

Multi-robot Topological Exploration Using Olfactory Cues

Ali Marjovi and Lino Marques

Abstract. This paper presents a distributed multi-robot system to search for odor sources inside unknown environments. The robots cooperatively explore the whole environment and generate its topological map. The exploration method is a decentralized frontier based algorithm that is enhanced by considering odor concentration at each frontier inside its cost/gain function. The robots independently generate local topological maps and by transferring them to each other, they are able to integrate these maps and generate a whole global map. The proposed method was tested and validated in real reduced scale scenarios.

1 Introduction

Search and rescue operations inside buildings, caves, tunnels and mines can be extremely dangerous tasks. An example of an extremely risky situation is the human operation inside an industrial warehouse during a fire. In these cases the human senses can become severely impaired: the visibility will be reduced by the smoke, the noise caused by the fire will make it impossible to communicate, and additionally dangerous vapors and toxins may be released. The use of autonomous robots to assist such tasks in complex environments reduces the risk of these operations. Robots can search for toxic chemicals and other targets while they explore the environment, providing real-time data about the discovered map and the status of the facility. It is well known that multi-robot systems can be faster and more robust doing a given task than single robot systems, but the reverse side of such benefits are the difficulties of efficiently coordinating those multiple robots, and accurately merging all the data gathered individually by each of them.

Ali Marjovi · Lino Marques
Universidade de Coimbra, Coimbra, Portugal
e-mail: {ali, lino}@isr.uc.pt

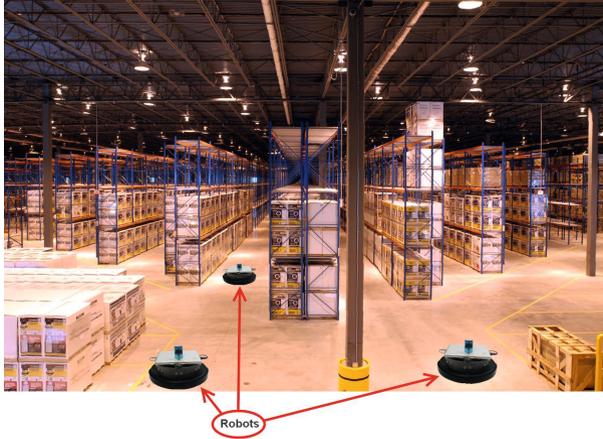


Fig. 1 Robots searching inside a warehouse composed by multiple straight corridors

The problem of search and exploration with multiple robots in an unknown environment can be stated as following; Consider a group of N mobile robots, moving in R^2 that are labeled as R_1, R_2, \dots, R_N . Each robot $R_i (i = 1, \dots, N)$ is able to communicate with the other robots localized at a short distance range Δ . The robots are equipped with sensors for measuring the odor intensity and air flow direction. There are unknown number of odor sources in the environment which emit odor gas into the area. There is no central base-station for the system. The robots should act separately and independently from the others. There is no global positioning system and the odometry of the robots is not very accurate. There is no knowledge about the environment before the mission except that it is similar to a structural building containing corridors, corners, branches, crosses, etc. similar to a warehouse (Fig. 1). The problem is to localize all the odor sources in the environment, explore the whole area, and generate a topological map of the environment.

The authors have been working on the problem of finding odor and fire sources with robots in previous studies [1, 2, 3, 4, 5]. This current project tries to address the problem of odor source searching in unknown environments by complementing it with the environment exploration and mapping.

The cooperation, communication, and management of the robots in a multi-agent system can be done either in centralized way by using a base station as the server, or decentralized way by having a distributed behavioral based algorithm (like in [6, 7]). Lacking a central station makes it difficult to distribute the tasks between the robots. Since the environment is unknown, the robots are not aware of the tasks before exploring the area, i.e. there can not be any kind of task allocation before the mission. Task allocation must be done during the mission automatically by the robots which are participating in the search and exploration mission.

Singh and Fujimura [8] presented a decentralized online approach for heterogeneous robots. In their method, most of the time the robots work independently.

