Evaluation of cooperative exploration strategies using full system simulation

Sebastian Kasperski and Johannes Wollenberg and Markus Eich
German Research Center for Artificial Intelligence
Mary-Somerville-Strasse 9
Bremen, Germany 28359

***Abstract*—This paper compares different state-of-the-art exploration strategies for teams of mobile robots exploring an unknown environment. The goal is to help in determining a best strategy for a given multi-robot scenario and optimization target. Experiments are done in a 2D-simulation environment with 5 robots that are equipped with a horizontal laser range finder. Required components like SLAM, path planning and obstacle avoidance of every robot are included in a full-system simulation. To evaluate different strategies the time to finish exploration, the number of measurements that have been integrated into the map and the development in size of the explored area over time are used. The results of extensive test runs on three environments with different characteristics show that simple strategies can perform fairly well in many situations but specialized strategies can improve performance with regards to their targeted evaluation measure.**

I. Introduction

A fundamental requirement for autonomous robots that is needed for almost any other thinkable task is the ability to navigate safely and efficiently within their working space. Many algorithms and techniques have been developed in this context, enabling robots to navigate safely around obstacles, find optimal paths within a given map, localizing themselves and keeping track of their current position within a map. Due to the development of Simultaneous Localization and Mapping (SLAM) algorithms, robots are finally able to create this map that represents all their knowledge of the surrounding world autonomously as they move within their environment. With all this integrated into an actual robot it became possible to put a robot anywhere and have it exploring its new environment, drawing a map - an internal model of the outside world - and using it to directly navigate toward places it has visited before, thus making one important step towards autonomy. Robotics today is taking the next step by making robots act together to solve more complex tasks, a field of research that still has a long way to go. Keeping that in mind, the task of exploring their environment together in a cooperative way may serve as a reasonable complex example for acting as a team. It requires to share collected information, keep track of the others, their decisions and possible goals and to include them into own decisions. Many strategies have been developed so far to guide teams of robots during this task, but yet none has proven to be the best. This work compares different state-of-the-art exploration strategies for teams of mobile robots in complex environments. The results of extensive tests using a very detailed simulation that includes most of the software run on a real robot are presented.

II. Related work

Besides the evaluation of new strategies to justify a proposed approach some more general publications on the evaluation and analysis of cooperation strategies have influenced this work. In [1] the authors analyze the advantage that robots can draw out of communicating with each other while performing a cooperative task. They compare different levels of communication based on three generic multi-agent tasks called “consume”, “forage” and “graze”, the latter being quite similar to the distributed exploration. Higher levels of communication are generally more complex and usually more expensive to implement. Evaluation is done using a rather abstract, grid-based simulation with discrete time, which does not respect robot motion restrictions or sensor limitations. An earlier approach to real-time strategy evaluation for a single robot can be found in [2], where several frontier-based strategies are evaluated against a reference strategy following a decision-theoretic approach. They found simple nearest-frontier based solutions to perform reasonably well in many realistic scenarios. A classification of cooperative exploration strategies is given in [3]. Different approaches are categorized by communication expenses, team architecture and synchronicity. Their cooperation strategy “MinPos” applies the widely used Wavefront-Propagation-Algorithm to assign robots to frontiers, thus spreading the robots in the area more efficiently.

A variety of other algorithms has been proposed that guide a team of robots to simultaneously explore an initially unknown terrain. The exploration of an unknown environment based on frontier cells is described in [4]. An exploration strategy usually describes some sort of metric and evaluation that is used to select one frontier cell out of many as the next movement target for the robot. A centralized strategy where one robot builds the map with sensor data received from all other robots in the team is given in [5]. The coordination is achieved using an auction-based system, where the exploring robots send “bids” to the central master to get a certain frontier assigned as a target. Evaluation is done in simulation by measuring how the coverage of the complete environment evolves over time. Another strategy is described in [6] that explicitly assigns frontiers to robots whenever one robot requests a new target. To keep two robots from exploring the same room or corridor, the explored environment is segmented and robots assigned to frontiers in different segments of the map. Due to this assignment, the algorithm strongly depends on the segmentation result. In an unstructured area that cannot be

