Department of Electrical
and
Computer Systems Engineering

Technical Report
MECSE-11-2003

A Simple and Efficient Algorithm for Robotic Swarm Exploratory Tasks

K.W. Tang, R. Jarvis

MONASH
UNIVERSITY

A Simple and Efficient Algorithm for Robotic Swarm Exploratory Tasks

By:

Kai Wing TANG

Supervisor: Prof. Ray JARVIS
Co-supervisor: A/Prof. Andy RUSSELL

Abstract

The application of multiple robots to the tasks of environmental exploration has been considered as appropriate and published by a lot of researchers [Pereira et al03], [Howard et al02], [Zlot et al02], [Burgard et al00], [Yamauchi98]. This interest in the exploratory tasks is due to the fact that generating maps is the fundamental requirement of a long list of industrial and military applications like: search and rescue, reconnaissance, surveillance, hazardous cleaning, and planetary exploration etc. One of the key issues of a multiple robots exploration problem is to minimize the total time required and / or the total energy consumed. This paper presents a simple, but efficient and robust, algorithm which effectively drives the robots to different areas of the unknown environment. Compared to those published approaches, this algorithm is with compatible efficiency, but it is so clear-cut that it can be claimed as the simplest method to date.

1 Introduction

The application of robotic swarms to the unknown environmental exploratory tasks can fully demonstrate the benefits of the robotic swarms, namely, exploitation of parallelism, fault tolerance, and scalability. As a matter of fact, the collaborative generation of an unknown environmental map is the kernel of very many mobile robotic tasks. A system which can efficiently complete the exploratory task requires an effective algorithm to minimize repeated coverages. Moreover, for some real-world applications other than the exploratory task like: distributed manipulation, carpet cleaning and transportation of objects etc., also require a well-organized method to repel the mobile robots away from each other.

Once a mobile robotic swarm is placed in an unknown environment, each member of the swarm has to solve two problems simultaneously, i.e. knowing own location in the environment and where to go next. The former is the problem of localization and the latter is the navigation problem. Both are of critical importance. For the sake of simplicity, this paper focuses only on the exploration strategy and the navigation problem. The method presented here is such that by using Distance Transform (DT) [Jarvis84], the crucial requirement of steering the mobile robots away, even in an cluttered environment, can be accomplished efficiently. Our strategy has been proven to be functional in simulation.

This paper is organized as follows. Section 2 gives a brief summary of previous unknown environmental exploration strategies. Section 3 presents a short description about the application of the distance transform to the single robot navigation problem. Section 4 is about the setup of the simulated world. Section 5 is a detail depiction of the utilization of the distance transform in the multiple robots exploration problem. Section 6 presents the results, and finally, section 7 is about the conclusions.

2 Related Works

There have been a lot of different published approaches to the problem of building environmental maps by multiple robots. This section only focuses on those

related to exploration strategies. One of the most often cited works is the frontier cell of [Yamauchi98]. In this approach, each member of a robotic swarm shares a common list of frontier cells; each moves to the closest frontier cell around itself autonomously. This approach is completely distributed, but it does not fully exploit the potential of parallelism. In short, some robots may move to the same frontier cell.

[Pereira et al03] and [Howard et al02] use a potential field approach; the fields are constructed in a way so that a robot is repelled either by an obstacle or by the other robots. This approach works fairly well in those environments with few or no obstacles but NOT in the cluttered environments.

The approach of [Burgard et al00] coordinates the robots to those frontier cells in various regions of the unknown area. The algorithm iteratively chooses target points for different robots based on a trade-off between the costs of reaching the target point and the amount of information gained by reaching it. A robot is always assigned that target location which has the best trade-off between the gain and the cost for it to reach the location. This approach works satisfactorily, but it requires a central process to do the comparison. It also has the common drawback of any iterative method: the computation grows geometrically while the number of robots, or the number of frontier cells, increases.

[Simmons et al02] also use a frontier cell approach. In addition, a central agent and a frontier cell bidding mechanism are applied to the task of target allocation. Since a greedy algorithm is used, the outcome sometimes may not be highly efficient.

