FRONTIER BASED MULTI ROBOT AREA EXPLORATION USING PRIORITIZED ROUTING

Rahul Sharma K.
Daniel Honc
František Dušek
Department of Process control
Faculty of Electrical Engineering and Informatics,
University of Pardubice, Czech Republic
E-mail: rahul.sharma@student.upce.cz
{daniel.honc, frantisek.dusek}@upce.cz

Gireesh Kumar T
TIFAC CORE In Cyber Security
Amrita School of Engineering
Amrita Vishwa Vidyapeetham University
Coimbatore, India
E-mail: t_gireeshkumar@cb.amrita.edu

KEYWORDS
Multi robot, Area exploration, Path Planning, Frontier based, Optimization, Agent based simulation.

ABSTRACT

The paper deals with multi-robot centralized autonomous area exploration of unknown environment with static obstacles. A simple reasoning algorithm based on pre-assignment of routing priority is proposed. The algorithm tracks the frontiers and assigns the robots to the frontiers when the robots fall into a trap situation. The algorithm is simulated with various multi-robotic configurations in different environments and compared with performance indices in the MATLAB simulation environment.

INTRODUCTION

Area exploration is one of the fundamental problems of autonomous robotics. The main goal of any exploration algorithm is to gain as much new information as possible of the unknown environment within the bounded time. Autonomous area exploration algorithms find applications in space robotics, military operations, disaster management, sensor deployment etc. Area exploration deals with exploring through all unknown areas and creating a map of the environment. Most of the area exploration keeps a map of the environment and updates when an unknown region is explored.

Yamauchi pioneered the research in the frontier-based area exploration (Yamauchi 1997). A frontier is a boundary that separates known (explored) regions from unknown regions. By moving towards frontiers, robots can focus their motion on discovery of new regions. The frontier based area exploration is extended to multi-robot system (MRS) in (Yamauchi 1998). (Yan et al. 2013) presented a systematic survey and analysis of coordination of multiple mobile robot systems. A comparative study of area exploration algorithms can be seen in (Dayanand et al. 2013).

An important task in an area exploration algorithm is "how the robots choose which cell is to be explored next". The aim of any frontier based algorithm is to explore all the frontiers in shortest time possible. In multi-robot exploration, the robots coordinate each other and decide which robot will explore which frontier, based on a coordination / routing policy. Various coordination policies (Burgard et al. 2000, Burgard et al. 2005, Ma et al. 2006 and Wang et al. 2011) have been proposed in the past. In this paper, we propose a simple reasoning based routing policy determined by a pre-assigned priority. A central agent (CA) coordinates with a group of multi robot agents (MRAs) to explore an unknown environment with only static obstacles. The MRAs, with the command from CA, move from one cell to another cell, sense the environment and communicate the information to the CA. The CA will make the routing decision based on the state of the adjacent cells and the pre-assigned priority of the corresponding MRA. The CA will also keep a list of frontiers and assigns MRAs, once a MRA falls into a "trap" situation. A MRA is said to be in a *trap* situation if all the adjacent cells are either explored and/or occupied with obstacles. The route to the frontier is the shortest path found by executing the A* algorithm (Hart et al. 1968). The algorithm terminates when all the cells are explored and no more frontier cells exist. The key advantage of prioritised routing is that, the predictability of the MRAs location is improved. This will help in a decentralised distributed coordination policy, where MRAs collectively make the routing decision as MRAs can predict each other's location with the help of a pre-assigned priority of routing.

FRAME WORK FOR AREA EXPLORATION

Environment

The unknown environment is considered as grids with cells of same dimensions. If any of the cells is occupied with an obstacle, the whole cell would be considered as occupied.

Mobile robot agent

A general scheme of MRA is shown in figure 1. The following are functions of MRA

- Receive motion commands (e.g. Move **F**orward, **R**ight, **L**eft or **B**ackward).
- Execute the commands by a motion control algorithm, for e.g. PID control with wheel

Proceedings 30th European Conference on Modelling and Simulation ©ECMS Thorsten Claus, Frank Herrmann, Michael Manitz, Oliver Rose (Editors)
ISBN: 978-0-9932440-2-5 / ISBN: 978-0-9932440-3-2 (CD)

encoders and motors as actuators to make the mobile robot move from one cell to another.

