MRESim, a multi-robot exploration simulator for the Rescue Simulation League

Victor Spirin[1], Julian de Hoog[2], Arnoud Visser[3], and Stephen Cameron[1]

[1] Oxford University, United Kingdom
[2] University of Melbourne, Australia
[3] Universiteit van Amsterdam, The Netherlands

Abstract. This paper describes MRESim, a multi-robot exploration simulator which aims to provide a middle ground between the RoboCup Agent and Virtual Robot competitions. A detailed description of this new infrastructure is provided, followed by examples and case studies of successful research outcomes arising from the use of MRESim. Our work on MRESim won the 2014 Infrastructure competition of the RoboCup Rescue Simulation League

1 Introduction

The RoboCup Rescue competitions provide benchmarks for evaluating robot platforms' usability in disaster mitigation. Research groups should demonstrate their ability to deploy a team of robots that explore a devastated area and locate victims. RoboCup is moving towards longterm goals of sophisticated resource allocation in disaster situations, both at a large scale (the Agents competition) and at a small scale (the Virtual Robots competition). Each year the benchmarks are made more challenging to accommodate and encourage progress by all teams. The Infrastructure competition showcases innovations that have enabled extensions of these benchmarks.

The Infrastructure competition was started in 2004 with the aim of promoting the development of new tools to improve rescue simulation. The simulation of various disaster situations turns out to be complicated and difficult to validate. Therefore, the infrastructure competition was launched to promote the maintenance and development of simulation environments. For example, the fire simulator [24] was developed by the winner of the infrastructure competition in 2004. Another nice example of a component developed in the infrastructure competition is the flood simulator [27]. Recently, an extension towards flying robots has been proposed, both in the Virtual Robot [7] and Agent competitions [9].

The Amsterdam Oxford Joint Rescue Forces, and their predecessor the UvA Rescue Team, has participated several times in the Infrastructure competition. In 2010 the simulator of the Virtual Robot competition was extended with a realistic response of laser scanners to smoke; a circumstance which is quite common in disaster situations. The response of the laser scanners was validated in a number of experiments in a training center of the Dutch fire brigade [8]. In

This is the author's version. The final publication will become available at Springer.

2011 a model of a humanoid robot was introduced in USARSim, which made it possible to model one of the robots in the RoboCup@Home League [31]. In 2012 a validated flying robot was introduced to USARSim and it was demonstrated that such a robot allows for fast exploration of disaster areas, while creating a visual map of the area [7]. This resulted in the UvA Rescue team winning the Infrastructure award in 2012.

What was still missing inside the Rescue Simulation League is an environment which is a common ground for both the Agent and Virtual Robot competitions. Inside the Agent competition the focus is on the coordination of rescue teams [20,33]. The competition consists of a simulation environment which resembles a city after an earthquake. Teams of fire brigade, police and ambulance team agents try to extinguish fires and rescue victims in the collapsed buildings. In contrast, inside the Virtual Robot competition the focus is on sensor-data fusion on maps automatically generated by teams of robots which explore inside buildings. The multi-robot exploration simulator MRESim[4], described in this paper, is proposed as a middle ground between the Agent and Virtual Robot competitions.

In the following section the background behind this new infrastructure is given, followed by a section which gives an overview of the research performed with this multi-robot exploration simulator.

2 Simulator

2.1 Simulation cycle

MRESim is a discrete-time simulator. Unlike the existing Virtual Robot competition simulator USARSim, which uses a three-dimensional representation, MRESim works from a two-dimensional grid representation. In each time step, the simulator moves the robots to their new positions, updates their sensor information, simulates communication between the robots and outputs the current state of the simulation. The ordering of the events that happen in each time step are outlined in Algorithm 1. This is equivalent with the discrete-time step procedure in the simulator environment of the Agent competition. All of the robots process their next steps simultaneously. Since this tends to be the most computationally intensive part of each time step, planning robot movements in parallel allows the simulator to take advantage of modern multi-core CPUs. Other events in each time step are processed sequentially.

