Multi-robot Planning Using Robot-Dependent Reachability Maps

Tiago Pereira, Manuela Veloso and António Moreira

Abstract In this paper we present a new concept of robot-dependent reachability map (RDReachMap) for mobile platforms. In heterogeneous multi-robot systems, the reachability limit of robots motion and actuation must be considered when assigning tasks to them. We created an algorithm that generates those reachability maps, separating regions that can be covered by a robot from the unreachable ones, using morphological operations. Our method is dependent on the robot position, and is parameterized with the robot's size and actuation radius. For this purpose we introduce a new technique, the partial morphological closing operation. The algorithm was tested both in simulated and real environment maps. We also present a common problem of multi robot routing, which we solve with a planner that uses our reachability maps in order to generate valid plans. We contribute a heuristic that generates paths for two robots using the reachability concept.

Keywords Multi-robot systems · Map representation · Reachability · Planning and scheduling · Morphological operations · Coordination

T. Pereira(✉) · M. Veloso
Carnegie Mellon University, Pittsburgh, PA 15213, USA
e-mail: tpereira@cmu.edu, mmv@cs.cmu.edu

T. Pereira · A. Moreira
FEUP - Faculty of Engineering, University of Porto, Porto, Portugal
e-mail: {tiago.raul,amoreira}@fe.up.pt

T. Pereira · A. Moreira
INESC TEC - INESC Technology and Science, Porto, Portugal

This work is financed by the ERDF European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness), by National Funds through the FCT Fundao para a Cincia e a Tecnologia (Portuguese Foundation for Science and Technology) within project FCOMP-01-0124-FEDER-037281, and under Carnegie Mellon-Portugal Program grant SFRH/BD/52158/2013.

L.P. Reis et al. (eds.), *Robot 2015: Second Iberian Robotics Conference*,
Advances in Intelligent Systems and Computing 417,
DOI: 10.1007/978-3-319-27146-0_15

1 Introduction

Multi-robot systems have been the focus of research for some time, because they can often accomplish tasks more efficiently than a single robot and are more resilient to failures [12]. Common applications of multi-robot systems are exploration, search and rescue, and scheduling of service robots.

The planning problems are usually solved using optimization techniques, such as the ones used in the multiple vehicle routing and the multiple traveling salesman problems [1]. Generally, one wants to distribute tasks among robots.

When planning, it is important to consider heterogeneity and the reachability limit of each robot. We have nowadays a high diversity of robot configurations, and should be able to use that in our favor [7]. Furthermore, in multi-robot systems, robots must not ignore their own heterogeneity, and should be able to plan together considering their different capabilities.

Although multi-robot problems have already been studied regarding the distribution of tasks, there is less work on how to determine if those tasks are feasible for a robot. Therefore, we investigated the geometric and spacial properties of robots in the world, their impact on which positions are achievable, and how it affects task assignment in multi-robot scenarios.

One simple example is when we want to coordinate two robots to go to a goal position to execute a task. Assuming we have circular robots, a bigger robot has more limited motion capabilities, and in principle cannot reach as many places as a smaller robot. This simple example shows it is important to consider physical characteristics, i.e. heterogeneity, when planning. Moreover, coordination of heterogeneous robots should not be hard-coded. Robots should be able to objectively represent their heterogeneity, having their own representation of the environment that is dependent on their physical characteristics. This representation would be useful not only for the robots themselves, but also to share information with teammates, and allow more optimal coordination.

One common way of representing the accessibility of a map considering the robot shape is the configuration space [13]. However, robots need also take into account their other capabilities, like actuation. And for that purpose, the configuration space does not give enough information.

As an example, we can consider a vacuum cleaning robot. We show in Figure 1 a simulated environment designed to test the many specifications of our problem, such as corridors, and rooms connected through openings of different shapes and sizes. In Figure 1(a) we have a map with black representing walls and obstacles. And in Figure1(b), we have the configuration space which inflates the walls and obstacles for some robot size. White represents where the robot center is allowed to be, and not what parts of the environment it can clean.

In Figure 1(c), assuming the robot is in the center of the image, we show what we introduce as *reachability map*, with white representing regions in the environment the robot can clean. Therefore, it adds to the configuration space (regions where the

(a) Map (b) Configuration space (c) Reachability map

Fig. 1 Difference between the configuration space and the actuation space for a vacuum cleaning robot.

robot can be) the information on the actuation capabilities, in this case, its cleaning reachability. As expected, the robot cannot reach corners.

