DEVOS Xavier
LAFRASSE Cédric

RICM4

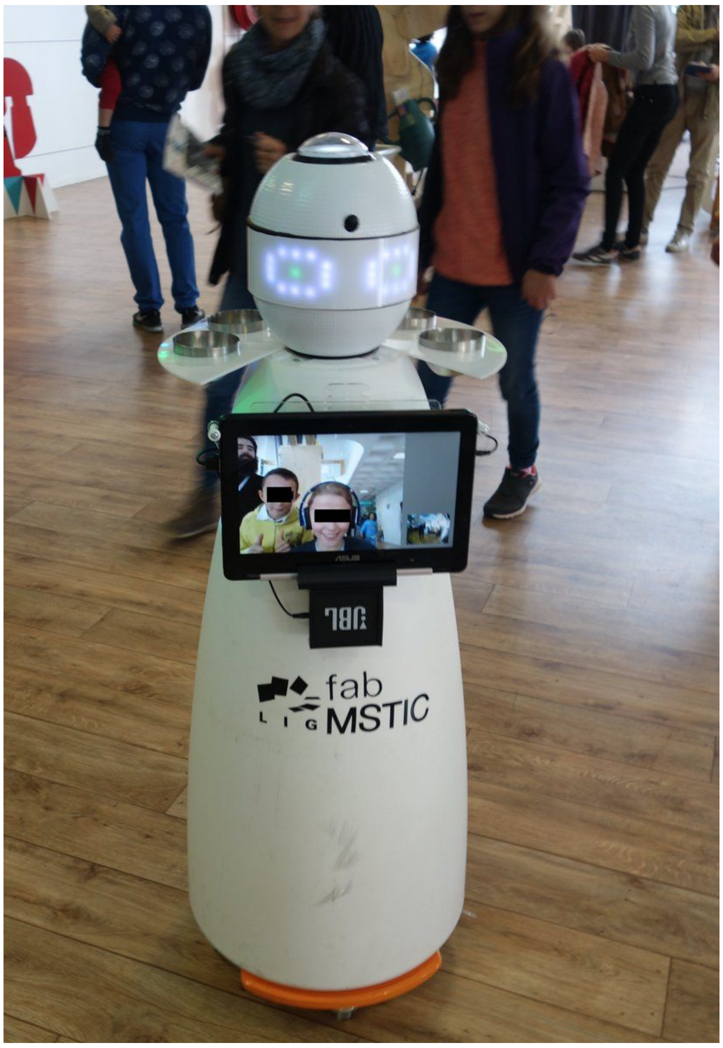
RobAIR

Project report

Introduction

What is RobAIR ?

RobAIR is initially a telepresence robot platform based on Grenoble. Intended for teaching robotics, many projects have been carried out on these robots.Telepresence robots are booming, and have more and more functionality. These robots could well be used by students who cannot come to class, to keep the elderly company…



RobAIR at the Makerfaire event in Grenoble

More and more large groups are developing their own telepresence robots, to follow user demand. This system is becoming more and more democratic, allowing a faster technological improvement due to competition.

What is the purpose of our project?

Our project is for a completely different use than the usual use of telepresence robots.
We wanted to give RobAIR the ability to become a guide. Indeed, he must be able to help newcomers in a building to get to the room they are looking for. And it is either thanks to a course that robair will display thanks to his tablet, or even by allowing him to move to the room.

Conception

Project organization

As we reuse functional code, we did not have to make any particular choice in the general architecture of the project. However, the user part (RobAIR side) had to be redesigned to fit our project.
The main choices were more algorithmic than organizational.
We chose to use JavaScript for the algorithmic part. We first chose C++ but we immediately faced the problem of data exchange between JavaScript -> C++ -> JavaScript -> RobAIR -> JavaScript, etc.... This data was fundamental for the use of our algorithms and we therefore preferred to do everything with JavaScript.

Technologies used

In order to control RobAIR, we use ROS technology. Widely used in the field of robotics, it works with message publications on topics. ROS is language neutral, so we chose JavaScript to publish the messages.

RobAIR then uses a Node.js server. Node.js allows among other things the event system.
It adapts perfectly to the ROS topic system. When a message is published on a topic, we can know it and thus carry out the necessary actions.
Our objective was to use this operation to retrieve the messages provided by the LiDar to manage the collision.

Development and functions

As said before, most of the work has been done in JavaScript. First of all, we developed algorithms to manage and transform tables.
These allow (from a table of 0 and 1, 0 obstacle and 1 free zone) to create a label table, an adjacency matrix, a displacement table…

The basic chart representing the map was generated by hand from an evacuation plan. We chose to work on half of the ground floor of Polytech Grenoble.
In this table, only the ready for input/obstacles fields are defined. The locations in front of which RobAIR must stop to get to the right room are then added in a hashmap. Then the shortest path is calculated from the adjacency table and the label table using Dijkstra.

