Distributed task allocation of mobile robotic sensor networks with guaranteed connectivity

Zhenqiang Mi, Ruochen Yu, Xiangtian Yi and Yang Yang
School of Computer and Communication Engineering, University of Science and Technology Beijing
Beijing, 100083, China
[e-mail: mizq@ustb.edu.cn, yyang@ustb.edu.cn]
*Corresponding author: Zhenqiang Mi

Received August 1, 2014; accepted November 1, 2014; published December 31, 2014

Abstract

Robotic sensor network (RSN) contains mobile sensors and robots providing feasible solution for many multi-agent applications. One of the most critical issues in RSN and its application is how to effectively assign tasks. This paper presents a novel connectivity preserving hybrid task allocation strategy to answer the question particularly for RSN. Firstly, we model the task allocation in RSN to distinguish the discovering and allocating processes. Secondly, a fully distributed simple Task-oriented Unoccupied Neighbor Algorithm, named TUNA, is developed to allocate tasks with only partial view of the network topology. A connectivity controller is finally developed and integrated into the strategy to guarantee the global connectivity of entire RSN, which is critical to most RSN applications. The correctness, efficiency and scalability of TUNA are proved with both theoretical analysis and experimental simulations. The evaluation results show that TUNA can effectively assign tasks to mobile robots with the requirements of only a few messages and small movements of mobile agents.

Keywords: robotic sensor networks; connectivity; task allocation; movement control; ad hoc networks

A preliminary version of this paper appeared in IEEE ICC 2014, June 10-14, Sydney, Australia. This version includes more detailed descriptions of TUNA, theoretical analysis of the message complexity and extensive simulation studies. This work was supported by the National Science Foundation of China [Grant No. 61272432, 61202508 and 61472033] and Fundamental Research Funds for the Central Universities [FRF-TP-14-045A2]

http://dx.doi.org/10.3837/tiis.2014.12.008

1. Introduction

Recent years have witnessed rapidly growing demands of wireless sensor networks (WSN). A properly configured network of sensor devices can perform series of tasks, such as environmental monitoring, planetary exploration, disaster management, etc,. Typically, WSN consists of some stationary sensor nodes that are distributed initially in a fixed formation, and could be manually reconfigured according to updated requirements. Nevertheless, in certain scenarios where the environment is dangerous and extreme, manually relocating the sensors is sometimes inapplicable, if not totally impossible. One of the feasible solutions to this issue is to enable the self-organized movement of the wireless sensors, letting them relocate automatically, which is typically denoted as mobile robotic sensor. Robotic Sensor Network (RSN) is normally composed of functionalized robots and mobile sensors, and the formation of the networked agents can be easily and automatically changed to fit the missions dynamically. The advantage of RSN holds the potential to revolutionize the application of WSN by enabling the users to collect data and perform tasks across complex and expending environment [1].

One of the critical issues to design the RSN is to find an efficient way to assign the decomposed tasks to the appropriate robots/sensors, i.e., task allocation. For instance, a petrochemical plant is jeopardized due to some natural hazards such as a major earthquake or typhoon, as shown in **Fig. 1**. Upon the disaster, a few buildings and oil pipeline are partially destroyed, and fires are triggered around the area. Shortly after the accident, a network of robots and mobile sensors is deployed into the scene to execute the missions of firefighting and maintenance. The RSN consists of firefighter robot, maintainer robot, surveillance sensor and flame sensor. The sensors will monitor the plant and locate the spots of interests, i.e., tasks. Then, the robot will receive the task information from the neighboring sensors and allocate the specific one to execute.

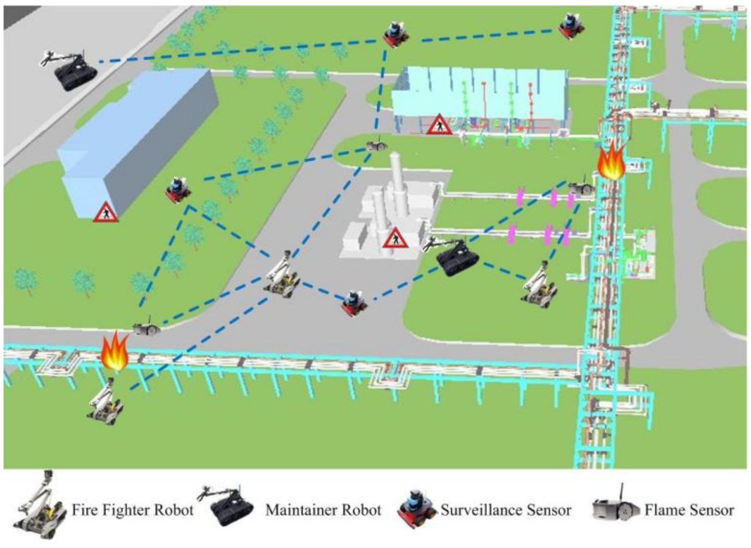

Fig. 1. Simulated task allocation scenario with RSN

The aforementioned scenario raises the important problem of task allocation, i.e., how to find the best available candidate to execute one mission. To date, task allocation problem in Multi-robot Systems (MRS) has been extensively investigated. Please refer [2] for a comprehensive review. Compared to centralized algorithms for task allocation in MRS, decentralized control and task assignment is rather a natural choice, considering the distributed characteristic of the multi-agent networks. The most effective methods for distributed task allocation are threshold- and market-based algorithms. In threshold-based methods, a robot/sensor accepts a task once its "tolerance" surpasses some threshold or abandons it otherwise [3], [4]. The "tolerance" can be continuously adjusted through self-reinforcement [4] to enhance the robustness of the algorithm and prevent agents from occupying a task that they cannot accomplish and ignoring others' needs [5]. Market-based methods let the agent to discover any tasks, and broadcast the messages to the team. Team members respond to the task with their proper biddings [6]. Then, the coordinator (or manager) accumulates the biddings and assigns the tasks to the most optimal agent according to some criteria. This auction mechanism requires negotiation across the entire team, thus that a stable network is often prerequisite [7]. Extensive research have been conducted to optimize the market-based methods by reducing the computing and communication cost, adding time limit, utilizing combinational auction algorithms, etc,.

