Problem settings and definition

March 17, 2020

1 Exploration problem

The exploration problem can be defined as the problem of mapping an unknown environment by means of a team of robots. Thus, the objective is the production of a map of the environment but this introduces further problems, concerning for example the localization of each robot dealing with the various sources of uncertainty, how to distribute the team and how to coordinate them. These are three of the main aspects considered in the development of a solution to this problem, providing the major impact on performances. To compare different approaches the main metric used is the time taken by the exploration [3, 10, 4], and also the distance traveled by the robots is sometimes considered [11].

A formal definition of the *exploration problem*, in the following abbreviated as *EP*, can be provided through a 4-ple $\langle A, E, P_0, T \rangle$ where:

- A is the set of robots used for the exploration;
- E is the environment to explore;
- P_0 is a vector of the initial poses of the team;
- T is the termination criterion.

This four elements characterize the instance of EP addressed, while the solution of the same consists in producing a map M of the environment E. The map M consists in a 2D occupancy grid representing the areas of E. The center of each cell c is characterized by its coordinates in the environment, provided as a vector (x_c, y_c). Each cell contains a variable, whose value tells if that cell is clear, occupied by an obstacle or still unknown. Clearly, the first two values refer only to already explored cells, while the third refers to those cells which are still not scanned by any robot of the team.

The experiments presented in this work are run on MRESim, a 2D simulator aimed at testing multi-robot exploration scenarios [1]. In particular, it allows to create an instance of *EP* through a configuration file which provides the robots in the team, their initial poses and a PNG image providing the map of the environment to explore. The termination criterion used is the percentage of explored area and it can be modified by varying a variable in the code.

The following sections describe the settings for the experiments, focusing at first on the environment and the agents modeling, then moving the attention to the exploration strategy and the coordination mechanisms.

2 Localization and mapping

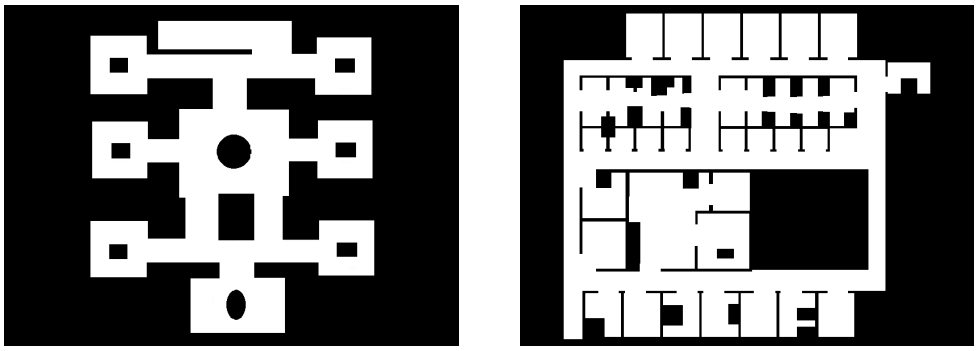
The problem of mapping an unknown environment and simultaneously localize the robot on the partial map built is one of the fundamental problems of robotics, called *SLAM problem,* from *Simultaneous Localization And Mapping.* The robot knows its initial pose and as it moves, the uncertainty about its motion increases, due to uncertainty in the odometry. For this reason it becomes necessary to localize it on the map, even a partial one. A formal description of it, as provided in [5], is suitably done in a probabilistic framework and distinguishes two versions, the *online* and the *full SLAM problem.* Let X_T, U_T and Z_T be three statistical variables representing respectively the sequence of locations assumed by the robot, i.e. the *path*, the sequence of odometry measurements and the sequence of measurements provided by the sensors. Let also m be the true map of the environment. The *full SLAM problem* is then defined as the problem of estimating the posterior probability of the map together with the whole path traveled by the robot, that is $p(X_T, m|Z_T, U_T)$. The *online SLAM problem* aims at estimating the posterior of the actual location of the robot, rather than the whole path, together with the map. Thus, $p(x_T, m|Z_T, U_T)$.

What happens in practice is that as the robot moves in the unknown environment, it perceives the surroundings through its sensors, has an estimate of its odometry measures and these are used to reduce the uncertainty both about the map built and the location of, or the path followed by, the robot, depending on the version of the SLAM algorithm implemented. As the exploration goes on, the partial map approaches to a complete map of the environment and it is also possible to get rid of the noise in the measurements. The model for the map can be both topological [6] or metric [7], in particular in this work the representation provided is metric, consisting in an occupancy grid.