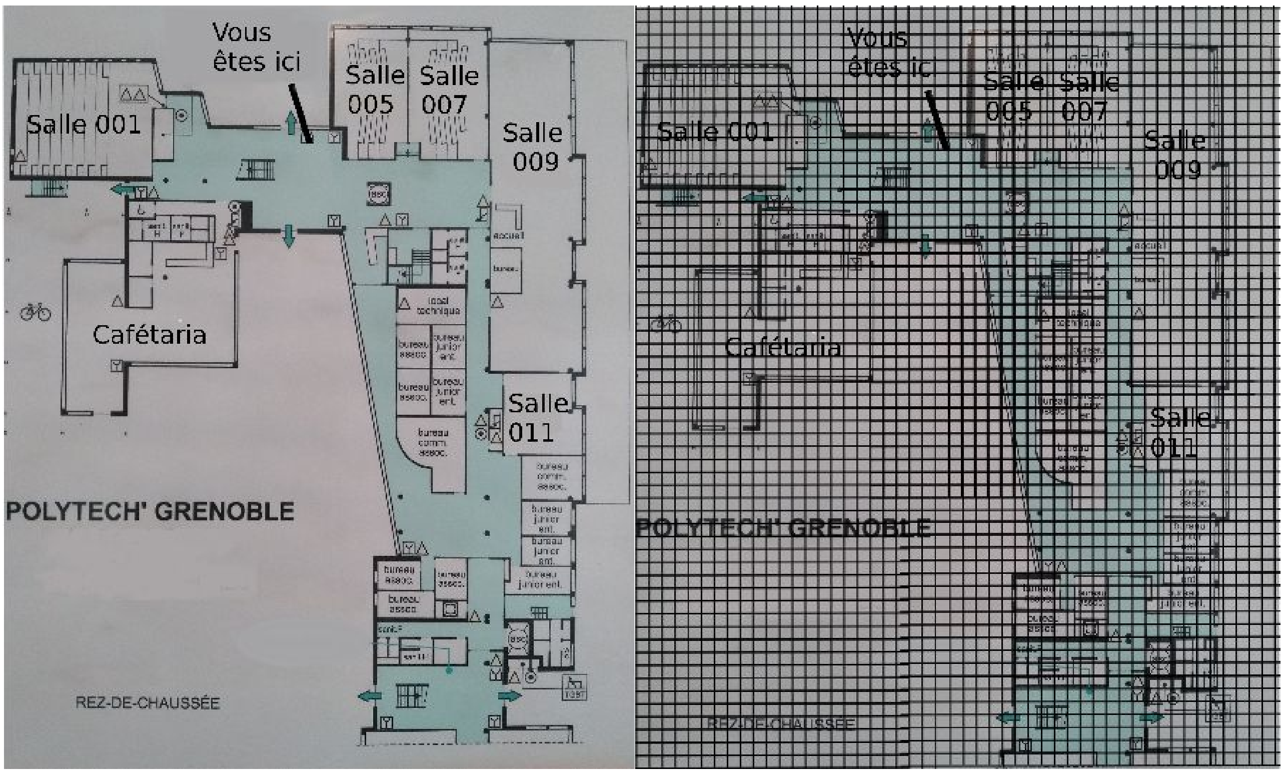
The calculated path is displayed using the HTML Canvas element. The map is displayed when the interface is initialized. After the user selects the room using a form, the path is calculated to the room and then displayed over the map.

Production

Functional parts

The project has the current state allows a user to choose, using the interface on the RobAIR tablet, the room to which he wishes to be guided. Then the shortest path is generated and displayed on the building plan.

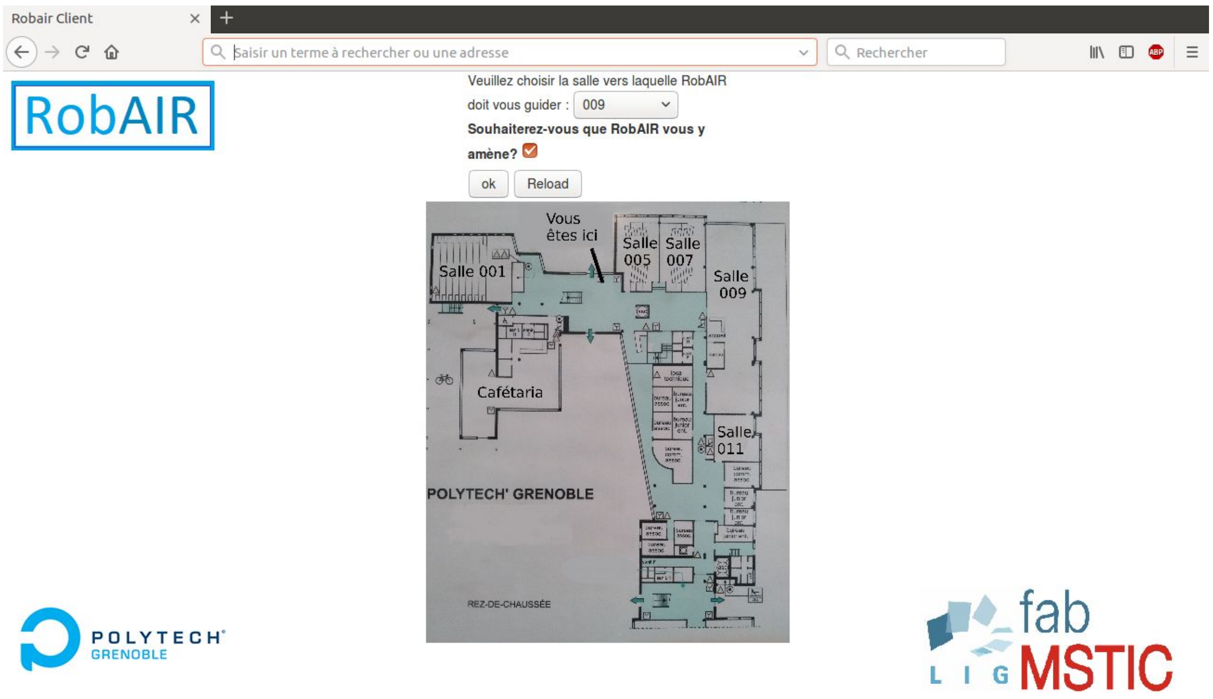
Currently, all the map present in the project (1st floor of Polytech), was generated by hand using the evacuation plan, on which we created a grid.