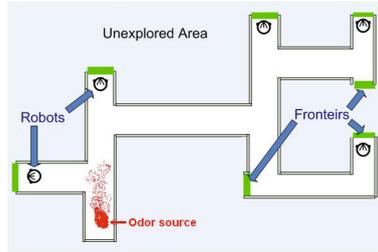


Fig. 2 Multi-robot frontier-based search and exploration

When a robot finds a situation that is difficult to solve by itself, it sends the problem to another robot which may be able to solve the situation. Yamauchi [9] proposed a distributed method for multi-robot exploration, yielding a robust solution even with the loss of one or more vehicles. A key aspect of this approach involves sharing map information among the robotic agents so they execute their own exploration strategy, independently of all other agents. While this technique effectively decentralizes control, exchange of map information is not enough to prevent inefficient cooperative behaviors. As a simple conclusion, “task sharing” is tightly related to two other issues; “cooperative technique” and “map merging”.

If the robots know their relative locations and share a map of the area that they have explored so far, then effective coordination can be achieved by guiding the robots into different, non-overlapping areas of the environment [6]. In other words, effective coordination can be achieved by extracting exploration frontiers from the partial maps and assigning robots to frontiers based on a global measure of performance [6, 7]. Frontier based exploration is a simple approach for decentralized multiple robot task allocation (Fig. 2). Frontiers are the borders of the partial map, between explored free space and unexplored area. These borders, thus, represent locations that are reachable from within the partial map and provide opportunities for exploring unknown terrain, thereby allowing the robots to greedily maximize information gain [10]. However, methodologies should have a strategy to not send two robots toward the same frontier.

While multiple robots cooperatively explore an environment, information from individual robots must be integrated to produce a single globally consistent map. This is a difficult problem when the robots do not have a common reference frame or global positioning system [11]. Topological maps provide a brief characterization of the navigability of an environment, and, with measurements easily collected during exploration, the vertices of the map can be embedded in a metric space [11]. These maps use a graph to represent possibilities for navigation through an environment. The current proposed approach employs a topological mapping technique, so the robots only exchange a few environmental features.

Most of the existing approaches to coordinate multi-robot mapping assume that all agents know their locations in a shared (partial) map of the environment [2, 5, 6, 7]. Having a general positioning system is a constraint in unknown areas where there

is no knowledge about the environment. This paper deals with two issues in this aspect, the first dealing with uncertainty in localization of each robot by correcting it by the information of the partial maps, and the second presenting an approach for map merging in the case that the robots do not know their relative locations and their coordinate systems are different with each other.

Using topological maps, the problem of “map merging” is reduced to the “graph merging” problem that is already addressed in several ways [11, 12, 13]. Whereas most approaches to topological map merging and related problems have focused on using either map structure [14] or map geometry [15], our algorithm takes advantage of both. Similar to [11], the use of map structure allows quick identification of potential vertex matches in the maps (and rejection of mismatches), while the use of map geometry enables the algorithm to directly merge maps with multiple (disconnected) overlapping regions. In another words, the algorithm in [11] uses both the structure and the geometry of topological maps to determine the best correspondence between maps with single or multiple overlapping regions. However, if the geometric data of the local maps are not obtained through a unique coordinate system this method (in [11]) will not be functional.

As already stated, the exploration must be done in a way that the group of robots automatically intends to look for the odor sources in the environment. Over the past years, odor localization has become recognized as a valuable area of robotics with practical applications. Most of related works to the problem have focused on mobile robots at a scale of the order of tens of centimeters, operating in open areas free of obstacles with a background fluid flow. Fluid flow in these environments is dominated by turbulence. The odor is carried downwind from the source forming a plume. Due to turbulence, the plume meanders, and the chemical concentration within the plume is patchy [16]. The researchers have developed methods that employ combinations and variations of plume acquisition and plume upwind following using reactive control algorithms. However, most of the significant projects have addressed the problem with a single robot [1, 17, 18, 19] and/or in open, free obstacle spaces [19, 20]. However, this paper attends to fulfill the goals using multiple robots in a structural environment (similar to a real building of a warehouse).

2 The Proposed Method

This section explains the concept of the proposed multi-robot cooperation technique. This method is illustrated in the schematic diagram of Fig. 3. As shown in the diagram, Some of the tasks of the diagram are briefly described below.

