International Journal of Advanced Research in Computer Networking, Wireless and Mobile Communications
Volume-1: Issue-3 JUNE 2013

A Static Bluetooth Process Continuously Depend on a Multi-Robot Sequential Analysis

A. Jasmine Xavier[1], R.Shantha Selva Kumari[2]

Assistant professor ECE Department, Jayaraj Annapackiam CSI college of Engineering, Anna University Chennai[1], Professor ECE Department, Mepco Schlenk Engineering College, Anna University Chennai[2]

jtony2012@gmail.com[1], r_ssk@gmail.com[2]

***ABSTRACT*—**

In the first part of the present paper, a multi-robot system can be highly beneficial for exploration, which is a core robotics task. We know that this exploration task is best performed when using a multi-robot system. We present an algorithm for multi-robot exploration of an unidentified environment, pleasing into account the communication constraints between the robots. A novel communication scheme of an autonomous robot team via Bluetooth radio is investigate. In the presented solution, an autonomous unit is equipped with two independent Bluetooth radios and so a relatively fast communication is possible in the team in a static (i.e. no ad-hoc) networking topology. An autonomous robot is a machine able to extract information from its environment and use knowledge about its world to move safely in a meaningful and purposive manner. The performance of such a network was tested by implementing a linear graph topology by NXT robots. It was establish that the reliability and the speed of such a communication scheme are satisfactory and give rise to applications in a robot team manage task. In the second part of the paper an area exploration method is presented based on the static linear communication system above. The method was tested by computer simulations for various obstacle configurations and density. It was establish that the proposed method performs better than the chosen reference methods in the case of zero or low obstacle density and when high (75% or 100%) exploration ratio is necessary. With a simple verification, we have shown that the proposed (fixed chain-like team) exploration method is optimal in the obstacle-free case under the constraint of the connectivity with the base station.

Keywords—linear graph topology,dual-radio scatternet,piconets, scatternet

I. INTRODUCTION

A. Motivation

Mobile Robotics is an important research area in the field of robotics. Lot of research has been done in the context of multi-robot exploration. Some of the major real world applications include explore and release, like military measures, lunar and planetary exploration, deep ocean exploration, underground mining etc. To achieve efficiency and reliability coordination among multiple robots is an important. For the purpose of improving competence, it is required to minimize the overall time and distance covered by the robot. In the past, most of the strategies or approaches have focused on coordination issues, efficiency of the metric or investigation, without bringing any communication constriction between the robots. Later on, algorithms have also taken into consideration the communication constraint during multi-robot exploration using a fixed support station, where each robot tend to be in communication with a fixed robot directly or indirectly. In this paper, we Extend the frontier based exploration approach where the robots move as a robot pack, and can always communicate with each other.

The advantage of this algorithm over that of fixed base station is that the robots can explore the whole unknown map as a pack, and are not restricted in their approach because of the communication constraint with the fixed base position. It also sense capabilities of the robots have been taken into account, through which the robots are able to explore faster compared to point Wise frontier basedapproach. This paper builds on the work done in Multi Robotic Exploration with Communication Requirement to a Fixed Base Station and A Two Phase Recursive Tree Propagation based Multi-Robotic Exploration Framework with Fixed Base Station Constraint, which present an approach based on construction of a tree network for multi-robot exploration while maintaining communication constraint throughout with a fixed base station.

Due to the considerable development in the field of team intelligence and multi-robot cooperation there are a wide variety ofproblems that can be solved efficiently by a group of autonomous mobile robots. A typical and intensively investigated problem is the exploration of an unknown area with the purpose of either mapping or finding certain aim object(s). The best solution strategy of such a task is mostly determined by the conditions imposed by the environment and the abilities of the robot team.

Almost all of such exploration methods are based on the concept of “frontier”, which is the boundary between explored and unexplored area [1, 2]. The basic idea is that the robots should be directed to or kept on this boundary, while the cost of the team-moves is minimal and the information gain (explored area) is maximal. In order to achieve the published algorithms apply a utility function, which is increasing in information gain and decreasing in the cost or exploration time [1-10].

B. Previous Work

Simmons et al. [3] applied a central method, where a support station considered the best team-move (with the peak utility) and controlled the robots. Here the thought that the robots should stay apart from each other was built into the usefulness function. This diffusion is required in order to maximize the long-term information gain in such models. Burgard et al. [4] propose a similar centralized method; however, in this method they also consider the case of limited communication range. If the originally continuous communication cluster of the robot team breaks up into more, smaller clusters so that the robots in the different clusters cannot communicate with each other, then each cluster continues the exploration algorithm independently. In the worst case, all of the robots are isolated and allocates the exploration tasks only for him, so the method crosses over to a decentralized system, though the task allocation is far from the optimal.

In their paper Sheng et al. [5] introduces a solution to the problem of separated communication clusters: the robots perform an individual exploration movement but they are compelled to get in (communication) contact with their fellows regularly so that the task allocation procedure can work regardingthe entire team. Thus, for a considerable part of the working time the team members are outside of the communication range of the other robots. Vazquez and Malcolm [6] proposed a decentralized method, where the robots individually perform their algorithms responsible for region examination, communication maintaining and collision avoiding. The prevailing activity is chosen according to the momentary situation: if the communication and collision avoiding is ensured then the exploration behaviour ispreferred. If an important communication link is close to a failure (because of the growing distance of the partners), then the connection maintainingbehaviour is preferred; and the same is true for the danger of collision. With this scheme, the algorithm guarantees the communication connectivity at any time meanwhile the exploration, and the exploration proceeds under the constraint of the connectivity and collision avoiding.

In a recent paper, Y. Pei et al. [7] establishes a method, where cost of the group moves is given by the total migration time between the two consecutive team positions. Their algorithm, being a central one, ensures the connectivity with a base station at the end of each team-move; however, meanwhile the migration phase the connectivity is not guaranteed.The task distribution is a critical part of these methods. In this phase, the fundamental source or the community of the robots decides the (believed) optimal target frontier position for each robot. In a decentralized system, this is based on the bids from each robot for the target positions. This bidding scheme is quite difficult to organize when there is no continuous communication between the whole team (see [5]). In numerous papers (e.g. [8-10]) this bidding is organized as a market auction, supposing that the communication under the bidding is continuous in the whole team. The task allocation as a scheduling problem applied also in complex industrial manufacture process. Their result taken into consideration in the case of heterogeneous robot team, where the abilities of the robots are different for the appointed tasks (like exploration for example).

