2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Robot-Assisted Discovery of Evacuation Routes in Emergency Scenarios

Ettore Ferranti and Niki Trigoni

Abstract— **When an emergency occurs within a building, it is crucial to guide victims towards emergency exits or human responders towards the locations of victims and hazards. The objective of this work is thus to devise distributed algorithms that allow agents to dynamically discover and maintain short *evacuation routes* connecting emergency exits to critical cells in the area. We propose two Evacuation Route Discovery mechanisms, *Agent2Tag-ERD* and *Tag2Tag-ERD*, and show how they can be seamlessly integrated with existing exploration algorithms, like Ants, MDFS and Brick&Mortar. We then examine the interplay between the tasks of area exploration and evacuation route discovery; our goal is to assess whether the exploration algorithm influences the length of evacuation paths and the time that they are first discovered. Finally, we perform an extensive simulation to assess the impact of the area topology on the quality of discovered evacuation paths.**

I. Introduction

When an emergency occurs within a building, the area is typically off-limits for anyone not wearing respiratory equipment, garments or barrier materials to protect themselves from exposure to biological, chemical, and radioactive hazards. In our recent work, we proposed the deployment of a group of autonomous robots, referred to as *agents*, to explore the area as fast as possible and acquire information about hazards and victims [1]. In [1], when agents enter the emergency area, they dynamically deploy a network of stationary sensor nodes, referred to as *tags*, in order to label the environment. Agents do not communicate directly with each other; instead, they coordinate indirectly by leaving traces of information on the tags that they deploy. Agents are able to read and update the state of local tags, and by doing so, they leave valuable information for other agents in order to help them make intelligent navigation decisions.

Although the exploration algorithms studied in [1] enable robots to explore unknown areas as fast as possible, they do not shed light on how to guide victims towards emergency exits, or how to guide human responders towards the locations of victims and hazards. In this paper, we address this crucial need for dynamic discovery and maintenance of efficient routes that connect emergency exits to critical cells in the area where interesting events are identified. We will hereafter refer to these routes as *evacuation routes*.

Our goal is to address two crucial requirements posed by emergency applications: 1) to discover evacuation routes as early as possible in the exploration process; and 2) to keep evacuation routes as short as possible to enable easy access of human responders to victims and hazards. The objective of this paper is thus to *devise distributed algorithms that allow agents to identify short evacuation routes at an early stage*. Our specific contributions are the following:

- We propose two distributed mechanisms for evacuation route discovery, one based on agent-to-tag communication (i.e., communication between robots and stationary sensors), and one based on inter-tag communication. We show that these route discovery mechanisms can be easily integrated with existing exploration algorithms, like Ants [2], MDFS and Brick&Mortar [1]. The idea is to activate the search for evacuation routes in parallel with the task of area exploration.
- We carefully examine the impact of the exploration algorithm on the efficiency of our route discovery mechanisms. We address the following questions: i) does the exploration algorithm affect the final length of the evacuation paths? ii) does the exploration algorithm affect when evacuation paths are first discovered, and how they are improved over time?
- We measure the performance of our discovery mechanisms in a wide variety of settings. Our goal is to understand how the topology of the area (e.g., size, number of rooms and obstacles, number of emergency exits) affects the quality of evacuation paths. This study could help human responders predict their accessibility to victims and hazards given a rough knowledge of the area's topological features.

The rest of the paper is organised as follows. Section II presents the assumptions of our model. Section III reviews existing exploration algorithms and Section IV presents two novel route discovery mechanisms that can easily be integrated with the existing exploration algorithms. Section V and VI present a thorough evaluation of our proposed route discovery mechanisms when combined with different exploration algorithms and tested in a variety of simulated topology settings. Section VII discusses related work and Section VIII concludes the paper.

II. Model

Emergency area: Consider an emergency area that is conceptually divided into small square cells as depicted in Figure 1. Although we realise the difficulty of such a discretisation of the environment in a real case scenario, we assume that this is possible for the sake of comparison of the different algorithms in the present work. Black cells represent walls, physical obstacles (e.g., desks) or hazards (rooms on fire) that make these cells inaccessible to roaming agents. All remaining cells are accessible and can be explored by agents in search for victims. The area has several emergency exits that can be used to evacuate victims. The map of the area, including the location of inaccessible and accessible cells, is considered to be unknown, especially since it may have

E. Ferranti and N. Trigoni are with the Computing Laboratory, University of Oxford, Oxford, UK {Ettore.Ferranti | Niki.Trigoni}@comlab.ox.ac.uk

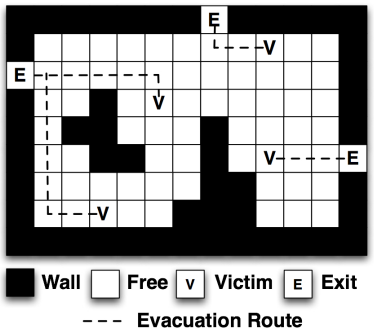

Fig. 1. Emergency area example.

changed as a result of an emergency. The location of victims is also not known in advance.

