calculate the orientation of the head at all times while considering that the robot goes forward for a fixed, average distance every time the body is unrolled. Figure 19 shows one of the tests made for the mapping, showing an approximate path, with the biggest difference being the distance. The real distance was 5 meters, with the mapped distance being approximately 6 meters.

[6] G. tracking with X-IMU, "https://x-io.co.uk/gait-tracking-with-ximu/."

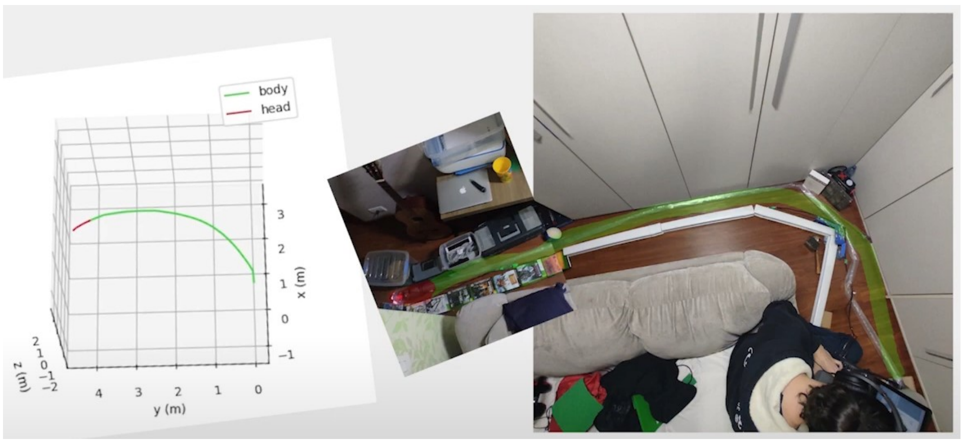

Figure 19: Comparison of the mapping with the real trajectory.

4.2 Budget

Table 1 shows the complete budget for the Vipper project. The *Other hardware Parts* entry refers for boards and jumpers.

Table 1: Complete budget of the Vipper project.

Material/component	Unit price (Reais)	Quantity	Total price (Reais)
12V Air compressor	133,90	2	267,8
12V DC motor	138,00	1	138,00
12V power source	119,90	1	119,90
3D printed parts	95,00	1	95,00
Polyethylene roll - 1m	0,63	100	63,00
Spider coupler 6,35/8 (mm)	48,52	1	48,52
10m usb cable extension	45,90	1	45,90
Air compressor regulator	39,90	1	39,90
Wooden board	34,90	1	34,90
Plastic pot	15,90	2	31,80
6,35mm diameter metal bar	24,90	1	24,90
Bearing 6,35/13/5 (mm)	11,85	1	11,85
Webcam	85,13	1	85,13
ESP32	34,20	2	68,40
MPU6050	14,00	1	14,00
MQ135	19,99	1	19,99
MAX98357A	31,90	1	31,90
Speaker	12,34	1	12,34
Other hardware Parts	50,00	1	50,00
Total			1203,23

4.3 Schedule

In the project schedule, the expected duration for the complete development of the project (after the decision of the project, starting development until the video recording and report writing) was 341,5 hours. The project had a 30% margin for error, making a total of 443,95 hours for the development of the project. In the end, the total time of development was 451,5 hours.

5 Conclusion

Most of the difficulties faced and time spent in this project were a result of either misconceptions about the functioning of the device or lack of knowledge about areas out of our scope, especially regarding the mechanical part. Notably, a substantial amount of time was spent on trying to properly seal the base vessel, making it an arduous job of repetition, test of materials and whatnot. This also led to subsequential obstacles in the reading of the accelerometers, since the body ended up being too slow, adding too much noise to the measurement.

Among the issues worth mentioning, the most prominent was the lack of courses about integrated hardware and software project planning, with the closest course currently in the curriculum being Embedded Systems, which still puts more emphasis on development rather than planning, along with the incoherency between the time needed to meet the deadlines and the available time to actualize them, taking into consideration other courses and an internship.

Looking past the problems and difficulties, the project was very fulfilling. The end result was satisfactory for the scale proposed in the Integration Workshop 3 course and for the planned tests, with all the requirements being met, and the team is satisfied with it.

References

[1] J. R. Balbin, R. G. Garcia, F. L. Valiente, Y. C. Hirota, G. B. Lasay, S. M. M. Lingad, and M. I. C. Villafranca, “Detection and localization for buried and alive human body after mudslides using pulse sensor and force sensing resistor with xbee technology and global positioning system to support rescue operations,” in *2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, pp. 1–6, 2019.

[2] A. De Leo, V. Petrini, P. Russo, L. Scalise, V. Di Mattia, V. M. Primiani, and G. Cerri, “An em modeling for rescue system design of buried people,” *International Journal of Antennas and Propagation*, vol. 2015, pp. 1–7, 2015.

[3] A. H. Reddy, B. Kalyan, and C. S. Murthy, “Mine rescue robot system – a review,” *Procedia Earth and Planetary Science*, vol. 11, pp. 457–462, 2015. Global Challenges, Policy Framework & Sustainable Development for Mining of Mineral and Fossil Energy Resources (GCPF:2015–20).

[4] T. Kamegawa, T. Yarnasaki, H. Igarashi, and F. Matsuno, “Development of the snake-like rescue robot" kohga",” in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA’04. 2004*, vol. 5, pp. 5081–5086, IEEE, 2004.

[5] L. H. Blumenschein, M. M. Coad, D. A. Haggerty, A. M. Okamura, and E. W. Hawkes, “Design, modeling, control, and application of everting vine robots,” *Frontiers in Robotics and AI*, vol. 7, p. 548266, 2020.

[6] R. Manoj, A. H, A. Asokan, A. A. R, K. T. Abraham, and D. G, “Upper gastrointestinal endoscopic imaging using vine robots,” in *2022 IEEE 19th India Council International Conference (INDICON)*, pp. 1–6, 2022.