2.1 Environments

This work considers only indoor environments. Reasons behind this restriction are related to the work of [4], of which this thesis is an extension, in order to provide a comparison against the same type of environments. However, this focus is justified by the application contexts, being the exploration of indoor environments more common with respect to outdoor ones.

The representation of the map provided by the SLAM algorithm used in this work is a two-dimensional occupancy grid. Moreover, being applied to a multi-robot scenario, comes out the difficulty of merging the maps computed by solving the SLAM problem for each agent. The solution to this is a centralized approach by means of a base station, which aim is to collect all the individual local maps and combine them into a single global map.



(a) S-O-LP environment (b) L-C-HP environment

Figure 1: Example of two different environments

Similarly to [4], the environment are classified according to some features characterizing them. In particular, the aspects considered are the dimension, the openness, and the parallelizability. The first two are easy to formally define. For the third one, a formal definition can hardly be provided being influenced by many factors. To make the definitions more readable, recalling that M denotes the map of the environment modeled as an occupancy grid, let $S : M \to \mathbb{R}$ be an auxiliary function which takes as input a cell of the map and provides its area. The features considered can be defined in the following way.

- Dimension: the amount of free area in the map. Thus, $D(M) = \sum_i S(f_i)$ with f_i being a clear cell. The size of the environment clearly affects the average time taken by robots to explore it. Smaller the environment, lower the amount of resources needed to efficiently map it. From this, the distinction into small (S) and large (L) environments.

- Openness: the property of an environment to be composed of large open spaces (O) rather than cluttered ones (C). It is related to obstacle density, so to the probability of a randomly picked cell to be occupied by an obstacle. This is formally defined in a probabilistic framework as $O(M) = \frac{\sum_i S(o_i)}{\sum_j S(c_j)}$ where o_i is an occupied cell and c_j is a generic cell.

- Parallelizability: the property of an environment to enforce the spreading of the robots during the exploration. A formal definition to this is hard to provide, but it can be informally presented considering that if the environment allows the team to spread, it is highly parallelizable (HP). Otherwise, if the robots are forced to stick together, it is lightly parallelizable (LP).

To make these distinctions clearer, it is worth looking at some of the environments used in this work and at how they are classified on these features. In Figure 1, two very different environments are presented.

The environment in Figure 1.a is composed by a series of large corridors, ending in some rooms. This is classified as S-O-LP, because its dimension is

small and the areas of which is composed are mainly open ones, not cluttered rooms. The lightly parallelizability comes out from its configuration, where the robots follow almost similar paths, forced by the long corridors and they are not allowed to spread into it.

The second one (Figure 1.b) is likely to be the map of an office. It is composed by a lot of rooms, corridors and spaces with different dimensions. It is classified as L-C-HP because the amount of free area is really large and is mainly composed of rooms and limited spaces. For the configuration of the corridors, the robots tend to spread in different directions, without sticking one another. For this reason, the environment is classified as highly parallelizable.

2.2 Agents

In MRESim, an agent is characterized through:

- a number and an ID to uniquely identify it;
- its pose, expressed by a vector $p = [x, y, \phi]$, where (x, y) are the coordinates of the grid cells occupied by the robot and ϕ is its orientation;
- the sensing range, set to the default value;
- the communication range, assumed to be infinite;
- the battery life, assumed to be infinite as well;
- its type, it can be the base station, a relay or an explorer.

The communication range and the battery life are assumed to be infinite because the focus of this work is on the results provided by the use of different coordination mechanisms. In this way is possible to simplify the mechanisms not to consider scenarios in which the robot runs out of fuel or they are unable to communicate, even if these situations are of particular interest in a more realistic context and object of study as in [2] and [3], respectively.

As stated above, the type of the robot, not to be confused with its role, can be: base station, relay or explorer. The *relay* type is never used in the simulation performed for this work, being useful in cases in which communication between a robot and the base station is not possible. Indeed, each team used is composed by one base station and from two to eleven explorers.

The *base station* is the central coordinator of the team, and stores the global map computed by merging the partial maps provided by the agents. It is configured similarly to other agents but its pose is fixed to the initial one. It is important for the purpose of exploration because allows agents to coordinate by means of a centralized unit, rather than a decentralized approach, which would make the merging of the map way more difficult. An example of this is in [8], where only as soon as two robots meet, they are able to merge their map and then proceed with the exploration based on this updated one.