[Zlot et al02] use a free market architecture to control the multiple robots. Goal points which should be visited are generated by each robot; these goal points are then treated as the commodity to be exhanged in the market. This approach is highly decentralized and distributed. Consequently, it is robust and reliable. However, since a good goal point generation mechanism is absent, its performance may not be stable in some particular environments.

3 Application of DT to Navigation Problems of a Single Robot

Using DT to plan paths for navigation problems performed by a single mobile robot was first reported by [Jarvis84]. The paper demonstrated that DT can be used to get an optimal path in a static, known environment. The approach regards the point being reached as a goal point and works from the goal location back to the location of the robot. This method propagates a distance wave front through all free space cells from the goal cell. The distance wave front flows around the obstacles and sweeps through the free space cells. Consequently, every free cell which is accessible from the goal cell will be assigned a value which indicates the distance in terms of step counts from the goal cell to it. For those cells inaccessible to the goal, a negative value, e.g. -1, will be assigned. Hence, the value of the goal cell will be zero, all of the goal's FREE 4-neighbours will be one, the diagonal 4-neighbours will be $\sqrt{2}$ etc. Using these values as a potential field, the shortest distance from the robot to the goal cell is the path formed by walking down hill via the steepest descent path. In the case of no down hill path existing, a conclusion of inaccessible goal can be drawn.

Figure 1: Distance transform and the path formed between a robot and its goal point.

Figure 1 shows an example of a DT map formed by a robot in a cellular world. The black blocks are obstacle cells. The number in each cell, which is rounded to integer, represents the distance relative to the goal point in terms of step counts. The shortest path formed by the steepest descent is superimposed on the DT map.

In next section we describe the setup of the simulation, then we will elaborate the details of utilizing DT to work efficiently in problems of multiple robots in section 5.

4 Setup of the Simulated World

The simulation represents the real world by a grid of rectangular cells (50 by 50 cells). Each rectangular cell represents one of the classes of: space, obstacle, and robot; and each cell is in either states of: known or unknown. In the beginning of each simulated exploration, except the robot class, all other classes–both space and obstacles–are in an unknown state.

Various swarm sizes, from 2 to 10 robots, are tried in such an environment.

For the experiments, full communication is employed among the participating robots. They instantaneously broadcast the detected range data to all other robots. Accordingly, they all share a common environmental map.

Also, each robot is equipped with a range sensor of five (5) cells in range.

For the initial locations of the robots, if they are randomly placed, the collective co-operating results cannot be shown effectively. Consequently, they are placed within an area of a certain limited radius (3).

Every robot of the swarm runs a repetitive cycle of: scanning the environment, planning a path, and following the planned path. Since the explored environmental map is shared among every robots, each robot gets the information of: where the unknown areas are, the locations of the other robots, and the locations of the known free space and the known obstacles, before it plans its path.

Once a robot realizes it is too close to one or more other robots, it will stop. As a result, all robots in the congested zone stop at once. Then, randomly, one of them will re-plan a new path and leave the zone. A short while later, another robot follows the same process. Hence, they disperse one by one from the congested zone and a collision is avoided.

5 Application of DT to Navigation Problems of Multiple Robots

During the path planning process, every robot runs two different DTs to get two different DT maps, namely, Self-Map and Others-Map.

The Self-Map is created by treating the location of a robot itself as the goal point, the locations of the other robots as obstacles, and the unknown cells as free spaces. Consequently, every cell in this Self-Map holds a value which indicates the shortest estimated collision-free step distance from the robot to that cell. If the value is negative, this implies the cell is inaccessible (either a known obstacle or another robot).

The Others-Map is created by treating the locations of ALL other robots as the goal points, itself as an obstacle, and the unknown cells as free spaces. Accordingly, every cell in this Others-Map holds a value which indicates the shortest estimated collision-free step distance from the closest robot, among ALL the other ones, to that cell. Similarly, a negative value implies the cell is inaccessible to ALL the other robots.

With these two DT maps, the path planning algorithm is very simple. Let:

- x, y be the coordinates of a cell of either of these two maps, then
- $S_{x1,y1}$ = the cell's value at x1,y1 of the Self-Map.
- $O_{x2,y2}$ = the cell's value at x2,y2 of the Others-Map.