Most of the cited methods take into consideration the problem of collision avoidance with the kin robots. On the level of the algorithm, it is done simply by adding a term to the utility function, that punishes the configurations in which the robots come too close to each other. In addition to this, it is necessary to implement a local, independently working collision avoiding behaviour on the robots that takes over if there is danger of collision [6]. In the present paper, the collision avoiding with robots is treated first from thepart of the proposed algorithm, and then from the part of the local method in a specific robot-simulator test. In our solution, we used only a few sensors, and no on-board or central camera. Much more advanced control and collision avoiding methods can be achieved with the help of an on-board camera. The application of image processing methods in robotic intelligence makes it possible to recognize, identify and locate the fellow robots and landmark objects [2, 1], or with a multiple camera system even the body poses of a humanoid can be recognized [4].

In the case when the robot team works under the connectivity constraint of the network, the exploration algorithm has a double role: it has to find movements that are optimal from theexploration point of view, and, at the same time, it has to ensure thenetwork connectivity. There are applications [5-7] where this latter role is trivial, since the supposed radio coverage of a robot is bigger than the working area (i.e. the area to be explored). In these cases the robot team is supposed to use a wireless shared media communication system and the Media Access Control has the most crucial task. Numerous other works, however, investigate multi-robot exploration problems when the radio coverage of an individual robot is much smaller than the area to be explored. Generally, in these problems the working plane is divided into cells and a step (or movement action) of a robot is determined by the source and the destination cells. This (coarser or finer) discretization of the explored area makes easier the mathematical characterization of the problem. In these cases, the exploration algorithms suppose that there is a Mobile Ad-hoc Networking (MANET) system providing communication channels between any two robots of the team. X. Cui et al. [8] applied a so called gradual expansion algorithm to keep the robots in communication contact while completing their task.

The basic idea of this algorithm is that only one robot moves in one time-step and the destination cell of this move must be a neighbour of a cell occupied by a kin robot. In the step planning procedure, they applied a fuzzy decision system to find the best discovery move under the constraint above. W. Sheng et al. [5] and later K. Cheng and P. Dasgupta [9] treated similar problems, but here the communication connectivity of the team was not enforced at each time-step, though the communication (if it was possible) played important role in the applied algorithm. So in this scene the communication network could be broken up into smaller clusters or individual robots, but on this account the team could move around more freely. M. N. Rooker and A. Birk [2] used an exploration scheme where the communication network had to be continuous at any moment; moreover, they investigated the important case when the team had to be in communication contact with a fixed base station, so the exploration range was limited in space. Here the whole robot team moved in one time-step, and they used a utility function method to find the best collective move. The collective moves that would break-up of the communication network obtained such amount of negative utility points so that the team never chose these moves. A theoretical solution to a problem of maintaining a multi-hop wireless communication chain between two mobile targets by mobile robots. However, their work did not deal with area exploration, since the positions of the targets were known during the simulation task.

Nevertheless, the most of the published schemes assume the wireless communication granted and consider the specifications of the wireless system beyond the scope of their interest. In the cases when a specific wireless system is applied (either in computer simulation or in reality) the mobile robots at hand are equipped with Wi-Fi (IEEE 802.11), Zig-Bee (IEEE 802.15.4) or Bluetooth (IEEE 802.15.1) radio system, since they render cheap and yet satisfactory solutions.

The Bluetooth radio system, which is the applied technology in the present work, is a cheap solution and compared to its relatively high data rate it is economic on power. Due to these advantages, the Bluetooth technology is the most commonly used on small mobile devices, and therefore it, is a good candidate in the case of mobile robots. More detailed pros and cons on Bluetooth communication in mobile robotics can be read in [2] or [3]. The most serious limitation of a Bluetooth network is that it is not scalable, since a Bluetooth piconet can consists of a master and at most seven slaves [4]. In order to overcome this limitation the so-called Bluetooth scatternet is invented shortly after the appearance of the original Bluetooth standard [5, 6]. The basic idea of forming a scatternet is that a slavedisconnects from its master and becomes only a passive member (park or hold mode) in its original piconet, and then asks for and gets admission into another piconet as a slave or a master. Thus, there can be communication between the two piconets through this, so called, bridge unit: if there is a packet or a message directed to the other piconet the bridge takes it, changes piconet and passes the packet towards its destination. However, each of these bridging actions causes some delay and acts as a bottleneck [7]. In addition to this, the position of the bridge unit is more restricted than that of the others, since it must be in the radio coverage in both piconets. Due to this shortcomings this bridge basedscatternet, although arbitrary scalable from theoretical point of view, is limited to not too big networks and low data rates in practice.

A novel and simple solution that employed two independent Bluetooth radios in single autonomous host to form a large scale wireless sensor network (dual-radio scatternet). In this scheme, the two Bluetooth radios are parts of two different piconets so that the host passes the information between its two radios. It is easy to see that this solution is free from the limitations of the former bridge based scatternet, although it has higher cost. The main advantage of this scheme over the usual scatternet is that a node does not have to disconnect from one neighbor and reconnect to another one regularly, in order to ensure the connectivity of the entire network. This switching from one neighbor to the other consists of a normal disconnection and a connection process, which altogether takes between 3 and 6 seconds [3, 1]. Under this period the network is not connected, though the packet finally reaches its destination. Since there are a number of bridge nodes in a big (conventional) scatternet, in almost any moment there is an unconnected part of the scatternet. The dual-radio based scheme, however, ensures that every nodes in the network is reachable continuously. (For further details on the advantage of the dual-radio based scatternet see [8])

