A Modified Multiple Depth First Search Algorithm for Grid Mapping Using Mini-Robots Khepera

Sally El-Ghoul[1], Ashraf S. Hussein and M. S. Abdel Wahab[2]
Faculty of Computer and Information Sciences, Ain Shams University
Abbassia, Cairo, 11566, Egypt
elghoul_girl@hotmail.com; ashrafh@acm.org; mswahab@gmail.com

U. Witkowski and U. Rückert
System and Circuit Technology, Heinz Nixdorf Institute, Paderborn University
Fuerstenallee 11, D33102, Paderborn, Germany
{witkowski,rueckert}@hni.upb.de

Received 17 July 2008; Accepted 4 December 2008

This paper presents a Modified Multiple Depth First Search algorithm for the exploration of the indoor environments occupied with obstacles in random distribution. The proposed algorithm was designed and implemented to employ one or a team of Khepera II mini robots for the exploration process. In case of multi-robots, the BlueCore2 External Bluetooth module was used to establish wireless networks with one master robot and one up to three slaves. Messages are sent and received via the module's Universal Asynchronous Receiver/Transmitter (UART) interface. Real exploration experiments were performed using locally developed teleworkbench with various autonomy features. In addition, computer simulation tool was also developed to simulate the exploration experiments with one master robot and one up to ten slaves. Computer simulations were in good agreement with the real experiments for the considered cases of one to one up to three networks. Results of the MMDFS for single robot exhibited 46% reduction in the needed number of steps for exploring environments with obstacles in comparison with other algorithms, namely the Ants algorithm and the original MDFS algorithm. This reduction reaches 71% whenever exploring open areas. Finally, results performed using multi-robots exhibited more reduction in the needed number of exploration steps.

[1]Scholarship from System and Circuit Technology, Heinz Nixdorf Institute, Paderborn University, February 2008.
[2]Visiting professor, System and Circuit Technology, Heinz Nixdorf Institute, Paderborn University, February 2008.

Categories and Subject Description: Artificial Intelligence [**Robotics, Distributed Artificial Intelligence**]

General Terms: Algorithms, Experimentation

Additiona Key words: Exploration, Indoor Mapping, autonomous robot and robot teams, mobile agents

1. INTRODUCTION

The mapping algorithms address the problem of acquiring spatial models of physical environments using mobile robots. We will focus on using the indoor exploration algorithms to construct a map for such environment. Therefore, the problem under consideration lies in the overlap between the mapping problem and the motion control of the robot as shown in Figure 1. Robot algorithms [Koenig and Liu 2001; Thrun 2003; Kuipers and Byun 1991; Thrun 1998] establish the connection between the data collected by sensors and the robot s activities. Mapping algorithms are considered the main component within robot algorithms for building truly autonomous mobile robots. This research work aims at the enhancements in the mapping algorithms considering single and multi-robot systems.

The use of multiple robots system is often suggested to have several advantages over single robot [Cao et al. 1997; Dudek et al. 1996]. The first advantage is the co-operating robots have the potential to accomplish a single task faster than a single robot. For example, Guzzoni et al. [Guzzoni 1997] built a system of collaborative robots that jointly schedule a meeting, which outperformed several single robot systems designed to accomplish the same task. Furthermore, multiple robots can localize themselves more efficiently if they exchange information about their position whenever they sense each other [Fox et al. 1999]. Finally, using several cheap robots introduces redundancy and therefore can be expected to be more fault-tolerant than having only one powerful and expensive robot. The problem to be solved when using multi-robot systems is to coordinate the actions of the robots. Without any coordination, all robots might follow the same exploration path so that the whole group of robots requires the same amount of time as a single robot would need. Therefore, the key problem in multi-robot exploration is to choose different actions for the individual robots so that they simultaneously explore different areas of their environment.

The rest of the paper is organized as follows. Section 2 provides an overview of the relevant research work. Section 3 discusses the Modified Multiple Depth First Search algorithm using a single robot. Section 4 presents the extended MMDFS algorithm using multi-robot system. Section 5 discusses the communication protocol used between the master and slaves to implement the extended MMDFS. Section 6 presents the obtained computer simulation results and illustrates the experimental results. Finally, the conclusions are provided in Section 7.

2. RELATED WORK

Mapping algorithms are a goal, which has been pursued by a diversity of research groups with various objectives and approaches in the last two decades. Generally, all

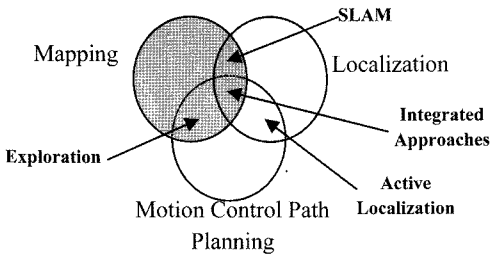


Figure 1. The tasks that the robot should accomplish in order to acquire accurate models of the environment.

of the state of the art mapping algorithms are probabilistic [Smith et al. 1990; Castellanos and Tardós 2000] because of the uncertainty and the noise in the measurements. Some algorithms are incremental that can be run in real time, whereas others require multiple passes through the data. Furthermore, some algorithms require exact pose information to construct a map, whereas others can do so using odometry measurements. On the other hand, some algorithms are capable of handling correspondence problems between data recorded at different points in time [Thrun 2003]. An exhaustive literature survey for robot exploration in the context of mapping has been given in [Thrun 2003].