The algorithm is to find an *unknown* cell (x, y), which maximizes

$$O_{x,y} - S_{x,y}$$

subject to conditions:

$$S_{x,y} > 0 \quad (1)$$
$$O_{x,y} \geq S_{x,y} \quad (2)$$
$$O_{x,y} > 0 \quad (3)$$

If there are more than one unknown cell met these requirements, move to the closest one.

If no unknown cell fits ALL the conditions, then, dependent on the optimal criterion, perform:

- If the requirement is to conserve the collective energy consumption, then stop.
- If the requirement is to complete the exploration in the shortest period, then move to the nearest unknown cell.

The idea of this algorithm is to make sure that any unknown cell is always explored by the nearest robot. Here, a robot does not approach the frontier cells, but the unknown cell whose estimated collision-free step distance is nearer to itself than to the others.

Figure 2 shows the trajectories of an exploratory task performed by three robots. The bigger circle is the initial position, whereas the three smaller circles are the end positions of the three robots.

Figure 3 shows the locations of the three robots after each has taken five steps. Grey and white areas are unknown and known cells, respectively. Here, an identity is given to each robot (A, B and C).

Figure 4 is the Self-Map of robot A. The darkest cells are the locations of the three robots and those detected obstacles. The DT with a lower value is shown as a brighter cell.

Figure 5 is the Others-Map of robot A.

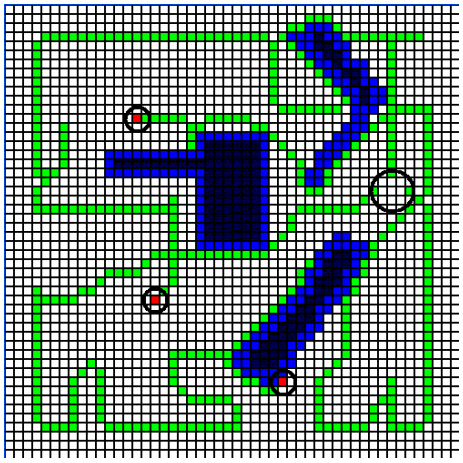

Figure 2: Trajectories of three robots.

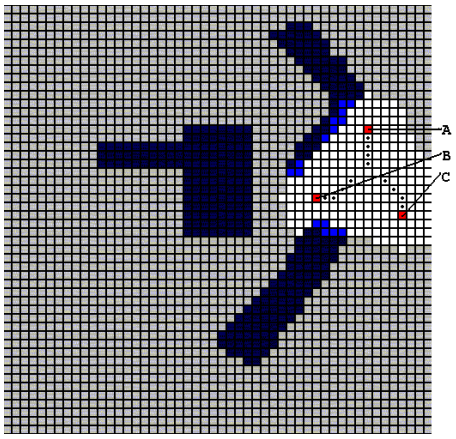

Figure 3: Snapshot of the exploration after each robot taken five steps.

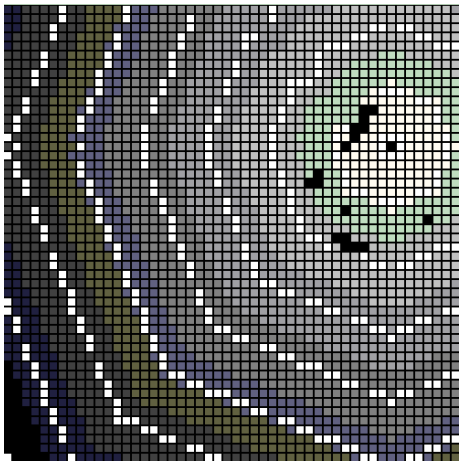

Figure 4: Self-Map of robot A, white cells are contour lines.

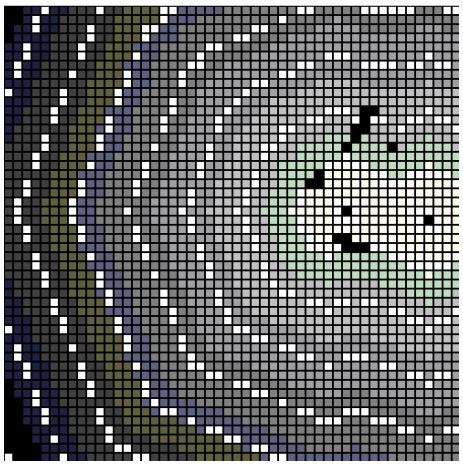

Figure 5: Others-Map of robot A, white cells are contour lines.