In the present paper, we employed the dual-radio scatternet scheme described above to form communication network of mobile robots. In order to test our scheme in reality we used the microcontroller based NXT robot assembling set produced by LEGO. This set is based on Bluetooth communication and planned mostly for educational purposes, however, there are also numerous research applications using NXT. The most remarkable advantage of the dual radio scatternet is that two communicating robots need not to break their Bluetooth connection while they are in the vicinity of each others (in the case of Bluetooth I technology this means a distance at most ten meters). This means that in order to exploit the advantages of the communication system above the robots should not change their neighbors in the communication network while the completing the task, that is, the networking should work as not a MANET but a static network. In one hand, this has restrictive consequences for the applied exploring algorithm, in other hand, we can exploit the advantages of the dual-radio based system, and the continuously connected network enhances the reliability and robustness of the robot system. First, a reality check of the dual-radio based communication system is performed with a simple scenario that consists of a linear communication chain formed by three to seven robots. An exploration algorithm for this fixed linear communication topology is also proposed.

In the next chapter the technical details of the invented NXT scatternet system are given. In Section 3, the reality check of the linear communication chain consisting of three to seven robots is introduced. Section 4 details the proposed exploration algorithm, and in Section 5 a simple proof is given regarding the optimality of the proposed algorithm in an obstacle-free area. Section 6 introduces the results of the tests concerning the proposed and reference methods. Finally, we conclude in the last section.

II. METHODOLOGY

Here we describe our algorithm to explore the given map, consisting of obstacles of any shape.

A. *Problem Description*

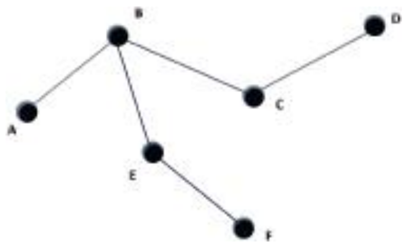


Fig. 1 Connection Graph

Given a map of m x n grid, consisting of obstacles, we need to explore the whole map with the help of N Robots. Each robot has a sensing range Rs, which helps it to detect obstacles nearby without any physical contact. The robots are initially placed such that they are within communication range Rc of at least one of the robots in the robot pack. Here, we assume that all the robots share the same environmental map. Robots have no previous information about the map to be explored. They know only the information available to them through their sensors or through their communication with other robots. Each point in our grid may have one of the following states:

Explored: A point in our map is considered to be explored if it has been seen by a robot.

Unexplored: A point is unexplored, if it has not yet been explored by any of the robots.

Frontier: A point is called frontier if it is at the boundary between the explored and the unexplored area.

Obstacle: If the point is occupied by an obstacle. Robots cannot visit these points in the map.

For simulations we will use a map of 512 x 512 grid cells. It is represented in the form of 2-D environment for the purpose of simulating.

B. *Definitions*

1) Visibility Gain: Visibility gain V is defined as the information gain of the robot, when it takes a complete 360^0 scan of areas around it. It is the fraction of unexplored areas around a point, which are inside the sensing radius. We use the concept of ray tracing to find the visibility gain.

2) Metric Gain: Metric gain M at a point (x,y) is defined as the ratio of the Visibility Gain to the distance dx;y, a robot has to travel from its current position to that point.

3) Connected: A robot A is said to be connected to another robot B, if the euclidean distance between the two robots is less than or equal to the communication range Rc.

d) Connection Graph: A connection graph refers to the topology or the layout pattern of the connected robots in a map shown in Connection Graph. An edge is said to exist between two robots in a connection graph if they are connected.

C. *RELATED WORK*

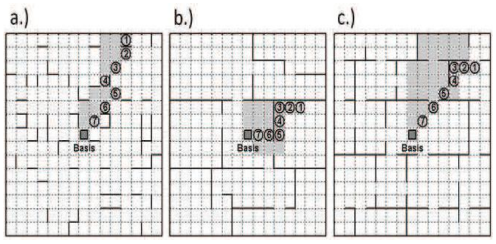
In this section, we give a quantitative test of the proposed recursive chain algorithm and compare its performance with the algorithm. We will call their method as "reference method" (or "reference algorithm") in the followings. As it was mentioned in the introduction their algorithm used a utility function in order to choose the best team-move from a relatively large pool of possible (or impossible) team-moves, which was generated randomly. They used different pool-populations from 50 up to 1000, and a test area with rectilinear wall-like obstacles that divided the area into large rooms (compared to the cell size). In order to give a correct quantitative comparison we implemented the reference algorithm with the pool population of 1000, and tested both in the same test areas.

One of the main differences between the two algorithms is that the reference algorithm uses a utility function that takes into consideration the utilities of the individual robot-steps with equal weight, and the roles of the robots are equal, so we can say that the exploration task and the utility function is decentralized. Besides, it assumes a continuously working MANET system while the proposed algorithm does not. Another considerable difference is that the team step planning method of the reference method involves a high computational load, while our method requires much less computation. This is because the reference method computes the utility for 1000 team configurations, while our method does a similar computation only for the leader robot. The computational load has importance here, because these algorithms based on central computation. If we have not a high-performance computer in the base-station, then one of the robots (or all of them) must carry on the computation. If all of the robots carry on the computation, then the time and energy of broadcasting the computed team-move can be saved.

Besides, we applied another reference algorithm, too. This is a special case of Rooker's algorithm, since its method is the same, basically, but here the best utility team-move is chosen from the pool of all of the possible team-moves. This pool contains 4^{N-1} team-configurations, which is $4^7 \approx 16000$ in the case of the present test with N=8. Let us call this a "Full search" algorithm. This algorithm provides the best possible team-move with the given utility function (see [2]). Actually, this method has only a theoretical relevance, because the calculation of the team moves includes the evaluation of the utility function of about 16000 team configurations, which is 16 times longer than the simple reference algorithm above.

D. *THE SIMULATION AREAS*

From the proposed algorithm it follows that the maximal size of the area can be explored is $(2N+1)^2$, since the robots are forced to be in the safety environment (E_S) of each other.



This restriction is applied for all of the tested algorithms.

