Available online at www.sciencedirect.com
ScienceDirect
Robotics and Autonomous Systems 54 (2006) 945–955
Robotics and Autonomous Systems
www.elsevier.com/locate/robot

Distributed multi-robot coordination in area exploration

Weihua Sheng[a,*], Qingyan Yang[b], Jindong Tan[c], Ning Xi[d]

[a] School of ECE, Oklahoma State University, Stillwater, OK 74078, United States
[b] Systems Division, Iteris Inc., Madison Heights, MI 48071, United States
[c] ECE. Department, Michigan Technological University, Houghton, MI 49931, United States
[d] ECE. Department, Michigan State University, East Lansing, MI 48824, United States

Received 6 December 2004; received in revised form 16 June 2006; accepted 16 June 2006
Available online 17 August 2006

Abstract

This paper proposes a reliable and efficient multi-robot coordination algorithm to accomplish an area exploration task given that the communication range of each robot is limited. This algorithm is based on a distributed bidding model to coordinate the movement of multiple robots. Two measures are developed to accommodate the limited-range communications. First, the distances between robots are considered in the bidding algorithm so that the robots tend to stay close to each other. Second, a map synchronization mechanism, based on a novel sequence number-based map representation and an effective robot map update tracking, is proposed to reduce the exchanged data volume when robot subnetworks merge. Simulation results show the effectiveness of the use of nearness measure, as well as the map synchronization mechanism. By handling the limited communication range we can make the coordination algorithms more realistic in multi-robot applications.

Keywords: Multi-robot coordination; Exploration

1. Introduction

1.1. Motivation

Exploration of unknown environment is an important topic in mobile robot research due to its wide real-world applications, such as search and rescue, hazardous material handling, military actions, etc. Recently, multi-robot exploration has received much attention from the research community [1–8]. The obvious advantage of multi-robot exploration is its concurrency, which can greatly reduce the time needed for the mission. Coordination among multiple robots is necessary to achieve efficiency in robotic explorations. Other desirable characteristics such as reliability and robustness also require cooperation among multiple robots [3].

Generally, the problem of multi-robot exploration can be stated as follows: *n identical robots set out to explore an unknown area. Each robot is equipped with sensing, localization, mapping and limited-range communication capability. Design a coordinated exploration algorithm to carry out the mission reliably and quickly.*

As stated above, the major criteria of the coordination algorithm are its reliability and efficiency. Reliability requires that the system is tolerant to the failure of robots. Since centralized coordination algorithms usually suffer from the single-point-of-failure problem, it is highly desirable to adopt a distributed coordination algorithm so that the system reliability can be improved. Efficiency requires that the exploration mission be finished with least cost, such as the exploration time, robot traveling distance, communication volume, etc. Recently, negotiation-based algorithms have been proved to reduce the time and traveling distance in previous work on multi-robot exploration [3–5]. The main idea is to use a bidding mechanism to choose the best one from several candidate movement plans submitted by multiple robots. As a result, the bidding algorithms lead to locally minimized cost. We adopt the bidding mechanism in designing our algorithm, but in a totally distributed fashion.

On the other hand, communication is one of the most important aspects in multi-robot coordination. In most of the existing research work, the availability of a communication link, either direct or multi-hop, between any two robots is always assumed. However, this assumption may not be true in many situations. For example, since the communication

* Corresponding author. Tel.: +1 4057447590; fax: +1 4057449198.
E-mail address: weihua.sheng@okstate.edu (W. Sheng).

0921-8890/$ - see front matter
doi:10.1016/j.robot.2006.06.003

range of each robot is limited, one or several robots may move out of the communication limits of all other robots and thus the communication network may be partitioned. The existence of interference in communication channels can be another cause of the separation of communication network. As a result of the network partition, different robots may have different views about the world they are working in, which poses serious challenges to the design and implementation of robust coordination algorithms for multiple mobile robots.

To summarize, the following questions need to be answered in order to implement a successful multi-robot exploration system: (i) how to design a totally distributed, bidding-based coordination algorithm? and (ii) how to accommodate the limited-range communications so that the mission can be completed without substantial performance downgrade?

This paper aims to answer the above two questions. Although the application scenario this paper focuses on is the exploration of an unknown environment, we believe the proposed concepts and methodologies can be extended to other multi-robot applications.

1.2. Previous work

Although there has been considerable work on single autonomous robots, much work remains to be done in multi-robot coordinated localization, mapping and exploration [9]. Yamauchi [6] developed a distributed, asynchronous multi-robot exploration algorithm which introduces the concept of frontier cells, or the border area between known and unknown environments. Their basic idea is to let each robot move to the closest frontier cell independently. This brings the fault tolerance capability. However their algorithm does not achieve sufficient coordination so that multiple robots may end up moving to the same frontier cell, which introduces inefficiency. Based on the similar frontier concept, Simmons et al. [4] developed a semi-distributed multi-robot exploration algorithm which requires a central agent to evaluate the bidding from all the other robots to obtain the most information gain while reducing the cost, or the travelling distance. Fujimura and Singh [7] presented a cooperative terrain acquisition strategy for distributed mobile agents, which can handle heterogeneous robots. However, efficiency is not their main concern. Grabowski et al. [10] investigated the coordination of a team of miniature robots that exchange mapping and sensor information. In their system, a robot will play as a team leader that integrates the information gathered by the other robots. This team leader directs the movement of other robots to unknown areas. Jung and Zelinsky [11] developed an action selection scheme for behavior-based agents which was verified in a two-robot cleaning task. Recently, Zlot et al. [3] developed a multi-robot exploration strategy based on a market economy, which inherits the basic concept from the bidding protocol used in Simmons et al.'s work. Their algorithm can improve the reliability, robustness and efficiency of the exploration through negotiation in a market architecture. However, the goal generation algorithm they used is not as efficient as the frontier-based algorithm and the negotiation process is complicated. Gerkey and Mataric proposed an auction method for multi-robot coordination in their MURDOCH system [5]. The use of an "auctioneer" agent is similar to the central agent method used in Simmons et al.'s work. In most of the previous work, the communication between robots is assumed to be perfect, which makes their algorithms unable to handle unexpected, occasional communication link breakdowns. Roy and Dudek [8] proposed a rendezvous-based exploration method for multiple robots to meet in order to accommodate the limited communication range. However, the rendezvous behavior may make the overall mission inefficient. Recently, Burgard et al. [12] developed a coordinated multi-robot exploration algorithm to increase the efficiency of exploration in various environments. They investigated the impact of the limited communication range upon the exploration time.

To summarize, we have the following observations regarding the current research work in multi-robot coordination. First, there is no totally distributed, negotiation-based method for multiple robots' coordination. Previous work adopts either central agents or group leaders for winner selection. Due to the use of central agents or group leaders, the negotiation protocols become complicated and the load on different robots is unbalanced. Another disadvantage of using central agents is that when the network becomes partitioned, new robots should be designated as the central agent of different subnetworks. Therefore, it is desirable to have a simple, totally distributed, load-balanced coordination algorithm. Second, beside the work conducted in [8,12], most of the existing work in multi-robot applications does not address the problems caused by the limited communication ranges. Instead, a fully-connected communication network is usually assumed, which is not true for many real world applications. In this paper, we propose that the coordination algorithm should be redesigned with a goal to minimize the performance downgrade and overhead. The work that is closest to this paper is [12]. However, the differences between our work and theirs are: (i) we provide a simple distributed bidding algorithm for the coordination; (ii) we not only consider the limited communication range, but also introduce a nearness measure in the bidding algorithm so that the robots tend to stay together.