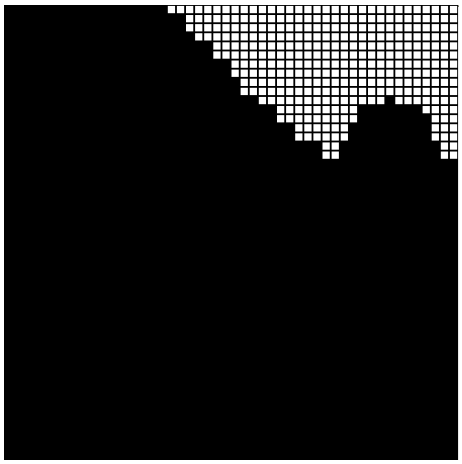

Figure 6: The cells fit the conditions of (1) to (3), deduced from Self and Others DT maps of robot A.

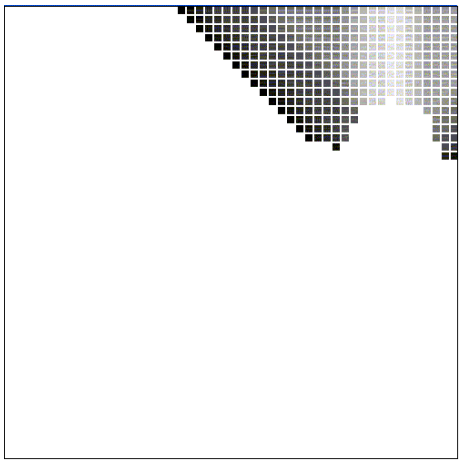

Figure 7: Magnitude of Others-Map minus Self-Map (robot A).

Figure 6 is a binary image. The white cells are those which fit ALL the conditions (1) to (3), (robot A).

Figure 7 shows the magnitude of $(O_{x,y} - S_{x,y})$ for the white cells shown in Figure 6. The brighter cell is of a larger magnitude.

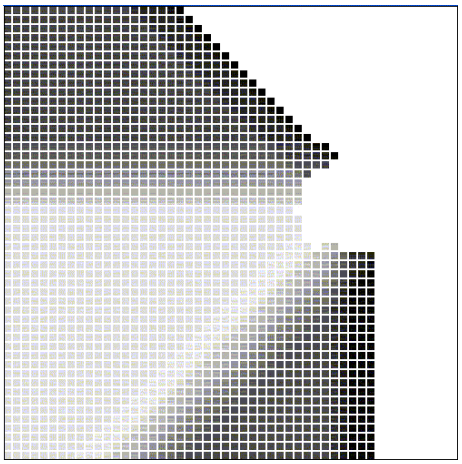

Figure 8: Magnitude of Others-Map minus Self-Map (robot B).

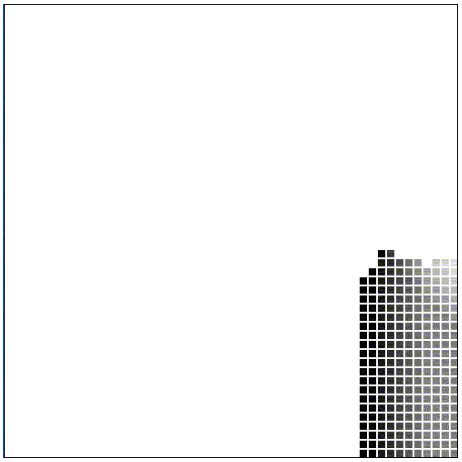

Figure 9: Magnitude of Others-Map minus Self-Map (robot C).

Figure 8 and 9 show the magnitude of $(O_{x,y} - S_{x,y})$ of robot B and robot C, respectively.

From these maps of magnitudes of differences, we can see that our method drive the three robots to three distinct areas.