Fig. 2 (a), (b), (c)

Two kinds of obstacle configuration were used in the simulation areas. One configuration type contained randomly distributed obstacles with obstacle density ρ, that is, any cell-dividing wall was chosen to be an obstacle with probability ρ without any correlation between the obstacles. The other configuration type was similar to that of the test area used by Rooker and Birk, which was a building- like environment with smaller or larger rooms, separated by rectilinear walls and opened doors. Here we constructed two different such a building-like obstacle configurations, which can be seen in Figure 2/b and 2/c. As a comparison, a random obstacle configuration of the first type with obstacle density ρ=0.15 is also shown in Figure 2/a. All simulation areas were chosen to be the same size of *(2N+1)* by *(2N+1)* cells square bordered by obstacles. Therefore, both algorithms could reach all of the cells (unless there were areas isolated by random obstacles).

E. *RESULTS AND DISCUSSION*

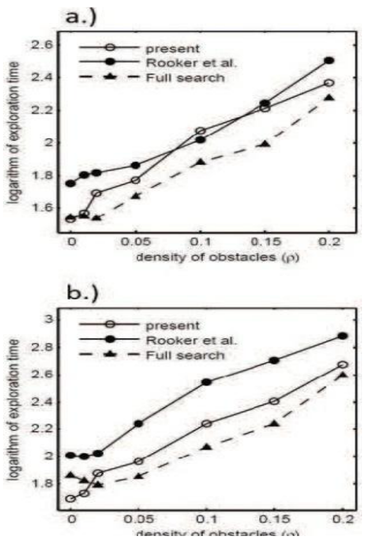


Fig. 3 (a), (b)

The algorithms were tested by computer simulation using MATLAB for parameter *N*=8, which corresponds 7 exploring robots. Therefore, the simulation area contained 225 cells to be explored. In the case of the random obstacle configuration type, the value of ρ was chosen to be 0, 0.01, 0.02, 0.05,0.1, 0.15 and 0.2. The case ρ=0 corresponds to the obstacle-free configuration. For each non-zero values of ρ five different obstacle configurations were generated. The tests were performed with the limiting parameter *MAX_MOVES*=400, which corresponds to the average measured value in the case of the NXT Tribot. With the tests, we measured the exploration times needed to explore 50% and 75% of the total 225 cells, which means 112 and 169 cells. The 100% exploration cannot be required, because a certain fraction of the 225 cells (0-20%) is not reachable under the constraint of connectivity with the base station. This is due to the randomly positioned obstacles. The exploration times were measured in time units (time-steps) we defined in section 5. Then the mean-value of the exploration times that belong to the same exploration ratio and obstacle density was calculated and graphed as a function of ρ. The results of these tests are shown in Figure 2/a (50% exploration) and 2/b (75% exploration).

It can be seen that in the case of 50% exploration, the proposed algorithm performs definitely better than the reference algorithm for an obstacle-free area or low obstacle density, and for higher obstacle densities the performances of the two methods are similar. The comparison with the Full search method is interesting too: at zero and 0.01 densities the two methods produces almost the same exploration times and for higher densities the Full search method performs better. In Section 5, we have shown that the proposed algorithm is optimal in the present conditions for the obstacle-free case (ρ=0). Thus, the Full search method seems to be also optimal in the obstacle free case when only half of the cells must be explored. The situation is different when the algorithms have to accomplish 75%. In this case, for low densities (zero and 0.01) the proposed algorithm performs better than the other two (being the optimal one), and beyond ρ=0.01 the Full search method achieves the best results. However, in the 75% case the proposed method performs definitely better than the reference method for all examined obstacle densities.

In an obstacle-free area, the robot team can explore all of the 225 cells, so we can measure the exploration times for the case of 100% exploration ratio. Table I shows these measured times in the first two data column. The advantage of the proposed method is the most pronounced in this case. Its exploration time (62 time units) is about half of that of the Full search method and one sixth of that of the reference method. This exploration time corresponds to the minimal theoretical value of $9(N-1)-1$ with N=8, which was given in section 5. The table also shows the success rates of the explorations in percents in the cases of the building-like obstacle configurations. In these simulations all three algorithms stopped before the 100% exploration success, because one of the robots reached the MAX_MOVES=400 value (see Eq. (5)). In this case, we can compare the success rates done before the algorithms stopped. The proposed and the Full search algorithms have similar success rates (about 70% and 80% in the cases of the two configurations), in the first case the former is better and in the second case the latter. However, they are much better in this measurement than the reference algorithm having only 25% and 43% success rates.

Since the proposed algorithm is optimal in the obstacle-free case, it is not surprising that it achieves the best exploration time value in the case of zero or low obstacle density. However, the question arises that why does it perform better than the reference method also for non-zero obstacle densities when 75% or higher success ratio is required, considering that the reference method is not restricted by the fixed chain-like topology as the proposed method. The answer can be found in the decentralized utility system of the exploration strategy of the reference method. This algorithm, similarly to other frontier and utility based algorithms [2-4, 6], determines the utility functions regarding the individual robots and takes them into consideration with equal weight. When there is a connectivity constraint with each other and the base station, then the robots, following their own utility-based exploration purposes, often obstruct each other, so the whole team moves with difficulty to an unexplored area. Consider, for example, the following situation: two robots, R_i and R_j, are at the frontier, but on the opposite sides of the base station as far from each other as permitted by the connectivity constraint, and the other robots are between them ensuring the connectivity. In this situation the team should move definitely into one direction, towards R_i or R_j, but the utility is approximately equal for both direction, therefore the team is in a deadlock situation for a while caused by the decentralized utility system. In the reference method there is a random factor, which helps the team out of this deadlock sooner or later. This effect is more pronounced if high exploration success is required (75% or beyond it), because the algorithm must reach also cells far from the base station. The proposed method is free from this obstructive effect, because there is an appointed leader, R_N, and the exploration utility of this robot overrides that of the others. Thus, the leader determines the exploration strategy alone; the other robots are responsible for ensuring the connectivity in the first place. It seems that this kind of task division is useful in the exploration strategy.