The remainder of the paper is organized as follows. Section 2 describes the model of multi-robot exploration and introduces the nearness measure for a single robot. Section 3 describes our distributed, bidding-based algorithm for exploration. Section 4 presents the map synchronization mechanism that minimizes the data volume in map exchanging. Simulation and testing are presented in Section 5 and Section 6 concludes the paper.

2. Multi-robot exploration model

2.1. Environment and robot model

In this section, we define the environment model and the mobile robot model. The environment to be explored is modelled as a 2D occupancy grid of a specified resolution. There are stationary obstacles of arbitrary shape in the area. During the exploration, each cell of the grid has one of three

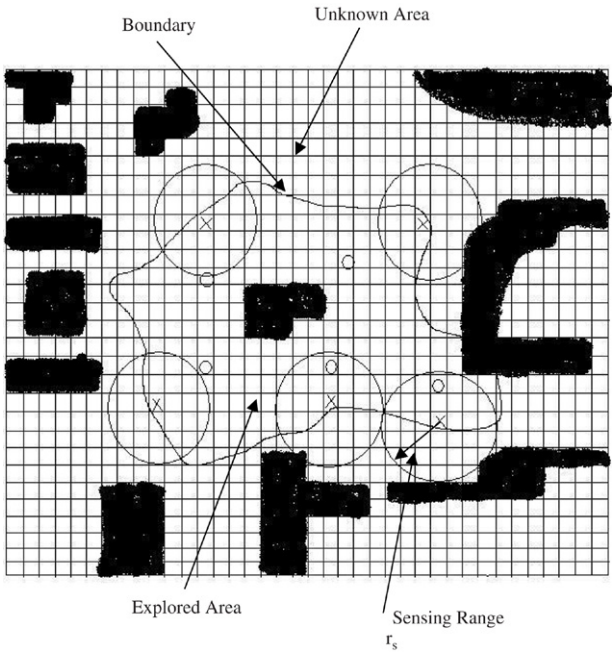


Fig. 1. A multi-robot exploration in an unknown environment. The circles represent the current robot positions and the crosses represent the next target positions.

statuses: occupied, free or unknown. Here "occupied" means that the cell is occupied by obstacles, such as a wall; "free" means that no obstacle exists in this cell; and "unknown" means that the occupancy of the cell is not known yet. The physical shape of a robot can be modelled as a square. For simplicity, its dimension is set as the size of a grid cell.

It is assumed that each robot R_i has mapping, localization and communication capabilities. It can sense the neighboring cells using range sensors such as a sonar or a laser-ranger-finder. Its sensing range is denoted by a circle of radius r_s centered on the robot. When the robot is travelling, it does not carry out sensing and mapping. Each robot is capable of localizing itself with respect to its own local map. The communication capability enables the robot to directly talk to other robots within the communication range r_c, or to a remote robot in the same subnetwork through multi-hop communication. Also, we assume that the local time on each robot is synchronized at the beginning of the mission, and the clock drift on each robot is within reasonable error tolerance. A typical scenario of multi-robot exploration is shown in Fig. 1.

2.2. Nearness measure

For each robot, the capability of maintaining communication links with other robots can be measured by its distances to other robots. Hence, the following nearness measure is defined to characterize its communication capability

$$\lambda_i = \mathrm{e}^{-\frac{d_1}{r_c}} + \alpha \mathrm{e}^{-\frac{d_2}{r_c}} + \cdots + \alpha^{n_k-2}\mathrm{e}^{-\frac{d_{n_k-1}}{r_c}}. \quad (1)$$

Here $d_1 \le d_2 \le d_3 \le \cdots \le d_{n_k-1}$ is an increasing order of $d(R_i, R_j)$, $j \in (1, 2, \ldots, n_k)$ and $j \ne i$. k is the number

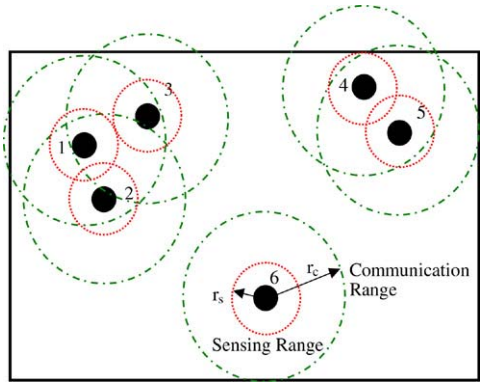


Fig. 2. Robot 6 has the least nearness measure among all the robots.

of robots within the same subnetwork. r_c is the communication range. α is a positive fading factor, which is smaller than 1. The use of the fading factor will help to disperse the robots and maximize the coverage area, since the nearness measure mainly depends on the closest neighbor.

As an example, in Fig. 2, robot 6 has the least nearness measure because its distances to other robots are all bigger than the communication range r_c. The nearness measure will be used in the coordination algorithms to increase the possibility that robots can communicate to each other.

3. The distributed bidding algorithm

To achieve reliability and robustness, the exploration algorithm should be totally distributed and symmetric. All the n robots start from initial positions which are close to each other and the relative positions are known to all. Each robot seeks to explore the area with the maximum information gain while reducing its cost, which is usually the distance from the current position to the target position. Instead of using a central-agent-based distributed model for the bidding process [3,4], we develop a simple, totally distributed bidding algorithm.

3.1. Sensing and mapping

All the robots work asynchronously. At any time instant, a robot is in one of the following three states: (1) sensing and mapping, (2) bidding and (3) travelling, as shown in Fig. 3. The robot uses its range sensor to detect the status of the cells (i.e., free or occupied) within its sensor range. The robot updates its local map when it obtains new information about the environment, either through its own range sensor or through the map updates from other robots within the same subnetwork. The robot also broadcasts its newly-obtained map information to all the other robots in the same subnetwork, which then update the local maps in their memory. Then based on the local maps, frontier cells are identified. Here a frontier cell is a "free" cell that is adjacent to at least one "unknown" cell, as shown in Fig. 4. Once the sensing and mapping step is finished, the robot determines if there is any current bidding session in its own subnetwork that is almost closed by checking the current time and the best bidding start time. If yes, it will wait for the next bidding

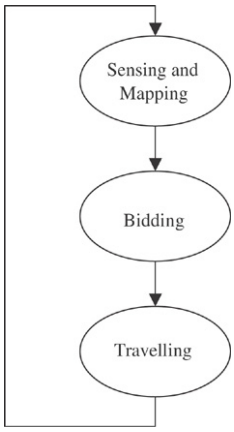


Fig. 3. The state transition of each robot.