The *explorer* is the main kind of agents employed, being the one which moves in the environment to map it. Apart from the configuration parameters presented

above, it is also characterized by a finite value of speed. They are equipped with a laser sensor allowing them to scan a semicircle of radius fixed to the sensing range in front of them. The value for the sensing range is specified in the configuration file. As an explorer perceives previously unknown cells with the laser beam, communicates the measurements to the base station which merges them with the stored map and updates it.

3 Exploration strategy

The frontier-based exploration presented in [9] is applied to identify the candidate locations to explore. A frontier is the boundary region between known and unknown space and by moving towards frontiers, this boundary is pushed accordingly.

As highlighted in the previous section, the model for the map is an occupancy grid. Each robot has its own local map, updated through the sensor measurements, and merged at the base station to provide a global map, which is the one used in the frontiers identification phase.

Their position is identified by at first drawing the polygon of the known space. Each side of this polygon which separates the known area from the unknown one is suitable to provide frontiers. If the area of this suitable locations is higher than a certain threshold, then the cell in the middle is considered as a possible navigation goal. There is also the possibility that the area is too large, in which case it is split into more than one frontier.

4 Coordination mechanisms

A coordination mechanism is the algorithm providing the allocation of robots to the possible navigation goals computed by the exploration strategy. Together with the SLAM and the exploration strategy, this completes the view on the settings needed for the exploration. In fact, the process of exploring an unknown environment starts with the robots scanning an area through the sensors and these data, together with the initial location, are used as input to the SLAM algorithm, which provides a partial raw map as output. This is given to the exploration strategy that, as stated above, performs a discretization of the polygon surrounding the known area and computes the frontiers, excluding the ones too small. At this point, the coordination mechanism is applied to find out an allocation of robots to frontiers.

The different coordination mechanisms are the main focus of this work. In the following are presented both the base mechanisms from [10] and their extensions provided by [4].

4.1 Base mechanisms

In [10], three mechanisms are presented, namely *reserve, buddy system* and *divide and conquer.* They are all focused on how *proactive* are the team members not

strictly needed for the exploration. The team can be separated in two sub-teams, the *active* and the *idle set*, with the first one composed by the robots which goal is to explore one of the frontiers detected, the second one is made up by the remaining robots. An example is useful to clarify this distinction.

Suppose that at the beginning of the exploration, the sensed environment provides only two frontiers (may add a figure) and the team is composed by four robots. The size of the team is greater than the number of frontiers, thus only two robots are needed to explore them. Therefore, they compose the active set and the remaining two robots form the idle set. The policy for the active set is the same for all the mechanisms, to explore the closest frontier. The differences come out dealing with the idle set, in fact the way in which robots are proactively moved allows to distinguish among each method. Moreover, this theme of a proactive use of the idle set is the leitmotiv linking the work started in [10] and extended both by [4] and this thesis.

Reserve is the less proactive mechanism because as the name implies, the idle set is left as a reserve at the initial location. As the exploration starts and the first frontiers are detected, the robots are split into the active set and the idle set, in a similar way to the example provided before. Then, the robots in the active set move towards their navigation goal and the ones in the idle set remain still in their initial position. As the exploration goes on and new frontiers are found, their number may be higher than the size of the active set, making some or all the robots from the idle set needed, which are then turned into active and assigned to a target location. Once the idle set is empty and all the robots are active, the exploration proceeds in an uncoordinated way.

Divide and conquer is the most proactive mechanism presented in that work because the idle set moves together with the active set. At the beginning of the exploration, active agents are assigned to the frontiers and the idle set is split in several subsets, one for each active agent. For example, assume that at the beginning, only a frontier is found. Then, an agent is marked as *leader* and assigned to explore it. The other robots follow it as it moves towards its navigation goal, until at least another frontier is detected. For the sake of the example, assume that at this point there are two frontiers. In this case, a robot from the idle set is turned into active and marked as leader. The idle set is split in two: one half follows the first leader, the other half follows the other one. This goes on until the idle set is empty, after which the exploration proceeds in an uncoordinated way. Differently from the reserve mechanism, this approach allows to have robots from the idle set nearer to the navigation goal to which they are going to be assigned. In this sense, the idle team members are considered to be more proactive with respect to reserve, where waiting at the initial location makes the distance between the robot turned into active and the navigation goal higher.