2.2 User interface and environments

The user interface (Fig. 1) contains a panel on the right for each robot with toggle buttons to visualize the information which is the basis of each robot's decisions. Example of information that can be made visible is the following: location; path; free space; safe space; frontiers; communication range; skeleton

[4] https://github.com/v-spirin/MRESim

```
   Input: Set R of robots
   // Simulate movement, range data
 1 foreach r_i ∈ R do
   |  // described in Section 2.5
 2 |  while distance_moved(r_i) < max_speed(r_i) do
 3 |  |  nextStep = r_i.takeStep();
 4 |  |  if isValid(nextStep) then
 5 |  |  |  move(r_i, nextStep);
   |  |  |  // described in Section 2.4
 6 |  |  |  rangeData = findRangeData(r_i, nextStep);
 7 |  |  |  r_i.receiveRangeData(rangeData);
 8 |  |  else
 9 |  |  |  r_i.setError(true);
10 |  |  |  break;
11 |  |  end
12 |  end
13 end
   // Simulate communication, described in Section 2.6
14 foreach r_i ∈ R do
15 |  foreach r_j ∈ R, i ≠ j do
16 |  |  if isInRange(r_i, r_j) then
17 |  |  |  r_i.receiveMsg(r_j.createMessage());
18 |  |  |  r_j.receiveMsg(r_i.createMessage());
19 |  |  end
20 |  end
21 end
   // Complete cycle
22 logData();
23 updateGUI();
```

Algorithm 1: A single simulation cycle

and potential rendezvous points; exact rendezvous points; potential rendezvous points through obstacles; exact rendezvous points through obstacles. There are further toggle buttons for the environment walls and the team hierarchy at the bottom right, and a run may at any time be paused, continued, or stopped using the green buttons at the bottom. This means that it is very easy to examine specific aspects of any supported exploration approach. In addition, simulation logs produced by MRESim can be replayed in the simulator; this means that each simulation run can be analyzed in detail after the run has completed.

Environments in MRESim can be uploaded from text-based or PNG files, as can be found on dataset repositories like Radish[5]. The environments are occupancy grid models, with each cell being either free or an obstacle.

[5] http://radish.sourceforge.net/

Input: Robot r_i, simulator settings $simConfig$

```
// Replan if necessary, then retrieve the next path point
if r_i.timeToReplan or r_i.getPath().isEmpty() then
    simConfig.getExplorationStrategy().replan(r_i);
end
return r_i.getPath().getNextPoint();
```