Agent movement and deployment of tags: Agents are initially deployed in one of the boundary cells and, in each step, they are able to move from the current cell to one of the four adjacent cells in the North, East, South or West directions. As they move to a previously unexplored cell, they deploy a miniature device (e.g., mote or RFID), referred to as *tag*, capable of storing small amounts of information about the state of the local cell.

Communication modes: In emergency situations, long-range wireless communication may be intermittent and unreliable, so we assume that agents (which we assume as equipped with a radio at least able to communicate with tags within a distance of 2 cells) are able to communicate only by reading and updating the tags installed in the local and adjacent cells. This is referred to as *agent-to-tag communication*. In addition, tags located in adjacent cells can exchange messages and alter their state based on the content of these messages. This is referred to as *tag-to-tag* communication. Because of the irregularities of radio propagation, we assume that some of the wireless links between adjacent tags are asymmetric or completely broken.

Agent tasks: At the event of an emergency, agents are charged with two distinct tasks: 1) to explore all accessible cells in the area as fast as possible and 2) to mark the instrumented area with evacuation paths that connect the victim locations with the emergency exits. These tasks must be accomplished in parallel by the roaming agents. Algorithms for the first task have been proposed in [1] and are briefly reviewed in Section III. The focus of this paper is on devising distributed mechanisms for the second task, and integrating them with the existing exploration algorithms.

III. Existing exploration algorithms

In this section, we briefly review three exploration algorithms: Ants, MDFS and Brick&Mortar. As described in [1], they all rely on agent-to-tag communication to explore an unknown area. Agents deploy tags and update their state as a means of coordinating with other agents. A cell is considered to be: i) *unexplored* when no agent has been there before, and no tag is deployed; ii) *explored* when the cell has been traversed at least once and has been tagged, but agents might need to move there again in their way to other *unexplored* cells; iii) *visited*, when the cell has been traversed at least once, and no agent needs to step on it again to reach other *unexplored* cells. When agents are surrounded by *visited* cells, they do not need to move to any of them, and consider the exploration task terminated. A qualitative difference between Ants and the other two algorithms is that agents running the Ants algorithm never mark cells as *visited*, and hence they continue to explore the area indefinitely. In contrast, the other two algorithms eventually mark all cells as *visited* and are aware when their exploration task is completed.

A. Ants

The Ants algorithm is a distributed algorithm that simulates a colony of ants leaving pheromone traces as they move in their environment [2]. Initially, all cells are marked with value 0 to denote that they are *unexplored*. At each step, an agent reads the values of the four cells around it and chooses to step onto the least traversed cell (the one with the minimum value). Before moving there, it updates the value of the current cell, for example by incrementing its value by one. Cells with at least one visit are considered to be *explored*. As pointed out in [1], the Ants algorithm often makes inefficient use of agent resources; agents tend to revisit certain parts of the area multiple times, before moving to completely *unexplored* parts. Another weakness is that agents do not know when they have *explored* all cells in the area, and therefore they continue traversing cells indefinitely.

B. Multiple Depth First Search (MDFS)

The idea behind this algorithm is that agents traverse the cells of an area in a depth-first-search manner [3]. The cell where each agent departs is the root of the tree. For simplicity, consider a single agent that sets out from the root to cover all cells in the area. She gradually builds a depth-first-search tree as she moves from one cell to another. She first navigates the tree downwards, by traversing previously *unexplored* cells, deploying tags there, and marking them as *explored*. The agent has reached the end of a branch when none of its neighbouring nodes is left *unexplored*. She then navigates the branch upwards marking the cells in her way as *visited*. On encountering an *unexplored* cell in her vicinity, she moves to it and starts navigating another branch downwards. This process continues until the agent has marked all cells as *visited* and returned to the root cell from where she started.

Agents running MDFS typically explore an area faster than agents running the Ants algorithm. The algorithm terminates when all agents return to their roots having marked all accessible cells as *visited*. However, MDFS is still not very efficient in terms of exploration time. By definition it traverses each cell at least twice (except for leaf cells), thus resulting in a long exploration time even in open areas without walls where a single traversal would suffice.