Our problem is then to determine 2D reachability maps, representing the points in the environment that can be actuated by a robot from some position that is reachable from the initial position.

To solve the problem of constructing a 2D reachability map, we propose a methodology that uses algorithms borrowed from image processing. Image processing is a common framework in robotics, but used preferably for perception tasks. Here, in a new intersection of image processing and map transformation, we use it to achieve extraction of higher level information from SLAM based grid maps. These occupancy grid maps can be obtained with, for example, rao-blackwellized particle filters [10].

In the next section we discuss related work, and then we present in depth our algorithm to obtain the Robot-Dependent Reachability Map (RDReachMap). We show results in a real map obtained from SLAM. We also contribute an heuristic for multi-robot planning for multiple goals, using the reachability maps. Finally, we present our conclusions and the directions for future work.

2 Related Work

We focus our work on heterogeneous multi-robot scenarios, creating the concept of reachability maps to extract information that is useful for coordination.

Multi-robot systems have been a research subject for some time, because robots can merge their measurements to produce more precise maps, also allowing for faster and more robust task execution. [3]

Various map representations can be used, such as occupancy grids, geometric features, or topology. We combine in our approach both lower level (occupancy grid maps) and higher level (reachability maps) data. Comparable approaches have been used before, using manifolds for the map representation [11].

Regarding cooperation, it is accomplished through exchange of information. Robots can communicate their reachability maps so that they choose where each should go in an efficient way. Related works on the literature use information sharing among teammates to improve the robot beliefs, complementing the different perceptions of each one [4]. They also show how important it can be to coordinate robots using transfers in order to achieve an efficient solution of the pickup and delivery problems [5].

It was also shown by [2] how to do multi robot path planning, with highly dynamic domains. However, they do not consider feasibility of goals before planning. Another alternative approach to coordination has been suggested by [6], using market-based approaches, and auctions, in domains ranging from mapping and exploration to robot soccer. However, none of these approaches determine feasibility of goals for each robot, as we do in this work.

Regarding heterogeneous multi-robot systems, not only terrestrial mobile robots of different sizes and characteristic, but also aerial robots, have been used for the purpose of rescuing missions [15]. In this example, they use the different sensing capabilities of each one to coordinate the robots.

Regarding the use of techniques borrowed from the image processing field, there are more examples, such as automatically extracting topology from an occupancy grid [8]. Finally, [16] uses reachability to coordinate robots with humans in shared workspaces.

3 RDReachMap

Considering maps are discrete representations of the environment, we found a duality between images and maps, because both of them are a discrete sampling of the world. Therefore, our main idea is to perform purely geometric operations on the map in order to obtain coverage, using image oriented algorithms.

The first step is to transform an occupancy grid map M (with probabilities of occupation in each cell) into a binary map of free and obstacle cells, using a given threshold. Then we have a image-like map, I, and we can extract the reachability coverage considering a given robot radius.

We assume robots have circular shape. While there are many robots with a circular footprint, there are others with different shapes. However, as we want to find the reachable regions, which can be seen from different positions and orientations that can be reached from the initial position, we need a rotation-invariant model of the robot. Moreover, many robots have a close to circular shape. Assuming circular robots with omni-directional motion, the actuator model is irrelevant, and only the maximum range of actuation is needed.

3.1 Morphological Operations

We create a simulation test map, which we show in Figure 2(a). Given a grid of positions G (discretized environment), a black and white binary image I (one pixel per grid point) as a binary map representation, and a structuring element R (robot), the morphological operation dilation is defined as

$$I \oplus R = \{z \in G \mid R_z \cap I \neq \emptyset\} = \bigcup_{z \in R} I_z \tag{1}$$

where $R_z = \{p \in G \mid p = r + z, r \in R\}, \forall z \in G$.

Applying the dilation operation to black points in the image, the algorithm inflates the obstacles of the map by the robot size, achieving the configuration space. In Figure 2(b), we show in white the free parts of the configuration space, C_{free}. The same kind of approach is used in most motion planning algorithms, inflating objects in order to find a path that minimizes some cost. Indeed, inflation is the solution used, for example, in the ROS navigation package[14].

However, the configuration space shows where the robot center can be, not giving any information on its covering capabilities.