6 Results

Four different environments named as 'Maze', 'Random', 'Empty' and 'Chamber' are used as the testing environments (Figure 10). Exploration simulation, with the number of robots varied from 1 to 10, initial locations randomly placed, is run in these environments for two hundred times (a total of two thousand times for each environment). The required energy for a robot moving to one of its 8-neighbouring cells is one unit, whereas the energy consumed for staying in the same cell is 0.3 unit. The efficiency comparison of collective energy consumption and the total time required are plotted in figures 11 to 14. Here, a 100% efficiency in time means a n-robot swarm has to spend 1 / n of the required time of a single robot to complete the exploration. A 100% efficiency in energy means a n-robot swarm has to consume the same amount of the energy of a single robot's exploration. For example, in fig. 12, the time efficiency and energy efficiency of a 4-robot swarm are 95% and 103%, respectively. These figures imply the swarm can complete the exploratary task in

$$1/4/0.95 = 0.263$$

of the time required, but just consumes $1/1.03 = 0.97$ of the energy required, by a single robot.

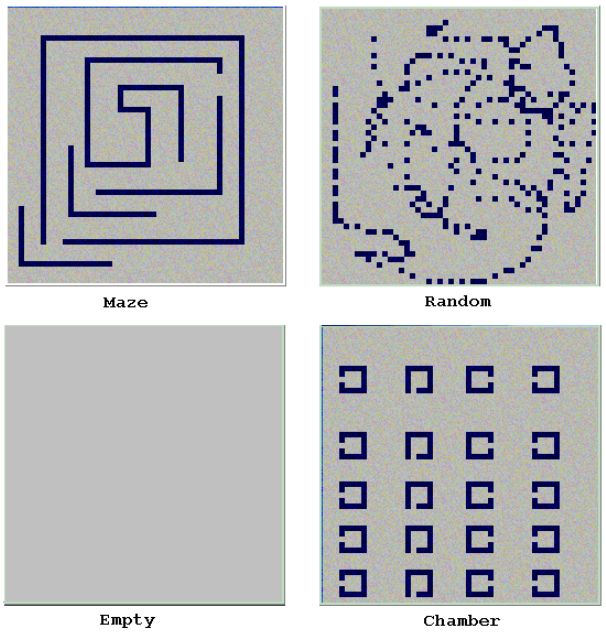

Figure 10: Four different testing environments.

Optimization in energy consumption is the criterion for all of the simulation. We can see that this algorithm can maintain an over 90% performance when number of robots is less than eight, except in environment 'Maze'. Since 'Maze' consists of a lot of narrow alleys which inhibit the exploitation of parallelism, 'Maze' is not a good environment for the application of robotic swarms.

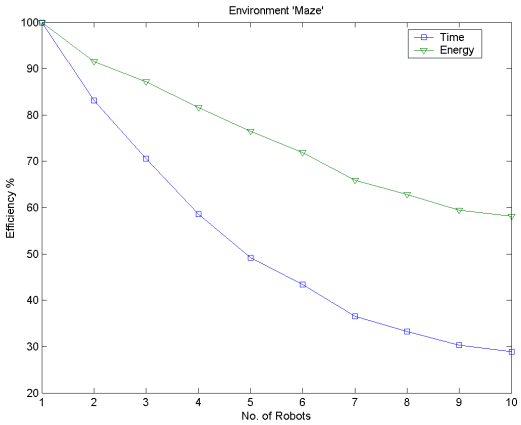

Figure 11: Efficiency in environment 'Maze'.

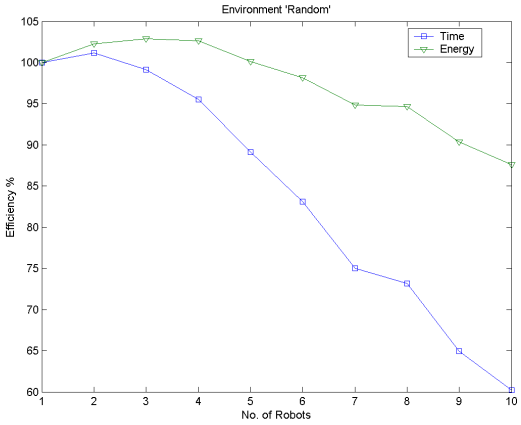

Figure 12: Efficiency in environment 'Random'.