C. Brick&Mortar

Brick&Mortar [1] is designed to address the weaknesses of Ants and MDFS. Unlike Ants, agents using Brick&Mortar

know when the exploration task is completed and they do not spend much time revisiting the same cells. Unlike MDFS, agents typically traverse each cell less than twice and explore the area faster.

The driving idea is that of thickening the existing walls by progressively marking cells that surround walls as *visited*. In a way, *visited* cells are similar to *wall* cells in that agents can no longer access them. The algorithm aims to progressively thicken the blocks of inaccessible cells, whilst keeping the remaining cells connected and accessible to roaming agents. As the blocks of inaccessible (*wall* and *visited*) cells become thicker, corridors of accessible (*unexplored* and *explored*) cells become thinner until they finally disappear.

IV. Building evacuation paths

In this section, we describe two mechanisms for building evacuation paths between victims and exits, one that utilises solely agent-to-tag communication and another that also exploits tag-to-tag communication. An evacuation path is a sequence of cells $[c_1, \ldots, c_n]$ such that a victim is located at c_1 and an emergency exit at c_n. To build such a path means to update tags placed in each cell c_i of the path with a pointer to the direction (N, E, S or W) of the next cell c_{i+1} towards the exit. In this way, once a victim is found, she can read the state of the tags, which will guide her toward one of the exits. Since in an emergency scenario prompt evacuation is essential, the goal which we would like to achieve is to keep the path between a victim and an exit as short as possible.

Let us now describe our two Evacuation Route Discovery (ERD) mechanisms: 1) *Agent2Tag-ERD*, which only allows agents to mark tags with evacuation paths (and thus consider tags as passive devices e.g. RFID); and 2) *Tag2Tag-ERD*, which also allows tags to exchange messages in order to find better evacuation paths. Both of these mechanisms run in parallel with the exploration process; they are thus seamlessly integrated with one of the exploration algorithms discussed in Section III.

Agent2Tag-ERD: Consider an agent running one of the exploration algorithms. Upon moving to a cell, the agent reads the state of the current tag, as well as the state of adjacent tags in the N, E, S and W direction. The state of each tag consists of two values: the length of the currently-known evacuation path from this tag to one of the exits (*distance*), and a pointer to the parent tag on this path (*parent*)[1]. Let d' be the distance value of an adjacent tag and d be the distance value of the current tag. If $d' + 1 < d$, then the distance of the current tag is updated to $d' + 1$ and the adjacent tag becomes the parent of the current tag on the evacuation path. This process is repeated in the other direction, i.e. the state of adjacent tags is updated based on the new state of the current tag, as shown in Algorithm 1.

Tag2Tag-ERD: With the introduction of tag-to-tag communication, information can be disseminated much more effectively among tags, to enable them to discover the shortest available path to an emergency exit. Tags exchange messages with their neighbouring tags to inform them of any updates in their state. Recall that the state of tag has two values, *distance* and *parent*, as discussed above. Each message contains the following fields: the ID of the tag that constructed the message (sender), its distance to the exit (*distance*) and a Time-To-Live counter (TTL). Moreover, each tag has a FIFO buffer to contain the messages that it receives from its neighbours. As shown in Algorithm 2, upon receiving a message, if the sender is an adjacent tag in the N, E, S or W direction (we assume that agents let neighbouring tags know when deploying new tag IDs in the field), and

[1]When tags are first deployed their distance is set to infinity and their parent to the NULL pointer. The only exception concerns tags deployed on emergency exits, which have distance 0.

Algorithm 1: Agent2Tag-ERD

```
/* each cell has a distance value (dist)
   representing the length of the path
   leading to the exit                    */
/* each cell has a pointer (parent) to
   the next cell along this path          */
/* c is the current cell                  */
/* First step - auto update               */
for (every adjacent visited or explored cell a) do
    if (dist_a + 1 < dist_c) or (dist_c == NULL) then
        dist_c = dist_a + 1;
        parent_c = a;
    end
end
/* Second step - neighbours update        */
for (every adjacent visited or explored cell a) do
    if (dist_a > dist_c + 1) then
        dist_a = dist_c + 1;
        parent_a = c;
    end
end
```

Algorithm 2: Tag2Tag-ERD

```
/* each cell has a buffer to store all
   the messages it receives               */
/* c is the current cell                  */
for every message msg in the buffer do
    if (msg originated from one of the four adjacent
    cells a) then
        if (dist_a + 1 < dist_c) then
            dist_c = dist_a + 1;
            parent_c = a;
        end
    end
    if (TTL_msg > 0) then
        decrement TTL_msg by 1;
        send the message to the adjacent cells (except
        the cell from which you received it);
    end
    remove msg from buffer;
end
```