Algorithm 2: Description of the agent takeStep method

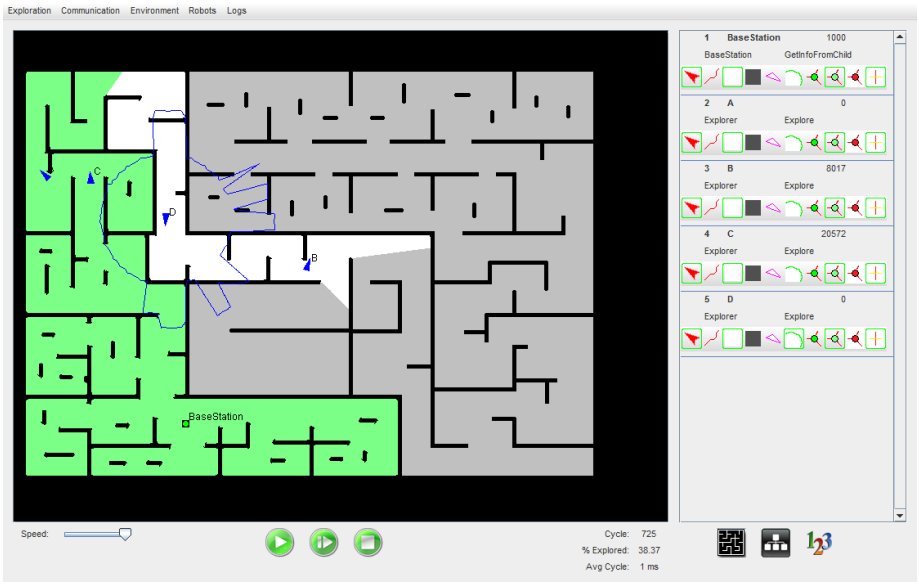

Fig. 1. Screenshot of MRESim. Obstacles are shown in black (—), unexplored area is shown in grey (▭), area known at base station is shown in green (▭). The blue arrows (►) represent exploring robots, and the communication range of robot D is shown as the blue polygon (▭).

2.3 Planning and Movement

In every time step, each robot has to decide where to move next. The next destination is decided by the robot by calling the *takeStep* method (line 3 of Alg. 1). If the robot has a path to its destination, and has replanned the path recently, then it simply continues on the current path by retrieving the next path point. Otherwise, it calls the corresponding *replan* method of the exploration strategy assigned to the robot team to generate a new path (line 2 of Alg. 2). The exploration strategy then selects the best destination for the robot, and generates a path to the destination. The exploration strategy can use all the information known to the robot when making that decision — including the information about last known locations of the robot's teammates and agreements made in previous communication with other robots or the base station. Note that this

information is not globally available, but communicated when in range. When no recent information is available, reasonable estimates are made.

Each robot r can move a maximum distance of d_r in each time step (line 5 of Alg. 1). This is defined as the robot "speed", and can be set up to be different for each robot, allowing for heterogeneous teams of robots to be created. By default, this parameter is set for each robot to $DEFAULT\_SPEED = 3$, configured in the *Constants* class of the *config* package. In each time step, each robot moves a maximum distance of d_r along the path generated in the planning stage (see line 2 in Algorithm 2). If the robot reaches the planned destination before d_r is exhausted, it will not move any further in that timestep.

2.4 Sensing and Mapping

Once a robot has taken a step, the simulator provides it with sensor data at its new location (lines 6-7 of Alg. 1). This sensor data is generated using raytracing from the robot's location at 1-degree intervals in the 180-degree field of view of the robot (the same field of view as many real laser scanners, as for instance the SICK LMS200). A maximum sensor range can be configured for each robot, allowing for heterogeneity of sensors across the team. An array of 181 measurements is returned to the robot. The measurements are assumed to be noise-free (see §2.7 for more detail).

The robot subsequently turns this sensor data into a polygon of free space, detecting obstacles where two points are sufficiently close to one another and below the sensor's range limit. This free space and obstacle detection is maintained in an occupancy grid. When robots are within communication range of one another, they can decide to exchange their local maps in the form of occupancy grids.

2.5 Path planning

Several path planning algorithms have been implemented in the MRESim simulator. The most straightforward one is the A* algorithm [11], operating directly on the occupancy grid representation of the map. While A* generates optimal paths, it can be very slow with a sufficiently high resolution of the occupancy grid. In an attempt to overcome this problem, we have implemented several more efficient path planning algorithms. One of them is Jump Point Path search [10], which is a modification of the A* algorithm which significantly improves the performance in environments with large open areas. However, complex paths can still take several hundred milliseconds to compute.

We found that using A* search on a combination of a simple topological map that captures the connectivity between regions and the underlying occupancy grid map can significantly improve performance over the other implemented approaches. In each time step, if the map of the environment has been updated, we can generate a new topological map as follows. First, we perform thinning on the occupancy grid map in order to obtain a skeleton of the free space. Then, we uniformly select a set of nodes from the skeleton. Each point of the occupancy

grid is then assigned to the nearest node. The set of nodes and edges connecting the nodes then represents a simple topological map of the environment. An illustration of the process is given in Fig. 2. We can then plan a path between any two points on the occupancy grid map by using the A* algorithm to find a path in the graph between the two corresponding nodes on the topological map, and using A* on the occupancy grid map to find paths from start and finish locations to their corresponding node locations. Generating paths then becomes very computationally efficient, which improves the speed of running the simulation and can increase the performance of exploration strategies by allowing them to calculate more accurate path lengths, instead of relying on crude approximations.