The exploration algorithms are categorized in two main categories; off-line and on-line ones [Choset 2001]. For the off-line algorithms, the robot is provided previously with the map of the assigned area to be explored. On the other hand, for the on-line algorithms, there are no assumptions on the environment and is based only on sensors' measurements (i.e. based on the local knowledge) such as the Ants algorithm proposed in [Makarenko et al. 2002; Koenig and Liu 2001], the Multiple Depth First Search algorithm and the Brick and Mortar algorithm [Ferranti et al. 2007]. The Ants algorithm [Ferranti et al. 2007] divides the area into regular square grid cells, and the robot explores the environment marking the traced cells during its movements. This algorithm is not efficient in terms of the number of needed steps to explore the whole environment as it has no stopping criterion, and the robot will keep working until its battery becomes empty or the user terminates the algorithm. Batalin and Sukhatme [Batalin and Sukhatme 2005] presented similar algorithm that works in a way similar to the Ants algorithm. Their algorithm uses a sensor network infrastructure to guide the robot to the least visited cells.

Some other exploration algorithms need partial information about the environment as the Frontier-based exploration algorithms that discussed in [Yamauchi 1998]. In this algorithm, which works with a team of robots, the robots divide the environment to be explored into cells. Each robot keeps in its memory a map for the whole environment to guide itself to the boundary between the open spaces in order to gather more information about the environment. The Depth First Search algorithm is used to move from the current position to the next frontier. In the local navigation algorithm, proposed in [Svennebring and Koenig 2004], the robot is provided with some information about its environment to decide where to go next.

Although the exploration problem has been extensively studied for a single robot, there are only few approaches that consider the multi-robot systems. Rekleitis et al. [Rekleitis et al. 1997; Rekleitis et al. 1998] considered the collaborative exploration by multiple robots in the context of reducing the odometry error during the exploration. They divide the environment into stripes that are explored successively by a robot team. Whenever one robot moves, the other robots are kept stationary and observe the moving robot in a way similar to [Kurazume and Shigemi 1994]. Whereas this approach can significantly reduce the odometry error during the exploration process, it is not designed to distribute the robots over the environment. Rather, the robots are forced to stay close to each other in order to remain in the visibility range. Thus, using these strategies for multi-robot exploration will not contribute to the reduction of the exploration time or steps. More sophisticated techniques for multi-robot exploration have been presented in [Singh and Fujimura 1993; Yamauchi 1998]. In these techniques, the robots share a common map that is constructed during the exploration. Singh and Fujimura [Singh and Fujimura 1993] presented a decentralized online approach for heterogenous robots. Whenever a robot 'A' discovers an opening to an unexplored area that can not be reached because of the robot size, the robot 'A' selects another robot 'B' to accomplish this exploration task. The candidate robot is selected by trading off the number of areas to be explored, the size of the robot and the straight-line distance between the robot and the target region. In the approach of Yamauchi [Yamauchi 1998], robots move to the closest frontier, which is the closest unknown area around the robot according to the current map.

Research in the robot mapping problem has provided two major paradigms for mapping indoor environments; grid-based and topological [Thrun 1998]. Elfes and Moravec's [Elfes 1989; Moravec 1988] produced grid-based maps, which represented by fine-grained grids to model the occupied and free spaces of the environment. Examples of topological approaches include the work of Kuipers [Kuipers and Byun 1991] and others [Choset and Burdick 1996; Kortenkamp and Weymouth 1994; Shatkay and Kaelbling 1997]. A brief comparison between grid-based and topological approaches for map building is illustrated in Table I.

In this work, the on-line exploration approach is adopted using a team of robots in

Table I. Comparison between grid-based and topological approaches for constructing maps.

	Grid Based (Metric) approaches	Topological approaches
Advantages	• Easy to build, represent and maintain • Facilitates computation of shortest path	• low space complexity (resolution depends on the complexity of the environment) • Does not require accurate determination of the robot s position
Disadvantages	• Space-consuming (resolution does not depend on the complexity of the environment) • Requires accurate determination of the robot s position	• Difficult to construct and maintain in larger environments • Recognition of places (based on landmarks) often ambiguous

grid-based maps "Metric". These maps are discrete, two-dimensional occupancy grids (in which each cell has a value measuring the subjective belief that this cell is occupied i.e. it represents the configuration space of the robot).

3. THE MODIFIED MULTIPLE DEPTH FIRST SEARCH (MMDFS) ALGORITHM FOR SINGLE ROBOT

In the MMDFS algorithm, the whole environment, Figure 2(a), will be divided into small quadratic patches "cells" as shown in Figure 2(b). The robot should be placed on one of the boundary cells as the starting position, Figure 2(c). In each step, the robot can move from the current cell to another one of the four adjacent cells in the North, East, South and West directions as shown in Figure 2(d).

The cell can be in one of the following states:

- **Wall:** The cell can not be traversed by the robot because it is blocked by an obstacle.
- **Unexplored:** No robot visited the cell yet, and whether it is free or occupied by an obstacle is totally unknown.
- **Explored:** The cell has been traversed at least once by the robot, but it might need to go through it again in order to reach unexplored regions. It also means that the cell is free.
- **Visited:** The robot has already passed through the cell and will not need to go through it again. In other words, it is treated as a wall cell, but it is known that it is a free cell.
- **Checked:** The robot has detected that the cell is free (i.e. it is not occupied with an obstacle), but it didn't traverse it.

The algorithm checks the four neighbor cells at each step and assigns the appropriate state to each one, which will minimize the time required for constructing the map. The characteristics of the mini robot Khepera II facilitate the achievement of this idea as the robot has eight infrared (IR) sensors as shown in Figure 3, but actually,

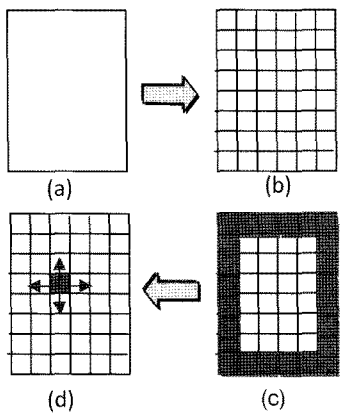


Figure 2. (a) The environment to be explored. (b) the environment is divided into square patches. (c) The robot should be placed to one of the boundary cells as the starting position. (d) the robot is allowed to move in one of the four adjacent cells.