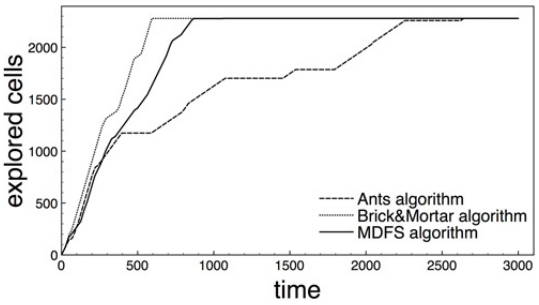

Fig. 2. Total number of explored cells over time, for each one of the exploration algorithms.

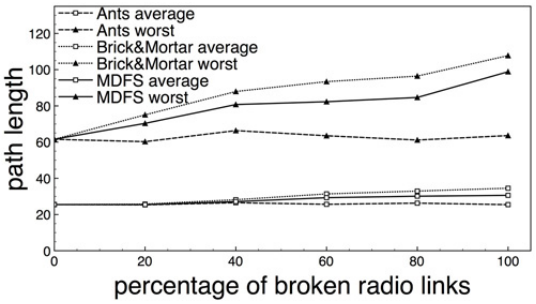

Fig. 4. Average and worst-case (longest) path lengths as we vary the percentage of broken links between adjacent tags. The left part of the x-axis corresponds to perfect inter-tag communication (Tag2Tag-ERD) whereas the right part corresponds to no inter-tag communication (Agent2Tag-ERD).

message's distance plus 1 is less than the distance of the current tag, the current tag selects that adjacent tag as its parent and updates the local distance accordingly. Moreover, if the TTL of the message is not zero, the TTL is decreased by 1 and the message is rebroadcasted to adjacent cells. The use of TTL is to ensure that messages reach all adjacent tags even if the direct communication link between two adjacent tags is asymmetric or completely broken. Moreover, the TTL will reduce the possibility of flooding the network, even if in this work we do not study the effects of Tag2Tag-ERD on network congestion. Note that unlike Agent2Tag-ERD, Tag2Tag-ERD continuously improves evacuation paths by changing the state of tags independent of the presence of an agent nearby.

V. Interplay between exploration and route discovery techniques

In this section, we study the interplay between exploration algorithms and mechanisms for discovering evacuation routes. We consider a testbed area with 2500 (50×50) cells consisting of 4 rooms and 5 emergency exits, in which we deploy 5 agents to explore it and discover evacuation routes.

Figure 2 shows the progress of the three exploration algorithms (Ants, MDFS and Brick&Mortar) as time passes. The results are consistent with our previous study [1]: in this scenario, Brick&Mortar is faster than the other algorithms in exploring new cells in the area, deploying tags there and identifying victims.

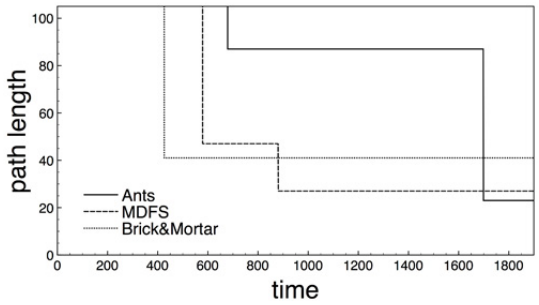

Fig. 3. Length of evacuation path from a victim to an emergency exit, when Agent2Tag-ERD is combined with the three exploration algorithms. As long as agents have not reached the victim, the path length is infinite.

The first question that arises is how the three exploration algorithms perform when integrated with Agent2Tag-ERD, our first mechanism for evacuation route discovery. Figure 3 shows the lengths of evacuation paths from a victim location as time elapses. Brick&Mortar, which explores the area faster, finds a path to the victim earlier than the other two algorithms. However, without tag-to-tag communication, once it finds a path, it rarely replaces it later with a shorter path. On the other hand, MDFS and Ants find evacuation paths later in time, but they often get the opportunity to gradually update these paths with shorter ones, yielding eventually more efficient paths than Brick&Mortar. The reason is that agents running Brick&Mortar usually traverse most of the cells only once in order to save time, and thus do not return to improve the paths in already visited areas. On the other hand, MDFS agents typically traverse each cell twice (first to explore it, and then to mark it as *visited*) and thus perform better in terms of disseminating path information. Finally, agents running the Ants algorithm always manage to find the optimal evacuation paths, because they continue to explore the area indefinitely, revisiting cells multiple times, and continuously improving existing evacuation paths.

