Team Description Paper: Momentum Robotics

Giulia E. O. Castro, Guilherme Senday, Pedro S. Chen, Rafael K. Sakamoto, Raphael F. Correia, Vitor S. Figueiredo

***Abstract*— This paper presents a description of the process of programming, assembly and the strategies used to make our team's robot, which will be used during the Latin American Robotics Competition of 2018 in the Rescue Maze category. The robot is moved using four Mecanum wheels, each rotated by a Dynamixel AX-18A Servo. It identifies the heated tiles using 4 Melexis Temperature Sensors, one on each side of the robot. The source code was written in C++ and is processed in an Arduino Mega board. The algorithms used - Breadth-First Search and Wall-Follower - combined with the RPLidar, ensure that the robot will visit all tiles possible.**

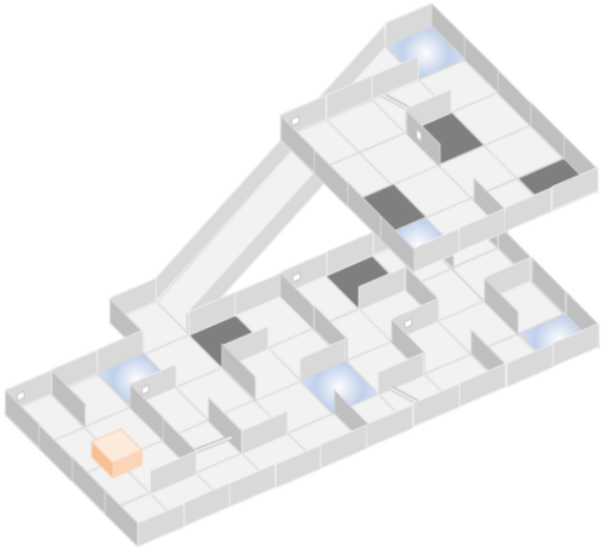

Fig. 1. Arena

I. INTRODUCTION

Mankind has always searched for better ways to solve tasks and problems that are presented to it. We first had stone tools and fire to conquer nature and now the advancements made in the robotics have opened the possility of helping people in a variety of situations.

The Rescue Maze competition[1] proposes a simulated environment of disaster with an irregular landscape, through which an autonomous robot navigates and is tasked with locating victims that are scattered around the map.

This challenge was created with the purpose of turning the attention of young generations to this growing technological field with the intention to create something that somehow serves a purpose to mankind. With this in mind, our team, Momentum Robotics, has decided to concentrate our efforts into the development of robots.

By publishing this paper, the team expects to encourage even more students to take an interest in robotics-related competitions and in developing new technologies. We also wish that this paper can be of help and serve as a reference to any competitors in future Rescue Maze tournaments.

II. THE COMPETITION

The arena (*Fig 1*), represents a disaster environment, which is separated in two areas of different size and height that are connected by a ramp. The floor can be covered by white, silver or black tiles, each symbolizing respectively, neutral area, safe points, and danger zones.The maze is limited and defined by walls, which can be linear (areas in green on *Fig 2*, these walls lead to the entry/exit point of the arena) or floating (areas in red on *Fig 2*, victims on these walls are harder to find).

The victims are represented by heated sources approximately 7 centimeters above the floor, with an average temperature of 34 degrees Celsius. Each victim shall receive one rescue kit.

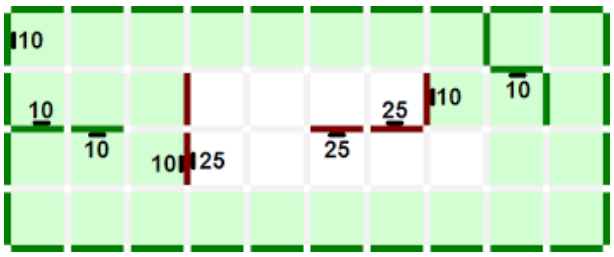

Fig. 2. Victims Scheme

III. THE ROBOT

The general structure of the robot (*Fig. 3 and 4*) was made using different materials, including laser-cut wood and acrylic boards as well as many 3D-printed pieces for the robot's main structure. Its components were chosen with fast and agile movements in mind, having the type of wheel chosen as the main differential for the accomplishment of the task above.