978-1-4799-2722-7/13/$31.00

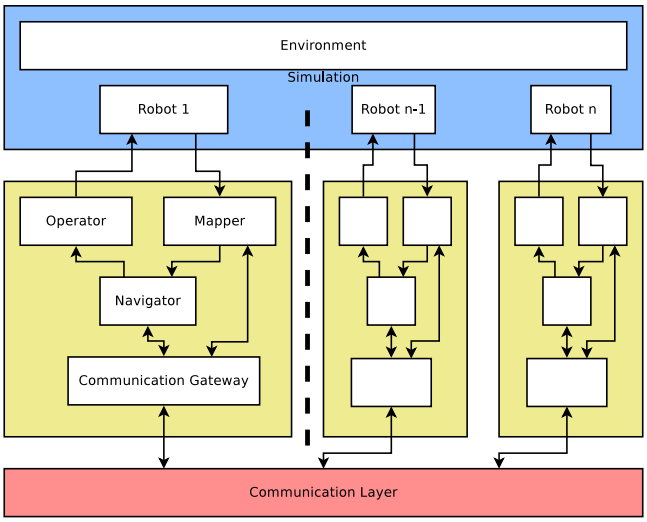

Fig. 1. **Schematic layout of our full system simulation:** The upper box represents the simulation environment with a number of simulated robots. The middle part shows the modules that are run on each real robot or once for each simulated robot respectively. A separate communication layer is required in case of real robots, but is not included in the simulation. Instead data is shared directly between communication gateways. More detailed simulations in the future might represent the communication layer as well.

partitioned the strategy reduces to nearest frontier exploration.

The rest of this paper is organized as follows: A description of the applied multi-robot simulation is given in Section III. Following this is a short description for each of the evaluated strategies in Section IV. Experiments that have been done to evaluate the strategies are described in Section V and results are presented in Section VI. Some general conclusions drawn from evaluation and an outlook towards future work are given in Section VII.

III. Full system simulation

Newly published cooperative exploration strategies are usually evaluated by their authors using specialized simulation and often setups of real robots in small environments [5], [7], [2], [3]. Time and state space are discretized in these simulations and the real-time performance of other required components like robot navigation, distributed mapping and communication is neglected. Of course this helps a lot in minimizing the influence of other components beside the actual strategy that are not to be evaluated, but also reduces the relevance of the results for real-world application. Evaluation on a team of real robots could make up for this drawback, but often only few robots and a small environment (office, constructed test area) are available, where there is no need for sophisticated cooperation. Additionally it is very time consuming and often nearly impossible to repeat experiments with the exactly same starting conditions many times. For these reasons we chose to put a selection of state-of-the-art exploration strategies to the test using a full-system simulation that includes all the software running on a mobile robot. The architecture of our setup is shown in Figure 1. For simulation of our robots with laser range finder we use “Stage” [8] (Figure 1, upper box), which allows to simulate multiple mobile robots moving in an arbitrary flat environment. Simulated robots are operated with the same motion commands and produce the same sensor output as real robots, so it is possible to evaluate strategies under most realistic conditions. This includes for example the time robots need to turn in place, as well as problems that arise when robots interfere with each other during the exploration process. Components running on the robots are started in separate runtime environments (Figure 1, middle boxes) for simulation. The module “Operator” controls the robot hardware and implements low level obstacle avoidance by maintaining a local obstacle map updated from laser scans and filtering control commands. The “Mapper” builds a global map by integrating laser scans from the robot’s own laser range finder as well as scans send by other robots shared via the inter-robot communication layer. All scans together with their relative pose information form a pose graph [9], [10] from which a grid map can be constructed on request of other modules. The “Navigator” module executes the exploration strategy and provides basic functionality like map updates and path planning to all strategies. This ensures that all strategies are executed at the same frequency and under equal conditions. The “Navigator” also checks whether the exploration target has been reached and realizes additional improvements like repetitive rechecking [2] for frontiers. The “Communication Gateway” provides sharing of data with other team members. For simple strategies only map data is shared between all instances of the “Mapper” module, but more advanced strategies can optionally share arbitrary data between “Navigator” instances, for example the robot’s current exploration target. On a real system communication between robots requires an additional communication layer to send data via WiFi. This is currently left out in simulation and the data shared directly between the “Gateway” instances. Communication between these modules is done using the “Robot Operating System” (ROS) [11], which also runs on our Pioneer robots and thus allows application of the complete system directly on our real robots.

