MIR Cleaner

Filipe André Seabra Gonçalves 98083
filipeg@ua.pt

[1] Universidade de Aveiro
Robótica Móvel
[2] Nuno Lau nunolau@ua.pt
[3] Vitor Santos vitor@ua.pt

Abstract. A robot is a computer-programmable machine capable of carrying out a complex series of actions automatically. Its applications vary based on the field of action or even the program it executes. Contrary to belief, robots are used on a daily basis, even if the owners don't realise it: it's in our kitchens (e.g., dishwashers), in our laundry (e.g., washing machines), or even in our house cleaning (e.g., roombas). These "simple" robots, easier to create compared to a humanoid robot, are nonetheless complex to program and manage and require a large amount of time and resources to complete. This work discusses the process of creating a cleaning robot.

Keywords: ROS · Mapping · Navigation

1 Introduction

The goal of this project is to create two distinct agents: one that will map the environment completely in the shortest amount of time, and another that will use a known map to clean the environment. Each agent is implemented in a specific ROS package.

This report will explain the approaches, methods and algorithms used to complete the tasks, as well as show the program and launch files of the ROS packages.

2 First Agent - MIR Mapper

The MIR Mapper's first and only objective is to map an unknown environment in the minimum time possible. To accomplish it, it must use some sort of navigation algorithm to efficiently navigate through the environment and save the map it slowly builds with its sensors using another algorithm.

The robot used, MIR, has two scan sensors, one in the front and one in the rear of the robot. While using two scans sensors is more efficient, it is also more complex to map the two sensory data it receives into a single map. As such, the MIR Mapper uses a single laser scan which is composed of the both sensors at the same time, which makes it easier to read the sensory data.

As for the exploration, the MIR Mapper uses a ROS package called explore_lite [3] to do a greedy frontier-based exploration [4]. In frontier based algorithms robots detect frontiers, which are boundaries between the explored and unexplored regions, and navigate towards the most promising frontier to complete the exploration. This process continues until no frontier cells are left.

Finally, to map the environment, the MIR Mapper uses GMapping [1]. This SLAM algorithm is a highly efficient particle filter to learn grid maps from laser range data. The mapping of the environment occurs while the robot is exploring the environment, which can be visible in the Rviz interface.

The Fig. 1 shows how the MIR Mapper navigates through the environment using the explore_lite ROS package, and maps the unknown environment using the slam mapping.

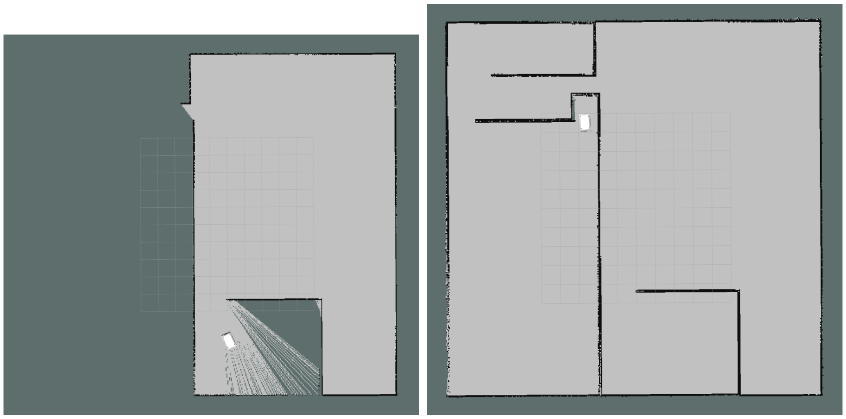

Fig. 1. Process of the MIR Mapper exploration and mapping

To run the agent we simply have to build the package and then launch the agent using its launch file. A launch file provide a convenient way to start up multiple nodes as well as setting up parameters for each node. The launch file is comprised of the following nodes:

- MIR Gazebo - launches the gazebo configuration for the assignment
- Move Base - launches the robot configuration
- Laser Scan Multi Merger - launches the merger for the forward and rear scans in the robot
- SLAM Mapping - launches the algorithm to map the environment
- Rviz - launches the rviz configuration for the assignment
- Explore - launches the explore_lite node for the exploration

Its important to mention that the Rviz node receives as argument the configuration file, which already contains the robot model and many other configurations to see the robot explore and map in a better way. The GMapping node remaps the subsccriber to listen to the scan_multi topic instead of the scan topic, to implement the dual sensor data transmission, instead of a single one.