F. *PROBLEMDEFINITONAND ASSUMPTIONS*

The tests introduced above characterize the effectiveness proposed algorithm regarding the number of exploration steps versus the explored area. However, the feasibility of the algorithm is also an important question, and we treated this at the end of the previous section in a rather theoretical way. Besides, we tested the proposed method by a simulation implemented in the Microsoft Robotic Developer Studio (version 2010), which provides a robot-simulator environment and a realistic package for the Lego NXT Tribot as well [9]. The seven simulated NXT Tribots (the eighth robot is the base station) were three wheeled differential-drive entities with two (0-9V) DC motors and rotation sensors on the motor shaft by which the motor rotation can be controlled with the accuracy of 1°. The robots were equipped by a front and a rear touch sensor and a compass sensor. This equipment satisfies the requirements given in the last subsection in Section 4. The applied travelling and turning speeds were approximately 30 cm/s and 180°/s, respectively. At these speed values, the average accuracy of the navigation was 2.0E-3, which can be really achieved by selected hardware elements [3] (or applying calibration). The physical dimensions of a simulated robot are 22×18×14 cm, its weight is approximately 450 g.

We tested the robot team in two environments: one obstacle-free and one with rectangular obstacles (see Figure 4/b). The area with obstacles was bounded by four rectangular walls, so they formed a 32 m long and 32 m wide room. The robots divided their common map of area into 4 by 4 meters square cells, and navigated themselves in this coordinate system by odometry and compass sensor. Due to the communication phase at the end of each team-move, all of the robots were aware of the positions of the other robots (with the navigational error), and the discovered obstacles. The specific purpose of the robots was simply to visit all of the cells that can be reached by the robots. This means that a cell was considered explored if a robot entered into it and occupied the center of it.\

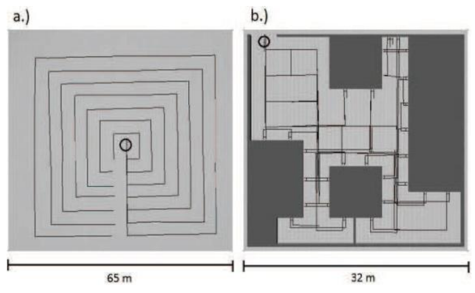


Fig. 4(a), (b)

The obstacle discovering and the kin robot collision avoiding was solved by the same simple method: if the front touch sensor of a robot sensed contact, then the robot moved backward 30 cm and tried to avoid that object on a rectangular path. If it could avoid the object in such a way, then it was a kin robot, else it was considered an extended obstacle. Thus, two robots could come into physical contact with each other, which slightly deflected the robots from their original directions. Because of this, at the beginning of every move cycle the directions of the robots were corrected with the help of the compass sensor. In the previous section, we discussed that two robots can occupy the same cell in the same step-cycle only if one of the two robots encountered an extended obstacle on its way out of the cell, and no three or more robots can occupy the same cell. This makes the implementation of the collision avoiding much easier. Note that we used only odometry navigation (with compass) and touch sensors, which is a simple solution, really. Figure 4 shows the trajectories of the seven robots in the two explored areas. These trajectories on the obstacle-free area form a regular pattern with concentric squares, which was discussed in the previous section. In the other area, one can see clearly the two contact points with the obstacles, which is resulted from the collision avoiding method.

III. AREA EXPLORATION ALGORITHM

A. *THE CELL-GRID AND THE RADIO COVERAGE*

The area to be explored is divided into non-overlapping equally sized square-shaped grid cells as it is usual in such problems. In our case, the cell size depends on the Bluetooth communication range. The specific task of the team is to visit all of the cells in the area and discover the possible obstacles. A practical application of this task, for example, is the case when the team has to find the locations of several objects or connect to a freshly discovered Bluetooth equipped host (e.g. cell phone) in order to establish a wireless multi-hop communication chain between the host and the base station. The onlyrestriction is that the robots have to keep up the linear fixed-topology communication chain between the base station and the robot farthest from the base station (measured along the chain).

In such tasks one of the most important parameter is the radius of the radio coverage of a robot. We denote this by r_{cov} and suppose that the radio contact of two robots within the coverage of each other is guaranteed. In several papers, however, more additional environments are defined beyond r_{cov} [6, 21, 33]. One of these is a so called “precaution” or “safety” environment (here denoted by r_{safe}), the radius of which is smaller than r_{cov} . The basic idea in this is that the algorithm tries to keep each robot in the safety environment of its neighbour’s, which greatly increases the robustness of the connectivity of the team. In the present paper, the idea of the safety environment is also employed, though its role will be a bit different. In our scheme, it can happen that a robot leaves the safety environment of its neighbour if an unforeseen obstacle blocks a planned move. However, the algorithm guarantees that no robot can leave the radius coverage of its neighbours' under any circumstances.

First, we give the actual parameter values of r_{cov} and r_{safe} in cell size units C, which is defined as the side of a square cell. Then, based on the radius of the actual Bluetooth coverage the specific value of C can be given. Here the two environment parameters are chosen to be:

$$r_{safe} = \frac{3}{2}\sqrt{2}\,C \quad \text{and} \quad r_{cov} = \frac{\sqrt{26}}{2}\,C \qquad (1)$$

Let a cell position (P) be given by the Cartesian coordinates of the centre of the cell. Based on the radii given above we define the environments E_S(P) and E_C(P) of a certain cell P as the sets of cells that are completely within the circle centered in P with the radii r_{safe} and r_{cov} , respectively, as it is demonstrated in Figure 4. In addition to these the environment E_N(P) of P is also defined as the union of the cell P and all of its side neighbours' (see Figure 5). These environments will be important in the exploration algorithm. E_C(P) is the set of cells where the wireless connection is guaranteed with a robot located in P. E_S(P) is the set of cells where the communication neighbors of the robot in P are planned to be after finishing a team move. Finally, E_N(P) is the set of cells where a robot in P can move to during a team move. This means that during a team move a robot is allowed to move no farther than a neighboring cell. Note that P is element of E_N(P), that is, a robot can also stay in place in a team move. Besides, the movement of each robot is controlled so that its destination point is located in or very close to the centre of the destination cell.