- Sense the surrounding cells using (short range) distance sensors, like infra-red or ultrasonic sensors.
- Send the information about the environment to the CA.

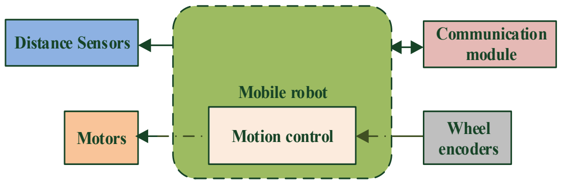

Figure 1: Scheme of Mobile Robot Agent

The mobile robots (for e.g. Sharma 2013) are assumed to be

- Point-sized and occupy only one grid at a time.
- Assigned with a fixed priority in path planning.
- Move at an even speed, and its status can be switched - between moving with a fixed speed and halting.
- In any of the following orientations - **N**orth, **E**ast, **S**outh, **W**est.
- Able to move in four directions (**F**, **R**, **L**, **B**). Each robot can move from one cell to four adjacent neighbouring cells (other than the diagonal cell) in one step.

Central Agent

The central agent can be a stationary computer or a mobile robot with sufficient computational power and a communication module. The following are the functions of the CA.

- Receive commands from the MRAs
- Update the map
- Frontier detection
- Trap detection and run shortest path planning algorithm
- Routing policy
- Send commands to the MRAs

Problem description

- To explore an unknown environment using multi-robots in the shortest time possible.
- An OPEN cell shouldn't be explored more than once and should follow the shortest path to the frontier.
- While exploring, no collision with the robots at any point of time.
- To detect any robot in a trap and assign them to move to any frontier.
- To keep track of frontiers and assign the robot to that frontier, in the shortest possible path, by executing a shortest path planning algorithm.

Assumptions

- The robots can communicate with the central computer or agent without information (packet) loss or any time delay and vice versa.
- The environment is considered as grids with cells as same length and width (L x L).
- A grid based environment is assumed (if any part of the cell is occupied with obstacle the whole cell is considered as occupied).
- The maximum sensing range of MRAs are just one cell length (L).

Terminology

Each *cell* is assumed in any one of the following states:

- OPEN – the cell is not explored and no obstacle is present (detected by the robot, but not visited)
- OCCUPIED– the cell is explored and obstacle is present
- UNKNOWN – the cell is not explored (neither visited nor detected by the robot)
- CLOSED – the cell is explored and no obstacle is present

A robot is said to be in a *trap* situation, when there are no possible moves to any of the unexplored adjacent (OPEN) cells. In other words, if the adjacent cells are either CLOSED or OCCUPIED. A robot is said to be in *on_command* state when it is in pursuit of exploring a frontier. The attributes used in the simulation are shown in Table 1 and 2.

Table 1: Attributes of MRAs

Attributes, $r(\bullet)$	**Meaning**	**Type**
r_num	ID of robot	int
r_priority	pre-assigned priority of robot	int 2D array
r_cxy	current position	int array
r_nxy	next position	int array
r_corient	current orientation	int
r_norient	next orientation	int
r_trap	if trap-1 ,else - 0	boolean
r_status	status of robot	int
r_on_comm and_route	route to the frontier	int 2D array

A robot will be in any of the following states,

$$r\_status = \begin{cases} 1, & \text{if robot is in } trap \text{ situation} \\ 2, & \text{if robot is } on\_command \text{ state} \\ 0, & \text{else} \end{cases}$$

The *fr_list* represents a list of all explored frontiers and n_r , the number of robots. The status of the frontier is assumed as,

$$fr\_closed = \begin{cases} 1, & \text{if a robot already assigned} \\ 0, & \text{if frontier is detected, but not assigned} \end{cases}$$

Table 2: Attributes of the Frontiers

Attributes, $fr(\bullet)$	**Meaning**	**Type**
fr_number	cell number of the frontier	int array
fr_length	path length to the frontier	int
fr_waiting	number of iterations waiting from detection of frontier	int
fr_route	route to reach the frontier from current location of the robot	int 2D array
fr_closed	status of frontier	boolean
fr_robo	ID of robot assigned to explore the frontier	int

FRONTIER BASED AREA EXPLORATION ALGORITHM BY PRIOROTIZED ROUTING

The algorithm terminates when there are no more frontiers or OPEN cells present. The following are the steps involved in the algorithm.