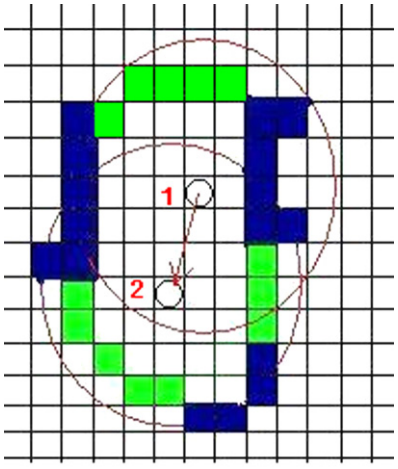


Fig. 4. As the robot moves from position 1 to 2, new frontier cells are found. The gray (green in web version) ones are frontier cells and the black (blue in web version) ones are obstacle cells.

session instead of participating in the current session. Otherwise, it will send out the new map updates to all the other members in the subnetwork. This checking step is very important to maintain information consistency when there is communication delay, as will be discussed later in the paper. If the map is not complete, which means there are more frontier-cells, the robot will decide the next target frontier cell to move to.

3.2. Bid calculation

The robot calculates the information gain I_i for each frontier cell i based on (i) the current local map, (ii) the current positions of other sensing-and-mapping robots within the same subnetwork and (iii) the target cell positions of those travelling robots within the same subnetwork. The information gain I_i for a specific frontier cell i is the number of unknown cells within the sensor range, but not within the sensor range of any other sensing-mapping robots or the target cells of those travelling robots. This information gain definition keeps different robots separated so their sensing areas will not overlap too much. The robot calculates the shortest travelling distance D_i to each frontier cell i. Due to the existence of obstacles and unknown areas, the shortest distance may not be the straight line distance. A quick algorithm based on Dijkstra's shortest distance algorithm [13] is developed. First, the partially known map is converted into a connectivity graph which describes the neighboring relationship among cells and the weights on the graph represent the travelling distance between neighboring cells. Second, on this connectivity graph, Dijkstra's algorithm is applied to find the shortest path which avoids the obstacles and the unknown areas.

In order to calculate the nearness measure λ_i of the robot at each frontier cell i, the travelling distances from that cell to each of the other robots' current positions or target positions are calculated. Then the nearness measure λ_i can be calculated using Eq. (1). The net gain g_i for frontier cell i is the weighted combination of the three measures:

$$g_i = \omega_1 I_i - \omega_2 D_i + \omega_3 \lambda_i. \tag{2}$$

Here $\omega_i, i \in \{1, 2, 3\}$ are the positive weights. By changing the weights, the importance of each component can be varied. The bid B of a bidding robot is the maximum net gain of all the frontier cells

$$B = \max_i g_i.$$

3.3. Bidding

The bidding robot broadcasts its bid B to all the other robots within the same subnetwork and waits for a constant bidding time t_{bid} to hear responses from them. If during that bidding time, there is no other robot participating in the current bidding session, or no other robot provides better bids, this robot wins the bidding. If during that time, one or several other robots finish sensing and mapping and send in the new map updates, this bidding robot recalculates its bid based on the new local map and broadcasts the bid again. If this robot receives any better bid from other robots, it will wait for the winner to declare its ID. Once the winner is identified and its assigned target is known, this robot recalculates the bid based on the updated information and the process repeats. The winning robot then moves to its target frontier cell and starts the sensing and mapping–bidding–travelling cycle again. All the robots stop when no new frontier cell is available. The skeleton of the bidding algorithm is shown in Fig. 5.

Obviously the bidding time t_{bid} is an important parameter. If t_{bid} is very small, it is highly possible that fewer robots will participate in the bidding. If t_{bid} is very big, it is very likely that many robots will participate in the bidding, which implies a better bid will be generated but the waiting time will be longer. In one extreme case if the bidding time approaches 0, it is very likely that there will be no more than one bidder in each session and actually each bidding robot wins its own session. In the other extreme case, if the bidding time is large enough, all the robots within the same subnetwork will participate in the bidding session and the exploration will significantly slow down.

3.4. Communication delay

As discussed in the coordination algorithm, when a robot finishes its sensing and mapping, it will check if the current

```
void robot_exploration()
{
        mode=SENSING-MAPPING;
        local_map=sensing_mapping();
        broadcast(local_map, robots);
        while (!is_map_complete(map))
        {
                mode=BIDDING;
                winner_declared=FALSE;
                bid_finish=FALSE;
                while ( !bid_finish )
                {
                        my_bid=calculate_bid(map, robots);
                        broadcast(my_bid, robots);
                        start_timer(bid_time);
                        timer_expired=FALSE;
                        while ( (!timer_expired || (better_bid_received()) && !
                        winner_declared )
                        {
                                if ( new_bidder_enter() )
                                break;
                        }

                        if ( timer_expired && ! winner_declared )
                        {
                                bid_finish=TRUE;
                                broadcast(my_ID, robots);  // I am the winner!
                                best_bid_begin_time=INFINITY;
                                mode=TRAVELLING;
                                move_to(my_bid.destination);
                                local_map=sensing_mapping();
                                update_globalmap(map, local_map);
                                if((current_time()-best_bid_begin_time) >
                                SEND_MAP_WINDOW)
                                {
                                        while(!winnder_declared);
                                }
                                broadcast(local_map, robots);
                        }
                }
        }
}
```

Fig. 5. The skeleton of the bidding algorithm on each robot.

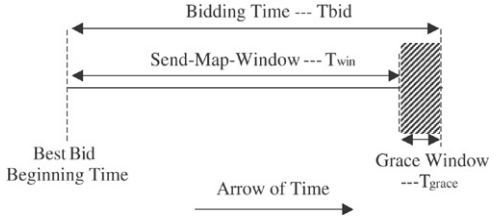


Fig. 6. The time parameters in coordination algorithm.

bidding session is near the end. If yes, it will not participate in the current session, otherwise, it will send out the new map updates. The basic idea behind this is to introduce a grace window T_{grace} to overcome the problem of information inconsistency caused by communication delay. The relationship between Send-Map-Window, bidding time and the grace window is illustrated in Fig. 6.

The information consistency concern of the distributed coordination algorithm is: *in the same bidding session, do all the bidding robots make their decisions based on the same map?* The answer is *yes* if the time parameters in the coordination algorithm are carefully chosen. We have the following proposition regarding the constraint on the grace window time and the maximum communication delay:

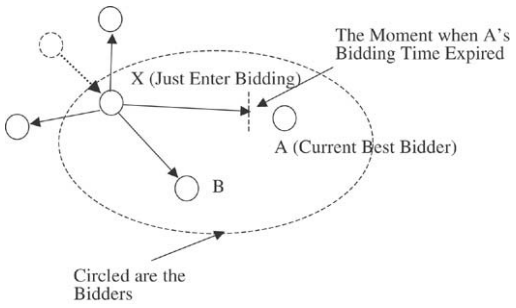


Fig. 7. A snapshot when map inconsistency occurs within the same subnetwork.