2.6 Communication

MRESim supports a variety of communication models. In principle messages from both robots are exchanged (as indicated in lines 17 and 18 of Alg. 1), but only when the robots are *inRange* (line 16 of Alg. 1). Simple models include a straight-line model (any robot within radius x is considered in range, regardless of obstacles) and a line-of-sight model (any two robots that can be connected by a line that doesn't hit an obstacle are in range). A more realistic communication model implemented in MRESim is a path loss model, originally proposed by Bahl and Padmanabhan [2]. This model is also used for simulating communication in the USARSim simulator used at RoboCup, and is considerably more realistic as it takes attenuation by walls into account. In this model, communication strength is calculated as follows:

$$S = P_{d_0} - 10 \times N \times log_{10}\frac{d_m}{d_0} \times \min(nW, C) \times WAF$$

where P_{d_0} is the signal strength at reference distance d_0, N is the rate of path loss, d_m is the distance, nW is the number of obstructing walls, WAF is the wall attenuation factor and C is the maximum number of walls where the attenuation factor needs to be considered.

Typical communication ranges using this model can be seen in Fig. 1.

2.7 Realism of Simulator Results

Clearly MRESim does not take into account a number of factors that any robot system in the real world would have to consider. The most significant ones are:

1. There is no sensor noise. In reality, wheel encoders provide inexact data, and laser range finders often have spurious measurements at either close or maximum range. Localisation remains a significant challenge in robotics, even if a number of techniques such as particle filters and scan matching show great promise.
2. Environments are two-dimensional and flat. In the real world this is almost never the case.

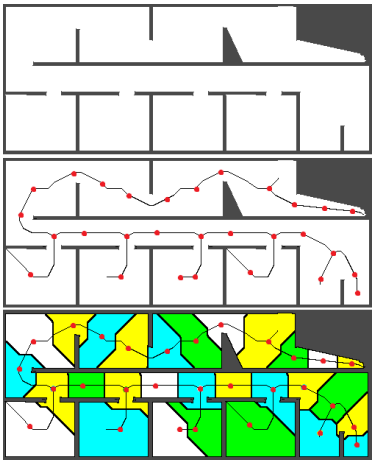

Fig. 2. Illustration of the process used by MRESim to generate topological maps from occupancy grid maps. On top, we see an occupancy grid of a partially explored environment, shown in white. In the middle, we apply "thinning" to obtain a skeleton, and sample uniformly points, shown in red (-•-), from the skeleton to act as nodes in the topological graph. In the bottom picture, we map each point of the occupancy grid map to the nearest node.

3. The simulator is discrete-time. Real robots would each run their own (multi-thread) processes, take different amounts of time to do the required processing, and would not always move the same distance in each time segment.
4. Communication in reality is highly variable and very difficult to predict. Nevertheless, for purposes of quick, simple, controlled, and repeatable comparison of exploration algorithms, MRESim remains a useful tool.

In the next section we give an overview of published work that used MRESim to evaluate exploration strategies. In some of the work, results are compared with those obtained from high-fidelity simulators such as USARSim, as well as with results obtained on real robots. Those experiments suggest that results obtained in MRESim have much in common with the results obtained in real life experiments.

3 MRESim as benchmark

A number of exploration strategies have already been implemented in MRESim and are available "out-of-the-box". This makes it a potentially useful tool for benchmarking various exploration strategies against each other. According to [19], the performance of most algorithms is compared using the following three methods, from the best to the worst:

1. Using the same implementation that was used in other work.

2. Using a custom implementation, created from descriptions of the algorithm in published work.
3. Simply taking the results from those in other papers, without re-running the algorithm.