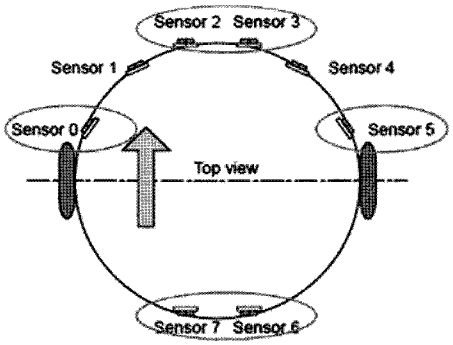


Figure 3. Top view of the Khepera II robot.

only six of them are used in our algorithm.

The robot will terminate the mapping algorithm when it finds that all the cells' states are known. In other words, there are no cells with unexplored states. The modified MDFS algorithm is explained in details in the following pseudo code.

Modified MDFS

IF any cell state is still unexplored ***then***
  Get The 4 neighbors cell states
  IF there are unexplored cells around ***then***
    The robot will move to one of them randomly
    IF the current cell is not visited ***then***
      Set The current cell state to explored
    END IF
  ELSE IF there are checked cells around ***then***
    The robot will move to one of them randomly
    IF the current cell is not visited ***then***
      Set The current cell state to explored
    END IF
  ELSE
    The robot should move to the parent cell
    Set The current cell state to visited
  END IF

  Update the state of the four neighbor cells to wall or checked
  Apply the appropriate movement to the robot
ELSE
  Terminate the algorithm
END IF

4. THE MODIFIED MULTIPLE DEPTH FIRST SEARCH (MMDFS) ALGORITHM FOR A TEAM OF ROBOTS

Development of the MMDFS algorithm for a team of robots aims at the accomplishment of a single task faster than that using a single robot. In our case, this will improve the time needed for exploring the environment by minimizing the number of steps needed to construct its map, and will minimize the Odometry error as well. The challenge in using more than one robot is to ensure that the robots can efficiently collaborate to explore the area. In the extended MMDFS algorithm, each robot constructs its own exploration tree, and tries not to interfere with the trees of the other agents by marking cells with its own Robot-ID.

The centralized technique is used for multi-robots in which we have a master robot and N slaves. The master robot's role is to guide the slaves within the environment as it has the global map of the whole environment. The role of the slaves is to check their four neighbors at each step, send their readings to the master robot and wait for the instructions to know where they should go next. In addition, each slave decides if it finished the exploration or not. The slaves are not capable to communicate directly with each others; they are only able to send/receive messages to/from the master robot. Hence, the master plays the role of the coordinator of messages between robots. The extended MMDFS algorithm is explained in details in the following pseudo code.

Master MMDFS

```
Initialize the communication between the master robot and the slaves

Wait for messages from the slaves

IF the message is ARUN (Asking to run) then
  IF there is any unexplored cell then
    Send to the slave to CRUN (Continue running the algorithm)
  ELSE
    Disconnect the slave
  END IF
ELSE IF the message if SRUN (Stop running) then
  Disconnect all the slaves
ELSE IF the message is the value of readings of the 4 neighbors' of the slave then
  IF there is any available move for the robot then
      Send to the slave the direction of the next movement or the back command to
      move for its parent cell
  Else
    Terminate the algorithm and disconnect all the slaves
  END IF
END IF
```

Slave MMDFS

After the communication is established with the master robot, the slave waits for a message from the master robot

IF the message is CRUN (Continue running the algorithm) ***then***
  IF there is no available movements around the robot ***then***
    Send to the robot SRUN (ask the robot to end the connection with this slave)
  ELSE
    Get the sensor values for the four neighbors of the robot and send these readings to the master robot
  END IF
Else IF the message is the next move to the robot ***then***
  Move the robot (slave) to the required position and send back to the master ARUN (ask to run)
ELSE IF the message is BACK
    Move the robot to its parent cell and send back to the master ARUN (ask to run)
END IF

5. COMMUNICATION PROTOCOLS

In this work, we used the BlueCore2-External Bluetooth module [Grosseschallau 2004] developed in Heinz Nixdorf Institute. This module allows the establishment of wireless networks with one master and three slaves (1-to-3 networks). Any interaction between the user and the Bluetooth module is performed by the American Standard Code for Information Interchange (ASCII). Messages are sent and received via the module's Universal Asynchronous Receiver/Transmitter (UART) interface. Every message is enclosed by <....> and starts with a numeric message ID. The typical message flow to establish a connection, transfer data and disconnect is shown in the below Figure 4.

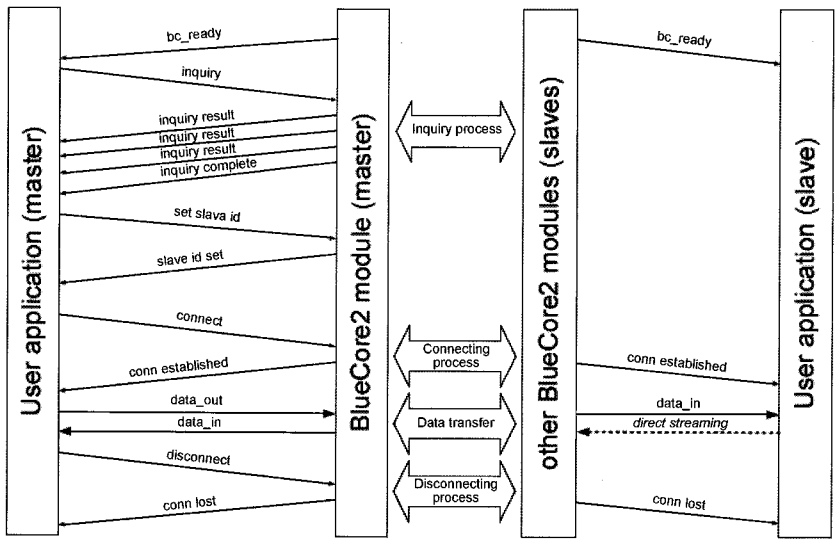


Figure 4. Typical message flow between one master and one slave application.