Proposition 1. *Within the same subnetwork, all the bidding robots calculate their bids based on the same map if the grace window time T_{grace} is greater than the upper limit of the communication delay $\overline{t_d}$. i.e.,*

$$T_{\text{grace}} > \overline{t_d}.$$

Proof. Without losing generality, considering the scenario illustrated in Fig. 7 where all the five robots are in one subnetwork. According to the distributed coordination algorithm, when Robot X finishes the sensing and mapping, it checks if the difference between the current time and the beginning time of the best bidder (assuming it is A) is greater than the Send-Map-Window T_{win}. If yes, it will not participate in this bidding session and will hold its map updates until the winner is declared, otherwise it will send out the new map updates. Supposing bidder X participates in this session, it sends out the new map updates to other robots, so all the bidding robots $A, B, \ldots$ and X are required to calculate their bids based on the same map. Since bidder A is the current best bidder, according to the coordination algorithm, it is waiting for the expiration time to declare itself as winner. All the other bidders are just waiting for the **Winner-Declared** message and they should be able to receive the map updates from X as long as A does not declare itself as winner. So the only possibility that all the bidders do not have the same map occurs when A declares itself as winner before the map updates from X arrive at it, as shown in Fig. 7. Because in the worst case, it takes the new map updates a time of $\overline{t_d}$ to get to A. Therefore, if the grace window time T_{grace} is greater than the maximum communication delay $\overline{t_d}$, we can guarantee that A will receive the map updates before it declares. This concludes the proof. ■

4. Map synchronization

By using the nearness measure in the distributed bidding algorithm, the robots tend to stay close to each other. However, it is still possible that the communication network is partitioned. This is due to the fact that the bidding algorithm makes decisions only based on a static "snapshot" of the robot locations while the multi-robot system is highly dynamic and the robot locations are constantly changing. For example, if at some time instant, a robot moves out of the communication range of all other robots, it becomes "communication-isolated"

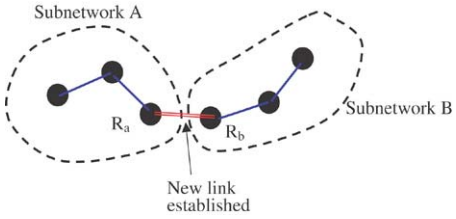


Fig. 8. Two subnetworks merge into one.

and no information can be transmitted between this robot and other robots. It is also possible that subnetworks are formed due to the interference on some critical communication links.

On the other hand, when the distances between robots change as the multiple robot system evolves, new communication links will be established and the network connectivity will be changed. Therefore, subnetworks may merge to form bigger subnetworks or the overall network becomes fully-connected again. Hence, map data should be exchanged between the merged subnetworks to achieve consistent internal maps in the corresponding robots. This raises the following question: *how to minimize the volumes of the map data exchanged between the subnetworks?* Take an example, as shown in Fig. 8, robot R_a in subnetwork A establishes a communication link with robot R_b in subnetwork B so the two subnetworks merge into one. In an ideal situation, robot subnetwork A should send to robot subnetwork B what B does not know, and vice versa. The difficulty is, *robots in subnetwork A have little knowledge about what map information robots in subnetwork B have known and not known.* A brute-force solution is that robot R_a sends all the explored map data to robot R_b and vice versa. After that, R_a and R_b send the received map data, through multi-cast, to all the other robots in subnetwork A and subnetwork B respectively. Although feasible, this solution incurs considerable redundancy, or unnecessary data exchange when both subnetworks have a large explored map. For example, when the mission approaches its end and most of the area has been explored.

4.1. Schemes for map storage and map update tracking

In order to solve the above map synchronization problem, we first propose a scheme to store the map in each robot. Then based on the map storage, another scheme is developed to track the map updating in each robot.

Each robot maintains a raw map table and a local map. The raw map table consists of the map information discovered by each robot. Therefore, the information in the raw map table may be redundant since different robots' sensing areas may overlap. The raw map table is indexed by the discoverer and the map data sequence number as shown in Table 1. The sequence number represents the order of map data discovery in each robot. The local map is derived from the raw map table using a map fusion mechanism.

Here $\Delta M_{pq} = \{\bigcup(x, y, \text{Value})\}$ represents the map information discovered by robot R_p at its qth sensing and mapping. x and y represent the coordinates of the cell and Value represents the status of the cell. Robot R_i gets new map information, either through its own exploration, where the discoverer of that map information is the robot itself, or through communication with other robots, where the discoverer of each part of the map is indicated. According to the discoverer of the map information, Robot R_i stores the map information in different columns of the raw map table, according to the sequence number. For example, in Table 1, Robot R_i has map information discovered by Robot R_1, R_2, $R_3, \ldots, R_n$ up to the sequence number i_1, i_2, $i_3, \ldots, i_n$ respectively. Let $\Delta M_i(p, q] = \bigcup_{j=p+1}^{q} \Delta M_{ij}$ represent the union of the map information discovered by Robot R_i between pth and qth sensing and mapping, not including the pth one. The local map is then derived from the raw map table. This local map is used in the bidding algorithm.

Table 1
Robot R_i's raw map table

R_1	R_2	R_3	...	R_n
ΔM_{11}	ΔM_{21}	ΔM_{31}	$\cdots$	ΔM_{n1}
ΔM_{12}	ΔM_{22}	ΔM_{32}	$\cdots$	ΔM_{n2}
ΔM_{13}	ΔM_{23}	ΔM_{33}	$\cdots$	ΔM_{n3}
.	.	.	$\cdots$	.
.	ΔM_{2i_2}	.	$\cdots$	.
ΔM_{1i_1}		.	$\cdots$	.
		.	$\cdots$	ΔM_{ni_n}
		ΔM_{3i_3}	$\cdots$	

Table 2
Robot R_i's last-sequence-number table

Robot	Last sequence number
R_1	i_1
R_2	i_2
R_3	i_3
$\cdots$	$\cdots$
R_n	i_n

Based on the above map storage scheme, a map update tracking scheme can be developed. Basically each robot maintains a last-sequence-number table, as shown in Table 2. This table records the largest, or latest sequence number of map information obtained from each discoverer. Using this table, it is possible for two robots to exchange map information based on the last sequence numbers that each robot has.

4.2. Map synchronization algorithm

When two robots R_i and R_j meet and a new communication link is established, they first check if the other robot is already within its own subnetwork. If yes, the two robots do not need to exchange map information since they already have a consistent map. If not, Robot R_i sends its last-sequence-number table to Robot R_j, and Robot R_j sends its last-sequence-number table to Robot R_i.

Based on R_j's last-sequence-number table, Robot R_i can calculate and send to Robot R_j the map information that R_j does not have, according to the following expression:

$$\Delta M_{i\to j} = \bigcup_{p=1}^{n} \Delta M_p(j_p, i_p]. \tag{3}$$

Table 3
Robot R_i's updated last-sequence-number table

Robot	Last sequence number
R_1	$\max(i_1, j_1)$
R_2	$\max(i_2, j_2)$
R_3	$\max(i_3, j_3)$
...	...
R_n	$\max(i_n, j_n)$

Similarly, Robot R_j should send the following map information to Robot R_i:

$$\Delta M_{j\to i} = \bigcup_{p=1}^{n} \Delta M_p(i_p, j_p]. \quad (4)$$

After the map information is exchanged, Robot R_i and Robot R_j should update each other's last-sequence-number table to reflect the exchange. Basically, both tables in Robot R_i and R_j are updated to Table 3.