2.1 *Olfactory Search and Exploration Algorithm*

It is desired to find odor sources and cover the whole environment as fast as possible. Therefore, it is essential that the robots share their tasks and individually achieve the objectives through optimal paths towards the odor sources. In an unknown

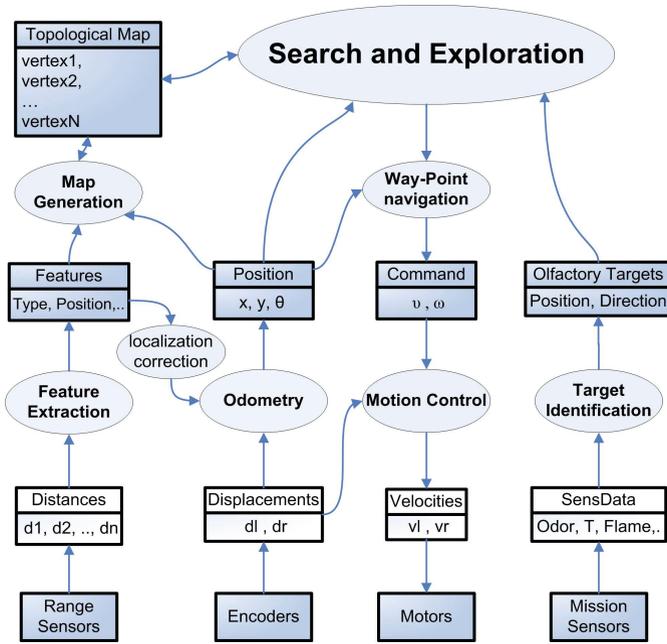


Fig. 3 Software architecture

environment, the immediate goals are the frontiers. Most of the time, when the robots are exploring an area, there are several unexplored regions, which poses a problem of how to assign specific frontiers to the individual robots without existence of a specific task allocator. In the proposed approach, the robots firstly decide to explore the frontiers which indicate higher odor concentration. The robots must avoid selecting the same frontier, this may result in collision concerns. Another problem is the lack of base station, so the robots should be able to explore autonomously and also avoid collisions between themselves. To address these problems, the proposed method is based on a behavioral exploration algorithm that is presented in algorithm 1.

Algorithm. 1 describes the core of the decision making technique of the proposed method. To briefly describe this algorithm; the robots start exploring the environment independently. Each robot looks forward to get into new features in the environment. It generates its own local topological map out of the extracted features of the environment and also transmits this local map to the other robots which are working on the same environment. Once the robot is in the situations that has to make a decision to select its future path (e.g. in the branches), it first measures the odor concentration in that place. If the odor concentration is more than a certain threshold, it means that the robot is in the zone of an odor source and it must go to the direction of up-wind in order to localize that. In this case the robot takes an action to go towards up-wind.

Algorithm 1: Odor source searching and map exploration algorithm

```

1 while there is at least one unexplored frontier in the map do
2   repeat
3     | Go_forward_and_follow_the_potential_field_algorithm();
4     | until getting a different feature in the environment;
5     | if the new node exists in the map then
6       | Update_map's_data();
7       | Correct_the_odometry_error_based_on_the_feature_data();
8     | else
9       | Add_new_node_to_map();
10    Send_the_new_local_map_to_the_other_robots();
11    if the current node has any unexplored link then
12      | M = Measure_the_odor_concentration();
13      | D = Detect_the_wind_direction_by_anemometer();
14      | if (M > odor_threshold) and (D is not explored) then
15        | Set_the_new_Objective_to_go(D);
16      | else
17        | F=Find_unassigned_unexplored_frontier_with_highest_Utility-Cost();
18        | D = Calculate_the_best_path_to_take_based_on_the_A*_algorithm();
19        | Set_the_new_Objective_to_go(D);
20    if M > Odor_Source_Threshold then
21      | Report_this_place_as_an_odor_source_to_all_other_robots();
22 End of algorithm

```

However if a robot is travelling inside the explored area and wants to select a frontier to explore; the frontier is selected based on the cost of reaching it and the utility it can provide to the exploration. The cost is calculated through the A* method, where it simultaneously determines the optimal path to reach the frontier and its distance. Therefore, the cost is proportional to the distance that the robot has to pass to reach the frontier.

$$cost = dist(A_{i=0,n}^*[(X_R, Y_R), (X_{f_i}, Y_{f_i})])$$

where (X_R, Y_R) is “position of the robot”,
 (X_{f_i}, Y_{f_i}) is “position of the frontier i ”
and n is “number of frontiers”.