As shown in Figure 4, the user application always has to wait for the event **bc_ready** from the **BlueCore2** module. Then, commands for performing the required task can be executed. The master robot starts with a device inquiry by calling the inquiry command. As a response, all the found devices answer (slaves), each with a single **inquiry_result** event. After receiving the **inquiry_complete** event, the inquiry procedure has ended and the master can issue the next command. From all the returned addresses, the master can then pick 3 devices that it wants to connect to. The command **set_slave_id** advises the master **BlueCore2** to associate a given address with a given ID, which can be 0, 1 or 2 for allowing shorter commands by using a short slave ID instead of the long Bluetooth device address. After all the selected peer devices are associated to slave IDs, the master can call the connect command to establish a connection to one of the slaves. It is important to wait for the response of this command (either **conn_established** or **conn_refused**) before trying to connect to the next device. If the connection is successfully established, data transfer can be commenced. The master's user application uses the **data_out** command while the slave application just writes directly to the UART. On both sides, incoming data is presented in **data_in** messages. After all the data is transmitted, the connection can be terminated from the master's side by using the **disconnect** command. Both sides receive a **conn_lost** event if the termination occurs.

This implementation is based on one main principal that is the master device always initiates any action while the slave devices just respond to the special requests. Although the message handling seems to be straight forward, it imposes some limitations on the usage possibilities. These limitations are originated from that the switching between the master and the slaves (using the **change_role** command) is possible but needs a firmware initiated reboot of the Bluetooth module afterwards. Therefore, the established connection cannot be kept while switching the role. The challenge in using more than one robot is to ensure that they can efficiently collaborate to explore the area. That is why we needed to implement a message protocol for the communication between robots (master and slaves). We used the commands **data_in** and **data_out** to exchange the messages between master and slaves. The implemented messages are as shown in Table II.

6. RESULTS AND DISCUSSIONS

6.1 Simulation Results

Computer simulation tool was developed with the user interface shown in Figure 5 to test the performance of the extended MMDFS algorithm. The developed tool allows users to automatically construct the environment map with different number of obstacles. In addition, the simulation tool is developed to test the algorithm using single or multi-robots. In these simulations, we assumed that there is no communication overhead, which is not the case in real experiments. Although the BlueCore2 module supports only 1 to 3 comminucation network in real experiments, we tested the algorithm using 1 to 1 up to 10 robots through the simulation tool to study the influence of using up to 10 robots on the exploration process.

In order to assess the performance of the modified MDFS algorithm against the original one, a single robot is used to perform this assessment on variable number of cells representing the open area to be explored. For each area size, the number of steps needed to explore the environment and construct its map is recorded as shown in Figure 6. It is interesting to point out that the reduction in the number of steps

Table II. Message protocol between master and slaves.

	Message	Description	Variables	Master	Slave
1	[ARUN]	To ask if the slave can continue running the MMDFS or not.	____	√	√
2	[SRPOS: XX, YY]	Setting the robot position at a cell (XX, YY).	**XX:** the robot's position in the x-axis **YY:** the robot's position in the y-axis	√	
3	[SRDIR: D]	Setting the robot direction to the direction (Up, Left, Right, Down).	**D:** The robot's direction (Up, Left, Right, Down)	√	
4	[CRUN]	It means that the slave can continue running the algorithm.	____	√	
5	[NVAL: FY, BN, RY, LN]	The slave sends it to the master indicating the result of the sensor reading. It shows if the forward is free or not and so on.	**FY/FN:** It indicates that the forward cell is empty or not **BY/BN:** It indicates if the backward cell is empty or not. **RY/RN:** It indicates if the right cell is empty or not **LY/LN:** It indicated if the left cell is empty or not		√
6	[MOVE: D]	The master will guide the slave to move in a certain (given) direction.	**D:** The robot's direction (Up, Left, Right, Down)	√	
7	[BACK]	The master commands the slave to backtrack and move to its parent cell.	____	√	
8	[URPOS: XX, YY]	The slave updates its position at the master variables.	**XX:** the robot's position in the x-axis **YY:** the robot's position in the y-axis		√
9	[URDIR: D]	The slave updates its direction in the master's variables.	**D:** The robot's direction (Up, Left, Right, Down)		√
10	[SRUN]	The slave has completed its task and asks the master to stop running its code.	____		√

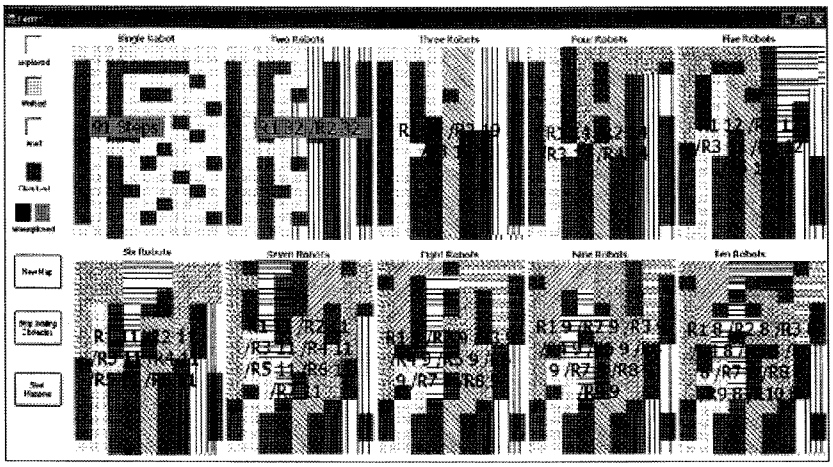


Figure 5. The simulation tool user interface.

using the modified MDFS increases with the increase of the number of cells representing the environment.

