UNIVERSITY OF PADUA

INFORMATION ENGINEERING DEPARTMENT (DEI)

MASTER'S DEGREE IN COMPUTER ENGINEERING

Report Assignment 1

Prof.
Emanuele Menegatti

Students of Group 19:
Francesco Agostini : Matricola
Andrea Felline : Matricola
Andrea Pietrobon : 2087639

Bitbucket Repository
Video example

Contents

1 Introduction

This report aims to provide a comprehensive overview of the functionality and methodology employed in our solution for the given assignment. Specifically, we will delve into the following key aspects:

- The structural framework of our solution, detailing the code architecture and organization.
- Our approach to detecting obstacles, encompassing the identification of walls and circular movable obstacles within the environment.
- The strategy adopted to navigate through narrow corridors utilizing laser data for effective pathfinding.

2 Movement

The movement of the robot is managed through the `move_base` action system of the robot. Each single topic (goal, feedback, and result) is read separately for convenience.
The server receives the orientation of the robot as an angle in radians with respect to the original orientation of the robot, but it should send it as a Quaternion. To achieve this, it uses an object of type `tf2::Quaternion` and the function `Quaternion::setRPY()`.

The position and the quaternion are sent to the robot on the `/move_base/goal` topic. The robot then automatically starts moving towards the given position and orientates itself with the provided angle once it reaches it.

While the robot moves, it sends its current position on the topic `/move_base/feedback` (along with other information), and when it reaches the goal, it sends a message on the topic `/move_base/result`.

3 Detection

The server, upon sending the goal to the robot, also initiates the process of reading the messages that the robot sends on the topic `/scan_raw`. On this topic, the robot transmits a list of distances obtained from a scan of the half space in front of it: the scan extends from an angle of **-1.9199 radians** to **1.9199 radians**, where **0 radians** represents the direction directly in front of the robot, resulting in a field of view slightly larger than **180 degrees**.

The server analyzes these distances and groups them: a group signifies a continuous surface, thus forming a new group whenever a distance significantly differs from the previous one. A continuous set of points within a group is defined as a profile, with each point in a profile not differing from its neighbors by more than **0.5 meters**.

Each profile undergoes scrutiny to ascertain whether it resembles a semi-circle, akin to what would be seen if a cylinder were present. The assessment of a profile's semi-circular nature involves several criteria:

- The distance between the nearest point to the first and last points of the profile should be similar (difference $<$ **0.1**).
- The midpoint between the first and last points, presumed to be the circle's center, should share a similar distance with the nearest point.
- The radius (i.e., the distance from the center to one of the profile points) should approximate the expected radius of the cylinders being detected (approximately **0.17 meters**).
- If these conditions are met, a stronger but slower check is performed to further validate the profile. This additional check examines every point's distance from the center, ensuring similarity to the circle's radius.

Upon satisfying all conditions, the profile is labeled as a circle; otherwise, it is marked as unknown. The list of profiles identified as cylinders is saved and transmitted with the feedback. If the current list's number of elements is greater than or equal to the largest previously seen list, it replaces the previous record. This approach aids in updating the list, accommodating any changes in cylinder positions that may occur while the robot continues detecting.

Updating the larger list, even if the current one holds the same number of elements, ensures adaptation in case any of the cylinders are repositioned during the robot's detection process. If the robot re-observes all the cylinders simultaneously, it updates the list with the revised positions.

4 Final Notes

1. It was required to reflect the status of the robot in the feedback. In our program, the server sends feedback only when the robot is moving and detecting, so there is only one type of status and one type of feedback message. The feedback content still reflects the status of the robot since it contains the current position of the robot (indicating movement and position updates) and the lists of cylinders (indicating detection).

2. In the video, we can observe occasional false positives in the feedback of the first goal sent, but memorizing the largest group of cylinders helps the server to rectify these errors. It would be highly improbable for the algorithm to produce multiple errors simultaneously. Additionally, different values for the position, orientation, and feedback flag yield various outputs, demonstrating the robot's correct behavior in response to these parameters.

GROUP 19

Francesco Agostini, francesco.agostini.5@studenti.unipd.it
Andrea Felline, andrea.felline@studenti.unipd.it
Andrea Pietrobon, andrea.pietrobon.4@studenti.unipd.it

Bitbucket repository link: https://bitbucket.org/unipd-projects/ir2324_group_19/
Video example link: https://drive.google.com/file/d/1OVGCOG_-ivf0-LZL4KkG6C-o00eFIXfU/view?usp=sharing