The second question that arises is how the three exploration algorithms perform when integrated with Tag2Tag-ERD, our second mechanism for evacuation route discovery. Recall that this mechanism exploits tag-to-tag communication to improve discovered evacuation paths without requiring the presence of agents. Figure 4 measures the length of the average and worst-case evacuation paths discovered when the three algorithms complete the exploration task. In the x-axis we vary the percentage of faulty links between adjacent tags. The value 0% corresponds to perfect communication between adjacent tags, whereas the value 100% reflects no inter-tag communication. As we increase the percentage of broken links from 0% to 100%, the Tag2Tag-ERD mechanism degrades from its ideal performance (when all inter-tag links are available) to the performance of Agent2Tag-ERD that utilises no inter-tag links.

Figure 4 shows that with perfect inter-tag communication (left part of the graph), the performance of the three exploration algorithms is identical. The average length of evacuation paths to various victims, as well as the longest (worst-case) path to the remotest victim, do not depend on the exploration algorithm. However, as we increase the num-

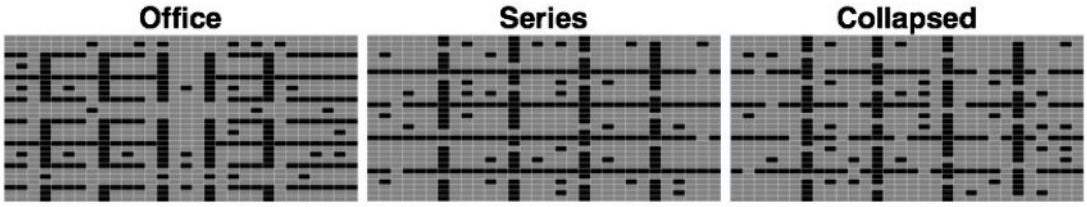

Fig. 5. Examples of different room layouts.

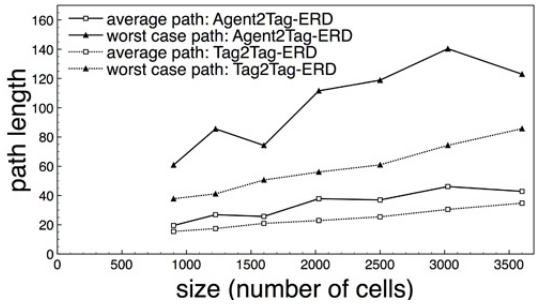

Fig. 6. Effect of changing the area size.

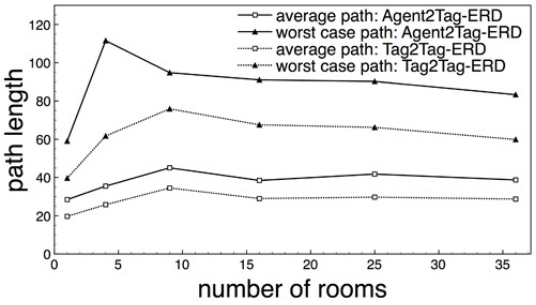

Fig. 7. Effect of changing the number of rooms.

ber of faulty links, the gaps between the three exploration algorithms increase, especially in the case of the worst-case (longest) evacuation path. As inter-tag links deteriorate, the Ants algorithm continues to identify short evacuation paths, because its agents roam the network multiple times, whereas the other two algorithms are only able to discover suboptimal paths.

In the following section, we will investigate in depth the performance of Brick&Mortar, combined with our two route discovery schemes, as we vary a number of characteristics of the area topology.

VI. Impact of area topology on evacuation route discovery

In this section, we set up a simulation framework to investigate the impact of various topological features of the area on the discovery of evacuation routes. In particular, we vary the number of cells, the number of rooms, the number of obstacles and the number of exits. We also consider three different area types (see Figure 5): i) *Office:* area inspired by a real building plan, with corridors, offices and an open-space area; ii) *Collapsed:* area in which a severe event occurred (e.g., earthquake) and disrupted the normal plan of the building and iii) *Series:* a long corridor which traverses several rooms; this scenario is inspired by a mine or another underground map in which each room has two doors, one connecting it to the previous room, and one to the next room.

For space reasons, we combine our route discovery mechanisms with one exploration algorithm, Brick&Mortar, which, as shown in the previous section, is typically faster than MDFS and Ants in exploring the area and identifying victims, so even if the discovered routes will not be shorter than Ants, the overall time to reach the exits is smaller.