IV. Strategies

This section describes the different strategies that form the basis of our experiments. At the core of basically every exploration algorithm is a frontier detection. A frontier is defined as the border between an area that has already been discovered to be free of obstacles and an unexplored area [4]. The actual implementation of this detection algorithm naturally depends on the underlying world representation. In commonly used grid cell representations of two-dimensional environments frontiers are defined as a set of connected frontier cells, which have been marked as free and have adjacent cells that are still unexplored. Within this context wavefront algorithms, that propagate a wavefront throughout the so far explored map starting at the robot’s current position, are frequently used. Two important advantages of these are that found frontier cells are not only in free space but also reachable from the robots current position and that frontier cells are found in order of their distance to the current position. Depending on the robots navigation algorithm it can also supply a shortest route to that cell, making it inherently more efficient.

The central question after the detection of frontier cells is how to assign robots to specific locations in order to improve the exploration result. Many approaches try to minimize the exploration time by evaluation of specific map locations and assigning robots to these positions according to this evaluation

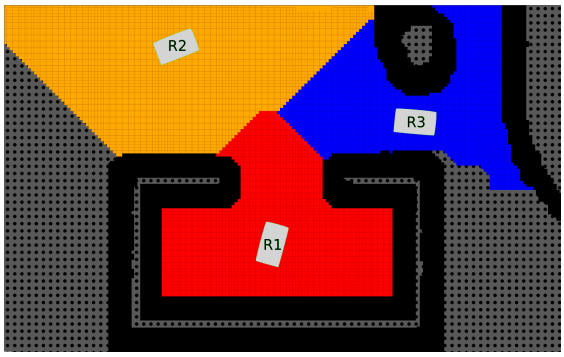

Fig. 2. **Multiple-Wavefronts-Algorithm:** A wavefront is started at every robot's position and opposing wavefronts stop at each others border. The picture shows an example, where the red wavefront started by robot R1 has stopped propagation without reaching unexplored terrain (gray area). This situation has to be handled separately.

process. Exploration strategies may emerge from intuitive motivations, like prioritizing the investigation of areas that, with high probability, yield a lot of new map information. Another idea is the minimization of the traveling cost in the frontier cell selection step. Moreover, domain-dependent knowledge may be incorporated, for example in the exploration of specific indoor environments [12].

In the following the evaluated exploration strategies in the simulation experiments are described. The nearest frontier approach serves as a reference strategy that needs no additional communication besides the map data, is simple to implement and computationally efficient. With MultiWave we propose a variant that includes other robots positions into the nearest frontier approach. Following these are two promising cooperation strategies taken from the literature that have been reimplemented and evaluated as well.

A. Nearest Frontier

This is the most simple frontier based exploration algorithm. Every robot moves toward the frontier that is the nearest to its current position. This very basic strategy tries to minimize the distance the robot has to travel through previously explored terrain, thus reducing overall path length as well as exploration time. Because every robot only considers its own position within the map, it is often called uncoordinated exploration. However in our case where the robots share their constructed map, it is not completely uncoordinated. Implicit coordination takes place via the shared world model causing robots to ignore areas explored by others. Once the robots are fairly distributed on the map this strategy performs quite good. In fact most other strategies are equivalent to a nearest frontier approach when the robots are widely spread across the map.