The existing works can effectively assign the tasks to some appropriate agents. However, they often assume that each robot/sensor is capable of discovering and locating all tasks. Such a conjecture is not practical in most application, where most tasks require specific sensors to discover. Therefore, developing the task allocation algorithms for the specific RSN, and separating the discovery and allocation process are of great importance. Furthermore, a major problem that is often neglected in the existing work is that, the underlying network of the multi-agent system is assumed to be always connected. We argue that, due to the rapid movement of the mobile devices, the network topology will be changed dynamically. Consequently, connectivity of the network will be jeopardized if not properly handled [8]. Connectivity of the underlying network is critical for the RSN to successfully execute missions, especially in most task allocation scenario where inter-agent communication is necessary, e.g., market-based methods that require negotiations.

To deal with these issues, a connectivity preserving task allocation strategy is proposed in this paper. We first model the task allocation in RSN to distinguish the discovering and allocating processes. Then, a simple Task-oriented Unoccupied Neighbor Algorithm (TUNA) is developed to allocate tasks with guaranteed connectivity of the underlying network. The algorithm is fully distributed and requires only partial view of the network topology, so that information exchange within the network can be minimized. Finally, a movement controller is designed to coordinate the sensors and robots to their designated locations to fulfill the objectives of task discovery and task execution. It is worthwhile to notice that, the proposed strategy aims to reach a compromise between network-wide connectivity and fast allocation of tasks, so that the efficiency of the task allocation may not surpass some of the existing methods. Nevertheless, we do guarantee a connected underlying network all time, which has been verified in both theoretical analysis and experimental simulations.

The reminder of this paper is organized as follows: Section 2 provides related works in task allocation and connectivity control techniques. The main problems are formulated in Section 3. Section 4 proposed our main strategy of task discovery and task allocation for RSN. In Section 5, the proposed method is then integrated with a potential based connectivity controller to maintain the connectivity of the RSN. Simulation studies are provided in Section 6. And the paper is finally concluded in Section 7.

2. Related Works

2.1 Distributed Task Allocation Techniques

Task allocation technology has been extensively investigated in multi-agent systems [9]. Take the dynamic environment and the mobility of the mobile nodes into consideration, distributed scheme is a natural choice for most applications. Furthermore, in most multi-agent applications, communication among mobile node is often required to achieve cooperation and coordination goals in some missions [10]. In distributed task allocation, tasks are often referred to as spatial locations or resources. A variety of online [11], [12] and off-line [13] approaches have been proposed: online approaches handle variations of the task environment, while off-line approaches assume that the tasks in the scenario keep constant. In [14], a negotiation scheme is developed to coordinate a team of UAVs to locate and assign tasks to specific team members though inter-neighboring communication. A Genetic Algorithm (GA) based task assignment approach is proposed to dynamically assign simultaneous tasks to team of heterogeneous UAVs carrying designated sensors and weapons [15].

To date, the most successful methods in task allocation are Market-based approaches [16]. Market-based approaches assign tasks through negotiation [17] and auction [18] among team members, and aiming at maximum the profit of each mobile node while minimize its cost, based on the strategy of its bid. Generally, agents send bids to a coordinator to allocate a specific task, and the coordinator will evaluate the bids and choose the winner with highest bid [19]. The approach could be either centralized [20] or distributed [21], depending on the selection of coordinator. Our approach is similar to that of auction-based task allocation. Considering the unique characteristic of RSN, our proposed algorithms are in a totally distributed manner with minimal inter-agent communications. Most importantly, differ from existing work, the connectivity of the network is preserved during the entire mission.

2.2 Connectivity Preservation

In market-based task allocation of mobile multi-agent systems, connectivity of the network is crucial throughout the lifetime of specific missions in order to meet the coordination requirement. To preserve global connectedness, various control approaches, including graph Laplacian based approach [22], [23], power iteration algorithm [24], artificial potential field [25] and navigation function [26] based method, have been introduced. These approaches aim at maintaining and increasing communication links. Other researches focus on reducing communication links while maintaining global connectedness. Zavlanos et al. [27] proposed a hybrid control system consisting of a market-based control strategy and a potential field based motion controller. The connectivity control system can reduce redundant communication links, as well as preserving connectivity based on local estimation of spanning subgraph. In [8], Mi et al. approaches the same objective, while the proposed approach requires only local information of network structure. Connectivity control approaches are further applied to the problem of connectivity preserving coordination, including flocking, rendezvous and formation control.

3. Problem Formulation

First, let the dynamic graph $\mathcal{G}(t) = (\mathcal{V}_1, \mathcal{V}_2, E(t))$ denotes a mobile network of N sensors and M robots, where $\mathcal{V}_1 = (1, 2 \cdots N)$ denotes the set of vertices indexed by the set of mobile sensors, $\mathcal{V}_2 = (1, 2 \cdots M)$ denotes the set of mobile robots and $E(t) = \{i, j \,|\, f_{ij}(t) \geq \varepsilon, i, j \in (\mathcal{V}_1 \cup \mathcal{V}_2)\}, 0 < \varepsilon < 1$ denotes the time-variant set of communication links. We define $0 \leq f_{ij}(t) \leq 1$ to be a normalized non-negative weighting function symmetric in its arguments, i.e., $f_{ij}(t) = f_{ji}(t)$, and assume that $f_{ij}(t) \neq f_{ik}(t)$, $j \neq k$.

Second, for any tasks in the scenario, we define a task set T_{set} , where $T_{set} = (T_1, T_2 \cdots T_k)$ denotes all task types in the mission. For each sensor node in the network, i.e., $n_i \in \mathcal{V}_1$, a sensing set $T_{si} \in T_{set}$ indicates its capability of sensing the tasks.