The influence of the number of obstacles on the number of needed steps for both the original MDFS and the modified one to construct an environment map is depicted in Figure 7. Each record in the graph represents the average of running each algorithm 30 times using the same number of obstacles but at different locations each time. A fixed map size of 126 (9×14) cells was used to represent the environment under consideration. Generally, using the modified MDFS algorithm led to achieving reduction, in the number of needed steps to build the map, between 20% (for 30 objects) and 70% (for 3 objects).

The influence of the number of the in-service robots on the required number of steps needed to construct the environment map is studied by constructing the map

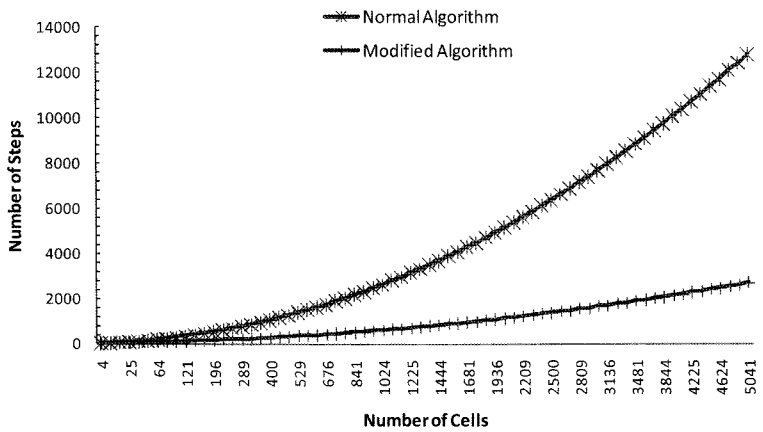


Figure 6. The impact of the number of cells representing the environment, to be explored, on the number of the needed exploration steps for each algorithm. The blue curve is for the original MDFS algorithm while the red one is for the Modified MDFS.

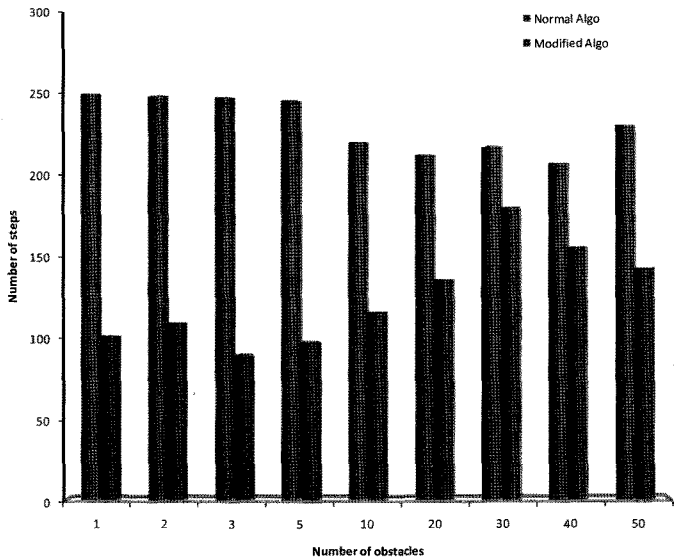


Figure 7. The impact of the number of obstacles on the number of needed steps for each algorithm.

of an open area using a map size of 9×14 cells. The MMDFS algorithm was employed varying the number of the in-service robots starting from 1 to 10 using the same map size. For each case, the needed number of steps to construct the map is computed as shown in Figure 8. Generally, using the extended MMDFS algorithm led to achieving reduction, in the number of steps needed to construct the environment map. This achievement reaches 64% (using 2 robots) and 91% (using 10 robots).

In real cases, obstacles are randomly distributed within the considered environments. Therefore, to study the influence of the number of the in-service robots on the

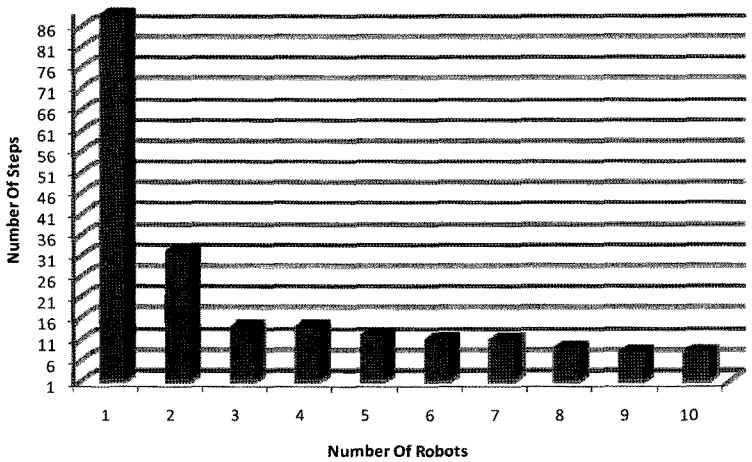


Figure 8. The number of in-service robots impact on the number of steps needed to explore an open area.

required number of steps needed to construct the real environment map, the MMDFS algorithm was employed varying the number of the in-service robots starting from 1 to 10 using the same map size. For each fixed number of in-service robots, 1, 2, 3, 5, 10, 20, 30, 40 and 50 obstacles were randomly distributed in the environment generating 90 unique test cases. For each test case, to measure a meaningful number of steps required to construct the environment map, the number of steps was computed as the average of 20 successive runs of MMDFS algorithm fixing the number of the in-service robots and the number of the distributed obstacles. As shown in Figure 9, for any fixed number of in-service robots, the number of the required exploration steps increases as the obstacles reaches 15% of the total area then decreases as the obstacles exceed this range. Using the extended MMDFS led to achieving reduction, in the number of steps needed to build the map, between 75% and 85%.