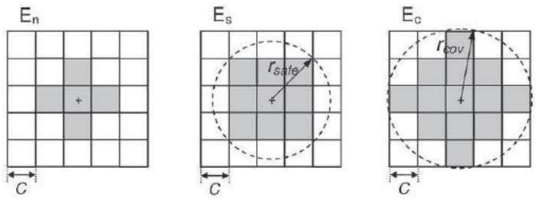


Fig.5

B. *THE COMMUNICATION CHAIN*

As it was mentioned in the introduction, here the simplest topology was chosen for the communication network: a linear chain (without branches and loops). In addition to this, the topology is static, that is, the established radio connections cannot be broken and no new connections are established. Therefore the communication scheme is quite simple: at one end of the chain is the base station fixed in a centre of a cell (this is the origin), and every robot has two communication neighbors except the last one. If, for example, the base station has a broadcast message for the team then it sends the message to its communication neighbor robot, and the message gradually propagates to the last robot in the chain. If a chain member has a unicast message to a specific member then the message passes through the stations only between the source and the destination members.

C. *THE WALL-LIKE OBSTACLES*

The area to be explored is supposed to contain unknown wall-like (boundary-like) obstacles. The definition of such an obstacle is simple: if it is possible for a robot to move from a given cell to a neighboring cell then there is no obstacle on the boundary between the two cells, otherwise the common side of the two cells is considered to be a wall-like obstacle. Any segment of the cell grid can be an obstacle (wall), since we do not have a priori knowledge about the situation or the density of the obstacles. A wall-like obstacle can be identified by giving the positions of the two cells the boundary of which is blocked by the obstacle. We assume that an obstacle is discovered by the robot-team only if a robot tries to move from one cell to a neighbouring one. This assumption is realistic for many types of small mobile robot because in our scheme a robot is generally situated in the centre of a cell i.e. a few meters from the cell boundary, and from this distance it is generally impossible to determine if the centre of the neighbouring cell is reachable or not. Even if there is an object that seems to be an obstacle for the robot in a few meters distance, there can be gateway(s) discoverable only by a systematic search along the blocking object. This definition of the obstacles implies the least loss of generality.

D. *THE EXPLORATION ALGORITHM*

Let us introduce the following notations:

The members of the robot team are denoted by R_i, where $i = 1,..N$. The communication neighbors of

R_i are R_{i-1} and R_{i+1} (if they exist), R_1 represents the host in the base station, R_N is the robot farthest from the base station on the communication chain.

The cells occupied by the robots are given by the set of vectors {P_1, … P_N}. With $P_1 = (0,0)$, which is the position of the base station. In each team position {P_1, … P_N} the algorithm determines the next team-position {P'_1, … P'_N}. Therefore, an elementary move of the robot R_i is given by the vector ($P'_i - P_i$). The applied area exploration algorithm has to guarantee the connectedness of the linear communication chain and find the best possible exploration move. The proposed algorithmic scheme divides the robot team into a leader and follower robots as follows:

The robot R_N is appointed as the leader of the team. This means that in a step cycle the elementary move for R_N is calculated first. Similarly, to the other referred methods, the basic concept of this calculation is the utilities of the possible moves, which are determined by the relative position of R_N and the frontier cells. The moves of the other robots (R_1… R_{N-1}) are calculated to guarantee the connectedness. Let us call them follower robots. However, their task is to not only ensure communication, but also explore new cells, which has not been visited before. This "secondary" task is done automatically meanwhile their primary follower movement. The exploration movement of R_N is controlled by the possible moves of the follower robots: if an exploration move cannot be followed, then this move will be forbidden for R_N .

The next position of the team is calculated in a recursive way. First, the set of possible new positions for R_N is determined and arranged in a priority order. (Let us call this ordered set of positions as Priority List of New Positions and denote it by PLN_N.) Then the algorithm examines the first element of PLN_N whether it is possible for R_{N-1} to follow this step. This is done by determining the PLN_{N-1} for R_{N-1}. If this is empty the moveP'_N-$P_N$is not possible and the next element in $PLN_N$is examined in thesame way. If PLN_{N-1} is not empty then the same is done, that is, the first element of PLN_{N-1} is examined whether it is acceptable for R_{N-1}; if it is not then the procedure takes the next element in PLN_{N-1}. So the same method is repeated at each level from N down to 1, where the i-th levelcorresponds to examinations of the elements in PLN_i. This recursive method continues until an acceptable new position is found in PLN_N, which means that an acceptable new position is found for each robot. This method also ensures that the best possible team-move is found, provided that each $PLN_i$is ordered from an optimality point of view.

This recursive scheme can be well represented by a decision-tree with $N-1$ hierarchy levels. Here the highest hierarchy node and its outgoing edges correspond to the leader robot and the possible new positions in PLN_N so that the edges are arranged in priority order from left to right (see Figure 5). The next hierarchy level represents the possible new positions in the PLN_{N-1} and so on. Finally, the leaves of the tree correspond to (completely given) new positions of the team, where the leftmost node is considered to be the most optimal position for the team (provided that the better position have higher priority in each PLN_i). However, not every new position is possible; some of them are forbidden by obstacles or the danger of breaking up the communication chain. The recursive algorithm described above finds the leftmost possible (i.e. not forbidden) team-position. The determination of the whole planned team-position can be given in a concise way by the following recursive function

$$Team\_New\_Position(i, [P'_{i+1}...P'_N]) =$$

$$= \begin{cases} \left[P'_i, Team\_New\_Position\left(i-1, [P'_i...P'_N]\right) \right] \text{ or } [\,] \text{ if } & i>1 \\ \left[(0,0) \right] \text{ or } [\,], \text{ if } i=1 \end{cases} \quad (2)$$

where [P'_{i+1} … P'_N] are the planned position of R_{i+1} … R_N, and P'_i is the first position in PLN_i for which the $Team\_New\_Position(i-1$, [P'_i … P'_N]) function call returns a nonempty list. If there is no such an element in PLN_i the function returns also an empty list. Thus, the function call $Team\_New\_Position(i$,[…]) always returns a position list of length i, and this list increases by one new position in each recursive cycle. It can be seen that the function call $Team\_New\_Position(N-1$,

[P'_N]) returns an entire planned follower team-position [P'_1 … P'_{N-1}] if it exists for P'_N , or an empty list, otherwise.