A quadruple set for each mobile robot $m_i \in \mathcal{V}_2$ is designed as $R_{set}^i = \{T_{ri}, C_{ki}, E_i, D_i\}$, the detailed description of the parameters in the quadruple set is provided as in **Table 1**:

Table 1. Parameter Description

Parameters	**Description**
T_{ri}	$T_{ri} \in T_{set}$ Indicates the types of tasks that robot m_i can perform. (a robot may perform multiple types of tasks)
C_{ki}	The synthesized cost of m_i to perform the *k*-th task
E_i	The residual energy level of m_i (in percentage)
D_i	Distance from m_i to the candidate task

To introduce the objectives in this work, we first make the following definitions:

Definition 1: *An undirected weighted dynamic graph $\mathcal{G}(t)$ is defined as connected at time t if and only if there is at least one communicative path between any two vertices within it.*

Definition 2: *A sensor can discover a task if and only if their direct distance is within the sensing range r.*

Definition 3: *A robot can respond to the task allocation request if and only if it is in vacancy, i.e., currently no assigned or performed tasks.*

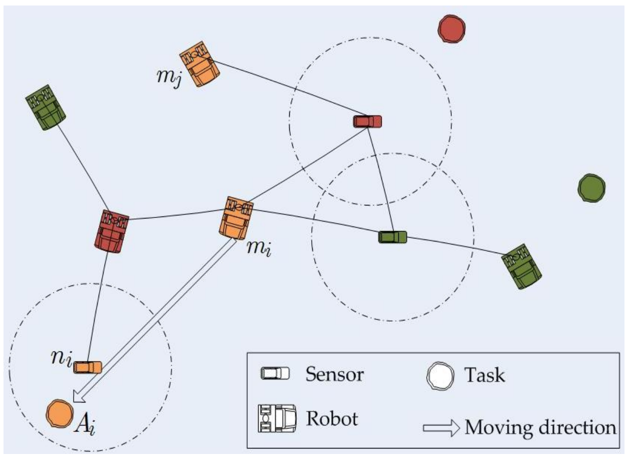

Fig. 2. Problem formulation of task allocation in RSN

The objective of this paper can be described as follows:

In any obstacle-free task allocation scenario as is shown in **Fig. 2**. A connectivity preserving task allocation strategy is designed so that, for any sensed task A_i of type T_j, A_i is assigned to the best available robot m_i in RSN according to the partial view of the network topology. Meanwhile, the connectivity of the network is always maintained while m_i moving towards A_i .

Assuming that robot m_i is allocated with the task A_i , it is easy to observe that the communication links between the robot and the two neighboring sensors will disconnect when m_i is moving towards A_i . Consequently, the connectivity of the network will be destroyed. The way in which we configure the network to avoid this situation is the main contribution of this work.

4. The Task-oriented Unoccupied Neighbor Algorithm

To successfully assign a task to the best available mobile robot based on local information, two main steps are introduced in this section: task discovery and task allocation. Note that the task allocation process is independent of connectivity control, i.e., connectivity is not a parameter for robot in allocating tasks, which makes the proposed strategy more compatible with other task allocation algorithms.

4.1 Task Discovery

According to definition 2, sensors in the RSN are responsible for discovering the tasks. Assume that the sensing range of the sensor devices is r , and the sensors in the RSN continuously move in a certain pattern, e.g., random way point. Further assume the location of a task A_i is ξ_{A_i} , task type of A_i is T_j . A sensor n_i at location ξ_{n_i} will discover A_i when both of the following two requirements are meet: 1) $T_j \in T_{si}$; 2) $|\xi_{A_i} - \xi_{n_i}| < r$.

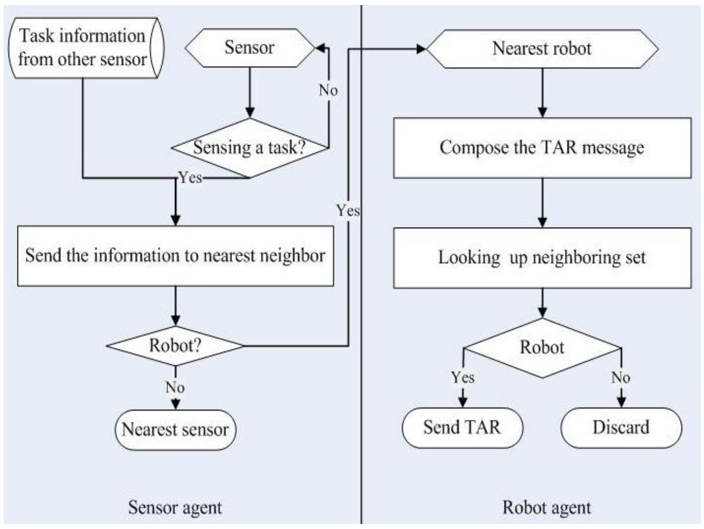

Fig. 3. Flowchart for task allocation process

When n_i discovers the task, it will generate a packet that describes the task's local ID, type and location, and pass it to its nearest neighboring robot. Then, the robot will act as a coordinator in this particular task allocation process. It will add its ID and hop-count *TTL*=1 in the packet to compose them to a Task Allocation Request (TAR) message, and then send the message to all of its one-hop neighboring robots. In case there is no robot existing in the neighbor set of n_i, the packet will be send to its nearest neighboring sensor, which will repeat the process. Note that, in case of a rare situation that two or more neighboring sensors have exactly the same nearest distances with n_i, i.e., a tie between sensors, n_i will wait for the next updated stutas to find the next-hop sensor. Furthermore, if the sensor is a leaf node, i.e., no neighboring robot and sensor besides its parent sensor, it will notify its parent sensor, and the parent sensor will again send the packet to the second nearest neighboring sensor to look for available robot. The complete flowchart of the task discovery process is shown in Fig. 3. It is worthwhile to mention that, the message dissemination process in our strategy is build upon the loop-free Ad hoc On-Demand Distance Vector (AODV) routing protocol, and sensors unicast the packet to the next-hops.

4.2 Task Allocation

Task allocation with consideration of global optimization in distributed manner is always tricky, more often than not, each mobile agent in the network needs the global view to make the best decision. In this case, market-based methods induce a large amount of message exchange throughout the network, which may hold severe impact on network utility, not to mention the unstable topology of the underlying ad hoc networks.

Therefore, a simple Task-oriented Unoccupied Neighbor Algorithm (TUNA) is introduced in this section to solve the task allocation problem locally. TUNA is fully distributed and requires only partial view of the network topology, so that information exchange within the network can be minimized. There are two algorithms in TUNA, as indicated in Table 2. The first algorithm (Algorithm 1, Table 2) is designed for the coordinator to accumulate biddings and assign tasks. The second algorithm (Algorithm 2, Table 2) is designed for other participating robots when they received a TAR.

As described in Algorithm 2, according to definition 3, any robot in vacancy will be able to participate in the task allocation process. When a robot m_i receives TAR, it will compare the type of task in TAR with its own task set, if $T_j \in Tr_i$, the robot will reply a Task Allocation Participation (TAP) message to the source node or reply a Task Reject (TR) message otherwise. A weighted function is introduced to compose TAP:

$$F_{m_i} = -K_1 \times C_{ki} + K_2 \times E_i - K_3 \times D_i \tag{1}$$

where K_1, K_2 and K_3 are all non-negative constants, and $D_i = \left|\xi_{A_i} - \xi_{m_i}\right|$. The constants can be adjusted according to specific application scenarios to achieve different objectives, e.g., energy-aware, or cost-aware.