Considering A to be a structuring element representing the actuation capabilities (also circular), the morphological operation erosion is defined as

$$I \ominus A = \{z \in G | A_z \subseteq I\} = \bigcap_{z \in A} I_{-z} \tag{2}$$

We can apply the erosion morphological operation to the configuration space, with a given actuation radius (size of A). The combination of erosion after dilation ($I \oplus R \ominus A$) is called in computer vision morphological closing operation [18].

As we show in Figure 2, if using the same radius for the erosion operation (actuation size) as the size of the structuring element for the dilation, we obtain the original map with rounded corners. Going back to the vacuum cleaner example, the "closed" map implies the robot can reach all positions, except the square corners, because of its circular shape.

However, Figure 2(c) implies all regions can be vacuum cleaned, which is not true. The configuration space has disconnected regions, meaning the robot cannot transverse the entire environment. The disconnected parts do not show up after the morphological closing, meaning the topology is changed. In Figure 2(b), a robot in the bottom cross cannot reach the upper cross. However, in Figure 2(c), there is a feasible path between the two crosses, even tough the robot cannot start from one cross and clean in the other cross position.

The lack of disconnected components after the closing operation happens because it returns the reachability for any initial position of the robot. Therefore, the map after the morphological closing does not serve our purpose, because it represents reachability from any initial position of the robot in the configuration space. And for the planning purposes, we need to obtain the reachability for the current initial position of each robot.

(a) Original map (b) Morphological dilation (c) Morphological closing

Fig. 2 Example of morphological operations in a map to obtain the (b) configuration and (c) actuation spaces. Configuration space shows feasible points for the robot center, and the actuation space shows in white what the robot can actuate.

3.2 Partial Morphological Closing

Although the morphological closing does not yield the desired reachability map for a given initial position, it is still interesting, because it represents a very similar concept, which is overall reachability for any initial position. Therefore, in order to use this technique to achieve our goal, we introduce the concept of partial morphological closing.

After applying dilation to the original map in order to get the configuration space, we obtain a set of disconnected regions. Going over the image, we cluster those regions using the "Flood Fill" technique, labeling C_{free}. There are more complex methods for labeling, with better performances [9]. Flood Fill proved fast enough for our requirements. This operation is equivalent to the morphological operation propagation (conditional successive dilations).

The algorithm finds the label of the robot's initial position x_0, and uses that region as the only reachable part of the configuration space, $Reach(x_0, C_{free})$.

For each image, with $Reach(x_0, C_{free})$ as white pixels and the rest as black, we can now apply the erosion operation $Reach(x_0, C_{free}) \ominus A$, and finish the partial morphological closing. The result obtained, in Figure 3, shows the reachability maps for three regions corresponding to three different initial positions.

In the end of the procedure we get the robot-dependent reachability map (RDReachMap) that closes small corridors, transforming them into obstacles. It also separates regions, and is able to detect all points of the world that can be actuated from any position in a feasible path from the initial location. Thus we call the result a reachability map, because it shows all the points that are reachable to the robot's actuators, given an initial position.

(a) Initial positions 1 (b) Initial positions 2 (c) Initial positions 3

Fig. 3 Reachability map as a result of partial morphological closing operation, shown for the three sets of possible initial positions.

3.3 Overall Algorithm

In this section we present the algorithm that we use to get the reachability map. The main idea of RDReachMap is to use the closing morphological operation to eliminate from the map regions that are not reachable. This technique is enough to erase from the final map corridors whose width is smaller than the circumscribing radius of the robot. However, in order to get the reachability map, our algorithm then uses the concept of partial morphological closing. The full algorithm is shown in Algorithm 1.

Algorithm 1. RDReachMap: Creating reachability maps from grid maps

Require: Grid Maps G
1: **for** each G **do**
2: Initialization: B&W Image I from M
3: Dilation: inflated image C^{free} from I
4: Labeling: Reachable set $Reach(x_0, C_{free})$ from C^{free} and initial position x_0
5: Partial Closing: Reachability map $RM = Reach(x_0, C_{free}) \ominus A$
6: **end for**
7: **return** RM