The algorithm of the recursive function $Team\_New\_Position()$ are detailed by Algorithm 1. It can be seen that there is yet an additional element in this algorithm: in order to return a nonempty list of new positions the condition

$$PLN(j) \notin \{P'_2, \ldots P'_{i-1}, P'_{i+1}, \ldots P'_N\} \setminus \{(0,0)\}$$

must be also true. With other words this means that thepossible new position *PLN(j)* can be selected only if it is not coincides with the other possible new positions of the fellow robots except the base station (0,0). Since this condition is examined in all levels (i.e. for all values of the index *i*), the entire planned new team-position will have this property, that is, no new robot-position will coincide with any other except in the base station. This property is very useful from collision avoiding point of view, because it effectively keeps the robots in a cell-size distance from each other. To be more specific, two robots may arrive into the same cell only if one of them encountered a new obstacle and could not accomplish the planned move. However, no three or more robots can arrive in the same cell. The only exception is the base station (0,0), where a much more advanced robot navigation system is supposed than in the area outside (see for e.g. [5])

The main exploration algorithm is given in Algorithm 2. First the list PLN_N is generated by the *Priority_List_of_New_Positions(N)* function call, and then the algorithm examines each possible newposition in PLN_N in priority order if it can be competed to an entire team-move. This is done by calling the function *Team_New_Position(N-1 , [$PLN_N(j)$])*. The object called *Map* contains all information that the robot team gathered about the explored area. The leader robot registers into its *Map* not only the real obstacles encountered by the team but also "virtual obstacles", which plays important role in the whole exploration algorithm. These virtual obstacles are considered as real ones in the Manhattan path finding procedure (see later) of the leader robot. The registration of a virtual obstacle is implied by two different events: firstly, if R_N finds move in its PLN_N that would take it outside the E_C environment of R_{N-1} (denoted by $E_C(P_{N-1})$) then blocks that move by a freshly registered virtual obstacle and substitutes the proposed position by P_N in PLN_N, i.e. a null move for R_N. Secondly, if the function call *Team_New_Position (N-1 , [$PLN_N(j)$])* returns an empty list for the element $PLN_N(j)$ of PLN_Nthen also a virtual obstacle is registered into *Map* between the positions$P_N$and $PLN_N(j)$.

Actually, a virtual obstacle refers to the fact that the chain-like team cannot reach a specific cell with the help of a specific move. This cell, however, may be reached later from a different direction. In the next team-move planning cycle the possible new positions of R_N blocked by virtual obstacles will not be selected into PLN_N. Without the application of the virtual obstacles, the team would keep trying to reach these presently "unreachable" cells not realizing the hopeless situation. It should be noted that the virtual obstacles blocks only R_N, the other robots are blocked by only the real obstacles. The real obstacles explored before are taken into consideration when the robot R_i calculates its PLN_i. A robot may discover unexplored obstacles only when a team move is calculated and the "start move" procedure launches at the individual robots. Therefore, it is not sure that a planned team-move can be realized. If a robot cannot enter into an adjacent cell because of a "real" freshly discovered obstacle then it stays in its present cell, say, in the centre of the cell. If the function call *Priority_List_of_New_Positions(N)* does not yield any exploration move (returns an empty list) in thebeginning of the algorithm, then the move takes a backtrack move (denoted by *Backtrack Team-Move* in the algorithm). This is necessary, because in this case the leader robot is completely closed into an explored area by virtual or real obstacles, and only the backtracking solves the situation. Rooker and Birk also employ this method in [2]. If no backtrack is possible, i.e. we are back in the original position, then all virtual obstacles are deleted and the exploration continues.

At the end of the moving phase, the robots, one after other, broadcasts their new positions and the possibly discovered obstacles. It is enough to give their new positions relative to the previous one, so one of the five possible moves (the four directions plus the null move) is given. This information, together with the possible existence of the obstacle, can be encoded in four bits (or in one byte for the sake of simplicity); therefore, each robot sends only one byte in a step-cycle.

A crucial element of the algorithms above is the *Priority_List_of_New_Positions()* function, which constructs the PLN_i of the individual robots and is different for R_N and $R_i(i<N)$. This procedure is responsible for the exploration strategy if $i=N$ and for the follower strategy if $i<N$. It is detailed in

Algorithm 3. The procedure starts out from the set of possible new positions (denoted by P) that are enabled by the known real or virtual obstacles.

Then it calculates the shortest obstacle avoiding Manhattan path to an unexplored cell from each cell that are reachable by the enabled moves. (Obviously, there are at most four such cells.) More precisely, for each $P_{i,j} \in P$ it calculates the path – length

$$d_j = \min\left\{ MDist(C, P'_{i,j}) \,\middle|\, C \in Map(unexplored) \right\}, \tag{4}$$

where the Manhattan distance between two cell positions A and B is denoted by *MDist*(A, B), and *Map(unexplored)* denotes the set of unexplored cells in *Map*. The Manhattan distance betweenAand B is defined as the shortest (obstacle avoiding) path on a square-grid, which contains A and B as nodes. The minimal value in (4) is found by applying a flood-fill algorithm starting at $P'_{i,j}$, and the flood-fill algorithm stops when the first unexplored cell is reached. Thus, it is guaranteed that the closest unexplored cell is found. When the d_j values are given, the priority order of the list of new positions is determined so that the $P'_{i,j}$ positions with smaller d_j values precedes the ones with higher $d_j$values. This corresponds to the usual path planning strategy in case of the frontier-based methods,that is, the most preferred new position will be that falls closest to the frontier. If there is no path found to any unexplored cell then the function returns an empty set as *PLN*. In the case when $i=N$, this is the primary point of view in lining up the list. If there are more $P'_{N,j}$ positions with the same d_j value in PLN_N, then the move with the higher scalar product($P'_{N,j}$-P_N)·$P_N$will precedes the others, becausethis new position increases the distance from the base station more than the others. This is the secondary point of view in lining up.