B. MultiWave

This strategy enhances the nearest frontier exploration in a way that it includes the positions of the other robots into the frontier detection. The basic algorithm is still the same, therefore keeping the approach distributed and asynchronous. A common problem with the uncoordinated approach is that two robots that are both near the same frontier will both move towards it, even with other frontiers present but far away. This is avoided by every robot starting multiple wavefronts simultaneously from its own and all other robot's positions, which have to be communicated additionally. The wavefronts are propagated at the same speed and stop at each other's borders so that every frontier cell can only be reached by one wavefront, namely the one that corresponds to the robot that is closest to that cell. This causes the robots to move into different directions wherever possible, a behavior that is generally desired during exploration.

A special situation like the one shown in Figure 2 may occur when one robot is locked up by the environment and the other robots so that its own wavefront cannot reach any frontier. The decision about what to do in this situation might depend on the evaluation criterion that should be optimized by the strategy. If the goal is to minimize the overall exploration time the robot should clearly keep on moving, for example to the nearest frontier cell by ignoring the other robots for now. On the other hand, if one wants to minimize traveling costs it might be better to just wait at the current position until the other robots move away and a new frontier becomes visible. This behavior might lead to situations where one or more robots do not join the exploration at all, especially if the environment is sufficiently small or narrow. From the perspective of efficiency this might be a desired behavior, therefore this strategy has been applied in two different variants. The standard version falls back to a plain nearest frontier when in a lock situation, while an additional version called "+Wait" will stop robots in locked situations until the lock is resolved.

C. MinPos

This recently published strategy takes a new approach on exploration by planning from frontiers to robots [3]. It is also decentralized and requires no other information than a shared map and the positions of the other robots. Here every robot evaluates its relative rank among the other robots in term of travel distance to each frontier. This is done by propagating multiple wavefronts like before, but starting at every frontier instead of every robot's position. A robot then starts moving toward the frontier for which it has the lowest rank. A nice feature of this approach is that the lock situation mentioned before is handled naturally, since the robots will be equally distributed among all available frontiers.

D. Decision-theoretic approach

While in case of a nearest frontier based exploration the only considered measure is the traveling cost, additional criteria can be integrated into exploration strategies to explicitly model coordination between robots. Basically the used decision-theoretic coordination approach follows the method proposed by [7] which reduces a utility value for frontier cells near other robots current targets. Robots need to share their current goal positions with others causing additional communication expenses. Based on these shared goals the utility for all cells in a circular neighborhood is reduced by a fixed value. The circular fixed value reduction is an approximative version of the original formulation and makes the approach simpler and less computational expensive. The fixed reduction in a circular area around shared goals models coordination in a local way, meaning that cells exceeding the radius are not affected. The selected radius is critical as a too

small radius might result in two robots still moving towards the same frontier while a too large radius has no effect in narrow situations. To achieve a more flexible coordination a second variant is tested that globally reduced the utility value according to the distance to shared goals, whereby the influence falls off with a quadratic decay. The distance of a frontier cell to all shared goals is calculated by starting a wavefront in each shared goal and afterwards accumulating the distances into a global utility value. The fusion of a cells utility value and traveling cost forms the basis for selection of frontier cells.

V. Experimental setup

While the very detailed simulation allows testing these strategies under nearly realistic conditions, it also brings in a great amount of additional influences that affect the performance result. Because all components on each of the virtual robots run in parallel, the exactly same test scenario can produce completely different results. The main reasons for this are the map building process and the robot navigation. Even a small change in the grid map generated from the pose graph can cause the selection of a different frontier cell at an early stage, thus completely changing the course of the exploration. Although this may seem to make evaluation results less clear, it is actually an advantage of this approach because all these problems are rather natural for real robot systems and should not be neglected. To compensate for this, every test scenario is executed 20 times in a row and results are averaged over all runs. All strategies are tested in three different environments to reduce the possible effect of a certain strategy being overly fitted for a single scenario. The first environment is designed to feature wide open spaces, irregular and regular borders, junctions, dead-ends and loops which are all common to outdoor scenarios. The second one is a hospital section that is commonly used in tests of mapping and exploration algorithms. Finally, the third one is a real map of an underground parking area that was generated using laser data from one of our robots and that was used to determine a best setup to use for our robots in this environment. With 3 environments, 5 strategies plus the single robot case as reference and 20 runs per experiment an overall of 360 runs was performed and evaluated.