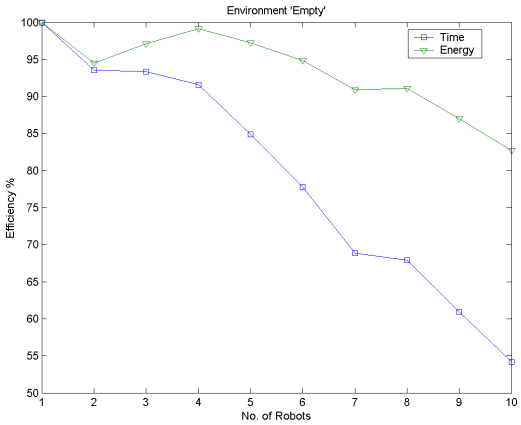

Figure 13: Efficiency in environment 'Empty'.

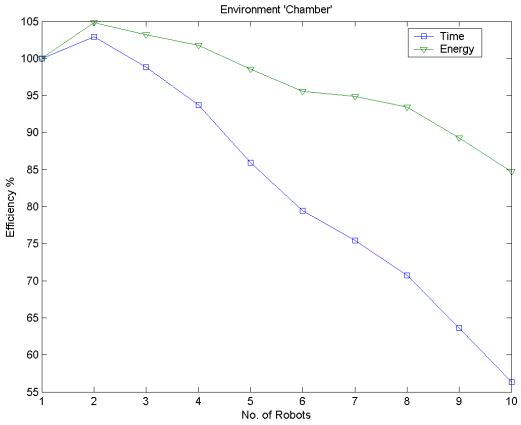

Figure 14: Efficiency in environment 'Chamber'.

Even so, this algorithm can provide a 80% performance in the aspect of energy consumption when the number of robots is less than five.

7 Conclusions

In this paper we present a very simple method to handle the problem of swarm navigation for exploratory tasks. It is completely distributed, performs with high efficiency, works perfectly in cluttered environments and finally, is computationally economic. It has the merits, but without the shortcomings, of those published approaches.

It works well in a model where the swarm members share a common environmental map. Moveover, we have also developed a variation form which works efficiently for a model where the robots do not broadcast the newly acquired range data [Tang & Jarvis03].

References

[Burgard et al00] W. Burgard, D. Fox, M. Moors, R. Simmons, S. Thrun. *Collaborative Multi-Robot Exploration*, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2000.

[Howard et al02] A. Howard, M.J. Mataric, G.S. Sukhatme. *Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem*, 6th International Symposium on Distributed Autonomous Robotics Systems DARS02, Fukuoka, Japan, 2002.

[Jarvis84] R.A. Jarvis. *Collision-free trajectory planning using distance transforms*, Proceedings of National Conference and Exhibition on Robotics, Melbourne, 1984.

[Pereira et al03] G. Pereira, A. Das, V. Kumar, M. Campos. *Decentralized motion planning for multiple robots subject to sensing and communication constraints*, Proc. of the Second Multi-Robot Systems Workshop, Kluwer Academic Press, pp. 267–278 2003.

[Simmons et al02] R. Simmons, D. Apfelbaum, W.Burgard, D. Fox, S. Thrun, H. Younes. *Coordination for multi-robot exploration and mapping*, Proceedings of the National Conference on Artificial Intelligence (AA AI) 2000.

[Tang & Jarvis03] K.W. Tang, R. Jarvis. *An Evolutionary Computing Approach to Generating Useful and Robust Robot Team Behaviours*, MECSE-10-2003, Departmental Technical Report, Dept. Electrical and Computer Systems Engineering, Monash University 2003.

[Yamauchi98] B. Yamauchi. *Frontier-based exploration using multiple robots.*, Proceedings of the Second International Conference on Autonomous Agents, Navy Research Laboratory, Washington, DC 20375-5337, 1998.

[Zlot et al02] R. Zlot, A. Stentz, M.B. Dias, S. Thayer. *Multi-Robot Exploration Controlled By A Market Economy*, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2002.