The skeleton of the map synchronization algorithm on Robot R_i is as follows:

Robot R_i's Map-synchronization algorithm

(1) if (any robot R_j is reachable from this robot)

(2) if (R_j is already in the same subnetwork as this robot) return;

(3) else do steps (4)–(10)

(4) send last-sequence-number table to R_j;

(5) wait for last-sequence-number table from R_j;

(6) calculate the missing map information $\Delta M_{i\to j}$ for R_j;

(7) send the missing map information $\Delta M_{i\to j}$ to R_j;

(8) wait for missing map information $\Delta M_{j\to i}$ from R_j;

(9) update the last-sequence-number table;

(10) multi-cast the missing map information $\Delta M_{i\to j}$ from R_j to all the other robots in the same subnetwork.

5. Simulation results

The distributed coordination algorithm is simulated in *Java*. One example environment is a square of 40 by 40, with two obstacles in it. A number of robots are simulated by identical programs with each representing one robot and running on a separate computer within a local area network. The bidding time used in the coordination algorithm is $T_{\text{bid}} = 1.5$ s. For all the robots, the sensor range radius is 4 cell units ($r_s = 4$) and the communication range is 16 cell units ($r_c = 16$).

First, we investigate how the nearness measure affects the behavior of the robots by comparing the movement of the robots in two different cases: the first case does not consider the nearness measure and the second does. When no nearness measure is concerned, the weights used in Eq. (2) are: $\omega_1 = 1.0, \omega_2 = 1.0, \omega_3 = 0$. Fig. 9(a) and (b) show the local maps in robot 1 and robot 2 at certain time instant. It is clear that the distances between robots are large and some of the communication links are broken, which results in partitioned networks. As a result, robots in different subnetworks have different local maps. For example, robot 1 and robot 2 have different views of the local map at this moment, since they

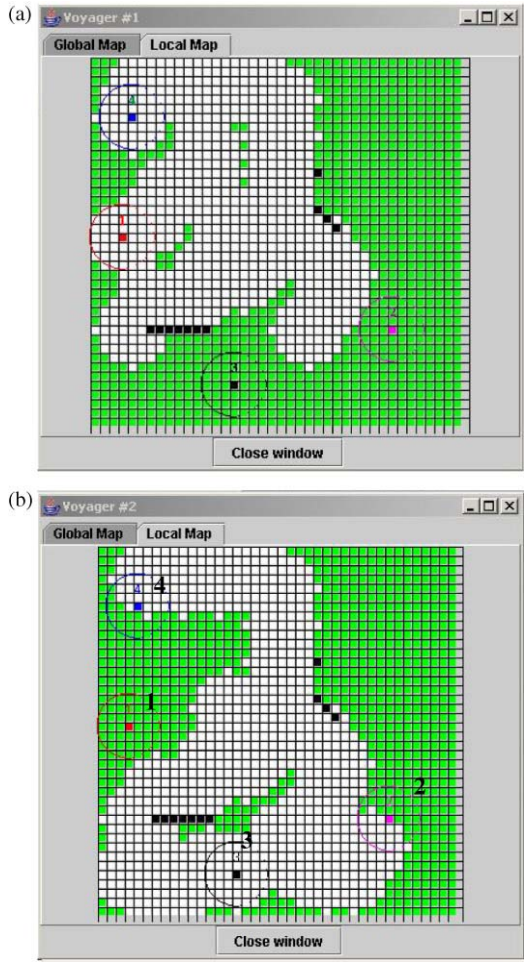


Fig. 9. No nearness measure is used and the robots tend to move apart quickly. Due to network partitioning, robot 1 and robot 2 have different local maps. (a) The local map of robot 1; (b) the local map of robot 2. The circles represent the sensing range and the circled dots are the robots.

are in different subnetworks. When the nearness measure is considered, the weights used in Eq. (2) are: $\omega_1 = 1.0, \omega_2 = 1.0, \omega_3 = 1.5$. Fig. 10(a) and (b) show the local maps of robot 1 and robot 2 at certain time instant. The four robots exhibit certain clustering behavior, which implies that the nearness measure is high for each robot. Due to the close distances, the four robots maintain a connected network and they share the same local map. More quantitatively, we measure the average of the diameters of the robot group (i.e., the maximum distances between any two robots) over the whole exploration process. Eight runs are conducted by varying the starting locations of the robots. Fig. 11 shows the average diameters in the two cases. This figure shows that, by including the nearness measure in the coordination algorithm, the robots tend to stay close to each other.

Second, how the bidding algorithm and nearness measure impact the exploration efficiency is investigated. We test three cases: (1) all the robots randomly pick frontier cells as next targets; (2) all the robots use distributed bidding algorithm but without nearness measure; (3) all the robots use a distributed bidding algorithm with nearness measure. The number of robots used in the testing ranges from 1 to 8. For each case

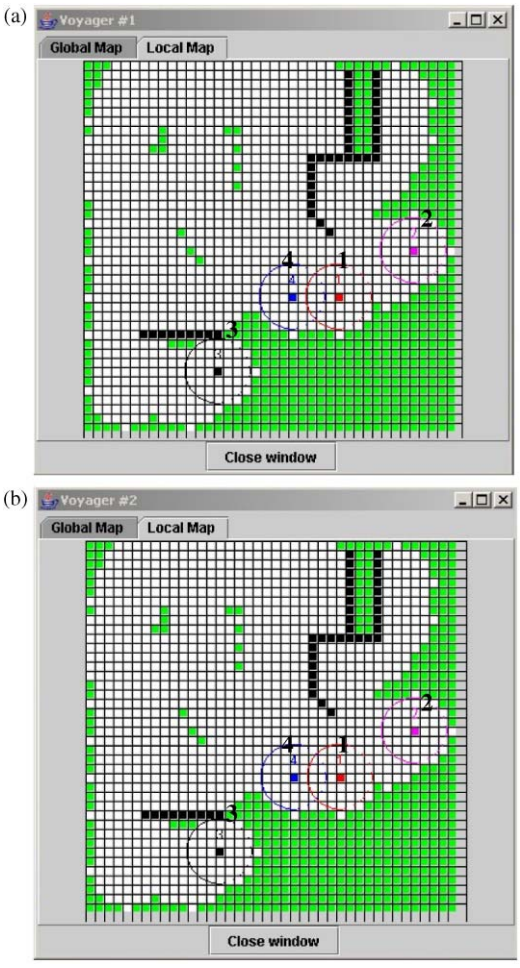


Fig. 10. The nearness measure is used and the robots tend to stay close to each other. All the robots have the same local map. (a) The local map of robot 1; (b) the local map of robot 2.