All experiments are done with 5 robots that are located close to each other at a remote area of the environment. Placing the robots at a central spot in the map results in a much easier exploration task where almost all strategies perform equally. The spatial proximity in the beginning forces each strategy to solve the task of spreading the robots across the map efficiently while there are only limited directions. The relative pose of robots is fixed and known in order to simplify the initial localization step. During exploration robots look for new frontiers when they are closer than 3 meters to a set goal point and also do a repetitive rechecking [2] for new targets every 3 seconds.

VI. Experimental results

Evaluation is done using two different performance values. The first is the time until the given environment is completely explored, that is no further frontiers to unexplored terrain can be found. This value is easy to understand and always used in this context, as it is quite natural to expect that the exploration process finishes faster when the robots act in a cooperative way. A problem with this measurement is that it does not tell how the covered area develops over time. In a real scenario it might not be required to completely cover every spot in the map, but maybe locate a number of targets as fast as possible. To account for this, development of the explored area over time is also logged and visualized in a graph for all strategies. This allows determining a situation where some strategy steadily explores the whole area while some other may quickly cover most of the area and then spend a longer time on the last few percent.

The second performance value is the number of measurements (laser scans) that were integrated into the world model to create the map. Having the whole area covered with fewer measurements is highly desirable for a number of reasons. Indirectly this also measures the distance covered by all robots together, because new scans are added only when a robot moves or turns for a certain minimum distance or angle. The world model has to be stored on every system thus increasing the required memory space and making operations on the pose graph like the creation of the grid map more expensive. Finally, readings have to be shared among all robots causing additional communication expenses for every new laser scan added to the map.

Figure 3 shows the exploration area and the visualized results for the artificial environment. The map shows where the 5 robots started, the resulting minimum, maximum and average results for the 20 runs of each strategy are shown in the diagrams below. Both strategies that use the other robot's position in addition to the shared map data (MultiWave and MinPos) significantly outperform the uncoordinated nearest frontier approach. The MultiWave strategy with wait-option on the other hand is even slower than the uncoordinated approach. This is quite natural as robots are stopped temporary to avoid unnecessary movements. The advantage of this approach can be seen clearly in the right diagram that shows the number of measurements stored in the shared world model. The number of nodes in the resulting pose graph is more than halved and still significant better compared to all other strategies, so this strategy is best in terms of efficiency. (Remember that new nodes are added only after the robot moves forward or turns in place, so this also measures the summarized movement of all robots). Following the decision-theoretic approach with local or global utility evaluation the robots share their current targets instead of their current locations. While supposed to spread the robots better when close to each other due to its consideration of future movements, this strategy performs somewhat below the best strategy MinPos. These results indicate that it is better to share the robots current positions than the robot's current target locations.

The results for the "hospital" environment usually included with "Stage" are shown in Figure 4. In this scenario most strategies perform only slightly better than the plain nearest neighbor approach. A possible reason for this is that there are only few possibilities of how to reach most places within this map. Best results could be achieved with the MultiWave and the MinPos strategy, which both perform the nearest neighbor algorithm unless the robots are close to each other. The strategy which performs slower is the MultiWave+Wait