The detailed execution process for coordinator to assign task is shown in Algorithm 1, Table 2, which is described as following: TAP messages that are composed by the corresponding robots contain the robot ID, source ID, and the results of the weighted function F_m. The coordinator will receive all the TAP and compare all results in F_m (including itself if the type of task included in its task set), as shown in line 8-16 of Algorithm 1. If $F_{m_i} = \max\{F_m\}$, then a Task Confirmation (TC) message will be sent from the controller to m_i to indicate that m_i has successfully allocated the task, as shown in line 22-24 of Algorithm 1. Upon receiving TC message, m_i will move directly towards A_i to execute the task.

Table 2. Pseudo Code for TUNA

Algorithm 1 (Coordinator)

Initialization: a mobile m_c with neighboring set N_c received task information from one of its neighboring sensor, and start looking for qualified candidate.

```
1:  $m_c$ composed TAR with $TTL$=1
2:   for $m_i \in V_2$
3:         if $m_i \in N_c$
4:               Send TAR to $m_i$
5:        otherwise   break
6:        end if
7:   end
```

Find Candidate: m_c received responses from all neighboring robots, and stored the responses in set R_c.

```
8:     if $T_j \in T_{rc}$
9:       put $F_{m_c}$ in set $F_m$
10:  end
11:  for  response $r_i \in R_c$
12:           if $r_i$ =TAP
13:                put $F_{m_i}$ in set $F_m$
14:         else if $r_i$ =TR        drop
15:         end if
16:  end
17:  if $F_m = \varnothing$
18:       $TTL$=$TTL$+1, compose TAR
19:       for $m_i, n_i \in N_c$
20:             Send TAR to all neighbors, end
21:       Go to 11
22:  else if $F_m \neq \varnothing$
23:       $m_i$ =$FindMAX$( $F_m$ ), Send TC to $m_i$
24:  end
```

Algorithm 2 (Mobile robots)

Initialization: a mobile m_i with neighboring set N_i received TAR from one of its neighboring robot

```
1: if $TTL$-1=0
2:      if $T_j \in T_{ri}$
3:            compose TAP and send it to $m_c$
4:      else
5:            compose TR and send it to $m_c$
6:      end if
7: else
8:      for $m_j \in N_i$
9:            send TAR to $m_j$
10:    end
11:end if
```

For scenarios with multiple types of tasks and robots, chances are the coordinator received only TR messages from all its neighboring robots, and the coordinator itself is also unable to perform the task. In this case, the coordinator will gradually add the hop-count of the TAR messages, e.g., *TTL*=2, and looking for all two-hop neighboring robots (line 17-21 of Algorithm 1). Note that, the TAR message will send to all the sensor and robot neighbors of the coordinator when TTL≥2. The hop-count of the message will gradually increase until a qualified robot replies TAP.

Theorem 1: *The worst-case message complexity of TUNA for each task is* $O(n)$, *where* n *is the number of mobile nodes.*

Proof: In the worst case, the topology of the network will be a line formation, where $n-1$ mobile sensors and only one mobile robot are in the network, as is shown in **Fig. 4**. In this case, each of the mobile sensors will send one message to the next hop neighbor to broadcast the task, which is a total number of $n-1$ messages generated. When the mobile robot receives the task information, a TAR with *TTL*=1 will be generated and sent to all of its one-hop neighbors. As none of the neighbors will reply to the task, new TARs with gradually increased TTL will be sent to the network, thus a total number of $n-1$ TAR messages will be generated by TUNA. Adding $n-1$, the total messages complexity in the worst case for TUNA is $2n-2$ which is $O(n)$.

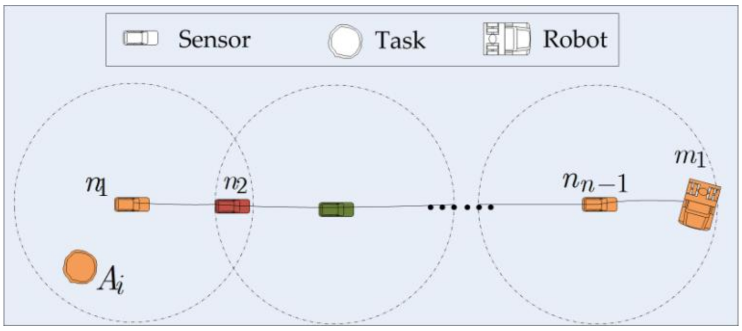

Fig. 4. Worst case topology for message complexity

Note that, in real scenario, the worst case topology is not likely to be formed, and a random network will require much less messages for each task allocation process.

The main advantages of TUNA are three-fold: 1) TUNA uses mobile robots as coordinators in the task allocation process, which could fully utilize the computing capability of robots and avoid unnecessary energy consumption of the sensors; 2) TUNA adopts a gradual expansion technique in searching candidate robot to avoid redundant message exchange in most scenarios; 3) TUNA uses only one coordinator for each task to avoid the possibility of concurrent allocation of tasks.

5. Hybrid Control of Connectivity

To facilitate the mobile robot to move towards the tasks without causing disconnections in the underlying network, a connectivity control method is required. We simply adopt the proactive topology control algorithm LT^{3}C, which we proposed earlier, please refer to [8] for details. In this paper, we will focus on the movement control of the candidate robot while it moves towards the task.