The utility depends on the level of odor concentration in that frontier, which means that if there are several frontiers at similar distances, the robot will go to the one that has higher utility, i.e. higher odor concentration;

$$utility(i) = Odor_Concentration(i) \quad \forall i \in \{1..n\}.$$

This procedure will make the robots disperse and explore the environment in an efficient way towards odor sources.

The robots generate the topological map of the environment during their search and exploration mission. Within the topological map, besides having information regarding the kind of nodes, their position and the odor concentration in that feature, it also has data describing the location of the robots and their frontier target. Through this data, a robot can see which frontiers are unexplored, their position and if any robot has targeted them as its objective. Therefore, the robots will not attend to explore the same frontiers. Each robot is aware of the frontier that the other robots have aimed to explore, so it will choose another frontier that is unexplored and

unassigned to any other robot. As a result, the robots will autonomously pick up the tasks in a way that not any frontier will be assigned to more than one robot.

2.2 Feature Extraction

With the type of environment considered in this paper, there are five types of features that can be extracted from structured environments; corridors, corners, crosses, branches (3-way junctions) and dead-ends. The robots should recognize environmental features based on the distances from left, right and front measured by range sensors. Fig. 4 is an example that shows how robots detect corridors and branches by their sensors. Since “feature extraction” is not the main theme of this work, this paper does not go to its details.

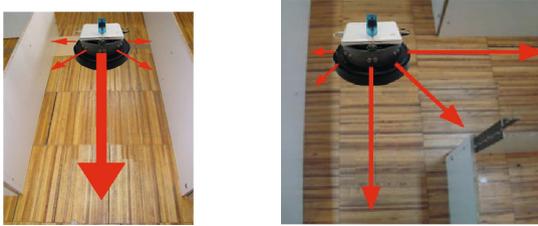


Fig. 4 Feature extraction, left: corridor, right: branch

2.3 Map Merging

Consider several robots exploring a single environment while each one has its own coordinate system. Their axes of X and Y do not match with each other and moreover they do not know where the reference point of the other’s localization system is. Each one of them is generating the topological map of its visited local area. They are simultaneously sending these local self-generated topological maps to each other. The problem is “how each one of them can integrate the data coming from the others to its local map and generate a bigger map?”. Fig. 5 shows an example of this problem. There is no central station unit for attuning the robots; moreover, there is not any specific landmark in the environment. Therefore, the robots should solve the problem in a distributed way.

Similar to [11], in this method, the generated map represents more than just the structure of the environment. Additional information, such as the degree of vertices, the orientation of edges at vertices, and other attributes, is recorded and stored in annotations of the graph. Fig. 7 shows an example of the topological map’s data with brief descriptions.

The next step is to match the topological maps and generate a merged global map out of them. Since the topological map includes geometric data of each node, we

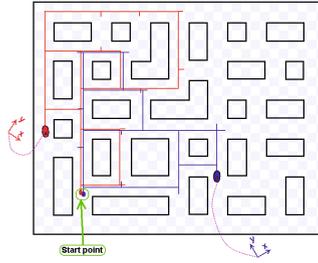


Fig. 5 An environment being explored by two robots with different coordinate systems

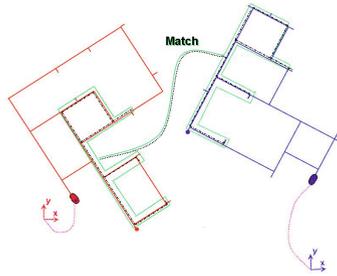


Fig. 6 Matching the generated maps of two robots exploring the environment in Fig. 5

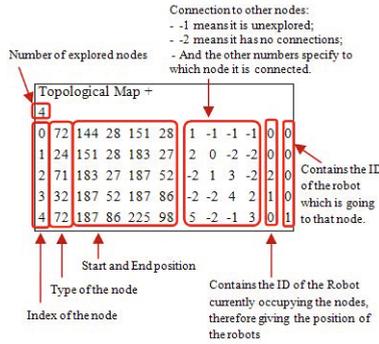


Fig. 7 An example of topological map data

can consider the maps as several subgraphs that belong to a unique graph. Now the problem of map merging is converted to the standard well-known subgraph isomorphism problem.