According to [1], most algorithm comparisons in robotics are currently conducted using the second approach. Publishing of the team's source-code, as done inside the Rescue Simulation League, made it possible to reimplement existing algorithms and to make a fair comparison (see for instance [28,1]). Using MRESim, the first approach can be used more often:

1. Results of using different exploration strategies can be directly compared with each other.
2. Existing implementations of exploration strategies can be used.

3.1 Studies based on MRESim

MRESim was first mentioned in [13]. In this study the effect of robots with a relay role was studied, when exploration was needed outside the direct communication range from the base station. This study inspired several other researchers [3,5,6] to base the coordination decisions on multiple criteria.

Important for the coordination of the explorer and relay role is the selection of an appropriate rendezvous point [15]. When those points are selected near gateways, the locations inside indoor environments become important junctions for a navigation algorithm. This work was followed up by several researchers [23,21,25] (although the concept of rendezvous points has already been known for a long time [26]). In addition, in [29] the possibility of communicating through obstacles, such as thin walls, was introduced into the planning.

Dividing the work into explorer and relay roles is already valuable, but sometimes the explorer finds a dead-end. This is an example of a situation where it is beneficial to switch roles dynamically using the *role swap rule*, as described in [14]. Other researchers have validated this results with real robots, such as [18].

In the study [35], the exploration indoors is extended to open areas which can be encountered in outdoor scenarios. The robots still use rendezvous points, but no longer near gateways. Instead, the robots divide the work into sectors and meet at the sector boundaries. This work has motivated several other studies [4,17,34].

The algorithms implemented in MRESim have been validated with Pioneer robots [16], although it is difficult to scale indoors to extensive environments with large robot teams. The MRESim environment was discussed in several dissertations [12,34,22] and a workshop contribution [32].

The latest extension are robot teams which can adjust their exploration strategy based on the information need of the base station [30]. When it is important that the base station gets timely updates more resources are allocated towards the relay role; when fast exploration is needed more resources are allocated towards the explorer role.

4 Discussion and Future Work

MRESim is a simulation environment with a well chosen level of detail, as discussed in 2.7. When the algorithms developed inside MRESim are applied in the real world issues will arise, which can be studied in separation inside the simulation. Precisely those issues could be identified as directions for future research.

In addition, the simulator itself can be extended in a number of ways to increase its realism and allow for using it to study a wider range of research problems:

1. **Variable terrain**. In real applications, the terrain of the environment being explored can vary significantly, affecting the speed of robots navigating over such terrain. For example, robots are likely to be able to travel much faster just outside a partially-collapsed building, than inside where there is likely to be a lot of rubble. It may then be possible to improve the exploration speed by reducing the amount of travel over rough terrain, particularly for the relays.
2. **Additional communication channels**. It may be possible to use low-frequency, low-bandwidth, high range radio communication channels in disaster scenarios to transfer some control information between robots, as suggested in [29]. (We are about to test this hypothesis experimentally.)
3. **Robustness against Failing Robots**. Robots may fail for a number of reasons (and often do). In greedy approaches this does not affect the failed robot's teammates, which continue exploring as if nothing happened. In some other approaches, meetings between robots at pre-agreed times and locations can be planned for explicitly. In those cases, the failure of robots is a scenario that must be dealt with carefully. A useful extension to the simulator could therefore be the ability to specify scenarios that include robot failure, making it a useful tool to evaluate the robustness of exploration strategies to individual robot failure.
4. **Dynamic Environments**. In many robotics applications, the environment may change as the exploration effort progresses. A possible extension to the simulator would be to specify how the environment should change over time, or allow for random changes to the environment. This is precisely where the current Agent competition accelerates and it would be nice if for example part of the fire, earthquake or flood simulation could be incorporated in MRESim.
5. **Prior knowledge**. In practice, some knowledge about the environment may be available to the team before the start of the mission, even though this information may not be accurate.