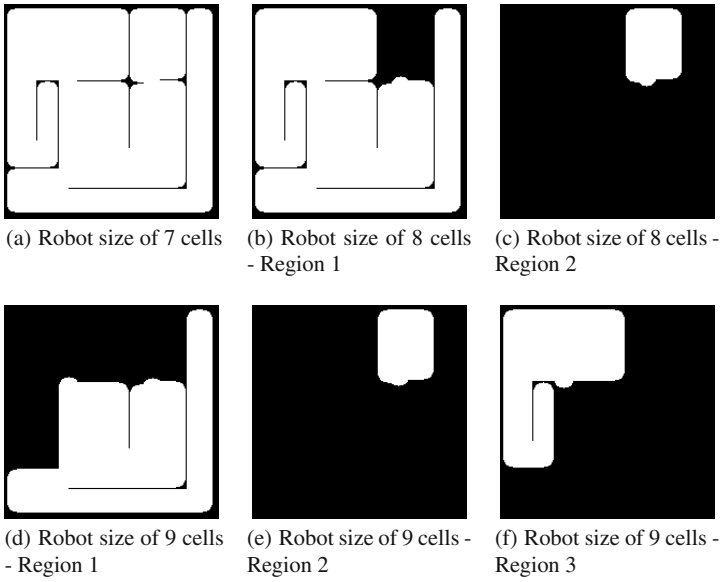
3.4 Topology Analysis for Different Robot and Actuation Radii

We now show the result of our algorithm in both simulated and real environments. The real world map was obtain by a robot doing SLAM inside a building.

Although we used static maps, this work can be used on online SLAM and exploration, because it is an algorithm that only depends on the current map, and allows for semi-explored regions. RDReachMap always considers unexplored regions as

free cells when extracting the reachability map. The optimistic assumption makes the planner always send robots to explore unkown regions.

In Figure 4 we show reachability maps for different robot sizes. For test purposes, we defined the robot size as a number of cells in the grid map.



(a) Robot size of 7 cells (b) Robot size of 8 cells - Region 1 (c) Robot size of 8 cells - Region 2

(d) Robot size of 9 cells - Region 1 (e) Robot size of 9 cells - Region 2 (f) Robot size of 9 cells - Region 3

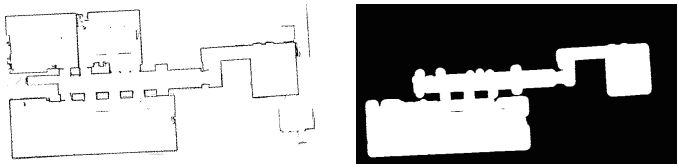
Fig. 4 Reachability maps for different sizes of robots. Size of map is 200 × 200 cells.

Depending on the size of the regions, corridors start being blocked when the robot size increases. Figures 4(b) and (c) show a scenario where only a room was separated from the others compared to the smaller robot case in Figure 4(a), where all regions were accessible, except for corners of the environment. Therefore, either the robot is in one region and the rest of the environment is unreachable, or the robot is in the other region and then the first one is unreachable. In figures 4(d) to 4(f) more rooms are separated from each other, due to the increase in robot size.

Figure 5 shows the reachability map of a real world map. Again, as the robot size increases, more and more regions will be added to the reachability map, reducing the number of reachable places at each point of the map.

If the actuation size is the same as the robot, then it is a special case because the total morphological closing yields the correct reachability.

It is important to notice that RDReachMap can only be used for actuation radii smaller then the robot size, otherwise the algorithm does not work properly. However, it is reasonable to assume equal size because most times the actuation range is given by the robot size, as with the vacuum cleaning robot.



(a) Original map (b) Robot size of 11 cells

Fig. 5 Reachability in a real world map

4 Coordination with Reachability Maps

The purpose of RDReachMap is to extract information that can be used to improve the efficiency of coordination. Knowing a priori the reachability, the combinatorial cost of finding solutions in multi-robot scenarios is reduced.

As an example, when doing search and rescue, or when doing SLAM, robots need to explore the environment. One possibility is to use the concept of exploration frontiers, and robots need to coordinate in order to efficiently distribute tasks [17]. When using RDReachMap, frontiers in unreachable places can be automatically given to smaller robots. But in this case, more than helping efficiency, RDReachMap is an essential feature, because it is needed to determine which goals each robot can accomplish.

4.1 Multiple Vehicle Routing

As an example, we use a scenario with two robots in the environment (with different reachability for each one), and a set of goals randomly distributed over the environment. The objective is to have a centralized planner, with access to reachability maps, distributing tasks to each robot.