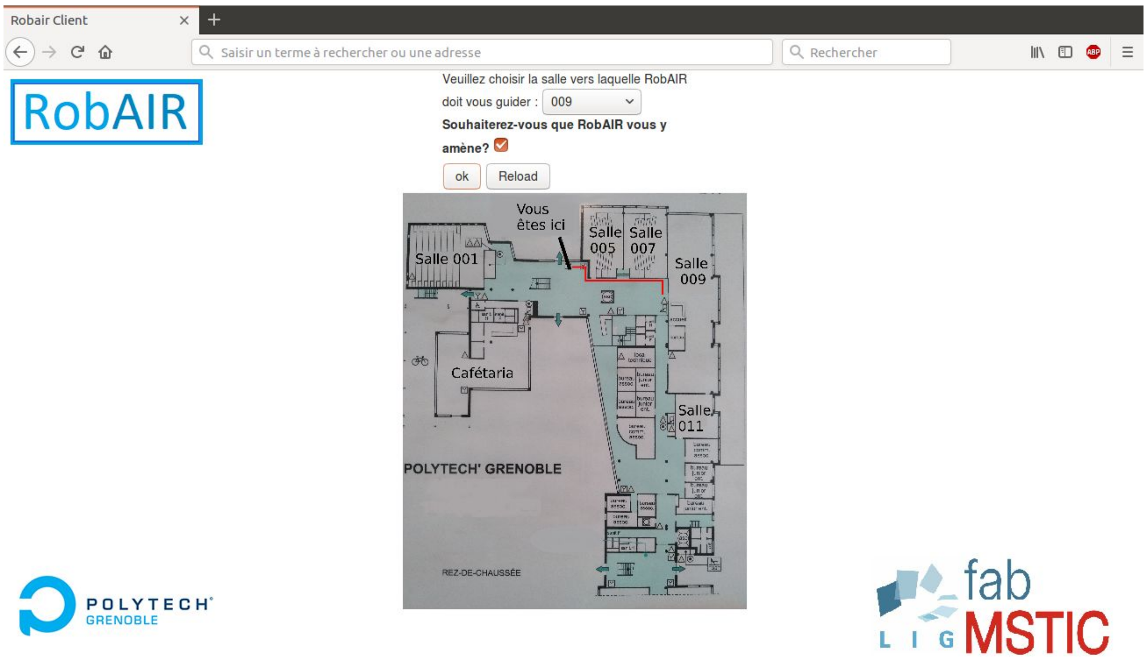
Setting up the grid on the plan

The work is tedious and can cause errors during field generation, leading to guidance errors.
Moreover, in the case of a larger building, with more room, more floors... the fact of generating the plan by hand is not possible.

Finally, the user to choose whether or not to be guided by RobAIR. If necessary, RobAIR moves to the room, showing the way, then once he arrives at the room, he turns around and then returns to his starting position (building entrance) in order to guide newcomers into the building.



Customer interface



Customer interface with the path to the room 009

Potential improvement

RobAIR could for example be able, before its installation in a building, to establish the plan of the building in which it is located (using SLAM-Gmapping for example). External help will still be needed to name the rooms and define an end position for each of them.
Next, a more "smart" guidance system with obstacle avoidance, which can be set up using the Smooth nearness diagram navigation (SND navigation) package.

Finally, one could imagine a voice command, giving for example the name of the room to which one wishes to be guided.

Conclusion

This project can be developed on a very large scale, for example in airports, railway stations, public buildings.
Nevertheless, we lose the main goal of RobAIR, which is basically a telepresence robot.
But demand in the robotics aid community is currently booming and we can already find robot guides (like the LG robot guide project for the South Korean Olympics: https://www.theverge.com/2017/7/21/16007680/lg-airport-robot-cleaning-guide-south-korea-incheon).

We found this project very interesting. We discovered JavaScript, ROS. It was our first project in which we took up work we had produced years before with the aim of developing and creating new functionalities.