5 Conclusions

The original motivating questions for this research were: How can a team of robots be coordinated to explore a previously unknown and communication-limited environment as efficiently as possible; and how can new information

obtained by this team be gathered at a single location as quickly and as reliably as possible?

These are precisely the research questions studied in both the Agent and the Virtual Robot competition of the RoboCup Rescue Simulation League. The studies performed with the simulation environment MRESim are well recognized, both inside the RoboCup community and outside. MRESim is applied to compare several coordination algorithms and could be easily extended with other coordination algorithms. In this paper some ideas for future research are given, but many other extensions are possible.

References

1. Amigoni, F., Schiaffonati, V.: Good experimental methodologies and simulation in autonomous mobile robotics. In: Magnani, L., Carnielli, W., Pizzi, C. (eds.) Model-Based Reasoning in Science and Technology, Studies in Computational Intelligence, vol. 314, pp. 315–332. Springer Berlin Heidelberg (2010)
2. Bahl, P., Padmanabhan, V.N.: Radar: An in-building rf-based user location and tracking system. In: Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM). vol. 2, pp. 775–784 (2000)
3. Basilico, N., Amigoni, F.: Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. Autonomous robots 31(4), 401–417 (2011)
4. Bayram, H., Bozma, H.I.: Decentralized network topologies in multirobot systems. Advanced Robotics 28(14), 967–982 (2014)
5. Capitan, J., Spaan, M.T., Merino, L., Ollero, A.: Decentralized multi-robot cooperation with auctioned pomdps. The International Journal of Robotics Research 32(6), 650–671 (2013)
6. Cepeda, J.S., Chaimowicz, L., Soto, R., Gordillo, J.L., Alanís-Reyes, E.A., Carrillo-Arce, L.C.: A behavior-based strategy for single and multi-robot autonomous exploration. Sensors 12(9), 12772–12797 (2012)
7. Dijkshoorn, N., Visser, A.: Urban Search and with micro aerial vehicles. In: Proc. CD of the 16th RoboCup International Symposium (June 2012)
8. Formsma, O., Dijkshoorn, N., van Noort, S., Visser, A.: Realistic Simulation of Laser Range Finder Behavior in a Smoky Environment. In: RoboCup 2010: Robot Soccer World Cup XIV. Lecture Notes on Artificial Intelligence series, vol. 6556, pp. 336–349. Springer, Heidelberg (June 2011)
9. Gohardani, P.D., Ardestani, P., Mehrabi, S., Yousefi, M.A.: Flying agent: An improvement to urban disaster mitigation in robocup rescue simulation system. Mechatronics Research Laboratory, Qazvin, Iran (2013)
10. Harabor, D.D., Grastien, A.: Online graph pruning for pathfinding on grid maps. In: 25th AAAI Conference on Artificial Intelligence (2011)
11. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics 4(2), 100–107 (1968)
12. de Hoog, J.: Role-Based Multi-Robot Exploration. Ph.D. thesis, University of Oxford (May 2011)
13. de Hoog, J., Cameron, S., Visser, A.: Role-based autonomous multi-robot exploration. In: Proceedings of the International Conference on Advanced Cognitive Technologies and Applications (Cognitive 2009) (November 2009)