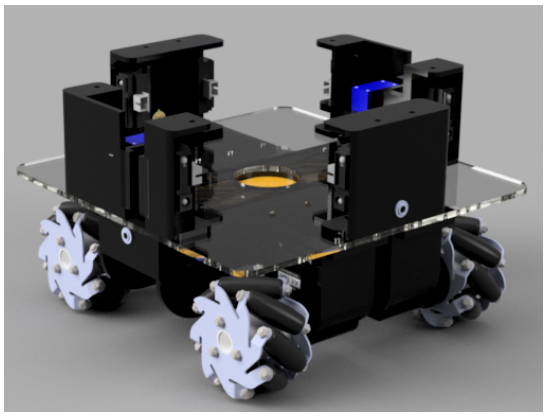

Fig. 3. Robot Bottom Layer 3D Render

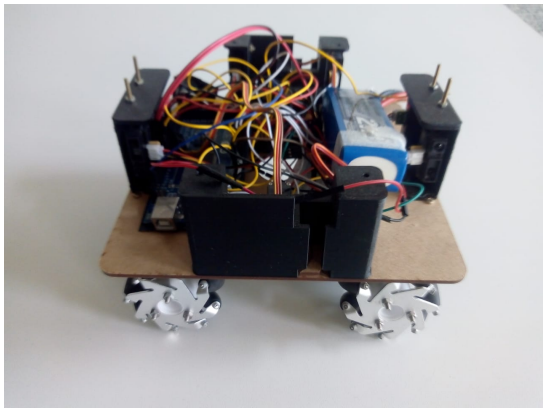

Fig. 4. Actual Robot Bottom Layer Assembled

A. Arduino Mega 2560

The Arduino Mega 2560 [2] (Fig. 5) is one of the largest and most powerful boards in the classic Arduino family. Its internal storage is 8 times larger than the more popular Uno board, having almost triple the analog inputs, thus making itself the most appropriate Arduino microcontroller for the task.

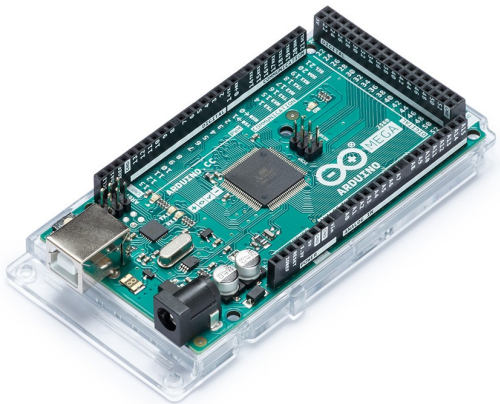

Fig. 5. Arduino Mega 2560

B. Dynamixel AX-18A Smart Serial Servo

We chose this motor mainly because of its high RPM and torque, but also due to its good precision and brand reputation in the smart-servo market.

C. Mecanum Wheels

The Mecanum wheel (*Fig 6*) is a type of wheel which consists on several rollers attached to its cicunference, revolving the main axle with a 45 degree angle between them. When four are used in combination, each with its own motor, the resulting force vector can be adjusted to virtually any direction, opening up the possibility for the robot to go anywhere on an X without needing to rotate itself. Although some energy will be wasted by the cancelling vectors, the maneuverability allowed by this wheel arrangement is extremely useful for small corrections, thus justifying its pick.

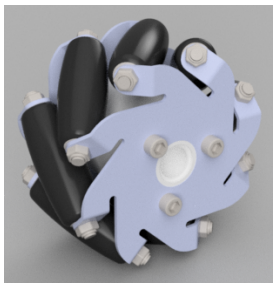

Fig. 6. Mecanum wheel 3D Render

D. Sharp Analog Distance Sensor

These infrared distance sensors have a 4cm to 30cm range, which is ideal for the 30cm x 30cm tiles. There are two sensors facing each side of the robot, one on each corner. A 3D model of a support strucure can be seen on *Fig 7*, having the rescue kit dropper mechanism integrated with it. These close range distance sensors are vital for aligning with the current tile.

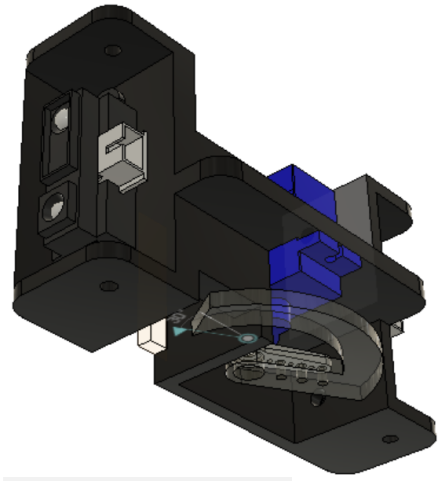

Fig. 7. Support structure for two distance sensors and kit dropper mechanism

E. TCS Reflectance Sensor

This color sensor is used to differentiate silver, black and white tiles. It is located beneath the battery compartment, in the center of the robot; this being a prime placement due to low foreign light interference.