Priority assignment strategy

The MRAs will be pre-assigned with a fixed priority, which will not change during the course of the area exploration iterations. A sample assignment strategy is mentioned in Table 3 for three MRAs. The basic idea of this assignment is that, when a robot encounters a junction, for example both right and left side are unexplored and OPEN, the robot#1 chooses the left turn and robot#2 will choose right turn. If the 1st priority movement is not possible (already explored or OCCUPIED), then the robot chooses the 2nd priority and if that is also explored then, the robot chooses 3rd priority and so on. The fourth priority is assumed as a backward motion, as it is assumed that area exploration proceeds forward.

Table 3: A Sample Priority Assignment Strategy for 3 Robot MRS

Priority / **ID**	1st	2nd	3rd	4th
#1	**L**	**F**	**R**	**B**
#2	**R**	**F**	**F**	**B**
#3	**F**	**R**	**L**	**B**

New frontier detection

The master robot maintains a list of frontiers to be explored. Whenever a new frontier is discovered, the master robot will add the details of the frontier into the database. A frontier is detected when a robot senses unexplored cells (OPEN) in more than one side. The robot will move to one of the OPEN cells based on the priority assigned to it. The other unexplored cell information will be stored as frontiers and these frontiers will be explored once a MRA enters the *trap* situation.

Table 4: Cell Increments when Robot Moves from One Cell to Another with Respect to Orientation

Motion / **Orientation**	**F**	**L**	**R**	**B**
N	[-1, 0]	[0, -1]	[0, 1]	[1, 0]
S	[1, 0]	[0, 1]	[0, -1]	[-1, 0]
W	[0, -1]	[1, 0]	[-1, 0]	[0, 1]
E	[0, 1]	[-1, 0]	[1, 0]	[0, -1]

Receiving the commands from MRAs and updating the map

At every time instance, the MRAs will send information (**F**, **L**, **R** or **B**) about the adjacent cells with their ID as a header. For e.g. an information from an MRA saying #1**FR** means, the robot#1 found the cell just in front of the current cell and the cell to right side is OCCUPIED (i.e. sensors detects obstacles). The first step is to initialize the map (grid) with UNKNOWN status. The map is updated with the information from MRAs as shown below.

Algorithm 1 : Updating the map

```
1:  for i=1:n_r
2:      Get the coordinates of the sensed cells by robot
        using r(i).cxy and Table 4(motion, r(i).c_orient)
3:      Mark grid(x, y) as OCCUPIED
4:  end for
```

Routing policy, Trap detection and Frontier detection

Algorithm 2 : Routing policy, trap detection, frontier detection

```
1:   for j=1:n_r
2:       if r(i).status is not in trap or on command
3:           for i=1:r(j).prio
4:               get the coordinates (x, y)
5:               if grid(x, y) is OPEN
6:               if next position is not found
7:                   update r(j).nxy (using Table 4) and
                     r(j).norient as i
8:               end if
9:               else
10:                  if (x,y) not in fr_list
11:                  Mark it as fr and update fr_list
12:                  end if
13:              end if
14:          end for
15:          if no next position or frontier detected
16:             mark robot r(j).trap=1 and r(j).status=1
18:          end if
19:      end if
20:  end for
```

The CA keeps a list of all frontiers detected and explored. The CA checks the adjacent cells of the currents positions of the MRAs and makes the routing policy. With respect to the priority assigned for each robot, the CA checks the adjacent OPEN cells in the order of priority and assigns the first encountered cells as next exploration cell, and all the following cells as frontiers. If the CA couldn't find any OPEN unexplored cell, the robot will be marked as in the *trap* situation. Algorithm 2 shows the steps involved.