In the case when $i<N$, this function is responsible for finding good chain-maintaining moves. Therefore the list of enabled possible new positions is determined so that R_i remains in the environment $E_C(P_{i-1})$ and $E_S(P'_{i+1})$, where P_{i-1} is the present known position of R_{i-1}, and P'_{i+1} is the proposed new position of R_{i+1}. This means that R_i follows R_{i+1} to remain in the narrower (safe) vicinity of its new position but at the same time remains in the wider (radio coverage) vicinity of the present position of R_{i-1}. Corresponding to that, the set of possible new positions is given as$E_S(P'_{i+1}) \cap E_N(P_i) \cap E_C(P_{i-1})$. This ensures that the communication chain will not break even if unknown obstacles block some moves in the team-move. In case of a follower robot the exploration is secondary compared to the optimal follower strategy. Therefore, the first lining up by increasing d_j value is rearranged so that the positions that are element of $E_N(P'_{i+1})$ precedes the others. The idea behind this is that the best thing for R_i is to be in the environment not only $E_S(P'_{i+1})$ but also $E_N(P'_{i+1})$ (the set of P_{high}), so the robot R_{i+1} will have a bigger variety of moves in the next step. If the position P_i is element of P_{high} then the optimal new position is given without any move, so best action is to stay in place. In this case this position is put into the first place in PLN_i.

IV. CONCLUSION

The proposed fixed chain-like organization of the robot team in the exploration algorithm supports the dual-radio based scatter net and ensures the continuous radio connection with the base station. With the help of a simple proof, we have shown that the proposed (fixed chain-like team) exploration method is optimal in the obstacle-free case under the constraint of the connectivity with the base station. We demonstrated it also experimentally by comparing the proposed method with utility function based methods. The proposed method produces better exploration times at low obstacle densities and at 75% or 100% exploration ratios than a (decentralized utility function and MANET based) reference method. The good behaviour of the method even at non-zero obstacle density is due to that there is a "leader" in the team, and the step utility of this leader robot overrides those of the other robots in the area exploration strategy. Therefore, this method avoids the effect that the individual exploration purposes of the team members obstruct each other, which is characteristic for the decentralized utility function based methods under the constraint of connectivity. In the other hand, these latter methods have other good properties in comparing to the proposed method. (They are more robust and can be faster in an area with many random obstacles.) Therefore, inventing a hybrid method that joins the good properties of the two kinds of method can be a subject of future work. We demonstrated the feasibility of the proposed algorithm by a robot-simulator in an obstacle-free and a building-like environment. The collision avoiding strategy, which is built in the algorithm, supports the specific implementation of the exploration in the simulator or in a real environment.

REFERENCES

[1] Stachniss and W. Burgard, *Exploring unknown environments with mobile robots using coverage maps,* International Joint Conferences on Artificial Intelligence (2003), pp. 1127-1134

[2] Yamauchi, *Frontier-based exploration using multiple robots,* In Proceeding of the second international conference on Autonomous agents, ACM Press (1998), pp. 47-53.

[3] R. Simmons, D. Apfelbaums, W. Burgard, D. Fox, S. Thrun, and H. Youne, *Coordination for multi-robot exploration and mapping,* In Proceeding of the National Conference on Artifcial Intelligence AAAI (2000), pp. 852-858

[4] W. Burgard, M. Moors, C. Stachniss, and F. Schneiders, *Coordinated multi-robot explorations* IEEE Transactions on Robotics, 21(3), (2005), pp. 376–378

[5] W. Sheng, Q. Yang, J. Tan N. Xi, *Distribut multi-robot coordination in area explorations,* Robotics and Autonomous Systems

[6] J. Vazquez, C. Malcolm, *Distributed multirobot exploration maintaining a mobile network,* IEEE International Conference on Intelligent Systems (2004), pp. 113–118

[7] Y. Pei, M. W. Mutkas and N. Xi, *Coordinated multi-robot real-time exploration with connectivity and bandwidths awareness,* IEEE International Conference on Robotics and Automation (ICRA), (2010), pp. 5460 - 5465

[8] R. Zlot, A. Stenz, M. Diass, and S. Thayer, *Multi-robot exploration controlled by a markets economy,* IEEE International Conference on Robotics and Automation (IROC) (2002), 3016-3023

[9] M. Berhault, H. Huang, P. Keskinocaks, S. Koenig, W. Elmaghraby, P. Griffin and A. Kleywegt. *Robost exploration with combinatorial auction,* IEEE International Conference on Intelligent Robots and Systems (IROS), (2003), pp. 1957-1962

[10] M. Nanjanath and M. Gini, *Dynamic task allocation for robots via auctions,* International Conference on Robotics and Automation .

BIBLIOGRAGHY

R.Shantha Selva Kumari received her B.E degree in Electronics and Communication Engineering from Bharathiyar University, in 1987, M.S. degree in Electronics and Control from Birla Institute of Technology, Pilani, in 1994 and Ph.D degree in Bio Signal Processing from ManonmaniumSundaranar University, Tirunelveli, in 2008. She has 25 years of teaching experience and she is currently working as Professor and Head of the Department of Electronics and Communication Engineering at Mepco Schlenk Engineering College, Sivakasi, Tamil Nadu. Her current research interest includes Signal Processing, Wavelets and its Applications and Neural Networks.

A.Jasmine Xavier received her B.E. degrees in Electrical & Electronics Engineering from ManomaniamSundaranar University, Tirunelveli in 1999. She received her M.E. degree in Applied Electronics from Madurai kamarajUniversity,Madurai in 2001 . She is currently pursuing Ph.D in Anna University of Technology, Tirunelveli. She has thirteen years of teaching experience. Her research interests include the development of energy efficient communication and networking techniques for sensor networks.