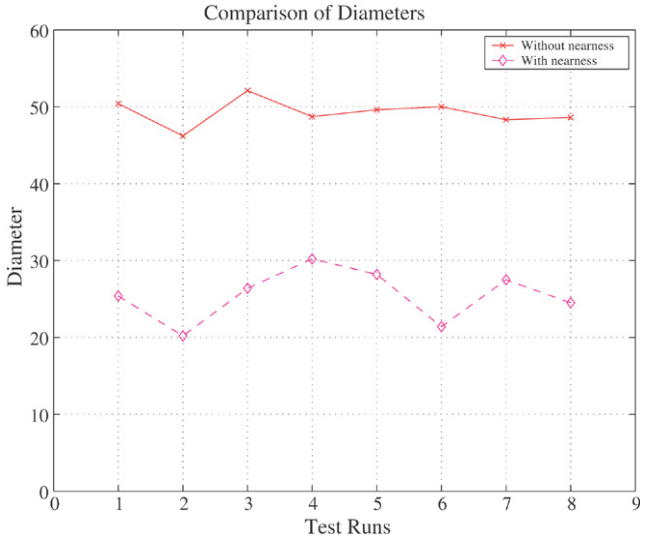


Fig. 11. The average diameters recorded in the two cases: with and without nearness measure. Eight runs are conducted by varying the starting locations of the robot.

and a fixed number of robots, six test runs are conducted and the total exploration times are recorded. Fig. 12 compares the

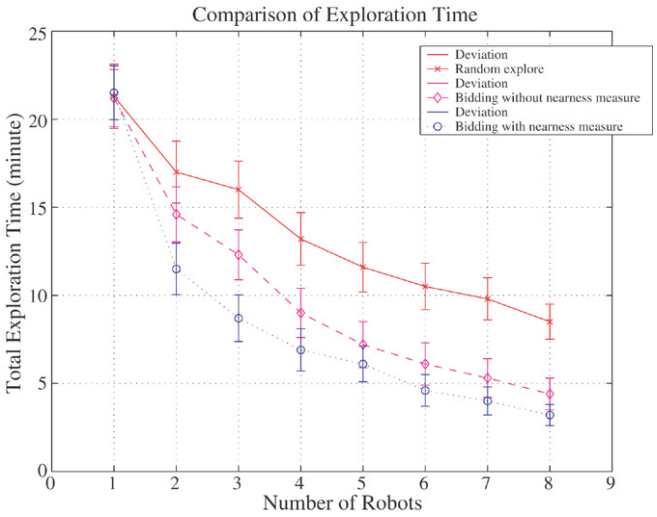


Fig. 12. The exploration time in three cases.

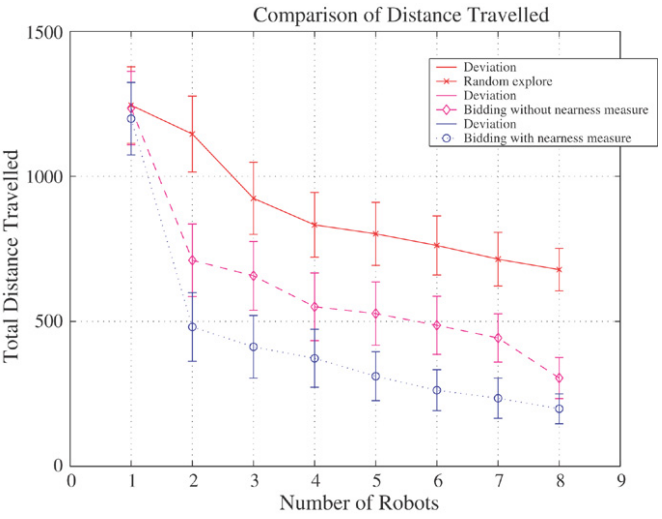


Fig. 13. The total distance traveled in three cases.

total exploration time needed for the three cases with regard to the number of robots. The thick vertical line segments show the deviation of the exploration time. This figure indicates that as the robot number increases, the total exploration time decreases. The bidding algorithm effectively reduces the time compared with a random exploration algorithm. When the nearness measure is considered in the algorithm, the total exploration time is further reduced. The reason is that by using the nearness measure, the robots tend to stay close to each other and maintain communication links among them, which leads to less repeated coverage and less exploration time. Fig. 13 shows the total distance traveled by all the robots in each case. Similarly, six test runs are conducted for each case and a fixed number of robots. The thick vertical line segments show the deviation of the total distance. It is obvious that similar saving in total distance traveled is achieved as the bidding algorithm is adopted and the nearness measure is used.

Third, we investigate the effectiveness of the map synchronization algorithm. For simplicity, we use only three robots. Fig. 14 shows the local maps of the three robots R_1, R_2

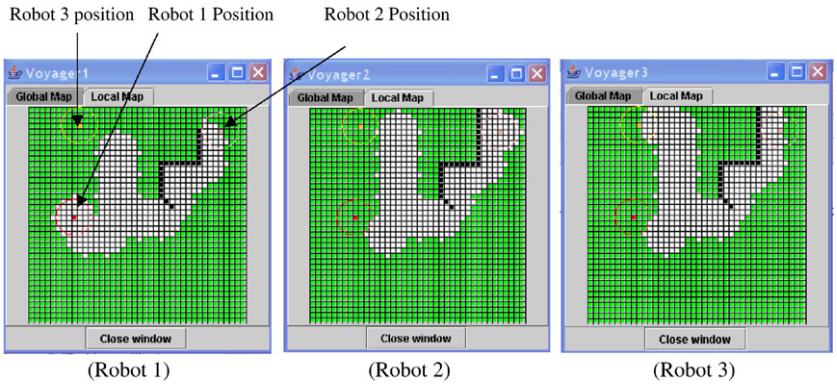


Fig. 14. The local maps one step before R_1 and R_3 can communicate with each other. Note that the map of R_1 is different from that of R_3.

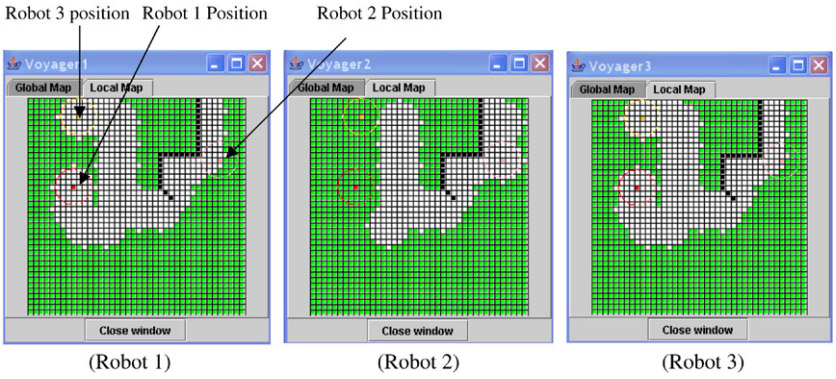


Fig. 15. The local maps right after R_1 and R_3 communicate with each other. Note that the map of R_1 is the same as that of R_3.

and R_3 one step before R_1 and R_3 establish a communication link. Obviously the local maps are inconsistent, or different, due to the network partition. Fig. 15 displays the local maps of robots R_1, R_2 and R_3 right after R_1 and R_3 synchronize their maps. It can be seen that R_1 and R_3 share the same map.