We want to minimize the total time the robots need to get to all goals. Being p a possible solution combining the paths of both robot, $t_r(p)$ is the time it takes for robot r to complete its part of the path in solution p. Therefore, the idea is to find the route that minimizes the maximum time of execution of all robots. The optimal exploration route is given by

$$p^* = \arg\min_p \{\max_r \left(t_r(p)\right)\} \tag{3}$$

We assume that robots travel at the same speed. So, we transform the time minimization problem into a distance minimization. This is a multi traveling salesman problem, and with the combinatorial explosion, it is impossible to calculate the optimal solution efficiently. One possible solution to the problem is to use a greedy

heuristic, where the nearest goal is added to the correspondent robot path. That solution is often very poor, getting stuck in local minima. Therefore, we propose a different heuristic.

Heuristic. We know the route for a robot includes with certainty all the goals only reachable to it. So, our planner can have a broader view of where the robots are moving in the future. With our cost function, we have at the same time to minimize the cost (total distance) of each path, and minimize the difference between the cost of paths of all robots. An intuitive heuristic is to add goals to one robot path if its distance to that path is much less than the distance to the other robot's path, thus not only minimizing distance to one path, but also maximizing the distance to the other robot current path.

In our problem we have a robot that is bigger (R_1) and can only reach some small set of regions (S_1), and another (R_2) that can reach a bigger set, including the regions only accessible to smaller robots ($S_2 \cup S_1$). We start to allocate the certain goals, which are the ones only accessible to R_2 (goals in S_2). Then in order to capture the non-local perspective of the problem, we search for the one with the maximum distance to the initial position. We are using this first goal as an initial estimate of the path it will have to do.

Furthermore, we add more goals to the paths. In order to choose where to add goal position in the current path of robots, we analyze the increase in cost (total distance) given by the additional size of the path when adding the goal. So, if we add goal g between points i and j and being $d(.,.)$ the distance function calculated with A* between two points, the additional cost of adding goal position g between i and j is

$$c(g, i, j) = d(i, g) + d(g, j) - d(i.j) \tag{4}$$

We can also add it to the end of the path, where the cost is only going to be $d(j, g)$. Goals are added at points where cost is minimized, and we choose a goal that minimizes the adding cost. Goals in S_2 are added until all of them are in the path of R_2 (the smaller robot that is the only to reach goals in S_2).

Then, we need to choose a goal for R_1 from S_1, also giving a good initial estimate of the global path. We want robots to move apart from each other for efficient exploration. Thus our metric now will return the goal which has the bigger distance to both robots, and we use the sum for this purpose.

In the final step, goals in S_1 are added to both robots' paths, until there are no more goals left. We want to start adding goals to some path if we know it is clear it should go for that robot and not the other one (minimizing uncertainty of attribution). Therefore, we compute for every goal left, the cost of adding a goal to both robot paths. Being d_1 the distance to path of R_1, and d_2 for R_2, the cost of adding the goal to R_1 path is $c_1 = d_1 - d_2$, and vice versa for R_2. This means we are trying to minimize the distance of the added goal to the current path, and at the same time, maximize the distance of that goal to the path of the other robot. We choose two goals, one for each robot, that minimize the cost for each one. However, we always add goals incrementally.

Algorithm 2. Heuristic for creating paths

Require: Grid Reachability Maps from 2 Robots
1: Separate Goals in categories, S_1 and S_2, using reachability maps
2: Choose goal from S_2 with maximum distance to R_2 (smaller robot) initial position
3: Allocate goals S_2 greedily to R_2
4: Choose first goal from S_1 to R_1 maximizing distance to both robots.
5: **while** Goals in S_1 **do**
6: Sort goals, minimizing uncertainty of attribution
7: Choose goal minimizing length difference of the two robots paths.
8: **end while**
9: **return** Path p

As we want to make the total distance for both robots more similar, we choose from the previous two goals the one that makes the total path distance for both robots more similar. In Algorithm 2 we show the overall heuristic. The same algorithm can be used for more robots, extrapolating the strategy of separating the regions that are reachable only to certain robots.

4.2 Results

In Figure 6, we use the simulated map presented in Figure 1(a). It shows in black the regions only reachable to one of the robots, R_2. We show the result of our route planning for two robots using reachability maps.