In each of the following simulations, we vary the values of one parameter, and assign default values to the remaining ones. The default values are an Office area of 2500 (50×50) cells, with 4 rooms, no obstacles and 5 emergency exits, which is explored by 5 Brick&Mortar agents. Our goal is to compare Agent2Tag-ERD and Tag2Tag-ERD in terms of two performance metrics: 1) the average length of evacuation paths and 2) the length of the worst-case (longest) evacuation path. To evaluate the average and worst case path lengths, we assume that victims are found in all cells of the area, and evacuation paths are formed between each cell and one of the emergency exits.

Impact of area size: Figure 6 measures the average and worst-case path lengths of Agent2Tag-ERD and Tag2Tag-ERD, as we vary the number of cells in the area. As one would expect, the path lengths increase as we increase the area size for both evacuation route discovery mechanisms. Whereas the difference between the average path lengths is not significant (see lower two line-plots), the use of inter-tag communication is shown to reduce significantly the length of the worst-case evacuation paths (see higher two line-plots). Hence, Tag2Tag-ERD is much more efficient in addressing the worst-case scenario where a victim is found far away from an emergency exit. The large gap between Agent2Tag-ERD and Tag2Tag-ERD in the worst-case scenario is observed for most area sizes.

Impact of rooms: Figure 7 shows an interesting trend in the interval between 1 and 10 rooms, while with more than 10 rooms the performance of Agent2Tag-ERD and Tag2Tag-ERD remains constant. The average and worst-case paths with 1 room are short, because the area is an open space, and agents do not have to navigate around room walls to access internal room cells. Initially, as we increase the number of rooms, the room walls act as long obstacles, leading to longer evacuation paths. As we increase the number of rooms further (> 10) what used to be long walls of a few rooms, become smaller walls interrupted by many room doors. Hence, although the total number of wall cells

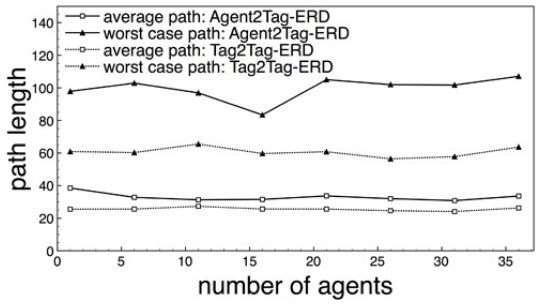

Fig. 8. Effect of changing the number of agents.

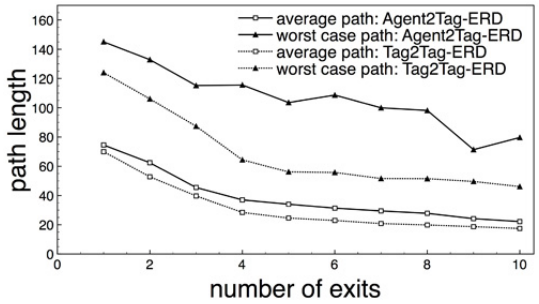

Fig. 9. Effects of changing the number of exits.

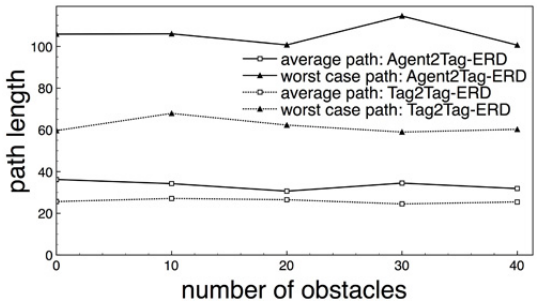

Fig. 10. Effect of changing the number of obstacles.

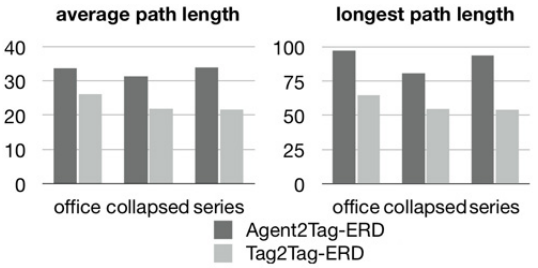

Fig. 11. Effects of changing the type of scenario.

increases (leading to longer paths), access to internal room cells becomes easier (leading to shorter paths). Hence the cumulative effect is that agents tend to find paths of similar length as we increase the number of rooms from 10 to 35.