We also look into the amount of map data exchanged during the whole exploration. Five robots are used and the tests are conducted using the following weights: (1) $\omega_1 = 1.0, \omega_2 = 1.0, \omega_3 = 0$; (2) $\omega_1 = 1.0, \omega_2 = 1.0, \omega_3 = 1.0$; (3) $\omega_1 = 1.0, \omega_2 = 1.0, \omega_3 = 2.0$. For each case, by varying the starting time of each robot, we run the programs eight times and record the volumes of map data exchanged and compare with the volumes of map data that are supposed to be exchanged if the map update tracking scheme is not used. Fig. 16(a)–(c) show the amount of map data exchanged under case 1, 2, and 3 respectively. Table 4 gives the mean and the standard deviation of the data exchange volume for the three cases, based on the data collected in 20 runs for each case. From the above three figures (Fig. 16(a)–(c)) and Table 4, it can be clearly seen that in all three cases, the map synchronization algorithm effectively reduces the communication volumes. Comparing the three cases, it can also be noticed that as the nearness measure has more weight in the bid, the network tends to experience less network partitions, which leads to less map synchronization and less communication volume.

Fourth, we studied the impact of communication range upon the efficiency of the exploration. Five different values of the communication range are used: 4, 8, 12, 16, 20, and 24. We consider the exploration efficiency vs. the communication range in the following two cases: (1) all the robots use a distributed bidding algorithm but without nearness measure; (2) all the robots use a distributed bidding algorithm with nearness measure. Five robots are used in the tests. For each case and each communication range, six test runs are conducted. The means and deviations are calculated. Fig. 17(a) and (b) show the impact of the communication range upon the total exploration time and traveled distance respectively. From these two figures, we can see that for case 1, as the communication range increases, the exploration time and traveled distance decrease, because larger communication range leads to less network partition, which results in less repeated coverage. It can also be seen that the efficiency gain due to the communication range increase is more significant when the communication range is below 16. This result is consistent with the results provided in [12]. For case 2, we observe the similar efficiency gain as the communication range increases. However, the magnitude of the gain is not so significant compared to case 1. This is mainly because that in case 2, the robots are less likely to be

Table 4
The mean and standard deviation of the data exchange volume in bytes

Case	Mean		Stand deviation	
	Without tracking	With tracking	Without tracking	With tracking
i	23 034	4893	976	393
ii	11 855	2126	685	130
iii	7 075	812	367	82

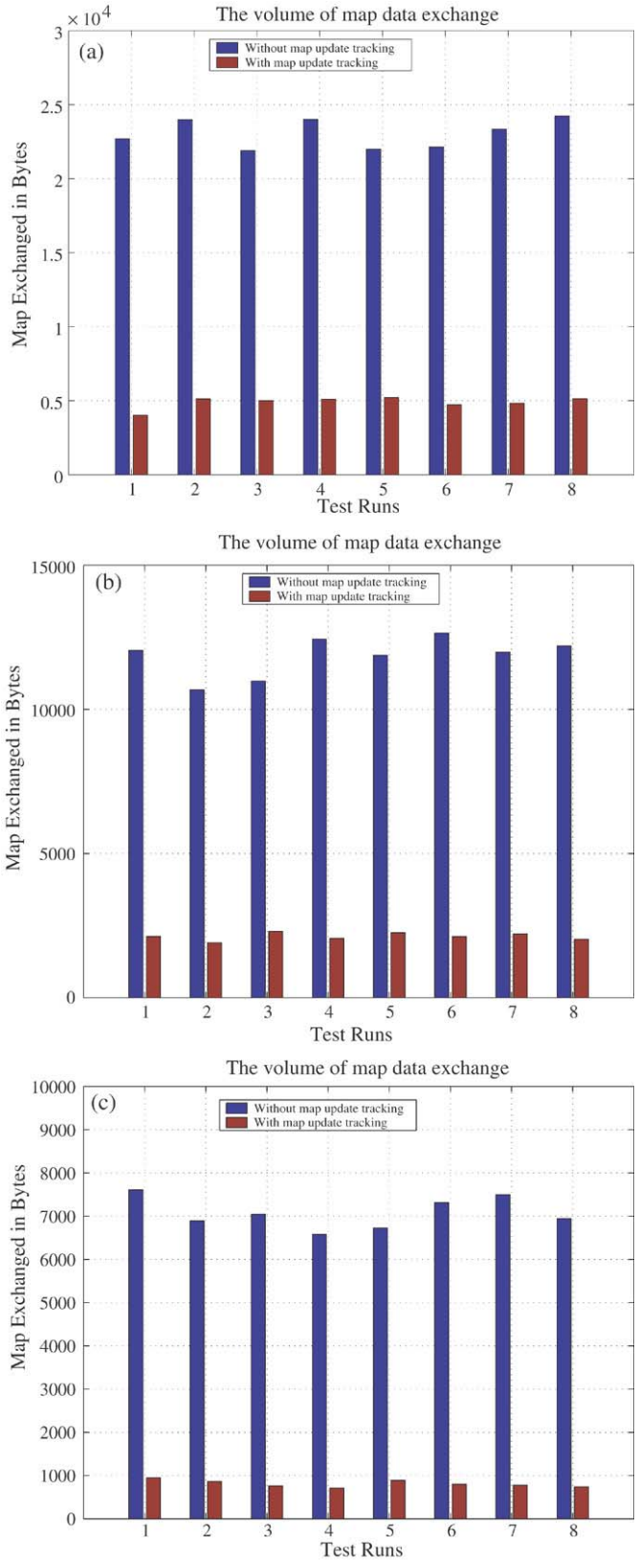


Fig. 16. (a) The amount of map data exchanged: case 1 ($\omega_3 = 0$). (b) The amount of map data exchanged: case 2 ($\omega_3 = 1.0$). (c) The amount of map data exchanged: case 3 ($\omega_3 = 2.0$).

partitioned. The change of communication range does not affect the connectivity much.

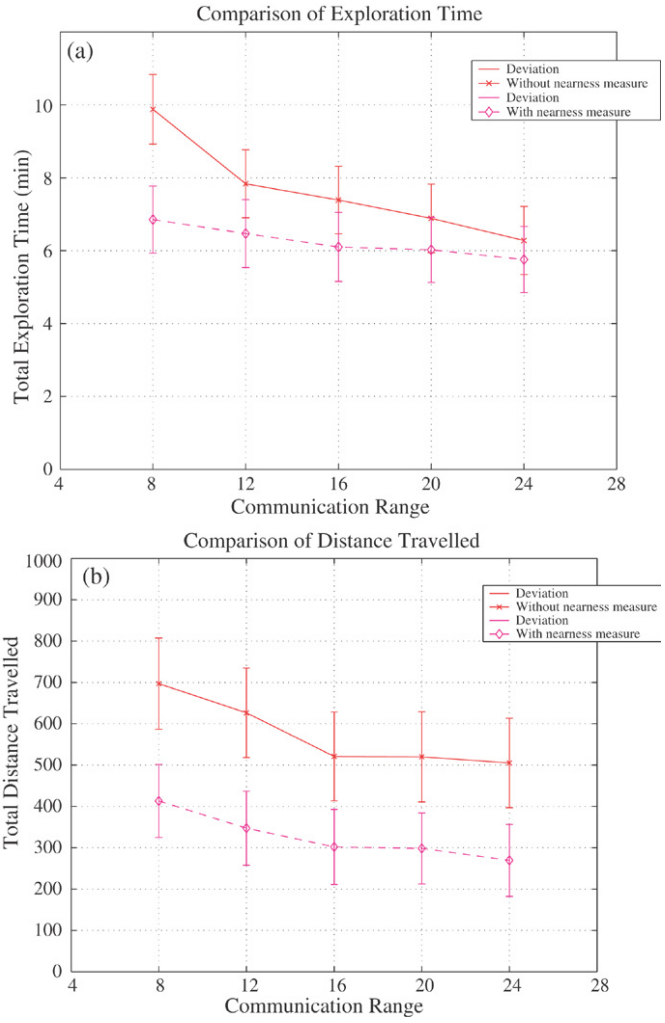


Fig. 17. (a) The total exploration time vs. the communication range. (b) The total distance traveled vs. the communication range.