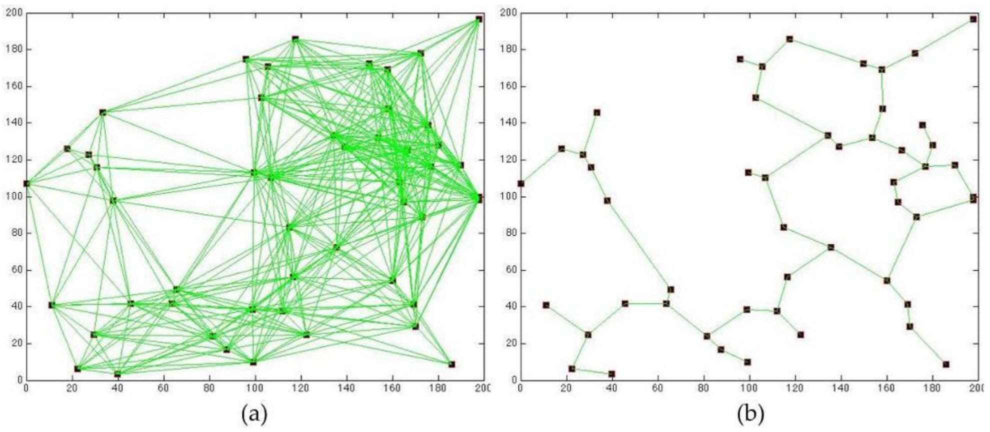

Fig. 5. An intial densed network in (a) turns to a sparse network with no triangle and quadruple topology in (b)

We first give a brief introduction of LT^{3}C. LT^{3}C is a light-weighted connectivity control algorithm that requires only two-hop neighbors' information, so that the message exchange in the network can be reduced. It simply removes all the triangle and quadruple topology of the network, and results in a sparse network structure with only essential communication links, as is shown in **Fig. 5**. With LT^{3}C, mobile robots in the network can be granted with much more freedoms when they move from one location to another, which could considerably improve the performance of task allocation.

With the aid of LT3C, a motion controller is designed to coordinate the robots and mobile sensors in the task allocation and execution process. We first make the following assumptions:

Assumption 1: Mobile sensors will move around the space in some predefined patterns, e.g., RWP. Sensor will stop as long as it senses a task, and will start moving again as soon as the allocation of the task is finished.

Assumption 2: An essential neighbor set N_{m_i} is formed for each robot with the aid of LT3C.

Assumption 3: An essential neighbor set N_{n_i} is formed for each sensor with the aid of LT3C.

Two attractive potential functions should be assigned to each mobile sensor and robot. First, for each two neighboring mobile agents in the RSN, e.g., $i, j \in (\mathcal{V}_1 \cup \mathcal{V}_2)$, an inter-agent attractive potential function $\psi_a(f_{ij}(t))$ is assigned, which is used to maintain the essential communication links. Second, two different potential functions should be assigned to sensor and candidate robot, respectively. 1) For each mobile sensor, a planning potential function $\psi_p(\xi_{n_i}, \xi_{next})$ for each mobile sensor is designed, where ξ_{next} is the location of next step that generated by movement pattern. $\psi_p(\xi_{n_i}, \xi_{next})$ is able to drive the sensors towards the designated point in the scenario according to the adopted movement model. 2) For a robot m_i that has successfully allocated a task A_j, a tracking potential function $\psi_t(\xi_{m_i}, \xi_{A_j})$ is assigned, which is able to drive the mobile robot to the task to execute the mission.

To summarize, the potential function of the entire system is defined as follows:

$$\begin{aligned}\Phi = &\sum_{i=1}^{N}\sum_{j\in Nn_i}\psi a(fij(t)) + \sum_{i=1}^{M}\sum_{j\in Nm_i}\psi a(fij(t)) \\ &+ \sum_{i=1}^{N}\psi p(\xi_{n_i}, \xi next) \\ &+ \sum_{i=1}^{M}\sum_{T_j\in Tm_i}\psi t(\xi_{m_i}, \xi_{A_j})\end{aligned} \tag{2}$$

$\psi p(\xi_{n_i}, \xi next)$ can take the quadratic form

$$\psi p(\xi_{n_i}, \xi next) = \frac{1}{2}Kp\left|\xi_{n_i}(t) - \xi_{next}(t)\right|^2 \tag{3}$$

$\psi t(\xi_{m_i}, \xi_{A_j})$ can take the quadratic form

$$\psi t(\xi_{m_i}, \xi_{A_j}) = \frac{1}{2}Kt\left|\xi_{m_i}(t) - \xi_{A_j}(t)\right|^2 \tag{4}$$

where Kp and Kt are positive constant gain.

Due to the mutual relationship between neighboring agents, the attractive potential functions satisfy:

$$\nabla_{\xi_i}\psi a(fij(t)) = -\nabla_{\xi_j}\psi a(fij(t)) \tag{5}$$

Furthermore, $\psi a(fij(t))$ has the following attributes:

1) For a specific small value σ , where $\sigma > \varepsilon$. If $fij(t) \le \sigma$, $\psi a(fij(t)) < 0$ and $\nabla_{\xi_i}\psi a(fij(t)) < 0$, otherwise $\psi a(fij(t)) = \nabla_{\xi_i}\psi a(fij(t)) = 0$.

2) $\nabla_{\xi_i}\psi a(fij(t)) \to -\infty$ when $fij(t) \to \varepsilon$.

2) $\nabla_{\xi_i}\psi a(fij(t))$ is differentiable.

A distributed controller for each sensor ni is developed as follows:

$$\dot{\xi}_{n_i} = -\left[\sum_{j\in Nn_i}\nabla_{\xi_{n_i}}(\psi a(fij(t)) + \psi p(\xi_{n_i}, \xi next))\right] \tag{6}$$

It is worthwhile to notice that, as indicated in the attributes of attractive potential and (6), the attractive potential will increased infinitely as the link is extended to the point which it "almost" breaks. So sensors may not be able to finally reach $\xi_{next}(t)$ if there is an attractive potential in conflict, but will stay at the local minima of these two potentials until $\xi_{next}(t)$ is updated by the movement pattern.

For each robot mi that has been assigned with a task, the controller is:

$$\dot{\xi}_i = -\left[\sum_{j\in Nm_i}\nabla_{\xi_{m_i}}(\psi a(fij(t)) + \psi t(\xi_{m_i}, \xi_{A_j}))\right] \tag{7}$$

Finally, for all other mobile robots mj in the network, the controller goes:

$$\dot{\xi}_i = -\left[\sum_{k\in Nm_j}\nabla_{\xi_{m_j}}\psi a(fjk(t))\right] \tag{8}$$

6. Experimental Simulations

6.1 Simulation Setup and Evaluation Metrics

In this section, we present a series of simulation studies to evaluate the correctness and efficiency of TUNA. The trajectories of the mobile nodes are calculated and generated by MATLAB, and then they are integrated in to NS-2 for the evaluation of the network performances. The parameters are set as shown in Table 3.We assume the typical sensing ranges vary from 30m-90m without losing generality, and the constant gains are designed with consideration of the nodes' realistic velocity. Furthermore, the specific small value σ is set to 0.2 of the uniformed link quality so that the attractive potentials will only initiated when the inter-node distance is close to the threshold of losing connections, which is set as 0.1. The setups for the simulation scenarios should obey the following principles:

(1) The design of $fij(t)$ and $\psi_a(fij(t))$ are strictly followed the proposed principles in Section 5.