3 Second Agent - MIR Cleaner

The MIR Cleaner's objective is to clean the known environment as quickly as possible, until a certain percentage of the room is cleaned. For that, it must use an algorithm to clean in the most efficient way.

The approach used is to simply find a corner of the map and zig zag the room until the percentage of cleaned room is satisfactory. The zig zag made by the robot will start in the right bottom corner of the map and will make its way up and down, ever so slightly to the left of the room. The robot will drive a number of predefined cells per step up or down, and whenever it reaches the end of the room, it will drive left exactly the robot's cleaner width, 0.64m.

To identify how much of the room it needs to clean, we can easily find the total number of cells the robot can drive on from the map and reduce that amount to the percentage desired. To calculate the number of cells the robot cleans, it must also know some important information about the map: it's resolution, width and height. Using its cleaner width, 0.64m, and knowing the resolution of the map, we can deduce the following formula:

$$cells\_cleaned\_step = (robot\_width * map\_resolution) * cells\_height\_step \tag{1}$$

This formula is nothing more than a simple area calculation, where the cells_height_step is the height and (robot_width * map_resolution) is the width. If we say that for each step the robot will drive cells_height_step cells, then using the resolution of the map we can know that the number of cells in width the robot will clean per step is (robot_width * map_resolution). The following figure, Fig. 2, is the representation of the formula explained.

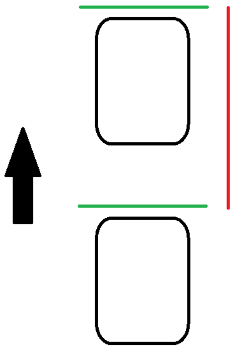

Fig. 2. Cells per robot step - Visualization

This is a simple approach but with some problems in itself. While it will clean the room in a somewhat efficient way using 100% of the usable room to clean, if a wall is placed in the middle of the room, then the robot will evade the said wall and continue its predefined path, even if its inefficient. To improve on this we can simply divide the map into different rooms, in which the robot can simply clean a room and go to the next, until the percentage of map cleaned is satisfactory. For this solution the robot should use a room segmentation like the ipa_room_segmentation ROS package [2], which divides the map into rooms. The algorithm that would suit best the robot would be the ground truth labeling.

To run the agent we have to build the package and then launch the agent using its launch file. The launch file is comprised of the following nodes:

- Map Server - launches the a server with the map built by the first agent
- MIR Gazebo - launches the gazebo configuration for the assignment
- AMCL - launches the move base localization
- Move Base - launches the robot configuration
- Laser Scan Multi Merger - launches the merger for the forward and rear scans in the robot
- Rviz - launches the rviz configuration for the assignment
- MIR Cleaner - launches the MIR Cleaner node with the desired percentage of the room cleaned as a parameter

4 Conclusion

In conclusion, the robot uses an exploration algorithm to navigate through the environment, a SLAM mapping algorithm to map the environment and, lastly, a simple zig zag algorithm to clean the room. As such, all the objectives of the assignment have been completed.

For more information, this assignment was made using the a github repository: `https://github.com/FlipGoncalves/MIRCleaner`.

References

[1] Giorgio Grisetti; Cyrill Stachniss; Wolfram Burgard. URL: http://wiki.ros.org/gmapping.

[2] R. Bormann; F. Jordan; W. Li; J. Hampp; M. Hägele. URL: https://wiki.ros.org/ipa_room_segmentation.

[3] Jiří Hörner. *Map-merging for multi-robot system.* Bachelor's thesis. Prague, 2016. URL: https://is.cuni.cz/webapps/zzp/detail/174125/.

[4] Upma Jain, Ritu Tiwari, and W. Wilfred Godfrey. "Comparative study of frontier based exploration methods". In: *2017 Conference on Information and Communication Technology (CICT).* 2017, pp. 1–5. DOI: 10.1109/INFOCOMTECH.2017.8340589.