Robot assignment to frontiers in case of trap situation

A robot is said to be in a *trap* situation, when there are no possible moves to any of the unexplored adjacent cells. This condition must be carefully studied; otherwise there are chances of the mobile robot getting in to an infinite loop. When the CA detects any robot in a *trap* situation, it looks for any frontier cells yet to be explored. If any frontier cell exists, the CA runs A* shortest path planning algorithm to find shortest distance to the frontier with source node as a robot's current position and destination node as a frontier cell. The algorithm returns the shortest path, if it exists. Algorithm 3 describes the steps involved.

```
Algorithm 3 : In case of trap situation
1:     If any of the robot in trap
2:         Get the ID j of the robot
3:         if fr_list is not empty
4:             get the coordinates of frontier k
5:             find the shortest path route by A* algorithm
6:             if path exists
7:                 remove k from fr_list
8:                 Mark r(j).status=2, r(j).trap=0 and
                   r(j).on_command_route=route
9:             end if
10:        end if
11:    end if
```

Send the commands to MRAs

The CA will send the commands (**F**, **R**, **L** or **B**) to MRAs with the header as MRAs ID, based on the routing policy as explained earlier. In case of *on_ command* status the CA will send the command (*r_on_command_route*) at every time instant, one by one, calculated by executing the A* algorithm.

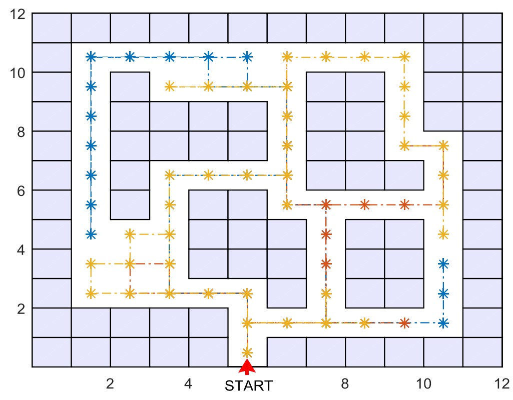

Figure 2: Stairway Exploration with Three Robots (Red, blue, orange – robot#1, #2 and #3 respectively)

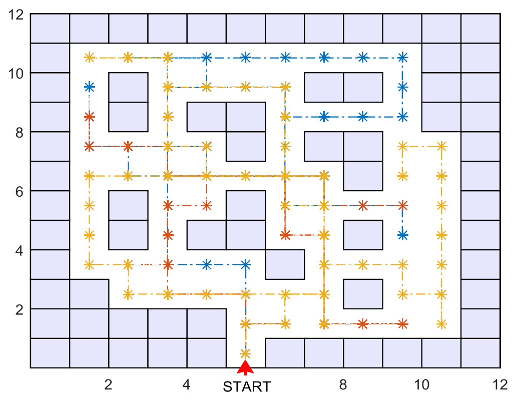

Figure 3: Unstructured Environment Exploration with Three Robots (Red, blue, orange – robot#1, #2 and #3 respectively)

SIMULATION RESULTS

The simulation studies were performed in MATLAB with two different environments - stairway and unstructured environment. The initial orientation of the robot were assumed to be the robot facing the North direction. Figure 2 and 3 shows the performance of the algorithm using three MRAs in a stairway and unstructured environment.

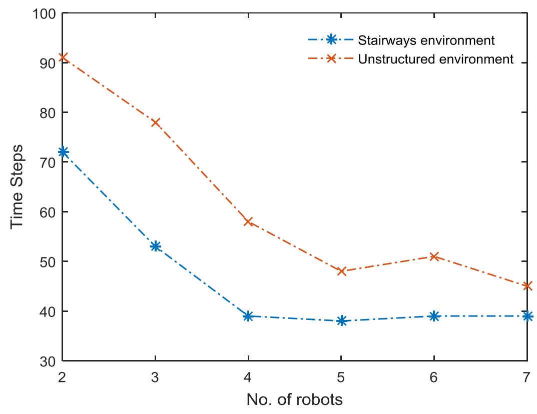

Figure 4: Time Steps Vs Number of Robots

Experiments were conducted with different robot configurations and different priorities were assigned to the MRAs. Figure 4 shows the simulation time steps taken for different robot configurations. The efficacy of different configurations can be compared with the performance indices.