Buddy system is considered to be halfway between the two for what concerns proactivity of the agents. This comes clearer by looking at how the system handles the idle set. As soon as robots are deployed on the environment, couples are formed, composed by a robot marked as *leader* and another marked as *follower*, each one is the *buddy* of the other. Once frontiers are identified, the minimum

number of leader agents is turned into active and assigned to them. Each follower follows its own leader towards this task. The other couples remain still at the initial location, similarly to the idle set in the reserve mechanism. When a branching point is met, that is a point where there are two or more frontiers, the couple is split and the leader is assigned to a frontier, while the follower to the other one. Here, the analogy with the divide and conquer mechanism can be seen. If a robot split from its buddy finds another branching point, a couple from the idle set is called and assigned to one of the two frontiers, while the single robot goes towards the other. This clarifies why the buddy system can be seen as a mix of the reserve mechanism and the divide and conquer. It both keeps the idle set waiting at the initial location for the moment in which it is needed, as reserve, and each leader goes with a follower from which splits when a branching point is met, similarly to the divide and conquer. In this way, when a follower is turned into active, it is already pretty near to its goal, providing a similar advantage of divide and conquer, without the disadvantage of having the whole team moving close.

4.2 Proactive mechanisms

In [4], three modifications for the mechanisms presented in [10] are provided. As stated previously, they focus on enhancing how proactive is the idle set. The way in which this is done is pretty straightforward for what concerns reserve and buddy system, while it is a little more tricky for divide and conquer. The mechanisms proposed are named *proactive reserve, proactive buddy system* and *side follower.*

The main problem with reserve mechanism is that once a robot from the idle set is turned into active, it has to move from the initial location to its goal. The distance it has to travel may be lower if the robot is moved in a nearer position while it is still idle. This is exactly what this modified mechanism tries to do by moving the idle robots to the barycenter of the active robots positions. In this way, they are likely to be nearer to the newly detected frontiers which have to be explored.

The buddy system faces a similar problem to reserve mechanism: the robots in the idle set wait to be turned into active at the initial location. Thus, the proactive version of the buddy system moves the idle couples at the barycenter of the active agents locations, for the same reason explained above concerning the proactive reserve.

The problem of divide and conquer is different, being linked with the interference caused by moving the whole team of robots together. Therefore, the team of robots is split into groups of three at the beginning of the exploration and roles are assigned to them. The central robot is the leader, the right one is the right follower and the left one is the left follower. This roles are statically determined and never modified. Moreover, they affect the assignment of frontiers to the group members, being the frontier along the direction of the movement assigned to the leader, the ones on its right assigned to the right follower and the ones on the left to the left follower symmetrically. In this way, there is a trade-off

between the interference caused by the number of robots moving together and the distance from the frontiers, reducing the first one without affecting too much the second one.

References

[1] J. de Hoog, A. Visser, S. Cameron. MRESim, a Multi-robot Exploration Simulator for the Rescue Simulation League. 2015.

[2] M. Betke, B. Awerbuch and R. Rivest. Piecemeal graph exploration by a mobile robot. In Proceedings of the 8th Annual ACM Conference on Computational Learning Theory (COLT), pages 374–386, 1995.

[3] M. Moors, W. Burgard and M. Schneider. Coordinated multi-robot exploration. IEEE Transactions on Robotics, 21(3):376–378, 2005.

[4] M. Cattaneo. An Analysis of Coordination Mechanisms for Multi-Robot Exploration of Indoor Environments. 2017.

[5] S. Thrun, J.J. Leonard. Handbook of Robotics, chapter 37. Springer, 2008.

[6] G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard. A Tutorial on Graph-Based SLAM. In IEEE Intelligent Transportation Systems Magazine 2(4):31-43, 2010.

[7] D. Koller, M. Montemerlo, S. Thrun. A Factored Solution to the Simultaneous Localization and Mapping Problem. In Proceedings of the National Conference on Artificial Intelligence (Edmonton 2002), volume 37, pages 81–91, 2002.

[8] A. Giannitrapani D. Benedettelli, A. Garulli. Cooperative SLAM using M-Space Representation of Linear Features. Robotics and Autonomous Systems, 60:1267–1278, 2012.

[9] B. Yamauchi. Frontier-based exploration using multiple robots. In Proceedings of the Second International conference on autonomous agents (Minneapolis 1998), pages 47–53, 1998.

[10] C. Nieto-Granda H. Christensesn and J.G. Rogers III. Coordination strategies for multi-robot exploration and mapping. The International Journal of Robotics Research, 33(4):519–533, 2014.

[11] F. Amigoni. Experimental Evaluation of Some Exploration Strategies for Mobile robots. In Proceedings of the IEEE International Conference on Robotics and Automation, pages 2818–2823, 2008.