6.2 Experimental Results

To validate the simulation results of the MMDFS algorithm, we used the Khepera II [2007] mini-robot(s) and the Teleworkbench [El-Ghoul et al. 2007] as a test bed. The

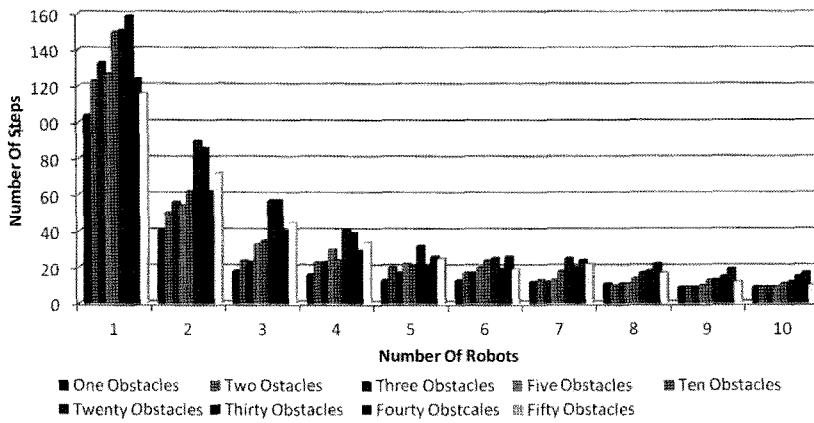


Figure 9. The impact of the number of obstacles and Number of robots on the number of steps needed by each algorithm.

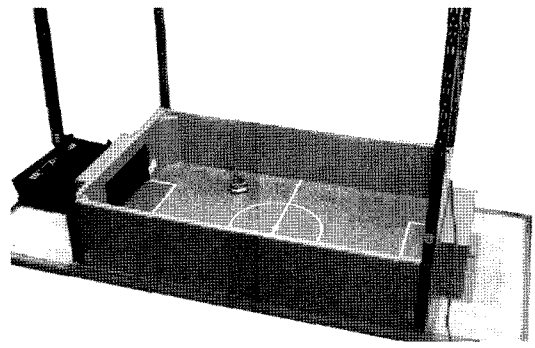


Figure 10. Teleworkbench.

teleworkbench platform, Figure 10, was locally developed by a team at the Faculty of Computer and Information Sciences, Ain Shams University to perform experiments using the Khepera series mini-robots. Users can upload the code of the experiment under consideration and view the result via live video streaming [El-Ghoul et al. 2007]. Long term autonomy of the platform is possible, as the platform is equipped with a docking station for recharging the robot's batteries. Any robot within the experiment can detect low battery level and proceeds automatically to the docking station. Moreover, this teleworkbench is web-based system. This allows outsiders to access the system and perform experiments remotely [El-Ghoul et al. 2007].

To evaluate the simulation results, we performed real experiments on an environment represented by 35 cells (each cell is 7×7 cm) with 8 obstacles. We adjusted the robot's speed to 5 cm/second. First, we started to compare the results of the original MDFS and the modified version using only one robot. For each run, we uploaded the corresponding algorithm's code on the robot via the web-based interface of the teleworkbench. The result of this experiment is shown in Figures 11(a) and 11(b). As shown, the original MDFS needed 67 steps to explore the environment and build its map while the MMDFS required only 16 steps. This means that the MMDFS reduced the needed number of steps by 76% and this in a very good agreement with

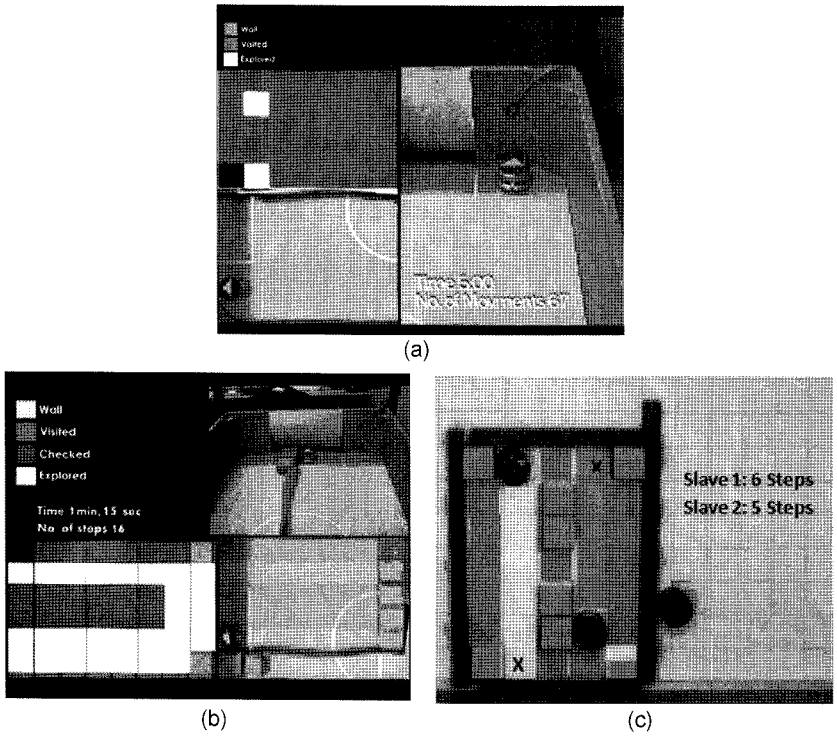


Figure 11. Snapshot of the real experiments results. (a) shows the result of the normal MDFS algorithm. (b) shows the result of the MMDFS algorithm using single robot (b) shows the result of the extended MMDFS using one master and two slaves.

the simulation results.

