RoboCupJunior Rescue New Simulation 2021

Team Description Paper

Titán

## Abstract

1. Our robot is capable of generating a representation of the competition map and finding an optimal sequence of movements for its navigation. Our code has an organized architecture and we have developed a toolset to make the programming at high level easier. All our work is documented on GitHub.

## Introduction

## Team

* + 1. Our team is composed by Máximo Cansino, Lucas Flores and Alejandro de Ugarriza. Alejandro participated with another team on the first simulated demonstration. Máximo and Alejandro together with another teammate were the winners of the Roboliga Simulada (Simulated Roboleague) in Argentina, which used the Erebus platform. Lucas is a new addition to the team. Máximo is in charge of the victim detection, Alejandro of the mapping and navigation and Lucas of the grid for submission to finalize the game.

## Aim, strategy, and overall plan

* + 1. Our aim is to make a robot capable of navigating intelligently, analyse and map any type of environment. Calculate optimal trajectories using complex algorithms for an efficient navigation. Detect victims and hazard maps using simple strategies of image processing. Make a more organized code, responsible, robust and documented via GitHub.
    2. Our strategy as a team consists mainly on the division and assignation of roles for each of the members, coordinating via GitHub and virtual meetings once a week. Make a new architecture for the code and then move and modify pieces of our prior code there, apart from creating new, more robust and efficient solutions.
    3. We started by designing a scheme for the new architecture, we defined the classes in the code, then we started moving pieces from the prior code, optimizing and adapting them to the changes required by the new platform. We continued defining the workings of the sensors and basic actuators, then the precision movements on the map, we implemented the mapping of the labyrinth, then the pathfinding algorithms, and then we integrated everything and defined the navigation logic. In parallel we developed a victim and hazard map detection system.

## Technical progress

## Navigation

* + 1. Our robot navigates by moving in straight lines from vortex to vortex, a vortex being considered on the scale of half tiles. Is capable of executing pre-programmed movement sequences in key situations.

Our navigation system is integrated by two sections: Analysis and Low-Level Movements:

* Analysis is in charge of planning the long-term navigation for travelling more efficiently through the map and doesn’t have any direct connection with the actuators. It uses, between other things, breadth first search for finding the best possible tile to move in the virtual representation of the map. Once found it uses A Star to find the shortest path to reach that position. It´s incapsulated on the Analysis class.
* The part in charge of Low-Level movements has the task of controlling the actuators directly to follow the path provided by Analysis, or to get the robot out of an emergency that would require immediate and direct control. It´s distributed between the Robot Layer and Abstraction Layer classes.
  + 1. Innovative strategies:
* We divided the navigation in short-term and long-term
* We used the GPS to obtain the orientation of the robot via the differences between positions. This allows us to define a reference point and maintain it, due to the inability of the gyroscope to register these changes in orientation when the robot is teleported from one position to another.
  + 1. What´s left to do is to make the robot move according to the long-term movement planification. In the future we plan to implement a more complex system to find the optimal tiles to move to and optimize the short-term movement to achieve more speed and fluidity. Thinking about the very long-term, we might introduce more complex AI tools, and maybe even machine learning.

## Detection

* + 1. Our program contains a recognition system based on different states controlled by the camera panel, which analyses and detects different types of objects having in to account their colour, shape and area.

First, the camera image is divided in three main filters: “Red Panel”, “White panel” and “Yellow panel”. Each one of these filters is configured to take a specific colour according to their hue and saturation. These values were defined based on multiple tests in different environments with the intention for them to only work on certain objectives with the least noise possible. After this, it´s capable of sending the angle and size in pixels of the victims or hazard maps. Once the camera is close to the victim, the image perspective is calibrated and it´s classified according to its shape, area and colour.

* + 1. Our method for identifying and classifying victims and hazard maps is not based on a complex algorithm, but simply in applying filters, which makes it easier to implement.
    2. In the future we have planned to implement a matrix comparison and pattern detection system to optimize the program in time and computational cost.

## Mapping

* + 1. To map we used the LIDAR sensor, the camera and the colour sensor. The data coming from these is processed and stored in a grid. To process the LIDAR data, it´s compared to different “templates” (with the shape of walls, obstacles, etc). We use the camera to identify the victim and hazard map types. With the colour sensor we identify the different floor types.

Each tile (taking as reference the small tiles), vortex and wall is represented in the grid as objects with different properties. The grid has the capability to expand dynamically as the map grows in size. The obstacle, curved wall and tiles with the colour sensor detection doesn’t work yet.

* + 1. We used a template system to process the LIDAR data. To prevent noise in the LIDAR data, a que of point clouds was implemented to filter random and inconsistent detections.
    2. For the obstacle and curved wall detection we can use the template system already implemented. About the detection of tiles of different types, we don’t have a system to map the entire tile, it can only do one half tile at a time.

## Hardware

Our choices in hardware are based in what we thought would be the most convenient to resolve the task at hand while keeping ourselves below the total budget.

## LIDAR: We chose this sensor instead of the distance sensors due to it having advantages in the detection of curved walls, detection distance and sampling angles. We also chose it because we considered that it would be interesting to try something new, with which none of us had ever worked before. It´s located in the top and centre of the robot, to take the maximum advantage of the full 360 degrees of detections.

## Two cameras: These cameras were chosen with the objective of detecting and identifying victims and hazard maps. At the start we needed 3 cameras, but due to the budget we could only buy two. In response to this problem we located them strategically on the sides to not loose field of view.

## Gyroscope: we used it with the objective to have a good measurement of the orientation of the robot. It’s in the centre of the robot to get more accurate measurements.

## GPS: It´s used to obtain the coordinates of where the robot its located, and to orient ourselves in the case of teleportation. It´s located in the centre to minimize noise.

## Colour sensor: It´s used to detect the different colours of the floor to know if we changed zone or if there´s an obstacle in front of the robot. We put it on the front and bottom of the robot pointing directly to the floor to get a good measurement.

## Program flow and sequential code execution:

* + 1. We developed a system to be able to control the flow of our program in the form of a state machine, apart from a way to insert pieces of sequential code to the program, where things like delays can be used without affecting other parts of the code that need to run continuously. This allows us to run tests and predetermined sequences very easily.

## Performance evaluation

* 1. We can observe that our robot is capable of navigating through the entire maze and has no problem with half-walls. We are experiencing some problems with the rotation detection; we believe it has to do mostly with the physics errors in the simulation. This causes errors in the mapping, and sometimes makes the robot do some unnecessary turns. It also turns a little bit abruptly. The victim detection is ready but not integrated to the final program yet, so we don’t get a lot of points, but we are more than happy that it can map and navigate, even with those physics errors.

## Conclusion

* 1. To conclude we can say that we improved our program and teamwork a lot, having into account the experience of the past competition, and, even though we have a lot to do, we’re going in a good path and we have a good trajectory going in to the future.

## Appendix

Our GitHub: <https://github.com/iita-robotica/rescate_laberinto/tree/master/Competencias/Robocup_2021>

A Star Algorithm: <https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2>

Breath First Search Algorithm: <https://medium.com/@yasufumy/algorithm-breadth-first-search-408297a075c9>

## Reference