We considered the problem of “map merging” as a “graph matching” problem (see Fig. 5 and Fig. 6). One of the problems in graph matching is error-tolerant subgraph isomorphism. The robots should identify a common subgraph when there may be missing vertices and edges. Solutions to this problem are generally evaluated in terms of the “edit distance”, the smallest sequence of elementary graph operations

(i.e., substitution, deletion, insertion) that transforms one subgraph into the other. When a robot cannot reliably detect “places” or when the environment is dynamic, the resulting topological map may have missing vertices and edges.

Common subgraphs $H_1 = [V, E, k]$ (where V is the set of vertices, E is the set of edges and k is the graph size) and $H_2 = [W, F, k]$ of two given graphs G_1 and G_2 are those, of equal size k , that are isomorphic to each other. This means that there should be such numeration of subgraphs’ vertices $x(i)$ and $y(i)$, that :

$$\forall i, j \in \{1 \dots k\}$$

$$\text{Equivalence_vertex_function}(v_{x(i)}, w_{y(i)}) = \text{true} \quad (1)$$

$$\text{Equivalence_edge_function}(e_{(x(i), x(j))}, f_{(y(i), y(j))}) = \text{true}$$

“*Equivalence_vertex_function*” : Two vertices are equivalent if and only if their environmental features match with each other.

“*Equivalence_edge_function*” : Two edges are equivalent if their lengths are approximately equal and their connecting vertices are equivalent.

In other words, all pairs of matching vertices in the subgraphs are connected by matching edges (including the virtual null edges). Now we just look for numbered sets $X = \{x(i)\}_{i=1}^k$ and $Y = \{y(i)\}_{i=1}^k$, satisfying conditions (1). The major issue here is to define the “*Equivalence_vertex_function*” and “*Equivalence_edge_function*” functions. Two vertices are equivalent if and only if their environmental features matches with each other and two edges are equivalent if their length is almost equal and their connecting vertices are equivalent. By that, the subgraphs match with each other regardless to the positions of the vertices but based on the topology of them.

The project has stated that the robots start from the same point (usually an entrance of the building) at the beginning of the mission. By this assumption it is guaranteed that the first vertex of all the local maps is from the same point, therefore the graphs always have at least one common subgraph and the robots are able to merge their maps in any case, while coordinate systems are not matched.

The value of k (common subgraph size) is a critical issue, Based on the experiments that we had in simulations and also in the real world, we figured out that it is good enough to set k to five. This means that if two robots find a common subgraph with five vertices, they can surly merge their local maps together.

Here is a description of the map merging method. Whenever a robot finds a new feature in the environment, it adds this feature as a new node to its local map and sends a message to all the other robots and reports the new map. In the other hand, each robot has a running memory-resident program that always is listening to the network and receives all the messages that are sent by the other robots. When a robot receives a message that shows another robot has found a new feature in the environment, it starts the hypothesis building process by creating the list of all vertices in the local topological map that are structurally compatible with the new map. Vertexes are tested for compatibility by examining their attributes: exactly known attributes (e.g., vertex type) must match perfectly; inexactly known attributes (i.e., due to measurement error) must be compared with a similarity test. Finally, this robot can find a common isomorphic subgraph and will modify its own local map. Therefore all the robots are able to merge their local maps with each other without having a base station.

Once the robot finds the subgraph in another map that is isomorphic to a subgraph of its own map, it will merge these maps with each other after finding the geometric transform function that converts the positions of the vertices of the second map to its current map. Since we have used a position correction method (described in the next section) that corrects the odometry of the robots, a linear transform function is good enough to convert the coordinates of two maps to each other. The transform function is found by a simple geometric calculation. Finally, the robot is able to add all the nodes of the other map to its own map after transforming their positioning data to its coordinate system.

Robots localization is a key issue in multi-robot mapping and exploration. There is no global or local positioning system or any kind of landmark or beacon for localization in this project. The only tool that the robots have for determining their position is their odometry. However, the odometry is unreliable because of uneven floors and wheel slippage. It is therefore necessary to augment the localization accuracy by measuring position of robot relative to known objects in the environment.