In this example scenario, when using 10 goals, the heuristic took around 3.5 seconds, and the brute force 27.5 seconds, both yielding the same paths for each

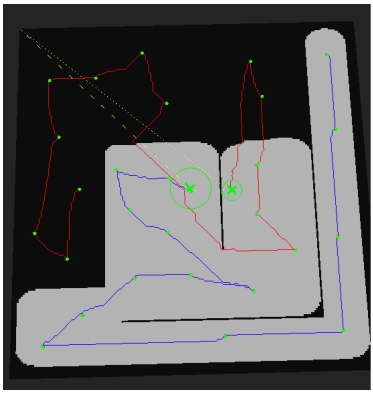


Fig. 6 Paths in multi-robot planning using reachability maps. Blue path for robot with smaller reachability, red for one with more reachability. Crosses are robot initial positions, with circles representing the robot sizes. Green balls are goal positions.

robot. The path costs for each robot were approximately 432 and 480 (units are grid cells). For 28 goals, the total distance for each robot was approximately 622 and 624 using the heuristic (no groundtruth, too many goals for brute-force), taking around 23.5 seconds to compute.

5 Conclusion

In this paper we presented RDReachMap, an algorithm that creates robot-dependent reachability maps. RDReachMap is able to create maps separating reachable regions from the unreachable ones, using only image processing operations. We introduced the concept of partial morphological closing in order to solve the reachability problem. We tested our algorithm both in a simulated environment and a real world map, showing the influence of the size of the robot on the resulting reachability map.

We showed the importance of using the concept of reachability for planning. When robots work in cooperation, it is important that they can execute tasks according with their own characteristics. Robot size and actuation radius are some of these characteristics, and in our work we built a framework so that robots can cooperate taking that into account. We presented a common multi vehicle routing problem, and we solved it contributing a heuristic that uses reachability to try to minimize the maximum time of operation.

As future work, we want to extend our framework to 3D maps. Moreover, we want to consider the heterogeneity of robots not only in their size, but also in their shape. The goal is to drop in the future the circular robot restriction. We want to use these concepts of reachability to be able to coordinate a fully heterogeneous multi-robot team.

References

1. Bektas, T.: The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega **34**(3), 209–219 (2006)
2. Bruce, J., Veloso, M.: Real-time randomized path planning for robot navigation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002, vol. 3, pp. 2383–2388. IEEE (2002)
3. Burgard, W., Moors, M., Stachniss, C., Schneider, F.: Coordinated multi-robot exploration. In: IEEE Transactions on Robotics (2005)
4. Coltin, B., Liemhetcharat, S., Meriçli, C., Tay, J., Veloso, M.: Multi-humanoid world modeling in standard platform robot soccer. In: 2010 10th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 424–429. IEEE (2010)
5. Coltin, B., Veloso, M.: Scheduling for transfers in pickup and delivery problems with very large neighborhood search. In: Twenty-Eighth AAAI Conference on Artificial Intelligence (2014)
6. Dias, M.B., Zlot, R., Kalra, N., Stentz, A.: Market-based multirobot coordination: A survey and analysis. Proceedings of the IEEE **94**(7), 1257–1270 (2006)
7. Dorigo, M., et al.: Swarmanoid: A novel concept for the study of heterogeneous robotic swarms. IEEE Robotics and Automation Magazine **20**(4), 60–71 (2013)

8. Fabrizi, E., Saffiotti, A.: Extracting topology-based maps from gridmaps. In: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA2000, vol. 3, pp. 2972–2978. IEEE (2000)
9. Grana, C., Borghesani, D., Cucchiara, R.: Fast block based connected components labeling. In: 2009 16th IEEE International Conference on Image Processing (ICIP), pp. 4061–4064. IEEE (2009)
10. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Transactions on Robotics **23**(1), 34–46 (2007)
11. Howard, A.: Multi-robot mapping using manifold representations. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA 2004, vol. 4, pp. 4198–4203. IEEE (2004)
12. Howard, A., Parker, L., Sukhatme, G.: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. The International Journal of Robotics Research **25**(5–6), 431–447 (2006)
13. LaValle, S.M.: Planning algorithms. Cambridge University Press (2006)
14. Lu, D.V., Smart, W.D.: Towards more efficient navigation for robots and humans. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1707–1713. IEEE (2013)
15. Luo, C., Espinosa, A., Pranantha, D., De Gloria, A.: Multi-robot search and rescue team. In: 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR) (2011)
16. Pandey, A.K., Alami, R.: Mightability maps: a perceptual level decisional framework for co-operative and competitive human-robot interaction. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5842–5848. IEEE (2010)
17. Pereira, T., Moreira, A.P., Veloso, M.: Coordination for multi-robot exploration using topological maps. In: CONTROLO2014–Proceedings of the 11th Portuguese Conference on Automatic Control, pp. 515–524. Springer (2015)
18. Soille, P.: Morphological image analysis: principles and applications. Springer-Verlag, New York (2003)