Performance Indices (PI)

Five PI's were chosen to measure the performance of the different robot's configuration using the proposed algorithm.

Iterations: The number of time steps taken to explore the whole unknown environment.

Frontiers explored (n_{fr}): The number of frontiers detected by the CA and explored.

Average path length (L_r): The ratio of total number of cells visited by the robots (S_r) to total number of robots (n_r).

Exploration efficiency index (η_e): It is the measure of the efficiency of the exploration algorithm that how efficiently (without re-visiting the same cells again and again) the robots are able to explore the environment. In other words, it is a scale of how many cells the robot revisited during the course of exploration of the frontier cells. A high value represents less efficient and vice versa. It is given by the following relation,

$$\eta_e = \frac{L_r \times n_r - S_c}{S_c} \times 100\,\%$$

Where S_c represents total number of cells without obstacle.

Coverage: It is the ratio of the number of explored cells to the total number of cells without obstacle (S_c). The algorithm terminates at 100% coverage.

Table 5 and 6 shows the performance indices of the area exploration algorithm in the stairway (figure 2) and unstructured environment (figure 3) respectively. Figure 4 depicts the iterations vs the number of robots. It can be seen that, the four robots perform significantly better than 2 or 3 robots, while there is not much gain in having more than four robots.

Table 5: Performance Indices of Area Exploration in Stairway

PI \ n_r	2	3	4	5	6	7
iterations	72	53	39	38	39	39
n_{fr}	19	27	30	19	32	34
L_r	72	50	34	32	31	28
η_e (%)	145	155	133	174	222	237

Table 6: Performance Indices of Area Exploration in Unstructured Environment

PI \ n_r	2	3	4	5	6	7
iterations	91	78	58	48	51	45
n_{fr}	31	34	33	34	40	34
L_r	91	74	57	39	34	35
η_e (%)	144	198	204	161	174	232

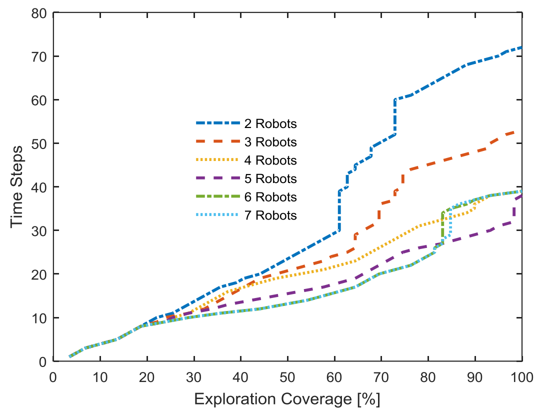

Figure 5: Coverage – Stairways Environment

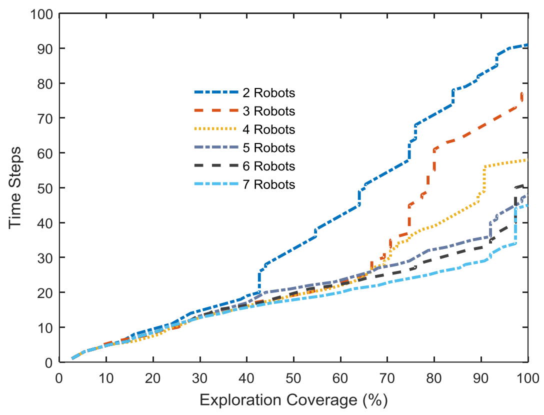

Figure 6: Coverage – Unstructured Environment

In order to say, which one is the best configuration (number of robots and pre-assigned priority) we need to have a tradeoff between the performance indices. For e.g. if only time taken is important, then the best option would be the maximum number of robots. If the PI's L_r and η_e are more important, then 6 robots for unstructured (figure 3) and 4 robots for the stairway (figure 2) would be the best option.

The main objective was to develop and simulate a frontier based area exploration algorithm with a pre-assigned priority. The performance of the algorithm heavily depends on two factors - number of robots and pre-assigned priority. The performance of the algorithm can be improved significantly, by introducing a cost function for the frontiers and the routing MRAs based on the cost function rather than waiting for the robot to fall into a *trap* situation.