14. de Hoog, J., Cameron, S., Visser, A.: Dynamic team hierarchies in communication-limited multi-robot exploration. In: Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR 2010) (July 2010)
15. de Hoog, J., Cameron, S., Visser, A.: Selection of rendezvous points for multi-robot exploration in dynamic environments. In: International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS) (May 2010)
16. de Hoog, J., Jiménez-González, A., Cameron, S., Martínez de Dios, J.R., Ollero, A.: Using mobile relays in multi-robot exploration. In: Proceedings of the Australasian Conference on Robotics and Automation (December 2011)
17. Hourani, H., Hauck, E., Jeschke, S.: Serendipity rendezvous as a mitigation of exploration's interruptibility for a team of robots. In: Robotics and Automation (ICRA), 2013 IEEE International Conference on. pp. 2984–2991. IEEE (2013)
18. Jiménez-González, A., Martínez-de Dios, J.R., Ollero, A.: An integrated testbed for cooperative perception with heterogeneous mobile and static sensors. Sensors 11(12), 11516–11543 (2011)
19. Johnson, D.S.: A theoretician's guide to the experimental analysis of algorithms. Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges p. 215–250 (2002)
20. Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., Shimada, S.: RoboCup Rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In: IEEE Conf. on Man, Systems, and Cybernetics(SMC-99) (1999)
21. Luo, C., Ward, P., Cameron, S., Parr, G., McClean, S.: Communication provision for a team of remotely searching uavs: A mobile relay approach. In: 2012 IEEE Globecom Workshops (GC Wkshps). pp. 1544–1549 (2012)
22. Martin, A.: A Framework for the Development of Scalable Heterogeneous Robot Teams with Dynamically Distributed Processing. Ph.D. thesis, University of Toronto (2013)
23. Meghjani, M., Dudek, G.: Combining multi-robot exploration and rendezvous. In: 2011 Canadian Conference on Computer and Robot Vision (CRV). pp. 80–85 (2011)
24. Nüssle, T.A., Kleiner, A., Brenner, M.: Approaching urban disaster reality: The ResQ firesimulator. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004: Robot Soccer World Cup VIII. Lecture Notes in Computer Science, vol. 3276, pp. 474–482. Springer (2004)
25. Pham, V.C., Juang, J.C.: An improved active slam algorithm for multi-robot exploration. In: SICE Annual Conference (SICE), 2011 Proceedings of. pp. 1660–1665. IEEE (2011)
26. Roy, N., Dudek, G.: Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations. Autonomous Robots 11(2), 117–136 (2001)
27. Shahbazi, H., Abdolmaleki, A., Salehi, S., Shahsavari, M., Movahedi, M.: Robocup rescue 2010 – rescue simulation league team description paper - brave circles - infrastructure competition. In: Proc. CD of the 14th RoboCup International Symposium (July 2010)
28. Shayesteh, M., Salamati, M., Taleghani, S., Hashemi, A., Hashmi, S.: Mrl team description paper for virtual robots. Team Description Paper for the 2014 RoboCup competition (July 2014)
29. Spirin, V., Cameron, S.: Rendezvous through obstacles in multi-agent exploration. In: Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR 2014) (October 2014)

30. Spirin, V., Cameron, S., de Hoog, J.: Time preference for information in multi-agent exploration with limited communication. In: 14th Towards Autonomous Robotics and Systems conference (TAROS 2013) (August 2013)
31. van Noort, S., Visser, A.: Extending virtual robots towards robocup soccer simulation and @home. In: RoboCup 2012: Robot Soccer World Cup XVI, Lecture Notes in Computer Science, vol. 7500, pp. 332–343. Springer Berlin Heidelberg (2013)
32. Visser, A., de Hoog, J., Jiménez-González, A., Martínez de Dios, J.R.: Discussion of multi-robot exploration in communication-limited environments. In: Workshop "Towards Fully Decentralized Multi-Robot Systems: Hardware, Software and Integration" at the ICRA Conference (May 2013)
33. Visser, A., Ito, N., Kleiner, A.: Robocup rescue simulation innovation strategy. In: RoboCup 2015: Robot Soccer World Cup XIX. Lecture Notes in Artificial Intelligence, Springer (2015)
34. Wellman, B.L.: Cooperation paradigms for overcoming communication limitations in multirobot wide area coverage. Ph.D. thesis, The University of Alabama (2011)
35. Wellman, B.L., de Hoog, J., Dawson, S., Anderson, M.: Using rendezvous to overcome communication limitations in multirobot exploration. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (October 2011)