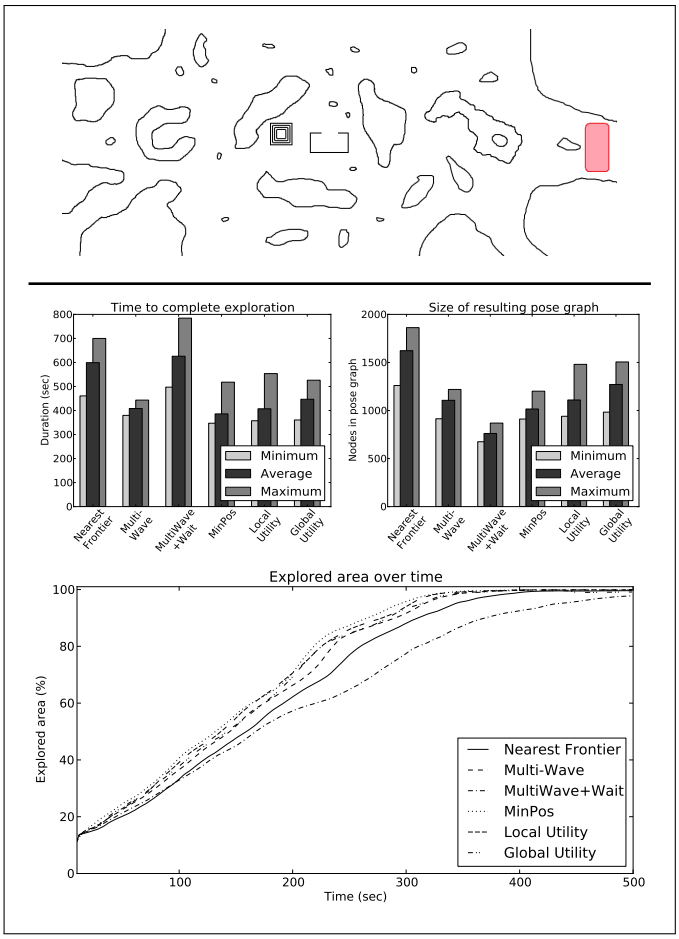

Fig. 3. **Results for the artificial environment:** On this map the "MultiWave+Wait" strategy performs only slightly worse than the uncoordinated approach regarding time, but significantly outperforms it regarding distance.

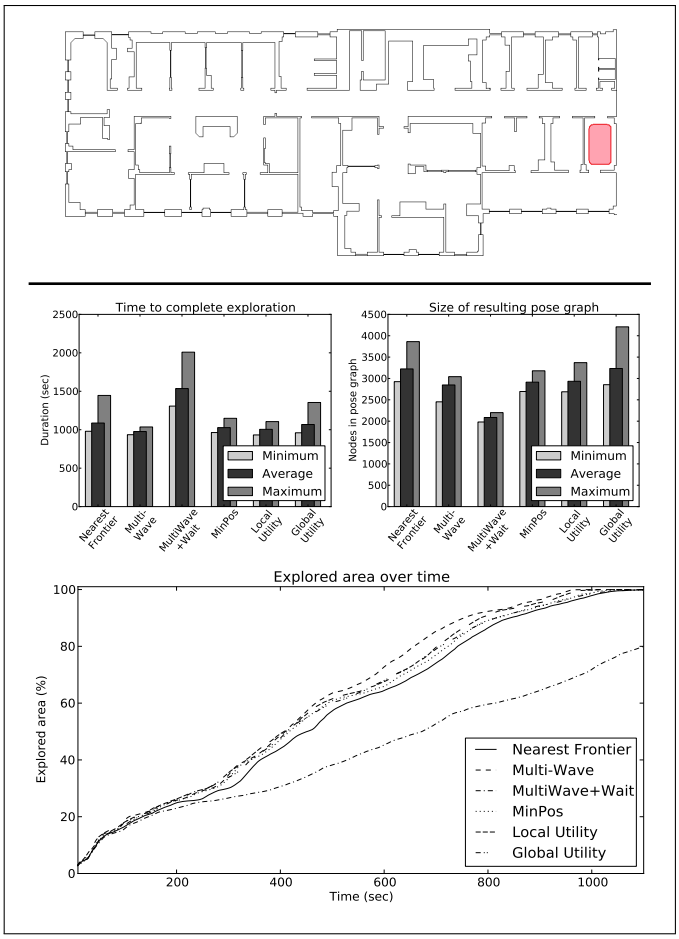

Fig. 4. **Results for "Hospital" environment:** This map is narrow with few connections between rooms, thus causing all strategies to result in similar move patterns. The different behavior of the "MultiWave+Wait" strategy is because there are only few passages in many situations causing the following robots to be put in the "wait" state. (map provided by [8])

variant, as it puts single robots to a paused state when there is another robot nearer to every visible border. Basically the same result can be observed with regards to the number of readings that were used to build the map, with the major difference that here the MultiWave+Wait variants ranks best. This is somewhat expected as the robots avoid any unnecessary movement towards frontiers better reachable by other robots at the price of increasing overall exploration time. The similarity in performance is even better seen in the area plot, where all strategies, except the much slower wait-algorithm, show a very similar development over time. Most of the time is lost right at the beginning, because the room where the robots start has only 2 exits. Thus 3 out of 5 robots are put to wait right after start and only restarted after the other two have started moving away and unveiled additional frontiers.