Normally odometry errors accumulate incrementally as while as the robot is travelling, therefore, the robot's odometry is more accurate at the beginning of an experiment and it loses its accuracy during the test. If the robot enters to an environmental feature that has been already explored, it can look at the map and find the start position of that feature, then correct its (x,y) based on the data that has been already stored inside the map. It does not matter if this feature was added to the map by the current robot or by another robot in the team, since the feature has been added to the map in the past; it means that the location that was saved in the map is more reliable than the odometry of the robot. This method is only used when the robot is passing an area that had been already explored.

Some other issues in multi-robot search and exploration, namely "feature extraction", "olfactory sensing", "communication" and "motion control" are out of scope of this paper and mostly were already addressed in [1, 2, 5] by the authors.

3 Experiments

The proposed algorithm has been tested and validated both in real reduced scale scenario using iRobot¹ Roomba robots and in simulation using the Payer/Stage environment [21]. The Roomba vacuum cleaning robot is an attractive research platform since it is inexpensive, readily available, and can be fully monitored and commanded through a serial port interface. In the current work, a set of Roombas were upgraded with small laptop computers (ASUS Eee PC 901) running a Linux based operating system and the Player environment. The computers interface through a micro-controller board with a set of five sonars, 2D anemometer and a gas sensing board (see Fig. 8). The gas concentration was measured with a custom sensing board based on metal oxide gas sensors (Figaro² TGS2620). The directional anemometer

¹ <http://www.irobot.com>

² <http://www.figarosensor.com>

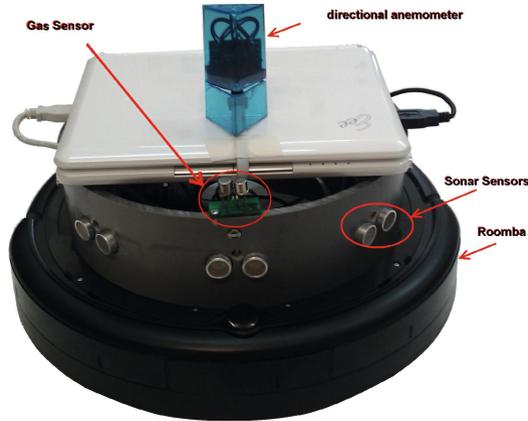


Fig. 8 iRobot Roomba robot equipped with a laptop, a sonar array, gas sensors, and 2D anemometer

was based on self heated NTC³ sensors placed around a triangular prism and processing the raw measurements with a method similar to the one previously described in order to estimate the wind direction [22]. The Player/Stage driver for Roomba robots makes it possible to run the same code either in simulation or on the real Roomba robots. The sonars range was approximately 1.5 m while the biggest width of reduced scale testing environments is about 40 cm.

During exploration and navigation, the robots are simultaneously acquiring information from the environment. Given the structure of the considered environment, mainly corridors, if a robot is travelling in the upwind direction and the gas concentration decreases suddenly, that means that the robot has just passed through an odor source. On the other hand, if the robot is travelling in the downwind direction and it starts detecting a high concentration of odor, it means that there is an odor source in this location.

The proposed method was tested in different small scale maze-like environments, like the one shown in Fig. 9. The shown testing arena, with $3 \times 4m^2$ area by 0.5 meters height, has controlled ventilation through a manifold that extracts air from the testing environment through a honeycomb mesh integrated into one of the walls. The opposite surface of the environment contains a similar mesh that allows the entrance of clean air that flows through the environment. Controlled gas sources are simulated with ethanol vapor, generated using bubblers and pumped to different localizations of the environment through a set of PVC tubes.

In order to evaluate the method in terms of odor source searching, the algorithm was experimented in the real world once without any odor source and another time with existence of an odor source. By comparing the results and analyzing the behavior of the robots, the functionality of the method was validated.

³ Negative Temperature Coefficient.