Next, in order to validate the simulation results of the extended MMDFS algorithm, we performed a real exploration experiment using one master robot and two slaves. For running the experiment, we uploaded to the robots the codes of the master and the slaves via the teleworkbench web-based interface. The result of this experiment is shown in Figure 11(c). As shown the MMDFS algorithm required 16 steps to explore the environment and build its map while the extended MMDFS required only 6 steps; 1 step from slave I and 5 steps from slave II. This means that the extended MMDFS reduced the number of the needed exploration steps by 63%.

7. CONCLUSION

In this work, we proposed a modified Multiple Depth First Search (MDFS) algorithm using single and multi-robots for the environment mapping. The modified algorithm was implemented and validated using both computer simulation and real experiments employing the Khepera II mini-robot running on the locally developed teleworkbench. Experiments performed using the MMDFS for single robot exhibited significant reduction in the number of the needed steps for the exploration of environments with obstacles by 46% in comparison with other algorithms, namely the Ants algorithm and the original MDFS algorithm. This reduction reaches 71% whenever exploring "open areas". Finally, results performed using the MMDFS for multi-robots exhibited more reduction in the needed number of exploration steps. It is interesting to point out that if the used robots have richer sensors capabilities, we could also check the cells on the diagonals

As a future work, we are planning to enhance the proposed extended MMDFS algorithm for further series of the Khepera mini-robots.

ACKNOWLEDGEMENTS

The authors of the Ain Shams University would like to appreciate the cooperation between the Department of System and Circuit Technology, Heinz Nixdorf Institute, University of Paderborn, Germany and the Department of Scientific Computing, Faculty of Computer and Information Sciences, Ain Shams University, Egypt. Special thanks are extended to Prof. U. Rückert for his great support.

REFERENCES

BATALIN, M. A. AND G. S. SUKHATME. 2005. The analysis of an efficient algorithm for robot coverage and exploration based on sensor network deployment, In *ICRA05: Proceedings of the IEEE International Conference on Robotics and Automation*: 3478–3485.

CAO, Y. U., A. S. FUKUNAGA, AND A. B. KHANG. 1997. Cooperative mobile robotics: Antecedents and directions. Autonomous Robots, 4.

CASTELLANOS, J. A. AND J. D. TARDÓS. 2000. Mobile robot localization and map building: A multisensor fusion approach, Kluwer Academic Publishers, Boston, MA. [ISBN: 0-7923-7789-3]

CHOSET, H. 2001. Coverage for robotics - a survey of recent results, *Journal of Annals of Mathematics and Artificial Intelligence*, 31(1–4):113–126. [doi: 10.1023/A:1016639210559]

CHOSET, H. AND J. W. BURDICK. 1996. Sensor based planning: The hierarhical generalized Voronoi graph, In *Proceedings of Workshop on Algorithmic Foundations of Robotics*,

Toulouse, France: 169–182.

DUDEK, G., M. JENKIN, E. E. MILIOS, AND D. WILKES. 1996. A taxonomy for multi-agent robotics. Autonomous Robots, 3(4).

ELFES, A. 1989. Occupancy grids: A probabilistic framework for robot perception and navigation, Ph.D. thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University.

EL-GHOUL, S., A. FOUAD, A. S. HUSSEIN, A. KAMAL, M. ABDEL-MENIEM, W. ABO EL KHAIR, A. GAMAL, M. S. EL-GHONIEMY, M. S. ABDEL WAHAB, U. WITKOWSKI, E. MONIER AND U. R CKERT. 2007. Construction of automated teleworkbench for mini robots, *International Journal of Intelligent Computing and Information Sciences*, 7(2):117–124.

FERRANTI, E., N. TRIGONI, AND M. LEVENE. 2007. Brick&Mortar: an on-line multi-agent exploration algorithm, In Proceedings of 2007 IEEE International Conference on Robotics and Automation, 761–767. [doi: 10.1109/ROBOT.2007.363078]

FOX, D., W. BURGARD, H. KRUPPA, AND S. THRUN. 1999. Collaborative multi-robot localization. In *Proc. of the 23rd German Conference on Artificial Intelligence*. Springer Verlag.

GROSSESCHALLAU, M. 2004. Implementation of an ASCII interface to the BlueCore2-External Bluetooth module for the minirobot Khepera Ph.D thesis, Department System and Circuit Technology, Heinz Nixdorf Institute, University of Paderborn.

GUZZONI, D., A. CHEYER, L. JULIA, AND K. KONOLIGE. 1997. Many robots make short work. AI Magazine, 18(1):55–64.

KOENIG, S. AND Y. LIU. 2001. Terrain coverage with ant robots: a simulation study, in AGENTS01: In *Proceedings of the 5th International Conference on Autonomous Agents*. ACM Press: 600–607.

KORTENKAMP, D. AND T. WEYMOUTH. 1994. Topological mapping for mobile robots using a combination of sonar and vision sensing, In *Proceedings of the 12th National Conference on Artificial Intelligence*, pp. 979–984, Menlo Park, MIT Press.

KUIPERS, B. AND Y. BYUN. 1991. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations, Toward Learning Robots, MIT Press, Cambridge, MA: 47–63. [ISBN: 0-262-72017-5]

KURAZUME, R. AND N. SHIGEMI. 1994. Cooperative positioning with multiple robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

MAKARENKO, A. A., S. B. WILLIAMS, F. BOURGOULT, AND F. DURRANT-WHYTE. 2002. An experiment in integrated exploration, In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1:534–539. [doi: 10.1109/IRDS. 2002.1041445]

MORAVEC, H. P. 1988. Sensor fusion in certainty grids for mobile robots, AI Magazine, 9(2):61–74.