F. Melexis Temperature Sensor

This infrared temperature sensor (*Fig 8*) will be used to identify heated victims. Four of them are present on the robot, one on each side, in order to promptly recognize the targets.

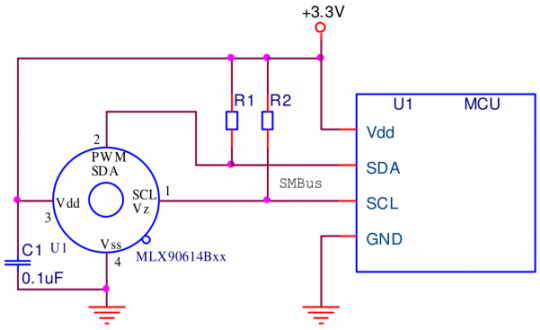

Fig. 8. Temperature Sensor Circuit Schematic

G. LCD Display 20x4

To attend the need of having some feedback from the program, this LCD screen (Fig 9) is essential to the project. It can present us any written information, for instance: sensor values, battery level, or map information; depending on what the programmer considered to be fundamental at a given moment.

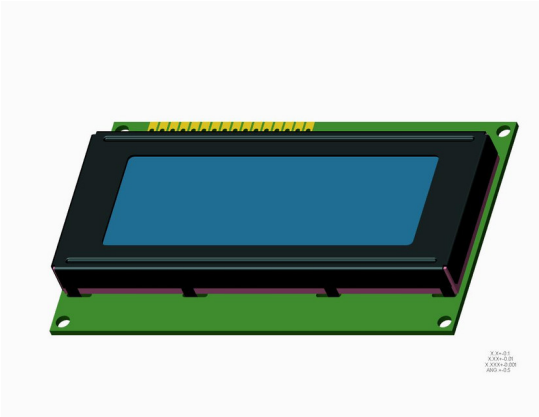

Fig. 9. LCD Display

H. Kit Dropper Mechanism

The simple rescue kit ejecting mechanism (*Fig 7*) consists of an extended servo horn, which controls the output of rescue kits, each being a parallelepiped of dimensions 1.5cm X 1.5cm X 0.5cm. The robot has two of this structure, each on opposite sides, aiming for a faster kit deployment.

Fig. 10. RPLidar A2

I. RPLiDAR A2

The LiDAR (Light Detection And Range) [3] (*Fig. 10*) is a remote optical detection technology that measures properties of the reflected light in order to obtain the distance to a certain object. With this sensor, the robot is able to precisely measure long distances all around the robot, being an excellent tool for mapping and navigation.

J. Circuit Board

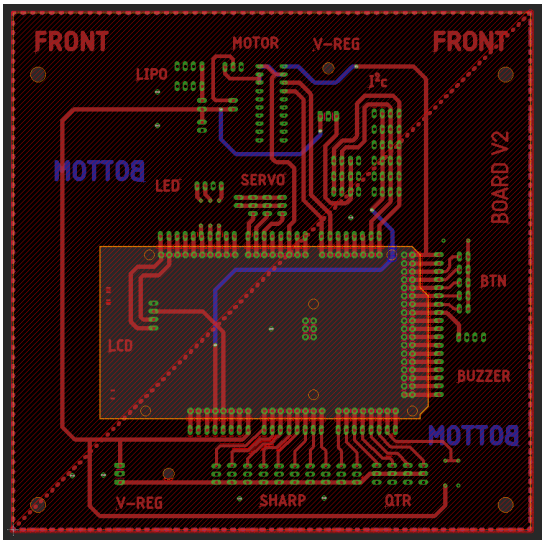

Fig. 11. Board file

Designed by our team using the Eagle CAD program, and manufactured using the laser etching technique, the circuit board (Fig. *11*), although not being of direct importance to the robot's operation, is very influential on cable management and overall organization, thus being practical to have one.

IV. THE PROGRAM

The program used was written in the Arduino IDE with its own simplified C++ language, making use of already existing and our own libraries and functions. It consists of two main layers of functions (*Fig. 12*).