(2) The number of mobile agents is constant in during simulation, but differs among scenarios. And the Ratio of Robot (RR) among all nodes could be varied.

(3) Each mobile sensor and robot randomly chooses two types of tasks in task set for sensing and allocation.

(4) Each simulation runs for 30 times, and the average results are reported.

To evaluate the efficiency of TUNA, we selected the following three metrics:

Allocated Tasks (AT): *due to the limited number of nodes and the connectivity constrain, there is a possibility that some of the tasks cannot be allocated, AT represents the number of tasks has been successfully allocated in the simulations.*

Average Generated Messages (AGM): *AGM refers to the average message generated for each task during the entire task allocation.*

Average Travel Distance (ATD): *ATD refers to the average distances traveled by each mobile robot during the entire task allocation.*

Table 3. Simulation Parameters

Parameters	Description	Value
r	Sensing Range	30m~90m
K_1	Constant gain of the synthesized cost	10
K_2	Constant gain of the residual energy level	6
K_3	Constant gain of the distance with task	1
σ	Specific small value to trigger $\psi_a(fij(t))$	0.2
ε	Threshold of losing connection	0.1
K_p	Constant gain of planning potential	15
K_t	Constant gain of tracking potential	40

6.2 Results and Discussions

In the first scenario, two types of tasks are included. A RSN of 5 mobile sensors and 4 mobile robots are distributed in the scenario of 150m×250m for task allocation, as shown in **Fig. 6**. The sensors are marked as round circle with colors that correspond to its designated sensing tasks. And mobile robots are marked as the same shape as the tasks. We assume that each mobile sensors and robots in this scenario can only perform one type of tasks for simplicity.

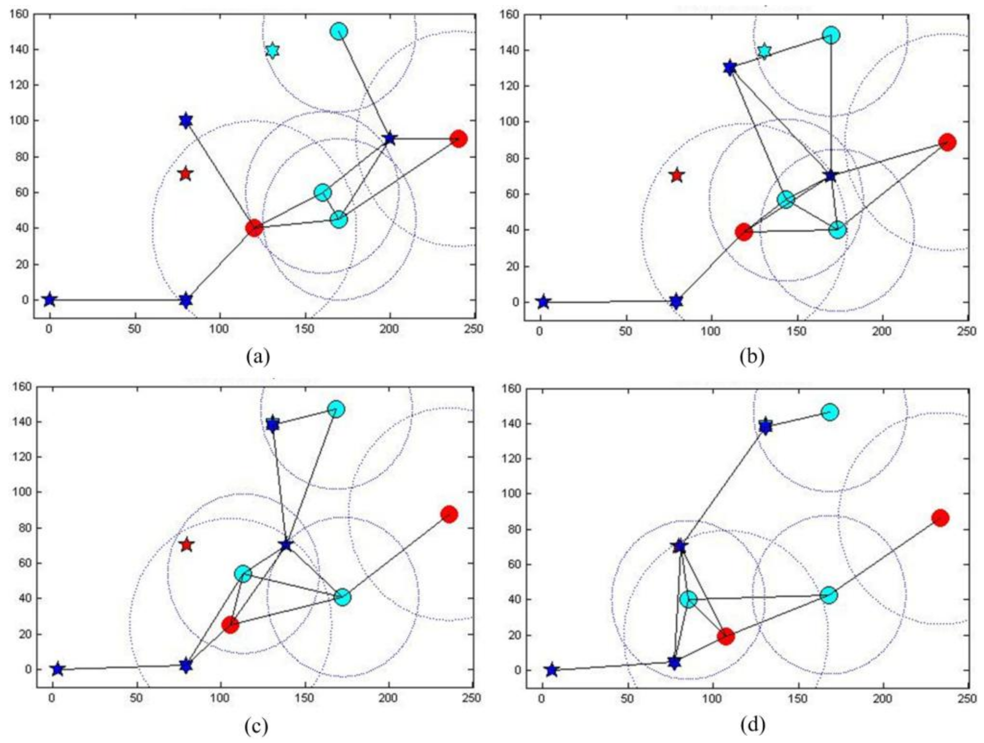

Fig. 6. Test scenario for task allocation, (a) t=0; (b) t=10; (c)t=20; (d)t=50

As is shown in Fig. 6, two sensors initially discovered each task, and initialized the task allocation process with proposed strategy. And two mobile robots are assigned with the tasks, respectively. While mobile robots move toward the tasks and sensors move to designated locations for further discovering other tasks, the connectivity of the entire network are always preserved.

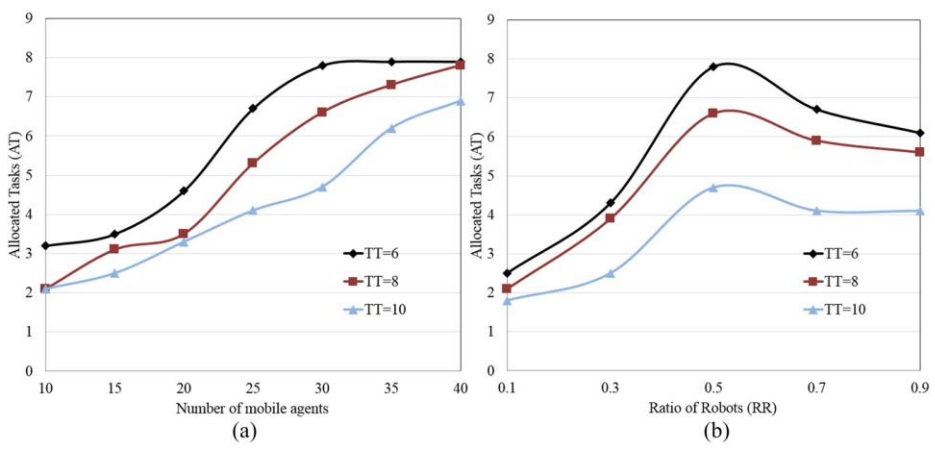

Fig. 7. Performance of task allocation subjected to
(a) different number of mobile agents; (b) different ratio of robots

In the second simulation, we further evaluate the performance of the task allocation strategy by randomly adding Types of Tasks (TT) in T_{set}. The total number of tasks in the scenario is 8, and the simulated area is 600m$\times$600m. And different number of agents and different RR are used respectively to shown the efficiency with respect to network scale and network compositions.