REKLEITIS, I. M., G. DUDEK, AND E. E. MILIOS. 1997. Multi-robot exploration of an unknown environment, efficiently reducing the odometry error. In *Proc. of International Joint Conference in Artificial Intelligence (IJCAI)*.

REKLEITIS, I. M., G. DUDEK, AND E. E. MILIOS. 1998. Accurate mapping of an unknown world and online landmark positioning. In *Proc. of Vision Interface (VI)*.

SHATKAY, H. AND L. KAELBLING. 1997. Learning topological maps with weak local odometric information, In *Proceedings of International Joint Conference on Artificial Intelligence*, IJCAI-97, 920–927.

SINGH, K. AND K. FUJIMURA. 1993. Map making by cooperating mobile robots. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*.

SMITH, R., M. SELF, AND P. CHEESEMAN, 1990. Estimating uncertain spatial relationships in robotics, Autonomous Robot Vehnicles, Springer-Verlag New York, Inc.:167–193. [ISBN: 0-387-97240-4]

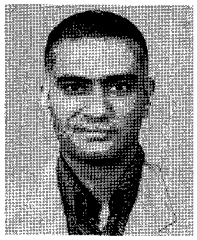
SVENNEBRING, J. AND S. KOENIG. 2004. Building terrain-covering ant robots: A feasibility study, *Journal of Autonomous Robots*, 16(3):313–332. [doi:10.1023/B:AURO.0000025793. 46961.f6]

THRUN, S. 2003. Robotic mapping: a survey, Exploring Artificial Intelligence in the New Millennium, Morgan Kaufmann Publishers Inc., San Francisco, CA:1–35.

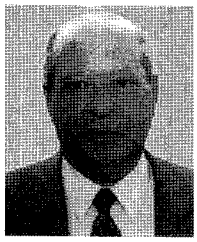
THRUN, S. 1998. Learning metric-topological maps for indoor mobile robot navigation, *Artificial Intelligence*, 99(1):21–71.

YAMAUCHI, B. 1998. Frontier-based exploration using multiple robots, In *Agents98: Proceedings of the 2nd International Conference on Autonomous Agents*: 47–53.

Sally El-Ghoul received her B.Sc. degree in Computer Science from Ain Shams University, Egypt in 2005. Currently, she is as a teaching assistant in the Department of Scientific Computing, Faculty of Computer and Information Sciences, Ain Shams University. Her research area covers Robotics, Artificial Intelligence and Modeling and Simulation. She has three published scientific papers in international conferences and journals. Sally is a member of the secretary board of one technical journal.

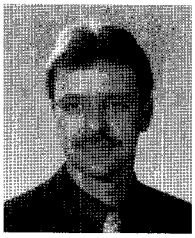


Ashraf S. Hussein was born in Giza, Egypt, in 1970. He received his B.Sc., M.Sc. and Ph.D. degrees in Aerospace Engineering from the Cairo University, Egypt in 1992, 1996 and 1999, respectively. The major fields of his studies are Modeling and Simulation, and Computer Visualization. Currently, he is an Associate Professor in the Department of Scientific Computing, Faculty of Computer and Information Sciences, Ain Shams University. He is also a Visiting Scientist, IBM Center for Advanced Studies in Cairo. His research areas cover Modeling and Simulation, Computational Techniques in Science and Engineering, Computer Graphics and Visualization, and High Performance Computing Applications. He has more than 40 published scientific papers in international conferences and journals. He has participated in more than 15 R&D and incubation projects. He is a member of the editorial board of one technical journal and a professional member of the Association of Computing Machinery (ACM).



M. S. Abdel Wahab received his B.Sc., M.Sc. degrees from the Ain Shams University, Egypt in 1967, 1971 respectively. Then, he received his Ph.D. degree from Karlsruhe University, Germany in 1978. The major fields of his studies are Modeling and Simulation and Artificial Intelligence. Since 1995 he has been a Professor of Scientific Computing at the Faculty of Computer and Information Sciences, Ain Shams University. He is the head of the research group Robotics. His other research interests are Distributed Intelligent Systems, Robotics and Modeling and Simulation. He is the x-dean of the Faculty of Computer and Information Sciences, Ain Shams University and one of the chief editors of the International Journal of Intelligent Computing and Information Sciences.

U. Witkowski received the diploma degree in electrical engineering in 1995 from the Technical University of Hamburg-Harburg, Germany. In 2003, he received the Dr.-Ing. degree from the University of Paderborn, Germany. Currently, he is working as a Senior Researcher in the research group System and Circuit Technology at the Heinz Nixdorf Institute, University of Paderborn. His research interests are among efficient microelectronic realizations of neural networks, mini-robotics and cooperative behavior strategies.



U. Rückert received the diploma degree in computer science and a Dr.-Ing. degree in electrical engineering from the University of Dortmund in 1984 and 1989, respectively. Since 1995 he has been a Professor of electrical engineering at the University of Paderborn. As a member of the Heinz Nixdorf Institute he is head of the research group System and Circuit Technology. The group is working on innovative circuit design and development of microelectronic systems for massive-parallel and resource-efficient information processing. The main research interests are distributed intelligent systems, neural information processing and microelectronic system integration. He is one of the three organizers of the first International Conference on Microelectronics for Neural Networks which is the only international conference devoted exclusively to hardware implementations of neural networks. He is member of the managing board of the Heinz Nixdorf Institute, and Adjunct Professor at the Fac ulty of Information Technology, Queensland University of Technology, Brisbane, Australia.