The third scenario is a simulation of the underground parking that is used for experiments with a team of real robots. The map was generated from real data, using the very same architecture and map generation that is used for evaluation. The area is relatively small and has few obstacles so that the 5 robots finish the exploration quite fast. Figure 5 shows the map and exploration results as before. The results show little differences in performance, with exception to the Multi-Wave+Wait strategy for the previously mentioned reasons. In terms of exploration time only MultiWave and Local-Utility rank above Nearest-Frontier. These results show the general problem one faces when evaluating exploration strategies: In a small area with too few obstacles, sophisticated algorithms can barely make a difference and hence do not justify their increased computational costs and additional requirements on communication between robots. The averaged results for all strategies in the three exploration scenarios are summarized in Table I.

	Exploration time (s)			**Size of pose graph**		
	Moon	Hospital	Parking	Moon	Hospital	Parking
Nearest Frontier	598	1086	333	1622	3222	899
MultiWave	408	**976**	**314**	1105	2847	842
MultiWave+Wait	625	1535	502	**761**	**2086**	**501**
MinPos	**385**	1027	363	1015	2913	956
Local Utility	407	1004	315	1109	2933	836
Global Utility	446	1067	348	1270	3233	954

TABLE I. OVERVIEW OF THE AVERAGED RESULTS FOR ALL STRATEGIES AND ENVIRONMENTS.

VII. CONCLUSIONS

In our experiments we evaluated 5 different exploration strategies using a full-system simulation against the straight-

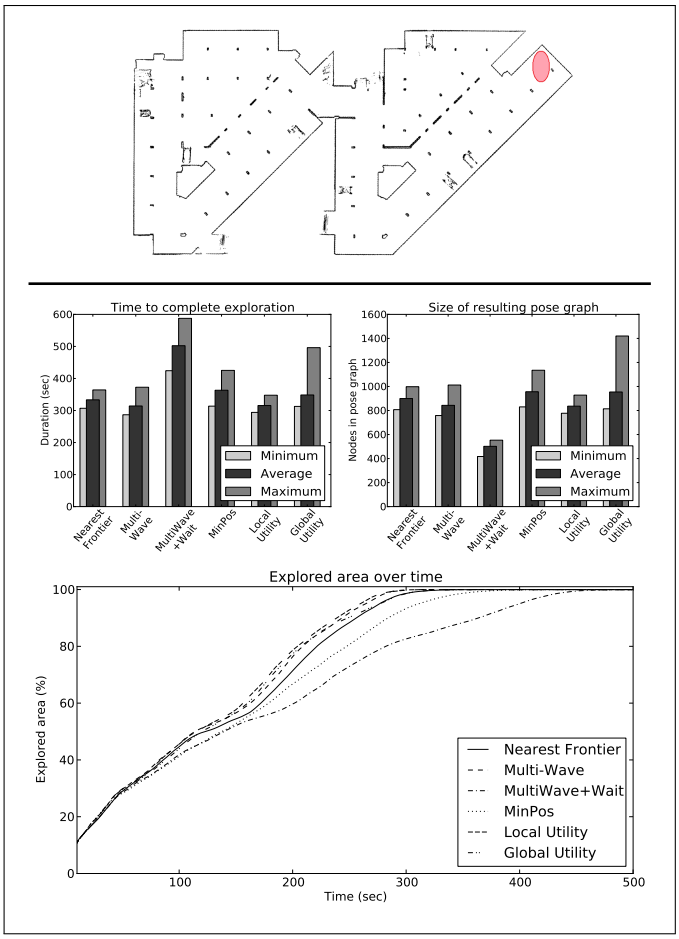

Fig. 5. **Results for the underground parking:** It was created from real sensor data and is structured like the hospital section but mostly open-spaced. The performance of the strategies starts to diverge at about half the exploration time depending on how fast a strategy manages to send robots through the narrow passageway into the left section of the parking area.