Impact of agents: Figure 8 shows that the number of agents that explore the area and look for evacuation paths, has little effect on the length of evacuation paths discovered eventually when the exploration process is finished. This is not to be confused with the fact that, during exploration, the more the agents the faster these paths will be discovered. As observed in most of the previous graphs, using or not inter-tag communication does not significantly improve the average path length. On the other hand, the fact that Tag2Tag-ERD uses inter-tag communication makes it much more efficient than Agent2Tag-ERD, in evacuating victims that are far from emergency exits.

Impact of emergency exits: As one would expect, Figure 9 shows that the more the emergency exits, the shorter the evacuation paths to them. Inter-tag communication only slightly improves the average path length - an observation that is consistent with most of the previous graphs. In terms of the worst case scenario (longest evacuation paths), Tag2Tag-ERD tends to take better advantage of adding more emergency exits than Agent2Tag-ERD. The reason is that it always manages to find the shortest paths to these exits, independent of paths that agents followed whilst exploring the area.

Impact of obstacles: In Figure 10 we can see that obstacles (e.g., desks in the middle of rooms) do not greatly influence the length of evacuation routes, because they typically occupy at most one cell.

Impact of room layout: Finally, in Figure 11 we used the default values of cells, rooms, obstacles, agents and emergency exits, and only varied the layout of rooms in the area, leading to three different scenarios: Office, Collapsed and Series (an example of each one can be seen in Figure 5). Introducing inter-tag communication is more beneficial in the Series scenario, followed by Office and Collapsed. In general, when considering all three scenarios, Tag2Tag-ERD is up to 35% more efficient than Agent2Tag-ERD in building evacuation paths on average, and up to 45% more efficient in finding evacuation paths to remotely located victims.

VII. Related Work

Choset [4] presents a survey on exploration (or coverage) algorithms and distinguishes them into *off-line* and *on-line*. In the former, agents are previously provided with a map of the area to explore, while in the latter, also called *sensor-based*, no assumption is made concerning the availability of an environmental map for the agents. A subclass of *on-line* exploration algorithms assumes that agents are able to coordinate their movements using direct agent-to-agent communication. For example, Kong et al. [5] assume perfect communication among agents to coordinate them and cover the area using a Boustrophedon technique. Rekleitis et al. [6] try to improve the exploration process by mapping the environment while the area is covered by two robots which are always able to communicate with each other. Finally, Batalin et al. [7] focus on agent dispersion and propose two algorithms to make the agents move away from each other when they are in sensing range using wireless communication. Howard et al.[8] present a general approach to exploring a building, finding objects and reporting them back to the human personnel using communication between robots, but they also show with extensive real experiments how wireless communication can be patchy and unreliable. For this reason, they try to cope with that problem adapting their algorithm to let robots be sometimes disconnected from the base station and from each other. Moreover, Park

et al. [9] and Kotz et al. [10] confirm with their experiments that agent-to-agent communication cannot be taken as an assumption. Radio propagation is widely accepted to be asymmetric, especially in the case of longer links, and non-isotropic, which means that the received signal, at a given distance from the sender, is not the same in all directions. Direct agent-to-agent communication through long-range wireless links is unreliable especially in indoor environments. For example, the range of a Tmote Sky node with integrated onboard antenna may decrease from 125m outdoors to 50m indoors. This motivates the use of agent-to-tag communication for the purpose of area exploration. The exploration algorithms that we considered in Section III of this paper represent a subclass of *on-line* algorithms that rely on agent-to-tag communication, following the paradigm first proposed by Koenig, Liu and Svennebring [11], [2] with their Ants approach. They let agents coordinate indirectly by first instrumenting the environment with tags, and subsequently reading and updating the state of the deployed tags. Agents communicate with tags when they visit the cells where tags are deployed, hence agent-to-tag communication relies on short and therefore reliable links. A more detailed description of these exploration algorithms is provided in [1]. Another use of the tags could be the storage of information about dangers in the area, which robots or human responders could use to avoid encountering hazards [12]. Hähnel et al. [13] showed how a robot can use RFID tags, which are already placed in an area, to localise itself and navigate through the rooms. Kleiner et al.[14] and Ziparo et al.[15] presented a robot, which is able to autonomously drop RFID tags in the environment and use them to explore the area. To the best of our knowledge, no *on-line* algorithms have been presented so far with the aim of combining both the exploration of an area with the dynamic creation and maintenance of short *evacuation routes* connecting emergency exits to critical cells in the area.