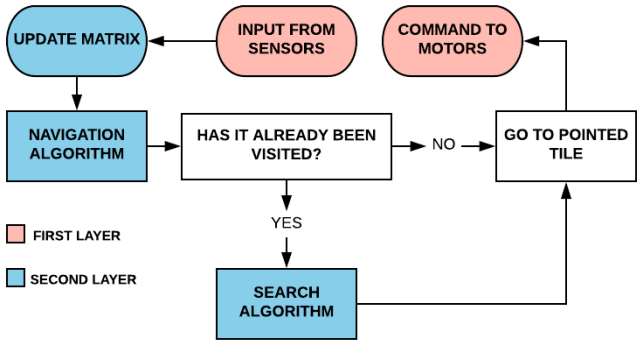

Fig. 12. Basic logic of the use of each algorithm

A. First Layer : Hardware Interaction

The first layer of the program is the most superficial one, since its duties are fairly simple and dismiss heavy logic, being responsible for interacting directly with the robot's *hardware*. For instance, its functions control the output for motors and servos as well as receive input from all sensors, having this data interpreted - or command received - by the other program tier.

B. Second Layer : Logic and Strategy

The second layer of the program is responsible for the data processing, that is, to receive inputs from sensors and send commands to the robot actuators according to the strategy used. The robot stores all the main information about the maze in a 3-dimensional matrix, including wall positions, tile colours, among others. The logic adopted is formed by two main algorithms, which together decide the robot's next movement.

1) Navigation Algorithm: The navigation algorithm is the first one approached when deciding where to go next.

It is a wall follower algorithm, more specifically a right-hand rule. The ususal logic would be a preference order:

- RIGHT
- FORWARD
- LEFT
- BACKWARD

The program checks the availability of each side according to the order above, heading to the fisrt one possible.

This algorithm could work in some environments, however in the case of floating tiles, for instance, the algorithm would fail, thus the implementation of the second is justified.

2) Search Algorithm: The search algorithm is used when the tile pointed by the *Navigation Algorithm* has already been visited by the robot, indicating the nearest unvisited one, or the initial position in case of having visited all tiles earlier.

It is a Breadth-First Search Algorithm (BFS) [4], which is used for traversing a graph data structure, by exploring the neighbors of the current node prior to moving on to nodes on another depth level (*Fig.13*).

The use of both algorithms ensures the robot will be able to navigate through the entire labyrinth, including before impossible floating victims. Moreover, it makes the robot able to find the shortest distance path to its initial position.

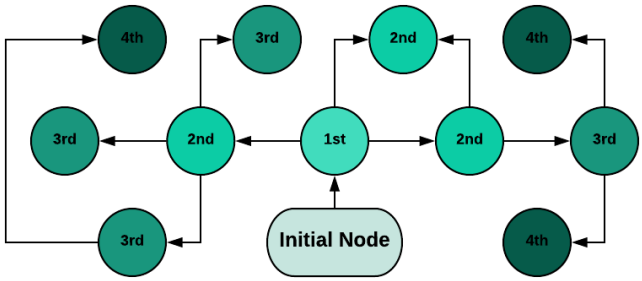

Fig. 13. Example of a BFS use, representing different depths in the search with darker greens

V. CONCLUSIONS

Although previous teams from our school have already taken part on this challenge, this competition took our group out of the confort zone as we had to build a new robot and algorithms from scratch to complete tasks that we weren't used to, making this a very useful experience on a academic sense, because of all the new programs we learned throughout the year. Furthermore, this competition has taught us alot on teamwork and represents a big opportunity for our future as students.

ACKNOWLEDGMENT

We are grateful for the opportunity proportionated to us by the organisers of the event and also for Colegio Etapa for supporting and encouraging our team to take part on this competition. We would also like to thank our mentors Raphael Correia and Henrique Martinez for providing the guidance and the companionship we needed, and our families and friends for staying along our side through this unusual journey.

REFERENCES

[1] RoboCupJunior Rescue Maze 2018 Rules. Available on: *http://www.cbrobotica.org/wp-content/uploads/rescue_maze_2018.pdf*
[2] Arduino Mega 2560. Available on: *https://store.arduino.cc/arduino-mega-2560-rev3*
[3] Slamtec RPLidar A2. Available on: *http://www.slamtec.com/en/lidar*
[4] Breadth-First Search Algorithm. Available on: *https://en.wikipedia.org/wiki/Breadth-first_search*