CONCLUSION

The problem of area exploration was solved by a simple reasoning based frontier area exploration algorithm. The algorithm is based on centralized CA-MRA coordination. The CA makes the routing decision based on the status of MRAs and the frontiers yet to be explored. The routing policy is based on a pre-assigned priority. The main advantage of the proposed priority based approach, when compared with random routing, is that the predictability of location of MRAs increases, and this will help in a more decentralized distributed exploration policy.

As a future research direction, we are looking forward,

- To increase the sensing range of the MRAs, which will have more practical implication.
- To develop a greedy approach based on a cost function for frontier exploration rather than waiting for any robot to enter a *trap* situation.
- To develop an advanced motion control algorithm for MRAs and to validate the algorithm by real experiments in the laboratory.

This research was supported by project SGS_2016_021, Mobile Robot Motion Control with Model Predictive Controller at FEI, University of Pardubice. This support is very gratefully acknowledged.

REFERENCES

Burgard, W., Moors, M., Fox, D., Simmons, R. and Thrun, S., 2000. Collaborative multi-robot exploration. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on* (Vol. 1, pp. 476-481). IEEE.

Burgard, W., Moors, M., Stachniss, C. and Schneider, F.E., 2005. Coordinated multi-robot exploration. *Robotics, IEEE Transactions on*, *21*(3), pp.376-386.

Dayanand, V., Sharma, R. and Kumar, G., 2013. Comparative Study of Algorithms for Frontier based Area Exploration and Slam for Mobile Robots. *International Journal of Computer Applications*, *77*(8).

Hart, P.E., Nilsson, N.J. and Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, *4*(2), pp.100-107.

Ma, X., Meng, F., Li, Y., Chen, W. and Xi, Y., 2006, June. Multi-agent-based Auctions for Multi-robot Exploration. In *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on* (Vol. 2, pp. 9262-9266). IEEE.

Sharma, R., 2013. Design and implementation of path planning algorithm for wheeled mobile robot in a known dynamic environment. *IJRET: International Journal of Research Engineering and Technology*, *2*(06), pp.967-970.

Yamauchi, B., 1997, July. A frontier-based approach for autonomous exploration. In *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on* (pp. 146-151). IEEE.

Yamauchi, B., 1998, May. Frontier-based exploration using multiple robots. In *Proceedings of the second international conference on Autonomous agents* (pp. 47-53). ACM.

Yan, Z., Jouandeau, N. and Cherif, A.A., 2013. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, *10*.

Wang, Y., Liang, A. and Guan, H., 2011, April. Frontier-based multi-robot map exploration using particle swarm optimization. In *Swarm Intelligence (SIS), 2011 IEEE Symposium on* (pp. 1-6). IEEE.

AUTHOR BIOGRAPHIES

RAHUL SHARMA K., was born in Kochi, India and went to the Amrita University, where he studied electrical engineering and obtained his M.Tech degree in 2013. He is now doing his Ph.D. studies at the Department of process control, Faculty of Electrical and Informatics, University of Pardubice, Czech Republic.
e-mail: `rahul.sharma@student.upce.cz`

DANIEL HONC was born in Pardubice, Czech Republic and studied at the University of Pardubice in the field of Process Control and obtained his Ph.D. degree in 2002. He is head of the Department of Process Control at the Faculty of Electrical Engineering and Informatics. e-mail: `daniel.honc@upce.cz`

FRANTIŠEK DUŠEK was born in Dačice, Czech Republic and studied at the Pardubice Faculty of Chemical Technology in the field of Automation and obtained his MSc. degree in 1980. He worked for the pulp and paper research institute IRAPA. Now he is the vice-dean of the Faculty of Electrical Engineering and Informatics. In 2001 he became an Associate Professor.
e-mail: `frantisek.dusek@upce.cz`

Gireesh Kumar T., was born in Wandoor, India, done Doctorate in Computer Science and Engineering, Anna University Chennai in 2011, domain of Cognitive Robotics. He is currently working as Associate Professor in Cyber Security, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India.
email: `t_gireeshkumar@cb.amrita.edu`