VIII. Conclusion

In this paper, we considered the problem of discovering evacuation paths in emergency areas, whilst exploring them. We proposed two distributed mechanisms that agents can use to discover paths from victims to emergency exits: Agent2Tag-ERD, which relies on agent-to-tag communication, and Tag2Tag-ERD, which exploits inter-tag communication. Since the discovery of evacuation paths is conducted in parallel with the exploration task, we studied the interplay between exploration algorithms and the two ERD algorithms.

The conclusions of our study are as follows: 1) The choice of exploration algorithm largely affects the time when victims are found and evacuation paths are first discovered. Brick&Mortar tends to be faster than MDFS and Ants in that respect. 2) Without inter-tag communication (Agent2Tag-ERD), the choice of the exploration algorithm also affects the length of discovered evacuation paths. Algorithms that tend to revisit the same cells multiple times (Ants and MDFS) gradually discover shorter evacuation paths than faster exploration algorithms (Brick&Mortar) that avoid going back to the same cells. 3) With inter-tag communication (Tag2Tag-ERD), all exploration algorithms have the same performance both in terms of the average and worst-case evacuation paths. 4) For a given exploration algorithm, Agent2Tag-ERD yields longer evacuation paths than Tag2Tag-ERD. The benefits of inter-tag communication become more obvious (up to 50%) when one considers the length of the longest path to remote victims. 5) The quality of evacuation paths depends on the topological features of the area. The dependence is higher on the area size, the layout of rooms and the number of exits, and lower on the number or rooms and the number of obstacles in the rooms.

IX. Acknowledgments

Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-06-1-3003. The U.S. Government is authorized to reproduce and distribute reprints for Government purpose notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

References

[1] E. Ferranti, N. Trigoni, and M. Levene, "Brick&Mortar: An On-Line Multi-Agent Exploration Algorithm," in *ICRA07*. IEEE Press, April 2007, pp. 761–767.

[2] J. Svennebring and S. Koenig, "Building Terrain-Covering Ant Robots: A Feasibility Study," *Autonomous Robot*, vol. 16, no. 3, pp. 313–332, May 2004.

[3] C. T. H., C. E. Leiserson, and R. L. Rivest, *Introduction to algorithms*. Cambridge, MA, USA: MIT Press, 1990.

[4] H. Choset, "Coverage for robotics - A survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 113–126, 2001.

[5] C. S. Kong, N. A. Peng, and I. Rekleitis, "Distributed Coverage with Multi-Robot System," in *ICRA06*. IEEE Press, May 2006, pp. 2423–2429.

[6] I. Rekleitis, G. Dudeck, and E. Milios, "Multi-robot Exploration of an Unknown Environment, Efficiently Reducing the Odometry Error," in *IJCAI97*. Morgan Kaufmann, August 1997, pp. 1340–1345.

[7] M. A. Batalin and S. G. Sukhatme, "Spreading Out: A Local Approach to Multi-robot Coverage," in *DARS02*, June 2002, pp. 373–382.

[8] A. Howard, L. E. Parker, and G. S. Sukhatme, "Experiments with a Large Heterogeneous Mobile Robot Team: Exploration, Mapping, Deployment and Detection," *The International Journal of Robotics Research*, vol. 25, no. 5–6, pp. 431–447, December 2006.

[9] J. Park, S. Park, D. Kim, P. Cho, and K. Cho, "Experiments on radio interference between wireless LAN and other radio devices on a 2.4 GHz ISM band," in *VTC03*. IEEE Press, April 2003, pp. 1798–1801.

[10] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott, "Experimental Evaluation of Wireless Simulation Assumptions, Tech. Rep. Technical Report TR2004-507, Dartmouth College Computer Science, June 2004.

[11] S. Koenig and Y. Liu, "Terrain Coverage with Ant Robots: a Simulation Study," in *AGENTS01*. ACM Press, May 2001, pp. 600–607.

[12] G. Alankus, N. Atay, C. Lu, and O. Bayazit, "Adaptive Embedded Roadmaps For Sensor Networks," in *ICRA07*. IEEE Press, April 2007, pp. 761–767.

[13] D. Hähnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose, "Mapping and Localization with RFID Technology," in *ICRA04*. IEEE Press, April 2004, pp. 1015–1020.

[14] A. Kleiner, J. Prediger, and B. Nebel, "RFID Technology-based Exploration and SLAM for Search And Rescue," in *IROS06*. IEEE Press, October 2006, pp. 4054–4059.

[15] V. Ziparo, A. Kleiner, B. Nebel, and D. Nardi, "RFID-Based Exploration for Large Robot Teams," in *ICRA07*. IEEE Press, April 2007, pp. 761–767.