Fig. 9 Three robots exploring a gas free environment

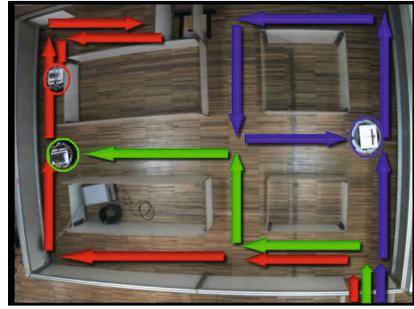


Fig. 10 Three robots exploring the environment and finding the odor sources

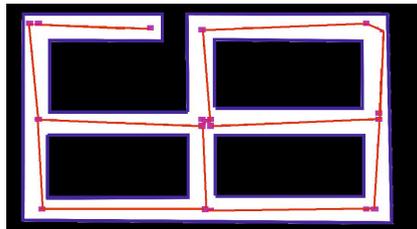


Fig. 11 The generated map for the environment in the Fig. 10

Fig. 9 shows three robots exploring a small maze and finding an odor source. In this experiment there were no odor sources. All robots started from the same point but not at the same time. We intentionally ran the robots a few seconds after each other. The red footprint shows the first robot's path, the blue footprint is related to the second robot and the green shows the footprint of the third robot. Fig. 11 shows the topological man of testing environment generated by the robots. The full algorithm is functional and it works in different maze structures and with different number of robots. For an example of the coordination between the robots, in Fig. 9 when the second robot reached the junction it figured out that the path in the front was already explored so it chose the left path.

The same maze structure was tested with the same robots with adding an ethanol odor source in the left side of the environment. The results show the effect of odor concentration on the behavior of the robots. Fig. 10 shows the path that robots took during exploring the environment. The first robot in the first branch made decision to go to the left-way because of a high clue of the odor and wind speed in that direction (red footprints).

The most important parameter for evaluation of the method is the exploration time. The environment shown in Fig. 9 was tested by one, two and three Roomba robots separately, once without having any odor or gas source, and once with having an odor source releasing gas in the environment. Fig. 12 shows that the exploration

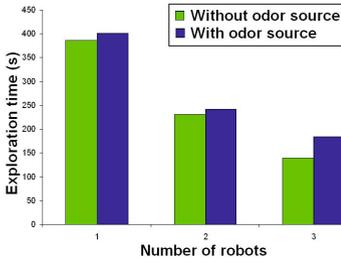


Fig. 12 Exploration time

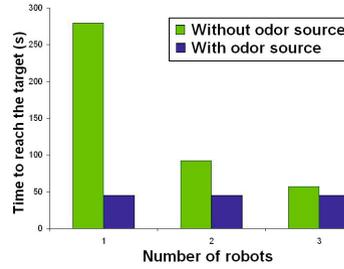


Fig. 13 Reaching the target (the odor source)

time is a bit more, with having gas cues, however it is not a big difference and they are still comparable. Fig. 13 shows the time to reach the target (the location of the odor source) in these two scenarios. The chart shows that the robots reach the target much faster with having gas cues rather than without having it, which proves the functionality of the algorithm. Each result is the average of five similar tests. Different tests with constant conditions had similar results with about eight percent variance. The maximum speeds of the robots were kept constant in all the tests.

4 Conclusions

A proposed method for multi-robot odor source search and exploration in unknown structural environment has been implemented and experimented in realistic reduced scale scenarios. The exploration algorithm is modified by integrating odor sensing cues in the frontiers selection and has been tested in the real world. The robots navigate towards the odor sources and are able to localize them while cooperating with each other by sharing information in their local maps.

In terms of mapping, the robots generate the topological map of environment during exploration. Map sharing is the main tool for automatic distributed task sharing and cooperation in our method. We showed that the robots can merge their topological maps based on common subgraph isomorphism even if they do not have common coordinate systems. After merging the maps, each robot will know the unexplored frontiers and assigns one of the frontiers to itself as a target to explore, so the other robots will not aim to that frontier anymore.

The algorithm was tested in the real world with different configuration and different number of robots and the results show the effect of gas cues on the behavior of the robots and it proves that based on the proposed algorithm, robots firstly explore the area with higher probability of existence of odor sources.

Acknowledgements. This work was partially supported by European project GUARDIANS contract FP6-IST-045269 as well as by the Portuguese Foundation for Science and Technology contract SFRH/BD/45740/2008.