6. Discussions and conclusions

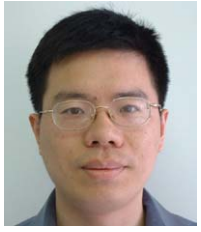
Due to the network partitioning, the bidding is limited within each subnetwork and the maps among different subnetworks are inconsistent. This will cause some problems, such as different robots may select the same cell as a target, or a robot may select a target cell that another robot already resides in. Hence, each robot should be equipped with the capability to avoid collision.

To summarize, this paper proposes a distributed bidding algorithm for multiple robots in exploration tasks and addresses the problem caused by the limited communication range. A new, totally distributed bidding algorithm is developed, which maximizes the net gain — a weighted combination of the information gain, the travelling distance and the nearness measure. This distributed bidding algorithm is efficient and simpler compared to existing multi-robot exploration algorithms. By introducing the nearness measure, robots tend to stay close to each other. From the simulation results, we observe that certain clustering behavior occurs during the exploration mission. The impact of the bidding algorithm and the nearness measure upon the exploration efficiency is investigated through multiple tests. The map synchronization mechanism proves

to save the communication volume significantly. We also investigated the impact of the communication range upon the efficiency. Our future work will implement the algorithms on several *Pioneer*^{TM} robots.

References

[1] Y. Cao, A.S. Fukunaga, A.B. Kahng, Cooperative mobile robotics: Antecedents and directions, Autonomous Robots (4) (1997).
[2] I.M. Rekleitis, G. Dudek, E.E. Milios, Multi-robot collaboration for robust exploration, Annals of Mathematics and Artificial Intelligence 31 (1–4) (2001) 7–40.
[3] R. Zlot, A. Stentz, M.B. Dias, S. Thayer, Multi-robot exploration controlled by a market economy, in: Proceedings of the 2002 IEEE International Conference on Robotics and Automation, 2002, pp. 3016–3023.
[4] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, S. Thrun, H. Younes, Coordination for multi-robot exploration and mapping, in: Proceedings of the National Conference on Artificial Intelligence, 2000.
[5] B.P. Gerkey, M.J. Mataric, Sold!: Auction methods for multirobot coordination, IEEE Transactions on Robotics and Automation 18 (5) (2002) 758–768.
[6] B. Yamauchi, Frontier-based exploration using multiple robots, in: Proceedings of Second International Conference on Autonomous Agents, 1998, pp. 47–53.
[7] K. Fujimura, K. Singh, Planning cooperative motion for distributed mobile agents, Journal of Robotics and Mechatronics (1996).
[8] N. Roy, G. Dudek, Collaborative robot exploration and rendezvous, Algorithms (2001).
[9] L.E. Parker, Current state of the art in distributed autonomous mobile robotics, in: G.B. Lynne, E. Parker, J. Barhen (Eds.), Distributed Autonomous Robotic System, vol. 4, October 2000, pp. 3–12.
[10] R. Grabowski, L. Navarro-Serment, C. Paredis, P. Khosla, Heterogeneous teams of modular robots for mapping and exploration, Journal of Autonomous Robots 8 (3) (2000) 293–308.
[11] D. Jung, A. Zelinksy, An architecture for distributed cooperative planning in a behaviour-based multi-robot system, Journal of Robotics & Autonomous Systems 26 (2–3) (1999) 149–174.
[12] W. Burgard, M. Moors, C. Stachniss, F. Schneider, Coordinated multi-robot exploration, IEEE Transactions on Robotics 21 (3) (2005) 376–378.
[13] E. Dijkstra, A note on two problems in connection with graphs, Numerical Mathematics 1 (1959) 269–271.



Weihua Sheng received his Ph.D. degree in Electrical and Computer Engineering from Michigan State University in May 2002. He obtained his M.S. and B.S. degrees in Electrical Engineering from Zhejiang University, China in 1997 and 1994, respectively. Currently he is an assistant professor in the School of Electrical and Computer Engineering at Oklahoma State University. He has been involved in research projects funded by Ford Motor Company and NSF. His research work has resulted in one patent in the US and more than 40 papers in major journals and international conferences in robotics and automation. His current research interests include distributed robotic systems, mobile wireless sensor networks, process planning for advanced manufacturing technologies and embedded computing. He is a member of IEEE Robotics and Automation Society, IEEE Computer Society and ACM.

Qingyan Yang (kyang@iteris.com) received his B.S. and M.S degree in electrical engineering in 1989 and 1992 from Dalian Maritime University, PR China, and a M.S. degree in Civil Engineering (Intelligent Transportation System) in 1999 from Michigan State University, East Lansing, Michigan. He is currently a senior software engineer at Iteris Inc. His research interests include distributed computing, cooperative driving and Vehicle Infrastructure Integration (VII). He is member of IEEE.

Jindong Tan is an assistant professor in the Department of Electrical and Computer Engineering of Michigan Technological University. He received his Ph.D. degree from Michigan State University, East Lansing, MI, in 2002, in Electrical and Computer Engineering. Dr. Tan has been the PI of a NSF funded research project on sensor/actuator networks, the PI of a project funded by Michigan Space Grant Consortium, and the PI of a project funded by Michigan Tech Research Excellence Fund. Dr. Tan's research foci are hybrid sensor/actuator networks and body area sensor networks. His current efforts in sensor/actuator networks focus on the coordination and navigation algorithms for a complex network of mobile and static sensors. Dr. Tan's research foci in biosensor networks include the development of energy efficient communication and networking techniques for embedded body area sensor networks and embedded bioinformatics for cardio-vascular disease.

Ning Xi received his D.Sc. degree in Systems Science and Mathematics from Washington University in St. Louis, Missouri in December, 1993. He received his M.S. degree in computer science from Northeastern University, Boston, Massachusetts, and B.S. degree in electrical engineering from Beijing University of Aeronautics and Astronautics. Currently, he is John D. Ryder Professor of Electrical and Computer Engineering in the Department of Electrical and Computer Engineering at Michigan State University. Dr. Xi received the Best Paper Award in IEEE/RSJ International Conference on Intelligent Robots and Systems in August, 1995. He also received the Best Paper Award in the 1998 Japan–USA Symposium on Flexible Automation. Dr. Xi was awarded the first Early Academic Career Award by the IEEE Robotics and Automation Society in May, 1999. In addition, he is also a recipient of National Science Foundation CAREER Award. Currently, his research interests include robotics, manufacturing automation, micro/nano systems, and intelligent control and systems.