Firstly, we evaluate AT with respect to increased number of mobile nodes from 10 to 40, and RR≈0.5 indicates that there are approximately the same amount of sensors and robots. The results are shown in **Fig. 7(a)**. It is clear that, in a scenario with fixed area, the success of task allocation can be improved by adding more mobile sensors and robots into the RSN. It is straightforward since there are two main reasons that a task cannot be successfully executed: 1) no mobile robot can perform the task in the RSN; 2) the robot that has allocated the task cannot be relocated to it, otherwise the connectivity would be in jeopardy according to the connectivity control method. So that adding more agents in the network could fix the aforementioned problems and consequently improve the successful rate of task allocations. Moreover, with different TT, the task allocation results are varied. Smaller TT shows better performance, this is due to the fact that the possibility of the mobile robots in RSN that can perform all the tasks is great when the number of mobile agents is much more than the number of TT.

Secondly, different RR are adopted to evaluate the impact of network composition on the task allocation efficiency. Interestingly, as shown in **Fig. 7(b)**, the RR has greater impacts on the AT, compared to the ratio of sensors. A possible explanation could be that the smaller number of robots always reduce the possibility of allocating tasks, however, since mobile sensors can move to another location after it discovered a task, and continue to sense others, so that smaller number of sensors will not severely affect the task allocation process.

To evaluate the advantage of TUNA in terms of message complexity and travelling distance, we conduct a third simulation with fixed number of tasks as 6, and RR=0.5. We use varies number of mobile nodes to show the impacts of node density on the performances of TUNA. The results are unfolded in **Fig. 8**.

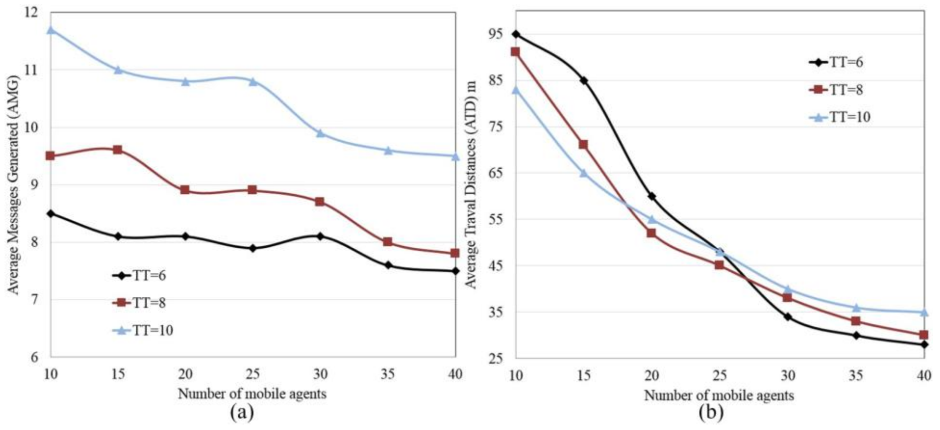

Fig. 8. Performance of task allocation subjected to different number of mobile agents (a) average messages generated; (b) average travel distances (m)

Firstly, as shown in **Fig. 8(a)**, the message complexity of TUNA is generally stable and slightly decreased with the increase of agents. This could be attributed to the increased possibility that a valid robot can be found with a smaller *TTL*, so that the number of TAR messages is reduced. However, with increased types of tasks, TUNA may need more hop-counts to find the unoccupied robot to allocate the task, therefore the AMG are greater with higher TT. Secondly, we evaluate the ATD, which is an important factor to show TUNA's scalability and energy-efficient. As shown in **Fig. 8(b)**, the ATD decrease dramatically with the increased number of mobile agents, because the average robots-to-tasks distances are decreased when the network becomes denser. Moreover, the ATD is not sensitive to the types of tasks, so that TUNA has good scalability when handling task allocation problems in large scale networks.

7. Conclusion and Future Works

Connectivity of the underlying network is perquisite for many applications with RSN, and the problem of how to effectively assign tasks with guaranteed connectivity has not yet been solved. To address the issue, we modeled the task allocation problem in RSN to distinguish the discovering and allocating processes. Then, we proposed a fully distributed task allocation algorithm (TUNA) to allocate tasks with only partial view of the network topology. We integrated TUNA with a distributed connectivity controller to coordinate the movement of the mobile agents with guaranteed connectivity of the entire RSN. Theoretical analysis and simulation studies are provided to show that TUNA is effective in allocating tasks, reducing message exchanges and nodes' traval distances. In the meantime, network-wide connectivity are always preseved. It is worthwhile to mention that the performance of task allocation varies with respect to the change of scenarios, but is acceptable if the number of mobile agents in the network reached a certain level.

Our futher work will focus on the development of effective task allocation strategies of mobile robot and sensor systems in complex envrioment with random obstacles. Meanwhile, we will try to design a more effective topology control strategy to further reduce the ATD in the task allocation and excecution process.

References

[1] M. Dunbabin and L. Marques, "Robots for environmental monitoring: significant advancements and applications," *IEEE Robotics & Automation Magazine*, vol.19, no.1, pp.24-39, March 2012. Article (CrossRef Link).

[2] A. Campbell and A. S. Wu, "Multi-agent role allocation: issues, approaches, and multiple perspectives," *Autonomous Agents and Multi-Agent Systems*, vol. 22, no. 2, pp. 317–355, March 2011. Article (CrossRef Link).

[3] L.E. Parker, "ALLIANCE: an architecture for fault tolerant multirobot cooperation," *IEEE Transactions on Robotics and Automation*, vol.14, no.2, pp. 220-240, April 1998. Article (CrossRef Link).

[4] A. Gage and R. Murphy, "Affective recruitment of distributed heterogeneous agents," in *Proc. of 9th National Conference on Artificial Intelligence*, pp. 14-19, July 2004.

[5] H. Goldingay and J.V. Mourik, "The effect of load on agent-based algorithms for distributed task allocation," *Information Sciences*, vol. 222, pp. 66-80, February 2013. Article (CrossRef Link).