References

1. Marques, L., Almeida, N., de Almeida, A.: Olfactory sensory system for odour-plume tracking and localization. In: IEEE Int. Conf. on Sensors, Toronto, Canada (2003)
2. Marjovi, A., Nunes, J., Marques, L., de Almeida, A.T.: Multi-robot exploration and fire searching. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, St. Louis, USA (2009)
3. Marques, L., Nunes, U., de Almeida, A.: Particle swarm-based olfactory guided search. *Autonomous Robots* 20(3), 277–287 (2006)
4. Marjovi, A., Nunes, J., Sousa, P., Faria, R., Marques, L.: An olfactory-based robot swarm navigation method. In: Proc. IEEE Int. Conf. on Robotics and Automation, USA (2010)
5. Marjovi, A., Nunes, J.G., Marques, L., de Almeida, A.: Multi-Robot Fire Searching in Unknown Environment. In: Howard, A., Iagnemma, K., Kelly, A. (eds.) *Field and Service Robotics*. STAR, vol. 62, pp. 341–351. Springer, Heidelberg (2010)
6. Fox, D., Ko, J., Konolige, K., Limketkai, B., Schulz, D., Stewart, B.: Distributed multi-robot exploration and mapping. *IEEE Special Issue on Multi-Robot Systems* 94(7), 1325–1339 (2006)
7. Burgard, W., Moors, M., Stachniss, C., Schneider, F.: Coordinated multi-robot exploration. *IEEE Trans. on Robotics* 21(3), 376–386 (2005)
8. Singh, K., Fujimura, K.: Map making by cooperating mobile robots. In: Proc. IEEE Int. Conf. on Robotics and Automation (1993)
9. Yamauchi, B.: Frontier-based exploration using multiple robots. In: Proc. of 2nd Int. Conf. on Autonomous Agents (1998)
10. Zlot, R., Stentz, A., Bernardine Dias, M., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proc. IEEE Int. Conf. on Robotics and Automation, USA (2002)
11. Huang, W., Beevers, K.: Topological map merging. *The International Journal of Robotics Research* 24(8), 601 (2005)
12. Garey, M., Johnson, D.: *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman, New York (1979)
13. Krissinel, E., Henrick, K.: Common subgraph isomorphism detection by backtracking search. *Software: Practice and Experience* 34(6), 591–607 (2004)
14. Dudek, G., Jenkin, M., Milos, E., Wilkes, D.: Topological exploration with multiple robots. In: Proc. 7th Int. Symp. on Robotics with Applications, Anchorage, Alaska (1998)
15. Dedeoglu, G., Sukhatme, G.: Landmark-based matching algorithm for cooperative mapping by autonomous robots. In: Proc. the 2000 Int. Symp. on Distributed Autonomous Robotic Systems, Knoxville, TN, pp. 251–260 (2000)
16. Kowadlo, G., Rawlinson, D., Russell, R., Jarvis, R.: Bi-modal search using complementary sensing (olfaction/vision) for odour source localisation. In: Proc. IEEE Int. Conf. on Robotics and Automation, Orlando (2006)
17. Loutfi, A., Coradeschi, S., Lilienthal, A., Gonzalez, J.: Gas distribution mapping of multiple odour sources using a mobile robot. *Robotica* 27(02), 311–319 (2008)
18. Marques, L., Nunes, U., Almeida, A.: Olfaction-based mobile robot navigation. *Thin Solid Films* 418(1), 51–58 (2002)
19. Ferri, G., Caselli, E., Mattoli, V., Mondini, A., Mazzolai, B., Dario, P.: SPIRAL: A novel biologically-inspired algorithm for gas/odor source localization in an indoor environment with no strong airflow. *Robotics and Autonomous Systems* 57(4), 393–402 (2009)
20. Hayes, A., Martinoli, A., Goodman, R.: Distributed odor source localization. *IEEE Sensors Journal* 2(3), 260–271 (2002)
21. Gerkey, B., Vaughan, R., Howard, A.: The Player/Stage project: Tools for multi-robot and distributed sensor systems. In: Proc. IEEE Int. Conf. on Advanced Robotics, Coimbra, Portugal, pp. 317–323 (2003)
22. Marques, L., de Almeida, A.: ThermalSkin: a distributed sensor for anemotaxis robot navigation. In: Proc. 5th IEEE Int. Conf. on Sensors, South Korea, pp. 1515–1518 (2006)