forward nearest frontier approach which has proven to be comparable. The results so far indicate that there is not a single best solution, but that the decision on a certain strategy should depend on the environment to be explored and the performance measurement that should be optimized. To be able to make a plausible statement on the performance of exploration strategies it is (1) necessary to use large environments with many obstacles, because in less complex scenarios even very simple strategies perform reasonably well. Because robots spend time on navigating, turning and avoiding each other it is (2) also required to include the robot's navigation so to evaluate the efficiency of the whole system. Also (3) the map-building process should be included as it can influence the exploration strategy and is influenced by the exploration strategy itself.

For the future we plan to repeat the evaluation with larger, especially more complex environments and implement other state-of-the-art strategies to compare them under realistic conditions. Newer Strategies could be evaluated which try to support the mapping process, for example by performing active loop-closing [13] that prevents mapping failures when too long cycles cannot be closed correctly anymore. Experiments with real robots have shown that communication loss plays a major role during exploration, thus indicating that the communication layer should also be included in the simulation to disconnect robots that are far away from each other. Finally, results taken from our work could be used to design specialized strategies, probably as a combination of existing ones, that optimize a given performance value given a set of system conditions and knowledge on the basic structure of the environment.

ACKNOWLEDGMENT

The work described in this paper has been done as a part of the project IMPERA, which is funded by the German Space Agency (DLR, Grant number: 50RA1111) with federal funds of the Federal Ministry of Economics and Technology (BMWi) in accordance with the parliamentary resolution of the German Parliament.

REFERENCES

[1] T. Balch and R. C. Arkin, "Communication in reactive multiagent robotic systems," *Autonomous Robots*, vol. 1, no. 1, pp. 27–52, 1994.

[2] D. Holz, N. Basilico, F. Amigoni, and S. Behnke, "Evaluating the Efficiency of Frontier-Based Exploration Strategies," in *Proceedings of the joint conference of the 41st International Symposium on Robotics (ISR 2010) and the 6th German Conference on Robotics (ROBOTIK 2010)*, Munich, Germany, June 2010, pp. 36–43.

[3] A. Bautin and O. Simonin, "MinPos : A Novel Frontier Allocation Algorithm for Multi-robot Exploration," *ICIRA*, pp. 496–508, 2012.

[4] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *In Proceedings of the IEEE International Symposium on Computational Intelligence, Robotics and Automation*, 1997, pp. 146–151.

[5] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. k. Younes, "Coordination for Multi-Robot Exploration and Mapping," *IEEE Transactions on Robotics*, pp. 852–858, 2000.

[6] K. M. Wurm, C. Stachniss, and W. Burgard, "Coordinated multi-robot exploration using a segmentation of the environment," *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1160–1165, Sep. 2008.

[7] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[8] R. Vaughan, "Massively multi-robot simulation in stage," *Swarm Intelligence*, vol. 2, no. 2-4, pp. 189–208, 2008.

[9] J. S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," *International Symposium on Computational Intelligence in Robotics and Automation*, pp. 318–325, 1999.

[10] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient Sparse Pose Adjustment for 2D Mapping," *International Conference on Intelligent Robots and Systems*, pp. 22–29, 2010.

[11] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[12] C. Stachniss, O. M. Mozos, and W. Burgard, "Speeding-up multi-robot exploration by considering semantic place information," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Orlando, FL, USA, 2006, pp. 1692–1697.

[13] C. Stachniss, D. Hähnel, and W. Burgard, "Exploration with active loop-closing for FastSLAM," *2004 IEEERSJ International Conference on Intelligent Robots and Systems IROS IEEE Cat No04CH37566*, vol. 2, pp. 1505–1510, 2004.