[6] M.B. Dias, R. Zlot, N. Kalra and A. Stentz, "Market-Based Multirobot Coordination: A Survey and Analysis," *Proceedings of the IEEE*, vol. 94, no. 7, pp.1257-1270, July 2006. Article (CrossRef Link).

[7] C.H. Caicedo-Nunez and M. Zefran, "Distributed Task Assignment in Mobile Sensor Networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2485-2489, October 2011. Article (CrossRef Link).

[8] Z. Mi and Y. Yang, "Topology control and coverage enhancement of dynamic networks based on the controllable movement of mobile agents," in *Proc. of 2011 IEEE International Conference on Communication*, pp. 1-5, June 5-9, 2011. Article (CrossRef Link).
[9] A. Campbell and A. S. Wu, "Multi-agent role allocation: issues, approaches, and multiple perspectives," *Autonomous Agents and Multi-Agent Systems*, vol. 22, no. 2. March 2011. Article (CrossRef Link).
[10] D. Tardioli, et. al., "Enforcing network connectivity in robot team missions," *International Journal of Robotics Research*, vol. 29, no. 4, pp. 460-480, April 2010. Article (CrossRef Link).
[11] M. Zavlanos and G. Pappas, "Dynamic assignment in distributed motion planning with local coordination," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 232–242, February 2008. Article (CrossRef Link).
[12] K. Lerman, C. Jones, A. Galstyan, and M. Mataric, "Analysis of dynamic task allocation in multi-robot systems," *International Journal of Robotics Research*, vol.25, no. 3, pp. 225–242, 2006. Article (CrossRef Link).
[13] M. Ji, S. Azuma, and M. Egerstedt, "Role-assignment in multi-agent coordination," *International Journal of Assistive Robotics and Mechatronics*, vol. 7, no. 1, pp. 32–40, March 2006. Article (CrossRef Link).
[14] P. B. Sujit, A. Sinha, and D. Ghose, "Multiple UAV task allocation using negotiation," in *Proc. of fifth international joint conference on Autonomous agents and multiagent systems*, pp. 471–478, May 2006. Article (CrossRef Link).
[15] T. Shima, S. Rasmussen, and A. Sparks. "UAV cooperative multiple task assignments using genetic algorithms," in *Proc. of 2005 IEEE American Control Conference*, pp. 2989-2994, June 8-10, 2005. Article (CrossRef Link).
[16] M. Dias, et al. "Market-based multirobot coordination: A survey and analysis," *Proceedings of the IEEE*. vol. 94, no. 7, pp. 1257-1270, July 2006. Article (CrossRef Link).
[17] G. A. Korsah, A. Stentz, and M. Dias. "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*. vol. 32, no. 12, pp. 1495-1512, 2013. Article (CrossRef Link).
[18] B. P. Gerkey and M. Mataric, "Sold!: Auction methods for multirobot coordination," *IEEE Transactions on Robotics and Automation*. vol. 18, no. 5, pp. 758-768, October 2002. Article (CrossRef Link).
[19] H. L. Choi, L. Brunet and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics*. vol. 25, no. 4, pp. 912-926, August 2009. Article (CrossRef Link).
[20] K. Zhang, E. Collins Jr, and D. Shi. "Centralized and distributed task allocation in multi-robot teams via a stochastic clustering auction," *ACM Transactions on Autonomous and Adaptive Systems*. vol. 7, no. 2, July 2009. Article (CrossRef Link).
[21] A. Brutschy, et al. "Self-organized task allocation to sequentially interdependent tasks in swarm robotics," *Autonomous Agents and Multi-Agent Systems*. vol. 28, no. 1, pp. 101-125, January 2014. Article (CrossRef Link).
[22] M. M. Zavlanos and G. J. Pappas, "Potential fields for maintaining connectivity of mobile networks," *IEEE Transactions on Robotics*, vol. 23, no.4, pp 812-816, Aug. 2007. Article (CrossRef Link).
[23] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian," *IEEE Transactions on Automatic Control*, vol. 51, no. 1, pp. 116-120, January 2006. Article (CrossRef Link).
[24] M. C. De Gennaro and A. Jadbabaie, "Decentralized control of connectivity for multiagent systems," in *Proc. of the 45th IEEE Conf. Decision Control*, San Diego, pp 3628-3633, December 13-15, 2006. Article (CrossRef Link).
[25] A. Ajorlou, A. Momeni and A. G. Aghdam, "Connectivity preservation in a network of single integrator agents," in *Proc. of the 48th IEEE Conf. Decision Control*, pp. 7061-7067, December 15-18, 2009. Article (CrossRef Link).
[26] D. V. Dimarogonas and K. H. Johansson, "Decentralized connectivity maintenance in mobile

networks with bounded inputs," in *Proc. of 2008 IEEE int. conf. Robotics and Automation*, pp. 1507-1512, May 19-23, 2008. Article (CrossRef Link).

[27] M. M. Zavlanos and G. J. Pappas, "Distributed connectivity control of mobile networks," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1416-1428, December 2008. Article (CrossRef Link).

Zhenqiang Mi received the BS degree in Automation and the PhD degree in Communication and Information Systems from University of Science and Technology Beijing, China, in 2006 and 2011, respectively. Currently, he is an assistant professor in School of Computer and Communication Engineering, University of Science and Technology Beijing. His technical interests include multi-robot systems, ad hoc and sensor networks and cloud computing. He has published more than 20 technical papers in refereed conference proceedings and journals. He is a member of the IEEE.

Ruochen Yu is with University of Science and Technology Beijing since 2011. She is currently an undergraduate student in the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. Her research interests include mobile multi-agent networks, mobile sensor systems and multi-robot systems.

Xiangtian Yi is with University of Science and Technology Beijing since 2011. He is an undergraduate student in the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. Currently, He is with the School of information technology and electrical engineering, The University of Queensland as a visiting student. His research interests include cloud computing and mobile wireless sensor and actor networks and multi-robot systems.

Yang Yang received the BS degree in University of Science and Technology Beijing, China, in 1982. He received the PhD degree from University of Science and Technology in Lille, France, in 1988. Currently, he is a professor in the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. His research interests include cloud computing, intelligence control and multimedia communications. He has published more than 100 technical papers in refereed conference proceedings and journals.